제22회 한국정보처리학회 추계학술발표대회 논문집 제11권 제2호 (2004. 11)

Product-Line 에서의 UI 자산화 기법

황길승, 윤석진, 송문섭, 양영종
한국전자통신연구원 기반기술연구소 임베디드 S/W 기술센터 S/W 공학연구팀
e-mail : {kshwang, sjyoon, sirius, yangyj}@etri.re.kr

Methods of managing UI Assets in Product-Line Engineering

Kil-Seung Hwang , Seok-Jin Yoon, Moon-Sub Song, Young-Jong Yang
Software Engineering Research Team, Embedded S/W Technology Center, Basic Research Laboratory, ETRI(Electronic and Telecommunication Research Institute)

요 약

Product-Line 기반의 소프트웨어 개발은 특정 도메인의 소프트웨어 제품군 내의 공통성과 가변성 분석을 통한 공통 아키텍처의 추출과 재사용으로 개발의 생산성과 효율성을 향상시킨다. 일반적인 Proudct-Line 기반 방법론에 따르면, 재사용되는 핵심자산은 아키텍처, 컴포넌트, 등의 소프트웨어 내부 비즈니스 로직에 한정되어 있으며, 실제로 소프트웨어 개발에 있어 많은 비용이 지출되는 UI(User Interface), 데이터베이스 등의 설계 및 개발에 대한 내용은 언급되어 있지 않은 실정이다.

본 논문에서는 소프트웨어 개발에 필요한 UI 모듈을 핵심자산의 형태로 Product-Line 에서 사용할 수 있도록 하는 방법을 제안한다. UI 모듈을 설계하여 명세하는 방법과 설계된 UI 를 디자인 템플릿과 연결하여 내부 로직과 연계하는 방법에 대해 설명한다. 이 방법을 이용하면 Product-Line 을 위한 핵심자산 구성시 UI 모듈을 포함할 수 있어 생산성과 효율성을 향상시킬 수 있을 것으로 생각된다.

1. 서론

80 년대까지의 구조적인 S/W 개발방법은 S/W 의 다양화, 대형화에 따른 요구의 증가로 90 년대 초 객체지향 방법, 90 년대 말 컴포넌트 기반 방법 등으로 진화해 왔고, 현재는 S/W 시스템 개발에 필요한 서비스 컴포넌트의 다양한 보급이 가능해지고, 컴포넌트 기반 기술이 성장함에 따라 시장성, 기술적 진보성이 보완된 Product-Line 기반 개발방법, Model-Driven 개발방법, Service 기반 개발방법 등의 한단계 진화한 S/W 개발 기술이 도입되고 발전하기 시작하고 있다.

S/W Product-Line 은 공통의 유사한 기능을 가지고 있는 S/W 제품 또는 S/W 시스템의 집합을 의미하며 [1], S/W Product-Line 기반 개발이란 특정 영역의 시장과 용도의 요구사항을 만족시키기 위해 S/W 제품을 미리 구축된 S/W 아키텍처 등의 S/W 핵심자산을 재사용하여 개발하는 방법이다. 이 방법은 미리 구축된 S/W 핵심자산을 재사용하므로, 처음부터 전체 시스템을 개발하는 방식보다 쉽고, 빠르게 S/W 를 생산할 수 있는 장점이 있다....

S/W 핵심자산을 효율적이고 정확하게 구성하는 것은 개발의 생산성과 효율성을 높이는 첫 번째 요구조건이다. 잘 정의된 자산은 재사용성을 높여주어 성공적인 Product-Line 을 구성할 수 있게 한다. 이를 위해서는 잘 검증된 자산을 확보하는 것도 중요하지만 S/W 의 개발에 필수적으로 필요한 요소들에 대한 자산화 방법도 중요한 요소중 하나이다.

일반적인 S/W 의 개발에 있어서 많은 개발자들의 노력과 시간소비를 필요로 하는 분야 중 하나가 UI 개발이다. UI 는 S/W 와 사용자와의 연결고리의 역할을 하는 중요한 부분이므로, 보통의 개발에서는 UI 설계 및 개발에 많은 시간을 할당한다. 그러므로, Product-Line 의 재 사용률을 높이기 위해서는 UI 부분의 자산화가 반드시 필요하다.

하지만 UI 설계 및 개발은 특정한 규칙을 따르거나 보편적인 방법이 있는 것이 아니라 UI 개발자 및 디자이너의 개성이나 S/W 의 특성이 포함되기 때문에

특정한 형태로 개발을 정형화하기는 힘들다. 그리고 UI 컴포넌트들의 다양성과 각각이 가지고 있는 특징들이 모두 다르므로 특정한 설계방법을 정의하는 것도 어렵다.

본 논문에서는 이러한 문제점을 해결하고 UI 의 핵심자산화를 통해 Product-Line 의 재사용성과 효율성을 높이기 위해 UI 를 모델링하고 구현하여 UI 자산화할 수 있는 방안을 제안한다. 제안된 방안은 다음과 같은 방법들을 포함한다.

- UI 모델링 기법
- UI 디자인과 모델의 매핑기법
- Product-Line 자산화 방법

제안된 방법을 이용하면 UI 개발에 있어 UI 컴포넌트의 구성과 Event Handling, 그리고 디자인 템플릿등을 상황에 맞게 재사용할 수 있으므로, 특정 형태로의 자산화가 가능하다. 따라서, Product-Line 의 핵심자산으로의 활용이 가능해지며, S/W 개발에서 오는 비용을 감소시켜 효율성과 경제성을 증대시킬 것으로 생각된다.

본 논문의 구성은 다음과 같다. 2 장에서는 UI 의 특징과 구성에 대해 설명하고, 3 장에서는 UI 모델링 방법에 대해 설명한다. 4 장에서는 디자인 템플릿 적용방법과 코드생성에 대해 설명하고, 5 장에서 결론을 맺는다.

2. UI 의 특징과 구성

S/W 의 비즈니스 로직은 컴포넌트 공학의 캡슐화 방법을 통해 특정 기능별로 모듈화되는 방법으로 정형화가 가능하다. CBD(Component Based Development)에서는 이러한 특정한 규칙을 준수하는 컴포넌트 단위를 조립하여 S/W 의 기능을 표현한다.

UI 의 경우는 버튼이나 테이블 등의 하나의 단위 행위를 표현하는 단위 UI 컴포넌트로만 정형화될 수 있다. 이러한 UI 컴포넌트는 현재 보편적인 개발도구나 지원기술로 사용되고 있다. 하지만, 전체 S/W 의 UI 전체를 특정 형태로 정형화하는 기술은 보편화되어 있지 못하다. 본 논문에서는 UI 전체를 정형화하여 재사용하기 위해 UI 의 View 에 해당하는 부분과 Model 에 해당하는 부분을 분리해서 모델링하여 연결하는 방법을 정의하였다.

일반적으로 최근의 UI 개발은 MVC 모델을 기반으로 하고 있으며, Model, View, Controller 의 3 부분으로 나뉘어져 이루어진다.

- Model: 프로그램상태에 대한 논리적인 표현이다. 데이터가 변경되었을 경우에는 이를 View 에게 통보한다.
- View: 변하는 데이터에 대한 시각적인 표현을 제공하며 응답메커니즘으로 Controller 를 사용한다.
- Controller: 사용자입력을 어떻게 다뤄야 할지를 명세한다.

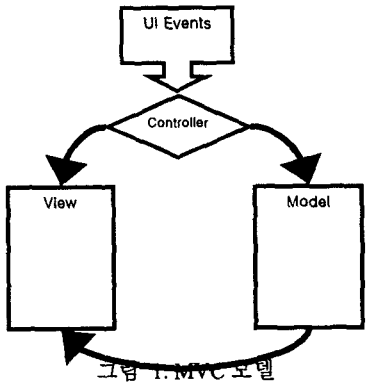

그림 1. MVC 모델

Model 과 Controller, 그리고 View 의 분리된 표현은 본 논문의 UI 모델링의 기반이 되는 조건이다. Model 부분과 Controller 부분은 UI Diagram 을 통해 표현되고 View 부분은 디자인 템플릿의 커스터마이징 방법으로 적용될 수 있다.

3. UI 모델링

UI 의 페이지 별 구성 컴포넌트와 데이터의 흐름 및 전달, 이벤트의 발생과 기술, 비즈니스 로직과의 연결관계 등을 표현하기 위해 특정형태로의 모델링 방법이 필요하다.

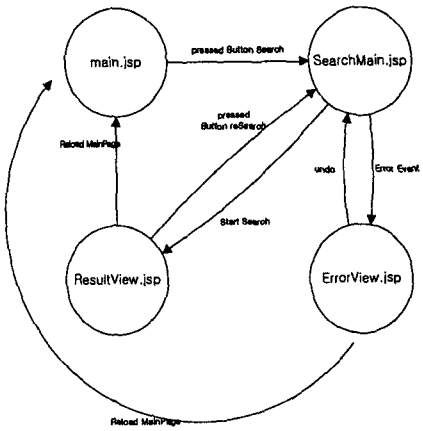

그림 2. UI Page 간의 View Interaction Diagram

UI 의 활동을 표현하기 위한 UI 모델링의 다이어그램은 다음과 같다.

- External Interaction Diagram
  UI 전체를 구성하고 있는 View 전체적인 상호작용을 기술하고 있는 다이어그램.
- Internal Interaction Diagram

하나의 View 내부에서 UI 컴포넌트들 간의 이벤트 전달 및 페이지로의 값 전달 등의 활동을 기술하고 있는 다이어그램

- View Sequence Diagram
  전체적인 UI 동작의 흐름을 기술하고 있는 다이어그램

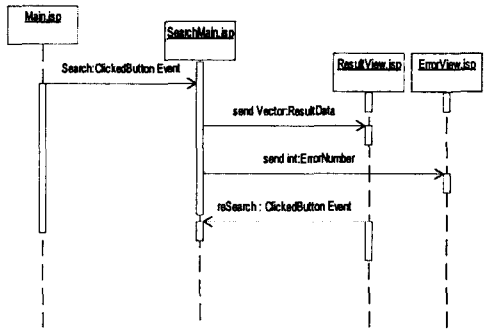

그림 3. View Sequence Diagram

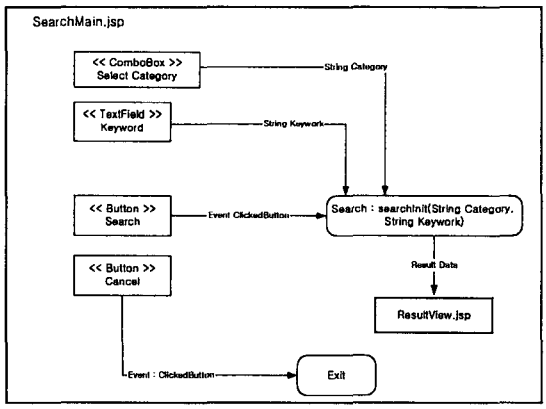

그림 4. Internal Interaction Diagram

위에서 정의한 세 개의 다이어그램을 통해 UI 사이에서 일어나는 이벤트 및 구조들을 정형화할 수 있고, 이는 UI 의 Model 을 표현한다. UI 모델링을 통해 표현할 수 있는 요소들은 다음과 같다.

- UI View 간의 전달되는 이벤트의 정보
- UI View 간의 흐름절차
- UI 컴포넌트의 선택에 따른 View 의 변화
- UI 컴포넌트 사이의 이벤트 발생
- UI 컴포넌트 사이의 데이터의 흐름
- UI View 를 구성하고 있는 UI 컴포넌트
- UI View 와 비즈니스 로직의 인터페이스와의 상호작용

4. 디자인 적용 및 코드생성

UI 의 디자인은 디자이너의 개성이 가장 강하게 들어가는 부분이므로 특정형태로 정형화할 수 없다. 그래서 본 논문에서는 디자인요소의 적용을 디자인템플릿을 UI 모델에 적용하는 형태로 정의하였다.

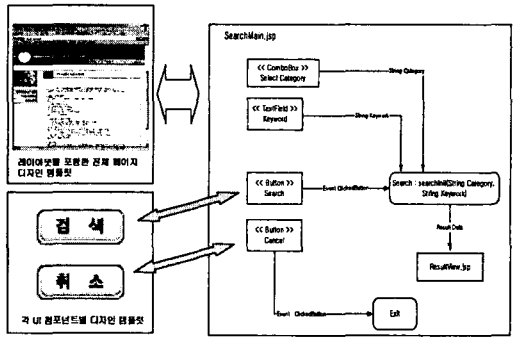

그림 5. 디자인 템플릿 적용

5. UI 의 자산화방안

위에서 정의한 모델링과 템플릿적용 방법을 통해 UI 에서 표현하고자 하는 많은 부분들을 정형화 할 수 있다. 정형화된 모듈을 Product-Line 을 위한 자산으로 표현하기 위해서는 모듈의 공통 부분과 가변 부분을 나누고 가변부분에 대해 관리할 수 있는 방법이 정의되는 것이 필요하다.공통 부분과 가변 부분은 다음과 같이 구분될 수 있다.

- 공통 부분
  - 디자인 레이아웃
  - UI 컴포넌트 구성
  - UI 의 전체적인 기능
- 가변 부분
  - 디자인 템플릿 선택
  - UI 컴포넌트의 형태 선택

일반적인 Product-Line 방법에서는 가변성관리에 대해 Decision-Resolution Model 을 적용한다. 자산화했을 경우 S/W 로의 적용은 Variant 의 Resolution 을 통해 이루어지며, Resolution 은 컴포넌트의 선택을 통해 아키텍처를 고정시키며, 세부적인 결정사항들을 결정함으로써 완성된 S/W 의 형태를 구성한다.

UI 에서의 Decision Model 도 이와 유사하다. UI 컴포넌트의 형태와 디자인 템플릿의 종류를 Variant 로 정하여 이를 Resolution 함으로써 UI 의 형태를 고정시킨다.

ID	Variation	Resolution	Effect
4.1	select Design Template	1..10	선택에 따른 디자인 템플릿 적용
4.2	Search	버튼	버튼 생성(4.2.1)
		하이퍼링크	하이퍼링크 생성
4.3	Cancel	버튼	버튼 생성(4.2.2)
		하이퍼링크	하이퍼링크 생성
4.4	Keyword	TextField	TextField 생성
		TextArea	TextArea 생성
4.5	Select Category	ComboBox	ComboBox 생성
		Option	Radio button 생성

ID	Variation	Resolution	Effect
4.2.1	버튼	Activate	Activate 버튼 생성
		Passivate	Passivate 버튼 생성
4.2.2	버튼	Activate	Activate 버튼 생성
		Passivate	Passivate 버튼 생성

표 1. UI 의 Decision Model

하지만, UI 모듈의 부분은 비즈니스 로직 부분에 비해 변화할 수 있는 범위가 넓다. 단순한 optional 또는 variant 가변성만 존재하지 않고 로직 부분의 자산의 형태에 따라 종속적으로 적용되어야 하기 때문에 사용자가 직접 수정하여 변경하여야 할 부분이 많을 것으로 생각된다. 그러므로, 가변부분에 대한 정의는 UI 자산의 정의된 형태에 따라 탄력적으로 정의하는 것이 바람직할 것으로 생각된다.

6. 결론 및 향후연구

본 논문에서는 Product-Line 을 기반으로 한 S/W 를 개발하는데 있어서 UI 부분을 특정 형태로 정형화하여 핵심자산화 하기 위한 방법을 제안하였다. UI 부분은 MVC 모델을 기반으로 하여 UI 의 내부 구성과 연관성을 표현하는 Model 부분과 외적으로 보여지는 디자인 부분을 표현하는 View 부분으로 나뉘어 질 수 있고, 이 구분을 이용하여 UI Model 을 정형화하고 여기에 디자인 템플릿을 연결하는 방법으로 UI 모듈을 정형화하였다.

이 방법에 대한 연구는 현재 S/W 의 생산성 향상을 위해 제안되는 다양한 방법론들의 세부연구에 위치한다. 가장 개발자의 노력이 많이 들어가는 부분인 UI 정형화 부분에 대한 연구가 성공적으로 이루어진다면, Product-Line 및 기타 S/W 개발방법론들의 생산성 향상에 큰 영향을 줄 것으로 기대한다.

본 논문에서는 UI 모델링 방법과 디자인 템플릿과의 연결방법에 대해 정의하였다. 하지만 UI 모델링의 구체적인 방법과 Case Study 적용에 따른 사실성 증명 부분에 있어 근거와 예시가 부족하다. 그리고 UI 의 자산화에서 가변성 관리의 Variant 의 종류가 매우 다양하게 나오기 때문에 가변성 관리 부분에 대한 정의를 기존의 가변성 관리와는 별도로 정의하여야 할 것으로 생각되며, 이 부분에서 향후 지속적인 연구가 필요할 것으로 생각된다.

참고문헌

[1] Northrop L., Framework for Software Product Line Practice, SEI Report, 2002. (http://www.sei.cmu.edu/plp/framework.html)

[2] Colin Atkinson et al., *Component-based Product Line Engineering with UML*, Addison-Wesley, 2002.

[3] David M. Weiss, Chi Tau Robert Lai, *Software Product-Line Engineering : A Family-Based Software Development Process*, Addison-Wesley, 1999.

[4] Booch & Rumbaugh & Jacobson, *UML User' s Guide*, Addison Wesley, 1999.

[5] 양회석 , 허광남 , 노재춘 , 이선재, *IT EXPERT 모델 2 로 다시 배우는 JSP*, 한빛미디어, 2003.