웹사이트의 효율적인 구조 관리와 평가 방법†
(A Method for Efficient Structure Management and Evaluation of Website)

유대승*, 엄정섭**, 이명재***
(Dae-Sung Yoo, Jeong-Seob Eom, Myeong-Jae Yi)

요 약 웹의 급속한 성장과 함께 기존의 시스템들이 웹을 기반으로 통합되며, 다양한 시스템들이 개발되고 있다. 일반적인 어플리케이션과는 달리 웹 어플리케이션들은 다양한 기술의 접목으로 개발된다는 점과 본래의 복잡성으로 인해 개발과 관리에 있어 어려움은 더욱 증대된다. 또한 급변하는 비즈니스 환경과 사용자들의 요구사항에 순응하기 위해서는 지속적인 진화가 요구된다. 본 논문에서는 웹 어플리케이션의 구조 정보인 링크 정보를 추출하고, 웹사이트에 대한 유용한 정보를 담고 있는 로그 파일을 분석하여 웹 어플리케이션의 보다 효율적인 개발과 유지보수에 활용하는 방법을 제시한다. 본 논문에서 추출한 정보들은 웹 어플리케이션 테스팅을 위한 기초적인 정보가 될 수 있으며 추출한 정보들을 웹 테스팅에 적용하는 방법을 설명한다. 그리고 링크 정보 추출과 웹 로그 분석을 수행하기 위해 개발된 시스템에 대해 기술한다.

Abstract With the rapid growth of WWW, the existing systems are integrated into web and various web-based systems are developed. Unlike the general applications, web applications are developed by combining the various technologies and have their own complexities. So, we have much difficulties in the development and maintenance of web applications. To accommodate to the rapidly changing business environments and user requirements, the continuos evolution is required. In this paper, we present a method for supporting the effective development and maintenance of web applications. Our method involves the extraction of web application's structure information and analyzes web log files containing the useful information about web site. We also describe a web testing method using the extracted information and our system developed for extracting hyperlink information and analyzing web log.

1. 서 론

웹의 빠른 발전과 함께 비즈니스, 지식 경영, 마케팅, 전자 상거래 등 기존의 다양한 IT 시스템들이 웹을 기반으로 변화되어 가고 있다. 웹을 기반으로 한 통합 개발 과정에서 필연적으로 파생되어지는 복잡함은 개발 및 유지보수를 더욱 어렵게 한다. 또한 비즈니스 환경의 빠른 변화에 따른 짧은 소프트웨어의 생명주기와 진화의 필요성은 기존의 개발방법보다 더 많은 비용과 인력을 요구한다.[1,2]

웹 개발자들은 웹 문서를 만들고 수정하는 과정에서 발생되어진 방대해진 웹 문서들을 관리하는 것에 많은 어려움을 겪고 있다. 웹 문서들은 서로 복잡하게 얽혀져 있기 때문에 웹 문서들에 포함된 링크들이 제대로 연결되어 있는지를 검사하고 전체 웹 문서들을 일관성 있게 수정하는 것은 매우 힘들다. 인터넷과 관련된 기업의 폭발적인 증가와 함께 성공적인 웹사이트 운영을 위해서는 빠르게 변화하는 환경과 사용자들의 요구사항에 순응하며 서비스를 제공하는 시스템을 지속적으로 진화시켜야 한다. 이를 위해서는 시스템 운영 상황을 지속적으로 감시하고 문제를 조기 발견해서 대처해야 한다. 또한 사용자에 대하여 끊임없이 연구하고 조사하여야 한다. 이것은 비용과 노력이 많이 소요되는 일이지만 운영 시스템이 진부화가 되는 것을 방지하기 위해서는 필수적이다. 다행히 웹 어플리케이션의 경우 웹 로그 등을 통해 비교적 쉽게 정보를 얻을 수 있다는 장점이 있다.

본 논문에서는 웹 어플리케이션을 효율적으로 개발, 유지보수, 운영, 진화시키기 위하여 중요한 정보라고 할 수 있는 웹 어플리케이션의 구조 정보와 실제 사용자의 행동 정보를 추출하고 이것을 활용하는 방법을 제시한다. 구조 정보는 링크 추출을 통해 얻을 수 있고 사용자에 대한 행동 정보는 웹서버에 로그 파일 형식으로 기록되는 로그 분석을 통해서 얻을 수 있다. 동적 페이지를 포함한 정확한 링크 추출을 위해서 4단계 추출 과정을 정의한다. 각 단계를 거치면서 좀 더 정확한 링크 추출이 가능하다.

로그 분석에서 가장 중요한 요소 중의 하나는 분석에 사용될

†본 연구는 정보통신부의 "정보통신 우수시범학교 지원사업"의 지원에 의해 이루어졌습니다.
* 울산대학교 컴퓨터정보통신공학부 박사과정
** 울산대학교 컴퓨터정보통신공학부 석사과정
*** 울산대학교 컴퓨터정보통신공학부 교수
유대승 : ooseyds@mail.ulsan.ac.kr

데이터의 질이다. 로그 파일에 저장된 순수 데이터는 프록시 서버나 쿠키 등의 사용으로 인하여 근본적으로 부정확한 데이터이므로 정확한 분석을 위해서는 분석 알고리즘을 적용하기 전에 로그의 순수 데이터를 정제하고 보완하는 것이 필요하다. 본 논문에서는 분석 항목을 사용자별 행동 정보와 그룹별 행동 정보로 구분하고 각 항목 하위의 분석 요소들을 추출하기 위한 정제 및 추출 알고리즘에 대하여 기술한다. 그리고 추출한 링크 정보와 행동 정보를 웹 테스팅에 활용하는 방안을 제시한다.

본 논문의 구성은 다음과 같다. 2장에서는 기존의 연구에서 제시된 웹 어플리케이션 분석 방법들에 대하여 살펴본다. 3장과 4장에서는 구조 분석 과정과 로그 분석 과정을 세부적으로 설명한다. 5장에서는 본 논문에서 제시한 방법을 기반으로 개발된 구조 분석과 로그 분석을 위한 시스템을 소개한다. 6장에서는 구조 및 로그 분석 과정에서 추출한 정보를 웹 테스팅에 적용하는 방안을 제시한다. 끝으로 7장에서는 결론과 향후 연구과제에 대해서 기술한다.

2. 웹 어플리케이션의 유지 보수

2.1 웹 어플리케이션의 특징

웹 어플리케이션은 기능적으로 웹을 통해 배포되어지는 소프트웨어 시스템이다. 인터넷과 웹을 이용한다는 장점으로 인해 많은 어플리케이션들은 더 이상 전통적인 클라이언트/서버 모델을 이용하지 않아도 된다. 대신에 새로운 어플리케이션들은 웹 브라우저, 웹 서버 그리고, 어플리케이션 서버와 같은 웹 기술들을 이용하여 개발된다. 웹 브라우저는 일반적인 어플리케이션의 사용자 인터페이스에 해당하고, HTTP(Hyper Text Transfer Protocol)와 같은 인터넷 프로토콜이 사용자 인터페이스와 어플리케이션의 중개자 역할을 한다.

웹 어플리케이션의 발전과 더불어 현대의 다양한 IT 시스템들이 웹으로 통합되어 가고 있다. 일반적으로 웹 어플리케이션들은 시장의 요구에 따라 매우 짧은 개발 기간을 통해 상품화되는 경우가 많으며, 빈번한 개발자의 교체로 인해 소프트웨어 유지보수에 필요한 모듈화, 구조화가 되어 있지 않다.[3] 또한 웹과 관련된 기술들이 빠르게 발전한다는 점에서 기존에 만들어진 웹 어플리케이션들의 생명 주기가 매우 짧다는 특징을 가지고 있다.

2.2 유지보수에 대한 문제점

웹 개발의 특성상 웹 어플리케이션은 데이터베이스, CGI, C++, JAVA, ASP, JSP, DCOM, EJB등 단일 개발에 대해 여러 가지 언어와 기술이 통합적으로 사용되기 때문에 유지보수에 있어 일반적인 어플리케이션 보다 더 많은 문제점을 가지고 있다. 특히 웹 어플리케이션의 통합 형태인 웹사이트는 파일 중복, 웹 문서 양식의 부조화, 휘발성 링크와 같은 근본적인 문제점을 안고 있다.

웹 문서의 기본이 되는 HTML언어에는 일반적인 언어의 "Include"와 같은 지시자가 없기 때문에 개발자는 링크를 통해 다른 문서들을 내포하거나 복사를 통해 문서들을 나타낸다.[3] 문서의 링크와 복사는 웹사이트에 끊어진 링크와 중복된 파일이 나타나게 하는 원인이 되며, 이것은 웹 개발자로 하여금 개발 어플리케이션에 대해 일괄적인 업데이트를 어렵게 한다. 끊어진 링크는 연결된 사이트가 옮겨졌거나 없어진 경우, 연결된 문서가 삭제되었거나 이름이 변경된 경우, 오자로 인한 링크를 부정확하게 기술된 경우에 나타날 수 있으며 웹 개발자와 관리자들은 이것을 주기적으로 검사하고 고쳐주어야 한다.

웹사이트는 소프트웨어 공학에 대한 일반적인 지식이 전혀 없는 사람들조차도 매우 짧은 시간 안에 만들 수 있다. 작은 규모의 개인 웹사이트는 문제점을 가지고 있든지 아니면 업데이트를 언제 하든지 문제되지 않는다. 하지만 상업적인 대규모의 사이트는 웹사이트의 유지보수가 사업 성공의 핵심이 된다.

2.3 기존 연구 방법들

일반적인 웹 어플리케이션들은 꽉 짜여진 개발 일정과 빈번한 개발자 교체, 그리고 웹 개발 기술의 빠른 발전으로 인해 웹 어플리케이션들은 구조화와 문서화가 부족한 실정이다. 이러한 문제들을 해결하기 위해 기존 연구에서는 도구, 방법론, 모델, 프로세스의 측면에서 여러 가지 방법들을 제시하고 있다.

(1) 도구를 이용한 측면

Brereton등은 HTML의 태그를 프로그래밍 언어의 블록 구조와 유사하다고 지적하고, 웹 문서 사이를 연결하는 링크들은 GOTO 문장과 유사하다고 지적하였다.[4] 그들은 웹사이트에 존재하는 문서들을 평가할 수 있는 도구를 연구하였다. 그들이 개발한 도구는 1년 정도의 기간동안 웹사이트를 여러번 방문하여 웹 문서 내에서 변경된 점을 보고해 주는 것이다. Ricca와 Tonella도 이와 유사한 도구를 개발하였다.[5] 이 도구들은 동적 문서, 웹 객체 혹은 데이터베이스를 제외한 정적인 문서에만 초점이 맞추어져 있기 때문에 동적인 웹 문서를 포함하는 사이트의 구조정보를 추출하지 못하였다. 본 연구에서는 4단계 추출 과정을 통해 동적인 웹 문서를 포함한 보다 정확한 구조정보 추출을 시도하였다.

(2) 방법론

Antoniol등은 Relation Management Methodology(RMM)를 이용하여 웹 어플리케이션 구성요소를 엔티티 관계 모델로 제시하였다.[6] RMM은 웹 어플리케이션을 엔티티, 속성 그리고 관계로 정의한다. RMM과 같은 방법론을 통해 웹 어플리케이션을 개발하고 유지보수를 하는데 있어 개발자와 관리자의 어플리케이션에 대한 이해를 높여 웹 어플리케이션에 부족한 구조화와 문서화의 취약점을 보완할 수 있다.

(3) 모델을 이용한 측면

기존 연구에서는 웹 어플리케이션을 쉽게 모델링 할 수 있는 다양한 방법들을 제시해 왔다. Ceri등은 Web Modeling Language(WebML)를 제시하였다.[7] WebML은 웹 어플리케이션의 상위 수준의 개념적인 설명을 제공한다. Conallen은 UML을 웹 어플리케이션 모델에 적용하였다. 그는 UML로 Web Application Extension(WAE)를 제시하였고, 웹 문서들은 UML 구성요소로 모델링 되었다.[8] WebML과 WAE는 실제적인 구현을 모델링 하는 것보다 웹 어플리케이션의 명세서에 더 적합하다. 왜냐하면 그것은 자료 흐름을 모델링하는 측면이 부족하기 때문이다. 또한 기존에 이미 만들어진 시스템에 이 모델을 적용하기 위해서는 기존의 시스템을 분석하는 방법이 먼저 요구되어진다.

(4) 프로세스 측면

Boldyreff등은 웹사이트를 분석하여 중복된 웹 컨텐츠와 스타일을 추출하는 방법을 제안하였다.[3] 기존의 역 공학을 이용하여 웹 어플리케이션의 모든 요소들을 분리하여 하나의 저장소에 저장하고 이것을 다시 하위에서 세부적인 정보로 나누어서 기존의 웹 개발과 유지보수 방법보다 효율적인 정보로 새롭게 만들어 낼 수 있다. 앞서 기술한 모든 측면들 또한 웹 어플리케이션의 개발과 유지보수에 있어서 개발자의 편의와 사용자의 신뢰성 향상을 위한 새로운 개발 프로세스라고 말할 수 있다. 본 논문에서는 웹 개발과 유지보수에 있어 기초가 될 수 있는 웹 어플리케이션 구조화 작업을 위해 링크 정보를 추출하고 웹 로그 파일을 분석하여 활용하는 방법을 제시한다.

3. 웹 어플리케이션의 구조분석

웹사이트의 각 페이지 사이를 연결하는 링크들은 일반적인 소프트웨어의 자료흐름에 해당한다. 웹 문서 속에 포함되어 있는 링크들을 추출함으로써 웹 어플리케이션의 자료흐름과 구조를 파악하는데 있어 기초적인 자료로 활용할 수 있다. 웹 문서로부터 링크들을 추출해 내기 위해서는 구문 파싱을 통한 웹 문서의 분석이 필요한데, 웹 어플리케이션은 여러 가지 다른 언어의 조합으로 이루어져 있는 특징 때문에 이것을 개별적으로 분석하기에는 어려움이 있다. 또한 애플릿, 플래시와 같이 이진 형태의 웹 어플리케이션에 포함 되어있는 링크는 추출할 방법이 없다. 본 연구에서 이러한 문제점을 해결하기 위해 4단계의 추출과정을 거쳐 신뢰할 수 있는 정보를 얻기 위해 노력했다. 우리가 이용한 방법은 HTML 파싱, 직접적인 사용자입력, 로그 정보, 물리적 디스크 정보를 이용하여 추출한 정보들을 상호 보정하는 방식을 이용하였다.

3.1 HTML 태그와 링크상태 분류

다음은 HTTP 헤더 요청에 대한 응답으로 얻을 수 있는 컨텐트 타입과 하위 타입들을 간략하게 정리한 것이다.

<표 1> 컨텐트 타입과 하위 타입

Type	Subtype	Type	Subtype
text	html, sgml, plain ...	image	jpeg, gif, tiff ...
multipart	mixed, form-data ...	audio	basic, 32kadpcm ...
message	http, news, ffc822 ...	video	mpeg, quicktime ...
application	Postscript, pdf, zip ...	model	iges, vrml, mesh ...

<표 1>의 컨텐트 타입 중 text형태는 또 다른 링크들을 포함할 수 있기 때문에 링크들을 추출하는 과정에서 text형태의 링크는 또 다시 하위 요소들을 추출할 필요가 있다. 또한 추출하고자 하는 대상이 해당 웹 서버에 존재하는 파일을 대상으로 하므로 외부 웹 서버에 존재하는 문서에 대한 링크들은 웹서버 자체에 가지고있는 정보인 내부 링크들과 구분해야 된다. 외부 링크의 경우 테스트 대상이 아니기 때문에 하위까지 추출할 필요가 없고 해당 파일에 대한 검증과 정보만 얻으면 된다.

웹 어플리케이션의 사용자가 클라이언트의 브라우저를 통해 웹 문서에 나타난 링크를 누를 때 브라우저는 링크에 연결된 웹 문서를 서버에게 요청하여 사용자에게 보여준다. 하지만 어떤 상황에 따라 원하지 않은 결과를 얻을 수도 있다. 링크가 올바르지 않은 경우를 상황별로 분류하면 다음과 같다.

- 연결된 사이트가 옮겨졌거나 없어진 경우
- 연결된 페이지가 삭제되었거나 이름이 변경된 경우
- 링크가 부정확하게 기술된 경우(오자 또는 존재하지 않는 문서를 기술)
- 페이지는 존재하지만 권한 설정이 잘못된 경우
- 링크정보가 관련 없는 정보를 연결할 경우(연결된 정보가 바뀐 경우)

이것은 서버에 존재하는 웹 문서들이 문제점을 가지고 있음을 의미하며, 올바르지 않은 링크들은 웹사이트 사용자의 신뢰도를 떨어뜨리는 원인이 된다. 웹 서버 관리자는 이러한 링크들을 주기적으로 확인하고 고쳐주어야 한다. 웹 문서의 상태는 웹 서버에서 응답으로 보내지는 HTTP 헤더를 통해 나타나므로 본 연구에서 HTTP헤더의 상태코드 분석을 통해 올바르지 않은 링크들을 구분하였다.

웹 서버는 3자리 정수로 이뤄진 HTTP헤더의 상태코드를 통해 <표 2>와 같은 정보를 사용자에게 알려준다.[9]

<표 2> HTTP 응답의 상태 코드와 설명

Status Code	Description
1XX (information status code)	부가적인 정보를 전달하는데 사용
2XX (success status code)	요청이 성공한 경우
3XX (redirection status code)	URL이 옮겨진 상태
4XX (client error code)	구문에러와 같은 클라이언트 측의 에러를 나타내는 상태
5XX (server error code)	서버 측의 에러를 나타냄

HTTP 헤더 요청에 의한 응답으로 상태코드가 "200 OK"일 경우 요청이 성공한 경우이며 컨텐트 타입이 "text"이면 하위 링크 포함이 가능하다고 할 수 있다.

앞서 기술한 링크들은 HTML 태그에 의해 표현된다. HTML 파서를 이용하여 웹 문서로부터 태그들을 분리하면 해당 태그가 표현하는 링크의 정보들을 얻어올 수 있다. 본 논문에서는 HTML파싱을 하여 웹 문서에 포함된 정적인 링크들을 추출하였다. 파싱된 태그들 중 링크를 표현하는 태그만을 처리하기 위해 HTML에 기술하는 모든 태그들 중 링크를 표현하는 것과 해당 태그의 속성을 <표 3>에 정리하였다.

<표 3> 링크를 표현하는 HTML 태그와 속성

HTML Tag	Attribute
<A>	HREF
<APPLET>	CODEBASE, CODE
<AREA>	HREF
<BGSOUND>	SRC
<BODY>	BACKGROUND
<EMBED>	SRC
<FIG>	SRC
<FORM>	ACTION
<FRAME>	SRC
<IFRAME>	SRC
<IMG>	SRC, DYNSRC, USEMAP
<INPUT>	SRC
<INSERT>	CLASSID, USEMAP, DATA
<ITEM>	CLASSID, USEMAP, DATA
<LINK>	HREF
<MAP>	NAME
<OBJECT>	CODEBASE
<OVERLAY>	SRC
<PARAM>	VALUE
<SCRIPT>	SRC, FOR
<TD>	BACKGROUND
<TH>	BACKGROUND
<TR>	BACKGROUND

3.2 링크 추출을 통해 얻을 수 있는 정보

웹 어플리케이션 개발자들은 링크 추출을 통해 해당 웹사이트에 포함된 링크들의 다양한 정보를 얻을 수 있다. 본 연구에서는 다음과 같은 정보들을 추출하여 분리하였다.

- 링크 URL
  웹사이트에 포함된 모든 링크들과 해당 링크의 정확한 경로를 알 수 있다. 웹 문서에 포함된 링크들은 상대경로와 절대경로로 나타나기 때문에 정확한 링크 경로를 통해 변경 시 수정이 용이하다.
- 최종 수정 날짜
  웹 문서가 언제 변경이 되었는지 알 수 있으며, 이를 통해 개발자들은 웹사이트의 일괄적인 업데이트 대상과 정보를 얻을 수 있다.
- 웹 서버 정보
  웹 문서를 담고있는 서버의 정보를 나타낸다. 내부링크인 경우는 관리자가 서버 정보를 알고 있지만 외부링크의 경우는 알지 못 한다. 웹 서버 정보는 HTTP 헤더에 포함되어 있기 때문에 이것을 추출하여 웹 문서의 추가적인 정보로 활용하였다. 이것을 통해 웹 관리자는 외부 웹 서버의 정보를 알아낼 수 있다.
- 링크 MIME 타입
  앞서 설명한 링크 종류를 판별할 때 사용한다. 또한 추가적인 하위 링크 추출 판별 여부에도 사용된다. 링크 MIME 타입 중 "text/html"은 하위에 또 다른 링크를 포함 할 수 있으므로 추가적인 추출 대상이 된다.
- 링크 깊이
  링크가 몇 번째 단계에서 추출되었는지를 나타낸다. 링크 깊이가 클수록 웹 사용자들은 원하는 정보를 얻기 위해 계속적인 클릭을 해야 하기 때문에 사용자들로 하여금 웹 서핑을 어렵게 만드는 원인이 된다. 링크 깊이가 큰 것은 바로가기와 같은 직접적인 연결이 필요하다.
- 부모 웹 문서
  추출된 링크가 어떤 문서로부터 추출되었는지를 나타낸다. 링크 변경 시 수정해야 할 문서를 쉽게 찾을 수 있다.
- 내부 / 외부 링크
  해당 웹 서버에 웹 문서의 존재유무를 나타낸다. 링크 MIME 타입과 같이 하위 링크 추출 판별 여부에 사용된다.
- 복제 상태
  웹 문서의 특성상 단일 문서에 대해 여러 개의 링크가 존재할 수 있다. 복제 상태는 다른 곳에서 같은 링크가 있음을 나타내며, 복제 상태 정보는 웹 어플리케이션을 수정할 때 일괄적인 업데이트를 용이하게 한다.

3.3 링크 추출 단계

본 연구에서는 보다 정확한 구조 정보 추출을 위해 4단계 링크 추출 과정을 정의한다. 1단계에서는 HTTP 헤더 요청과 헤더 분석, 문서 요청과 응답문서 파싱을 통해 정적인 웹 문서 속에 포함된 링크들을 추출하고, 2단계에서는 사용자 입력을 요구하는

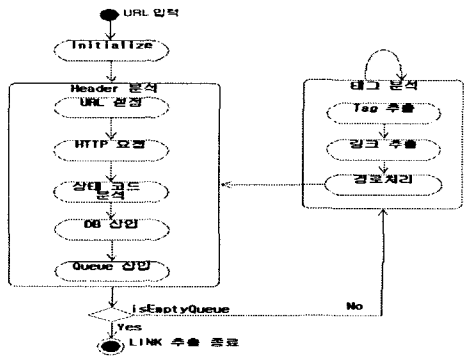
동적 웹 문서에 대한 링크 정보를 추출한다. 3단계에서는 로그 분석을 통하여 전 단계에서 추출한 링크를 보정하고 링크 분류 작업을 수행한다. 마지막 4단계에서는 물리적인 디스크에 저장된 폴더 구조와 파일을 검색하고 전 단계에서 추출한 링크 정보와 상호 비교하여 사용되지 않는 파일에 대한 정보를 추출하고 링크 추출 단계를 완료한다.

(1) 1단계 과정 (정적 웹 페이지 분석)

1단계 링크 추출은 시작페이지로부터 HTTP 헤더 요청과 헤더 분석, 문서 요청과 응답문서 파싱을 통한 링크 추출 과정을 반복적으로 수행한다.

반복적인 링크 추출 과정을 수행하기 위하여 큐를 사용할 수 있으며 계속적인 링크 추출이 필요한 페이지인 경우 큐에 삽입하고 큐가 비워지면 링크 추출 과정을 종료한다.

<그림 1>은 1단계 링크 추출 과정을 보여준다.



<그림 1> 링크 추출 과정

<그림 1>과 같이 1단계 링크 추출 과정은 헤더 분석 모듈과 태그 분석 모듈로 나뉘어진다.

헤더 분석 모듈은 HTTP 헤더를 요청하고 응답 헤더를 분석하여 응답 상태, 마지막 수정일, 서버 정보, 컨텐트 타입 등의 정보를 저장하고 계속적인 링크 추출이 필요한 링크인지 판별하여 큐 삽입 여부를 결정한다. 큐 삽입 가능 조건은 다음과 같다.

- 이전에 링크 추출 과정을 수행하지 않았던 문서
- 상태 코드가 "200 OK"
- 컨텐트 타입이 "text/html"
- 내부링크(외부 링크인 경우 하위 링크 추출 안함)

태그 분석 모듈은 하위 링크 추출이 필요한 링크가 저장된 큐에서 하나의 링크를 가져와서 문서를 요청하고 응답 문서 파싱을 통해 <표 3>과 같이 링크 정보를 내포하는 태그를 분리하고 하위 링크 정보를 추출한다.

다음으로 새롭게 추출한 링크에 대하여 외부링크와 내부링크를 분리하고 상대 경로가 헤더 분석에서 HTTP 요청 시 처리 할 수 없기 때문에 이를 절대 경로로 바꾸어 주기 위한 과정을 거친 후 헤더 분석 과정을 반복하게 된다. 추가적으로 1단계에서는 응답 문서 파싱을 통해서 새롭게 추출한 태그 정보에서 사용자 입력을 필요로 하는 폼을 포함하는 문서를 식별하여 2단계 링크 추출을 위한 폼 처리 큐에 삽입한다.

(2) 2단계 과정(폼 입력을 통한 동적 웹 페이지 분석)

1단계의 태그 분석 과정에서 얻어진 태그들 중 <FORM> 태그는 사용자 입력에 따라 다른 페이지를 나타낼 수 있는 동적 페이지를 포함 할 수 있다. 본 연구진행 과정 중 초기에는 자동적인 추출을 시도하였으나 웹 게시판이나 방명록과 같이 검색을 통해 생성되는 방대한 링크들은 분석 시간이 오래 걸리고, 정확하지 않은 결과를 나타낼 수 있기 때문에 사용자로부터 직접적인 입력을 통해 실제적인 결과를 얻을 수 있도록 하였다.

폼 처리 방식은 1단계 과정이 끝난 후 폼 처리 큐에 삽입된 링크를 순차적으로 가져와서 웹 브라우저와 동일한 환경에서 사용자가 입력을 할 수 있게 하고, 사용자 입력 확인 후 응답 문서는 정적 문서이므로 1단계 링크 추출 과정과 동일한 방법으로 링크추출이 가능하다.

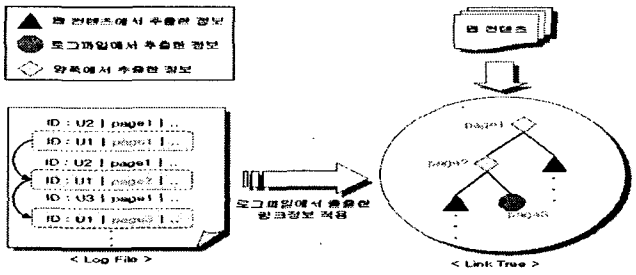
2단계 과정을 통하여 사용자 입력을 요구하는 동적 문서에 대한 링크 추출이 가능하며 2단계 과정은 하위 링크 추출을 위한 큐와 폼 처리를 위한 큐가 모두 비워졌을 때 종료된다.

(3) 3단계 (로그 분석 및 링크 보정)

웹 서버에 포함된 로그 파일에는 다양한 정보들을 포함하고 있다. 웹 로그 파일에 기록되는 정보들 중 "서비스 상태 코드"와 "요청 형식"은 앞서 추출한 링크정보와 동일한 정보를 담고 있다.

또한 로그 정보는 사용자 행동을 순차적으로 기록함으로 웹 페이지의 링크 정보를 내포하며 사용자들이 직접적으로 요청한 문서를 기록하고 있기 때문에 로그 분석을 통해 앞 단계에서 분석한 링크정보를 신뢰할 수 있는 정보로 만들 수 있다. 2단계 과정을 거쳐 얻어진 정보와 로그 파일에 기록된 링크정보를 상호 비교하여 링크에 대한 세부적인 정보를 얻을 수 있다.

다음 <그림 2>은 로그 정보를 이용하여 링크 정보를 보정하는 것을 도식화 한 것이다.



<그림 2> 로그파일 분석정보를 링크 정보 추출에 이용

추출한 링크 정보와 로그 분석을 통한 링크 정보에 모두 존재하는 링크는 정확한 링크라고 볼 수 있다. 링크 정보에는 추출되지 않았지만 로그정보에는 기록된 링크는 앞 단계에서 추출하지 못한 새롭게 추가된 링크이다. 반대로 링크 정보에는 기록되어 있고 로그정보에는 기록되지 않은 링크는 실재적으로 웹 문서에 표현된 링크이지만 사용자의 요청이 없는 링크라고 볼 수 있다.

로그 파일에는 링크 정보 이외에 웹사이트를 관리하는데 필요한 유용한 정보들을 포함하고 있는데 이것은 4장에서 다시 상세히 논의한다.

(4) 4단계 과정(물리적 디스크 분석)

웹 서버에 존재하는 자원들은 사용자의 요청에 의해 호출되어지고, 링크에 의해 서로 연결 되어 있다. 개발 과정에서 만들어진 파일들은 최종적으로 배포되는 제품에 포함될 수도 있고, 삭제 또는 변경에 의해 웹 서버 자체에 그대로 남아 있을 수도 있다. 웹 어플리케이션 배포전이나 수정 후 발생되는 사용되지 않는 파일들은 웹 서버의 자원을 낭비하는 결과를 가져오게 된다.

전 단계에서 얻어진 정보와 물리적으로 존재하는 폴더 구조와 파일들을 서로 비교함으로써 사용되지 않는 파일을 찾을 수 있다. 물리적 디스크에는 존재하지만 링크정보에 포함되지 않은 정보는 사용되지 않는 파일이라고 볼 수 있으며, 관리자는 이러한 불필요한 파일들을 찾아서 백업하거나 제거해 주어야 한다.

물리적 디스크에 존재하는 파일의 경로(물리적 디스크 경로)와 링크정보에 표현된 경로(URL)는 서로 형태가 다르기 때문에 상호 비교하기 위해서는 두 형태의 경로를 상호 전환하는 것이 필요하다.

아래의 <표 4>는 두 경로 표현에 대한 상호 전환을 보여준다.

<표 4> URL과 Disk Path 전환

URL	Disk Path
http://localhost/	D:\Home\
http://localhost/Images/home.gif	D:\Home\Images\home.gif

4. 웹 로그 분석 및 링크정보 보정

4.1 로그의 종류 및 공통적인 로그 항목

로그 파일에 저장되는 항목들은 웹 서버에 따라 조금씩 다르다. 초기에 웹 서버에 저장하는 로그 데이터 항목들은 웹 서버마다 달라서 로그 분석에 어려움이 많았다. 항목들이 다르기 때문에 한 로그 분석 프로그램이 모든 로그 파일을 분석할 수 없었다. 그래서 등장한 것이 CLF(Common Log Format)이다. 이 경우 공통적인 형식으로 로그 데이터를 저장하기 때문에 웹 서버가 다르다 하더라도 로그 데이터 형식은 비슷하게 된다. 그리고 그 항목들은 대부분의 상황에 필요한 항목들을 제공한다. 현재의 대부분의 웹 로그는 CERN과 NCSA에서 HTTP 프로토콜로 규정한 CLF를 따른다. 이 표준을 따르는 로그 항목에는 사용자의 IP 주소, 데이터 전송을 위한 프로토콜, 에러 코드, 전송된 데이터 길이 등이 포함된다.

현재 세계적으로 운영 중인 웹 서버 중 가장 높은 점유율을 차지하는 것은 Apache 서버이고 그 다음이 Microsoft-IIS 서버이다. 이 둘이 80%이상을 차지한다.[10] 따라서 본 논문에서는 Apache와 Microsoft-IIS에 공통적으로 기록될 수 있는 NCSA 공통 로그 파일 형식을 분석하였다. NCSA 공통 로그 파일은 <표 5>과 같은 정보를 담고 있다.

<표 5> NCSA 공통 로그 파일 형식

저장 필드	저장 예제
원격 호스트 이름	127.0.0.1
사용자 이름	REDMOND\fred
날짜	08/Apr/1998
시간과 GMT 오프셋	17:39:10 -0800
요청 형식	GET /scripts/iiadmin/ism.dll?http/serv, HTTP/1.0
서비스 상태 코드	200
보낸 바이트 수	3401

4.2 로그 분석 정보

NCSA 공통 로그 형식은 기본적인 사항만을 저장한다. 또한 웹 로그를 이용하여 웹사이트를 통해 이루어지는 모든 정보를 정확하게는 알 수 없다. 따라서 로그 데이터가 정확하지 않을 수 있다는 한계를 극복하고 필요한 정보를 얻기 위해서는 다양한 웹사이트 특성에 맞는 다양한 순수 로그 정보에 대한 정제 및 분석 알고리즘의 개발이 필요하다.

다음은 일반적으로 로그를 분석하는 항목을 분류한 것이다.

(1) 페이지 분석

총 페이지뷰수, 일평균 페이지뷰수, 기본 페이지뷰수, 방문당 페이지뷰수

(2)파일 분석

총 히트수, 일평균 히트수, 방문당 히트수

(3)방문자 분석

방문수, 일평균 방문수, 순 방문수, 총 방문수, 일평균 방문자, 방문자 1인당 방문수, 평균 이용시간

(4)에러 분석

총 에러수, 일평균 에러수, 페이지당 에러수

(5)서버 분석

총 전송 데이터량, 일평균 전송 데이터량, 히트당 전송된 평균 데이터량, 브라우저별 사용율

본 논문에서는 일반적인 로그분석 목적보다는 더 정확한 링크

정보 추출과 향후 웹 테스팅에 활용하기 위하여 사용자 행동 정보추출을 수행하였다.

4.3 로그 파일에서의 링크 정보

웹 로그에서 링크 정보를 추출하고 4.4절에서 설명하는 사용자 행동 정보를 추출하기 위해서는 먼저 웹 로그에 기록된 IP 정보를 단일 사용자로 가정해야 한다. 그러나 이러한 가정은 다음과 같은 경우 정확한 정보 추출을 할 수 없도록 한다.

- 사용자가 유동 IP를 사용할 경우 정확한 클라이언트 IP를 파악할 수 없다.
- 캐쉬를 통할 경우 사용자는 페이지를 봤지만 로그에는 그 데이터가 저장되지 않는다.
- 웹 브라우저의 이동 메뉴를 사용한 경우 그 행위들은 로그에 기록되지 않는다.
- Proxy Server를 사용하는 경우
- 한 사용자가 여러 개의 브라우저로 동일 사이트를 방문하는 경우

<표 5>에서의 "요청형식"은 링크정보를, "서비스 상태 코드"는 링크의 상태 정보를 가진다. 따라서 웹 로그 파일에서 웹사이트 이용자의 실제적인 이동 경로를 파악할 수 있다. 이 정보를 링크 추출 과정에 이용하면 더 정확한 링크 정보를 추출할 수 있다.

링크 추출과정에서 추출한 링크정보와 로그에서 추출한 링크 정보를 이용하면 다음과 같이 세부적으로 링크를 분류하고 이용할 수 있다.

- **링크 정보와 로그 파일에 모두 기록된 링크**
  웹 문서에 존재하며 사용자가 접근이 가능한 정확한 링크라고 볼 수 있다. 이 링크들 중 링크 연결 상태가 올바르지 않은 경우 즉, "서비스 상태 코드"가 200번이 아닌 경우는 사용자가 웹 문서들을 보는 동안 에러에 접하게 하는 원인이 되므로 웹 서버 관리자는 이 정보를 통해 문제점을 찾고 올바르게 고쳐 주어야 한다.
- **링크 정보에는 없지만 로그 파일에 기록된 링크**
  실재적으로 존재하지만 앞 단계에서 자동적으로 추출하지 못한 링크에 해당한다. 이 링크 정보들은 앞 단계에서 추출한 링크정보들과 같이 새로운 링크로 추가하였다.
- **링크 정보에는 존재하지만 로그 파일에는 없는 링크**
  실재적으로 웹사이트 문서들 내에 존재하는 링크이지만 웹사이트 이용자에게 직접적으로 이용되지 않는 링크에 해당한다. 사용되지 않는 링크가 존재하는 이유는 해당 링크를 나타내는 문구가 사용자들로부터 관심을 얻지 못한 경우이거나 혹은, 사용자들이 해당 링크들을 찾지 못한 경우이다. 웹 서버 관리자는 이용되지 않는 링크들의 원인을 찾아서 적절하게 대처해야 한다.

4.4 사용자 행동 정보

로그 분석을 통해 추출한 정보를 사용자별 패턴과 그룹별 패턴으로 구분해서 패턴 정보를 추출한다. 추출한 정보는 웹 테스팅을 위한 환경 설정과 가상 사용자에 대한 행동 정보 생성에 중요하게 이용된다.

4.4.1 사용자별 행동 패턴 정보 추출

사용자별 행동 패턴 정보는 문서 이동 경로, 문서 당 요청 수, 문서 당 머문 시간으로 구분한다.

문서 이동 경로는 동일한 IP를 묶고 요청시간별로 정렬하여 해당 IP가 접근한 문서들의 이동 경로를 추출한다.

문서 당 요청 수는 동일한 IP를 묶고 동일한 요청 페이지를 합산하여 추출한다.

문서 당 머문 시간은 사용자의 문서 이동 경로 정보에서 뒤의 경로에서 대한 요청 시간과 앞의 경로에 대한 요청 시간의 차를 계산하여 추출한다. 머문 시간의 계산에서 고려해야 할 점은 문서 요청 후 아무것도 하지 않는 경우 실제보다 길어지게 된다. 따라서 최대 머문 시간에 대한 제한이 필요하다. 4.4.3절에서 자세히 설명하겠다.

4.4.2 그룹별 행동 패턴 정보 추출

그룹별 행동 패턴 정보는 동시 접속자수, 문서 당 총 요청 수, 문서 당 총 머문 시간, 전체 요청 문서 수로 구분한다.

동시 접속자 수는 특정 시점에서의 동시 접속자 수와 일정 시간 내의 동시접속자수로 구분할 수 있다. 로그에는 로그인과 로그아웃에 대한 정보가 기록되지 않으므로 정확하게 계산하는 것은 어렵다. 본 연구에서는 4.4.3절에서 설명하는 세션 종료 시간과 아래의 계산 방법을 이용하여 동시 접속자수를 계산한다.

특정 시점에서의 동시 접속자수는 다음과 같이 계산 할 수 있다.

T1 : 특정 시점
T2 : IP당 마지막 문서 요청 시간
T3 : 세션 종료 시간
S : 동시 접속자 수
①로그의 처음부터 T1까지 다른 IP가 나타나면 S + 1
②T1을 기준으로 IP당 T2 추출
③만약 T1 - T2 > T3 이면 S - 1

일정 시간 내의 동시 접속자수는 다음과 같이 계산 할 수 있다.

T1 : 처음 시점
T1′ : 나중 시점
T2 : IP당 마지막 문서 요청 시간
T3 : 세션 종료 시간
S : 동시 접속자 수
①S = T1과 T1′사이의 다른 IP수
②각 IP별로 만약 T1‘ - T2 > T3 이면 S - 1

문서 당 총 요청 수는 요청된 문서 정보에서 동일한 문서를 묶어서 합산하여 추출한다.

문서 당 총 머문 시간은 사용자별 패턴에서 추출한 문서 당 총 머문 시간을 합산하여 추출한다.

전체 요청 문서 수는 요청된 문서 중 다른 문서 수로 계산한다.

4.4.3 세션 종료 시간(최대 머문 시간)

로그에는 로그인과 로그아웃에 대한 정보가 기록되지 않으므로 머문 시간의 계산과 동시 접속자 수 계산에서 로그아웃 시점을 예상하기 위해서는 세션 종료 시간에 대한 결정이 필요하다.

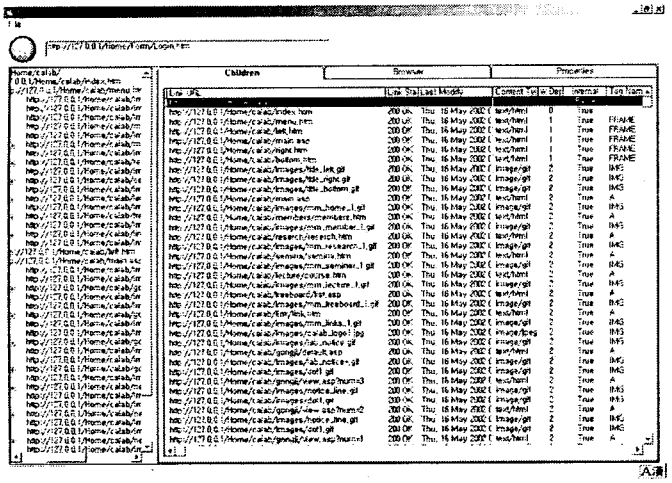
다음과 같은 다양한 형태의 세션 종료 시간을 적용할 수 있다.

- 모든 사용자, 모든 페이지에 동일한 시간 적용
- 모든 사용자에게 페이지별 다른 시간 적용
- 사용자별 다른 시간 적용
- 사용자별, 페이지별 다른 머문 시간 적용

5. 구조 분석 및 로그 분석 시스템

5.1 전체 화면 구성

<그림 3>은 본 논문에서 구현한 시스템의 실행화면이다.



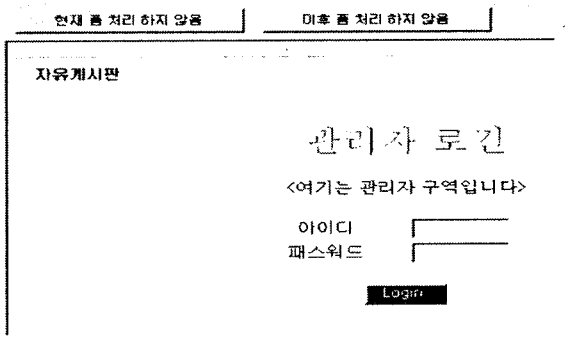
<그림 3> 시스템 전체화면 구성

그림에 나타난 것처럼 상단의 "URL 입력 필드", 링크의 깊이에 따라서 구조를 나타내는 "탐색 창" 그리고, 추출한 링크들의 정보들을 나타내는 "링크정보 창"으로 구성되어 있다.

"링크정보 창"은 추출한 링크들의 세부 정보를 표현하기 위해 테이블, 브라우저, 속성 탭으로 나누어서 표시하였다. 테이블 탭에서는 링크들의 정보를 테이블로 정렬해서 사용자가 알아보기 쉽도록 하였고, 브라우저 탭을 통해 링크들을 직접 확인할 수 있도록 하였다. 속성 탭에는 테이블 탭에서 나타내지 못한 정보들을 한 페이지에 함께 나타내어 좀 더 자세한 정보를 확인할 수 있도록 하였다.

5.2 폼 입력 처리

최초 사용자가 "폼 입력 필드"를 통해 초기 URL을 입력 후, 1단계 정적 웹 페이지 분석이 끝나게 되면 추출한 태그 중에 폼 태그의 존재유무에 따라서 <그림 XX>과 같이 브라우저와 유사한 화면을 사용자에게 보여준다.



<그림 4> 폼 입력 화면

사용자들은 직접 화면을 확인하고 계속적인 추출을 할 것인지, 아니면 이후의 모든 폼들을 무시한 체 다음 3단계인 로그 분석 단계로 넘어 갈 것인지 선택할 수 있다. 또한 현재 폼을 생략할 수 있는 단추를 두어서 방명록이나 게시판과 같이 링크 추출시간이 오래 걸리는 방대한 링크 페이지를 생략할 수 있도록 하였다.

6. 웹 어플리케이션 테스팅에 적용방안

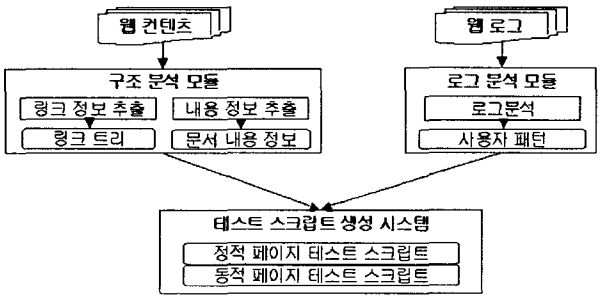
웹 기반 시스템에 대한 효율적인 개발과 유지보수 및 확장을 위해서는 시스템 개발의 생명주기에 따라 적절한 테스팅 방법에 대한 연구가 필수적이다. 개발 단계에서는 개발중인 시스템에 존재할 수 있는 병목현상과 데드락과 같은 문제점을 미리 예측함으로써 즉각적인 문제 해결이 필요하며, 테스팅 단계에서는 가상 사용자 시뮬레이션을 통하여 실제 시스템 운영시에 발생할 가능성이 있는 문제점들을 시험할 수 있어야 한다. 또한 시스템을 배포하여 운영하는 단계에서는 운영중인 시스템에 대한 모니터링을 통하여 발생 가능한 문제점을 예측하고 미리 대처하거나 사용자 수의 증가에 따른 시스템 확장 계획 수립에 적용할 수 있어야 한다. 이와 같이 웹 어플리케이션의 테스팅에 대한 연구는 시스템의 생명주기에 따라 발생 가능한 문제에 대한 조기 발견과 시스템의 성능 향상 및 시스템의 확장 계획 수립에 있어 중요한 선결 과제로 제기 되고 있다.

웹 테스팅은 가상의 환경을 설정하고, 행동 정보를 가진 가상의 사용자를 생성하여 실제 운영 환경과 유사한 시뮬레이션을 구현함으로써 이루어진다. 이런 가상의 환경과 사용자에 대한 정

보를 저장하는 문서를 테스트 스크립트라고 한다. 웹 테스팅을 위한 테스트 스크립트를 작성하는 방법으로 웹 브라우저상의 사용자 행동(마우스 클릭 또는 키보드 입력)을 레코딩하는 방법, 로그파일을 분석한 정보를 이용하는 방법, 디렉토리 구조 정보를 이용하는 방법 등이 제시되고 있지만 실제 운영 환경에 근접한 테스트 스크립트를 작성 하기는 어렵다. 특히, 사용자 입력을 필요로 하는 동적 페이지의 경우에는 매우 제한적이다.

본 논문에서는 정확하고 신뢰성 있는 웹 테스팅을 위한 테스트 스크립트 생성을 위해서 3장과 4장에서 설명한 구조분석 정보와 로그 분석 정보를 이용하는 방법을 제시한다. 구조분석 과정에서 링크 정보와 사용자 입력 요구에 대한 정보를 추출하고 웹 로그를 분석하여 테스팅 환경 설정에 필요한 정보와 사용자 행동에 대한 패턴 정보를 추출하였다. 이렇게 추출한 정보를 이용하여 테스팅 환경 설정과 가상 사용자 생성에 이용할 수 있다.

다음의 <그림 5>는 테스팅을 위한 테스트 스크립트 생성 과정을 보여준다.



<그림 5> 테스트 스크립트 생성 과정

로그 분석 정보와 링크 정보를 이용하여 정적 페이지에 대한 테스트 스크립트를 작성하고 내용 정보를 고려하여 동적 페이지 테스팅 시에 문제를 발생시키는 주요 요인인 사용자 입력 폼, 값 전달 방식(POST, GET, HEAD), 데이터베이스에 접근 등을 해결하는 동적 페이지에 대한 테스트 스크립트를 생성한다. 본 연구에서 작성한 테스트 스크립트는 사용성과 확장성을 위해 XML 형식을 따른다.

다음은 간단한 테스트 스크립트의 예이다.

```
<group name="default">
  <duration>
    <minute>10</minute>
  </duration>
  <stresslevel>50</stresslevel>
  <stressdelay>10</stressdelay>
  <testlist>
    <testitem>
      <method>post</method>
      <path>http://127.0.0.1/login.asp</path>
      <thinktime>5</thinktime>
      <attlist>
        <attitem>
          <name>id</name>
          <value>guest</value>
          <name>pd</name>
          <value>guestpassword</value>
        </attitem>
      </attlist>
    </testitem>
    . . .
  </testlist>
</group>
```

7. 결론 및 향후 연구과제

본 논문에서는 웹 어플리케이션의 효율적인 개발과 유지보수를 위하여 웹 어플리케이션의 구조 정보인 링크 정보를 추출하고, 웹사이트에 대한 유용한 정보를 담고 있는 로그 파일을 분석하고 활용하는 방법을 제시하였다.

보다 정확한 링크 추출을 위하여 4단계에 걸쳐 링크 추출과정을 수행하였다. 4단계 링크 추출 과정을 통해서 동적 문서에 대한 링크 추출도 가능하였고 추가적으로 링크 깊이, 끊어진 링크, 사용되지 않는 파일 등과 같은 웹사이트 유지보수를 위한 유용한 정보를 추출할 수 있었다.

웹 로그 분석 과정을 통하여 더 정확한 링크 정보를 추출할 수 있는 방법을 제시하였으며 사용자 행동 정보를 추출하여 링크 추출 과정에서 추출한 링크 정보와 문서 내용 정보와 함께 웹 테스팅을 위한 테스트 스크립트 생성에 이용함으로써 웹사이트의 실제 운영환경과 유사하고 신뢰성 있는 웹 테스팅 방법을 제시하였다.

그리고 본 연구를 기반으로 개발한 링크 정보 추출과 웹 로그 분석 시스템을 보였다.

향후 연구과제로는 웹사이트에서 추출한 여러 가지 정보의 활용도를 높이기 위하여 구조적이면서 쉽게 이해될 수 있는 표현하는 방법에 대한 연구와 다양한 뷰를 제공하는 방법들에 대한 연구가 필요하다. 그리고 웹 테스팅과 관련하여 테스팅 환경 설정, 테스팅 시뮬레이션, 모니터링 등에 관한 연구를 지속적으로 수행하여 통합적인 웹 테스팅 도구를 개발할 예정이다

[참고문헌]

[1] Benoit Leger, Jean-Christophe Cimetiere, "Web Load and Performance Testing Tools", "www.trendmarkers.com", 2000

[2] Hung Q. Nguyen, "Testing Applications on the Web",

Wiley Computer Publishing, 2001

[3] Boldyfeff C., Kewish R., "Reverse engineering to achieve maintainable WWW site", Reverse Engineering, 2001. Eighth Working Conf. on, 2001 [pp. 249 - 257]

[4] P. Brereton, D. Budgen, G. Hamilton., "Hypertext: The Next Maintenance Mountain", Computer 31(12):49-55, Dec, 1998

[5] F. Ricca and P. Tonella, "Visualization of Web Site History", In Proceedings of euroREF: 7th Reengineering Forum, Zurich, Switzerland, Mar. 2000.

[6] G. Antoniol, G. Canfora, G. Casazza, A. D. Lucia, "Web Site Reenginnering using RMM", In Procceedings of euroREF: 7th Reengineering Forum, Zurich, Switzerland, Mar, 2000.

[7] S. Ceri, P. Fraternali, A. Bongio. "Web Modeling Labguage(WebML): a modeling language for designing Web sites", In The Ninth International World Wide Web Conference, Amsterdam, Netherlands, May 2000

[8] J. Conallen, "Building Web Application with UML", object technology, Addison-Wesley Longman, Massachusetts, USA, first edition, Dec 1999

[9] RFC 2616: Hypertext Transfer Protocol -- HTTP/1.1, ftp://ftp.isi.edu/in-notes/rfc2616.txt

[10] Netcraft Web Server Survey, http://www.netcraft.com/survey/