개념적 컴포넌트 중심의 컴포넌트 모델링 기법 및 지원 도구의 설계

김민정 이우진 신규상
한국전자통신연구원 소프트웨어공학연구부 컴포넌트공학팀
e-mail : {minjkim, woojin, gsshin}@etri.re.kr

Component modeling methodology focused on conceptual component & design of a supporting tool

Min-Jeong Kim Woo-Jin Lee Gyu-Sang Shin
Component Engineering Team, Software Engineering Department, ETRI

요 약

소프트웨어의 대형화와 급격히 변화하는 소프트웨어 시장에 시기 적절히 대응하기 위해, 기존의 소프트웨어를 재사용함은 물론이고 재사용이 용이한 구조로 소프트웨어를 구성하여 다른 부분에 영향을 미치지 않고 특정 부분을 변경시킬 수 있는 솔루션이 요구된다. 이러한 기술에 부합되는 것이 재사용성, 대체가능성 등의 기능을 강조한 컴포넌트 기술이라 할 수 있겠다. 컴포넌트 기반 시스템을 설계하는데 있어서는 기존의 객체지향 접근방식을 포함하면서 컴포넌트의 고유의 특성을 반영하는 다른 접근방식을 필요로 하게 된다. 본 논문에서는 이러한 객체지향적 접근방식에 개념적인 컴포넌트 설계 방식과 이를 EJB 컴포넌트로 다시 상세화하는 컴포넌트의 모델링 프로세스와 이를 지원하는 도구에 대해 다룬다.

1. 서론

최근들어 IT 시장의 급성장으로 다양한 형태의 소프트웨어 요구되고 적시성을 맞추기 위해 소프트웨어의 개발 기간 또한 급속히 줄어들고 있다. 클래스 재사용에 중심을 두고 있는 객체지향 개발 방법론만으로는 이러한 시장의 요구사항들을 만족시키는데 한계를 느끼고 있으며 새로운 개발 기술을 필요로 하고 있다. 컴포넌트 기술은 재사용성, 적시성, 유지 보수성 등 IT 시장에서 요구하는 특성들을 어느 정도 만족시킬 수 있는 한가지 대안으로 인식되기 시작했다. 현재까지 소개된 컴포넌트 개념으로는 Sun 의 Enterprise JavaBeans(EJB)[1], OMG 의 CORBA Component Model(CCM)[2], 마이크로소프트의 COM, DCOM[3] 등을 들 수 있다. OMG 의 CCM 은 개념만 소개되어 있으며 아직까지 CCM 을 지원하는 플랫폼 아키텍처를 내놓지 못하고 있다. Sun 의 EJB 개념은 Java 2 Enterprise Edition(J2EE) 아키텍처 [4]로 제공되고 있으며 COM 과 DCOM 은 DNA 아키텍처와 웹 서비스(Web Services)를 지원하기 위해 최근에 소개된 .Net 아키텍처[5]에서 제공되고 있다. 이러한 컴포넌트 기술들은 컴포넌트 개념, 컴포넌트의 기본 단위, 구현 방법 등에서 조금씩 차이를 보이고 있어 사용자는 특정 플랫폼 아키텍처에 맞춰 컴포넌트를 정의하고 구현하여야 한다.

본 논문에서는 특정 영역의 고유 특성을 반영하고 구현 기술이라 볼 수 있는 플랫폼 아키텍처에 의존적이지 않는 재사용 가능한 개념적인 컴포넌트를 정의하고 이들간의 의존성을 명세할 수 있는 컴포넌트 모델링 기술과 개념적인 컴포넌트를 필요에 따라 EJB, DCOM 등의 특정 플랫폼 아키텍처 기술로 세분화하여 상세 설계할 수 있는 컴포넌트 설계 기술을 다루고 이를 지원하는 모델링 도구의 설계 및 구현을 제공하고자 한다. 이 논문의 구성은 다음과 같다. 우선, 제 2 절에서는 컴포넌트 생성 과정인 컴포넌트 모델링, 상세설계, 구현, 전개(Deployment) 과정에 대해 간략히 다룬다. 제 3 절에서는 개념적 컴포넌트들을 명세하는 컴포넌트 다이어그램에 대해 기술한다. 제 4 절에서는 컴포넌트 모델링 도구의 설계에 대해서 다루고 제 5 절에는 개발된 도구의 주요 화면을 보여주

고 결론과 향후연구에 대해 논한다.

2. 컴포넌트 모델링 및 생성 프로세스

컴포넌트 기반 소프트웨어 개발 프로세스는 도메인 분석, 컴포넌트 식별, 컴포넌트 설계, 컴포넌트 구현, 컴포넌트 시험 및 전개 과정으로 나뉘어진다. 그림 1 은 이러한 프로세스들을 보여주고 각 단계별로 필요한 모델링 산출물들을 오른쪽에 보여주고 있다.

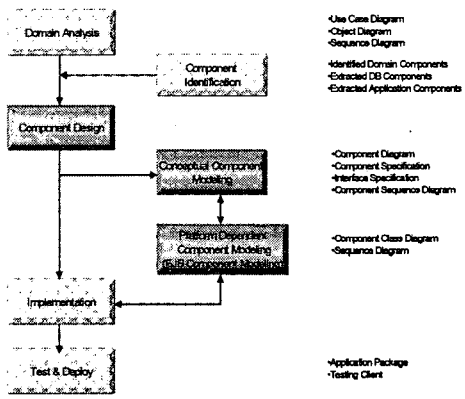

그림 1. 컴포넌트 모델링 및 개발 프로세스

우선, 마르미 III [6]에서 제안된 공통성 및 가변성 분석 방법을 통해 얻어진 어플리케이션 도메인의 영역 정보를 바탕으로 Use Case 모델과 객체 모델을 생성한다. 객체 모델에 기술된 객체들의 연관 관계 정보와 Use Case 에 나타난 객체들의 기능적 연관성 및 사용 횟수 등을 고려하여 개념적 컴포넌트 식별이 이루어진다. 식별된 개념적 컴포넌트들에 대해서 컴포넌트간의 의존성과 각 컴포넌트의 인터페이스를 명확히 정의하기 위해 컴포넌트 다이어그램과 순차도(sequence diagram)를 작성하고 최종적으로 컴포넌트 인터페이스를 정의한다. 컴포넌트 인터페이스가 정의되면 각각의 컴포넌트별로 독립적으로 내부 설계 및 구현이 가능하다. 컴포넌트 내부설계 및 구현은 컴포넌트 플랫폼 아키텍처의 특성을 고려하여야 하므로 EJB, DCOM, CCM 에 따라 조금씩 차이가 있다.

이 논문에서는 EJB 컴포넌트 설계 및 구현에 대해서만 언급한다. EJB 컴포넌트 내부 설계시에는 영역 분석에서 얻어진 객체들에 대해 EJB 특성에 맞게 클래스 다이어그램을 세분화하고 EJB 의 홈/원격 인터페이스를 빈 클래스에 추가한다. 이때 세분화된 클래스의 인터페이스를 명확히 정의하기 위해 클래스간의 순차도를 작성하기도 한다. EJB 컴포넌트의 내부 설계를 바탕으로 인터페이스 구현 코드의 자동 생성 및 클래스의 골격 코드 생성이 이루어지며 이들을 기반으로 하여 사용자가 비즈니스 로직을 직접 구현한다. 그리고 컴포넌트의 인터페이스의 기능 테스트를 원할 경우는 테스트 클라이언트 생성 과정을 거친다. EJB 컴포넌트를 어플리케이션 서버에 배치(deploy)하기 위해서는 먼저 DD 파일을 생성하고 Ejb-jar 파일로 컴포넌트를 묶어야 한다. 본 논문에서 초점을 두고 살펴볼 단계는 컴포넌트 설계 단계로서 개념적인 컴포넌트 모델링 및 플랫폼 의존적인 컴포넌트 모델링 단계가 되겠다. 현재 플랫폼 의존적인 컴포넌트 내부 설계 단계에서는 Sun 사의 J2EE 아키텍쳐 기반의 EJB 컴포넌트들이 사용된다.

3. 컴포넌트의 설계 방법

본 논문에서 제안하는 컴포넌트의 모델링 단계는 개념적인 컴포넌트 모델링 단계와 플랫폼에 의존적인 컴포넌트 상세설계단계로 이루어진다. 개념적 컴포넌트가 의미하는 바는 하나 이상의 인터페이스를 가지며, 여러 클래스들로 내부가 구성되며 독립적으로 기능을 수행할 수 있는 전개(Deployment)단위라고 할 수 있다. 만약 여러 개념적 컴포넌트로 구성되어있는 시스템에서 내부 로직이 변경될 경우 인터페이스의 변경 없이 컴포넌트의 내부 설계만 변경하여 서버에 전개하게 되므로 전체 시스템을 신경 쓸 필요가 없게 된다. 또한 개념적 컴포넌트 단위로 배포하게 되면 인터페이스의 정보만으로 컴포넌트를 사용할 수 있게 된다. 업그레이드 역시 동일한 인터페이스를 가진 개념적 컴포넌트로 교체하면 되므로 전체 시스템에 영향을 미치지 않게 된다. 제안하는 모델링 프로세스는 다음과 같다.

1. 도메인 분석을 통해 개념적 컴포넌트를 식별한다.
2. 각 컴포넌트 인터페이스를 정의한다.
3. 컴포넌트 인터페이스에 사용된 객체를 정의한다.
4. 컴포넌트 의존성을 파악한 후 연결한다.
5. 각 개념적 컴포넌트 마다 컴포넌트 클래스 다이어그램을 이용하여 컴포넌트를 상세설계 한다.

우선 개념적인 컴포넌트 모델링 단계에서는 재사용을 고려하여 최대한 독립적으로 실행 가능한 모듈들로 설계하여야 한다. 이 단계에서는 재사용성이 높은 컴포넌트들을 식별하여 개략적으로 설계하고 컴포넌트의 인터페이스를 명확하게 정의하는 것이 목적이다. 인터페이스를 식별하기 위해 컴포넌트간의 순차도를 참조할 수도 있다. 컴포넌트의 인터페이스에는 컴포넌트에 접근하기 위해 사용되는 오퍼레이션들이 파라미터 및 리턴타입들과 함께 명시가 되며 이 오퍼레이션 정보는 플랫폼 의존적인 컴포넌트 모델링시에 자동적으로 알맞은 클래스들로 변환되게 된다. EJB 컴포넌트의 경우 외부에서의 접근을 담당하는 인터페이스 세션 빈으로 변화하게 된다.

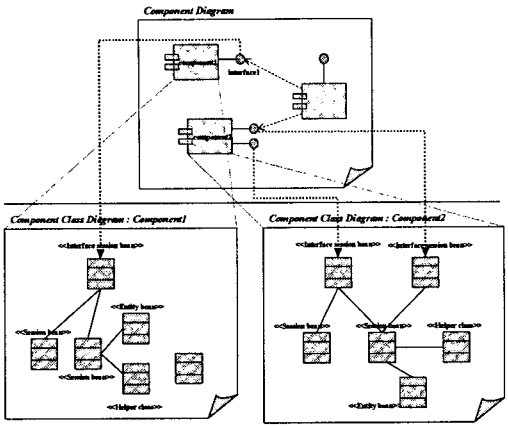

그림 2. 개념적 컴포넌트 모델링 및 EJB 컴포넌트로의 세분화

그림 2 는 개념적 컴포넌트와 EJB 컴포넌트와의 매핑 관계를 나타내는 그림이다. 개념적 컴포넌트의 모델링 단계에서는 아키텍쳐를 각 개념적 컴포넌트들로 분리시킨 후, 시스템에서의 역할을 고려하여 컴포넌트의 인터페이스를 정의하게 된다. 인터페이스에 사용되는 객체들에 대해서도 이 단계에서 정의하게 되며 만약 한 컴포넌트가 다른 컴포넌트와 의존관계에 있을 경우 그 부분을 명시한다. 개념적 컴포넌트 모델링 단계에서 정의된 인터페이스의 정보는 EJB 컴포넌트 모델링 단계에서 기능연결을 위한 클래스들로 그림 2 에서와 같이 자동적으로 변환되게 된다. 만약, 하나의 컴포넌트가 두개 이상의 인터페이스를 가질 경우 컴포넌트 클래스 다이어그램에 인터페이스의 개수와 동일한 인터페이스 세션 빈이 생성된다. 이 경우는 여러 역할에 따라 동일기능을 하는 컴포넌트를 다시 설계하기보다는 내부 클래스를 공유하게 함으로서 보다 효율적으로 컴포넌트를 개발할 수 있다.

컴포넌트 상세설계 단계에서는 컴포넌트 클래스 다이어그램을 사용하여 플랫폼에 의존적인 컴포넌트의 내부를 설계하게 된다. 상세설계 단계의 첫번째 단계에서는 우선 컴포넌트의 인터페이스 정의 과정에서 생성된 인터페이스의 오퍼레이션들이 EJB 컴포넌트의 속성에 맞는 인터페이스 세션 빈으로 자동적으로 변경된다. 인터페이스 세션빈은 외부에서 컴포넌트로의 접근을 담당하는 빈으로서 이 인터페이스 세션 빈을 통해 컴포넌트 내부에 접근할 수 있다. 만약 컴포넌트에서 담당하는 역할에 따라 인터페이스가 분리되어 한 개 이상의 인터페이스가 사용될 경우 인터페이스 세션빈은 역할에 따라 인터페이스 개수만큼 존재하게 되며 내부 설계에 사용된 동일한 클래스를 참조할 수도 있게 된다.

컴포넌트 클래스 다이어그램을 이용한 컴포넌트 상세설계에서 사용되는 노테이션들은 기존 클래스 다이어그램에서 사용되는 노테이션들을 확장하여 사용하였다. 또한 EJB 특성에 맞는 몇가지 클래스를 추가하였다. 추가된 클래스는 외부에서의 컴포넌트로의 접근을 담당하는 인터페이스 세션 빈과 EJB Session/Entity Bean, DB 와 직접 연결되어있는 Extracted DB Entity Bean, 그리고 일반 Helper Class 등이 있다.

4. 컴포넌트 모델링 지원도구의 설계

컴포넌트 설계를 위한 편집기는 그림 3 과 같은 구조를 가지며, 개념적 컴포넌트 모델링을 위한 컴포넌트 다이어그램 편집기 및 인터페이스 명세 편집기, 컴포넌트 상세설계를 위한 컴포넌트 클래스 다이어그램 편집기 및 컴포넌트/개체 순차도 편집기들로 구성된다. 컴포넌트 다이어그램 편집기는 컴포넌트 클래스 다이어그램 편집기와 모델을 공유하며, 순차도 편집기도 컴포넌트 및 컴포넌트 클래스 다이어그램 편집기를 참조하게 된다. 따라서 컴포넌트 다이어그램 편집기에서 정의된 인터페이스의 경우 직접적으로 컴포넌트 클래스 다이어그램 편집기 내부의 인터페이스 세션 빈 클래스로 변경되게 된다. 또한 컴포넌트 클래스 다이어그램 편집기의 경우 SRE(Simultaneous Roundtrip Engineering) 모듈과 연결되어 Java 코드를 자동적으로 생성하게 된다.

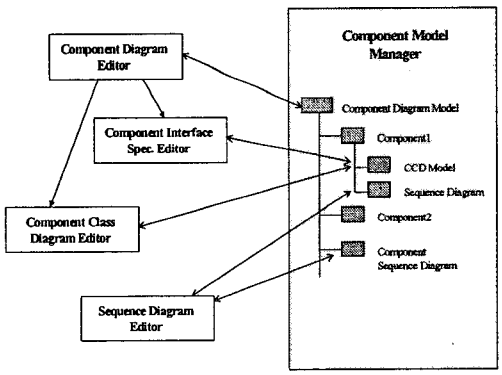

그림 3. 컴포넌트 모델링 지원도구의 구조

5. 결론 및 향후 연구

본 논문에서는 재사용성을 강조한 컴포넌트를 설계하기 위한 모델링 프로세스를 제안했다. 개념적 컴포넌트 기반의 컴포넌트 설계 방법을 제안하였으며 J2EE 플랫상에서 동작하는 EJB 컴포넌트에 맞는 컴포넌트 상세 설계 방법을 제안하였다. 또한 이 개념을 적용시킨 CASE 도구를 설계 및 개발하였다. 그림 4 는 현재 개발중인 컴포넌트 설계지원도구(COBALT_GEN v1.0)의 일부인 컴포넌트 다이어그램의 모습이며, 그림 5 는 컴포넌트의 상세설계에 사용되는 컴포넌트 클래스 다이어그램의 모습이다.

향후 연구로는 우선 지속적인 도구의 검증을 통해 효용성을 입증해야 하며 컴포넌트 설계를 위한 보다 체계적인 방법에 관한 연구도 지속되어야 한다.

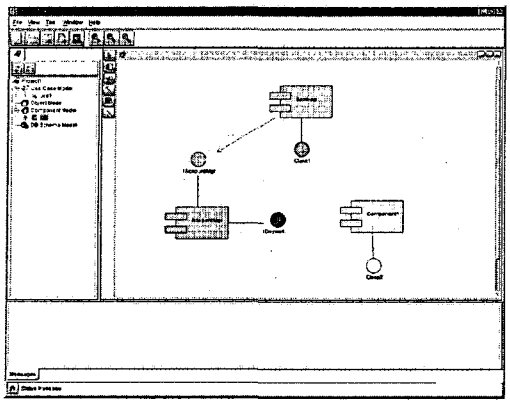

그림 4. 컴포넌트 생성지원 도구 (COBALT_ GEN v1.0)의 컴포넌트 다이어그램 -

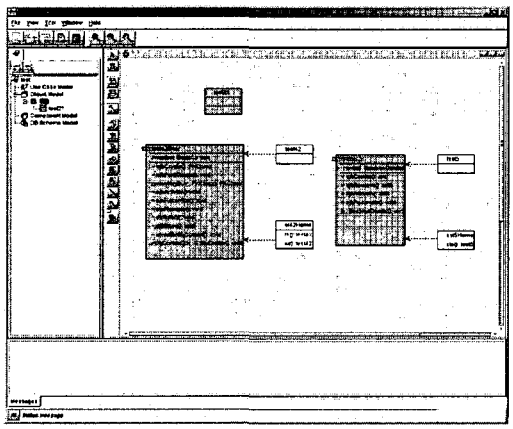

그림 5. 생성지원 도구(COBALT_ GEN v1.0)의 컴포넌트 클래스 다이어그램

참고문헌

[1] Sun, Enterprise JavaBeans Specification, Version 1.1, Sun Microsystems Inc, 1999

[2] OMG, CCM Revised Submission, OMG TC Document orbos/99-07-01, 1999.

[3] Microsoft, DCOM Technical Overview, URL: http://msdn.microsoft.com/library/backgrnd/html/msdn_dcomtec.htm, 1996.

[4] Sun, Designing Enterprise Applications with the JavaTM 2 Platform, Enterprise Edition, Version 1.0, Mar. 2000.

[5]"An Introduction To Microsoft .Net", http://www.microsoft.com/net/intro.asp.

[6] "컴포넌트 기반 개발방법론 마르미-III", 한국전자통신연구원, 2001

[7] "EJB 컴포넌트의 모델링 및 생성 지원 도구의 설계", 정보과학회 가을학술발표논문집 제 27 권 2 호, 2000