2001년 한국정보처리학회 춘계 학술발표논문집 제8권 제1호

웹 기반의 분산 객체 지향 소프트웨어 개발 환경을 위한 중앙 버전 관리자

o
김수용* 최동운*
서남대학교 전자계산학과
*e-mail:vibe@kornet.net
**e-mail:cdo@tiger.seonam.ac.kr

Central Version Manager for Distributed Object-Oriented Software Development Environment Based on Web

Soo Yong Kim, Dong Oun Choi
Seonam University Computer Science

요 약

본 논문에서는 웹 기반의 분산 소프트웨어 개발 환경에서 원시 코드 중심의 버전뿐만 아니라 원시 코드 이전 단계의 UML 기반의 소프트웨어 개발 환경에서 발생하는 다양한 설계 객체들을 일정한 형태로 구축하여 효율적으로 관리하는 방법론을 제시하였다. 또한, 웹을 기반으로 한 분산 소프트웨어 개발 과정에서 발생하는 버전들을 일관성 있게 관리하기 위해 버전 규칙에 기초한 웹 기반의 중앙 버전 관리자를 설계하였다.

1. 서 론

최근 개발되고 있는 소프트웨어들은 규모가 대용량이고 복잡하여지는 경향을 가지게 되는데, 이를 위해서 객체 지향 개발 방법론이 소개되어 많이 이용되고 있으며, 현재까지 많은 객체 지향 CASE 도구들이 사용되고 있지만 이들은 모두 지리적으로 인접한 지역에서 독립적으로 소프트웨어 시스템을 개발하는 도구만을 제공하고 있다. 최근에 분산 환경에서 협동적인 소프트웨어 개발을 위해 많은 연구가 진행되고 있다[1,7]. 그러나, UML(Unified Modeling Language)은 웹 상의 분산 개발 팀들 간의 설계 정보를 공유할 수 있는 적절한 방법을 제공하고 있지 못하고 있다. 본 연구팀은 웹을 기반으로 한 분산 소프트웨어 환경에서 하나의 프로젝트를 수행하는 과정에서 발생되는 다양한 버전들을 관계형 데이터 베이스를 이용하여서 저장 관리하는 버전 관리 방법에 관한 연구이다[3]. 웹 환경의 분산 객체지향 소프트웨어 개발은 개발자들 간의 공동 작업을 원활하게 하기 위한 다양한 기능들을 제공한다. 따라서 연구의 최종 목적은 기존의 원시 코드 중심의 버전 관리 시스템에 원시 코드 이전 단계인 UML을 기반으로 한 소프트웨어 개발 환경의 전체 개발 주기에서 발생하는 다양한 설계 객체들을 관리하는 웹 기반의 규칙 버전 관리자를 개발하는 것이다. 본 논문에서는 중앙버전 관리자에 관한 내용을 기술하겠다.

본 논문에서 전개될 내용은 다음과 같다. 먼저 2장에서는 버전이 발생되는 UML 소프트웨어 개발 환경과 기존의 버전 제어에 대한 특징을 살펴보고, 3장에서는 중앙버전 관리자를 설명하였다. 마지막으로 4장에서는 결론 및 앞으로의 연구과제를 논의한다.

2. 관련 연구

2.1 기존의 버전 제어

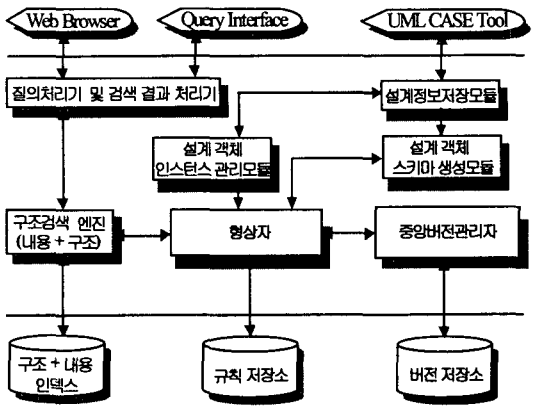
소프트웨어 개발 중에 발생하는 식별 가능하고 기계가 인식할 수 있는 문서 종류들을 소프트웨어 객체(software object)라 정의한다[11]. 소프트웨어 객체는 원시 객체(source object)와 유도 객체(drived object) 등으로 구성된다. 원시 객체는 소프트웨어 개발자에

의하여 생성되는 소프트웨어 객체이다. 유도 객체는 코드 생성기, 컴파일러, 링커, 문서 생성자(document formatter)와 같이 유도자(driver)라 불리는 프로그램이 원시 객체들로부터 자동으로 생성되는 객체이다. Rational Software사의 ClearCase는 현대 소프트웨어 개발의 복잡성을 동적으로 해결해 주는 실용적인 소프트웨어 형상 관리 시스템으로 4가지 기능적 영역으로 나누어 설명하는 게 유용하다. 4가지 기능에는 작업공간 관리, 관리 구축, 프로세스 제어, 버전제어가 있다. 그 중에 Rational ClearCase의 Version Control는 모든 화일과 디렉토리에 변경을 추적하는 것에 의해 버전을 완전하게 유지한다. Rational CelarCase의 핵심 개념은 버전 트리이며, 그것은 그래픽으로 정보를 표현하고 구조화하며, 디렉토리 트리와 비슷한 계층적인 포맷으로 표현하는 특징을 갖는다. 그러나 Rational ClearCase는 파일과 디렉토리 기반으로 버전을 관리함으로써 웹 상의 분산 개발 팀들 간의 설계 정보를 공유할 수 있는 적절한 방법을 제공하고 있지 못하고 있다. 즉, 웹 기반의 분산 환경에서 소프트웨어 개발하는 팀들이 설계 정보를 공유하고, 관리하기에 많은 어려움을 느낀다. 본 논문에서 제안하는 웹 기반의 버전 관리자는 UML의 설계 정보를 관계형 데이터베이스에 저장 관리한다. 즉, 설계 정보의 의미 정보와 표기 정보를 관계형 테이블에 사상하여 웹 기반의 분산 환경에서 개발자들간의 공동 작업이 가능하도록 하였다.

3. 웹을 기반의 버전 관리자

3.1 웹 기반의 규칙 버전 관리자 시스템 구조

본 연구는 웹을 기반으로 한 분산 소프트웨어 개발 과정에서 발생하는 버전들을 일관성 있게 관리하기 위해 버전 규칙에 기초한 버전 제어 기법을 제안하였다. 규칙 기반 버전 시스템은 하부 저장 시스템으로 관계형 데이터베이스 시스템을 사용하였다. 이 저장 시스템은 질의 처리 및 검색 결과 생성기, 형상자, 중앙버전 관리자 등으로 구성된다. 본 시스템의 전체 구조는 객체 지향 CASE 편집 도구에서 생성되는 설계 정보들을 관계형 데이터 베이스인 설계 정보 저장소에 저장하게 된다. 규칙 저장소에 저장된 규칙에 의해 설계 정보들은 중앙버전 관리자를 통해서 저장소에 저장된다. 버전 저장소의 설계 정보들은 웹을 통해서 개발자들 간에 상호 공유하고, 개발 도구를 위해서는 CASE 편집 도구의 *.mdl 파일 형태로 변환되어서 공유한다. 규칙 기반 버전 추출 시스템의 전체적인 개괄 구조는 아래 (그림 1)과 같다.



(그림 1) 규칙 기반 버전 관리자 시스템 구조

설계 정보 저장 모듈은 실제 .mdl 파일을 저장하기 위한 스키마 생성 및 인스턴스의 저장 및 검색을 담당한다. 구조검색엔진은 구조 검색 및 애트리뷰트 검색을 위한 인덱스를 생성하고 관리한다. 질의처리기는 웹 브라우저를 이용하여 사용자가 질의할 때 이를 분석하여 구조 검색, 애트리뷰트 검색, 내용과 구조 검색 등 혼합 검색은 구조 검색 엔진을 사용한다. 그리고 검색 결과 생성기에서는 검색엔진에서 찾아진 내용을 문서의 전체 혹은 일부분을 사용자에게 보여 줄 수 있는 형태로 변환하여 사용자의 웹 브라우저에 보내진다. 설계 정보저장 모듈은 설계 객체를 규칙에 맞게 생성하기 위해서 개발되었다. UML에서 모델링한 클래스나 설계객체의 특성을 고려하여 스키마를 설계하여 관계형 데이터베이스에 생성한다. 또한 설계 객체, 구조 정보 등을 데이터베이스에 저장하여 이를 사용자가 원하는 문서나 이의 일부분 검색하는데 사용한다. 설계 정보 저장모듈은 또한 규칙에 맞는 버전을 생성하는 형상자와 인터페이스 역할을 한다.

3.2 중앙 버전 관리자

3.2.1 버전 관리자의 설계

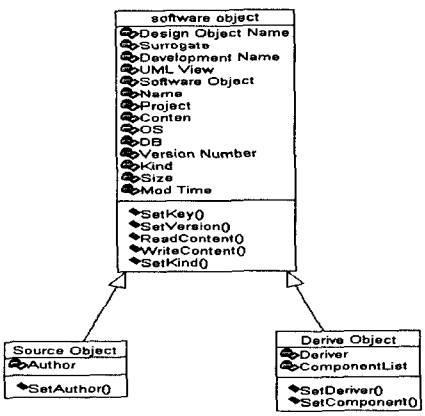
버전 관리 도구는 다음의 기본 기능을 제공해야 한다. 첫째로 버전 제어 아래에 소프트웨어 객체를 놓기 위해 주어진 내용과 프로젝트 이름을 줄 수 있는 기능을 가져야 한다. 둘째로 소프트웨어 객체를 추론하기 위해서는 버전 제어 아래에 이미 소프트웨어 객체

가 존재해야 한다. 셋째로 수정 제어 시스템에서 수정된 각 소프트웨어 객체의 수정 히스토리 기능을 가져야 한다. 넷째로 연합 소프트웨어 객체로부터 소프트웨어 객체를 제거하기 위해서는 내용과 프로젝트에 이름을 주어야 한다.

상이한 클래스를 정의하고 사용자 인터페이스 설계를 생각해 볼 때, 단순하게 하기 위해 전통적인 설계 개념을 적용하며, 모든 원소의 완전한 뷰를 위해서는 구현한 것과 추가한 코드를 참조해야 한다.

1) 소프트웨어 객체 설계

시스템은 다른 내용과 분산 프로그램의 개념을 받아들인 버전 관리 서비스 제공을 강화했다. 여러 장소에서 많은 사람이 버전제어 서비스를 제공 받으며 개발한다고 가정할 때, 버전 도구는 분산환경에서 존재하며 새로운 서비스의 통합이 가능하게 되고, 설계와 구현을 계속해서 유지한다. 소프트웨어 객체 클래스 계층의 개요는 (그림 2)와 같다.

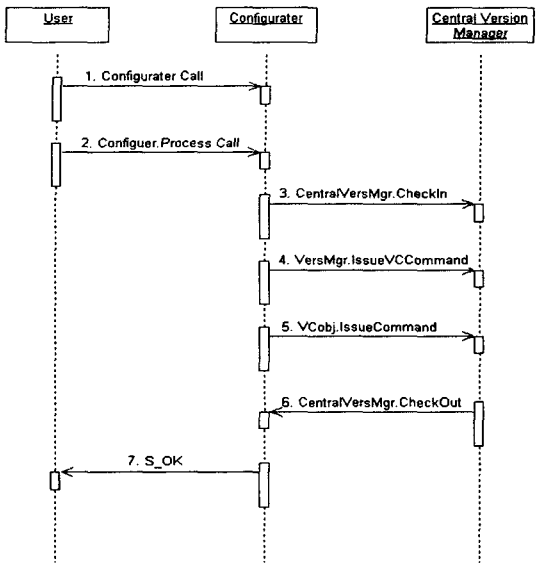


(그림 2) 소프트웨어 객체 클래스 계층

멤버 기능으로 setkey(IN 설계객체명, IN 개발단계명, IN UML관점명, IN 프로젝트명)는 소프트웨어 객체 집합을 식별하는 키이다. Setversion(IN Version)은 명확한 버전에 의해 소프트웨어 객체를 식별하는 것이다. ReadContents()는 설계객체를 데이터베이스에서 주기억장치로 읽어드린다. SetAthor(IN Author)는 원시 소프트웨어 객체를 만든 사람을 정의한다. SetComponentList(IN Component List)는 구성요소와 수정이라는 유도 소프트웨어 객체들을 결합했을 때 발생한다.

2) 체크인 프로세스

버전 관리 도구를 사용하여 소프트웨어 객체를 체크인 했을 때 어떤 단계들이 일어나는지 중요한 동작들만 알아보자. 다음은 형상자와 중앙버전관리자 사이의 체크인과 체크아웃 하는 예이다. 첫째로, 사용자는 명령 셀에 찾고자 하는 소프트웨어 객체에 대해 CI<options> <설계 객체명, UML 관점명, 개발단계명, 버전번호, 운영시스템, 데이타베이스>"를 입력한다. 또한, CI는 사용자에 의해 제공된 튜플과 옵션을 정리해서 조사한다. 각각에 명시된 설계객체에 대해서 CI는 소프트웨어 객체를 만들고, 그에 따른 키를 설정하고, 소프트웨어 객체를 읽어들인다. 둘째로, 명시한 설계 객체에 대해, CI는 설계객체를 생성하고 요구하는 서비스를 제공하기 위해 Configuer.Process를 호출한다. 셋째로, Configuer.Process는 중앙버전 관리자에 체크인 하고 VersMgr.IssueVCCommand를 호출 한다. 넷째로, VersMgr.IssueVCCommand는 버전 제어 객체를 만들고 파라메터 속성을 설정하며, 버전 제어 객체를 만드는 VCObj.IssueCmd를 호출한다. 다섯째로, VCObj.IssueCmd는 RCS 입력을 임시 파일에 기록하고 올바른 명령 행을 구성하며 시스템 함수를 통해서 이와 같은 명령행 COUT<options> <설계 객체명, UML 관점명, 개발단계명, 버전번호, 운영시스템, 데이타베이스>"을 출력한다. 시스템 호출로부터 실행 포인터가 되돌아 왔을 때 그 함수는 결과를 반환한다. 여섯번째로, CentralVersMgr.CheckIn은 RCS의 상태를 받아들인다. 일곱 번째로, CentralVersMgr.CheckOut은 RCS 상태를 반환하고, 여덟 번째로, S_OK는 결과를 출력한다. 만약 사용자가 즉각적인 후속 점검을 요구했다면 CentralVersMgr.Checkout은 동일한 소프트웨어 객체에 대해서 호출한다. 만약에 즉각적인 후속 점검이 이루어지지 않는다면 CI는 다음 파일 이름으로 넘어간다. 상태변수는 얼마나 많은 소프트웨어 객체가 적절한 점검을 받을 수 있는지를 나타낸다. 다음 (그림 3)은 버전 관리자의 체크인, 체크아웃 되는 과정이다.



(그림3) 버전관리자의 체크인, 체크아웃 되는 과정

4. 결론 및 향후 연구과제

본 논문에서는 기존의 원시코드 중심의 버전제어 시스템에 원시 코드 이전 단계 UML을 기반으로한 소프트웨어 개발 환경에서 다양하게 발생되는 설계 객체들을 효율적으로 관리할 수 있는 버전제어 관리 규칙을 제안하였다. 우리의 중앙 버전 관리자는 웹 기반의 분산 설계 환경에서 효과적으로 설계 객체를 관리할 수 있도록 고안되었다. 보다 더 많은 개발자가 자유로운 장소에서 자유로운 시간에 데이터를 공유하며 서로의 의견에 대해 교환할 수 있다. 그렇지만, 아직 설계 객체 저장 구조를 완전히 설계하지 못하였으며, 앞으로 완전한 시스템으로의 구현 작업이 연구과제로 남아있다.

참 고 문 헌

[1] Black, E. Paul, "GDIST : a distributed configuration control system," in Berichte des German Chapter of the ACM, Inter. Workshop on Software Version and Configuration Control, Teubner, pp.276-284, 1988.

[2] M. Chen and R. Norman, "A Framework for Integrated CASE," IEEE Software, pp.18-22, Mar., 1992.

[3] 최동운, "객체 지향 설계 데이터 관리자를 위한 시각 질의 처리기", 박사학위논문, 전북대학교, 1997.

[4] B. Korel, Wedde, Horst, Nagaraj, Srinivas, Nawas, Kaliaque, Dayana, Venugopal, Santhanam, Babu and Xu, Mandai, "Version management in distributed network environment," in Proc. of the 3rd Inter. Workshop on software Configuration Management, Trondheim, June 12-14, 1991.

[5] M. Rochkind, "The source code control system," in IEEE transaction Software Engineering, vol. 1 (4), December, pp.255-265, 1975.

[6] W. F. Tichy, "RCS - A System for Version Control," Software-Practice and Experience, Vol. 15, No. 7, 1986.

[7] A. Wasserman, "Tool Integration in Software Engineering Environment," Int'l Workshop on Environment,. F. Long, ed., Springer-Verlag, Berlin, pp.37-149, 1990.

[8] Martin Fowler, Kendall Cott, UML Distilled, Addison-Wesley Longman. 1997.

[9] Hans-Erik Eriksson, Magnus Penker, UML Toolkit, John Wiley & Sons, 1998.

[10] M. Gaedke, Hans-W. G., A. Schmidt, Ulf S., Wolfgang Kurr, "Object-oriented Web Engineering for Large-scale Web Service Management", Proc. of the 32nd Hawaii international conference on System Science, IEEE, 1999.

[11] http://www.rational.com/uml, Rational Rose Co.