□특집□

CORBA 컴포넌트 모델의 분석 및 전망

최 성 운† 홍 선 주††

◈ 목 차 ◈

1. 서 론

소프트웨어 산업은 하드웨어 산업에 비해 역사가 짧을 뿐만 아니라, 생산성 또한 매우 낮다. 이러한 문제를 해결하기 위해 1970년경부터 소프트웨어의 컴포넌트 조립 생산에 대한 많은 시도가 있었으나, 모듈화, 표준화, 독립화 등의 문제를 효율적으로 해결하지 못하여 일부 기능을 재사용[1] 하는데 그치고 있었다. 1980년대 이후 객체 기술[2] 및 컴포넌트 기술[3]의 등장은 이러한 문제들을 해결하여 본격적인 소프트웨어 조립 생산을 가능하게 하고 있다.

소프트웨어 재사용의 핵심기술은 표준화된 인터페이스를 갖추는 기술, 컴포넌트를 포장하는 기술 및 이들을 유연하게 조립하는 기술 등이며, 이러한 기술적 배경을 중심으로 컴포넌트의 기능 및 역할 등 컨텍스트(Context)의 표준화 또한 중요한 이슈로 등장하고 있다.

2. 컴포넌트 모델

소프트웨어 컴포넌트란 정의된 컨텍스트와 계약된 인터페이스를 통한 조합이 가능한 단위를 말한다. 소프트웨어 컴포넌트는 서드 파티들에 의한 조합을 목적으로 독립적으로 배포될 수 있다.[4] 즉 컴포넌트는 표준화된 인터페이스를 바탕으로 독립적 재사용이 가능한 단위를 말한다.

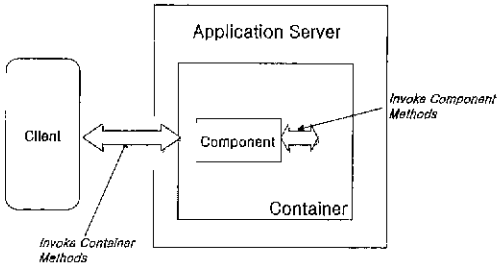
컴포넌트를 기반으로 하는 컴포넌트 모델은 일반적으로 시스템의 서버 측 재사용 아키텍처를 말한다. 컴포넌트 모델은 컴포넌트가 조립 수행될 수 있는 런타임 환경 및 트랜잭션 처리나 보안, 데이터베이스 연결 등의 분산기반 기능을 정의한다.

2.1 컴포넌트 모델 구조

일반적으로 컴포넌트 모델은 다음과 같은 기본 구조를 가진다.(그림 1)

- 컴포넌트 : 컴포넌트는 표준화된 컨텍스트 및 인터페이스를 가진다. 컨텍스트란 컴포넌트가 제공하는 기능이나 역할 등의 내용이며, 인터페이스란 컴포넌트의 기능을 외부에 제공하기 위해 필요한 표준화된 규약이다. 인터페이스는 표준화된 IDL(Interface Definition Language)을 사용하여 명세화된다.
- 컨테이너 : 컨테이너는 컴포넌트 구현을 위한 서버 측 런타임 환경이다. 컴포넌트 인스턴스의 생성 및 소멸 등의 관리를 담당하며, 클라이언트가 컴포넌트를 참조하기 위해 공통적

† 정 회 원 : 명지대학교 컴퓨터공학부 교수
†† 정 회 원 : 명지대학교 컴퓨터공학과 박사과정



(그림 1) 컴포넌트 모델의 기본구조

으로 필요한 서비스를 제공한다.

- 어플리케이션 서버 : 어플리케이션 서버는 멀티 프로세싱이나, Load-Balancing, Device Access와 같은 시스템 서비스 및 네이밍(Naming), 트랜잭션 서비스, 컨테이너 관리 등을 제공하는 프로세스이다.

2.2 컴포넌트 모델의 종류

컴포넌트 관련 기술의 세계적 표준화는 OMG (Object Management Group), Microsoft, Sun Microsystems 세 조직을 중심으로 이루어지고 있다. OMG는 분산 객체 기술의 표준인 CORBA(Common Object Request Broker Architecture)를 기반으로 컴포넌트 기반 시스템의 구조를 정의한다. Microsoft는 COM(Component Object Model), OLE(Object Linking and Embedding)등 Windows 플랫폼을 기반으로, Sun은 인터넷 기반의 Java 기술을 기반으로 컴포넌트 기반 시스템의 구조를 정의한다.

1) OMG의 CORBA Component Model(CCM)

CCM은 CORBA 3.0 명세에 정의되어 있으며, CORBA기반의 분산 엔터프라이즈 어플리케이션의 개발 패턴을 종합한 모델이다. CCM의 표준 명세를 정의하기 위한 RFP(Request for Proposal)가 1997년 6월에 OMG로부터 제안되었고, IBM, Oracle, BEA Systems 등 여러 회사가 참여하여 제안작업을 수행하여 현재 거의 완성된 명세가 나와있는 상태이다.

2) 마이크로소프트사의 COM+[5]

마이크로소프트사의 COM+는 COM과 MTS (Microsoft Transaction Server)를 통합시킨 컴포넌트 모델이다. 윈도우즈 NT를 위한 미들웨어 컴포넌트 모델인 COM에 기반을 두고, 그 위에 MTS와 MSMQ (Microsoft Message Queue)등의 서비스를 통합하였다.

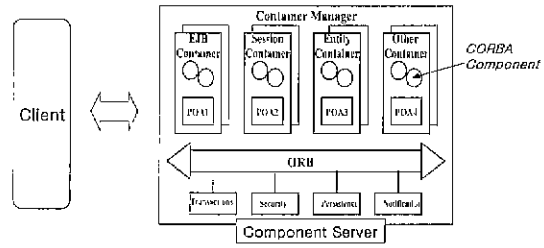
3) Sun Microsystems사의 Enterprise JavaBeans [6]

EJB(Enterprise JavaBeans)는 서버 컴포넌트 모델을 정의하며, 엔터프라이즈 레벨의 컴포넌트를 생성하는 방법에 대한 메커니즘을 제공한다. EJB는 엔터프라이즈 어플리케이션을 개발하기 위한 표준 플랫폼인 J2EE(JavaTM2 Platform, Enterprise Edition)의 컴포넌트 모델이다. J2EE는 EJB에서 사용하는 서비스의 API 및 표준 클라이언트를 포함하는 인프라 스트럭처를 정의한다.

3. CORBA Component Model [7][8]

3.1 CCM 주요 구성요소

CCM 아키텍처의 주요 구성요소는 다음과 같다.(그림 2)



(그림 2) CCM 아키텍처

3.1.1 컴포넌트

1) 컴포넌트의 구성

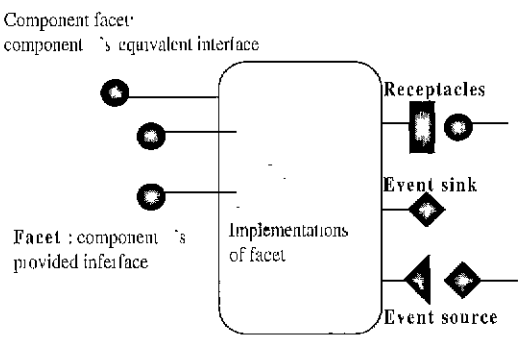
컴포넌트는 기본적으로 내부 데이터를 정의하는 속성(Attribute)과 특정 서비스를 구현하는 메소

드로 구성된다. 또한 일반적으로 클라이언트로부터 컴포넌트의 내부를 캡슐화(Encapsulation)하는 인터페이스 부분과 구현 부분을 분리하여 정의한다.[6] 컴포넌트 인터페이스는 컴포넌트를 참조하는 클라이언트와의 규약을 의미한다. 인터페이스는 컴포넌트의 메소드와 파라미터를 정의하며 클라이언트는 인터페이스를 통해서 컴포넌트가 제공하는 메소드의 결과만을 참조하게 된다. 컴포넌트의 구현 부는 객체가 제공하는 주요 프로그래밍 로직 부분으로, 여러 알고리즘과 데이터 등을 포함한다. 이 데이터들은 클라이언트에게 공개되지 않는다.

CCM에서는 컴포넌트의 인터페이스를 포트(Port)라 하며, 포트는 컴포넌트와 어플리케이션을 구성하는 다른 요소들이나 클라이언트와의 상호작용을 담당한다.

포트에는 네 가지 타입이 있다. (그림 3)

- Facets : 컴포넌트가 클라이언트와 상호작용하기 위해 제공하는 인터페이스. Facet 중에서 컴포넌트가 정의되기 위해 필요한 공통 인터페이스는 Component Facet으로 정의된다.
- Receptacles : 컴포넌트간의 상호작용을 위한 인터페이스
- Event Sources : 외부로 이벤트를 보내기 위한 커넥션 포인트
- Event Sinks : 외부에서 발생한 이벤트를 받아들이기 위한 커넥션 포인트



(그림 3) 컴포넌트 포트의 타입

2) 컴포넌트의 종류

컴포넌트는 상태의 지속성과 관련하여 네가지 유형으로 구분된다.

- 서비스(Service) 컴포넌트

단일 메소드만을 제공하는 컴포넌트이며, 상태나 식별성을 갖지 않는다. 서비스 컴포넌트의 생성과 소멸은 클라이언트에 의해 호출되는 단일 메소드에 의해 결정된다. 클라이언트가 컴포넌트의 레퍼런스를 저장하고 있지 않으므로, 다음 메소드를 호출할 때 또 다시 해당 컴포넌트의 레퍼런스를 참조해야 한다.

- 세션(Session) 컴포넌트

클라이언트와 상호작용 하는 세션동안만 일시적으로 상태를 유지하며, 지속적인 식별성을 갖지 않는다. 세션 컴포넌트의 생성과 소멸은 그것을 참조하는 단일 클라이언트에 의해 결정된다. 컴포넌트가 세션 내에서 상태를 유지하는 동안에는 클라이언트가 컴포넌트의 레퍼런스를 알고 있기 때문에 컴포넌트내의 메소드를 여러번 반복해서 호출할 수 있다.

- 프로세스(Process) 컴포넌트

비즈니스 프로세스를 모델링하기 위한 컴포넌트이며, 상태를 지속적으로 유지하며, 식별성을 갖는다. 트랜잭션적인 행위적 특성을 가지며 여러 클라이언트에 의해 공유될 수 있기 때문에, 프로세스 컴포넌트의 생성과 소멸은 그것을 참조하는 클라이언트들에 의해 결정된다. 클라이언트는 지속적으로 프로세스 컴포넌트의 레퍼런스를 저장할 수 있다.

- 엔티티(Entity) 컴포넌트

비즈니스 엔티티를 모델링한 컴포넌트이며, 프로세스 컴포넌트의 특성을 모두 갖는다. 프로세스

컴포넌트와의 차이점은 클라이언트가 주 키(Primary Key)를 통해 컴포넌트를 식별할 수 있다는 점이다.

3.1.2 컨테이너

컨테이너는 ORB, POA, CORBA 서비스 등과 함께 컴포넌트를 구현하기 위한 서버 측 환경을 제공한다. 컨테이너는 그것이 관리하는 컴포넌트의 종류에 따라 크게 7가지 범주로 나뉘어진다. CCM에서는 EJB 환경과의 통합을 위해 EJB 컴포넌트를 위한 컨테이너를 따로 정의한다.

- 서비스 컨테이너 : 서비스 컴포넌트 관리
- 세션 컨테이너 : 세션 컴포넌트 관리
- 프로세스 컨테이너 : 프로세스 컴포넌트 관리
- 엔티티 컨테이너 : 엔티티 컴포넌트 관리
- EJB Session 컨테이너 : EJB 세션 빈 관리
- EJB Entity 컨테이너 : EJB 엔티티 빈 관리
- Empty 컨테이너 : 컴포넌트 구현을 가능하게 하는 표준 CORBA 3.0 인터페이스를 생성

3.1.3 컴포넌트 서버

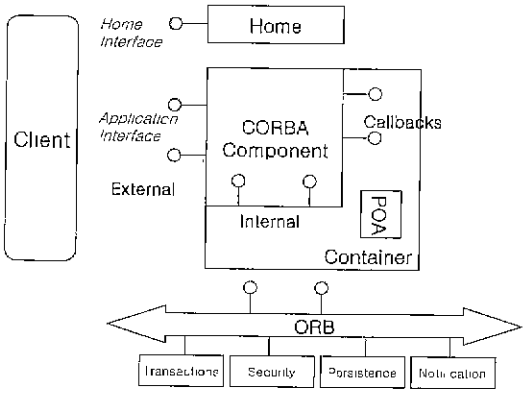
컴포넌트 서버는 다수의 컨테이너를 제공하는 하나의 프로세스로서 ORB 및 CORBA 서비스를 제공하며 컨테이너의 관리를 담당한다. 컴포넌트 서버와 컨테이너는 컴포넌트가 배포될 때 배포 어플리케이션에 의해 생성된다.

3.2 CCM 프로그래밍 모델

분산 엔터프라이즈 서버 컴퓨팅 아키텍처를 정의한 CCM은 컨테이너 프로그래밍, 컴포넌트 구현, 컴포넌트 패키징 및 배포, EJB 매핑 및 통합에 관련된 모델들을 제공한다.

3.2.1 컨테이너 프로그래밍 모델

컨테이너 프로그래밍 모델은 컴포넌트의 런타임 환경을 정의하기 위해 컨테이너가 제공하는 API 프레임워크의 구체적인 내용을 기술한 것이며, 그 구조는 다음과 같다.(그림 4)



(그림 4) 컨테이너 프로그래밍 구조

- External API 타입

컴포넌트의 External API 타입은 클라이언트와 컴포넌트 개발자 사이의 규약이다. External API 타입은 컴포넌트의 메소드 수행을 위한 어플리케이션 인터페이스와 클라이언트에게 어플리케이션 인터페이스의 참조(Reference)를 넘겨주기 위한 홈(Home) 인터페이스 두가지로 구분된다.

- 컨테이너 API 타입

Internal 인터페이스와 Callback 인터페이스로 구성되는 컨테이너 API는 컴포넌트 개발자와 컨테이너 사이의 규약이다. Internal 인터페이스는 컴포넌트에게 컨테이너의 서비스를 제공하기 위한 인터페이스이며, Callback 인터페이스는 컨테이너가 컴포넌트에 의해 구현된 메소드를 사용하기 위한 인터페이스이다.

컨테이너를 통해 다음과 같은 서비스들이 컴포넌트에 제공된다

- 트랜잭션(Transactions)
- 보안 (Security)
- 이벤트 (Events)

- 지속성 (Persistence)

3.2.2 컴포넌트 구현 모델

CCM에서는 컴포넌트를 구현하기 위한 프로그래밍 모델로 CIF(Component Implementation Framework)를 제공한다. CCM은 컴포넌트 및 컴포넌트 홈의 구현을 CIDL(Component Implementation Definition Language)이라는 선언적인 언어를 사용하여 표현한다. CIDL은 플랫폼이나 사용언어에 독립적으로 컴포넌트의 행위를 정의할 수 있게 한다. CIDL 컴파일러를 이용하여 CIDL을 컴파일하면, 네비게이션이나 컴포넌트 식별, 활성화 등과 같은 컴포넌트의 기본적인 행위들을 수행하는 프로그래밍 스켈레톤이 생성된다. 컴포넌트 개발자는 생성된 스켈레톤 코드를 바탕으로 컴포넌트의 기능적 특성을 구현하기 위한 코드를 추가하여 컴포넌트의 구현을 완성한다.

3.2.3 패키징 및 배포 모델

CCM은 EJB와 마찬가지로 패키징과 배포에 관한 정보를 나타내기 위해 XML(eXtensible Markup Language)을 사용한다. XML은 전자문서의 내용을 구조화하기 위한 국제 표준이므로, 이기종의 플랫폼에서 인식이 가능하다.

1) 컴포넌트 어셈블리와 패키징

컴포넌트 패키지는 단일 컴포넌트, 컴포넌트 어셈블리 패키지는 연관성을 가진 다수의 컴포넌트를 배포하기 위한 매개체이다. 컴포넌트들은 zip 압축파일 형태로 패키지화 되며, 각각의 내부 정보를 XML로 기술한다.

2) 컴포넌트 배포

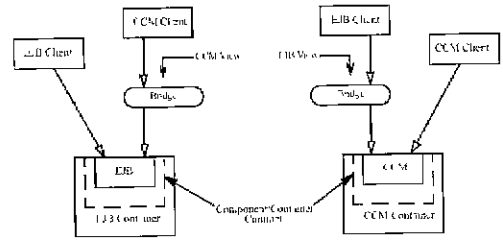
컴포넌트 패키지와 어셈블리 패키지는 배포 툴이나 어플리케이션을 사용하여 네트워크상에 타겟 호스트에 배포된다. 어셈블리 패키지 내에 있는 컴포넌트들은 서로 다른 위치에 배포가 가능하다.

XML로 기술된 패키지 내부 정보와 사용자가 제공하는 정보가 조합되어 배포 툴의 입력 정보가 되고, 배포 툴은 그 정보를 바탕으로 컴포넌트 서버, 컨테이너, 컴포넌트 홈과 인스턴스를 생성하고 활성화한다.

3.2.4 EJB 매핑 및 통합 모델

CCM에서는 동일한 어플리케이션에서 EJB와 CORBA 컴포넌트를 통합하여 사용할 수 있도록 하기 위한 구조를 제공한다.

각 EJB 컴포넌트를 관리하기 위한 컨테이너를 따로 정의하며, EJB 정의와 매핑이 가능하도록 IDL을 확장하였다. CCM과 EJB가 혼합된 모델에서는 브리지(Bridge)가 각기 다른 시스템의 인터페이스간의 매핑을 가능하게 해준다.(그림 5) CCM 클라이언트에서 EJB 컴포넌트를 참조하기 위해서는 EJB 정의가 CORBA IDL로 매핑되어야 하고, 런타임 시 CCM 클라이언트에 의해 호출된 CCM 메소드는 브리지에 의해 EJB 메소드로 변환된다.



(그림 5) EJB와의 통합모델

4. 결 론

이제까지는 CCM의 구조 및 개발 환경에 대해 살펴보았다. 하지만 재사용 가능한 컴포넌트의 개발은 단순히 그 구조 및 환경의 개발에 이루어지는 것은 아니다. 기술적 구조 및 환경의 표준화

작업과 함께, 컴포넌트가 사용되어지는 응용 영역별 컴포넌트의 기능 및 역할의 표준화가 이루어져야 한다. 위에서 정의한 컴포넌트 모델의 구조 및 환경을 기반으로 응용 영역별로 사용되어져야 할 컴포넌트의 기능 및 역할이 대다수의 개발자에게 수용될 수 있는 형태로 표준화 작업이 이루어져야 하는 것이다.

이러한 문제를 해결하기 위해 현재 OMG에서는 응용 영역별 컴포넌트 기능 및 역할의 표준화를 위한 DTF(Domain Task Forces)를 구성하여 활동하고 있으며, 그 종류는 다음과 같다.

- Business Object DTF
- Electronic Commerce DTF
- Finance DTF
- Manufacturing DTF
- Healthcare(CORBAmed) DTF
- Telecommunication DTF
- Transportation DTF
- Life Science Research DTF

Microsoft의 COM+, Sun의 EJB 등 컴포넌트 모델들은 나름대로의 장단점을 가지며 시장점유율을 넓혀 나가고 있다. 그러나 재사용을 극대화시키는 요소가 궁극적으로는 응용 영역별 표준화임을 감안할 때, 현재 시장 점유율에서 우위를 차지하고 있는 Microsoft사의 COM+나 Sun의 EJB보다는 800여개 이상의 컴포넌트 관련기업의 컨소시엄 형태로 이루어진 OMG의 CCM이 EJB 및 COM+를 수용하여, 세계 컴포넌트 모델의 표준을 이룰 것으로 예상된다.

참고문헌

[1] C. McClure, Software Reuse Techniques: A Guide to Adding Reuse to the Software Process, Extended Intelligence, Inc., 1996.

[2] Booch, G., "Object Solutions-Managing the Object-Oriented Project", Addison-Wesley, 1995.

[3] R. Johnson, "Frameworks= (Components+Patterns)", CACM, Vol.40. No.10, Oct.1997, pp.39-42.

[4] Clemens Szyperski, "Component Software", Addison Wesley, 1998.

[5] White Paper - Object - Oriented Software Development Made Simple with COM + Runtime Services, Mary Kirtland, 1997, URL : http://www.microsoft.com/msj/1197/ complus.htm

[6] Ed Roman, "Mastering Enterprise JavaBeans and the J2EE", John Wiley & Sons, 1999.

[7] Jon Siegel, "CORBA 3 Fundamentals and Programming", John Wiley & Sons, 2000.

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

최 성 문

1985년 한국 외국어대학(상학학사)
1988년 미국 오레곤 주립 대학 (공학석사)
1992년 미국 오레곤 주립 대학 (공학박사)

2000년 현재 명지대학교 컴퓨터공학부 교수, OMG KSIG 부회장, 쌍용정보통신 기술자문, ONC Technology 기술자문, 한국정보컨설팅 기술자문
관심분야 : 컴포넌트 프레임워크, 객체지향 소프트웨어공학
E-mail : choisw@mju.ac.kr

홍 선 주

1997년 명지대학교 컴퓨터공학과 (공학학사)
1999년 명지대학교 대학원 컴퓨터 공학과 (공학석사)
2000년 현재 명지대학교 컴퓨터 공학과 박사과정 재학중

관심분야 : 객체지향 소프트웨어공학, 분산시스템, 컴포넌트 기술
E-mail : hongsj@mju.ac.kr