웹 어플리케이션 설계 환경 개발

(A Study on Environment for Web Applications Design)

이 미 경* 강 병 도*
(Mi Kyong Lee) (Byeongdo Kang)

요 약 네트워크와 인터넷의 급격한 성장과 함께 기존의 많은 소프트웨어들이 웹 기반으로 그 모습을 변화하고 있으며, 이로 인해 웹 어플리케이션의 복잡성이 증가되고 개발에 있어 많은 비용과 인력이 소요되고 있다. 하지만 아직까지 웹 어플리케이션을 위한 명확한 개발 환경이 정의되지 않았으며 대부분이 경험에 입각하여 나름대로 개발하고 있는 실정이다. 따라서 웹의 특성을 잘 반영하는 개발 환경의 구축이 무엇보다 중요시된다. 본 논문에서는 웹 환경의 특성을 파악하여 웹 어플리케이션 개발에 적합한 프로세스를 정의하고 이에 따라 모델링 할 수 있는 환경을 제시한다. 또한 모델링 결과를 바탕으로 자동으로 소스코드를 생성한다. 결국 분석과 설계 단계에서 발생되는 결함을 줄여 품질을 향상하고, 개발하는데 드는 시간을 줄여 생산성을 향상함으로써 빠르게 변화하는 웹 환경에 능동적으로 대처하고자 한다.

Abstract In recent years, the progress of Network and Web technology is made more demanding and pervasive of Web-based information systems. At the same time, the complexity of web applications has grown significantly. As a result, developers, users, and other stake-holders have become increasingly concerned about the manner in which complex Web-based systems are created as well as the level of system performance, quality, and integrity. In this paper, we propose an efficient developing process and modeling environment of the web applications. Also, after modeling web application using the diagrams, web pages are extracted automatically. It is aimed at facilitating the design, implementation and maintenance of large, complex web applications and the reuse of previously gathered design experience.

1. 서 론

최근 네트워크와 인터넷의 급격한 성장과 더불어 기존의 많은 소프트웨어들이 웹 기반으로 그 모습을 변화해가고 있다. 이로 인해 웹 어플리케이션의 규모와 복잡성이 크게 증가되었으며, 많은 인력과 비용이 개발에 소요되고 있다. 따라서 최근 웹 어플리케이션 개발을 위한 표준 방법론을 모색하고, 생산성과 품질을 향상시키고자 하는 웹 공학(Web Engineering)에 대한 연구가 많이 진행되고 있다. 그러나 아직까지 웹 어플리케이션 개발을 위한 명확한 개발 단계의 정의나 단계별 역할 및 산출물 등에 대한 정의는 거의 없다. 대부분이 경험에 입각한 나름대로의 프로세스를 이용하여 개발하고 있는 실정이다.

소프트웨어를 개발함에 있어 분석과 설계는 그 프로젝트를 성공적으로 이끄는 중요한 요소이다. 분석과 설계단계에서 발생하는 결함은 소프트웨어 개발 전반에 치명적인 영향을 끼치게 된다. 이로 인해 지금까지 소프트웨어 개발에서 분석과 설계를 위한 다양한 기법들이 제시되었다. 하지만 대부분이 일반 어플리케이션을 위한 것이었고, 최근 몇몇 웹 어플리케이션을 위한 분석과 설계 기법들이 제시되고는 있으나 웹의 특성을 반영하기에는 부족함이 많다 [1].

따라서 본 논문에서는 대규모 웹 어플리케이션의

* 대구대학교 컴퓨터정보공학과

품질과 생산성을 향상시키기 위한 보다 효율적인 개발 환경을 제시한다. 첫째, 웹 어플리케이션 개발을 위한 프로세스를 정의하고, 분석과 설계를 도와주는 모델링 환경을 제시한다. 둘째, 모델링 결과를 바탕으로 웹 어플리케이션을 위한 소스코드를 생성한다. 이렇게 함으로써 분석과 설계 단계에서 발생되는 결함을 줄여 품질을 향상하고 개발하는데 드는 시간을 줄여 생산성을 향상하여 빠르게 변화하는 웹 환경에 능동적으로 대처하고자 한다.

2. 관련연구

웹 공학(web engineering)은 소프트웨어의 개발 예산 초과, 개발 일정의 지연, 저조한 생산성, 미흡한 품질 등과 같은 문제점을 해결하기 위하여 표준 방법론을 모색하고, 생산성과 품질을 향상시키기 위한 소프트웨어 공학을 웹 어플리케이션 개발에 적용시키려고 하는 것이다. 물론 소프트웨어 공학을 그대로 웹 어플리케이션 개발에 적용할 수는 없지만, 점차 웹 어플리케이션이 소프트웨어처럼 되어가고 있고, 현재 웹 어플리케이션 개발에 있어서 필요한 표준 개발 방법론이나 프로젝트 관리 등을 소프트웨어 공학에서 이용할 수 있다. 이처럼 웹 공학은 소프트웨어 공학의 영역에 부분적으로 포함되기는 하나, 그 밖의 시스템 분석/디자인, 요구사항 분석, 사용자 인터페이스 디자인, 정보 공학, 시뮬레이션 모델링, 프로젝트 관리 등이 포괄된 새로운 연구 분야로 간주하는 경향이 강하다.

이와 같이 웹 어플리케이션은 웹의 특수한 환경으로 인해 기존의 개발방법을 그대로 적용하기에는 부족함이 있다. 따라서 지금까지 웹 어플리케이션 개발 방법에 대한 연구가 계속해서 진행되고 있다. 초기에는 단지 웹사이트의 GUI에 대한 설계에 관심을 가졌으나, 최근에는 웹 어플리케이션의 논리적인 설계를 위한 시스템적 접근 및 네비게이션에 대한 설계에 많은 노력을 기울이고 있다. 웹 어플리케이션의 대표적인 개발 방법론인 HDM, OOHDM, RMM에 대해 살펴보면 다음과 같다.

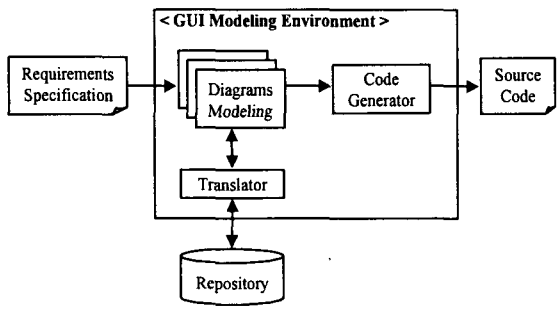
HDM(Hypermedia Design Model)[2]은 하이퍼미디어 설계에서 시스템적, 구조적 접근 방법론으로 OOHDM에 많은 영향을 주었으며, OOHDM (Object Oriented Hypermedia Design Model)[3]은 하이퍼미디어 어플리케이션 개발을 개념 설계, 네비게이션 설계, 추상 인터페이스 설계, 구현의 4단계로 나누어 수행하는 방법론으로 객체지향 프레임워크를 이용한다. 현재 웹, 멀티미디어 등 다양한 분야에 적용되고 있다. 그러나 이 방법론은 설계 단계에서의 모듈성과 재사용성은 제공하지만, 설계와 구현 사이의 매핑을 지원하지는 못한다. RMM(Relationship Management Method) [4]은 웹 기반 정보 시스템(WIS: Web-based Information Systems) 설계 및 개발자 지침을 제공하며 관리의 효율성을 제공하는 방법론이다. 이는 동적 데이터베이스를 유지하는데 소요되는 비용을 줄이고자 하는 데 목적이 있다. RMM은 웹 어플리케이션의 분석보다는 데이터베이스를 지원하는 동적 웹사이트를 생성하고 유지하기 위한 넓은 범위의 접근법으로 현재 웹 어플리케이션에 적용되는 스크립트, 애플릿, ActiveX와 같은 기술들을 포함하는 웹 어플리케이션을 관리하기에는 그 한계가 있다.

최근에는 위의 개발 방법론들을 바탕으로 웹 어플리케이션의 설계를 위한 다양한 환경들이 제시되고 있으나, 대부분이 UML의 단순 확장을 보인다. 웹 어플리케이션은 웹 환경만이 가지는 특성을 지니고 있기 때문에 일반 어플리케이션의 모델링을 위해 개발된 UML로 표현하는 데는 어려움이 따른다. 따라서 웹의 특성을 잘 반영하는 설계 환경의 구축이 무엇보다도 중요시되고 있다.

3. 웹 어플리케이션 개발 환경

3.1 구조

본 논문에서 웹 어플리케이션 개발을 위한 보다 효율적인 환경을 제시한다. <그림 1>은 제시된 웹 어플리케이션 개발 환경의 전체 구조를 보여준다. 이 개발 환경은 기능에 따라 Modeling, Translator, Generator의 세 부분으로 나누어 볼 수 있다.



<그림 1> 웹 어플리케이션 개발 환경

▪ Modeling

요구사항 명세서를 바탕으로 개발하고자 하는 웹 어플리케이션을 다이어그램(diagrams)과 그에 따른 표기법(notations)을 이용하여 설계한다. 모델링에 사용되는 다이어그램은 Architecture Design Diagram, Navigation Design Diagram, Page Detail Design Diagram이 있으며, 표기법에는 컴포넌트(components)와 커넥터(connectors)가 있다.

▪ Translator

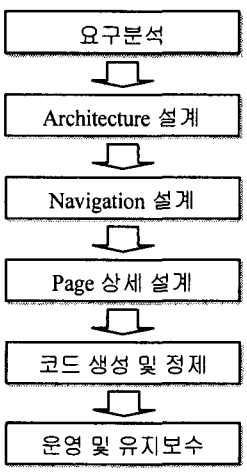
다이어그램들로 표현된 웹 어플리케이션의 모델링 결과를 모델링 언어로 변환한다. 모델링 언어는 시스템의 기능과 구조적 특성을 기반으로 하여 패턴을 정형화하고 구성 및 속성을 구체화한다. 또한 이 모델링 언어는 다시 다이어그램으로 변환될 수 있다.

▪ Generator

모델링 결과를 바탕으로 실제 웹 어플리케이션을 구성하는 웹 페이지를 자동으로 생성한다. 이는 각각의 다이어그램들을 구성하는 컴포넌트들의 속성과 그들 사이의 관계를 바탕으로 생성된다. 하지만 이렇게 자동으로 생성된 소스코드는 웹 어플리케이션을 완벽하게 표현하지는 못하므로 시스템 개발자에 의한 정제 과정을 거치게 된다.

3.2 프로세스

본 논문에서는 웹 어플리케이션 개발을 위한 프로세스 모델을 <그림 2>와 같이 6단계로 나누었다.



<그림 2> 웹 어플리케이션 개발 프로세스 모델

▪ 요구분석

웹 어플리케이션 개발의 목표와 기능을 결정하는 단계이다. 사용자의 요구사항을 개발자 측면에서 정확하게 파악하여 각각의 문제를 이해하기 쉽고 원활하게 접근한다.

▪ Architecture 설계

요구분석 결과를 바탕으로 해당 웹 어플리케이션에 가장 적합한 구조를 결정하는 단계이다. 어플리케이션을 기능별로 구분하여 각 기능에 따른 상하 · 수평관계를 결정한다.

▪ Navigation 설계

웹 어플리케이션을 구성하는 페이지들 사이의 링크관계를 결정하는 단계이다. 페이지들 사이의 연결관계 및 데이터의 이동에 대한 개략적인 모습을 보여준다.

▪ Page 상세 설계

웹 어플리케이션을 구성하는 각 페이지별 상세설계 단계이다. 각 페이지의 기능에 따라 컴포넌트와 커넥터의 관계를 결정하고 그들의 속성을 정의한다.

▪ 코드 생성 및 정제

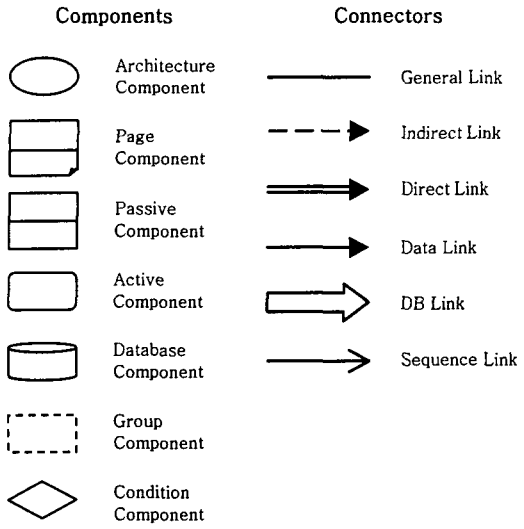
지금까지 단계에서의 결과를 이용하여 웹 어플리케이션을 구성하는 각 페이지별 소스코드를 생성한다. 또한 자동으로 생성된 소스코드를 개발자가 정제하는 단계이다.

▪ 운영 및 유지보수

웹 어플리케이션을 운용하고 이에 따른 결함이나 오류를 수정하며, 사용자의 새로운 요구에 따라 기능을 추가, 수정, 삭제하는 단계이다.

3.3 표기법

웹 어플리케이션을 모델링하기 위해 사용하는 표기법은 컴포넌트와 커넥터로 구분한다. 컴포넌트는 시스템을 구성하는 기능을 나타내며, 커넥터는 컴포넌트 사이의 상호작용을 나타낸다. <그림 3>은 웹 어플리케이션 모델링에 사용되는 표기법들을 보여준다.



<그림 3> 웹 어플리케이션 모델링 표기법

Architecture Component는 웹 어플리케이션의 구조를 표현할 때 이용하며, 어플리케이션의 기능을 크게 구분하여 나타내고, Page Component는 웹 어플리케이션을 구성하고 있는 웹 페이지들을 표현한다. 또한 Passive Component와 Active Component는 페이지를 구성하는 정적 및 동적인 기능을 표현한다. Database Component는 데이터를 저장하는 장소를 말하며, Group Component는 여러 컴포넌트를 하나의 특성으로 묶어서 표현한다. Condition Component는 조건을 명시하기 위해 사용된다.

General Link는 특별한 의미를 지니지 않고 단순히 두 컴포넌트 사이의 관계 유무만을 나타낸다. Indirect Link와 Data Link는 사용자의 클릭에 의한 링크를 말하며, 아무런 데이터의 이동이 없으면 Indirect Link로 나타내고 데이터의 이동이 있으면 Data Link로 표현한다. Direct Link는 프로그램 상에서의 자동 링크를 나타낸다. DB Link는 데이터베이스와의 데이터 전송을 나타내고 Sequence Link는 단지 순서를 표현한다.

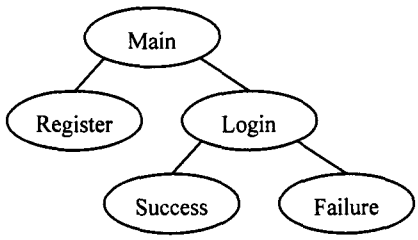
3.4 다이어그램

본 논문에서 제시한 모델링 환경에서는 다음과 같은 세 개의 다이어그램이 사용된다.

- Architecture Design Diagram
- Navigation Design Diagram
- Page Detail Design Diagram

1) Architecture Design Diagram

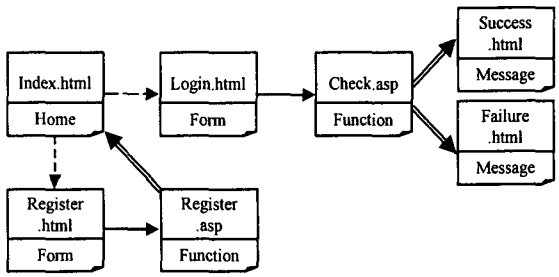
Architecture Design Diagram은 개발하고자 하는 웹 어플리케이션의 구조를 나타낸다. 잘 정의된 구조는 대규모 소프트웨어의 개발에 있어 각 부분별 개발 결과를 통합하는데 유용하다. 이 다이어그램에서는 Architecture Component, General Link 표기법을 사용하여 모델링 하고, 각 컴포넌트와 커넥터의 속성도 정의한다. <그림 4>는 Architecture Design Diagram의 한 예를 보여준다.



<그림 4> Architecture Design Diagram

2) Navigation Design Diagram

Navigation Design Diagram은 웹 어플리케이션을 구성하는 웹 페이지들 사이의 상호관계를 나타낸다. 다시 말해서, 각각의 웹 페이지들 사이의 링크관계를 정의한다. 이 다이어그램에서는 Page Component, Indirect Link, Direct Link, Data Link 표기법을 사용하여 모델링 하고, 각 컴포넌트와 커넥터의 속성도 정의한다. <그림 5>는 Navigation Design Diagram의 한 예를 보여준다.

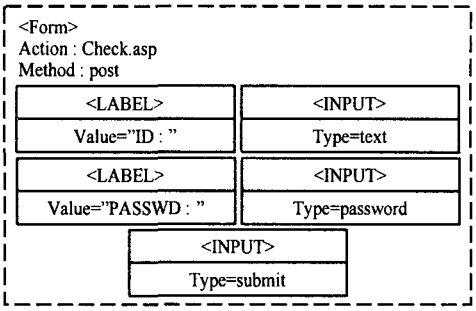


<그림 5> Navigation Design Diagram

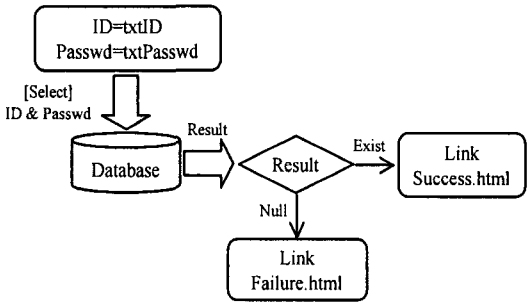
3) Page Detail Design Diagram

Page Detail Design Diagram은 웹 어플리케이션을 구성하는 각 페이지를 상세하게 설계한다. 웹 페이지는 정적 페이지와 동적 페이지 그리고 이 둘이 섞여 있는 페이지로 나누어 볼 수 있다. 정적 페이

지는 디자인 중심 페이지를 말하고 동적 페이지는 기능 중심 페이지를 말한다. 본 다이어그램에서는 Passive Component, Active Component, Database Component, Group Component, Condition Component, DB Link, Sequence Link 표기법을 사용하여 모델링 하고, 각 컴포넌트와 커넥터의 속성도 정의한다. <그림 6>은 Page Detail Design Diagram의 예를 보여주며, (a)는 디자인 중심 Page Detail Design Diagram이고 (b)는 기능 중심 Page Detail Design Diagram이다.



(a) 디자인 중심



(b) 기능 중심

<그림 6> Page Detail Design Diagram

3.5 모델링 언어

모델링 언어는 다이어그램으로 표현된 어플리케이션을 언어를 이용하여 표현한 것으로, 이는 다이어그램을 통한 모델링 결과를 바탕으로 자동으로 변환된다. 모델링 언어는 시스템의 기능 및 특성을 정형화하고 구성 및 속성을 구체화한다. 다음 <그림 7>은 임의의 Architecture Design Diagram을 모델링 언어로 표현한 것이다.

```
<Diagram Type="ArchitectureDesignDiagram" Name="archBoard">
    <Component Type="ArchitectureComponent" Order="1" x="70" y="10"
    width="50" height="30">
        Name = "ac1"
        Caption = "List"
        Comment = "Start Page & List"
    </Component>
    <Component Type="ArchitectureComponent" Order="2" x="40" y="100"
    width="50" height="30">
        Name = "ac2"
        Caption = "Read"
        Comment = "Read"
    </Component>
    <Component Type="ArchitectureComponent" Order="3" x="100"
    y="100" width="50" height="30">
        Name = "ac3"
        Caption = "Write"
        Comment = "Write"
    </Component>
    <Connector Type="GeneralConnector" Order="4" x1="85" y1="40"
    x2="105" y2="40'>
        Name = "cnt1"
        Relation = "ac1→ac2"
    </Connector>
    <Connector Type="GeneralConnector" Order="5" x1="65" y1="100"
    x2="125" y2="100">
        Name = "cnt2"
        Relation = "ac1→ac3"
    </Connector>
</Diagram>
```

<그림 7> 모델링 언어

3.6 웹 페이지 생성

웹 어플리케이션은 여러 개의 페이지들로 구성된다. 본 논문에서는 설계 단계에서의 산출물인 다이어그램들을 이용하여 자동으로 웹 페이지를 생성한다. 이는 각 다이어그램을 구성하는 컴포넌트들의 속성과 그들 사이의 관계를 바탕으로 생성된다. 다음 <그림 8>은 <그림 6>의 (a)에서 모델링한 결과를 바탕으로 자동으로 생성된 웹 페이지 소스코드를 보여준다.

```
<html>
<head>
</head>
<body>
    <form action="Check.asp" method="post" name="frmLogin">
    <center>
    <br>
    ID :
    <input type="text" name="txtID" maxlength="10" size="10">
    <br>
    PASSWD :
    <input type="password" name="txtPasswd" maxlength="10" size="10">
    <br>
    <input type="submit" name="smOK" value="등 록">
    </center>
    </form>
</body>
</html>
```

<그림 8> 소스코드 생성

4. 결 론

본 논문에서는 대규모 웹 어플리케이션의 개발을 위한 프로세스의 정의 및 설계 환경을 제시한다. 개발 프로세스는 6단계로 구분하였으며, 설계 환경은 Modeling, Translator, Generator의 세 가지 모듈로 구성된다. 모델링 단계에서는 Architecture Design Diagram, Navigation Design Diagram, Page Detail Design Diagram을 작성하고, 이를 바탕으로 모델링 언어와 실제 웹 어플리케이션을 구성하는 웹 페이지를 생성한다.

따라서 분석과 설계 단계에서 발생하는 결함과 개발하는데 드는 시간을 줄일 수 있고, 이로 인해 웹 어플리케이션의 품질과 생산성을 향상하는 효과를 가져온다.

참고문헌

[1] Jyhjong Lin, et al., Object-Oriented Analysis and Design of Web-Based Information Systems, Proceedings of the 8th Annual IEEE International Conference and Workshop on the ECBS, 2001, pp. 68-75.

[2] Franca Garzotto, et al., HDM - A Model-Based Approach to Hypertext Application Design, ACM Transactions on Information Systems, vol.11, no. 1, Jan. 1993, pp. 1-26.

[3] Daniel Schwabe, et al., Systematic Hypermedia Application Design with OOHDM, Proceedings of the ACM International Conference on Hypertext, March 1996, pp. 116-128.

[4] Tomas Isakowitz, et al., RMM: A Methodology for Structured Hypermedia Design, CACM, vol. 38, no. 8, August 1995, pp. 34-44.