

CBD 아키텍처 기반의 컴포넌트 저장소 프로토타이핑

*김행곤 차정은 최문경 **김철홍
*대구가톨릭대학교 컴퓨터공학과 **ETRI 컴퓨터소프트웨어연구단 소프트웨어공학연구소
*(hangkon, jecha)@cuth.cataegu.ac.kr, ditocmk@hanmail.net, **kch@etri.re.kr

Prototyping for Component Repository Based on CBD Architecture

*Haeng-Kon Kim Jung-Eun Cha Mun-Kyoung Choi **Chul-Hong Kim
*Dept. of Computer Engineering, Catholic University of Daegu
**ETRI Computer & Software Technology Lab SE Department

요 약

대부분의 응용 시스템들이 인터넷과 웹 기술을 기반으로 개발, 운영됨에 따라 상호운영성을 허용하고 동적 변화에 민첩하게 대응할 수 있는 소프트웨어 개발을 위한 방법론 및 툴 지원이 요구된다. 이에 따라 컴포넌트 기반의 소프트웨어 개발(CBD : Component Based Development)은 소프트웨어 시스템 구축을 위한 가장 이상적 전략으로 인식되고 있다. CBD 개발을 지원하기 위한 컴포넌트 저장소는 컴포넌트의 개발과 유통, 활용을 위한 컴포넌트 및 관련 모든 프로젝트 정보들의 통합적인 관리 중심체이다. 그러므로 효과적인 CBD의 전개를 위해서는 정규화된 컴포넌트 참조 아키텍처에 기반한 저장소 구축은 필수적이다.

본 논문에서는 CBD 방법론에 의한 컴포넌트 저장소 구축을 목적으로 저장 아키텍처를 제시하고 이를 기반한 저장소 프로토타이핑을 개발한다. 이를 위해 컴포넌트 기능성 영역의 수직적 관점과 라이프 사이클 관점의 수평적 관점이 혼합된 아키텍처를 정의하고 이에 식별, 정의되어진 컴포넌트를 적용한다.

1. 서론

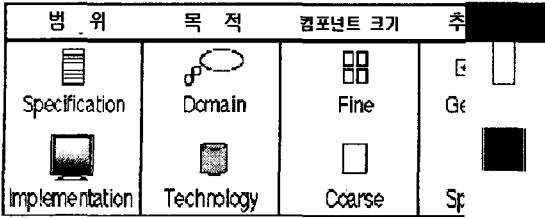
현재의 비즈니스 시스템들은 최신의 기술적 배경으로 빠르게 개발, 운영하고 경제적으로 유지보수하며 다른 응용의 요구와 그에 대한 응답으로 자유롭게 사용할 수 있도록 요구받는다. 컴포넌트 기반의 소프트웨어 개발(CBD : Component Based Development)은 이미 존재하는 컴포넌트들을 시스템 요구에 따라 조립하고 적용시키며 연결함으로써 소프트웨어 개발에 대한 모든 관점의 결과물들을 생산해내는 것이다[1]. 그러므로 컴포넌트 개념과 적용 프로세스는 소프트웨어 각 개발 단계 뿐 아니라 아키텍처에 기술적 하부구조 그리고 프로젝트 관리에 이르기까지 소프트웨어 생산을 위한 모든 절차와 지원 환경으로의 접근을 포함해야만 한다. 또한 컴포넌트 저장소는 컴포넌트의 개발과 유통 그리고 활용을 지원하기 위한 통합적인 중심체로 재사용이라는 궁극적인 접근

을 위한 실제적인 도구가 된다. 기존의 저장소는 소프트웨어 개발 부산물, 언어지원 서브루틴 그리고 링크 라이브러리등을 제공한다[2].

현재 특정 비즈니스 도메인을 위한 컴포넌트 저장소와 관련된 유용한 컴포넌트 검색, 추출 그리고 조합 등이 정보산업 비즈니스의 이슈가 되고 있다.

본 논문에서는 컴포넌트 전체 라이프 사이클 각각에서 생산되어진 컴포넌트 소프트웨어 생산물 및 이들에 대한 활용 및 관리 지원 도구를 포함하고 있는 CBD 기반의 컴포넌트 저장소를 개발하기 위해 저장 아키텍처를 정의하고 이를 기반으로 저장소 프로토타이핑을 개발한다. 본 저장소는 컴포넌트 정보의 일관성 있는 관리를 위해 서버측에 저장 컴포넌트의 정보베이스를 구축하고 형상관리 하며 클라이언트 측에 컴포넌트 검색 및 획득, 유통을 위한 웹 사이트를 구축한다. 이를 위해 컴포넌트 기능성 영역의 수직적 관점과 라이프 사이클 관점의 수평적 관점이 혼합된 아키텍처를 정의하였다.

본 논문은 2000년도 한국전자통신연구원 컴포넌트 사업 계획에 의해 수행됨.



(그림 1) 컴포넌트 분류 기준

2. 관련연구

2.1 컴포넌트 분류

아키텍처를 정의하기 위해서는 기본적으로 컴포넌트를 분류하는 것이 필요하다. 컴포넌트 분류를 위해서는 컴포넌트를 생성, 저장하기 위한 체계적 스키마를 확보하고, 최종 컴포넌트 사용자를 위한 컴포넌트 식별, 검색, 이해의 메타정보가 필요하다[3].

컴포넌트의 분류 기준은 (그림 1)과 같이 범위, 목적, 컴포넌트 크기 그리고 추상화 정도로 나누며, 추가적 분류 요소가 있다.

2.2 컴포넌트 아키텍처(Component Architecture)

컴포넌트 아키텍처는 컴포넌트와 그들의 동작 환경 사이의 상호작용을 정규화하고, 인터페이스의 표준화를 규정하는 것으로 Standard Infrastructure와 Evaluated Program Model로 나눌 수 있다. 일반적인 컴포넌트 참조 모델은 분산 컴퓨팅을 목적으로 멀티 벤더의 멀티 솔루션 통합에 초점을 두고, 다음의 3계층으로 구성된다[4].

- ① 하부 계층 : 미들웨어 역할을 하는 API 및 확장가능한 RMI, CORBA Facility, CORBA Service들로 구성되어 응용의 하부구조 특히, 분산 객체 서비스를 위한 기본 포맷 제공
- ② 중간 계층 : 많은 어플리케이션에서 필요로 하는 객체(기능)들을 지원하기 위한 독립적인 컴포넌트들로 다중 도메인 상의 일반화된 메카니즘 지원
- ③ 최상위 계층 : Vertical 도메인을 위한 특정 응용 로직(프로세스) 컴포넌트

2.3 컴포넌트 저장소

컴포넌트를 효율적으로 관리하기 위해서 프로세스 관리, 프로젝트 관리, 데이터 관리, 문서관리, 변경 관리, 형상 관리 등과 관련하여 체계적인 컴포넌트 저장소를 구축해야 한다. 컴포넌트 저장소는 소프트웨어 생명 주기동안 모아진 시스템 정보를 관리, 저장되는 곳으로, 각 도구들, 각 개발 단계들, 사용자들, 응용프로그램 사이의 시스템 정보를 공유할 수 있도록

<표 1> 컴포넌트 저장 아키텍처

아키텍처 / 수직적 관점	물리적 / 수평적 관점
“컴포넌트 아키텍처”를 위한 참조 모델 기반	실질적인 컴포넌트의 검색 및 유통의 결론적인 관점
비즈니스 영역의 규모에 따라 분류, 저장	컴포넌트의 생산의 abstraction과 realization
- 의미적(기능적) 관점의 컴포넌트 분류	사이의 수준에 따라 구분
- 비즈니스에 따른 네비게이팅 검색	- 분석/ 설계 정보를 위한 모델링 파일
- 컴포넌트 조합을 위한 기본 시나리오 제공	- 컴포넌트 메타 정보
- 컴포넌트 메타 정보로서 이용	- 인터페이스 및 조립을 위한 메타 텍스트 문서
	- 원시코드(개별 클래스 및 헤더파일, 구현파일,...)
	- 실행파일(Plugable file, DLL, EXE,...)

해준다. 이러한 컴포넌트 저장소는 시스템 유지 보수를 용이하게 하고, 재사용이 가능하게 하는 등의 소프트웨어 개발, 유지보수 등에서 비용의 감소라는 큰 이점을 내포한다[5].

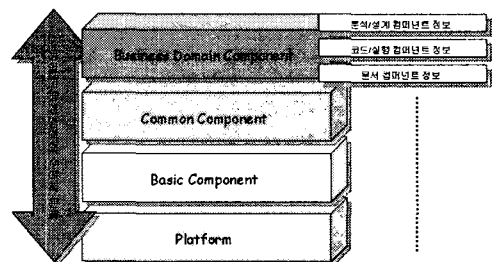
3. 컴포넌트 저장 아키텍처

3.1 컴포넌트 분류 체계 및 저장 아키텍처 정의

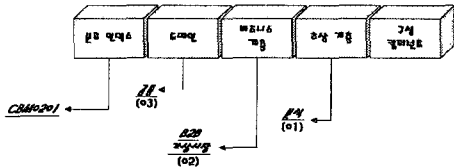
컴포넌트 저장소를 구축하여, 효율적으로 사용하기 위해 아키텍처를 정의하는데, 본 논문에서는 두 가지 관점에서 아키텍처를 제시하였다. 크게 컴포넌트 기능성 영역의 수직적 관점과 컴포넌트 라이프 사이클 관점의 수평적 관점이 혼합된 아키텍처를 정의하였다 <표 1>.

컴포넌트 참조 아키텍처 제안을 위해 분산 컴퓨팅 환경과 비즈니스 솔루션을 위한 표준 모델의 멀티 벤더/멀티 솔루션을 통합하여, 적용범위(컴포넌트 스펙)를 중심으로 계층적으로 분류한다.

<표 1>에서 정의된 아키텍처에 따라 저장소 구축을 위한 컴포넌트 저장 아키텍처를 구조화하면 (그림 2)와 같다.



(그림 2) 컴포넌트 저장 아키텍처



(그림 3) 컴포넌트 저장 메타정보

본 논문에서 제시하는 아키텍처에 따른 컴포넌트의 분류 코드를 지정을 위해 4개의 저장 메타데이터를 정의하였다. '도메인' 항목은 컴포넌트가 적용될 수 있는 총체적인 비즈니스 영역이며, '비즈니스 계층' 항목은 실질적인 기능성을 포함하는 업무를 식별한다. 그리고 '추상계층'은 분석·설계·구현에 따르는 컴포넌트 저장 형태를 나타내며, '구성 컴포넌트' 항목은 해당 컴포넌트가 조립될 수 있는 비즈니스 프로세스 컴포넌트를 나타낸다. (그림 3) 즉, 공통(C)비즈니스관리(BM)컴포넌트 인덱싱(O2)의 분석정보(O1) 코드는 (그림 4)와 같이 CBM0201과 같이 표현된다.

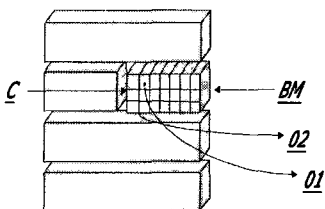
4. 컴포넌트 저장소 프로토타이핑

4.1 개요

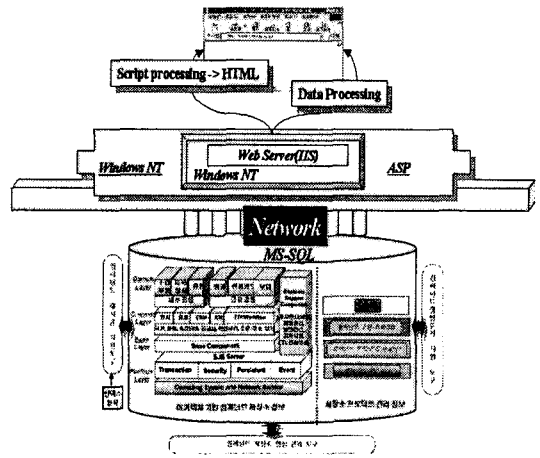
본 논문은 컴포넌트 기반의 소프트웨어 개발 기술과 비즈니스 솔루션 제공을 위한 요구에 기반하여 컴포넌트 각 생명 주기에서 생산되어진 결과물 및 이와 관련된 메타 정보들을 형성 관리의 측면에서 체계적으로 관리하고 사용자에게 편리한 검색, 획득 기능을 제공할 수 있는 컴포넌트 저장소(CRMS : Component Repository Management System)를 개발한다.

CRMS는 CBD 방법론에 기반하여 분석, 설계, 구축되어진 클라이언트 서버 시스템이다. 즉, 컴포넌트 프로젝트 및 메타 정보의 일치성 있는 형상 관리를 위한 서버 시스템과 웹 상에서 사용자의 편리하고 효과적인 컴포넌트

분류 코드 : (C)공통(BM)비즈니스 관리(O2)컴포넌트 인덱싱(O1)(분석정보)



(그림 4) 분류코드 및 메타데이터 적용



(그림 5) 저장소 프로토타이핑 개발을 위한 아키텍처

검색 및 이해 그리고 획득의 과정을 허용하는 클라이언트 시스템으로 구성된다.

4.2 구현환경

본 논문에서는 컴포넌트 저장소 형상 관리 시스템인 CRMS를 개발하기 위해 Windows NT하의 MS-SQL을 컴포넌트 관리 데이터베이스로 사용하고, 클라이언트를 위한 인터페이스 시스템 및 관리 로직은 웹 서버상의 ASP를 이용하여 구축한다.

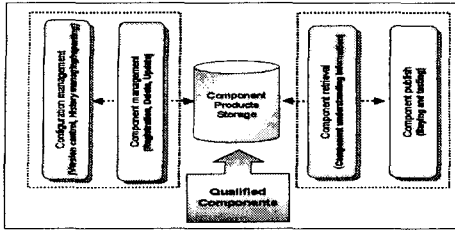
(그림 5)는 CRMS의 프로토타이핑을 개발하기 위한 소프트웨어 환경이다.

4.3 설계

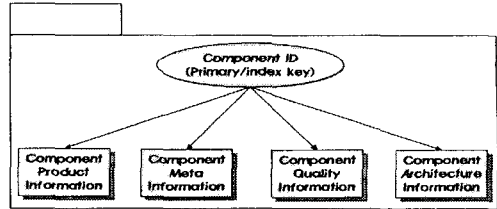
컴포넌트 저장소로서 현재 개설 중인 컴포넌트 유통 사이트와는 달리 본 논문의 CRMS는 구현 단계의 컴포넌트

<표 2> CRMS의 기능

기능	설	명
보안	모든 사용자에게 접근은 허용하나 컴포넌트의 판매는 시스템으로부터 인증받은 사람에 한한다.	
관리	컴포넌트 관련 모든 정보 패키지들을 등록, 검색, 삭제, 갱신한다.	
형상관리	· 컴포넌트 버전 관리를 수행한다. · 저장소 관리 컴포넌트의 히스토리를 감지, 보고한다.	
컴포넌트 다운로드 및 업로드	· 검색, 이해되어서 사용이 확정되어진 컴포넌트는 무료/유료로 다운 로딩될 수 있다. · 사용자가 작성한 컴포넌트의 품질 평가를 위해 저장소로 업로드 한다.	
컴포넌트 브라우징	· 웹 상에서 등록되어진 컴포넌트를 사용자가 검색할 수 있도록 다양한 접근 방식을 제공한다. · 사용자가 선택한 컴포넌트에 대한 자세한 정보를 제공하여 컴포넌트에 대한 이해를 증가시킨다.	



(그림 6) CRMS의 개념적인 아키텍처



(그림 8) CRMS의 컴포넌트 프로덕트 정보 패키지

트 실행 파일 뿐 아니라 분석, 설계 및 메타 정보의 텍스트 문서 등의 통합 정보의 관리를 위한 컴포넌트 정보 저장 서버를 중심으로 클라이언트에서의 사용자가 컴포넌트를 다양하고 진보화된 접근 방식으로 검색하고 이해하며 구매할 수 있도록 충분한 정보를 제공한다.

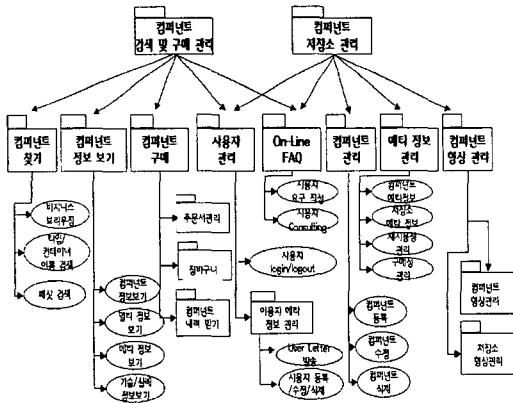
다음은 CRMS의 기능을 상세한 분류 범주 하에 정리한 것이다<표 2>.

(그림 6)은 CRMS의 개념적인 시스템 개략도이다.

CRMS 개발 및 운영의 측면에서 분석되어진 비즈니스 프로세스의 계층을 도식화하면 (그림 7)과 같다.

크게 "컴포넌트 검색 및 구매" 패키지와 "컴포넌트 저장소 관리" 패키지로 구분되며, CBD에 의한 시스템 구축시 독립적인 컴포넌트로서 간주되어질 수 있는 중간 규모의 비즈니스 컴포넌트 패키지가 8개 존재한다. CRMS에서는 하나의 컴포넌트에 관련된 메타 정보를 포함한 모든 프로덕트들을 하나의 논리적인 패키지 단위로 작성, 관리함으로써 저장소의 형상 관리를 위한 컴포넌트의 정보 단위로 이용하고 있다.

(그림 8)은 CRMS에서 정의한 컴포넌트 프로덕트 정보 패키지이다.



(그림 7) CRMS의 비즈니스 프로세스 계층도

5. 결론 및 향후 연구

컴포넌트 저장소는 소프트웨어 생명 주기동안 모아진 시스템 정보를 관리. 저장되는 곳으로, 각 도구들, 각 개발 단계들, 사용자들, 응용프로그램 사이의 시스템 정보를 공유할 수 있도록 해준다. 이러한 컴포넌트 저장소는 시스템 유지 보수를 용이하게 하고, 재사용을 가능하게 하는 등의 소프트웨어 개발, 유지보수 등에서 비용의 감소라는 큰 이점을 내포한다. 본 논문에서는 컴포넌트 기반의 소프트웨어 방법론을 통해 컴포넌트 저장소를 구축하고자하였다. 이를 위해 컴포넌트 생성과 저장, 검색 및 조립의 기본 시나리오로 참조되어질 수 있는 저장 아키텍처를 정의하였다. 본 논문에서 제시하는 저장소는 일련의 컴포넌트 프로세스 과정을 지원하는 서브 시스템 및 형상 관리를 위한 기능성을 포함하고 있다. 따라서 본 논문을 통해서 표준 아키텍처에 기반한 저장소 구축을 지원하고 컴포넌트 분류, 카탈로그의 기준 메타 정보 및 실질적인 저장소 구축을 위한 논리적, 물리적 저장스키마 제시하였다. 향후, 사용자 및 개발자들의 요구를 적절히 수용하는 실질적인 저장소 구축 및 CBD 지원 도구로의 완성이 요구된다.

6. 참고문헌

[1] D'Souza, D, Objects, Components, And Frameworks With UML, Addison Wesley Longman, 1999
 [2] Peter Herzum, Oliver Sims, Business Component Factory : A Comprehensive Overview of Component-Based Development for the Enterprise, OMG press, December, 1999
 [3] Amy Moormann Zaremski and Jeannette M. Wing. "Specification Matching of software components". ACM Transactions on Software Engineering and Methodology, Vol. 6, pp. 333-369, October 1997
 [4] IBM, Sanfrancisco Technical Report, at URL : <http://www-4.ibm.com/software/ad/sanfrancisco>
 [5] Scott Henninger, "Supporting the Construction and Evolution of Component Repositories", ICSE, 1998