2001 년 한국정보처리학회 춘계 학술발표논문집 제 8 권 제 1 호

웹 기반 소프트웨어의 테스트 모델에 관한 연구

권영호, 최은만
동국대학교 컴퓨터멀티미디어공학과
e-mail : {yhkwon, emchoi}@dgu.ac.kr

A Study on Test Model for Web-Based Software

Young Ho Kwon, Eun Man Choi
Dept. of Computer Engineering, Dongguk University

요 약

이 논문은 웹기반 소프트웨어를 테스트하기 위한 오러클을 생성하는 방법을 제안하고 설명한다. 웹 페이지를 구성하는 응용 컴포넌트들의 구조를 파악하고 각 페이지를 구동시키는 액션들을 찾아내어 상태기반의 테스트 데이터를 찾아내는 방법이다. 테스트 스크립트를 작성하기 위하여 partial-W 방법을 도입하였으며 이를 이용하여 테스트 케이스의 값을 선택할 수 있다. 테스트 슈트는 언어 독립적이며 실행가능하다. 웹 응용의 특징인 동적인 인터렉션을 유한 상태기계(Finite State Machine)로 표현하고 각 상태를 변화시키는 웹 페이지의 사용자 입력을 결합하여 테스트 오러클을 생성한다.

1. 서 론

웹과 인터넷이 보급되면서 일반인이 컴퓨터에 더 가깝게 되었고 인트라넷의 구축으로 웹 환경에 사용할 목적으로 개발되는 소프트웨어가 늘어나고 있다. 특히 E-비즈니스의 확장과 함께 웹 기반 소프트웨어 개발에 관한 기술은 필수적이며 중요하다.

일반적인 소프트웨어 엔지니어링 기술이 웹 기반 소프트웨어의 개발에도 적용될 것인가? 이것이 웹 엔지니어링이 다루는 문제이다[1][2]. 웹 사이트의 규모가 커지면서 여기에도 엔지니어링 어프로치가 필요하다. 단순한 HTML 파일이 아니라 웹 기반 응용 프로그램과 데이터베이스가 복잡하게 연결되어 있어 설계 원리와 엔지니어링 기술이 분명 필요하다.

웹 기반 응용 시스템은 일반 소프트웨어와 여러 면에서 차이가 난다. 네트워크 상에서 수행되며 클라이언트와 서버 부분이 있고 네비게이션과 링크를 제공하며 서버에서 정보를 다운로드하는 기능을 가질 수도 있고 사용자와 여러 가지 인터렉션이 이루어진다. 또한 트랜잭션 중심으로 데이터베이스를 접근하여 사용자가 원하는 작업을 하는 경우도 있다.

웹 기반 소프트웨어를 테스트하는 일은 일반 소프트웨어를 테스트하는 것과 많은 차이가 있다. 서로 다른 기종과 웹 브라우저에서 분산 형태 때로는 병렬로 처리되기 때문이다[3]. 더구나 웹 응용은 구조적인 프로그래밍에 의하여 표현되지 않는 경우가 많고 네비게이션과 상태 의존적인 동작이 많아 규모가 큰 경우 이해가 매우 어렵다. 이러한 어려움을 극복하고 그 특성을 고려하기 위하여 웹 기반 소프트웨어의 테스트는 다른 어프로치가 필요하다.

이 논문에서는 날로 증가하고 있는 웹 기반 소프트웨어를 테스트하기 위한 프레임워크를 제안한다. 대부분의 웹 기반 소프트웨어들이 화면에서의 사용자 자극과 이에 대한 프로그램의 반응이 테스트의 주된 관심사 이므로 유한상태기계(Finite State Machine)를 도입하는 것이 자연스럽다. 또한 웹 사이트들의 규모가 커지면서 이를 테스트하기 위하여 테스트 슈트를 자동으로 생성할 수 있는 도구는 매우 필수적이다. 이 연구에서는 이를 위한 테스트 스크립트를 생성하는 방법을 제안한다.

2. 웹 기반 소프트웨어의 테스트와 관련 연구

웹 이라는 특수한 환경에서 수행되는 소프트웨어를 테스트하는 방법에 대한 연구는 우선 도메인 기반의 소프트웨어 테스트 방법과 상태 기계를 표현하는 모델이 관련되어 있다.

웹 기반 소프트웨어는 도메인 모델이 웹 페이지에 잘 나타나 있고 웹 페이지를 이용하여 쉽게 유추할 수 있다. 따라서 웹 페이지의 정보를 도메인 모델로 표현하는 방법만 잘 연구한다면 도메인 모델을 이용하여 테스트하는 방법에 대한 여러 연구의 결과[4][5]들을 잘 이용할 수 있다.

도메인 기반의 테스트 방법은 시스템이 구현된 후 통합적인 시스템의 기능을 점검하는 과정에 분석단계의 결과들을 사용하는 것을 말한다. 특히 UML 을 사

용하여 표현된 도메인 모델과 스펙을 테스트 단계에서 사용하는 연구 결과도 유용하게 사용될 수 있다[5].

웹 사이트와 사용자 사이에는 여러 층의 소프웨어가 존재한다. 서버 쪽의 응용과 웹 브라우저에 다운되어 실행되는 GUI 중심의 클라이언트가 있고 이들은 HTML, Java Applet, Javascript, CGI script, Active Server Page 등 여러 가지 형태로 표현된다.

웹 기반 응용을 테스트하는 방법은 웹 페이지가 동적이냐 혹은 정적이냐에 따라 달라질 수 있다. 정적인 웹 응용은 구문 테스트와 기능 혹은 서비스 테스트 방법으로 시험할 수 있지만 동적인 웹 응용은 모듈 테스트, 통합 테스트, 시스템 테스트가 필요하다.

웹 테스트 방법이 만족하여야 할 사항과 지원되어야 할 도구를 정리하면 다음과 같다.

- 테스트 과정이 사용자가 웹 사이트와 상호작용하는 것을 점검하는 방법
- 웹 사이트가 수행될 때 브라우저와 서버에서 지원되는 모든 기능을 테스트하는 방법
- 웹 응용이 수행될 때 관련된 HTML 코드, 링크, 이미지 등을 체크할 수 있는 도구
- 웹 응용을 이루는 하위 컴포넌트의 기능과 상호작용을 테스트할 수 있는 드라이버
- 웹 테스트를 자동화할 수 있는 테스트 스크립트의 자동생성

이 논문에서는 위와 같은 이상적인 웹 테스트 환경을 조성하기 위하여 유한상태기계를 이용하여 웹 사이트와의 상호작용 과정을 표현하고 이를 바탕으로 테스트 스크립트를 작성하는 과정을 제안하였다.

3. FSM 을 이용한 테스트와 웹 페이지 표현 방법

트랜잭션 기반 시스템이나 상태 기반 시스템들은 주로 유한 상태로 시스템을 표현하고 이를 테스트에 응용한다. 유한 상태 기계를 이용한 테스트 방법은 Fujiwara 에 의하여 제안되었다[6]. 이 방법은 시스템의 행위를 유한상태의 집합으로 나타내고 여기서 접근 가능한 최소의 유한상태기계(minimal finite state machine)를 찾아내어 각 상태를 구동 시킬 수 있는 트랜지션들을 찾아내는 것이다. 즉 테스트 대상이 유한상태기계에서는 $p \in P$(P 는 유한 상태 기계에서의 부분 경로의 집합)와 $z \in Z$(Z 는 P 를 구동시키는 입력의 집합)으로 구성된다.

P 는 시스템의 상태를 파악하여 이를 유한 상태 기계로 바꾸고 테스트에 필요한 부분 경로를 찾아내기 위하여 테스트 트리를 만들면 쉽게 구할 수 있다. 반면 Z 는 설계 상에 나타나 있는 모든 입력 집합 W 에 의하여 결정되며 특정한 두 상태 사이의 트랜지션을 가능하게 하는 입력의 집합이다.

따라서 Z 가 될 수 있는 부분적인 W 를 구하면 시스템의 상태 변화를 테스트할 수 있는 값들을 찾을 수 있다. 이를 Fujiwara 는 partial-W 라 부르고 있다.

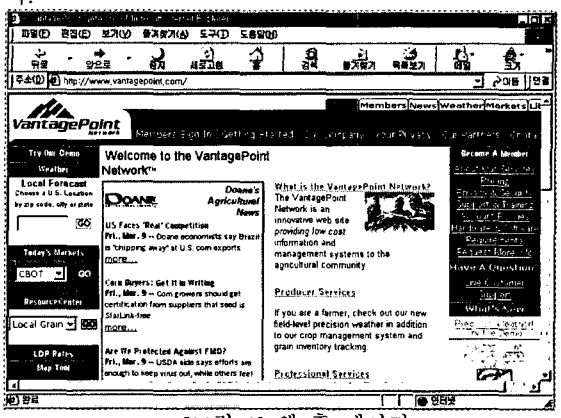
웹 기반 시스템의 partial-W 를 구하기 위하여 우선 웹 홈페이지에 나타난 시스템의 상태를 FSM 으로 나타내어야 한다. 그 방법은 다음과 같다.

[표 1] FSM 과 웹 페이지 요소 매핑

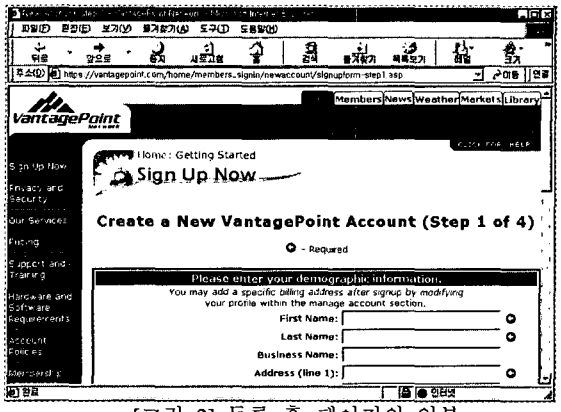
상태기계의 요소	웹 페이지 요소
상 태	웹 응용이 가질 수 있는 모든 상태(입력 상태, 처리 중, 등등)
트랜지션	웹 페이지에 가해진 자극으로 인하여 이루어지는 웹 페이지의 상태 변환
입 력	상태 변환을 일으키는 사용자 자극(필드 입력, 메뉴 선택)
출 력	새로운 상태 변화에 의하여 출력되는 결과

FSM 은 웹 응용이 화면에 표현된 것으로 구할 수 있다. 즉 웹 페이지의 상태를 변화시키는 입력과 그 결과인 출력을 찾아내어 그래프로 구성한 것이다.

그림 1 은 유한상태기계로 표현하기 위한 샘플 웹 페이지이다. 이 웹 페이지는 일반적인 전자상거래가 갖는 여러 가지 기본 기능을 가지고 있다. 예를 들면 그림 2 와 같은 회원가입 기능을 시작으로 멤버 기능, 뉴스, 날씨, 시장 정보, 라이브러리 등이 구현되어 있다.

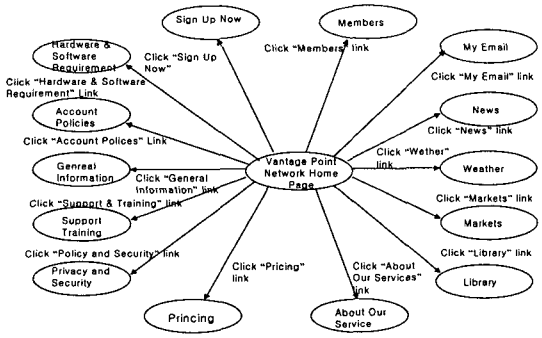


[그림 1] 웹 홈 페이지



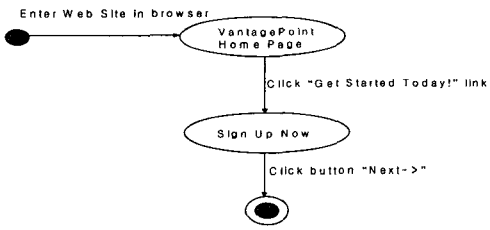
[그림 2] 등록 홈 페이지의 일부

위에서 제시한 최상위 홈 페이지는 Home, My E-mail, News, Weather, Market, Library, Help, Contact Us 등 여러 다른 페이지와 링크되어 있다. 홈 페이지에 표현된 링크를 FSM 으로 나타내면 그림 3 과 같다.



[그림 3] 샘플 홈페이지에 대한 최상위 FSM

그림 3 에 나타낸 링크는 최상위 홈페이지에서의 상태 변화만을 나타내었으나 각 기능에 따라 더 자세한 상태를 그림 4 와 같이 나타낼 수 있다.



[그림 4] Sign Up Now 를 위한 FSM

4. 유한 상태기계의 구성

웹 응용은 규모가 커서 단번의 FSM 으로 그리기는 힘든다. 따라서 웹 페이지의 정보를 기초로 하여 적당히 서브 시스템으로 분할할 필요가 있다. 이렇게 나누어 상태기계를 구성하는 것이 테스트 경로를 줄일 수 있다.

논문에서 예로 든 VantagePoint 홈 페이지의 경우 표 2 와 같이 다섯 가지 서브 시스템으로 재구성할 수 있다.

[표 2] 유한 상태를 구성하기 위한 시스템 분할

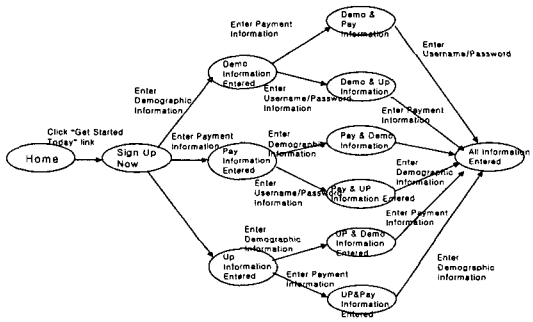
상 태	웹 페이지	근 거
Home	VantagePoint 홈 페이지[그림 1]	Sign Up Now 페이지로 전환되기 위한 첫 관문
Sign Up Now	Sign Up Now 전반	가입을 위한 신청
Enter Demographic Information	Sign Up Now 중반	기초 인적 정보
Enter Payment Information	Sign Up Now	회비 납부와 관련된 사항
Enter Username/Password Information	Sign Up Now	시스템 사용 계정에 관한 사항

하나의 웹 페이지 안에 있는 정보도 논리적으로 그루핑할 수 있다. Sign Up Now 와 같이 웹 페이지 안에 있는 정보가 많고 하나의 상태로 다루기 어렵다면 위와 같이 여러 상태로 나누는 것이 좋다.

웹 응용을 유한상태기계로 구축하는 다음 과정은 각 상태로부터 트랜지션할 수 있는 상태를 결정하는 것이다. 트랜지션은 두 가지로 나눌 수 있다. 웹 페이지에서 링크의 사용은 상태의 변화를 직접적으로 일으킨다. 또 다른 형태는 사용자에 의하여 제공되는 정보나 버튼을 누름으로 구동되는 상태의 변화이다.

가입을 위하여 정보를 입력하는 과정은 사용자가 입력하는 정보에 따라 상태의 변화가 달라질 수 있다. 모든 상태는 트랜지션을 가지며 웹 페이지에 직접 연결된 링크의 개수와 홈 페이지의 각 입력 필드의 조합으로 결정되어진다.

표 2 에서 분할한 다섯 가지 상태는 그 입력 순서에 따라 다른 상태변화를 가져올 수 있다. 예를 들면 인적 사항을 입력하고 다음으로 회비 납부 정보, 사용자 이름 및 패스워드 순서로 입력할 수도 있고 회비 납부정보를 먼저 입력하고 다음으로 사용자 패스워드, 마지막으로 인적 사항을 입력할 수도 있다. 그림 5 는 모든 가능 오퍼레이션의 조합을 FSM 으로 나타낸 것이다.



[그림 5] 사용자 등록을 위한 상태변환

위 그림에서 볼 수 있는 바와 같이 Sign Up Now 페이지의 입력은 어떤 순서로도 이루어질 수 있기 때문에 무조건 FSM 으로 구성하는 것은 매우 복잡하다.

FSM 을 간략하게 하기 위하여 all() 오퍼레이션이 필요하다. all() 오퍼레이션은 트랜지션이 어떤 순서로도 발생할 수 있음을 의미한다. 예를 들어 all(Demo, Pay, UP)은 Demo, Pay, UP 각각의 오퍼레이션이 어떤 순서라도 입력될 수 있음을 의미하며 이를 고려한 FSM 은 그림 6 과 같이 된다.

[그림 6] 축약된 형태의 FSM

5. 테스트 스크립트의 작성

FSM 에 표현된 모든 상태는 웹 페이지에서 이루어진 사용자 액션을 기초로 도달할 수 있는 상태와 링크되어 있다. 유한기계에서 입력이란 사용자가 행한 액션이다. 웹 테스트를 위하여 필요한 것은 바로 사용자 액션을 WinRunner 와 같은 도구의 스크립트로 표현하는 것이다. WinRunner 는 윈도우 응용을 위한 자동화 테스트 도구이다.

웹 페이지에서는 다른 트랜잭션에서와 같이 입력은 두 가지 형태가 있다. 웹 링크를 눌러 다른 페이지로 가는 액션과 웹 페이지에 사용자가 입력하거나 버튼을 선택하는 액션이다. 웹 페이지에서의 입력은 매우 다양하다. 텍스트, 체크박스, 선택박스 등이 입력이 될 수 있다.

이 논문에서 예로 든 VantagePoint 웹 페이지의 경우 사용자 등록을 위하여 다음 두 가지 종류의 입력이 있다. 홈 페이지에서 사용자 등록 페이지로 트랜지션하기 위하여 단순히 링크만을 클릭하면 된다. 이것을 WinRunner 를 위한 입력 스크립트로 표현하면 다음 그림 7 과 같이 된다.

```
web_browser_invoke(IE, "c:/sss_test/vantagepoint.html");
set_window("VantagePointNetwork: Home", 5);
web_link_click("Get Started Today");
```

[그림 7] 홈 페이지 입력 스크립트

위에서 작업한 바와 같이 사용자 가입 페이지는 세 부분의 하위 웹 페이지로 나눌 수 있다. 하위 웹 페이지에는 양식에 입력하는 부분과 선택하는 입력 버튼을 선택하는 입력이 있다. 최종적으로 "Next->" 버튼을 누르면 정보는 웹 서버에 제출된다. 이 과정을 스크립트로 표현하면 다음과 같다.

```
set_window("VantagePoint Network: Members Sign In", 4);
edit_set("firstName", "Kevin");
edit_set("lasttName", "Muir");
edit_set("businessName", "My Firm");
edit_set("add1", "1 Any Street");
edit_set("city", "Fort Collins");
list_select_item("region", "Colorado");
edit_set("zip", "80525");
edit_set("phone", "970 221-1234");
list_select_item("paymentType", "Annual Invoice");
button_set("FreePromo", ON);
edit_set("username", "muir99");
edit_set("pass1", "kevin2");
edit_set("pass2", "kevin2");
edit_set("keyword", "fname+2");
button_press("Next->");
```

[그림 8] Sign Up Now 입력 스크립트

결국 그림 6 에 있는 축약된 형태의 FSM 에 사용자 입력 대신 그림 8 에 있는 테스트 스크립트를 WinRunner 에서 실행하면 자동 테스트 된다.

특정한 액션이 이루어진 후 사용자에게 주어지는 출력은 웹 페이지나 다이얼로그 박스로 제공된다. 따라서 예상되는 출력을 미리 기술하여 테스트 스크립트를 수행한 후의 결과와 비교하여 테스트한다.

6. 결 론

웹 응용을 테스트하기 위한 모델을 제시하였다. 제시한 테스트 모델은 웹 응용의 중요한 특징인 사용자 자극에 대한 상태 변화를 잘 체크할 수 있다. 웹 응용의 각 상태변화를 FSM 으로 표현하고 FSM 을 축약하여 필요한 테스트 시퀀스를 정할 수 있다.

웹 테스트 모형은 웹 컨텐트 내부에 표현된 HTML 으로 찾기 어렵고 웹 페이지가 동적으로 실행되는 모습을 추적하여 찾을 수 있다. 따라서 웹 페이지에 표현된 각 필드와 입력 자료, 선택 버튼 등을 바탕으로 상태를 정의하고 각 요소에 대한 사용자의 자극으로 트랜지션이 이루어지고 있음을 파악하여야 한다.

웹 테스트는 상당히 이질적인 요소들 예를 들면 HTML, CGI, Java, JavaScript 등이 웹을 구성한다. 따라서 이들 상이한 형태의 컴포넌트들에 대한 단위 테스트 방법도 제안되어야 한다.

참고문헌

[1] Roger S. Pressman. *Software Engineering, A Practitioner's Approach*, 5th Ed. McGraw Hill, 2000

[2] 소프트웨어공학회지 특집, 2000, 12 월호

[3] T.A. Powell et. al., *Web Site Engineering: Beyond Web Page Design*, Prentice-Hall, 1998.

[4] L. Liuying, Q. Zhichang, "Test Selection from UML Statechart", Proceedings of TOOLS 31, 1999, pp.273-279.

[5] B. Regnell, P. Runeson, C. Wohlin, "Towad Integration of Use Case Modeling and Usage-Based Testing", Technical Report, Lund Institute of Technology, 1999., pp85-114.

[6] S. Fujiwara, G. Bochmann, et. al., "Test Selection Based on Finite State Machine", IEEE Trans. on Software Engineering, Vol. 17. No. 6, June 1991, pp.591-603

[7] D. Kung, Chein-Hung Liu, Pei Hsia, "An Object-Oriented Web Test Model for Testing Web Applications", Quality Software, 2000. Proceedings. First Asia-Pacific Conference on , 2000, pp. 111-120