제 19 회 한국정보처리학회 춘계학술발표대회 논문집 제 10 권 제 1 호 (2003. 5)

동적환경에서의 컴포넌트 정보를 이용한 형상관리기법에 관한 연구

박경석*, 김종완*, 류성열*
*숭실대학교 컴퓨터학과
e-mail : {p6891, wany}@selab.ssu.ac.kr, syrhew@computing.ssu.ac.kr

A Study of Configuration Management Technique by using Component Information in Dynamic Environment

Kyung-Seok Park*, Jong-Wan Kim*, Sung-Yul Rhew*
*Dept. of Computer Science, Soonsil University

요 약

최근 소프트웨어 개발 및 운영 환경은 C/S, 분산환경, 다중 플랫폼, 다양한 방법론 등으로 매우 복잡해 지고 있으며, 이러한 환경에서 만들어지는 모든 리소스들을 효율적으로 관리한다는 것은 쉽지 않은 일이다. 이에 따라 최근 몇 년 동안 국내의 개발환경에 효과적으로 적용할 수 있는 소프트웨어 형상관리(SCM: Software Configuration Management)시스템들이 개발되고 있다. 그러나 지금까지의 형상관리시스템들은 버전관리, Build 관리, 작업공간관리, 프로세스관리들에 국한되어 있다. 본 논문에서는 컴포넌트의 형상관리에 대한 방법을 제시하고 있다. 먼저, 컴포넌트의 정보를 리파지토리에 저장하여 관리한다. 리파지토리에 저장된 정보를 이용하여 컴포넌트의 버전, 크기, 개발환경, 데이터베이스, 만든날짜, 수정날짜, 엑세스한 날짜들을 파악할 수 있다. 또한 컴포넌트가 실행 될 때 연동되어 사용대는 컴포넌트 객체들을 파악할 수 있으며, 그 access 빈도수를 도표화하여 개발자나 관리자가 모니터링이 가능하도록 지원한다.

1. 서론

오늘날의 개발환경은 다양한 종류의 개발도구들과 다양한 종류의 OS 로 구성되어 있으며, 많은 개발자들이 함께 작업함으로써 많은 개발 소스 파일들을 체계적으로 관리할 필요가 있다. 그리고 최근 시스템들은 대형화, 복잡화되고 있어 소프트웨어에 대한 형상관리를 하지 않고서는 소프트웨어에 대한 품질을 확신하기가 어렵다. 또한, 소프트웨어의 각 개발공정별로 발생하는 다양한 형태의 정보를 체계적으로 수집, 변환, 분석, 보관, 보고하며 개발공정을 관리하여 낭비와 비능률을 제거하여야 한다. 이러한 작업들을 수작업으로 관리하는데는 한계가 있으며, 많은 시간과 노력이 필요하기 때문에 형상관리 도구가 필요하게 되었다.

이러한 지원도구는 각 소프트웨어 규모와 특성에 따라 지원가능한 형태이어야 한다. 소프트웨어 개발과정상 다양한 형태의 소프트웨어 개발정보가 관리 가능하게 해야 하며, 소프트웨어의 규모별로 가장 능률적인 개발공정 표준과 관리지원체계가 구축되어야 한다. 또한 각 개발 공정별로 품질보증을 할 수 있도록 소프트웨어의 성격과 규모별로 품질관리 체계를 구축하여야 한다. 이렇게 해서 소프트웨어의 생산과정이 체계적이고 과학적관리가 되면 생산성 향상과 제품의 완전성을 꾀할 수 있게 된다.

소프트웨어 형상관리(SCM)는 소프트웨어 가시성 결여로 인해 발생하는 소프트웨어 부재현상을 극복하고자 연구된 것으로 소프트웨어를 각 구성요소별로

구분하여 관리한다. 즉 소프트웨어의 전생명주기를 통해 생성되는 산출물을 관리하기에 효율적인 크기의 형상으로 식별하고, 이에 대한 변경을 체계적으로 제어하고, 형상에 대한 추적성과 일치성을 유지하기 위하여 여러 기법을 통하여 관리하고자 하는 행위를 말한다.

소프트웨어 형상관리의 역할은 다음과 같다[5][6]. 첫째, 이전 리비전, 버전에 대한 정보를 언제든지 접근할 수 있어야 한다. 둘째, 불필요한 사용자가 소스를 수정할 수 없도록 해야 한다. 셋째, 동일한 프로젝트에 대하여 여러 개발자가 동시에 개발할 수 있어야 한다. 넷째, 에러가 발생했을 경우 빠른시간내에 수정할 수 있어야 한다. 마지막으로 사용자의 요구에 따라 적시에 최상의 소프트웨어를 공급할 수 있어야 한다.

본 논문의 구조는 다음과 같다. 2 장에서는 소프트웨어 형상관리의 개념 및 구성요소, 컴포넌트에 대하여 살펴보고, 3 장에는 컴포넌트 형상관리에 대한 방법들을 제시하여 프로토타입을 보여주고 있다. 마지막으로 4 장에는 결론과 향후 발전 방향을 제시한다.

2. 관련연구

2.1 소프트웨어 형상관리의 개념 및 구성요소

소프트웨어의 형상관리란 보이지 않는 소프트웨어 개발 과정을 시각화하기 위해서 소프트웨어의 각 구성항목을 관리하는 것을 말한다. SCM 은 개발과정에서 변화되어가는 소프트웨어의 형상을 질서있게 통제하고 또한 개발과 유지보수시 소프트웨어의 변경을 수렴하는 새로운 소프트웨어공학의 개념이다. [4]

일반적으로 형상관리는 4 가지 활동 즉, 형상식별, 형상통제, 형상상태보고, 형상감사라는 기본적인 활동으로 구성되어 있다(그림 1) [2][3].

(1) 형상식별(Identification)

형상 항목을 제어하고 관리하기 위해서 독립적인 이름이나 식별자를 갖게 하는 것이다.

(2) 형상통제(Change Control)

형상 변경을 위한 일련의 정책으로, 변경의 필요성을 식별하는 것으로부터 변경을 실행하여 새로운 버전에 변경을 포함한 후 새로운 버전을 배포하는 순서에 이르는 동안 따라야 할 규칙들이다. 이러한 순서에 따라 사용자는 소프트웨어 저장소 혹은 프로젝트 데이터베이스에 존재하고 있는 각 형상항목에 접근하게 된다. 이때 중요한 것이 권한을 관리하는 접근관리(Access Control)와 다른 두 사람이 수행하는 병렬 변경이 서로 겹치지 않게 도와주는 동기관리(Synchronization Control)이다.

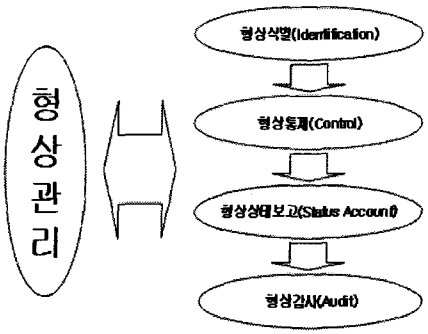
(3) 형상상태보고(Status Account)

무슨 일이 발생했고, 누가 그것을 수행했으며, 언제 발생했고, 그로 인해 어떠한 것이 영향을 받는지 조사하는 활동이다.

(4) 형상감사(Audit)

형상관리가 '잘' 되고 있는지 검사하는 것이다. 예를 들면, 소프트웨어공학 표준들을 적적하게 준수했는지 조사한다든지, 실제 변경이 만들어 졌는지, 변경날짜와 변경자가 명시되었는지 조사하는 것이다.

이러한 형상관리를 통하여 프로그램 설계문서나 코드의 재사용을 넘어서 분석 및 영업, 유지보수자료까지 포괄적으로 관리를 하게 된다. 현재 형상관리는 ISO, CMM, SPICE 등 소프트웨어 개발 인증 및 프로세스 평가 기준에서 소프트웨어 개발 프로젝트의 필수 요소로 지정할 만큼 중요하게 인식이 되고 있다.



(그림 1) 형상관리 기능

SCM 은 단순히 라이브러리 관리나 임의대로 부여되는 버전관리가 아닌 변경관리, 빌드와 릴리즈관리 등 이 모든 요소들을 완전히 충족시킨 관리 형태를 말한다. 제품의 라이프 사이클동안 변화하고 있는 파일· 프로그램· 제품들을 저장하고 관리하는 버전관리, 소프트웨어 변경을 계획· 구현 · 모니터링· 제어· 기록· 리포트하는 변경관리, 소프트웨어 제품의 릴리즈 또는 빌드에 반영된 변경을 감사하고 관리하는 빌드와 릴리즈관리들을 형상관리의 중요 구성요소라고 할 수 있다[7].

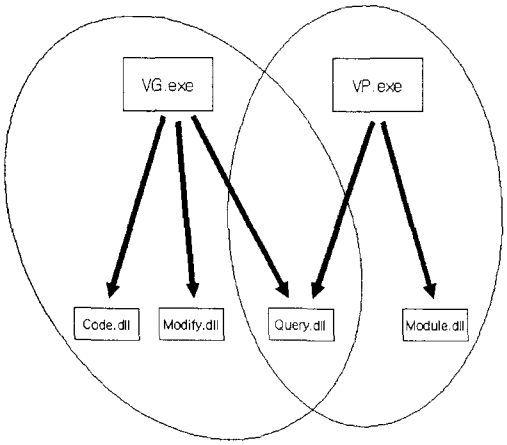
2.2 컴포넌트

컴포넌트는 독립적으로 배포가 가능하며 잘 정의된 소프트웨어 구조에서 특정하게 정의된 기능을 수행하는 대체 가능한 소프트웨어 부품이다[8]. 컴포넌트를 이용한 소프트웨어 시스템을 구축하는 것은 이미 검증된 컴포넌트의 합성 및 조립을 의미한다. 또한, 각 컴포넌트는 독립된 활용의 단위로 각각의 생명주기와 각각의 버전을 갖는다. 효율적인 컴포넌트를 사용하기 위해서는 각 컴포넌트의 버전을 관리해야 한다. 컴포넌트는 독립적 배포단위를 가지며 동일한 인터페이스

를 갖는 다른 컴포넌트와 대체가 가능하며, 다른 컴포넌트와 상호 동작하도록 개발된다. 컴포넌트를 사용하면 소프트웨어 개발의 유연성을 주고 반복적인 개발을 감소 시켜 유지보수 비용도 절감된다. 컴포넌트는 컴포넌트의 사용자가 구축하려는 어플리케이션의 기능에 따라 달리 사용되어지는 것이다.

3. 컴포넌트 형상항목

컴포넌트란 실질적으로 개발단계에서 파악할 수 있는 것이 아니라 실행단계에서 동적으로 연동 될 때 파악 할 수 있을 것이다. 예를 들어 보면 다음 (그림 2)과 같다. VG.exe 가 실행 될때는 Code.dll, Modify.dll, Query.dll 이 연동이 되며, VP.exe 가 실행 될 때는 Query.dll, Module.dll 이 연동된다. 즉 VG.exe 와 VP.exe 는 Query.dll 이라는 컴포넌트 객체를 같이 사용하고 있는 것이다. 즉 컴포넌트를 관리 하기 위해서는 VG.exe, VP.exe 컴포넌트 뿐만 아니라 거기에서 동적으로 연동되어 관계를 가지고 있는 컴포넌트 객체들도 형상관리를 하여야 한다.



(그림 2) VG.exe 와 VP.exe 컴포넌트

그러므로 컴포넌트 형상관리함은 정적인 관점보다는 동적인 관점에서 접근을 하여야 한다. 다음 [표 1] 은 현재 사용되는 컴포넌트 모델 COM, JavaBeans, CORBA 를 비교하고 있다. [1]

[표 1] COM, JavaBeans, CORBA 비교

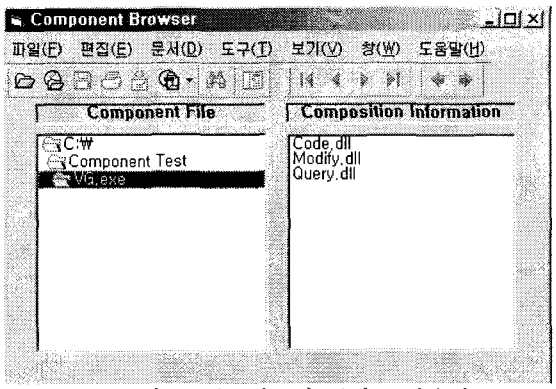
Feature	COM	JavaBeans	CORBA
Instantiation	Dynamic	Dynamic	Dynamic
Distribution	DCOM	RMI	IIOP
Interfaces	Multiple	Single	Single
Interface Inheritance	Single	Multiple	Multiple

4. 컴포넌트 형상관리

컴포넌트와 컴포넌트 객체의 버전, 날짜, 크기등의 정보들을 리파지토리에 저장하여 관리를 한다. 또한 컴포넌트에 동적으로 연동되는 컴포넌트 객체들을 관리 함으로써 다른 컴포넌트에서의 컴포넌트 객체의 사용 빈도수를 추출할 수 있다. 이 논문에서는 다음과 같은 모니터링들을 함으로써 관리자나 개발자에게 컴포넌트의 세부적인 정보까지 관리 할 수 있도록 도움을 준다.

4.1 컴포넌트의 구성 관리

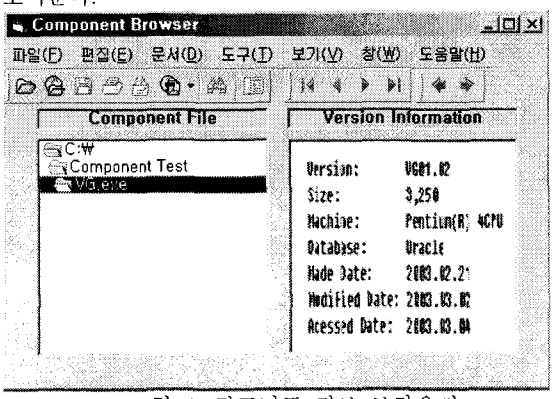
이 브라우저는 컴포넌트를 구성하고 있는 컴포넌트 객체를 보여준다. 아래의 (그림 3)과 같이 VG..exe 컴포넌트에 포함되어 있는 컴포넌트 객체는 Code.dll, Modify.dll, Query.dll 로 구성이 되어 있다.



(그림 3) 컴포넌트의 구성 브라우저

4.2 컴포넌트 정보 관리

이 브라우져는 컴포넌트의 버전, 크기, 개발환경, 데이터베이스, 만든 날짜, 수정날짜, 엑세스 날짜들을 보여준다.



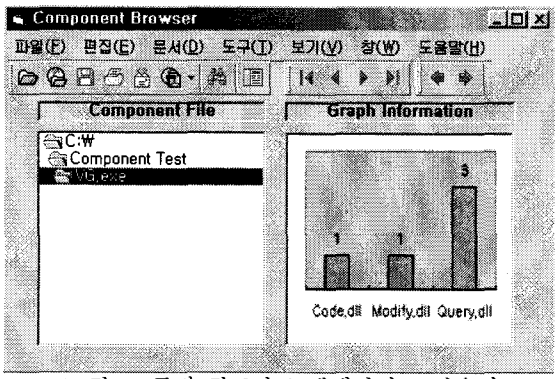
(그림 4) 컴포넌트 정보 브라우저

(그림 4)를 보면 버전은 VG01.02 이며, 크기는 3,250

이다. 작업환경은 펜티엄 4 이며, 데이터 베이스는 Oracle 를 사용하고 있다. 또한 이 컴포넌트가 만들어진 날짜는 2003.02.21 이며, 수정날짜는 2003.03.02 이고, 엑세스 날짜는 2003.03.02 임을 알 수 있다. 이 정보는 리파지토리에서 관리를 해 주며, 컴포넌트의 전반적인 정보를 알 수 있게 해 준다.

4.3 동적 컴포넌트 객체 관리

컴포넌트가 실행될 때 동적으로 연동되어지는 컴포넌트 객체에 대한 정보를 관리한다. 컴포넌트 객체의 정보를 리파지토리에 저장함으로써 다른 컴포넌트에서 이것을 사용할 시에는 Assess 포인트를 증가하여 사용빈도수를 체크하여 보여준다.



(그림 5) 동적 컴포넌트 객체관리 브라우저

(그림 5)를 보면 컴포넌트에서 사용하고 있는 컴포넌트 객체에 대한 사용빈도수를 파악할 수 있다. 그림 VG..exe 의 컴포넌트에서는 3 개의 컴포넌트 객체 Code.dll, Modify.dll, Query.dll 로 구성이 되어 있다. Code.dll, Modify.dll 은 VG..exe 컴포넌트에만 동적으로 사용이 되어지는데 반하여, Query.dll 은 다른 컴포넌트 사용시에 동적으로 연동됨을 알 수 있다. 컴포넌트의 수정 및 삭제시에 모니터링 참조하여 관리자나 개발자가 파일에 대한 정보를 다시 검토 할 수 있을 것이다. 만약 컴포넌트를 삭제 할 경우, 한번씩 사용이 되는 Code.dll, Modify.dll 은 삭제 가능하나 Query.dll 은 삭제가 되면 안 된다. 만약에 VG..exe 에 포함되어 있는 파일을 모두 삭제한다면 다른 컴포넌트에서 기능을 제대로 하지 못하고 에러를 발생한다.

5. 결론

본 논문은 소프트웨어 형상관리 항목 중 컴포넌트 형상관리를 제시하고 있다. 컴포넌트 정보를 리파지토리에서 관리하여 기능을 알 수 있도록 도와주며, 컴포넌트와 연동되는 컴포넌트 객체들을 체크하여 그래프 형식으로 보여주고 있다. 이러한 방법을 사용 할 경우, 컴포넌트의 생성, 수정, 삭제 시 의존성을 파악하여 관리를 할 수 있다. 또한 생산성이 향상되어 재사용의 활용으로 신속한 컴포넌트 제공과 재사용 비용의 극대화를 가져올 수 있다.

앞으로 향후 연구과제는 컴포넌트 객체에 대한 버전관리를 구체화하는 것이다. 컴포넌트가 컴포넌트 객체와 연동하여 사용하게 될 때 실질적으로 호출되는 함수들의 목록 데이터를 관리하여 파악할 수 있도록 하는 것이다. 이것을 함으로써 컴포넌트 객체에 대한 소스 정보를 파악할 수 있을 것이다. 또한 관리자와 사용자를 위한 자동화 도구를 개발하여 컴포넌트에 대한 정보를 가시적으로 볼 수 있도록 할 것이다.

참고문헌

[1] Magnus Larsson, Ivica Crnkovic "Component Configuration Management" In proceedings ECOOP Conference, Workshop on Component Oriented Programming Nice, France, June 2000.

[2] Crnkovic I., "Experience with Change-oriented SCM Tools", In Proceedings of 7th Symposium on Software Configuration Management, Lecture notes in Computer Science, nr 1235, Springer Verlag, 1997.

[3] R. Conradi and B. westfechtel, Version Models for Software Configuration Management, Software Configuration Management Symposium, SCM-7, 1997, Springer, ISBN 3-540-63014-7, ACM Computing Surverys, VOL. 30, NO.2, June 1998.

[4] H. Thane, A Wall, formal and Probabilistic Arguments for Component Reuse in Safety-Critical Real-Time System, Technical report CBSE, State of the Art, Mardalen Unversity, 2000

[5] C, Burrows, G. George, and S. Dart, Configuration Management, Ovum Ltd., 1996

[6] J.J. Hunt, F. Lamers, J. Reuter, and W.F> Tichy, Distributed Configuration Management via java and the World Wide Web, In Proceedings of the Seventh International Workshop on Software Configuration Management , New York, 1997.

[7] Dart Susan, "Concepts in configuration Management System", In Proceedings of 3rd International workshop on Software Configuration Management, ACM Press, 1991.

[8] Kangtae Kim, jeMin Bae, Jeong Ah Kim, Kying Whan Lee, "Developing O-O Framework for Web Collaboration System,: 17th IASTED International Conference Proceeding, pp.165-167. 1999.