論文 97-22-9-4

CMS-MX 소프트웨어의 버전 관리

正會員 신 재 욱*, 박 광 로**, 이 남 준*

Version Management of CMS-MX Software

Jaewook Shin*, Kwangroh Park**, Namjun Lee* *Regular Members*

요 약

CMS-MX(CDMA Mobile System-Mobile eXchange) 이동통신교환기 소프트웨어는 그 규모가 크고 새로운 기능 추가에 따른 지속적인 변경이 요구되기 때문에 체계적인 버전 관리를 필요로 한다. 그러나 CMS-MX 소프트웨어는 다양한 형태의 파일들로 구성되어 있고 고유의 소프트웨어 개발 체계를 따르기 때문에 범용의 버전 관리 시스템을 그대로 적용하기가 어렵다. 본 논문에서는 CMS-MX 소프트웨어의 개발 환경 및 개발 체계에 적합하도록 구현된 CMS-MX 소프트웨어 버전 관리 시스템을 소개하고 그 구성과 기능에 대해 기술하였다. CMS-MX 소프트웨어 버전 관리 시스템은 소프트웨어 형상과 소프트웨어 개발자 정보를 기초로 하여 파일의 등록 관리, 버전 제어, 그리고 정보 관리의 기능을 수행하며 파일 전체 저장, 파일 검증, 소프트웨어 종합 작업과의 연계 등의 특징을 가지고 있다.

ABSTRACT

CMS-MX(CDMA Mobile System-Mobile eXchange) software has a large-scale and needs frequent changes to adopt new functions. Therefore, an efficient version management is necessary for the software. General-purpose version management systems are not suitable for the CMS-MX software, which consists of various types of files and has its own development method. In this paper, we present a version management system for the CMS-MX software. The version management system provides file registration control, version control, and information management on the basis of software configuration and software developer information in accordance with CMS-MX software development process. It has characteristics such as full file storing, file verification, and association with software integration process.

*한국전자통신연구원 이동교환기기연구실
**한국전자통신연구원 초고속서비스연구실
論文番號:97148-0506
接受日字:1997年 5月 6日

Ⅰ. 서 론

오늘날의 교환 시스템은 하드웨어 기술의 발달과 시스템 환경의 변화로 인해 대용량(high capacity), 실시간 처리(real-time processing), 분산 처리(distributed processing), 및 고장 감내(fault tolerance) 등의 특징을 가지고 있다[1]. 그리고 이런 특징을 효율적으로 지원하기 위한 교환 소프트웨어도 점점 규모가 커지고 복잡한 구조를 이루고 있다. 따라서 대규모의 교환 소프트웨어를 개발하기 위해서는 많은 전문 인력과 더불어 체계적인 개발 방법론이 필요하다. 특히, 교환 소프트웨어는 새로운 사용자 요구사항의 수용 및 다양한 시스템 운용 현장에 따른 지속적인 변경이 필요하기 때문에 이를 효과적으로 관리할 수 있는 방법이 절실히 요구되고 있다[1, 2, 3].

복잡한 소프트웨어를 체계적으로 관리하는 방법으로 SCCS[8], RCS[9] 등의 범용 버전 제어 시스템(version control system)을 들 수 있다. 버전 제어 시스템은 파일의 등록(check-in), 편집(edit), 검색(check-out)의 세 단계에 기초하여 소프트웨어 개발자가 새로운 버전의 파일을 버전 제어 시스템에 등록하거나 이미 등록된 기존의 파일을 버전 제어 시스템으로부터 검색할 수 있는 수단을 제공한다[4, 5, 6, 7]. 대부분의 버전 제어 시스템은 디스크를 절약하기 위해서 각 파일에 대해 기초(baseline)가 되는 하나의 버전을 저장하며, 나머지 버전에 대해서는 기초 버전과의 차이인 델타(delta) 정보를 저장한다. 따라서 파일의 등록과 검색을 위해서는 델타 정보를 처리하는 과정이 필수적으로 요구된다.

이와 같이 기존의 버전 제어 시스템은 파일간의 텍스트(text) 변화에 기초하고 있기 때문에 소스 파일, 데이타베이스 파일, 이진 실행 파일 등 여러 가지 형태의 파일로 구성되어 있는 교환 소프트웨어에 이를 그대로 적용하기는 어렵다. 그리고, 교환 소프트웨어의 개발은 나름대로의 개발 체계 및 개발 환경에 따라 이루어지고 있기 때문에 소프트웨어 버전 관리는 전체적인 개발 과정과의 연계를 필요로 한다. 따라서 대부분의 교환 소프트웨어는 기존의 버전 제어 시스템을 그대로 사용하기 보다는 이들을 개발 환경 및 개발 체계에 적합하도록 수정 및 보완하고, 다른 개발 도구들과 서로 연계하여 사용하고 있다[10, 11]. 그러나 국내의 경우, 아직까지 교환 소프트웨어의 버전 관리에 관한 연구는 거의 발표되어 있지 않은 상황이다.

본 논문에서는 CMS-MX 소프트웨어의 특징 및 개발 환경에 적합하도록 구현된 버전 관리 시스템을 소개하고 그 구성 및 기능에 대해 기술하였다. CMS-MX 소프트웨어 버전 관리 시스템은 소프트웨어 형상 정보와 소프트웨어 개발자 정보에 기초하여 파일의 등록, 검색 및 접근 제어에 의한 파일 등록 관리 기능을 제공한다. 그리고, 기존의 버전 제어 시스템과는 달리 각각의 버전에 대해 파일 전체를 저장함으로써 UNIX 파일 시스템에서 제공하는 디렉토리 구조 및 파일 접근 제어 체계를 쉽게 적용할 수 있도록 하였으며, 소프트웨어 종합 작업과 쉽게 연계가 가능하도록 하였다. 또한 전체 소프트웨어 개발 일정에 따라 각각의 소프트웨어 버전을 그 중요도에 따라 개발용 호스트 시스템과 백업용 호스트 시스템에 분리하여 저장함으로써 시스템 자원을 효율적으로 이용할 수 있게 하였다.

본 논문의 2장에서는 CMS-MX 소프트웨어의 구성 및 개발 환경에 대해서 기술하고 3장에서는 CMS-MX 소프트웨어의 버전 관리 체계에 대해서 기술한다. 4장에서는 CMS-MX 소프트웨어 버전 관리 시스템의 구성 및 상세한 기능을 기술하고 5 장에서 결론을 맺는다.

Ⅱ. CMS-MX 소프트웨어 개요

2.1 소프트웨어의 구성

CMS-MX 소프트웨어는 다양한 종류의 파일들로 구성되어 있으며 각 파일은 교환기에 로딩될 목적 파일 생성을 위해 서로 다른 처리 과정을 필요로 한다. CMS-MX 소프트웨어는 호스트 시스템의 파일 시스템상에 그림 1과 같이 구성된다. 여기서 V_i는 CMS-MX 소프트웨어의 한 버전으로서 UNIX 파일 시스템상에 하나의 디렉토리로 저장된다. 한 버전내 각각의 파일은 그 종류에 따라 해당 서브 디렉토리에 저장된다.

각 디렉토리 이름이 파일의 종류를 나타낸다고 할 때, 소프트웨어 개발자가 직접 제작하는 소스 파일에는 db, dg, include, imd, libsrc, md, omd, src가 해당되며, 컴파일러나 기타 도구에 의해서 생성되는 파일에는 catalog, lib, mms, sig, slt가 해당된다. 교환기에 직

접 로딩될 파일에는 블록 실행 모듈, 운영 체제, 데이타베이스 파일, 입출력 메시지 양식 파일이 있으며 slt에 최종적으로 저장된다. doc은 CMS-MX 소프트웨어의 형상 및 개발자에 대한 정보를 저장하고 있다. 라이브러리 소스나 블록 소스 파일의 컴파일 시에 참조될 헤더(header) 파일은 catalog, include, lib, mms, sig가 해당된다. catalog 파일은 db로부터 생성되며, lib는 libsrc로부터, mms는 imd와 omd로부터, sig는 md로부터 각각 생성된다.

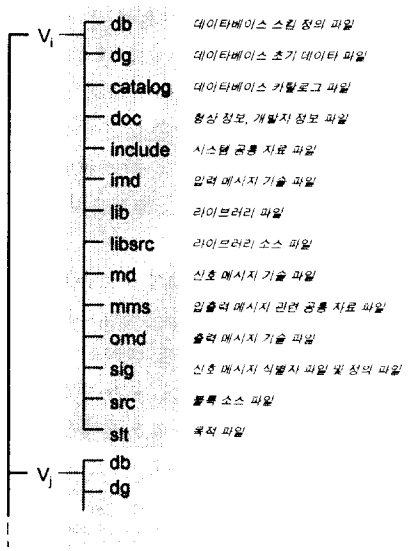

그림 1. CMS-MX 소프트웨어의 구성
Fig. 1 Configuration of CMS-MX software

2.2 소프트웨어 개발 환경

CMS-MX 소프트웨어는 저장을 위해 대용량의 디스크를 필요로 할 뿐만 아니라 많은 소프트웨어 개발자를 지원하기 위해서 고속의 처리 능력을 필요로 한다. 그리고 소프트웨어 개발자의 위치가 서로 분산되어 있더라도 소프트웨어 개발을 위한 호스트 시스템에 쉽게 접근할 수 있고 개발자 서로간에 정보 교환이 용이하도록 시스템 환경이 설정되어야 한다. 이와 같은 조건을 만족하기 위하여 그림 2와 같은 형태의 시스템 환경이 CMS-MX 소프트웨어의 개발에 사용되고 있다.

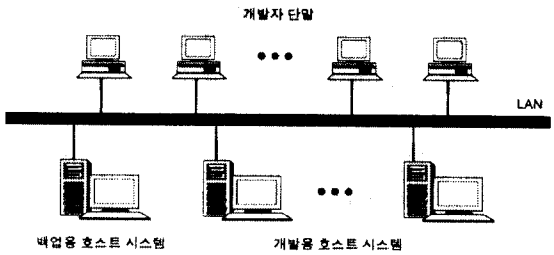

그림 2. CMS-MX 소프트웨어 개발을 위한 시스템 환경
Fig. 2 System environment for CMS-MX software development

CMS-MX 소프트웨어의 개발에는 네트워크 파일 시스템(Network File System: NFS)으로 구성된 여러 대의 UNIX 시스템이 사용되며, 이들 호스트 시스템에 저장된 데이타를 백업하기 위한 백업 시스템이 네트워크로 연결되어 있다. 소프트웨어 개발에 필요한 모든 환경은 시스템간에 공유된 디스크에 설정되어 있다. 따라서 소프트웨어 개발자는 여러 호스트 시스템중 어느 시스템으로 로그인하여도 동일한 환경에서 소프트웨어 개발 작업을 수행할 수 있다. 이로서 소프트웨어 개발과 관련된 시스템 부하를 여러 호스트 시스템에 분산시킬 수 있다.

각각의 소프트웨어 개발자는 LAN으로 연결된 단말을 통하여 UNIX 호스트 시스템에 접속할 수 있으므로 위치적으로 분산된 작업 환경에도 용이하다. 그리고 소프트웨어 개발과 관련된 문서 작업은 개발자 단말 또는 호스트 시스템상에서 이루어지며 네트워크를 통해 쉽게 공유할 수 있다. 소프트웨어 관리자가 소프트웨어 개발 일정과 관련하여 필요한 메시지를 전송할 때는 전자 우편을 사용함으로써 통신 부담을 크게 줄일 수 있다.

Ⅲ. CMS-MX 소프트웨어의 버전 관리 체계

3.1 소프트웨어 구현 단계

CMS-MX 소프트웨어에서 기능 추가는 그림 3과 같은 단계에 의해 이루어 진다. 여기서 F_i는 소프트웨어에서 새로 추가하고자 하는 기능의 집합을 나타낸다.

새로운 기능 F_i의 분석 및 설계 단계를 거친 후 기능 추가에 따른 새로운 파일의 생성 또는 변경이 소

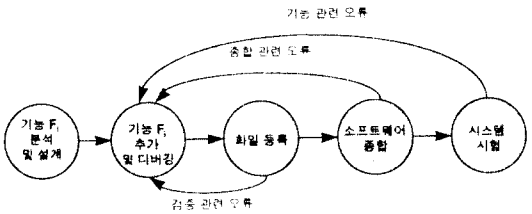

그림 3. CMS-MX 소프트웨어의 기능 구현 단계
Fig. 3 Function implementation cycle in CMS-MX software

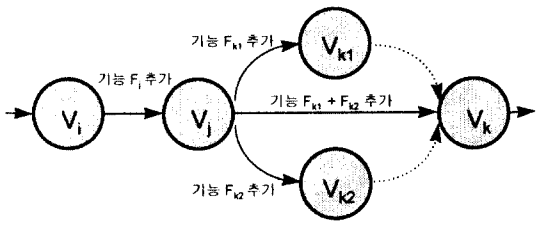

그림 4. CMS-MX 소프트웨어의 버전 진화
Fig. 4 Version evolution of CMS-MX software

프트웨어 개발자에 의해 이루어 지며 해당 파일은 소프트웨어 관리자에게로 등록된다. 파일의 등록이 완료되면 소프트웨어 종합 작업이 이루어 진다. 소프트웨어 종합 단계에서는 소스 파일로부터 필요한 헤더 파일을 생성하고 블록 소스 파일을 컴파일하여 교환기에 로딩될 목적 파일을 생성한다. 모든 목적 파일이 생성되면 교환기에 로딩하여 추가된 기능의 시험을 수행한다. 각 단계에서 오류가 발생하면 해당 소프트웨어 개발자에게 통보되어 재작업이 이루어 지도록 한다.

이와 같은 CMS-MX 소프트웨어의 개발 과정에서 버전 관리 작업은 크게 두 가지로 나눌 수 있다. 하나는 새로 생성되거나 변경된 파일의 등록 작업을 관리하는 것이고, 다른 하나는 소프트웨어 개발 계획에 따라 버전을 생성, 삭제 또는 백업하는 등 버전 단위의 작업을 수행하는 것이다.

3.2 버전 진화

버전은 CMS-MX 소프트웨어 관리의 기본 단위이다. 전체 소프트웨어의 공식적인 버전은 소프트웨어 관리자에 의해서 관리되어 지며 그림 4와 같은 진화 과정을 가진다. 즉, 버전 V_i에 새로운 기능 F_j의 추가가 필요할 때 V_i로부터 새로운 버전 V_j가 생성된다. 새로운 버전 V_j의 초기 상태는 V_i와 동일하나 새로운 기능 추가에 의한 파일 생성 및 수정에 의해 나중에는 V_i와 다른 상태가 된다. 기능 추가가 성공적으로 수행되면 그 버전은 고정 상태가 된다.

한 번에 여러 종류의 기능 추가가 필요할 때는 두 가지 방법의 버전 진화를 고려할 수 있다. 그림 4에서와 같이 버전 V_j로부터 새로운 기능 F_{k1}과 F_{k2}의 추가가 요구될 때, 첫번째 방법은 V_j에 F_{k1}과 F_{k2}를 모두 추가하여 새로운 버전 V_k를 만드는 것이고, 두번째 방법은 V_j에 F_{k1}을 추가하여 V_{k1}을 만들고, V_j에 F_{k2}를 추가하여 V_{k2}를 만든 후, 일련의 종합 과정을 거쳐 F_{k1}과 F_{k2}가 추가된 새로운 버전 V_k를 만드는 방법이다. 첫번째 방법은 디스크를 절약할 수 있고 버전 관리가 용이한 장점이 있으나 여러 기능을 동시에 구현하였기 때문에 기능 시험이 복잡하다. 반면에 두번째 방법은 여러 버전을 생성하기 때문에 버전 관리가 어렵고 디스크가 많이 소요된다. 그러나 개개의 기능을 쉽게 시험할 수 있는 장점이 있다.

기능 추가에 의한 새로운 버전의 생성 외에 소프트웨어 개발 과정에서 더 이상 필요성이 없는 버전이 시스템에서 삭제되기도 한다. CMS-MX 소프트웨어에서 대부분의 경우, 새로운 버전은 이전 버전에 새로운 기능을 추가한 것이기 때문에 새로운 버전에 대한 기능 시험이 성공적으로 이루어 지면 이전 버전은 그 중요성이 감소하게 된다. 따라서 이와 같이 오래된 버전은 압축되어 백업 시스템으로 이동되어 관리된다. 백업 시스템에서도 더 이상 불필요한 버전은 테이프에 저장하여 보관된 후 시스템에서 삭제된다.

3.3 버전 관리 작업의 개념 모델

CMS-MX 소프트웨어의 버전 관리를 위한 개념 모델은 그림 5와 같이 소프트웨어 개발을 위한 작업 영역, 소프트웨어 개발자, 소프트웨어 관리자, 그리고 소프트웨어 버전 관리 시스템으로 이루어진다.

소프트웨어 개발자와 관리자의 작업 영역은 호스트 시스템의 디스크 상에 위치하여 소프트웨어 개발 및 소프트웨어 버전 관리를 위한 환경을 제공한다. 각각의 작업 영역은 UNIX 사용자 계정에 대응되며 작업 영역간에 이루어지는 파일 복사, 접근 제어 등은 기본적으로 UNIX 파일 시스템에 기초한다. 따라서 소프트웨어 개발자는 기본적으로 자기 작업 영역

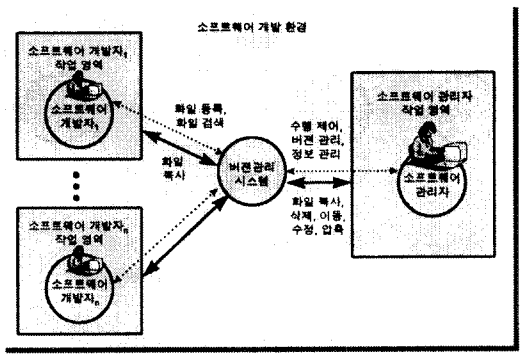

그림 5 CMS-MX 소프트웨어 버전 관리를 위한 개념 모델
Fig. 5 Conceptual model of CMS-MX software version management

외의 다른 개발자의 작업 영역에는 접근이 불가능하며, 소프트웨어 관리자의 작업 영역은 버전 관리 시스템을 통해서만 접근이 가능하다.

소프트웨어 개발자는 소프트웨어 개발 환경을 이용하여 자신에게 할당될 블록을 개발하면서 자체적으로 그 블록에 대한 버전 관리를 수행한다. 그리고 소프트웨어 개발 일정에 따라 정해진 기간 동안에 버전 관리 시스템을 통하여 새로 작성하거나 수정된 파일을 자신의 작업영역에서 소프트웨어 관리자의 작업 영역으로 복사한다. 또한 필요한 파일을 버전 관리 시스템을 통하여 소프트웨어 관리자 작업 영역에서 자신의 작업 영역으로 복사해 온다.

소프트웨어 관리자는 CMS-MX 소프트웨어 전체를 관리하는 역할을 담당한다. 소프트웨어 개발을 위한 환경의 설정 및 소프트웨어 개발 일정에 따라 새로운 버전을 생성하거나 오래된 버전을 삭제하는 등의 버전 관리 작업을 수행한다. 이를 위하여 파일 등록 관리에 필요한 버전 관리 시스템의 수행 인자를 제어하거나 버전 관리 명령을 버전 관리 시스템으로 전달한다. 또한 소프트웨어 형상 및 개발자 정보의 검색 또는 수정이 필요할 때 버전 관리 시스템을 통하여 이를 수행한다.

소프트웨어 버전 관리 시스템은 버전 관리와 관련된 여러 개의 도구(tool)로 구성되어 있으면서 CMS-MX 소프트웨어의 버전 관리 작업을 용이하게 하여 준다. 이것은 소프트웨어 개발자와 소프트웨어 관리자간의 인터페이스로 사용되며 소프트웨어의 버전에 대한 이력 및 로그 정보 등을 관리한다. 또한 소프트웨어 형상과 개발자 정보를 관리하며 소프트웨어 관리자가 이를 쉽게 검색 및 수정할 수 있는 기능을 제공한다.

Ⅳ. CMS-MX 소프트웨어 버전 관리 시스템

4.1 버전 관리 시스템의 구성

CMS-MX 소프트웨어 버전 관리 시스템은 그림 6과 같이 사용자 정합부, 파일 등록 관리부, 버전 관리부, 정보 관리부, 접근 제어부, 로그 처리부, 파일 처리부로 구성되어 있다.

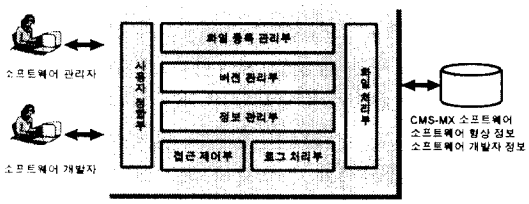

그림 6. CMS-MX 소프트웨어 버전 관리 시스템의 구성
Fig. 6 Configuration of CMS-MX software version management system

사용자 정합부는 소프트웨어 개발자와 소프트웨어 관리자로부터 입력된 명령을 분석하는 기능을 담당한다. 파일 등록 관리부에서는 소프트웨어 개발자로부터 파일을 등록 받거나 요청된 파일을 검색하여 주는 기능을 수행한다. 버전 관리부에서는 버전 진화의 각 단계를 지원하기 위해 버전 생성, 버전 삭제, 버전 백업 등의 기능을 수행한다. 정보 관리부에서는 소프트웨어 형상 정보 및 소프트웨어 개발자 정보를 관리 및 검색한다. 접근 제어부는 각각의 소프트웨어 개발자의 파일 등록 및 검색을 위한 접근 제어 기능을 담당하며 로그 처리부에서는 수행된 작업의 성공 여부 및 작업 시각 등의 로그 정보를 관리한다. 파일 처리부에서는 소프트웨어 관리자 영역 및 소프트웨어 개발자 영역의 파일을 접근하고 이를 복사하거나 이동하는 등의 기능을 수행한다.

4.2 버전 관리 시스템의 기능

CMS-MX 소프트웨어 버전 관리 시스템이 수행하

는 기능은 크게 파일 등록 관리, 버전 관리, 그리고 정보 관리로 나눌 수 있다. 이중에서 파일 등록 관리는 소프트웨어 개발자와 버전 관리 시스템간에 이루어지며 버전 관리와 정보 관리는 소프트웨어 관리자와 버전 관리 시스템간에 이루어 진다.

4.2.1 파일 등록 관리

화일 등록 관리는 소프트웨어 개발자로부터 파일을 새로 등록받거나 이미 등록된 파일을 검색해주는 작업으로서 그림 7과 같은 수행 절차를 가진다.

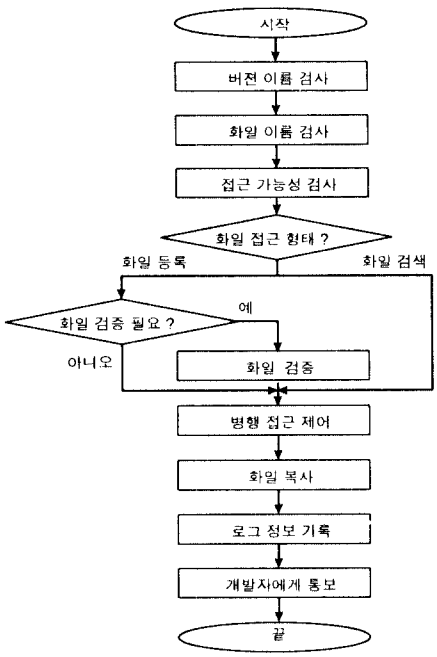

그림 7. 파일 등록 관리 수행 흐름도
Fig 7. Execution flow of file registration management

1) 접근 제어

버전 관리 시스템은 소프트웨어 개발자가 입력하는 파일 등록 명령으로부터 소프트웨어 버전 이름과 파일 이름을 분석하여 등록하고자 하는 파일이 어떤 버전의 어느 블록에 속하는가를 알아냄으로써 소프트웨어 개발자가 파일을 등록 또는 접근할 수 있는 권한이 있는가를 결정한다.

CMS-MX 소프트웨어의 버전 관리에는 세 가지의 접근 제어 방법이 사용된다. 첫째는 UNIX 파일 시스템에서 기본적으로 제공되는 파일 접근 제어 방법이다. 즉, 파일 단위로 그 파일의 소유자, 그룹, 다른 사람이라는 세 가지 종류의 접근자에 대해 읽기, 쓰기, 실행의 세가지 형태의 파일 접근에 대한 권한을 명시하고 이에 따라 접근을 제한하는 것이다. 이 방법은 다른 접근 제어 방법의 기초가 된다.

둘째는 소프트웨어 개발자 이름과 블록 이름에 기초한 접근 제어이다. 소프트웨어 개발자는 자신에게 할당된 블록에 대해서만 버전 관리 시스템을 통하여 접근을 할 수 있다. 블록은 CMS-MX 소프트웨어 구현의 기본 단위로서 블록 이름에 따라 관련된 데이타베이스 스킴 파일, 신호 메시지 기술 파일 등 소프트웨어 개발자가 작성하는 대부분의 소스 파일들이 결정된다. 버전 관리 시스템은 소프트웨어 형상 정보와 소프트웨어 개발자 정보를 관리하며 이를 이용하여 소프트웨어 개발자의 접근을 제어한다.

세째는 소프트웨어 개발 일정에 따른 접근 제어로서 여기에는 버전 이름, 등록 날짜 등에 의해서 접근 가능성이 결정된다. 소프트웨어 개발자는 일반적으로 현재 개발 진행중인 버전에 대해서만 파일을 등록할 수 있으며 등록 시간은 허가된 개발 일정의 범위 안에서 이루어 진다. 단, 파일 검색은 소프트웨어 버전 자체에 변화를 주지 않으므로 이전 버전에 대해서도 수행이 가능한다.

2) 파일 검증

소프트웨어의 등록은 특정 기능의 시험을 위한 소프트웨어 종합 작업의 선행 단계라고 할 수 있다. 따라서 소프트웨어에 내재된 오류를 등록 시에 미리 찾아내어 정정할 수 있다면 소프트웨어 종합 및 시험에 소요되는 시간을 크게 단축할 수 있다. 소프트웨어 개발자가 파일을 등록하기 전에 검증 작업을 미리 수행할 수도 있다. 그러나 어떤 오류는 하나의 파일만을 검증해서는 발견될 수 없고 관련된 모든 파일에 대한 연계 검증을 수행해야만 가능하기 때문에 버전 관리 시스템에 의한 검증이 필요하다.

소프트웨어 개발자가 등록하는 파일 중에서 신호 메시지 기술 파일, 입출력 메시지 기술 파일, 데이타베이스 스킴 파일 등은 소프트웨어 종합 과정에서 일련의 처리 과정을 거치며 이들로부터 블록 소스의 컴파일에 필요한 헤더 파일이 생성된다. 따라서 이들

파일에 존재하는 오류를 미리 발견하여 정정하지 않는다면 블록 소스의 컴파일 단계까지 오류가 확산되어 교환기 시스템 전체의 동작에 영향을 주게 된다. 특히 신호 메시지 정의 파일은 블록간에 교환되는 신호 메시지를 정의하고 있기 때문에 여러 블록에 영향을 줄 수 있다.

버전 관리 시스템은 소프트웨어 개발자로부터 검증이 필요한 파일에 대한 등록 요청이 있을 때마다 그 파일의 형태에 대응되는 검증 도구를 호출하여 검증을 수행하게 한다. 오류가 없다면 파일의 등록이 성공적으로 수행되고, 오류가 있다면 검증 도구에 의해서 출력된 오류 메시지를 해당 개발자에게로 통보한다. 개발자는 통보 받은 오류 메시지를 참고하여 파일의 오류를 정정한 후 재등록 작업을 수행한다.

3) 병행 접근 제어

병행 접근은 특정 파일을 둘 이상의 사람이 동시에 접근하고자 할 때 발생한다. 접근하고자 하는 사람중 한 사람만이 읽기/쓰기의 권한을 가지고 있다면 아무런 문제가 없으나 둘 이상이 읽기/쓰기의 권한을 가지고 있다면 이들간의 접근을 제어할 필요성이 있다. 즉 둘 이상의 사람이 한 파일을 동시에 수정하고자 할 때 이들의 작업이 순서적으로 일어날 수 있도록 하여 두 사람에 의한 수정이 모두 반영되도록 해야 한다.

CMS-MX 소프트웨어에서 대부분의 파일은 이를 접근할 수 있는 사람이 한 사람으로 지정되어 있기 때문에 병행 접근 제어가 필요없다. 그러나 시스템 헤더 파일은 둘 이상의 블록에서 사용되는 자료의 형태를 정의한 파일로서 모든 소프트웨어 개발자에 의해서 동시에 접근되어 수정될 수 있으므로 병행 접근 제어가 필요하다. 시스템 헤더 파일을 수정하고 등록할 수 있는 사람을 한 사람으로 한정하여 여러 사람으로부터의 수정 사항을 취합하여 이를 파일에 반영한다면 병행 접근 제어가 필요하지 않다. 그러나 이 방법은 소프트웨어 개발자간의 통신 부담을 크게 하며 시스템 헤더 파일의 즉각적인 수정이 어렵다는 단점이 있다.

버전 관리 시스템에서는 시스템 헤더 파일에 대한 병행 접근을 제어하기 위해서 이런 파일의 등록은 항상 검색 후에 이루어 지게 한다. 즉, 어떤 소프트웨어 개발자가 시스템 헤더 파일을 수정하기 위해서는 먼저 그 파일을 검색하여 자신의 작업 영역으로 가져온 후 이를 수정하고 다시 등록하는 과정을 거쳐야 한다. 파일을 수정하여 등록하기 전에 여러 번의 검색 과정을 거쳐도 상관없다. 그림 8은 버전 관리 시스템에서의 병행 접근 제어 흐름을 나타낸다.

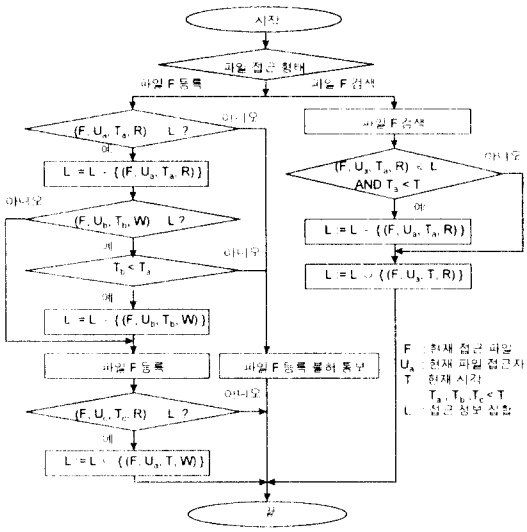

그림 8. 병행 접근 제어 흐름도
Fig. 8 Control flow of concurrent access

버전 관리 시스템은 파일을 검색하는 동작에 대해 파일 이름, 파일 접근자 이름, 검색 시각 정보를 저장한다. 파일의 등록 동작이 있을 경우, 그 등록자가 파일을 검색한 시각과 현재 등록하고자 하는 시각 사이에 다른 사람에 의해서 그 파일에 대한 등록이 이루어 졌다면 그 파일에 대한 등록 요청은 거부되며, 그렇지 않은 경우 그 파일을 등록하고 그 파일의 등록 정보를 기록한다. 파일의 등록이 거부된 사용자는 그 파일을 다시 한번 검색한 후 수정하여 등록하는 과정을 수행한다.

4) 작업 정보 로깅

버전 관리 시스템은 특정 파일에 대한 접근이 있을 때마다 그 정보를 로그 파일에 기록한다. 로그 정보는 소프트웨어 개발 과정에 따른 그 파일의 이력 정보로서 유용하게 사용될 수 있다. 또한 그 파일이 어

느 버전에서 어떤 기능과 관련해서 변경되었는지를 알 수 있으므로 디버깅하는 데에도 중요한 역할을 한다. 버전 관리 시스템은 등록되는 파일에 대해서 파일 이름, 접근 일자 및 시각, 등록자 이름, 등록 성공 여부 등의 정보를 기록하며 특정 파일을 검색만 할 때도 이와 같은 로그 정보가 생성되어 기록된다.

4.2.2 버전 관리

CMS-MX 소프트웨어의 버전 관리에서는 각각의 버전에 대해 파일 전체가 저장되도록 하는 방법을 사용하고 있다. 따라서 각 버전에 대해 델타 정보만을 저장하는 방법과는 달리 파일의 등록, 검색 또는 소프트웨어 종합을 수행하기 전에 원래의 파일을 복구하기 위한 전처리 과정이 필요 없다. 그러나 하나의 버전은 UNIX 파일 시스템상에서 하나의 디렉토리로 저장되기 때문에 버전 진화를 위해서는 하나의 버전 전체에 대해 수행되어 지는 버전 단위의 작업이 필요하다.

버전 단위의 작업은 파일의 등록, 검색과는 달리 소프트웨어 관리자와 버전 관리 시스템간에 이루어지며 소프트웨어 개발 일정 및 호스트 시스템의 디스크 사용 상태에 따라 수행되어 진다. 개발용 호스트 시스템과 백업용 호스트 시스템을 이용한 CMS-MX 소프트웨어의 버전 관리 흐름은 그림 9와 같다. 하나의 버전은 생성, 압축, 이동 및 삭제의 과정을 거친다.

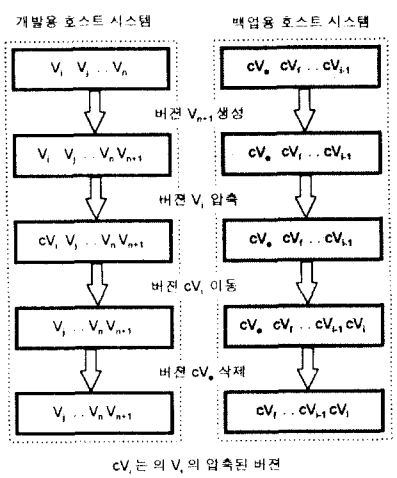

그림 9. CMS-MX 소프트웨어 버전 관리 흐름

Fig. 9 Flow of CMS-MX software version management

버전 삭제는 하나의 버전에 속하는 파일을 UNIX 파일 시스템상에서 제거하며 삭제되는 파일의 범위에 따라 세 가지로 나눌 수 있다. 첫째는 하나의 버전 전체를 삭제하는 것이며, 둘째는 블록 소스의 컴파일에 사용되는 헤더 파일을 제외한 나머지 파일을 삭제하는 것이다. 그리고 세째는 한 버전으로부터 임시화일 등의 불필요한 파일만을 삭제하는 것이다. 버전 삭제는 디스크를 절약하기 위해서 주로 이루어 지며 대개 더 이상 필요로 하지 않는 오래된 버전을 대상으로 한다.

버전 압축은 디스크를 저장할 목적 또는 버전을 백업할 목적으로 사용된다. 따라서 버전 압축이 이루어지기 전에 그 버전으로부터 불필요한 임시 파일을 제거하는 작업이 주로 이루어진다. 압축에는 UNIX에서 제공되는 compress 와 tar 명령이 사용된다. 이렇게 함으로써 하나의 버전을 저장하는 디렉토리는 하나의 압축된 파일로 저장이 가능하다. tar 형태의 파일은 디렉토리의 구조를 그대로 저장하기 때문에 압축 해제하여 원래의 상태로 복원하기가 쉬우며 압축된 상태에서 원하는 특정 파일만을 쉽게 추출하여 검색할 수 있다.

버전 이동은 현재 소프트웨어 개발이 이루어지고 있는 시스템에 저장된 버전을 네트워크로 연결된 다른 시스템으로 이동하는 것으로서 주로 백업 목적으로 사용된다. 따라서 버전 이동이 이루어지기 전에 버전 압축 작업이 먼저 수행된다. 소프트웨어를 개발용 호스트 시스템과 백업용 호스트 시스템에 분리하여 저장함으로써 얻을 수 있는 이점으로는 시스템 고장에 따른 위험 부담을 줄일 수 있고, 중요도가 작은 버전을 백업 시스템으로 이동함으로써 디스크를 저장할 수 있을 뿐만 아니라, 백업된 버전에 대해서는 더 이상의 변경이 어렵게 함으로써 소프트웨어의 이력 관리가 용이한 점을 들 수 있다.

하나의 파일에 대한 등록이 있을 때마다 로그 정보가 생성되어 저장되는 것처럼 하나의 버전 관리 작업에 대해서도 로그 정보가 생성된다. 즉, 각각의 버전 관리 작업에 대해서 작업 이름, 작업 시작 및 완료 시각, 성공 여부 등을 기록한다. 소프트웨어 관리자는 이와 같은 버전 관리 작업의 로그 정보로부터 특정 버전에 대한 이력 정보를 검색할 수 있다.

4.2.3 정보 관리

버전 관리 시스템은 CMS-MX 소프트웨어 형상 정보 및 소프트웨어 개발자에 관한 정보를 관리하고 있다. CMS-MX 형상 정보에는 서브 시스템 정보, 프로세서 정보, 블록 정보 등이 있다. 이중에서 블록 정보는 CMS-MX 소프트웨어 버전 관리에서 가장 중요한 역할을 담당한다. 블록 정보에는 각각의 블록에 대해 블록 이름, 개발자 이름, 로딩될 프로세서 이름, 실행 모듈 이름 등이 포함된다. 개발자 정보에는 개발자 이름, 소속, 전화 번호, 전자 우편 주소 등이 포함된다.

버전 관리 시스템은 블록 형상 정보 및 개발자 정보를 이용하여 파일의 등록 및 검색을 관리한다. CMS-MX 소프트웨어는 새로운 기능의 추가로 인한 이들 정보의 변경이 자주 요구되므로 이를 체계적으로 관리하여야 한다. 블록 및 개발자 정보 관리를 위해 버전 관리 시스템이 버전 관리자에게 제공하는 명령으로는 블록 정보 추가, 블록 정보 변경, 블록 정보 삭제, 블록 정보 검색, 개발자 정보 추가, 개발자 정보 변경, 개발자 정보 삭제 등이 있다. 이중에서 블록 정보 추가에서는 블록의 등록 관리에 필요한 모든 정보가 새롭게 생성되며, 블록 삭제에서는 파일간의 일치성을 보장하기 위하여 블록 등록과 관련된 모든 정보 및 그 블록과 관련된 모든 파일이 함께 삭제된다.

그 외, 버전 관리 시스템에서는 등록 허가 정보, 디렉토리 정보 등을 관리함으로써 특정 형태의 파일 또는 블록에 대한 파일 등록을 허가하거나 불허하며, 등록된 파일이 특정 디렉토리에 저장될 수 있도록 디렉토리를 지정할 수 있다. 또한 특정 블록에 대해 개발자 외의 사람이 등록 또는 접근할 수 있는 접근 관리 정보를 관리한다.

V. 결 론

CMS-MX 소프트웨어는 복잡한 연관성을 가진 여러 종류의 파일로 구성되어 있으며 새로운 기능 추가로 인한 지속적인 변경을 필요로 한다. CMS-MX 버전 관리 시스템은 이와 같은 소프트웨어의 특성 및 소프트웨어 개발 환경에 적합하도록 구현된 시스템으로서 소프트웨어의 등록 및 검색 관리, 백업 시스템과 연계한 버전 관리, 그리고 블록 형상 및 개발자 정보 관리 등의 기능을 수행한다.

CMS-MX 소프트웨어 버전 관리 시스템에서는 각 소프트웨어 버전을 UNIX 파일 시스템상에 하나의 디렉토리로 저장하며, 파일 전체를 저장하는 방법을 사용하기 때문에 UNIX 시스템에서 제공하는 파일 접근 제어 및 파일 처리 기능을 쉽게 적용할 수 있다. 또한 파일 검증에 의해 파일 등록의 다음 단계인 소프트웨어 종합 작업과도 쉽게 연계가 가능하며, 백업 시스템에 의한 효율적인 디스크 및 버전 관리를 수행한다.

앞으로의 연구 방향은 네트워크 파일 시스템으로 구성된 시스템에서뿐 만 아니라 일반적인 네트워크로 연결된 다른 시스템에서도 쉽게 파일의 등록 및 검색이 용이하도록 하는 클라이언트/서버 형태의 버전 관리 시스템에 관한 것이다.

참 고 문 헌

1. 김대식, "교환 소프트웨어 특징과 과제", 텔레콤, 제 10권 1호, 1994.
2. 김대식, 김성희, 안지환, 이충근, "이동통신교환기 소프트웨어", 한국통신학회지, 제 11권 3호, pp. 34-49, 1994.
3. T. Murakami, S. Sato, H. Sakamoto, and R. Moriya, "Information Management of Switching Software Development Based on Distributed Development Environment", Proceedings of BLOBECOM'93, pp. 357-361, 1993.
4. R. S. Pressman, Software Engineering-A Practitioners Approach, McGraw-Hill International. 1992.
5. W. F. Tichy, "Tools for Software Configuration Management", Proceedings of the International Workshop on Software Version and Configuration Control, pp. 1-20, Jan. 1989.
6. E. H. Bersoff, V. D. Henderson, and S. G. Siegel, "Software Configuration Management: A Tutorial", IEEE Computer, vol. 12, no. 1, pp. 6-14, Jan. 1979.
7. T. Parker, "Software Configuration Management Tools", UNIX Reviews, pp. 91-98, Oct. 1995.
8. M. J. Rochkind, "The Source Code Control System", IEEE Transactions on Software Engineering,

vol. 1, no. 4, pp. 364-370, 1975.

9. W. F. Tichy, "RCS-A System for Version Control", Software-Practice and Experience, vol. 15, no. 7, pp. 637-654, July 1985.
10. W. F. Tichy, "Programming-in-the-Large: Past, Present, and Future", Proceedings of the International Conference on Software Engineering, pp. 362-367, 1992.
11. A. A. Pal, and M. B. Thompson, "An Advanced Interface to a Switching Software Version Mangement System", Proceedings of the International Conference on SETSS, pp. 110-113, 1989.

신 재 욱(Jaewook Shin) 정회원

1992년 2월 : 경북대학교 컴퓨터과학과 졸업(이학사)
1994년 2월 : 경북대학교 대학원 컴퓨터과학과 졸업(이학석사)
1994년 3월~현재 : 한국전자통신연구원 이동교환기기 연구실 연구원

박 광 로(Kwangroh Park) 정회원

1982년 2월 : 경북대학교 전자공학과 졸업(공학사)
1985년 2월 : 경북대학교 대학원 전자공학과 졸업(공학석사)
1984년 3월~현재 : 한국전자통신연구원 초고속서비스 연구실 선임연구원

이 남 준(Namjun Lee) 정회원

1978년 2월 : 서강대학교 전자공학과 졸업(공학사)
1983년~현재 : 한국전자통신연구원 이동교환기기연구실 선임연구원