CBSD를 지원하는 형상관리 모델 설계

최상균°, 송영재
김포대학 컴퓨터계열°, 경희대학교 컴퓨터공학과
skchoi@kimpo.ac.kr°, yjsong@khu.ac.kr

The Design of Configuration Management Model Supporting CBSD

Sang-Kyoon Choi°, Young-Jae Song
Division of Computer, Kimpo College°, Dept. of Computer Engineering, KyungHee University

요 약

형상관리가 소프트웨어 개발과 유지보수 단계에서 중요하게 사용되고 있다. 연구와 실제 구축을 거듭하면서 형상관리는 소프트웨어 개발의 성숙한 기술이 되었다. CBSD(Component Based Software Development)는 소프트웨어 개발의 새로운 패러다임으로 자리 잡고 있다. 즉, CBSD가 소프트웨어 재사용과 소프트웨어 컴포넌트 기술에 관한 연구로 시작되어 왔고, 소프트웨어 개발에 새로운 패러다임으로 인식되고 있다. 그러나 CBSD에 관한 형상관리 연구가 뒤따르지 못하였고, 관련 문헌도 상당히 미흡한 실정이다. 본 논문에서 설계한 모델은 CBSD를 더 효율적으로 지원하기 위하여 사용될 것이다. 또한 본 모델은 CBSD 개념을 이용한다. 이 모델은 전통적인 소프트웨어 형상관리(SCM : Software Configuration Management)와 관련이 있고 이를 컴포넌트 환경을 지원하도록 개선시킨 모델이다.

1. 서 론

정보시스템의 크기가 커지고 복잡해지면서 시스템의 구조와 구성, 제어구조, 상호작용 프로토콜, 또한 시스템의 특성 및 행위 등이 기존의 알고리즘과 데이터 구조보다 더 비중 있게 다루어지고, 그 중요성이 강조되어지고 있다. 이러한 문제를 해결하기 위한 방안의 하나로 CBSD을 도입하여 시스템을 개발하는 사례가 늘고 있다. 컴포넌트 기반 소프트웨어 개발의 목적은 새로운 Application을 개발하고자 할 때 검증되고 테스트된 컴포넌트 로직을 재사용 함으로써, 개발 기간을 단축하고 제품의 품질을 높이고자 하는 것이다. 그러나 저장소에 저장된 컴포넌트를 조립하여 시스템을 개발할 때, 컴포넌트의 버전이 바뀜에 따라 문제가 발생할 수 있다. 또한 개발된 시스템이 고정된 시스템이 아니라 컴포넌트 기반으로 개발되어 다른 시스템과 통합하여 사용될 수 있고, 자체적으로 업그레이드 될 수 있는 이른바 개방성(Open) 또는 유동성(Flexible) 있는 시스템 개발이 이루어지고 있다. 이러한 경우에 시스템이 실시간으로 업그레이드되기 때문에 제품의 형상에 영향을 준다. 결국 이러한 패러다임은 시스템 개발의 효율성(Efficiency)을 증가시키나, 제품 형상의 일관성(Consistency)을 감소시키는 위험에 빠질 수 있다[1]. 즉, 새로운 버전의 컴포넌트가 대체될 때 변경과 컴포넌트 사이의 관계가 불확실하다는 것과, 실시간으로 컴포넌트가 대체되면 하나의 패키지는 동작을 하지만 이전의 컴포넌트와 연결된 패키지는 동작하지 않을 수 있다는 것이다. 이러한 문제점의 해결방안이 컴포넌트 형상관리(CCM : Component Configuration Management)이다[2].

본 논문은 컴포넌트 형상관리를 통하여 버전이 바뀌는 경우, 이를 제어하고 확인하여 올바른 패키지가 작동될 수 있도록 하는 컴포넌트 기반의 형상관리 모델을 설계한다.

2. 기존 연구

2.1 소프트웨어 형상관리

소프트웨어 형상관리는 소프트웨어 개발 및 유지보수 과정에서 발생하는 소스 코드, 문서, 인터페이스 등 각종 결과물에 대한 계획, 개발, 운용 등을 종합하여 시스템의 형상을 만들고, 이들 형상에 대한 변경을 체계적으로 관리, 제어하기 위한 활동이다[3]. 형상관리 활동은 소프트웨어 Life-cycle 전체, 즉 인도 이전의 변경관리와 인도 이후의 변경관리를 모두 포함한다. 형상관리 내용으로는 Baseline과 소프트웨어 형상항목(Configuration Item) 등이 있고, 형상관리 기능으로는 소프트웨어 품질보증의 중요한 요소로서 중요한 임무는 변경(Change)를 제어(Control)하는 것이다. 그림 1과 같이 형상관리 기능은 형상식별(Configuration Identification), 형상제어(Configuration Control), 형상감사(Configuration Auditing), 형상 상태기록(Configuration Status Log)으로 구분된다.

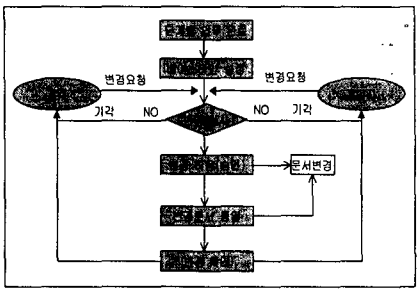

그림 1 소프트웨어 형상관리 개념도

2.2 컴포넌트 저장소

컴포넌트를 효율적으로 관리하기 위해서 체계적인 컴포넌트 저장소를 구축하여야 한다[4]. 이는 CBD 프로젝트를 진행하면서 발생하는 프로젝트 관리, 프로세스 관리, 데이터 관리, 문서관리, 변경 관리, 형상관리 등과 관련되어 있다. 컴포넌트 저장소는 소프트웨어 Life-cycle 동안 모아진 시스템 정보들을 관리하고 저장하는 곳이다. 저장소는 시스템 개발에 적용되는 각각의 도구, 개발 단계, 사용자 그리고 응용 프로그램 사이의 시스템 정보를 공유할 수 있도록 해준다. 컴포넌트 저장소의 관리 조직은 그림 2와 같이 컴포넌트와 관련된 정보과 컴포넌트 관리, 그리고 재사용 컴포넌트를 위한 기능 지원하는 조직으로 구성된다.

3. 컴포넌트 형상관리

컴포넌트 형상관리는 시스템 내부의 구성원들 사이의 일치성을 제어함으로써 신뢰도를 증가시키는 역할을 한다. 컴포넌트 기반으로 개발된 패키지에 새로운 컴포넌트가 추가 또는 대체될 때, 두 가지 문제점을 해결해야 한다. 첫째, 기존의 컴포넌

트가 다른 패키지와 연결되어 있을 경우 이를 대체시킬 때 발생하는 문제점이다. 즉 새로운 버전의 컴포넌트가 대체될 때 변경과 컴포넌트 사이의 관계가 불확실하다는 것이다. 둘째, 실시간 환경에서의 시스템에 대한 동적 행위는 더 심각한 문제점이다. 즉, 실시간으로 컴포넌트가 대체되면 하나의 패키지는 동작을 하지만 이전의 컴포넌트와 연결된 패키지는 동작하지 않을 수 있다는 것이다. 이러한 문제점의 해결방안이 컴포넌트 형상관리이다[2]. 즉, 시스템이 동작하고 있는 동안 컴포넌트를 관리하는 것은 어려운 일이다. 컴포넌트 시스템은 시스템을 생성하는 동안 컴포넌트가 알려지고 테스트된 버전이라면 단 한번 형상되어진다. 교체된 컴포넌트의 버전을 점검하는 것은 최소한 최초의 버전보다 같거나 새로워야 한다. 오래된 컴포넌트를 사용하는 것으로부터 시스템을 보호하는 시도는 새로운 컴포넌트가 설치되었을 때, 그 기능을 보장할 수 없는 일이 발생할 수 있기 때문이다[5].
컴포넌트 형상관리는 컴포넌트 식별, 형상모델, 변경관리, 의존관리 등으로 구분되어 관리되어진다.

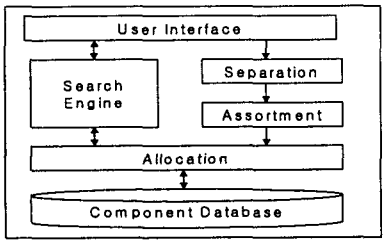

그림 2 컴포넌트 저장소 구조

3.1 컴포넌트 식별
의존성 그래프는 새로운 컴포넌트를 갖는 시스템의 업데이트 영향을 예측할 수 있는데 필요하다. 컴포넌트 이름, 생성 시간, 크기, 매직넘버(컴파일러가 주는 유일한 번호)를 식별하는 것은 지금까지 사용되어 왔다. 식별 날짜가 컴포넌트를 비교하는 유일한 키를 계산하는데 사용되어 왔다. 그 키는 식별(Identification)과 버전(Version)으로 관리된다.

3.2 형상 모델
형상모델은 컴포넌트가 어떻게 버전 통제 하에 있는지를 정의한다. 시스템 안에 설치된 컴포넌트의 집합은 컴포넌트 시스템이라 한다.

3.3 변경 관리
변경 관리는 내부 혹은 외부 컴포넌트 모두에 적용할 수 있다. 내부 컴포넌트의 경우는 변경 관리를 위하여 같은 Tool의 사용을 가능하게 하고, 컴포넌트 자체의 개발에서도 마찬가지이다. 외부 컴포넌트의 경우는 변경 관리가 변경의 감시를 할 수 있다. 일반적으로 컴포넌트는 공유 라이브러리로 구성된다[1]. 컴포넌트를 사용하는 프로그램은 직접 라이브러리(Library)를 사용하는 것이 아니라, 컴포넌트 인터페이스(Interface)를 참조한다. 즉, 라이브러리는 인터페이스의 구현을 의미한다. 여기서는 컴포넌트의 논리적, 물리적 단계의 변경 정보뿐만 아니라 관계(Relation) 정보까지 알아야 한다. 즉, 라이브러리와 인터페이스가 일치되어야 하기 때문에 컴포넌트 형상관리는 라이브러리와 인터페이스의 두 레벨에서 이루어져야 한다.

3.4 의존 관리
컴포넌트는 조립으로 구성된 복합체의 한 부분이다. 따라서 컴포넌트 형상관리는 기존의 소프트웨어 형상관리에서 사용하던 버전관리, 형상과 구축관리, 변경관리 이외에 의존관리가 추가되어 컴포넌트 형상관리가 이루어진다[7]. 컴포넌트 사이의 의존의 이점은 여러 가지이다. 즉 시스템에 적용된 것이 무엇인지 분석할 수 있고, 새로운 컴포넌트를 갖는 시스템으로 업데이트 될 때 시스템 생성에 관한 결정을 할 수 있도록 한다. 시스템 사이의 의존성은 그림 3과 같이 그래프로 표현된다. 컴포넌트는 그래프에서 Node이고 Edge는 의존성이다. 의존성은 모든 의존 집합을 정의할 수 있게 한다. 컴포넌트간의 의존성을 간결하게 표현하는 방법은 그림 4와 그림 5처럼 매트릭스 표현과 링크 리스트로 표현하는 방법이 있다.

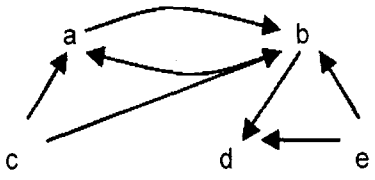

그림 3 Node a, b, c, d, e를 갖는 그래프

$$A = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 \end{bmatrix}$$

그림 4 매트릭스 표현

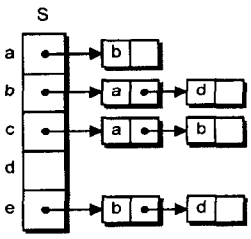

그림 5 링크 리스트로 표현된 그래프

4. CBSD를 지원하는 형상관리 모델 설계
4.1 CBSD에서의 형상관리 요구사항
형상관리 측면에서 보면 전통적인 소프트웨어 개발방법과 CBSD와의 차이가 있다. 첫째는 시스템 구축에 있어서 다른 언어의 사용이다. 두 번째로 시스템 구축에서의 다른 방법을 사용한다는 것이다. 전통적인 소프트웨어 개발 방법과 CBSD에 근거하여 CBSD의 형상관리 요구사항을 요약하면, 컴포넌트의 버전관리와 무결성(Integrity) 유지로 구분될 수 있다[6].

4.2 설계 기본개념
본 논문에서 설계한 모델은 CBSD를 더 효율적으로 지원하기 위하여 설계되었다. 그리고 본 모델은 CBSD 개념을 이용한다. 이 모델은 전통적인 SCM과 관련이 있고 개선된 모델이다. 이 모델에서 표현된 개념을 설명하면 다음과 같다.
• 파일 : 모델에서 파일은 기본적인 물리적 저장 단위를 뜻한다.
• 컴포넌트 : 컴포넌트의 개념은 CBSD에서 핵심요소이다. 컴포넌트는 시스템에서 필수의 논리적 구성요소이다.
• 형상과 형상항목 : 형상은 조립 컴포넌트를 표현하는데 사용되고 형상항목의 세트로 구성된다. 각 형상항목은 원시 컴포넌

트이거나 존재하는 형상이다.
• Baseline : Baseline은 조립 컴포넌트의 범위에 대한 기준이다.
• 관계 : 컴포넌트 간에는 하나의 컴포넌트가 다른 컴포넌트에 의존되는 것과 같은 관계가 있다.

4.3 관리되어지는 객체들

컴포넌트 형상관리에서 관리되는 객체는 원시(Primitive) 컴포넌트와 조립(Composite) 컴포넌트가 있다. 이는 조립되어지는 컴포넌트 자체와, 원시 컴포넌트를 이용하여 조립되는 컴포넌트를 구분하여 표기하였고, 이들 컴포넌트간의 관계도 표기하였다. 또한 본 모델은 원시 컴포넌트를 만들고, 이를 이용하여 조립 컴포넌트를 생산하는 전 과정의 형상관리를 지원한다.
• 원시 컴포넌트 : 원시 컴포넌트는 CBSD에서 기본적인 논리 단위이다. 원시 컴포넌트의 관리에 대한 작업은 체크아웃 및 체크인, 분리(Branching) 및 합병(Merging), 동시성 제어로 구성된다.
• 조립 컴포넌트 : 조립 컴포넌트는 CBSD에서 원시 컴포넌트와 조립 컴포넌트의 조합에 의하여 만들어진다. 요구사항이 변경되면 조립 컴포넌트는 컴포넌트 조합 후에 발전된다. 이는 형상을 생성하고 형상을 변경하며, 형상을 분리한다.
• 컴포넌트 간의 관계 : 컴포넌트 간의 관계는 객체들을 관리하는데 필요한 모든 것들로 취급된다. 컴포넌트 간의 관계를 관리하는 방법은 생성(Creation)과 추적(Tracing)이다.

4.4 설계된 모델

컴포넌트 기반 SCM 모델을 그림 6과 같이 설계하였다. 그림의 하단은 원시 컴포넌트의 진화를 나타내고, 상단은 조립 컴포넌트의 구축과 진화에 대하여 나타내고 있다. 이는 각각 SCM의 통제하에 있게 된다. 본 모델은 원시 컴포넌트 및 조립 컴포넌트 모두를 관리한다. 원시 컴포넌트를 관리하는 방법은 원시 컴포넌트의 체크아웃, 체크인, 분리, 합병 및 동시성이다. 조립 컴포넌트를 관리하는 방법은 형상 생성, Baseline 생성, 형상 변경, 및 형상 분리이다. 그림 7은 원시 버전을 갖는 컴포넌트를 나타내고, 그림 8은 Baseline에 의해 관리되는 형상이다.

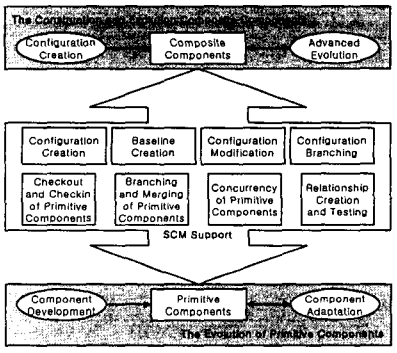

그림 6 컴포넌트 기반 SCM 모델

4.5 모델에서 제공하여야 할 중요한 기능들

본 모델에서 제공하는 중요한 기능은 컴포넌트의 유지관리를 위한 일관성이다. 이는 원시 컴포넌트의 일관성을 유지 관리하는 기능, 형상의 일관성을 유지 관리하는 기능, 이웃한 버전 및 Baseline의 일관성을 유지 관리하는 기능으로 구분되어질 수 있다. 또한 원시 컴포넌트의 동시성 통제 전략도 함께 지원되어야 할 기능이다.

5. 결 론

본 논문에서는 기존의 SCM에 대해서 살펴보고, 컴포넌트 형상관리에 대해서도 특징을 살펴보았다. 컴포넌트 형상관리 이론을 바탕으로 하여 CBSD 형상관리가 요구하는 사항을 바탕으로 하는 형상관리 모델을 개발하였다. 설계된 모델은 CBSD 형상관리를 보다 효율적으로 지원하기 위하여, 전통적인 소프트웨어 개발방법을 기초로 하는 기존의 형상관리 모델을 활용하여 새로운 컴포넌트 기반 소프트웨어 형상관리 모델이다.
향후 연구계획으로는 제안된 모델의 개념을 바탕으로 실제 시스템 설계를 통하여 형상관리 Tool로 구현되어 시스템 평가 및 성능을 시험하게 될 것이다. 이를 위하여 본 모델에서 중요하게 여겨지는 핵심 개념을 좀더 구체적으로 발전시키고 각각의 형상관리 방법을 시스템화 시키는 설계 및 구현작업이 뒤따를 예정이다.

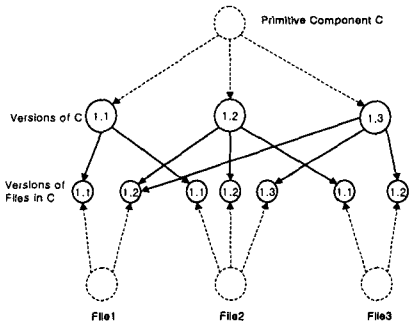

그림 7 원시 컴포넌트

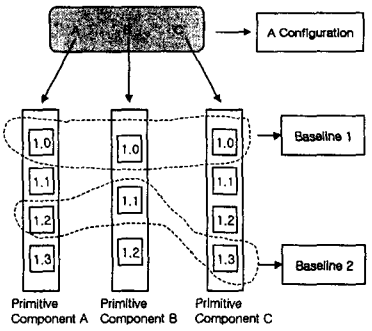

그림 8 형상

참고문헌

[1] Magnus Larsson, Ivica Crnkovic, "New Challenges for Configuration Management, In System Configuration Management", SCM-9, 1999
[2] Magnus Larsson, Ivica Crnkovic, "Configuration Management for Component-based Systems", In Software Configuration Management - SCM 10, 2001
[3] 윤청, 소프트웨어형상관리, *http://cs.cnu.ac.kr/~cyoun/*
[4] 김지현, 강병욱, "컴포넌트 저장소를 위한 업데이트 엔진 설계", 한국정보처리학회 추계학술발표대회 논문집 제9권 제2호, 2002
[5] Magnus Larsson, "Applying Configuration Management Techniques to Component-Based Systems," MRTC, 2000
[6] Ivica Crnkovic, Brahim Hnich, Totte Jonsson, Zeynep Kiziltan, "Specification, Implementation, and Deployment of Components", Communications of the ACM (CACM), 2002
[7] Magnus Larsson, Ivica Crnkovic, "Component Configuration management", In ECOOP Conference, Workshop on Component Oriented Programming Nice, 2000
[8] George T. Heineman, William T. Councill, "Component-based Software Engineering", Addison-Wesley, 2001