□특집□

CBD 지원 도구의 핵심 기능 (Major Features of a Component-Based Development Supporting Tool)

권 오 천† 신 규 상††

◈ 목 차 ◈

1. 서 론

현재 몇몇 컴포넌트 기반 개발(CBD: Component-Based Development)을 지원하는 도구들이 해외에서 출시되어 국내외에서 사용되고 있다. 대표적인 CBD 지원 도구는 Computer Associates사의 Cool Products, Compuware사의 Uniface, IBM사의 SanFrancisco Framework, Select사의 SCF(Software Component Factory), TogetherSoft사의 Together 등이 있다.

CA사의 Cool 도구는 컴포넌트와 어플리케이션 아키텍처의 모델링 도구로서 행위분석(Behavioral analysis)과 인터페이스 기반 설계(Interface Based design)의 개념을 사용한다. 이것은 Cool:Gen(코드 생성), Cool:Plex(설계 패턴), Cool:Jex(객체지향 방법론), Cool:Biz(데이터 모델링)로 구성되며, EJB (Enterprise JavaBeans) 빈을 시각적인 환경 하에서 생성해 주는 Cool:Joe도 발표되었다.

Compuware사의 Uniface는 컴포넌트 구축, Legacy Wrapping, 컴포넌트 조립, 컴포넌트 배치 등의 모듈로 구성된다. 이 도구는 생성된 컴포넌트를 선택하고 이들을 인터페이스를 통해 서로 링크시켜, 어플리케이션을 통합하는 그래픽한 접근 방식을 제공한다.

IBM의 SanFrancisco Framework은 프레임워크 기반의 개발도구로서 인터넷, 인트라넷 및 엑스트라넷과 같은 컴퓨팅 환경의 중요한 비즈니스 솔루션의 빠른 개발을 위해 이용될 수 있는 서버 측(Server Side)의 EJB 컴포넌트를 제공한다. 이 프레임워크는 7200 클래스, 54,500개의 메소드로부터 1,121개의 공통 비즈니스 객체(Common Business Objects)와 핵심 비즈니스 프로세스(Core Business process)를 구성하는 컴포넌트를 제공한다.

TogetherSoft사의 Together 4.0은 Together Solo, Together Enterprise, Together Control Center 등의 3종이 있으며, e-Solution 개발을 위한 컴포넌트 모델링, 설계 패턴, 편집, 컴파일, 디버깅, 버전 관리, 문서화, 조립, 배치, 실행 등의 과정을 지원한다. 특히, 클래스 다이어그램과 Java/C++ 코드간, 시퀀스 다이어그램과 Java/C++ 코드간에 구현 및 수정 사항이 발생시, 시각적인 동시성을 보여 주는 "simultaneous round-trip" 기능을 제공해 준다. 또한, 지원하는 EJB 응용 서버도 다양하여 BEA

† 정 회 원 : ETRI 컴퓨터·소프트웨어기술연구소 S/W공학연구부 선임연구원
†† 정 회 원 : ETRI 컴퓨터·소프트웨어기술연구소 S/W공학연구부 응용컴포넌트연구팀 팀장

WebLogic, IBM WebSphere, iPlanet, SilverStream, Generic을 지원한다.

상기의 CBD 도구의 대부분은 그 기능 면에서 완전치 못하고 세련되어 있지 못하다. 특히, 특정 어플리케이션을 구축하기 위해 바이너리 컴포넌트를 생성하기는 쉽지만 컴포넌트의 조립 방법은 직접적인 코딩을 통해 결합되기 때문에 실행 환경 하에서 컴포넌트의 대체 (Replacement, Deletion/Connection)가 어렵다[3]. 이들 컴포넌트들의 인터페이스를 이해하고 관련 컴포넌트들을 'plug-&-play'를 해서 컴포넌트 시스템을 구축할 수 있는 소프트웨어 아키텍쳐 기반의 컴포넌트 개발도구가 아직까지 없는 상황이다. 컴포넌트 기술이 보다 강력한 기능을 제공하기 위해서는 아키텍쳐 기반으로 컴포넌트의 생성, 조립 및 추출이 가능해야 하며 현재 해외에서도 소프트웨어 아키텍쳐 기술을 컴포넌트 기술 분야에 적용하려는 연구가 진행중이다[6].

소프트웨어 아키텍쳐의 핵심인 아키텍쳐 기술 언어(ADL: Architecture Description Language)는 해외의 여러 대학을 중심으로 여러 종류의 프로토타입[4,5,7,8]이 나와 있어 이들을 비교 평가한 후에 성능을 개선하여 CBD 도구를 개발하는 데 활용하면, 현재 컴포넌트 시장에 출시되어 있는 기능이 부족하고 사용하기 불편한 컴포넌트 개발도구 보다 더 우수한 제품을 만들 수 있을 것이다.

본고에서는 현존 CBD 지원도구를 소개하고 제 2장에서는 컴포넌트 생성 및 조립 기술 개발과 관련하여 본 연구팀에서 개발하고 있는 CBD 지원도구의 구성을 설명하고 제 3장에서는 그 기능에 대해 간략히 제시한다.

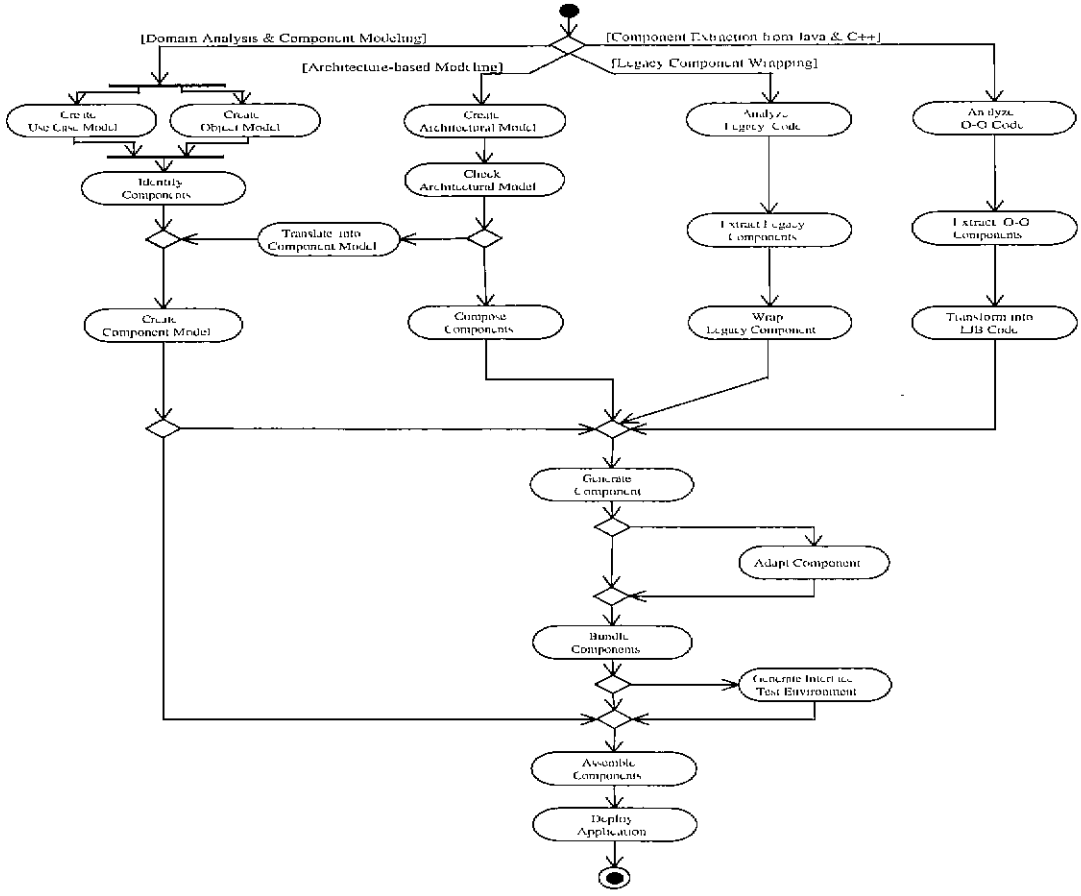
2. CBD 지원 도구의 구성

현재 본 연구팀에서 개발되고 있는 CBD 지원도구는 컴포넌트에 대한 모델링, 설계, 구현, 시험, 조립 및 응용 서버로의 전개 기능 등을 포함한다. 소프트웨어 아키텍쳐를 기반으로 대상 어플리케이션 시스템 구조의 구축뿐만 아니라 구문검사, 의미 검사 및 구현코드 생성까지를 지원하는 광범위한(Wide Spectrum) 도구이다. (그림 1)은 CBD 지원도구의 활동도(Activity Diagram)를 보여 주며, 이것은 아키텍쳐 분석 지원도구, 컴포넌트 생성 지원도구, 컴포넌트 추출 지원도구 및 컴포넌트 조립 지원도구의 4개 시스템으로 구성되어 있다.

아키텍쳐 분석 지원도구는 (그림 1)의 윗부분의 첫째와 둘째 부분의 흐름(Flow)에 해당하며, 어플리케이션 영역에서 요구되는 기능을 모델링하여 이 모델링 정보로부터 컴포넌트를 식별하고 그 명세를 정의하는 영역 모델러와 대상 어플리케이션을 구성하고 있는 컴포넌트들과 그들 간의 관계를 아키텍쳐 기반으로 모델링을 하고 그 결과를 컴포넌트 다이어그램 정보로 변환하는 기능을 가진 아키텍쳐 모델러 등 2개의 서브 시스템으로 구성되어 있다.

컴포넌트 생성 지원도구는 (그림 1)의 중앙 부분의 "Generate Components"에서 "Generate Interface Test Environment"까지의 흐름이며, 컴포넌트를 개발하기 위하여 분석 및 설계 단계의 모델링을 지원해 주는 컴포넌트 모델러와 컴포넌트 모델링 정보로부터 EJB 컴포넌트의 구현을 위한 Java 언어 기반 구현 환경을 제공하고, EJB 컴포넌트의 최종 코드를 생성하는 코드 생성기 등 2개의 서브 시스템으로 구성되어 있다.

컴포넌트 추출 지원도구는 (그림 1)의 윗부분의 셋째와 넷째 부분의 흐름에 해당하며, C 또는 COBOL 언어로 되어 있는 기존 시스템으로부터 컴포넌트를 추출하기 위하여 기존 코드를 Wrapping 여 컴포넌트화(Componentization) 해주는 Legacy 시스템 연계기와 Java 또는 C++ 언어로 되어 있는 객체지향 시스템으로부터 컴포넌트를 식별하여 EJB 컴포넌트로 변환해 주는 컴포넌트 추출기



(그림 1) CBD 지원 도구의 Activity Diagram

등 2개의 서브 시스템으로 구성되어 있다.

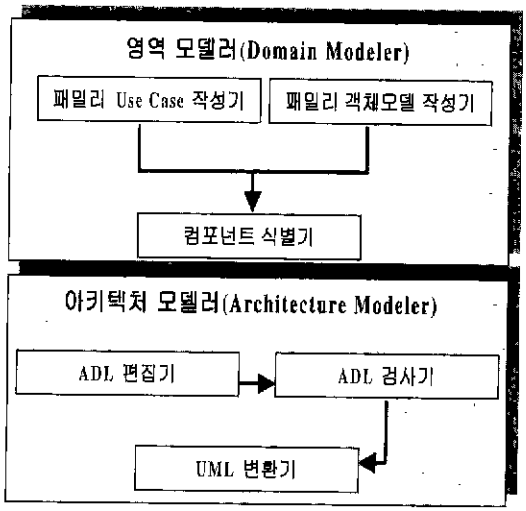
컴포넌트 조립 지원도구는 (그림 1)의 하단 부분의 "Assemble Components"와 "Deploy Applications"의 흐름이며, 바이너리 컴포넌트를 실제 어플리케이션의 요구사항에 맞도록 컴포넌트의 적합화(Adaptation)를 통하여 수정 사용할 수 있도록 하거나 아키텍처 모델을 기반으로 컴포넌트들의 바인딩을 통하여 최종 어플리케이션 코드를 생성해 주는 컴포넌트 합성기와, EJB 컴포넌트를 응용 서버에 전개하기 위한 특성들을 설정하거나 실행 환경에서 응용 서버의 성능에 대하여 동적으로 모니터링해 주는 컴포넌트 배치기 등 2개의 서브 시스템으로 구성되어 있다.

3. CBD 지원 도구의 기능

3.1 아키텍쳐 분석 지원도구의 기능

아키텍쳐 분석 지원도구는 응용 영역에 대하여 기능적, 객체적 분석 등을 수행하여 재사용 단위인 컴포넌트를 식별해 내고 이러한 컴포넌트를 이용하여 소프트웨어 아키텍쳐를 생성하는 응용 어플리케이션의 요구사항 분석 과정을 지원한다. 아키텍쳐 분석 지원도구는 (그림 2)와 같이 영역 모델러와 아키텍쳐 모델러라는 서로 다른 별개의 서브 시스템으로 구성된다. 영역 모델러는 UML 1.3 기반의 객체지향 기법을 사용하여 특정 영역의 분석을 지원하는 도구이며 아키텍쳐 모델러는 소

프트웨어 아키텍처 구축을 지원하는 분석 도구이다. 아키텍처 분석 지원도구의 기능은 다음과 같다.



(그림 2) 아키텍처 분석 도구의 구조

3.1.1 영역 모델러(Domain Modeler)

3.1.1.1 패미리 Use Case 작성기

Use Case는 시스템의 기능을 사용자 관점에서 사용자의 입력, 시스템의 반응 형태로 기술된다. 패밀리 Use Case는 특정 어플리케이션에 국한되지 않고 패밀리에 존재하는 모든 어플리케이션에 대한 Use Case를 포함한다. 패밀리 Use Case 작성기는 Use Case 다이어그램을 생성할 수 있는 그래픽 편집 기능, 연속적인 오퍼레이션으로 나타내는 시스템과의 상호작용 기술 기능, 어플리케이션에 따라 변할 수 있는 Use Case의 가변적 정보를 나타낼 수 있는 기능 등을 제공한다.

3.1.1.2 패밀리 객체모델 작성기

패밀리 객체 모델 작성기는 패밀리에 존재하는 클래스들을 모델링하고 클래스들의 관계를 기술한다. 패밀리 객체 모델은 컴포넌트 모델러의 여러 다이어그램들에서 객체 모델의 일부를 그대로 이용하거나 객체 타입 정의로 이용된다.

3.1.1.3 컴포넌트 식별기

컴포넌트의 재사용성을 높이기 위해서는 컴포넌트 식별 과정에서 영역(Domain)에 존재하는 공통(Commonality) 요소를 중심으로 컴포넌트를 구성해야 하며, 가변성(Variability)을 고려하여 컴포넌트를 추출한다. 컴포넌트 식별 시 다양한 영역 정보, 영역 전문가의 경험 등의 정보를 필요로 한다.

3.1.2 아키텍처 모델러(Architecture Modeler)

3.1.2.1 ADL 편집기

본 연구의 아키텍처 기반 CBD 지원 도구는 아키텍처 스타일(Architctural Style)로서 C2[6]를 채택하였으며, 컴포넌트 아키텍처로서 EJB를 지원한다. C2 컴포넌트, 커넥터와 같은 아키텍처의 구성요소를 팔레트에서 설계화면에 끌어다 놓기(drag-nd-drop)를 하여 소프트웨어 아키텍처를 그래픽 방식으로 손쉽게 설계할 수 있으며, 아키텍처 모델의 그래픽 형태의 표현과 C2 SADEL[6]를 사용한 텍스트 형태의 표현들 간의 변환을 자동적으로 지원한다. 일반적인 편집기가 제공하는 기본적인 파일 입출력 기능과 복사, 붙여넣기, 삭제 등과 같은 기본적인 편집 기능을 제공한다.

3.1.2.2 ADL 검사기

ADL 검사기는 아키텍처 명세가 C2 SADEL의 문법에 맞게 작성되었는지를 검사하는 가장 기본적인 구문 검사 기능을 제공한다. 구문 검사는 아키텍처 구조의 명세뿐만 아니라 각 컴포넌트의 인터페이스 명세도 포함한다. ADL 검사기는 아키텍처 명세의 타입 검사 기능을 제공한다. 타입 검사에는 각 컴포넌트의 행위에 관한 명세뿐만 아니라 컴포넌트들 간의 상호작용이 C2 SADEL의 타입 규칙을 만족하는지도 검사한다.

아키텍처 명세는 C2 스타일이 규정하는 형태(Topology) 규칙에 맞게 작성되어야 한다. 예를 들면, C2 아키텍처는 계층(Layered) 구조를 가져

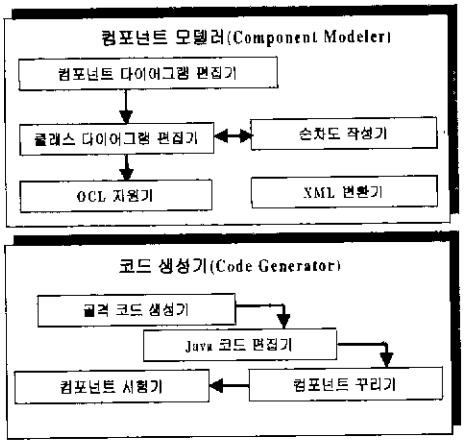
야 하며, 컴포넌트는 일정한 규칙에 따라 커넥터에 연결되어야 한다. 따라서 ADL 검사기는 아키텍처 명세가 C2 규칙과 제한사항을 만족하는지를 검사하는 기능을 제공한다.

3.1.2.1 UML변환기

UML 변환기는 아키텍처의 구성요소가 되는 C2 컴포넌트의 분석, 설계 및 구현을 위하여 UML 기반의 컴포넌트 모델러와 연동할 수 있도록 C2 SADEL로 작성된 소프트웨어 아키텍처 명세를 UML 명세로 자동으로 변환한다. UML 변환기는 C2 SADEL 명세를 UML로 변환하기 위한 규칙을 정의하여야 하며, 정의된 규칙에 의거하여 UML 명세를 생성한다.

3.2 컴포넌트 생성 지원도구의 기능

컴포넌트 생성 지원도구는 아키텍처 분석 지원도구에서 식별된 컴포넌트들에 대해 컴포넌트 인터페이스를 명확히 기술하고 각각의 컴포넌트를 개발하는데 필요한 설계 및 구현 과정을 지원하는 기능을 제공한다. (그림 3)에서 처럼 컴포넌트 생성 지원 도구는 UML 기반의 객체지향 기법을 사용하는 컴포넌트 모델러와 코드 생성기라는 서브 시스템으로 구성되며, 컴포넌트 생성 지원도구



(그림 3) 컴포넌트 생성 도구의 기능

의 기능은 다음과 같다.

3.2.1 컴포넌트 모델러 (Component Modeler)

3.2.1.1 컴포넌트 다이어그램 편집기

컴포넌트들 간의 연관성을 그래픽 다이어그램으로 나타내는 컴포넌트 다이어그램을 위한 편집 기능을 제공한다. 컴포넌트의 불변성(Invariant), 가변성(Variability), Home/Remote 인터페이스 정보 등을 포함한다. 인터페이스 명세는 오퍼레이션의 집합으로 구성되며 오퍼레이션을 기술하기 위해 입출력 인자와 오퍼레이션 사전과 사후 조건을 기술한다. 컴포넌트 다이어그램 편집기는 영역 모델러의 컴포넌트 식별기 혹은 아키텍처 모델러의 UML 변환기로부터 컴포넌트 관련 정보를 입력으로 받는다.

3.2.1.2 클래스 다이어그램 편집기

컴포넌트 인터페이스를 명확히 하고 컴포넌트 내부 구현을 위해 클래스들을 정의하고 그들 간의 관계를 기술한다. 컴포넌트 클래스 다이어그램의 클래스들은 패밀리 객체 모델에서 정의한 클래스들을 상세 설계 수준으로 정제하여 사용한다. 컴포넌트 내부 클래스는 인터페이스와 밀접한 연관 관계를 가지므로 이러한 연관 관계를 다이어그램 상에 나타낸다. 즉, 클래스 다이어그램과 인터페이스와의 연관 관계는 새로운 Stereotype으로 정의한다. 각 클래스에 대해 Entity/Session빈 여부를 Stereotype으로 표현한다.

3.2.1.3 순차도 작성기

컴포넌트 인터페이스를 명확히 파악하기 위해 Use Case 별로 컴포넌트들 간의 메시지 흐름을 파악하는 것이 요구되며, 또한 컴포넌트 내부의 객체의 상호작용을 명확히 파악하기 위해 내부 객체들 간의 메시지 흐름을 기술하는 것이 필요하다. 순차도는 시간을 나타내는 세로축과 상호작

용을 하는 객체 혹은 컴포넌트들을 나열하는 가로축으로 구성되어 있으며, 시간의 흐름에 따른 객체 또는 컴포넌트들 간의 메시지 전송, 조건적 메시지 전송, 재귀적 호출, 객체 생성 및 소멸 등을 나타낸다.

3.2.1.4 OCL 지원기

OCL(Object Constraint Language) 지원기는 클래스의 불변성, 메소드/오퍼레이션의 사전과 사후 조건을 기술할 수 있는 기능, OCL 표현(Expression)의 구문 검사 및 타입 검사 기능을 제공한다. OCL의 구체적인 기능과 표기법은 UML에 정의된 OCL의 기능과 표기법을 그대로 따른다.

3.2.1.5 XML 변환기

XML(eXtensible Markup Language) 변환기의 기능은 다이어그램 편집기에서 생성된 모델 정보를 XML로 변환하여 다른 모델링 도구에서 사용할 수 있게 하며, 다른 모델링 도구에서 생성된 모델의 XML 정보를 해석하여 다이어그램 편집기에서 사용할 수 있게 한다.

3.2.2 코드 생성기(Code Generator)

3.2.2.1 골격 코드 생성기

컴포넌트 다이어그램의 인터페이스 명세와 클래스 다이어그램의 내부 객체들의 모델링 정보를 바탕으로 EJB의 홈/원격(Home/Remote) 인터페이스를 생성하고 내부 클래스 구현을 단순화하기 위한 클래스의 골격 코드를 생성한다. 세션/엔티티(Session/Entity) 빈이냐에 따라 홈/원격 인터페이스의 내용이 달라지므로 Session/Entity 빈의 정보를 설계로부터 입력받는다. 객체의 지속성(Persistency) 여부를 선택할 수 있어야 하며 지속(Persistent) 객체에 대해 특정 DBMS에 대한 DDL을 자동 생성한다.

3.2.2.2 Java 코드 편집기

컴포넌트의 인터페이스 코드들은 자동으로 생성 가능하지만 내부 클래스의 비즈니스 로직은 프로그래머가 직접 구현하여야 한다. 코드 편집기의 주요 기능으로는 골격 코드 생성기에 의해 생성된 내부 클래스의 골격 코드를 바탕으로 Java로 내부 기능들을 구현할 수 있는 Java 코드 편집 기능, Java 코드를 컴파일 할 수 있는 환경 등을 제공한다.

3.2.2.3 컴포넌트 꾸리기

EJB 코드의 수행을 위해서 응용 서버의 컨테이너에 EJB 코드를 먼저 전개해(Deploy) 두어야 한다. 컴포넌트 꾸리기는 EJB 모듈, Web 모듈, 어플리케이션 클라이언트 모듈, 자원 어댑터(Resource Adapter) 모듈, 어플리케이션 모듈 각각에 해당하는 전개 파일(Deployment Descriptor)의 생성과 모듈을 하나로 묶어주는 꾸러미 파일인 .jar 파일, .war 파일, .rar 파일, .ear 파일 생성 업무를 수행한다. 꾸러미 파일 생성은 J2EE에서 제공되는 Packager 도구를 이용하여 연관된 컴포넌트들을 하나로 묶는다.

전개 파일에 명시되는 공통 요소로는 환경 요소(Environment Entries), EJB의 참조, 자원 관리자와의 연계를 담당하는 객체들에 대한 참조, 보안(Security) 요소 등이 있다. EJB 모듈 고유의 요소로는 컨테이너/빈 관리 트랜잭션을 결정하는 트랜잭션 요소와 지속성(Persistence) 요소가 있다.

3.2.2.4 컴포넌트 시험기

컴포넌트 시험기는 하나의 EJB컴포넌트 단위로 인터페이스가 제대로 작동하는지를 시험할 수 있는 환경을 제공한다. 컴포넌트 안의 인터페이스 메소드를 파악하고 메소드에서 요구되는 타입의 데이터 값들을 입력할 수 있는 환경을 제공하고, 입력 값을 이용하여 인터페이스 메소드를 호출하고, 인터페이스 메소드가 올바르게 수행되는지를

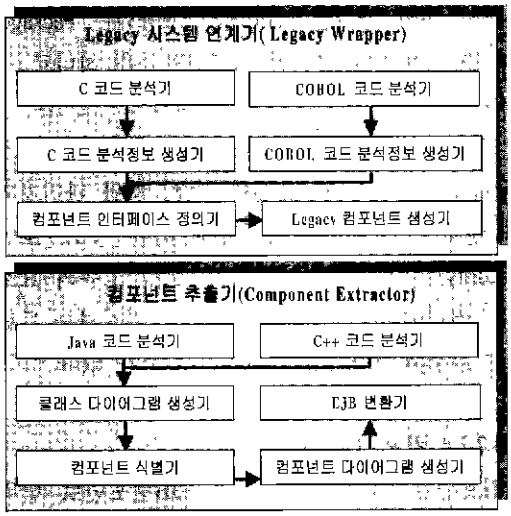
판단할 수 있도록 결과를 보여준다.

3.3 컴포넌트 추출 지원도구의 기능

컴포넌트 추출 지원도구는 절차 중심적인 언어로 개발된 소프트웨어와 객체지향 언어로 개발된 소프트웨어로부터 컴포넌트를 추출하여 EJB 컴포넌트화(Componentization) 해주는 기능을 제공한다. (그림 4)에서 처럼 이것은 Legacy 시스템 연계기와 컴포넌트 추출기라는 서브 시스템으로 구성된다.

3.3.1 Legacy 시스템 연계기(Legacy Wrapper)

Legacy 시스템 연계기는 기존의 COBOL, C 언어로 된 프로그램을 사용자가 이해할 수 있도록 시각적으로 표현하고, 컴포넌트 인터페이스 정의기가 프로그램 분석 정보를 이용하여 컴포넌트 대상 모듈의 식별 및 추출후, 컴포넌트 Wrapping을 통하여 재사용이 가능한 컴포넌트를 생성한다.



(그림 4) 컴포넌트 추출 도구의 기능

3.3.2 컴포넌트 추출기(Component Extractor)

컴포넌트 추출기는 기존의 Java 및 C++언어로 된 프로그램으로부터 컴포넌트를 식별하여 컴포넌트 다이어그램을 생성하고 EJB Bean으로 변환한다. 컴포넌트를 식별하기 위해서 Java 코드 및 C++ 코드 분석기를 통해서 소스코드를 분석하여 클래스 다이어그램을 생성하고, 생성된 클래스 다이어그램과 기타 분석 정보들을 바탕으로 하여 컴포넌트를 식별한다. 식별된 컴포넌트는 컴포넌트 다이어그램 형태로 만들어지고, EJB 형식에 맞는 컴포넌트로 생성된다.

3.4 컴포넌트 조립 지원 도구의 기능

컴포넌트 조립 지원 도구는 기 개발되어 있는 컴포넌트를 조립하여 어플리케이션(컴포넌트 시스템)을 구축하는 역할을 수행하며, 이 지원 도구는 컴포넌트 합성기와 컴포넌트 배치기라는 서브 시스템으로 구성된다.

3.4.1 컴포넌트 합성기(Component Composer)

컴포넌트 합성기는 기 개발된 컴포넌트를 이용해 'plug-&-play'의 조립 과정을 통해 원하는 복합 컴포넌트(Composite Component)를 작성하거나 응용 시스템을 구축하는 기능을 수행한다. 컴포넌트 합성기는 컴포넌트 브라우저(Component Browser), 컴포넌트 어댑터(Component Adapter)와 컴포넌트 바인더(Component Binder)로 구성된다.

컴포넌트 브라우저는 컴포넌트 사용자가 컴포넌트 저장소로부터 원하는 컴포넌트를 검색하여 사용할 컴포넌트를 선택하게 한다. 컴포넌트 어댑터는 컴포넌트를 합성하기 전에 컴포넌트 인터페이스의 불일치를 해결하거나 컴포넌트의 기능을 보완하기 위해 바이너리 컴포넌트의 인터페이스를 수정하거나 컴포넌트를 Wrapping하는 기능을 제공한다. 컴포넌트 바인더는 아키텍처 모델러에서 사용하는 C2 아키텍처를 기반으로 사용자가 새로운 복합 컴포넌트 혹은, 특정 어플리케이션의 구조를 ADL 편집기를 통해 편집하면 연결 코드를 생성하고 또한 실행 환경 하에서 컴포넌트 조립 상태를 변경할 수 있는 동적 재구성 기능을

제공한다.

3.4.2 컴포넌트 배치기(Component Deployer)

컴포넌트 배치기는 생성된 EJB 컴포넌트 파일을 응용 서버에 실제로 전개(Deploy)하는 과정을 지원하는 운용 환경 관리기와 실행 환경에서 EJB 응용 서버의 성능에 대해서 동적으로 모니터링 하는 기능을 제공하는 컴포넌트 모니터링기로 구성된다. 컴포넌트 배치기는 특정한 응용 서버에 종속적으로 기능을 수행하게 되므로 가장 많이 사용되고 기능이 좋은 대표적인 몇 개의 응용 서버를 지원한다.

4. 결론 및 향후 연구 방향

본 연구는 현재 기본 설계를 완료하였으며, 몇 가지 측면에서 기존의 CBD 도구와 구별된다. 첫째로, 컴포넌트 조립 방법에 있어서 아키텍쳐 기반의 컴포넌트 조립을 지원하므로 'plug-&-play'로 시각적이고 손쉽고 빠르게 복합 컴포넌트 및 응용 시스템을 구축할 수 있는 기능을 제공한다. 둘째로, 컴포넌트의 적합화(Adaptation) 시 소스코드를 수정하지 않고, 바이너리 컴포넌트의 인터페이스를 수정하거나 컴포넌트내의 특정 기능을 추가할 수 있는 BCA(Binary Component Adaptation) 기법[9]을 지원한다. 셋째로, 컴포넌트의 추출 방법에 있어서 기존 도구는 주로 현존 Legacy System의 특정 기능을 컴포넌트화 하여 연계 사용하는 것을 지원하고 있으나, 본 과제의 목표는 기존 기법을 지원함은 물론, Legacy System으로부터 역공학 및 Refactoring 기술을 이용하여 단위 대상 모듈을 추출 후, Component Wrapping을 통하여 EJB 컴포넌트를 생성시켜 주므로 기존 시스템의 부가가치를 높여준다.

본 과제에서 개발될 CBD 지원도구는 현재 3개 EJB 응용 서버를 지원하나 향후 Upgrade시에 몇 개 서버의 추가 지원을 할 것이며, 재사용 경제(Reuse Economics) 측면에서 컴포넌트 개발 기간, 개발비, 사용 빈도, 투자비용 회수 기간 등의 메트릭 정보를 관리하고 제공할 것이다.

참고문헌

[1] D. D'Souza and A. Wills, "Object, Components, and Frameworks with UML: the Catalysis Approach", Addison-Wesley, 1998

[2] C. Szyperski, "Component Software: Beyond Object-Oriented Programming", ACM Press, Addison-Wesley, 1998

[3] K. Bergner, A. Rausch and M. Sihling, "Componentware- The Big Picture", Proceedings of 1st Workshop on CBSE in conjunction with 20th ICSE, April 1998, Kyoto, Japan

[4] D. Garlan, R. Monroe, and D. Wile, "Acme: An Architecture Description Interchange Language", Proceedings of CASCON'97, Nov. 1997

[5] R. Melton, "The Aesop System: A Tutorial", The ABLE Project, School of Computer Science, CMU

[6] R. N. Taylor, N. Medvidovic, K. M. Anderson, et al., "A Componentand MessageBased Architectural Style for GUI Software", IEEE Transactions on Software Engineering, Vol. 22, No. 6, June 1996, pp. 390-406

[7] D. C. Luckham and J. Vera, "An EventBased Architecture Definition Language", IEEE Transactions on Software Engineering, Vol. 21, No. 9, September 1995, pp. 717-734

[8] R. J. Allen, R. Douence, and D. Garlan, "Specifying Dynamism in Software Architectures", Proceedings of the Workshop on Foundations of Component-Based Software Engineering, September 1997

[9] R. Keller and U. Hölzle, "Binary Component Adaptation", Technical Report TRCS97-20, Department of Computer Science, University of California, Santa Barbara, Dec. 1997

권 오 천

1994년 영국 Teeside 대학교 대학원 졸업(소프트웨어공학 석사)
1998년 영국 Durham 대학교 대학원 졸업(공학 박사)
1991년 미국 RTP IBM 연구소 객원연구원
현재 ETRI 컴퓨터·소프트웨어 기술연구소 S/W공학연구부 선임연구원
관심분야 : 재사용, 컴포넌트 기반 개발(CBD)

신 규 상

1981년 성균관대학교 통계학과 졸업(학사)
1983년 서울대학교 대학원 계산통계학과 졸업(이학석사)
1983년 시스템공학연구소 연구원
1987년 시스템공학연구소 선임연구원
1997년 시스템공학연구소 책임연구원
현재 ETRI 컴퓨터·소프트웨어기술연구소 S/W공학연구부 응용컴포넌트연구팀 팀장
관심분야 : 컴포넌트웨어, S/W 재공학, CASE, 객체지향 기법