영역지향 소프트웨어 개발방법론

김태영*, 김치수*, 임재현**
*공주대학교 컴퓨터공학과
**공주대학교 컴퓨터 멀티미디어 공학과

e-mail:youngimk@gjue.ac.kr

The Software Development Methodology with Aspect-Oriented

Tae-Young Kim*, Chi-Su Kim*, Jae-Hyeon Yim**
*Dept. of Computer Engineering, Kong-Ju National University
**Dept. of Computer Multimedia Engineering, Kong-Ju National University

요 약

소프트웨어를 부품화하고 이를 조립·합성하여 새로운 어플리케이션을 개발하는 방식의 CBD 방법론이 소프트웨어 개발방법론으로 많이 연구되고 있다. 그러나 이 방법론은 시스템의 기능적인 특성을 중심으로 분할하는 경향이 많아 컴포넌트에 대한 추론, 문서화, 코드의 이해를 어렵게 하는 단점이 있다. 따라서 본 논문에서는 "영역지향 CBD 방법론"이라고 명명한 CBD 방법론을 새롭게 제시하여 컴포넌트의 재사용을 용이하게 함으로써 시스템 개발 시간 단축과 개발비용의 감소를 유도하였다.

1. 서론

최근 가장 대표적인 소프트웨어 개발 방법론은 사용자의 복잡·다양한 요구사항들을 정해진 시간 안에 정확하게 충족시켜 주고, 개발자의 편의와 효율성을 높일 수 있도록 하는 CBD 방법론이다[1].

대표적인 컴포넌트 기반 기술의 예로는 자바빈즈, 엔터프라이즈 자바빈즈, COM, CORBA, JViews등이 있다[5]. 그러나 이 기술들의 대부분은 이벤트 흐름, 프로세스 뷰 프로세스 단계 등과 같은 시스템의 기능적인 서비스에 초점을 두고 있다. 따라서 시스템 컴포넌트에 대한 사용자 인터페이스, 처리관리, 영속성, 협력과 보안 영역 등과 같은 시스템의 비기능적인 부분과 크로스컷팅에 대한 정보가 부족한 것이 단점이다.

본 논문에서는 영역지향 CBD 방법론이라고 명명한 새로운 방법론을 제안하여 CBD 방법론의 단점을 보완함으로써 컴포넌트의 재사용을 용이하게 하고 시스템 개발 시간 단축과 개발비용의 감소를 유도하고자 하였다.

본 논문은 관련연구를 2장에 기술하였고, 3장에서는 본 논문에서 새롭게 제시한 "영역지향 CBD 방법론"에 대해 기술하였으며, 4장에서는 새로운 방법론을 적용한 사례를, 5장에서는 본 논문의 결론과 향후 연구 과제에 대해 간략하게 서술하였다.

2. 영역지향 프로그래밍(AOP:Aspect-Oriented Programming)

영역의 개념은 절차 중심 프로그래밍 기술과 객체 지향 프로그래밍 기술에서 충분히 처리될 수 없는 문제들을 다루기 위해 소프트웨어 개발 분야에서 소개되었다.

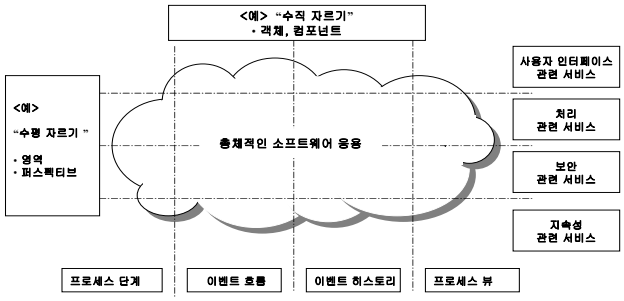
영역지향 프로그래밍은 하나의 기능이 프로그램의 여러 모듈들에 흩어져 있는 것을 모아서 영역이라는 새로운 프로그램 구조 내에 정의하게 된다. 전체적으로 볼 때, 어플리케이션의 각 클래스는 고유 기능만을 수행하고, 추가된 각 영역들이 횡단적인 기능들을 모아 처리함으로써 크로스컷팅 문제를 해결하면서 전체적인 프로그램을 이루는 형태를 만들게 된다. 횡단적인 기능이란 사용자 인터페이스, 처리, 지속성, 암호화, 배치, 메모리 관리, 협력 등과 같은

비 기능적인 제약이나 수준 높은 서비스들을 의미한다. 영역지향 프로그래밍의 장점으로는 첫째, 모듈간의 결합력과 개발 인력간의 상호 의존도가 약해져 개발이 용이하다. 둘째, 모듈 수정이 쉬워 기술의 변화에 효율적으로 대응할 수 있다. 셋째, 한 모듈을 보기 위해 다른 모듈들까지 보아야하는 필요성이 적어 코드의 이해가 쉽다. 넷째, 시스템의 안정성 향상과 성능의 향상에 기여할 수 있다.

3. 영역지향 CBD 방법론

영역지향 CBD 방법론은 현저히 높은 재사용성과, 뛰어난 신장성, 동적인 순응성이 존재하는 소프트웨어 컴포넌트를 개발하기 위해 본 논문에서 제시하는 새로운 방법론이다. 영역지향 CBD 방법론은 소프트웨어 개발 라이프사이클을 통한 소프트웨어의 전체적인 개발 과정에 적용되어 진다.

본 논문을 통해 제시한 영역지향 기술은 컴포넌트의 기능적인 특성을 나타내기 위해 수직으로 분해시킨 소프트웨어 컴포넌트를 다시 수평으로 잘라 크로스컷팅 이슈와 비 기능적인 특성을 나타낼 수 있도록 영역 표기법을 도입하는 기법을 제안하였으며, 이것을 기존의 CBD 방법론에 적용하였다.



(그림 1) 일반적인 개념의 컴포넌트와 영역 개념이 포함된 컴포넌트

보통 일반적인 영역(Aspect)은 사용자 인터페이스, 처리, 지속성, 암호화, 배치, 메모리 관리, 협력 등을 포함한다. 컴포넌트 개발자가 관련 영역 서비스를 확인 할 때, 영역지향 CBD 방법론은 다양한 체계의 영역에서 컴포넌트들 간의 상호작용에 대한 추론을 허용한다. 컴포넌트 개발자는 응용 컴포넌트가 가질 수 있는 크로스컷팅과 관련된 주요 시스템 레벨과 적절한 영역 세부 사항, 고수준의 컴포넌트 정보, 컴포넌트의 복구와 내부 검색 자료 교환을 위한 영역 특성들을 자세히 기술하여야 한다.

각 컴포넌트의 영역마다 일부 영역 정보를 가지고 있는 타 컴포넌트로 공급되거나 요청될 수 있다. 이때의 영역 상세 정보는 영역과 관련 있는 컴포넌트의 시스템 적인 특징을 보다 명확하게 묘사하는데 사용된다. 또한 영역 세부 사항은 하나 이상의 영역 세부 사항 특성이 있어서 영역 정보를 특성화 할 수 있다. 영역의 세부 속성들은 기능적이거나 비 기능적인 특징들과 관련될 수 있을 것이다. 예를 들면, 보안 영역을 위해 우리는 데이터 부호화 알고리즘, 암호화 알고리즘, 암호 키 길이와 키 타입 등 세부적인 속성들에 관심을 가질 것이다. 영역지향 CBD 방법론에서의 영역 개념은 컴포넌트의 기능적인 정보들을 담을 수 있을 뿐만 아니라, 컴포넌트들의 비 기능적인 제약 사항도 담을 수 있다. 영역지향 CBD 방법론의 영역은 컴포넌트의 크로스 컷팅 서비스들을 위해 더욱 효과적인 특징과 구현 방법을 지원함으로써 개발자와 사용자들에게 컴포넌트들에 대한 지식을 풍부하게 지원하는데 초점을 둔다.

영역지향 CBD 방법론의 장점으로는 첫째, 컴포넌트들이 다양한 전망(영역), 구조적 컴포넌트 요구사항과 설계의 풍부함을 제공한다. 둘째, 동적인 구성과 독립된 컴포넌트 상호작용을 돕는다. 셋째, 영역지향 컴포넌트 개발자들이 정확하고 완전하게 컴포넌트를 문서화 할 수 있도록 도와준다. 넷째, 영역을 가지고 있는 컴포넌트는 개발자와 최종사용자에게 컴포넌트의 다양한 관점을 줄 수 있게 한다. 다섯째, 컴포넌트의 영역 정보가 개발자들로 하여금 컴포넌트 색인과 저장을 용이하게 한다.

3.1 영역지향 컴포넌트 요구사항

영역지향 컴포넌트 요구사항은 기능적이거나 비 기능적인 요구들을 입증하고 지정한다. 컴포넌트 요구사항들의 명세를 규정하는 동안 개발자는 많은 영역 서비스들을 갖는 컴포넌트와 단지 약간의 영역 서비스들을 갖는 컴포넌트가 있다는 것을 알게 될 것이다. 때때로 하나 이상의 컴포넌트는 중복되는 영역들이 들어 있는 컴포넌트의 영역들을 제공받거나 요구할지도 모른다. 이러한 겹쳐진 영역들은 고수준 체계의 특징을 반영하는 것인데, 영역지향 컴포넌트의 요구사항 명세는 컴포넌트를 필요로 하는 기술자들에게 관련된 컴포넌트 속성들을 이해시키는데 매우 유용하다.

3.2 영역지향 컴포넌트 설계

요구사항을 명세 하는 동안 개발자들은 영역 체계

들로부터 기능적이고 비 기능적인 제약들을 확인한다. 이러한 제약들과 영역들은 영역을 갖는 컴포넌트 설계로 정련되는데 사용된다. 요구사항 수준의 세부적인 영역들은 소프트웨어로 세분화된 디자인 수준의 컴포넌트 영역들로 정련될 수 있는데 이것들은 디자인 수준의 컴포넌트 영역 서비스들을 특징짓는다. 예를 들면, 세분화된 디자인 단계의 영역 명세들은 사용자 인터페이스, 컴포넌트 지속성, 분배, 보안과 처리 모델 그리고 컴포넌트 구성을 말한다.

설계 단계에서, 개발자들은 구현 중립 요구사항들에서 서비스에 관련된 소프트웨어 컴포넌트들의 실행 전략을 규정하는 영역 세부사항들과 영역 세부사항 특성들로 정련할 것이다. 설계 단계에서 컴포넌트 서비스들은 관련된 영역을 문서화시키고 설계자가 이러한 예상 연결들을 추적하는 것을 허용하기 위해 다른 것들과 겹쳐진 영역들 또한 문서화한다.

3.3 영역지향 컴포넌트 구현과 런타임

영역지향 CBD 방법론을 사용해서 설계된 컴포넌트들은 어떠한 컴포넌트 기반 구현 기술이나 컴포넌트 기반 프레임웍을 사용함으로써 실행될 수 있다. 컴포넌트 영역은 인터페이스, 언어 반사 또는 디자인 패턴을 통해 실행 될 수 있다. 영역을 가지고 실행되는 컴포넌트는 각각의 다른 기능을 접근하고, 컴포넌트 상호간의 인터페이스 정의를 안내할 수 있게 하기 위해 관련된 컴포넌트들에 대한 테크닉을 제공할 수 있다. 하나의 컴포넌트 안에서 실행되는 영역 서비스들은 다른 컴포넌트들이나 최종사용자에 의해 런 타임에서 사용될 수 있다. 영역 서비스들은 컴포넌트 영역들을 질의하는데 사용되는 반면 컴포넌트 또한 컴포넌트의 구성과 확인을 위한 일관성 확인을 수행하는데 필요한 다른 컴포넌트 영역 정보를 질의 할 수도 있다. 영역 서비스들은 또한 런 타임 동안 컴포넌트의 능력을 높이는데 사용된다.

3.4 영역지향 컴포넌트 테스팅

영역 서비스를 가지고 실행되는 컴포넌트가 다른 응용으로 개발 또는 한 시스템의 부품으로 사용되었을 때 컴포넌트 요소의 영역이 얼마나 잘 정의되었고, 명세 되었으며, 설계되었는지를 테스트해 볼 필요가 있기 때문에 컴포넌트 영역들에 대한 시험 계획과 전략들을 만들어 낼 필요성이 있다. 그것은 컴포넌트 영역 기술자를 규정하고, 특별한 테스팅 에이전트를 만들어 사용함으로써 가능 할 것이다.

4. 적용사례

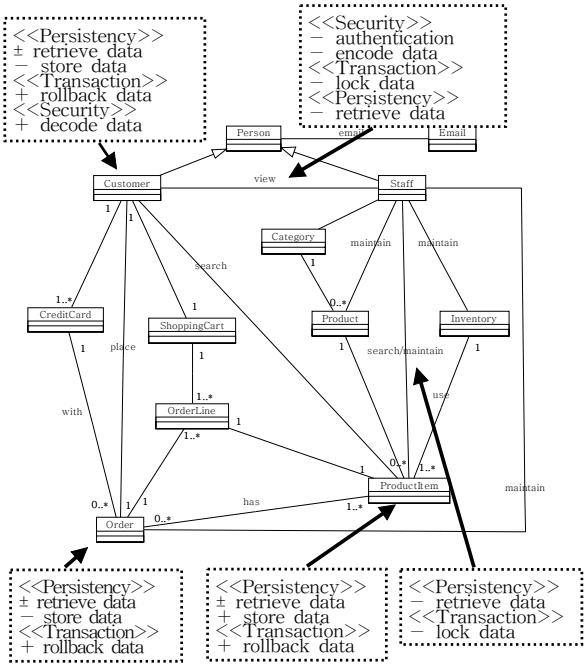
본 논문에서는 온라인 가전 판매 시스템에 새로운 방법론을 적용한 예를 들어 이해를 돕고자 하였다.

4.1 유즈케이스 모델링

웹 사용자는 멤버 등록을 통해 편리한 쇼핑을 즐기며, 개인정보를 유지할 수 있다. 권한이 허가된 직원은 제품, 주문, 직원정보, 물품목록 등의 관리 업무를 수행할 수 있다. 유즈케이스 모델링은 유즈케이스 다이어그램을 통해 나타낼 수 있으나 기존의 방법론 적용시와 상이점이 없어 생략하였다.

4.2 클래스 모델링

(그림 3)은 주석을 달아 고객 클래스가 영속성, 처리, 보안에 관한 영역을 필요로 하고 있음을 훨씬 분명하게 보여주고 있다.



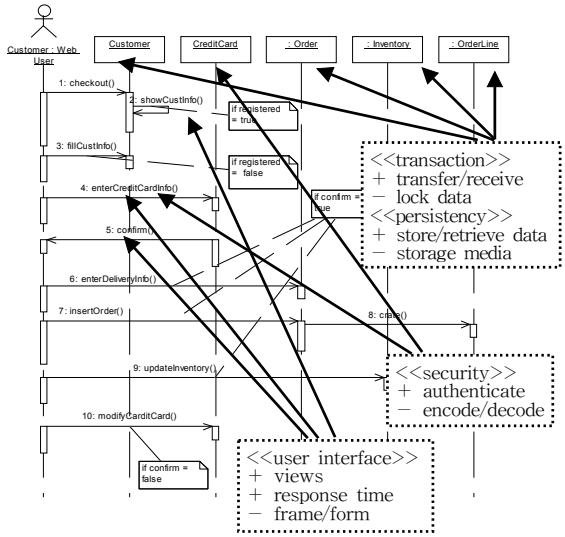
(그림 2) 영역지향 개념이 포함된 클래스 다이어그램

4.3 동적 객체 상호작용 모델링

영역	영역 세부 사항 및 속성과 간단한 추론
<<User Interface>>	<<+Provided>> 고객 정보/신용카드 정보 뷰 현재 처리 정보와 피드백 정보를 웹 사용자에게 보여준다. 이 서비스는 온라인 쇼핑 고객 같은 웹 사용자에 의해 요구되어 진다.
	<<–required>> 프레임/폼 세부 사항 속성은 GUI 컴포넌트나 HTML 태그일 것이며, HTML과 JSP, 스윙 컴포넌트에 의해 제공될 수 있다.
	<<+provided>> 응답 시간 5초 미만, 웹 사용자에 의해 요구되어 진다.
<<Transaction >>	<<+provided>> 데이터 송신/수신 운반 데이터에 트랜스포트 서비스가 필요하며, 이것은 코바/RMI와 같은 미들웨어 시설을 통하여 웬/랜 접속에 의해 제공될 것이고, 전송 속도가 제한되어질 것이다.

영역	영역 세부 사항 및 속성과 간단한 추론
<<Transaction>>	<<−required>> 락/롤백 데이터 병렬 접근을 피하고 에러 핸들링을 위해 사용한다.
<<Persistency>>	<<+provided>> 저장/회복 데이터 지속성과 회복에 객체들의 정보를 필요로 한다. 한꺼번에 저장할 수 있는 최대 저장 크기는 최고 10MB로 제한한다.
	<<−required>> 저장 매체 파일이나 데이터베이스이다.
<<Security>>	<<+ provided>> 접근 인증 사용자 체크아웃을 위해서 로그인이나 올바른 접근을 획득하기 위한 등록을 필요로 한다. 세부사항 속성은 패스워드 마스크이다.
<<Security>>	<<−required>> 부호화/복호화 데이터 64 베이스 암호화/복호화 또는 암호화 알고리즘을 사용한다.

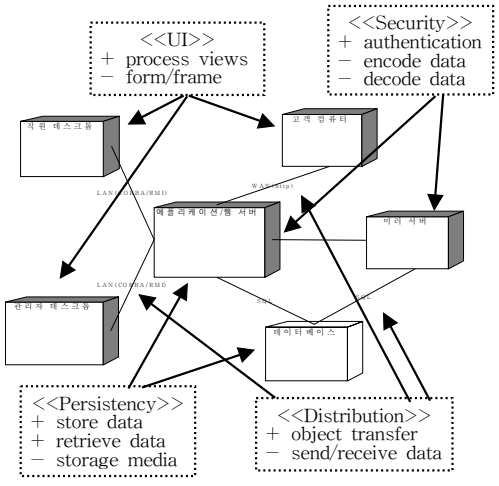
(표 1) 체크아웃 시퀀스다이어그램의 영역정보의 예



(그림 3) 영역 분석을 포함한 체크아웃 시퀀스 다이어그램

4.4 컴포넌트 디자인 명세

■ 배치 모델링



(그림 4) 영역지향 개념이 포함된 배치 다이어그램

5. 결론 및 향후 연구

CBD 방법론은 시스템 설계 영역에서 컴포넌트의 기능적인 설계와 서비스 실행에 집중되어 짐으로 인해 제 3의 개발자가 시스템 환경을 확장하거나, 기존 컴포넌트를 사용한 시스템 개발을 추진할 경우 컴포넌트를 이해하고 사용 가능한 컴포넌트로 만드는데는 많은 어려움이 있었다.

본 논문에서는 이러한 어려움을 극복하기 위한 방법으로 UML 다이어그램에서 주석과 스테레오 타입을 사용하여 영역 정보를 상세하게 기록할 수 있도록 한 영역지향 CBD 방법론을 제안하였다.

영역지향 CBD 방법론은 제3절에서 논한바와 같이 많은 이점을 제공하여 시스템 개발에 드는 노력과 비용의 경감을 가져오는 결과를 얻을 수 있다.

향후 과제로는 이미 만들어져 사용되고 있는 재사용 가능 컴포넌트들도 영역지향 CBD 방법론을 적용하여 영역 정보를 추가해 나감으로써 컴포넌트의 재사용성을 증대시키고 효율적이며 경제적인 시스템 개발의 기틀을 마련하는 것이다.

참고문헌

[1] A. W. Brown, K. C. Wallnau, "The Current State of CBSE", IEEE Software, Vol. 155, pp. 37-46, September/October, 1998.

[2] Alan W. Brown, "Large-Scale, Component-based Development, 2000.

[3] Booch G., Kozaczynski Wojtek, “Component-Based Software Engineering", IEEE Software, pp. 34-36, October, 1998.

[4] Choi, Jung pil, "Aspect-Oriented Programming with Enterprise JavaBeans", Enterprise Distributed Object Computing Conference. EDOC 2000. Proceedings Fourth International, pp. 252-261, 2000.

[5] Ho, W. M., Pennaneach, F., Jezequel, J. M. and Plouzeau, N., "Aspect-oriented Design with the UML", Proceedings of the ICSE2000 Workshop on Multi-Dimessional Separation of Concerns in Software Engineering, Jun 2000(Limerick, Ireland).

[6] Karl J. Lieberherr, "Early Definition of Aspect-Oriented Programming", Available at http://www.ccs.neu.edu/research/demeter/AOP/early-def/AP-AOP.html

[7] Rashid, A., Blair, L., "Editorial: Aspect-oriented Programming and Separation of Crosscutting Concerns", The Computer Journal, Vol. 46 Noo. 5, pp. 529-541, 2003.