Journal of Digital Contents Society Vol. 8 No. 1 Mar. 2007(pp. 93-106)

웹 응용의 적응하는 문맥 기반 컨텐츠 항해 모델링

이병정*, 홍지원**

요 약

웹 응용이 급격히 증가하고 있고, 그 구조 또한 매우 복잡해지고 있다. 그러나 사용자가 복잡한 웹 응용 컨텐츠를 항해할 때, 현재 위치를 파악하지 못하거나 또는 원하는 정보를 얻지 못하는 경우가 잦다. 따라서 웹 응용 컨텐츠 항해를 모델링하기 위한 체계적인 접근 방법이 필요하다. 본 연구에서는 웹 응용의 적응하는 문맥-기반 컨텐츠 항해 모델링 프레임워크를 제안한다. 본 프레임워크에서는 항해 분석, 항해 설계, 그리고 항해 실현 등의 활동을 수행한다. 항해 분석 활동에서는 항해 관점에 초점을 두고 유즈케이스를 사용하여 영역 분석을 한다. 항해 설계 활동은 항해 정보 모델, 프로파일, 그리고 항해 인터페이스 모델을 생성한다. 마지막으로, 항해 실현 활동에서는 웹 응용 메타 모델을 정의하고 웹 페이지 항해 모델과 컴포넌트 항해 모델을 정의한다. 또한 본 연구에서는 항해 모델의 유효성을 검사하기 위한 정형적인 정의와 규칙을 제시한다.

Modeling Adaptive Context-Based Contents Navigation of Web Applications

Byung-Jeong Lee*, Ji-Won Hong**

Abstract

Web Applications are rapidly increasing and the structure becomes very complicated. However, when users explore such complex Web applications, they cannot often grasp the current location and get the information that they want. Therefore, a novel approach to model the navigation of Web application contents is required. In this study, a framework has been presented for modeling adaptive context-based contents navigation of Web applications. The framework performs activities including navigation analysis, navigation design, and navigation realization. First, in navigation analysis domain is analyzed by using use case, focusing on navigation. Next, in navigation design three models have been produced: a navigation information model, a profile, and a navigation interface model. Finally, in navigation realization a WebPage navigation model and a component navigation model have been produced. In this work, several formal definitions and rules for checking validity of navigation model have also been provided.

Keywords : 웹응용, 적응형 컨텐츠 항해 모델링, UML, 항해모델 검증,

1. 서론

웹의 발전으로 다양하고 폭 넓은 사용자를 대상으로 여러 목적의 웹 응용이 개발되고 있다. 무선 랜, 모바일 이동통신, 유비쿼터스 등의 기술 발전은 현실의 일상 곳곳에 웹을 기반으로 한 생활이 이루어지도록 하고 있으며 그 영역은 더 넓어질 것으로 보인다. 웹은 다수의 광범위한 사용자들의 다양한 요구를 처리하는 방향으로 발전하면서 사람들의 일상의 모습을 바꿔 놓고 있다. 그 예로서 온라인 강의는 사용자에게 거리와 시간의 제약에서 벗어나 원하는 시간에 자신에게 맞는 강좌를 수강할 수 있게 해 주고 있다. 그러나 다른 측면에서는 웹의 사용자와 정보의 급속한 증가로 특정 사용자는 자신에게 맞는 정

※ 제일저자(First Author) : 이병정
※ 교신저자(Corresponding Author) : 이병정
접수일자:2007년02월18일, 심사완료:2007년03월14일
* 서울시립대학교 컴퓨터과학부
bjlee@uos.ac.kr
** 삼성 SDS
▣ 이 논문은 2004년도 한국학술진흥재단의 지원에 의하여 연구되었음.(KRF-2004-003-D00349)

보를 얻기 위해 오랜 시간과 노력을 들이게 되었음을 의미한다. 따라서 웹 응용에서 사용자의 효과적인 항해를 위해 사용자의 특성을 인식하고 이에 적응하도록 웹 응용의 항해 설계가 요구된다. 이는 정적인 내용에 단일한 인터페이스를 가정한 기존의 전통적인 하이퍼미디어 모델링으로는 한계가 있음을 의미한다.

웹은 연결된 많은 하이퍼텍스트 링크로 이루어져있으며, 이로 인해 비선형적인 항해를 특징으로 갖는다. 웹 응용 내에서 사용자는 계층 구조의 한 가지에서 다른 가지로 수평적 이동과 한 가지에서 임의의 레벨로의 이동이 가능하다. 이러한 웹의 특성으로 인하여 문맥지표를 제공하지 않는 웹 응용에서 사용자는 원하는 정보에 접근하기 위해 오랜 시간을 소요하게 되며, 찾지 못하는 경우도 발생한다. 이 과정에서 사용자는 인지 과부하 현상으로 인한 혼란을 겪게 되며 이것은 해당 웹 응용을 기피하는 원인이 된다 [1]. 따라서 웹 응용 항해를 모델링하기 위한 체계적인 방법이 필요하며, 이러한 항해 모델은 웹 응용 구조 문서화에 유용하며, 또한 항해를 용이하게 한다 [2]. 웹 응용 항해 설계 과정의 초점은 이러한 문제를 해결하는 것이며, 해결 방법 중 하나는 문맥 지표를 제공하는 것이다. 문맥 지표를 제공하는 웹 응용은 사용자에게 웹 응용을 구성하고 있는 정보의 계층적 구조를 명확하고 일관되게 보여줄 수 있고 그 구조 안에서 사용자의 현재 위치를 알 수 있게 한다. 이전 연구에서는 항해를 하이퍼미디어 설계의 일부로 간주하거나 [2][3], 동적 컨텐츠로 구성된 웹 응용 항해에 대한 설명이 부족하였다 [4][5].

본 논문에서는 웹 항해 모델링을 위하여 웹 응용의 문맥 지표를 정의하고 동적 컨텐츠를 포함한 적응하는 웹 응용 설계 개념을 정의한다. 그리고 적응하는 웹 응용 항해 모델링 프레임워크를 제안한다. 프레임워크에서는 항해 분석, 항해 설계, 그리고 항해 실현 활동을 수행한다. 기본적으로 프레임워크의 활동들은 반복적 점진적으로 수행되며 각 활동에서 하나 이상의 UML[6]을 확장한 모델들이 산출물로 새로 만들어지거나 갱신된다. 또한 항해 모델을 검사하기 위한 정형적인 정의와 규칙을 제시한다.

본 논문의 구성은 다음과 같다. 2장에서 관련 연구를 기술하고, 3장에서 웹 항해 모델링 개요를 기술한다. 4장에서 적응하는 항해 모델링 프레임워크를 기술하고, 5장에서 결론을 맺는다.

2. 관련 연구

2.1 UML 확장 메커니즘

복잡한 소프트웨어 시스템의 시각적인 모델링을 위한 UML은 소프트웨어의 명세뿐 아니라 소프트웨어 중심의 시스템 모델링에 관한 범위까지 확장되고 통합된 표기법을 제공하여 개발자들이 문제 해결에 집중하는 것을 돕는다. 하지만, UML을 다른 분야에 적용할 때에는 모델링되는 요소들의 특징에 대한 주위 깊은 분석이 선행되어야 한다. 웹 응용의 중요한 특징은 앞서 설명된 것처럼 서로 다른 문서나 정보의 조각들 간의 비선형적인 항해가 가능하다는 것이다.

설계 과정에 UML을 사용하면 개발자들이 문제 해결에 더 집중할 수 있도록 하는 장점이 있지만 UML 표준만으로 특정한 도메인이나 아키텍처를 모델링 할 때 충분하지 않다는 한계가 있다. 이를 보완하기 위한 방법이 스테레오타입(stereotype), 태그값(tagged value), 그리고 제한조건(constraints) 등의 확장 메커니즘을 사용하여 UML을 특정 영역에 알맞도록 확장 정의하여 사용할 수 있다 <표 1>. 따라서 본 연구에서도 UML 확장 메커니즘을 사용하여 메타모델(meta-model) 레벨에서 웹 응용 모델을 기술한다.

<표 1> UML 확장 메커니즘 방법

스테레오타입	모델링 요소에 대한 새로운 의미를 정의
태그값	모델링 요소에 허용 값을 붙이며 모델링 요소와 쌍을 이룸
제한조건	모델을 정의하는 규칙, 자유형식이나 OCL로 표현

2.2 웹 응용 모델링

웹 응용 또는 하이퍼미디어 응용의 효과적인 설계를 위한 많은 연구들이 이루어졌다. 객체지향 하이퍼미디어 개발은 개념 설계, 항해 설계, 추상 인터페이스 설계, 그리고 구현 단계로 구성된다 [4]. 항해 공간의 기본 요소인 항해 문맥은

6가지 방법으로 정의된다: 단순 클래스 기반, 클래스 기반 그룹, 링크 기반, 링크 기반 그룹, 번호 나열, 동적 항해 문맥. 그러나 이 방법은 웹 응용의 동적 컨텐츠 항해에는 적당하지 않다.

UML 프로파일 기반 하이퍼미디어 설계 방법은 개념 (conceptual), 항해 (navigational), 표현 (presentational) 설계 등의 세 단계로 구성된다 [7]. 개념 설계 단계에서는 유즈케이스를 사용한 기능적 요구사항을 기반으로 개념 모델이 작성된다. 항해 설계 단계에서는 개념 모델에 기반하여 항해 공간 모델과 항해 구조 모델이 생성된다. 표현 설계 단계의 모델은 항해 설계 단계로부터 유도되며 HTML 프레임으로 구현된다. 그러나 표현 모델에서 프레임은 두 개의 프레임으로 나누어진다. 오른쪽 프레임은 항해 클래스 또는 색인 클래스를 표현하고, 왼쪽 프레임은 항해 트리를 표현한다. 따라서 브라우징할 때 사용자 활동에 제약을 준다. 또한 이 방법은 동적 컨텐츠에 대한 항해를 다루지 않는다.

UML의 표준 제정과 이를 이용한 설계 방법에 관한 연구들이 수행되었다. [8]은 웹 기반 시스템의 설계에 UML을 이용한 모습을 보여주고, [7]는 UML 확장 메커니즘을 이용하여 웹 응용의 모델링에 필요한 요소들을 새롭게 정의하였으며, [3]는 UML을 메타모델 수준에서 확장하는 시도를 하였다. 이들 연구가 하이퍼미디어 특성을 가진 웹 응용 설계를 주제로 했던 반면, [9]는 특히 웹 응용의 항해 설계에 초점을 두었다. [5]는 유즈케이스를 웹 응용의 항해 요구분석을 위해 사용하는 방법을 보여준다. 적응하는 웹 응용에 대한 연구 [10][11]는 다양한 웹 영역의 문제를 인식하고, 각 영역의 사용자 특성에 적응하는 방법을 찾는다. 특히 [11]에서는 적응성의 개념을 모바일 환경에 적용하는 연구를 하였다.

웹 응용에서 항해의 문제를 올바른 인터페이스 설계를 통해 해결하고자 하는 연구들도 수행되었다. [12]은 사용자 항해를 도와주는 기능들을 분류하여 각각의 효과에 대하여 밝혔고, [2]는 UML을 사용한 인터페이스의 설계의 예를 보였다. [13]은 사용자의 인지작용을 도울 수 있는 항해 보조 기능들에 대해 기술하였고, [14]는 항해의 흐름을 도울 수 있는 항해 보조 기능들에 대해 정리하였다. [15]는 웹 응용을 이루는 웹 페이지들의 연결 구조를 분석하여 공통적인 구조와 그 특징에 대해 밝히고 있다. [16]은 웹 페이지 내의 항해를 UML 상태도로 모델링했으나, 웹 페이지 내의 모델링 요소를 정의하지 않았고, 항해 설계 단계를 체계화하지 못하고 있다.

[17]에서는 문맥분석, 구조 모델, 항해 모델, 행해 구조 모델, 표현 모델을 제안한다. 문맥 분석 단계에서는 대상 시스템의 요구사항을 분석하고, 구조 모델에서는 개념적 영역 모델을 구축한다. 항해 모델은 웹 응용에서 방문되는 객체들을 명세하고, 항해 구조 모델에서는 메뉴, 질의, 색인 등의 객체를 스테레오타입을 사용하여 영역 객체와 함께 표시한다. 항해 설계 모델들은 기술하고 있으나, 항해 문맥/요구사항으로부터 설계 모델, 표현 모델들 사이의 관계는 구체적으로 기술되어 있지 않다.

웹 응용과 CASE 지원 설계를 위하여 UML 메타모델을 확장한 UWE 메타모델을 제안하였다 [18]. UWE에서는 UML 메타모델 요소를 변경하지 않고 상속하여 보수적으로 확장하며, UWE 메타모델 개념이 UML 메타모델로 매핑되어 도구에 의하여 검사된다. 반면 UWE 프로파일은 추가된 구조, 태그 값, 관계들로 인해 너무 복잡하다.

3. 웹 응용 항해 모델링

3.1 웹 응용의 문맥 지표

웹 응용에서 사용자는 하이퍼텍스트를 따라 원하는 정보를 찾아가는 과정에서 되돌아가기를 반복하게 되고, 이 과정에서 이전에 지나온 경로들을 반복적으로 기억하게 되면서 인지과부하를 겪게 된다 [1]. 이것은 웹 응용 항해의 큰 문제 중 하나로 이를 위한 많은 기술적인 보완책이 고안되어 왔다. 대부분의 웹 응용들은 고정적인 위치에 메뉴를 만들어 두어 사용자가 항해의 어느 지점에서라도 각 메뉴의 초기지점으로 돌아갈 수 있게 한다. 웹 브라우저의 "뒤로" 버튼이나 "열어본 페이지 목록" 기능도 비슷한 기능으로 이해할 수 있는데 이러한 노력으로 사용자의 인지과부하를 줄이는 효과가 있으며 혼란도 감소시킨다 [1]. 그러나 이런 보완적인 기능의 제공보다 개발단계에서 체계적인 항해 설계과정을

통해 사용자에게 지속적인 문맥 지표를 제공하는 것이 더 중요하다.

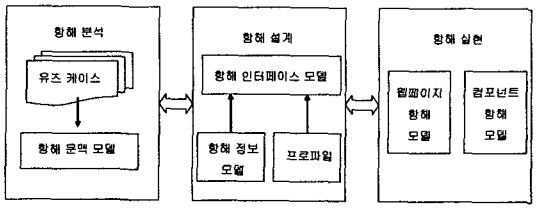
문맥지표는 웹 응용의 사용자들이 항해 중에 길을 잃지 않도록 하는 정보를 의미한다. 특히 사용자가 항해 중에 특별한 노력을 들이지 않더라도 그들의 현재 위치를 쉽게 알 수 있도록 하는 정보들을 구성하는 것을 뜻한다. 웹 응용이 담고 있는 주요 개념에 링크를 걸어 마우스가 링크의 위로 올라왔을 때 팝업 창을 띄워 그 용어에 대한 설명을 하는 방법은 사용자의 항해의 흐름을 방해하지 않고, 문맥 지표를 제공하는 방법 중 하나이다. 또 메뉴의 하부로 항해할 때 전체 카테고리 중 어느 위치인지를 화면의 일부에 동적으로 표시해 주는 것도 같은 목적을 위해서다.

3.2 적응하는 웹 응용 설계

앞에서 설명한 문맥지표는 사용자 중심으로 설계되어 사용자가 이해하고 알아보기 쉬워야 한다. 따라서 궁극적으로는 사용자에 대한 정보 - 개인정보 및 이전까지의 항해 로그 - 를 바탕으로 사용자의 항해 목적을 분석, 예측하고 이에 알맞은 항해 경로를 구성해야 할 것이다. 웹 응용의 항해 문제는 한 링크를 거치면서 기하급수적으로 증가하는 하이퍼링크의 구조적 특징을 고려해야 하며, 체계적인 항해 설계 과정을 통해 해결할 수 있다. 또한 사용자 환경에 대한 정보를 수집하여 적절한 정보의 형태를 제공하는 것이 필요하다. 예를 들면 사용자 시스템이 개인용 컴퓨터인지 또는 PDA인지, 네트워크 용량이 충분한지 또는 그렇지 않은지 등에 따라 처리 또는 전송에 적합한 정보로 보내야 사용자가 효율적인 웹 응용 접근이 가능하다.

4. 적응하는 항해 모델링 프레임워크

웹 응용의 항해 설계에 대한 연구[4][7][8]들은 웹 응용이 담고 있는 정보를 효과적으로 표현하는 것에 초점을 두고 있다. 이와 비교하여 적응하는 웹 응용 설계는 특정 사용자마다 적합한 항해 정보를 문맥지표로 제공하는 것을 목적으로 한다. 사용자에 대해 적응적으로 반응하는 항해 구조는 특정 사용자의 목적에 알맞은 항해구조를 동적으로 생성해 줄 수 있다. 이를 위한 항해 모델링의 과정이 (그림 1)에 제시되어 있다.



(그림 1) 적응하는 항해 모델링 프레임워크

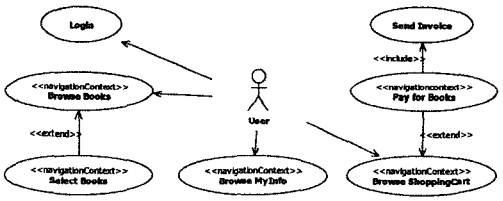
(그림 1)은 웹 응용의 적응하는 항해 모델링 단계를 보여준다. 항해 분석 단계에서는 유즈케이스 분석을 통해 웹 응용의 주요 항해 문맥을 찾아낸다. 항해 문맥 모델은 웹 응용의 목적에 맞도록 사용자의 항해를 유도하기 위한 항해 설계의 기준으로 사용된다.

항해 설계 단계에서는 항해 분석을 통해 얻은 정보를 바탕으로 구체적인 설계과정이 이루어진다. 첫 번째로 항해 정보 모델에서는 웹 응용을 구성하는 정보들이 항해 노드로서 구분되고 주요한 속성들이 명시된다. 예를 들어 온라인서점에서 책은 중요한 항해노드로 구분되며 제목, 저자, 출판사, 가격 등이 주요 속성으로 명시될 수 있다. 두 번째로 프로파일은 적응하는 항해를 위해 필요한 부분으로 웹 응용의 동적인 컨텐츠와 항해경로 구성의 기본으로 사용된다. 사용자의 이전 항해기록, 개인정보, 선호도, 사용자의 웹 사용 환경정보 등이 프로파일에 포함된다. 항해 인터페이스 모델에서는 정보 모델에서 정의된 항해 노드들과 프로파일에 근거해서 적응 항해 경로가 어떻게 화면에서 구성될 지 나타낸다. 대부분 웹 응용은 일관된 위치에 메뉴와 검색 기능을 두어 효과적으로 사용자의 항해를 돕고 있다. 내용의 구성과 위치뿐만 아니라 사용자의 인지적인 측면을 고려한 인터페이스 일관성 또한 항해 인터페이스 모델에서 고려해야 할 부분이다. 그래서 연관된 정보를 모으고, 필요시 요구하면 스크린에 정보를 표시하는 것이 인지과부하를 줄이는데 도움이 되므로 [19], 관련 정보를 스크린에 점진적으로 표시하는 방법을 기술함으로써 항해 인터페이스 모델을 정의하는 것이 요구된다.

항해 실현 단계는 웹 페이지 항해 모델과 컴포넌트 항해 모델을 포함한다. 웹 페이지란 화면에 보여지는 한 화면을 뜻하는 것으로, 한 의미 단위 내용이 여러 웹 페이지로 구성되는 것이 일반적이다. 예를 들어 결제에 대한 부분은 보안상의 이유와 정확성의 필요로 5~7 단계의 웹 페이지를 통해 이루어진다. 이러한 경우 밀접하게 연결된 웹 페이지들이 어떻게 연결되어 항해 되는가에 대한 모델링이 필요하다. 내용에 따라서는 한 화면 내에서 항해가 이루어지는 경우도 있다. 이 경우 한 화면에 포함된 컴포넌트들이 항해 노드가 된다.

4.1 항해 분석

4.1.1 항해 문맥 모델



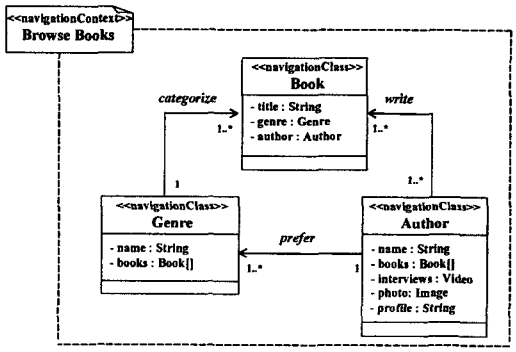
(그림 2) 온라인 서점의 항해 문맥 모델

일반적으로 유즈케이스 모델은 시스템에 대한 사용자의 기능적 요구사항을 분석하는데 사용되나 사용자의 역할에 따른 항해 분석에도 사용될 수 있다 [5]. 웹 응용 항해 모델을 사용하여 분석된 사용자 항해경로는 웹 응용 전반에 계층적, 수평적 이동을 위한 글로벌 항해 시스템으로 구현되어 웹 응용 전체를 통해 일관된 문맥지표로 사용된다. 본 연구에서는 <<navigationContext>> 스테레오타입을 정의하여 기존의 유즈케이스를 항해중심 문맥으로 확장하여 사용하였다 (그림 2). 본 연구에 사용된 스테레오 타입에 대한 정의는 <표 2> 웹 응용 항해모델을 위한 스테레오타입에 정리하였다.

(그림 2)는 유즈케이스 분석을 통한 온라인 서점의 항해 문맥 모델을 나타낸 것이다. 이 모델에서 분석된 항해 문맥에 대하여 글로벌 항해 시스템을 구성하면 웹 응용 전체에 걸쳐 일관되게 보이며, 웹 응용 내의 어느 위치에서든지 접근 가능하다. 글로벌 항해 시스템은 메인 페이지 상단에 아이콘을 이용한 홈(Home)과 메뉴 항해바(Bar) 등을 통해 사이트 전반에 계층적, 수평적 이동을 가능하게 한다 [12]. (그림 2)에서 책 검색, 장바구니 보기, 로그인은 글로벌 항해 시스템으로 구성될 수 있고, 지불이나 개인정보 수정은 권한에 따라 보여야 하는 메뉴로 분류되어 접근될 수 있다.

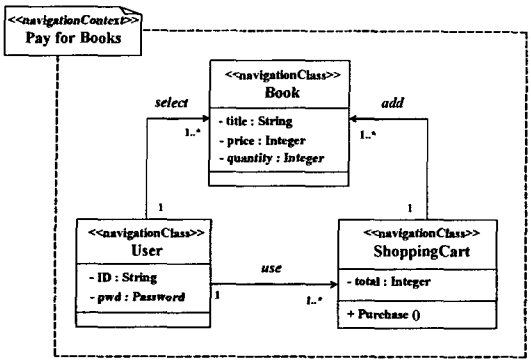
4.2 항해 설계

4.2.1 항해 정보 모델



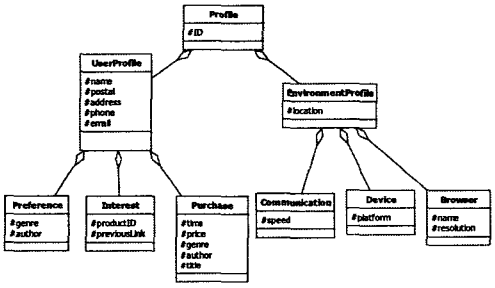
(그림 3) 책 찾기 문맥에서 항해 정보 모델

항해 정보 모델은 항해 노드가 될 개념들을 구분해내고, 보여 질 필요가 있거나 관리되어야 할 속성들을 추출한 후에 그들 간의 연결 관계와 계층 관계를 표현한 것이다. 이 과정은 [4]의 개념 설계 과정과 유사하나 본 연구에서는 특정한 문맥에서의 정보 설계 과정에 초점을 둔다. (그림 3)은 온라인 서점에서 책 찾기 문맥에서 사용되는 항해 클래스들과 그들의 관계를 나타낸다. (그림 3)에서 보면 하나의 의미 단위가 세부 문맥에 따라 다른 방법으로 탐색되는 것을 알 수 있다. 즉, 저자 조앤 롤링을 찾은 후에, "해리 포터" 책을 검색할 수 있고, 또한 환타지 장르 책을 검색하다가 "해리 포터" 책을 발견할 수 있다. (그림 4)는 책 구입 문맥에서 사용되는 책, 사용자, 장바구니 항해 클래스와 이들의 관계를 나타내고 있다.



(그림 4) 책 구입 문맥의 항해 정보 모델

4.2.2 프로파일

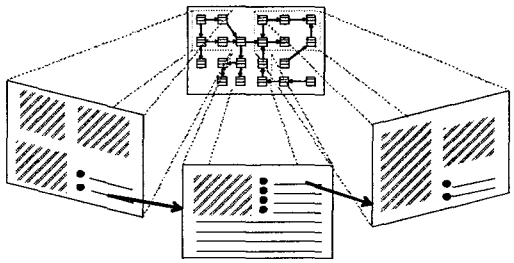


(그림 5) 사용자 프로파일 메타모델

항해 설계 단계에서 정의되는 프로파일은 웹 응용의 개발 목적에 맞는 사용자들에 관한 정보를 담을 수 있다. 프로파일은 크게 사용자에 대한 정보와 사용자의 환경에 대한 정보로 나누어지며 사용자의 선호도, 관심, 과거 기록 등을 근거로 컨텐츠를 구성하고, 사용자 환경 정보에 맞게 인터페이스, 화면분할, 정보의 종류/형태 등을 적합하게 구성할 수 있다.

(그림 5)는 UML로 작성된 온라인 서점 사용자에 대한 프로파일을 나타내고 있다. 사용자 프로파일(UserProfile)은 항해에 영향을 미치는 요소들 - 선호(preference), 흥미(interest), 구매기록(purchase) - 중 선호는 사용자가 등록한 정보이며, 흥미는 현재 웹 응용에 항해중인 사용자의 링크들에 대한 정보로 사용자의 항해 동안 계속 갱신된다. 구매기록을 기준으로 사용자를 여러 타입으로 나누어 놓고 이후의 이벤트, 할인행사 등에 유용한 기준으로 사용할 수 있다. 사용자의 환경에 대한 정보도 이와 마찬가지 방법으로 모델링 할 수 있다. 예를 들면 사용자가 온라인 서점을 인터넷으로 방문하는 통신(communication) 방법, 사용하는 PC, PDA, TV 등의 장치(device), 그리고 사용하는 인터넷 익스플로러, 네비게이터, 모질라 등의 브라우저(browser) 정보를 프로파일로 구성한다.

4.2.3 항해 인터페이스 모델

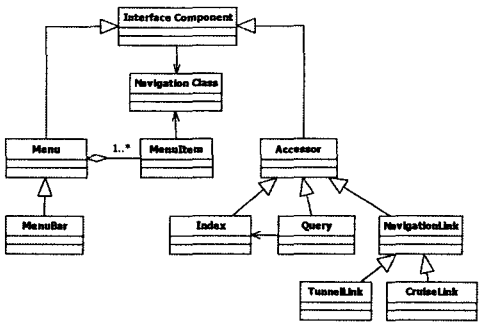


(그림 6) 문맥지표와 인터페이스 설계

항해 정보 모델링 과정에서 구별되어진 개념들과 그 속성들은 항해 문맥에 따라 선택되어 인터페이스를 구성하게 된다 (그림 6). 이렇게 항해 문맥을 고려해 설계된 인터페이스는 사용자의 항해를 향상시킬 수 있다. 사용자의 항해의 목적에 알맞은 문맥 지표란 현재 사용자에 대한 개인정보와 이전 항해의 로그 등 사용자의 항해 목적을 파악할 수 있는 자료에 근거하여 제공되는 정보를 의미하며 이를 통해 사용자의 항해 목적을 예측해 이에 알맞은 항해 경로를 구성해 줄 수 있다. 예를 들어 교육용 웹 응용에서는 사용자의 현재 수준에 맞는 내용에 대한 링크만을 선별해 보여줄 수 있다. 가상 박물관에서는 사용자의 관심 주제에 따라 링크를 추천하여 보다 효과적인 관람을 유도할 수 있다. 전자 상거래 중에 사용자가 검색하는 상품을 인식하고 그와 관련된 제품에 대한 추천링크를 구성할 수도 있다.

웹 응용이 포함하는 내용이 방대해지고 복잡한 계층구조를 갖게 되면서 질의기능과 항해 링크 기능을 제공하여 사용자의 효과적인 항해를 보조한다. 특히 메뉴는 글로벌 시스템의 대표적인 요소로 사용자가 웹 응용의 어느 곳에서도 일관된 위치에 제공되며, 직접 접근이 가능하다.

앞에서 설명하였듯이 (그림 7)의 글로벌 항해 시스템은 항해 문맥 모델을 통하여 만들어 지며, 메뉴 항목(MenuItem)은 항해 정보 모델에서 찾아낸 중요 항해 노드들로 구성된다.



(그림 7) 글로벌 항해 메타모델

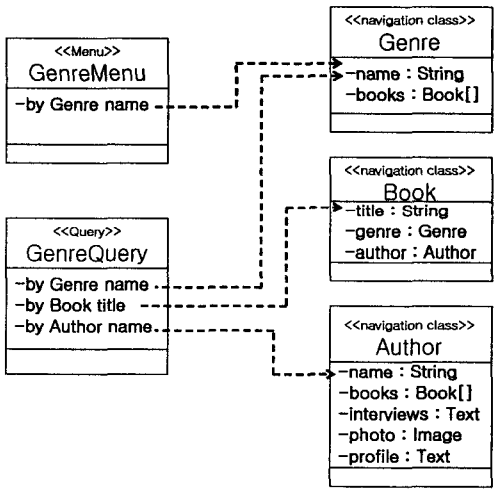
(그림 8)는 (그림 7)의 인터페이스 클래스와 항해 설계 단계에서 정의한 항해 클래스와의 연관 관계를 나타낸다. 인터페이스 클래스는 항해 클래스 속성들이 세부 문맥에 맞게 구성되어 사용자에게 문맥 지표 정보를 제공하는 역할을 한다. 현재 운영 중인 웹 응용들은 사용자의 효과적인 항해를 위해 여러 가지 항해 보조 방법을 사용하고 있다. 크게 사용자의 인지작용을 돕거나[13] 항해의 흐름[14]을 돕는 역할로 나눠볼 수 있다. 사용자의 인지작업을 돕기 위한 방법으로는 인덱스와 찾기 기능 또는 '홈으로', '처음으로'와 같은 표시를 통해 항해 경로를 단축시키는 방법을 사용할 수 있다.

사용자의 항해 흐름을 돕기 위해 추천 링크, 사용 가능한 링크만 활성화하기, 방문한 링크 구별, 특정 경로 강조하는 방법을 사용한다. 인터페이스 설계에서는 웹 응용이 담고 있는 내용을 어떻게 구성하여 보일 것인지를 결정하고 위에서 제시된 항해 보조 기능들을 배치하여 웹 응용의 항해를 향상시킨다.

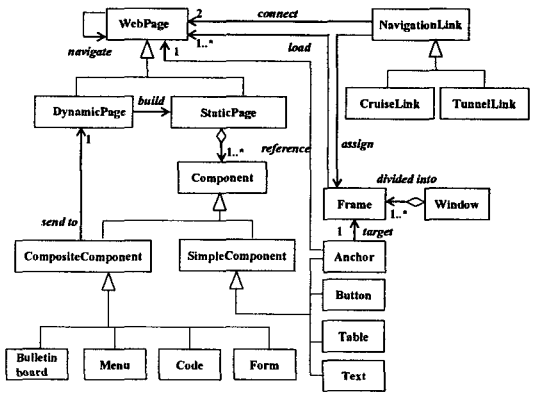
4.3 항해 실현

4.3.1 웹 응용 메타 모델

(그림 9)는 웹 응용의 물리적인 구성 요소들의 메타모델이다. <표 2>는 본 연구에서 사용된 물리적 구성요소들의 UML 스테레오타입을 설명한다. 윈도우는 프레임들로 나누어지며, 각 프레임에는 여러 웹 페이지들이 로드된다. 또한 각 웹 페이지는 정적(staic). 동적(dynamic) 페이지에 대한 링크를 갖는다. 정적 페이지는 html 파일일 수도 있고, 인자가 주어지면 동적 페이지로부터 생성될 수도 있다. 웹 페이지에 포함된 단순(simple) 또는 복합(composite) 컴포넌트는 동적 페이지와 상호작용한다. 단순 컴포넌트는 텍스트, 테이블, 리스트, 버튼, 앵커 등이고, 복합 컴포넌트는 폼, 코드, 메뉴, 게시판 등이 포함된다. 이러한 컴포넌트들은 하이퍼링크와 같이 연결 기능과 연산 기능을 동시에 할 수도 있다.



(그림 8) 인터페이스 클래스와 항해 클래스 관계



(그림 9) 웹 응용 메타모델

<표 2> 웹 응용 항해모델을 위한 스테레오타입

Stereotype	Base class	Parent	Description
NavigationContext <<navigationContext>>	UseCase	NA	A NavigationContext describes a navigation context of Web application
NavigationClass <<navigationClass>>	Class	NA	A NavigationClass defines a semantic unit appropriate for a navigation context
InterfaceClass <<interfaceClass>>	Class	NA	A InterfaceClass defines a element to assist user in navigation
Window <<window>>	Class	NA	A Window is a display in browser
Frame <<frame>>	Class	NA	A Frame is a region of a screen
WebPage <<webPage>>	Class	NA	A WebPage includes a StaticPage or a Dynamic Page
StaticPage <<staticPage>>	Class	WebPage	A StaticPage is a HTML page
DynamicPage <<dynamicPage>>	Class	WebPage	A DynamicPage is a WebPage that contains dynamic content on the server
NavigationLink <<navigationLink>>	Association	NA	A NavigationLink is a kind of link between WebPages
CruiseLink <<cruiseLink>>	Association	NavigationLink	A CruiseLink represents a normal link between WebPages
TunnelLink <<tunnelLink>>	Association	NavigationLink	A TunnelLink represents a navigation between WebPages with tunnel structure
Component <<component>>	Class, State	NA	A Component includes a CompositeComponent or a simple component
CompositeComponent <<compositeComponent>>	Class, State	Component	A CompositeComponent contains several Components
SimpleComponent <<simpleComponent>>	Class, State	Component	A SimpleComponent is a Button, a Label, or, etc.
BulletinBoard <<bulletinBoard>>	Class, State	CompositeComponent	A BulletinBoard is a message board to allow members to exchange their information.
Menu <<menu>>	Class, State	CompositeComponent	A Menu is a list of various options available
Code <<code>>	Class, State	CompositeComponent	A Code is an applet, an ActiveX control, a script, etc.
Form <<form>>	Class, State	CompositeComponent	A Form is a HTML form.
Anchor <<anchor>>	Class, State	SimpleComponent	An Anchor is a HTML anchor
Button <<button>>	Class, State	SimpleComponent	A Button is a simple HTML button.
Table <<table>>	Class, State	SimpleComponent	A Table is a HTML table.
Text <<text>>	Class, State	SimpleComponent	A Text is a text string.
TunnelNote <<tunnelNote>>	Comment	NA	A TunnelNote is a note used for describing a tunnel structure.

본 연구에서는 [20]에서 사용된 링크 클래스를 확장하여 항해 링크(NavigationLink)를 정의한다. 또한 웹 페이지의 하이퍼링크의 연결 구조에 따라, 예를 들면, 크루즈 링크(cruise link)와 터널 링크(tunnel link), 웹 페이지간의 연결정도가 결정되기도 한다. 간단한 개념은 한 웹 페이지로 보여 질 수도 있지만, 내용이 많거나 여러 단계를 보여야 할 경우에는 여러 웹 페이지로 이루

어지기도 한다. 결국 웹 응용은 구성 요소들이 항해 지표로 묶어서 웹 페이지 단위로 사용자에게 전달되는 것으로 이해할 수 있다.

OCL(Object Constraint Language)은 객체지향 모델링에서 세밀한 정보를 나타내기 위해 사용하는 언어이다. (그림 9) 메타모델의 의미를 더 분명히 표현하기 위해 다음과 같이 OCL을 사용하여 기술한다.

- 두 웹 페이지가 항해 링크에 의하여 연결된다. 링크의 목적 페이지가 동적 페이지라면 그 링크는 하나 이상의 인자를 가진다. 그렇지 않으면 인자를 갖지 않는다.

```
context NavigationLink inv:
  self.connect->size() = 2
  if self.connect.target.oclIsTypeOf(DynamicPage)
    then self.connect.arguments->size() >= 1
    else self.connect.arguments->size() = 0
  endif
```

- 터널 링크는 포함하고 있는 페이지에 존재하는 링크를 활성화시켜 항해한다. 만약 다른 페이지로 항해한후 다시 그 페이지로 복귀하는 항해는 허용되지 않는다. 즉, 터널 링크는 터널 구조안에 존재하는 이미 방문한 페이지들로의 복귀는 허용하지 않는다.

```
context TunnelLink::activate(args): Boolean
pre: self.navigate.target->include(self.connect.target)
```

- 복합 컴포넌트는(composite component)는 동적 페이지에 매개변수를 보낸다.

```
context CompositeComponent::send(parameters): Boolean
pre: self.sendTo.oclIsTypeOf(DynamicPage)
```

- 하나 이상의 웹 페이지가 프레임에 로드될 수 있다.

```
context Frame inv:
self.pages->size() >=1
```

- 앵커(anchor)는 하나의 웹 페이지를 참조하고 로드할 프레임을 명시한다.

```
context Anchor inv:
self.pages->size() = 1
self.frames->size() = 1
```

4.3.2 웹 페이지 항해 모델

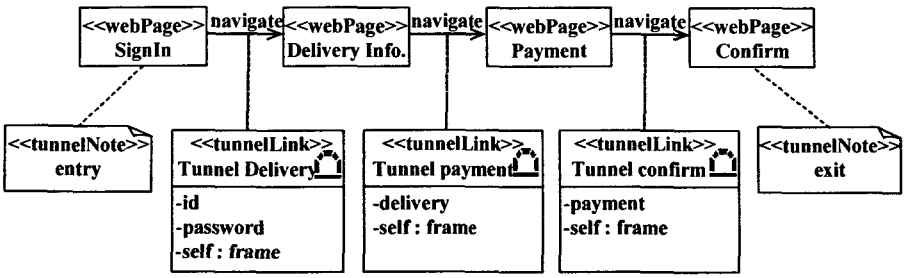
웹 응용들은 모두 각자의 웹 페이지간의 연결구조를 갖고 있다. [15]에서는 서로 다른 목적의 웹 응용에서 나타나는 서로 다른 웹 페이지 간의 연결구조와 공통적으로 나타나는 연결구조에 대해 기술하였다.

하이퍼링크는 연결된 곳은 어디든지 갈 수 있는 항해의 자유를 준다. 하지만 내용에 따라 웹 페이지의 연결정도를 다르게 해야 할 필요가 있다. 예를 들어 결제 과정에 해당하는 5~7 단계의 웹 페이지들은 강한 결합으로 묶을 필요가 있다. 이 웹 페이지들 간의 연결구조는 (그림 10)처럼 설계할 수 있고, 실현 단계에서 기술적으로 묶을 수 있다. 터널 구조는 입구와 출구가 있는 연결구조이며, 원칙적으로 중간 과정에서 빠져나갈 수 없다. 만약, 터널 중간 과정에서 사용자가 글로벌 항해 시스템을 통해 빠져나갔다면 터널 입구부터 다시 시작해야 한다. 터널 링크에 클래스에 표시된 아이콘은 페이지 사이의 연결이 터널 링크임을 한눈에 알 수 있도록 도와준다.

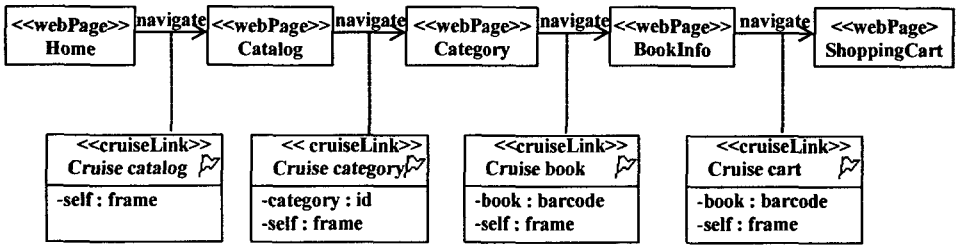
(그림 11)은 가이드투어를 나타내는 크루즈 링크를 사용한 연결 구조를 보여준다. 터널 연결 구조보다는 약한 결합 구조이지만 크루즈 링크 구조도 여러 장의 웹 페이지가 연결된다. 일반적으로 크루즈 링크 구조는 사용자에게 전체경로와 현재 위치에 대한 직관적인 그림 등을 통해 특정 경로를 유도하는 방식을 사용한다. 크루즈 연결 구조는 깃발 모양의 아이콘을 사용하여 나타낸다.

4.3.3 컴포넌트 항해 모델

웹 응용은 정보에 대한 항해와 브라우징만 가능했던 것에서 컨텐츠에 대한 연산까지 가능하게 발전되었다. 또한 기존에 하이퍼링크 클릭을 통해서만 항해할 수 있었지만, 페이지에 클라이언트 측의 스크립트나 프로그램을 사용하면 하이퍼링크에 대한 클릭 없이도 정보간의 항해가

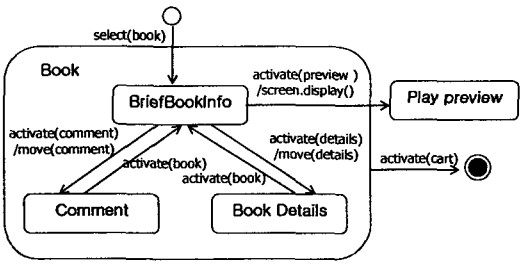


(그림 10) 책 구입 과정에 대한 웹 페이지 항해 모델



(그림 11) 책 검색 과정에 대한 웹 페이지 항해 모델

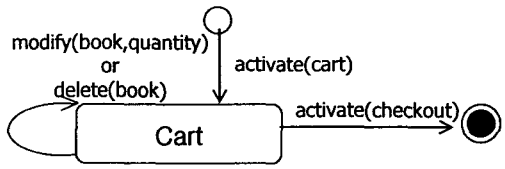
가능하다. 웹 응용의 광범위한 사용과 함께 이와 관련된 기술은 급속하게 발전하지만, 이것을 모델링하기 위한 방법은 부족하다.



(그림 12) 책 검색 웹 페이지에서의 컴포넌트 항해모델

개발이 끝난 후에도 빈번하게 이루어지는 새로운 항해 경로의 추가/삭제는 동적 컨텐츠와 사이트 구조에 영향을 주어 웹 응용의 일관성과 보안 등에 예상치 못한 문제를 야기시킬 수 있다. 따라서, 한 페이지 내에서 복잡한 동적 컨텐츠들의 항해에 대한 표현은 컴포넌트 수준에서 표현될 필요가 있으며, 특히 웹 응용 분야의 새로운 기술들 - Javascript, iframe, ASP, JSP - 은 웹 응용을 페이지간의 항해만으로 기술하는 데 한계를 갖는다. 페이지를 구성하는 컴포넌트들 간의 항해로 그에 따라 페이지의 동적인 상태가 달라진다 [16]. 이를 표현하기 위해 UML 상태도가 적합하며, UML을 확장하여 컴포넌트 수준의 항해를 기술하였다.

(그림 12)는 책을 찾는 웹 페이지에서 컴포넌트들의 항해를 통해 책에 대한 간략정보, 서평, 책의 세부정보를 각각 펼칠 수 있는 것을 표현한다. 작가의 인터뷰를 보거나 선택한 책을 장바구니에 담으려 하면 다른 웹 페이지로 이동하게 된다.



(그림 13) 쇼핑카트에 대한 컴포넌트 항해 모델

장바구니를 보여주는 웹 페이지에서는 컴포넌트를 통해 책의 수량 수정, 선택한 책 제거에 대한 연산을 할 수 있다 (그림 13). 결제를 선택하면 다른 웹 페이지로 이동하며, 현재 장바구니 상태를 빠져 나온다.

4.4 항해 모델 검증

항해 모델의 가장 큰 문제는 페이지와 항해의 수가 늘어날수록 복잡도가 증가하고 모델 내의 항해 오류를 검출하기 어렵다는 것이다. 이런 이유로 작은 단위의 항해 모델은 유용하게 활용될 수 있으나 복수의 모델을 통합한 전체 항해 모델을 검증하기에는 부적합한 측면이 있다. 이러한 기존 항해 모델의 단점을 보완하기 위해 항해 규칙을 이용한 모델 검증 방법을 제시한다.

모델 검증을 위해 웹 응용을 페이지와 항해의 그래프로 보고 항해 모델의 요소를 다음과 같이 정의할 수 있다. 웹 응용은 페이지들을 정점(vertex)으로 하고 그들 간의 관계인 항해를 간선(edge)으로 가지는 그래프 구조로 기술하도록 한다.

$$W = G(P,N) \quad (1)$$
P: 페이지들의 집합
N: 항해들의 집합

항해의 근원페이지, 목적페이지와 경로, 경로 항해 집합을 다음과 같이 정의한다.

정의 1. 근원페이지, 목적페이지

페이지 p에서 페이지 q로의 항해 n에 대해, p를 n의 근원페이지라 하고 source(n)으로 표시한다. q를 n의 목적페이지라 하고 target(n)으로 표시한다.

정의 2. 경로

유한 연쇄 np(p,q) = {n1, n2, n3, ⋯, nm}가 다음의 조건을 만족할 때, np(p,q)는 페이지 p에서 페이지 q로의 경로라고 정의한다.

$$\begin{aligned} &source(n_1) = p, \\ &target(n_m) = q, \\ &target(n_i) = source(n_{i+1}) \\ &\qquad where\ 1 \leq i \leq m-1 \end{aligned} \quad (2)$$

정의 3. 경로 항해 집합

페이지 p에서 페이지 q로의 경로 np(p,q) = {n1, n2, n3, ⋯, nm}의 모든 항으로 이루어진 집합을 경로 항해 집합이라고 하고 pns(p,q)로 나타낸다.

정의 4. 경로 집합

페이지 p에서 페이지 q로의 모든 경로 항해 집합들의 집합을 p에서 q로의 경로 집합이라고 정의하고 PS(p,q)로 나타낸다.

각 페이지와 항해는 식별자(identifier)를 가지고 있다고 가정하고 페이지 식별 함수와 항해 식별 함수를 다음과 같이 정의한다.

정의 5. 페이지 식별 함수

다음 조건을 만족하는 함수 page를 페이지 식별 함수라고 한다.

$$\begin{aligned} &page : string \rightarrow P \\ &page(id) = \text{식별자가 } id\text{인 페이지} \end{aligned} \quad (3)$$

정의 6. 항해 식별 함수

다음 조건을 만족하는 함수 nav를 항해 식별 함수라고 한다.

$$\begin{aligned} &nav : string \rightarrow N \\ &nav(id) = \text{식별자가 } id\text{인 항해} \end{aligned} \quad (4)$$

웹 응용 요소에 대한 정의를 이용하여 항해 규칙을 정의하고 이를 항해 모델을 검증하는 데 이용할 수 있다. 여기에서 항해 규칙이란 페이지와 항해를 설계할 때 반드시 지켜야 할 제약 조건을 말하는 데 항해 규칙은 크게 다음의 네 가지 유형으로 분류할 수 있다.

- **페이지 직접 연결 조건**

페이지 직접 연결 조건이란 서로 다른 두 페이지 p와 q에 대해 p에서 q로의 경로는 p에서 직접 q로 연결되는 항해 이외에는 없어야 한다는 조건을 의미한다. 식별자가 각각 x와 y인 두 페이지 간의 항해에 페이지 직접 연결 조건이 있다면 DL(x, y)라고 표현하고 그 정의는 아래와 같다.

$$\begin{aligned} &\forall n \in N, \\ &(source(n) = page(x) \Leftrightarrow target(n) = page(y)) \end{aligned} \quad (5)$$

- **마지막 페이지 조건**

일부의 페이지의 경우 더 이상의 항해를 허용

하지 않는 경우가 있다. 이 때 사용되는 조건이 마지막 페이지 조건이다. 식별자가 x인 페이지에 마지막 페이지 조건이 적용된다면 LP(x)라고 표현하고 그 의미는 다음 정의와 같다.

$$\forall n \in N, (source(n) \neq page(x)) \quad (6)$$

● **필수 항해 포함 조건**

페이지 p에서 q로의 모든 경로는 반드시 하나 이상의 항해 n을 거쳐야만 한다는 조건을 말한다. 식별자가 각각 x와 y인 두 페이지 간에 필수 항해 포함 조건이 있고 반드시 식별자가 z인 항해를 거쳐야 한다면 IN(x, y, z)라고 표현하고 다음과 같이 정의한다.

$$\forall pns \in PS(page(x), page(y)), (nav(z) \in pns)) \quad (7)$$

● **항해 불포함 조건**

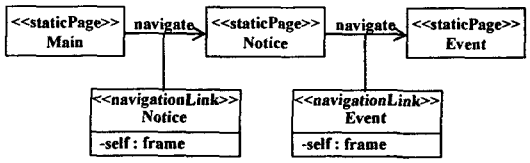
페이지 p에서 q로의 모든 경로는 반드시 항해 n을 거치지 않아야 하는 조건을 말한다. 식별자가 각각 x와 y인 두 페이지 간에 항해 불포함 조건이 있고 반드시 식별자가 z인 항해를 거치지 않아야 한다면 XN(x, y, z)으로 표현하고 다음과 같이 정의한다.

$$\forall m \in N, \forall pns \in PS(page(x), page(y)), \quad (8)$$
$$(m \in pns \Rightarrow m \neq nav(z))$$

항해 규칙이 설정되면 이를 이용해 항해 모델의 오류를 검증할 수 있다. 항해 모델링 결과 추출된 페이지들의 집합을 P, 항해들의 집합을 N이라고 하고 모든 항해 규칙들의 집합을 R이라고 하자. R의 모든 규칙에 대해 규칙의 유형에 따라 다음과 같이 검증을 시행할 수 있다.

● **규칙의 유형이 페이지 직접 연결 조건인 경우**

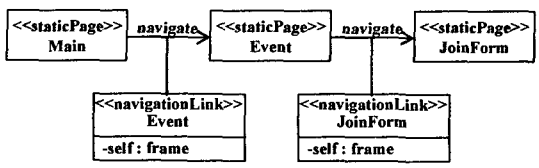
항해 규칙이 DL(x, y)로 주어진 경우 항해 모델의 모든 항해들에 대해 식별자가 y인 페이지로 향하는 항해의 근원페이지를 검사한다. 만약 근원페이지의 식별자가 x가 아니라면 해당 모델은 페이지 직접 연결 조건을 위반한 것이라고 할 수 있다. (그림 14)는 페이지 직접 연결 조건에 위배되는 항해 모델의 예이다. 항해 규칙이 DL('Main.html', 'Event.html')로 정의되어 있다면 'Event.html'로 가는 항해는 반드시 'Main.html'에서 출발해야 하지만 모델에서는 'Notice.html'에서 연결되는 항해가 발견된다.



(그림 14) 페이지 직접 연결 조건을 위배한 항해 모델

● **규칙의 유형이 마지막 페이지 조건인 경우**

항해 규칙이 LP(x)로 주어진 경우 항해 모델의 모든 항해들에 대해 근원페이지의 식별자가 x인 항해의 존재 여부를 검사한다. 만약 그러한 항해가 존재한다면 해당 모델은 마지막 페이지 조건에 위배된 모델이라고 판단할 수 있다. (그림 15)는 마지막 페이지 조건에 위배되는 항해 모델의 예이다. 항해 규칙이 LP('Event.html')로 정의되어 있다면 어떤 항해도 'Event.html'에서 출발할 수 없지만 항해 모델에서는 'JoinForm.html'로 연결될 수 있는 항해가 존재한다.

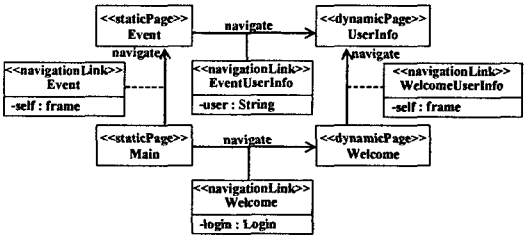


(그림 15) 마지막 페이지 조건을 위배한 항해 모델

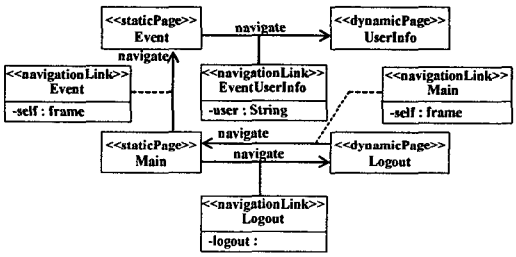
● **규칙의 유형이 필수 항해 포함 조건인 경우**

항해 규칙이 IN(x, y, z)로 주어진 경우, 항해 모델의 모든 페이지 중 식별자가 x인 페이지를 먼저 검사한다. 그래프 탐색 알고리즘을 이용해 식별자가 y인 페이지로의 경로를 추출하고 모든 경로가 식별자가 z인 항해를 포함하는지를 조사하면 항해 모델을 검증할 수 있다. (그림 16)은 필수 항해 포함 조건을 위반한 항해 모델의 예이다. 항해 규칙이 IN('Main.html', 'UserInfo.jsp', 'Welcome')으로 주어진 경우 'Main.html'에서 'UserInfo.jsp'로의 모든 경로는

반드시 식별자가 'Welcome'인 항해를 거쳐야만 하지만 모델에서는 식별자가 'Welcome'인 항해를 거치지 않는 경로가 존재하게 되어 오류가 있음을 알 수 있다.



(그림 16) 필수 항해 포함 조건을 위반한 항해 모델



(그림 17) 항해 불포함 조건을 위반한 항해 모델

- **규칙의 유형이 항해 불포함 조건인 경우**

항해 규칙이 XN(x, y, z)로 주어진 경우 필수 항해 포함 조건의 경우와 비슷한 과정으로 검증을 실시할 수 있다. 항해 모델의 모든 페이지 중 식별자가 x인 페이지를 먼저 검사한 후 그래프 탐색 알고리즘을 이용해 식별자가 y인 페이지로의 경로를 추출하고 모든 경로 중 식별자가 z인 항해를 포함하는 경로가 존재하는 지 여부를 조사한다. (그림 17)은 항해 불포함 조건을 위반한 항해 모델의 예이다. 항해 규칙이 XN('Main.html', 'UserInfo.jsp', 'Logout')으로 주어진 경우 'Main.html'에서 'UserInfo.jsp'로의 모든 경로는 반드시 'Logout' 항해를 거쳐서는 안 되지만 모델에서는 'Logout' 항해를 거치는 경로가 존재하는 오류를 보인다.

위와 같이 항해 규칙을 이용함으로써 항해 모델을 검증할 수 있다.

5. 결론

본 연구에서는 웹 응용의 적응하는 문맥-기반 컨텐츠 항해 모델링 프레임워크를 제안하였다. 프레임워크에서는 항해 분석, 항해 설계, 그리고 항해 실현 활동을 수행한다. 기본적으로 프레임워크의 활동들은 반복적 점진적으로 수행되며 각 활동에서 하나 이상의 UML을 확장한 모델들이 산출물로 새로 만들어지거나 갱신된다. 항해 분석 활동에서는 외부 관점에 초점을 두고 유즈케이스를 사용하여 영역 분석을 하며 텍스트와 UML 유즈케이스도를 사용하여 항해 문맥 모델을 작성한다. 다음으로, 항해 설계 활동은 정보 단위와 항해 경로를 정하는 항해 정보 모델, 사용자의 개인 정보와 환경 정보를 제공하는 프로파일, 그리고 정보를 보여주기 위한 항해 인터페이스 모델을 생성한다. 마지막으로, 항해 실현 활동에서는 웹 응용 메타 모델을 정의하고 웹 페이지 항해 모델과 컴포넌트 항해 모델을 작성한다. 또한 항해 모델을 검사하기 위한 정형적인 정의와 규칙을 제시하였다.

본 연구의 프레임워크는 웹 응용 항해 분석, 설계, 그리고 실현의 활동단계까지 일관성 있는 모델을 제공함으로써 웹 디자이너와 소프트웨어 개발자가 함께 웹 응용 항해에 대하여 의사소통하기 위한 일관성있는 방법을 제공한다. 둘째로 이 프레임워크의 문맥-기반 항해 모델링은 웹 응용 항해를 개선함으로써 사용자가 웹 응용을 사용한 정보검색 또는 상거래를 용이하게 할 것으로 기대된다. 셋째로 적응하는 항해 모델링은 모든 사용자에 모든 것을 제공하는 기존 웹 응용의 단점을 극복하여 사용자마다 다른 항해를 제공하며 또한 같은 사용자일지라도 컴퓨터 또는 PDA 등의 사용 환경에 따라 다른 항해를 가능케 하는 맞춤형 항해의 기초를 제공한다.

웹 컨텐츠와 복잡성은 빠르게 증가하지만 웹 응용의 목적에 따라 항해가 특정한 패턴을 갖는 경향을 보인다. 따라서 웹 항해를 분석하여 항해 패턴을 추출하는 연구를 할 것이다. 또한 본 모델링 방법을 적용한 다양한 사례 연구를 수행, 사례 연구 자료를 바탕으로 프레임워크를 개선해 나갈 것이다.

참고문헌

[1] J. Park and J. Kim, "Effect of Contextual Navigation *Aids on Browsing Diverse Web Systems," In Proc.* of the CHI '2000 Human Factors in Computing Syst ems, 2000.

[2] R. Hennicker and N. Koch, "Modeling the User Interf ace of Web Applications with UML," Practical UML -based Rigorous Development Methods - Counterin g or Integrating the eXtremists, Workshop of the pUML-Group at the UML2001, pp.158-172, 2001.

[3] P. Dolog and M. Bielikova, "Hypermedia Modeling Using UML," In Proc. of ISM Conference, 2002.

[4] D. Schwabe and G. Rossi, "An Object Oriented Appr oach to Web-based Application Design," Theory an d Practice of Object Systems, Vol. 4, Iss. 4, pp. 207-225, Wiley & Sons, 1998.

[5] L. Baresi, F.Garzotto, and P. Paolini, "Extending UM L for Modeling Web Applications," In Proc. of the 34th Hawaii International Conference on System Sci ences, 2001.

[6] Object Management Group, Inc.: OMG Unified Model ing Language Specification. Version 2.0, OMG, Dec. 2004..

[7] H. Baumeister, N. Koch, and L. Mondel, "Towards a UML extension for hypermedia design," In Proc. of UML'99 Conference,LNCS 1723, pp. 614-629,1999

[8] G. Larsen and J. Conallen, "Engineering Web-Based System with UML assets," Annals of Software Engi neering, Vol. 13, pp. 203-230, 2002.

[9] J. Hong, B. Lee, H. Kim and C. Wu, "Modeling contex t-based Navigation of Web Application Using UM L," In Proc. of The 2th ACIS International Congeren ce on Software Engineering Research, Management and Applications, 2004.

[10] P. Brusilovsky, "Adaptive Hypermedia," In Proc. of the 7th international conference on Intelligent user interfaces, 2002.

[11] T. Alatalo and J. Peraaho, "Designing mobile-aware Adaptive hypermedia," In Proc. of third Workshop on Adaptive Hypertext and Hypermedia at ACM Co nference on Hypertext and Hypermedia, 2001.

[12] L. Rosenfeld and P. Morville, Information Architect ure for the World Wide Web, 2nd ed., O'Reilly, 2002.

[13] M. Neerincx, J. Lindenberg and S. Pemberton, "Sup port Concepts for Web Navigation: a Cognitive Engi neering Approach," In Proc. of Tenth International World Wide Web Conference, 2001.

[14] T. Murray, T. Shen, J. Piemonte, C. Condit and J. Thibedeau, "Adaptivity for Conceptual and Narrativ e Flow in Hyperbooks: The MetaLinks System," In Proc. of the International Conference on Adaptive Hypermedia and Adaptive Web-based System,2000

[15] M. Gillenson, D. L. Sherrell and L. Chen, "A Taxono my of Web Site Traversal Patterns and Structures," Communications of the AIS, Vol. 3, June 2000.

[16] K.R.P.H. Leung, L.C.K. Hui, S.M. Yiu and R.W.M. Tang, "Modeling Web navigation by statechart," In Proc. of International Computer Software and Appli cations Conference (COMPSAC), Oct. 2000.

[17] A. Adamko, "Modeling Data-Oriented Web Applica tions using UML," In Proc. of EUROCON, Vol. 1, pp. 752 - 755, Nov. 2005.

[18] A. Kraus, and N. Koch, "A Metamodel for UWE," Technical Report 0301, Ludwig-Maximilians-Unive rsitat Munchen, 2003.

[19] G. Rossi, D. Schwabe, and A. Garrido, "Design Reus e in Hypermedia Applications Development," In Pro c. of The ACM International Conference on Hyperte xt, 1997.

[20] J. Conallen, Building Web Applications with UML, 2nd ed., Addison Wesley, 2002.

이 병 정

1990년 : 서울대학교 계산통계학과 (이학사)

1998년 : 서울대학교 전산과학과 (이학석사)

2002년 : 서울대학교 컴퓨터공학부 (공학박사)

2002년 - 현재 : 서울시립대학교 컴퓨터과학부 교수

관심분야 : 소프트웨어 방법론, 품질 평가, 소프트웨어공학

홍 지 원

2002년 : 서울시립대학교 전산통계학과 (이학사)

2005년 : 서울시립대학교 컴퓨터통계학과 (이학석사)

2005년 ~ 현재 : 삼성 SDS

관심분야 : 웹 공학, 웹 응용 항해 모델링