버전제어를 위한 소프트웨어 구성요소의 검색 시스템

오 상 엽[†] · 김 홍 진[††] · 장 덕 철[†††]

요 약

소프트웨어 재사용과 형상 관리, 그리고 버전 제어를 위해서는 소프트웨어 구성요소를 검색할 수 있는 검색 시스템과 라이브러리의 구성이 중요한 문제로 제기된다.

검색 시스템은 많은 구성요소(component)를 저장하고, 빠른 시간 안에 키워드를 이용하여 원하는 구성요소를 검색할 수 있어야 한다. 기존의 검색 방법은 대부분 키워드나 내용을 기반으로한 역 파일 개념 등이 사용되고 있다. 본 논문에서는 객체지향 프로그래밍 언어인 Smalltalk에서 Set와 Bag 클래스를 이용하여 키워드를 사용하면서도 내용을 기반으로 구성요소를 찾는 검색 시스템을 제안한다. 이 방법은 사용자 인터페이스와 이를 관리하기 위한 기능을 향상시킨다. 또한, 검색시스템과 함께 라이브러리를 제안하고, 이를 관리하고 제어하기 위한 사용자 인터페이스를 설계·구현하였다.

본 논문의 검색 시스템은 다른 언어와의 인터페이스를 통해 사용될 수 있으며, 이 시스템은 버전 제어를 위한 검색 시스템과 라이브러리를 제공한다.

Software Component Retrieval System for Version Control

Sang Hyeub Oh[†] · Hong Jin Kim[††] · Deog Chul Jang[†††]

ABSTRACT

For the reuse, configuration management, and version control of softwares, the composition of retrieval systems and library are most important matters, which makes it possible to retrieve the concerned software components.

Retrieval systems, which is able to store many components, must make it possible to retrieve the concerned components with keywords in the fastest way. Based either on keywords or the concept of inverted file on the part of content is usually used in the current retrieval systems. However, in this paper, new retrieval systems are suggested with using set and bag class with Smalltalk language, one of object-oriented programming language, based either on the keywords or on the part of content to find out the concerned components. This method is improved the function of user interface and its management. In this paper, library is also suggested along with the new retrieval systems, and user interface is designed and implemented for its management and control.

The new retrival systems of this paper can be employed by interface in another language, and this system is to provide the concerned user with the appropriate retrieval systems and library for the version control.

1. 서 론

오늘날의 컴퓨터 산업 분야에서 소프트웨어는 그 규모와 중요성이 증가하고 있으며, 이에 비해 소프트웨어 개발 및 유지 보수 기술은 아직 그 수요를 충족시키지 못하고 있어 소프트웨어 위기를 맞고 있다.

이러한 소프트웨어의 개발 및 유지 보수에 도움을 주기 위해 새로운 방법론 및 자동화 도구들이 계속해서 개발되고 있는데, 그 중 하나가 버전 제어 시스템이다. 버전 제어 시스템은 오류 수정, 사용자 요구 사

† 정 회 원:경원전문대학 교양과 조교수
†† 종신회원:경원전문대학 전자계산과 부교수
††† 정 회 원:광운대학교 전자계산학과 교수
논문접수:1995년 8월 30일, 심사완료:1996년 3월 15일

항의 변경, 소프트웨어 기능 향상 등의 이유로 시간이 지남에 따라 변화하는 소프트웨어 구성요소 및 소프트웨어 형상을 체계적으로 관리하기 위한 시스템을 말한다[15].

버전 제어를 위해서는 라이브러리에 데이터를 어떻게 구성하여 저장, 관리할 것인가의 문제가 제시되며, 이를 위해 효율적인 검색 시스템이 제공되어져야 한다[5, 9, 14, 15].

검색 시스템은 사용자로 하여금 관심있는 영역만을 탐색할 수 있고, 구성요소에 대한 정보를 제공하여야 한다. 버전 제어를 위한 구성요소를 식별하는 것은 버전 제어에서 기본이 되는 기능이다. 구성요소들의 집합이 크고, 그 구성요소들이 여러 응용 분야에 걸쳐 널리 사용되면서 우수한 모듈성을 갖는 환경에서는 필요한 구성요소를 검색하고 식별하는 것이 중요한 문제로 제기된다[3, 10, 13, 17]. 기존의 검색 방법은 대부분 키워드나 내용을 기반으로한 역 파일 개념 등이 사용되고 있다. 본 논문에서는 객체지향 프로그래밍 언어(OOP, Objcet Oriented Programming)인 Smalltalk 언어를 이용하여 키워드를 사용하면서도 내용을 기반으로 구성요소를 찾는 검색 시스템을 제안한다. 또한, 이 시스템은 중복되는 파일의 이름을 Smalltak에서의 Bag 클래스를 수정 보완하여 처리하였다.

라이브러리는 구성요소의 접근, 탐색, 제어, 보안에 용이한 포괄적인 DBMS이며, 사용자가 구성요소를 생성, 편집, 검증, 합성하기 위한 기능을 제공하며, 확장이 용이해야 한다[10].

이를 위한 방법으로서 구성요소는 특정 기능을 수행하는 단위로서 개발되며, 논리적으로 서로 독립된 기능을 수행하거나 분리 번역되었다가 후에 연결되어 사용되는 프로그램의 일부인 모듈을 구성요소로 한다. 해당 구성요소를 라이브러리에 저장할 때는 Facet 분류 방식을 사용한다.

본 논문에서는 검색 시스템을 개발하고, 검색 시스템의 지원으로 라이브러리를 관리하며, 이를 사용할 수 있는 사용자 인터페이스를 설계 및 구현한다. 사용자 인터페이스는 질의어와 사용자 메뉴, 마우스를 사용하며, Smalltalk로 구현한다.

본 논문의 제 2장에서는 라이브러리, 검색 시스템, 버전 제어에 대한 관련 연구를, 제 3장에서는 라이브러리와 검색 시스템의 설계와 방법론, 제 4장에서는 구현 방법과 실행 예를 다룬다.

2. 관련 연구

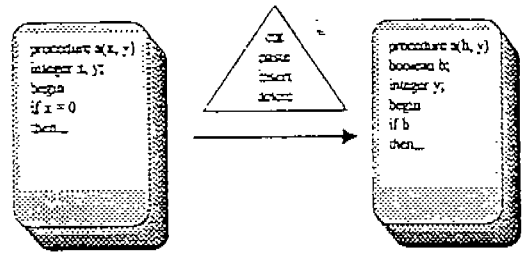
2.1 버전 제어

프로젝트 개발과 유지보수되는 동안 구성요소는 변경되며, 이러한 변경(changes)의 집합 즉, 대개 브라우저에 의해서 수행되는 텍스트상에 있는 오퍼레이션인 cut, paste, insert, delete 등의 작업으로 구성요소의 한 버전들을(연속적인) 다음 버전으로 변환시키는 이력 과정(history step)을 버전 제어라 한다.

하나의 구성요소는 시간이 지나고 프로젝트가 계속되는 동안 진화(evolve)되는데, 이 결과의 구성요소들을 version, revision, variant, 또는 derivation이라고 한다[2, 12, 15].

일반적인 소프트웨어 구성요소의 버전도 수정 버전과 변형 버전으로 나눌 수 있다. 수정 버전은 형상의 수정과 비슷한 개념이나 변형 버전은 이와는 달리 두 가지 정의가 가능하다. 그 하나는 소프트웨어 객체의 추상(abstraction)에 있어서 서로 구별할 수 없는 두 객체 사이에 존재하는 것과 이전의 버전(old version)에 대한 변경이 있을때 생성되는 버전을 말한다. 실제에 있어서 변형 버전은 이 두가지 개념을 모두 만족할 수도 있고, 하나만 만족할 수도 있다.

(그림 1)은 한 버전을 새로운 버전으로 변환하는 이력 과정을 나타낸다.



(그림 1) 이력 과정
(Fig. 1) History Step

변환하는 오퍼레이션들은 많은 시스템에서 이력 과정을 위한 delta의 형태를 제안하기 위해서 삼각형

〈표 1〉 버전제어 시스템의 비교 분석
〈Table 1〉 Compare and analysis of version control system

종류 / 구분	SCCS	RCS	DSEE	CCC	NSE
적용 대상	source code	source code	Ada source code, object code	Textual	Textual
동시성 관리	check-in check-out	check-in check-out	check-in check-out	check-in check-out	Copy-Modfy-Mergy
이력 관리	internal annotation	log message	modification request	log message	
버전 유형	revision variation	revision variation	revision variation	revision	object management system
버전 트리 디스플레이	제공안함	제공안함	제공함	제공안함	제공안함

박스 내에 들어있다.

버전 제어를 지원하는 주요한 도구들의 특성을 비교 분석하여 표 1에 나타내었다[18].

SCCS[9], Make, RCS[4] 등의 버전 제어 시스템들은 프로그램의 변경 이력으로 버전을 유지하고, 필요시 기존의 버전을 이용하여 새로운 프로그램 작성을 돕기 위한 도구로 개발되었다.

버전 제어를 위해서는 기본적으로 표준화된 소프트웨어 구성요소의 작성과 이들을 라이브러리에 체계적으로 저장하여 검색이 용이하도록 하여야 하며, 각 구성요소들의 분류가 체계적으로 이루어져야 한다[5]. 이것은 버전 제어를 위한 기반이 된다. 또한, 필요한 소프트웨어 구성요소를 명세하여 검색하고 다른 소프트웨어 구성요소들과 결합하여 새로운 소프트웨어를 작성하여 소프트웨어의 이력을 효율적으로 관리해야 한다[18].

본 논문은 이 중에서 버전 제어를 위한 검색 시스템을 제안하고, 이를 사용하기 위한 라이브러리와 사용자 인터페이스를 제공한다. 여기에서 사용하는 소프트웨어 구성요소와 새로 생성되는 소프트웨어는 모두 원시 코드이면서 프로그램의 한 단위 기능을 가진 모듈이다.

2.2 라이브러리

라이브러리는 구성요소의 접근, 탐색, 제어, 보안에 용이한 포괄적인 DBMS이며, 사용자가 구성요소를 생성, 편집, 검증, 합성하기 위한 기능을 제공하며, 확장이 용이해야 한다.

유사한 구성요소의 선택은 분류 문제이며, 유사성의 정도는 어떻게 집합(collection)을 조직하느냐에 따라 정해진다. 그러므로 집합의 조직과 선택은 이 모델에 있어서 중요하며, 분류 구조는 버전 제어를 위한 라이브러리를 위해서 중요하다.

분류는 한정된 영역안에서 비슷한 성질의 구성요소들을 하나의 집단(group)으로 묶고, 그 집단과 집단들 사이의 관련성을 표현하는 과정으로 정의한다[3].

소프트웨어 구성요소의 관련성을 부여하는 방법에 따라 Enumerative와 Facet의 2가지 방법이 있다[10].

Enumerative 분류는 구성요소들의 집합을 좁은 클래스로 분할해가면서 그들 사이의 계층성을 표현한다.

Facet 분류에서는 대상들의 공통적인 측면의 값들을 모아 facet를 구성한 뒤 이러한 Facet 들에서 해당 소프트웨어에 알맞는 구성요소들을 선택하여 합성함으로써 특정 프로그램을 구성한다. Facet방법은 공통적 성질의 객체들을 모아 Facet을 구성하여 기억시켜 놓은 재사용 라이브러리에서 알맞는 요소의 소프트웨어 구성요소를 선택하여 결합할 수 있도록 한다. 이 방법은 새로운 객체를 추가하기 쉽고 라이브러리 확장에 쉽게 적용되지만 객체들의 계층적 관련성 표현이 어려운 단점이 있다.

또한, 라이브러리는 구성요소와 연계된 정보를 관리하고, 구성요소의 등록과 삭제, 구성요소에 필요한 사항의 수집과 관리 등에 관한 사항도 포함해야 한다.

구성요소 라이브러리의 예로는 Raytheon Company에서 3200개의 코볼 코드 구성요소로 라이브러리를 구성하였으며, AT & T Pacific Bell사에서 250명의 개

발자에게 공유된 약 1000개의 C언어 구성요소를 가지고 라이브러리를 구성하였고, 이외에도 prieto Diaz의 라이브러리, EIFFEL, LaSSIE, AIRS, ROSE사 등에서 구축한 라이브러리가 있다.

2.3 검색 시스템

검색 시스템은 라이브러리에서 소프트웨어의 구성요소를 카탈로깅(cataloging)하고 검색(retrieving)하기 위해 사용된다.

검색 시스템은 버전 제어에 필요한 구성요소를 식별하여 검색하며, 라이브러리에 구성요소를 등록시키고 새로운 구성요소를 작성할 수 있어야 한다.

검색 시스템은 다양한 사용자 사이에서 같은 의미를 가지는 용어(terminology) 차이를 해결하고자 할때 라이브러리에서 각 텍스트의 유사 단어를 검색하고, 저장된 정보와 사용자의 질의어 사이의 용어 차이 또는 모호성(ambiguitious)를 제거하는데 도움을 주기위하여 검색 과정 동안 사용된다.

라이브러리는 관련된 동의어(synonyms)를 가진 단어나 구를 디렉토리(directory)에 저장하며, 이 구조에 저장 되어있는 기본 단어(primary word)는 모든 메뉴 키워드와 전반적인 조직에 대해 중요하거나 공통적인 것으로 간주되는 비키워드를 포함한다.

검색 시스템을 위한 다른 고려 사항은 처리되는 데이타의 형태(type)이다. 검색 시스템에 의해 처리되는 정보는 사실상 원문(textual)이며, 정보 검색(IR, Information Retrieval) 시스템은 텍스트 정보의 처리를 위주로 하며, 대부분의 다른 시스템은 실제의 데이타를 다룬다. 이러한 고려하에 IR 시스템이 검색 시스템을 위해 선택된다. 일반적인 기억 구조와 인덱싱, IR 시스템의 질의어 과정도 고려되어야 한다[14, 16].

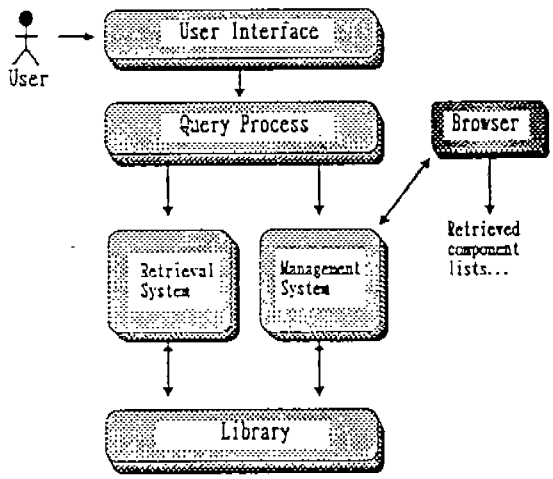
라이브러리 항목(entries)은 기본 단어와 유사어의 목록(list)도 찾을 수 있어야 하며, 완전한 소프트웨어 구성요소는 IR 시스템에서 하나의 텍스트 문서로서 저장된다. 검색 시스템의 일반 사용자에 의해 수행되는 네가지 기본적인 기능은 다음 항목이 지원 되어야 한다.

- 새로운 구성요소의 추가
- 현 구성요소의 수정
- 현 구성요소의 삭제
- 현 구성요소의 검색

새로운 구성요소가 검색 시스템에 입력될 때, 사용자는 소프트웨어 구성요소와 연관된 정보를 명세(specifies)하여야 하며, 라이브러리에 대한 관리도 이루어 져야 한다. 소프트웨어 구성요소를 위한 검색 능력은 검색 시스템의 핵심이며, 검색된 소프트웨어의 카탈로그는 저장된 정보에 엑세스되지 못하면 소용이 없다. 사용자는 적절한 사용자 메뉴와 질의어를 통해 모든 정보를 처리할 수 있어야 하며, 라이브러리 관리도 지원되어야 한다.

3. 검색 시스템의 설계

버전 제어에 필요한 소프트웨어 구성요소를 잘 식별하기 위해서는 이들을 저장·관리하고, 유지하며, 필요한 구성요소를 찾아내는 검색 메카니즘의 성능이 좋아야 한다. 버전 제어를 위한 구성요소의 검색시에는 해당 구성요소에 관한 정보 뿐만아니라 다른 구성요소와의 관계를 표현하여 검색 능력을 향상시켜야 한다[13]. 그러므로 구성요소를 관리하기 위한 시스템의 주기능으로 구성요소를 등록시킬 수 있는 라이브러리와 이 라이브러리에서 저장된 구성요소를 검색



(그림 2) 시스템 모델의 구성도
(Fig. 2) Structure of the system model

하고, 질의어를 이용하여 새로운 구성요소를 작성하기 위한 검색 시스템이 필요하다. 이를 위해 화면에서 제시하여 주는 도움말과 질의어에 따라 원하는 처리를 할 수 있도록 사용자 인터페이스를 설계한다.

3.1 시스템 모델

본 논문에서 개발한 시스템 모델은 검색 시스템과 이의 지원을 받는 라이브러리, 구성요소를 관리하기 위한 관리 시스템, 브라우저, 사용자 질의어 및 인터페이스로 구성되며, 구성도는 다음과 같으며, 각 구성은 다음 절에서 설명한다.

3.2 시스템 모델의 기능

1) 검색 시스템

구성요소 검색 시스템은 시스템 모델의 전반적인 기능 중 가장 중요한 기능으로써, 버전 관리를 위한 중요한 기능을 제공한다. 검색 시스템은 구성요소의 라이브러리를 초기화 하여 주는 기능과 라이브러리에 구성요소의 추가, 삭제 및 검색할 수 있는 기능을 지원하도록 한다. 이 기능에 의해서 처리되는 구성요소는 실제의 원문(text)이 된다. 그러므로 데이타의 중복을 최소화하여 실제의 데이타를 저장한 데이타베이스의 성격보다도 질의어를 이용하여 원문속의 핵심 단어를 가지고 원하는 구성요소를 검색하는 정보검색 시스템의 성격을 가진다.

구성요소의 검색 기능을 위하여 구성요소 자체 내에 구성요소에 대한 정보 또는 버전 제어 과정에 도움이 될 수 있는 주석을 포함한다. 즉, 구성요소의 소재, 구성요소의 복잡도, 버전 제어에 필요한 절차나 주의사항 그리고 관련 문서 등의 정보를 구성요소 내에 작성한다. 구성요소의 검색시에는 이러한 주석 또는 구성요소에 기술된 특정 명령어를 가지고 検도록 한다. 이를 위한 질의어는 사용자가 구성요소를 検는데 필요한 단어를 사용하여 작성하도록 설계하고, 검색 기능에서 각 단어를 가지고 이에 관련된 구성요소를 検도록 한다. 이러한 과정은 사용자가 입력한 질의어를 통해 자동적으로 수행되도록 하여 시스템을 관리하고 확장하는 부담을 줄일 수 있게 하였다.

구성요소 검색 시스템은 라이브러리를 초기화 하기 위한 initialize, 구성요소 추가 기능을 지원하는 addModule, 구성요소 검색 기능을 지원하는 searchModules, 구성요소 삭제 기능을 지원하는 removeModule의 서비스를 구현한다.

구성요소를 위한 검색 시스템의 기능을 지원하기 위해서는 〈표 2〉와 같은 3가지 구체화된 facet를 두어 구현한다.

〈표 2〉 Facet의 종류
〈Table 2〉 Kind of the facet

종 류	내 용
대상 Facet	라이브러리에서 사용자가 질의어로 검색한 연관된 구성요소들을 모아 놓은 Facet이다.
언어 Facet	사용하고자 하는 언어의 종류별로 Facet를 구성한다.
특성 Facet	구성요소중에서 특성이 비슷한 것들을 모아 놓은 것이다.

전체 facet에서 사용자가 질의어로 조건에 맞는 후보 구성요소를 대상 facet에서 가져오면, 이 중에서 사용자가 언어 또는 특성으로 해당 facet를선택하여 처리할 수 있도록 하였다. 이를 위해 본 논문에서 언어 facet는 일반적으로 다음과 같은 종류로 제한하였다.

<언어 facet>
A. COBOL
B. C
C. C++
D. Lisp
E. Smalltalk

(그림 3) 제한된 언어 facet의 종류
(Fig. 3) kind of the limited language facet

전체 facet에서 해당 facet로 제한하면서 필요로 하는 소프트웨어 구성요소를 찾아 작업하므로 작업의 흐름을 파악할 수 있고 버전 제어 측면에서 구성요소를 선택하는 부담을 줄일 수 있다. 또한 질의어에 의한 검색과 해당 작업 항목을 마우스로 선택하여 사용자의 편의성을 도모한다.

2) 구성요소 관리 시스템

이 부분에서는 라이브러리에서 검색 시스템으로 검색한 구성요소가 해당 facet에 존재할 때, 이들의

수정, 추가, 삭제, 결합, 저장, 읽기 작업을 처리할 수 있는 기능과 이를 위한 브라우져를 포함한다.

3) 브라우져

일반적인 화면 편집기에서 제공하는 기능을 가진다. 화면 편집기의 기능은 라이브러리 관리 기능이나 새로운 구성요소 관리에 이용된다.

화면 편집기는 Smalltalk 시스템이 가지고 있는 filbrwsr 클래스를 이용한다. 편집용 화면 크기의 정의, 종료는 사용자 질의어를 이용하여 사용자가 직접 작성하도록 한다.

4) 사용자 인터페이스

사용자와 시스템 간의 상호 대화를 위한 경계이며, 이것은 자연스럽고 사용자에게 편리성을 제공해야 한다[19].

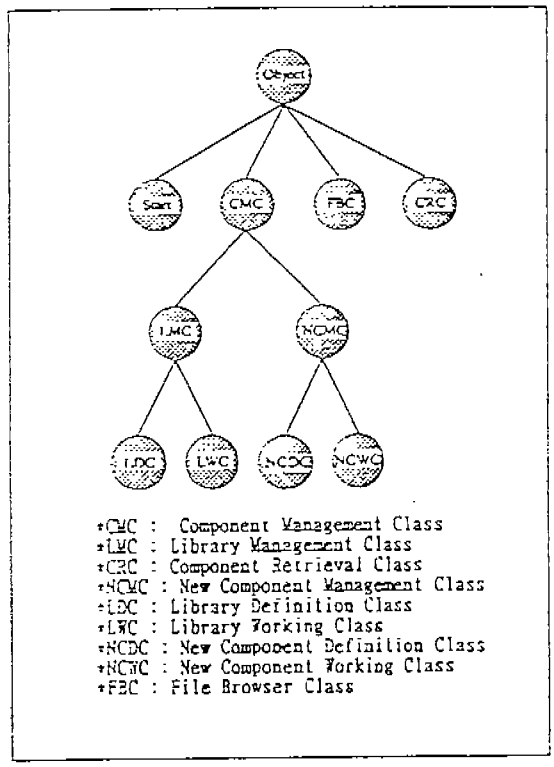
본 논문의 사용자 인터페이스는 주 인터페이스와 서브 인터페이스로 분류한다. 검색 시스템에서 설명한 라이브러리를 관리하고 사용하기 위한 단어의 인덱스와 질의어를 생성하도록 한다. 이것은 사용자 주 인터페이스에서 서브 인터페이스를 사용하여 생성·처리한다. 기본 인터페이스는 검색 시스템과 구성요소 관리 시스템을 질의어나 해당 항목을 마우스로 선택하여 처리할 수 있으며, 4장의 구현 예에서 살펴보기로 한다.

4. 검색 시스템의 구현과 분석

4.1 클래스와 메소드의 구성

본 논문에서는 객체 지향 프로그래밍 언어인 Smalltalk를 사용하여, 버전 제어를 지원하는 검색 시스템을 구현하고, 이를 위한 사용자 인터페이스를 제공한다. 이것은 구성요소의 등록과 검색을 지원하기 위한 사용자 환경을 구축하며, 버전 제어를 위한 기반 시스템으로 사용된다. Smalltalk는 구성요소를 발견하고, 이 구성요소를 변형하여 수정하기 위한 환경을 가지고 있다[1, 4, 6, 7, 8, 11].

검색 시스템에서는 Smalltalk에서 제공하는 강력한 사용자 환경(user environment)과 메시지 전송, 상속성, 동적 연결 등의 기본 특성을 이용하여 구성요소를 검색하고 카탈로깅한다. 이미 개발되어 저장된 구성요소가 구성요소화가 잘 되어있고, 빌딩 블록(building block) 개념에 충실한 특성을 가지고 있으면, Smalltalk와의 명확한 인터페이스를 통하여 사용될 수 있다. 또한, 자료 추상화를 이용한 자료구조에서의 알고리즘과 view/controller 등을 통한 응용에서의 프레임워크(framework)를 사용할 수 있다. (그림 4)는 본 시스템에서 설계, 구현한 클래스의 구성도이다.



(그림 4) 시스템의 클래스 구성
(Fig. 4) Structure of the system class

Smalltalk에서는 객체가 클래스의 인스탄스이며, 클래스는 인스탄스와 특정한 종류의 객체를 사용하고 구축하는데 필요한 모든 정보를 제공한다. 클래스 내의 메소드(methods)는 프로시저로서, 클래스에 메시지를 전송하면 메소드를 호출하여 처리하고, 발생될 처리 방식의 동적(dynamic) 결정에 도움을 주는 상속성을 이용하여 구성요소를 구성한다. 클래스 상의 메소드는 그 실행 중에 사용할 임시 변수를 할당한다.

1) 라이브러리

라이브러리 클래스는 버전 제어를 위한 소프트웨어를 등록할 수 있는 라이브러리를 생성하여 초기화한다. 검색 시스템을 사용하여 라이브러리에 구성요소를 추가, 검색, 삭제할 수 있는 메소드를 지원한다.

라이브러리는 구성요소 관리 프로그램에서의 사용자 화면과 질의어 처리를 통한 구성요소의 추가, 삭제, 검색, 재명명, 수정 등의 작업으로 관리되며, 3장에서 설명한 검색 시스템의 각 기능을 Smalltalk에서의 메소드로 구현하였다.

(그림 4)의 LMC 클래스에서 라이브러리를 관리하며, NCMC 클래스에서는 기존의 구성요소로부터 수정등의 작업을 통해 새로운 구성요소를 작성하고 관리해 주기 위한 기능을 지원한다.

2) 검색 시스템

검색 클래스는 원문을 검색할 수 있는 정보 검색 시스템을 생성하고, 이 원문에 포함된 단어를 가지고 원문의 이름인 구성요소명을 찾아 낼 수 있다.

또한, 검색 클래스를 사용하기 위한 단어의 인덱스와 질의어를 생성할 수 있도록 하며, 사용자 인터페이스에서 생성·처리한다. 인덱스와 질의어를 생성하기 위한 메시지로 initialize를 사용하여 원문과 구성요소명을 저장할 set과 dictionary를 초기화 시킨다.

구현된 검색 시스템은 Smalltalk 상의 Set 클래스를 이용하여 원문을 저장하고, Dictionary 클래스를 사용하여 구성요소명을 기록한다. 즉, 본 논문에서의 검색 시스템을 위해 Set과 Dictionary 클래스를 사용한다. Set 클래스는 중복을 제거하여 객체를 저장하고, Dictionary 클래스는 키 값에 의해서 객체를 저장하고 검색하는 기능을 가진다. Set과 Dictionary 클래스를 이용하여 원문의 내용을 검색한다.

구성요소의 검색에서는 asSet 메시지를 사용하여 검색하고자 하는 구성요소명을 얻는다. 구성요소의 추가는 add:, addWord:메시지를 사용하고, 추가시에는 각 단어의 대소문자에 대한 검색시의 오류 방지를 위해 asLowerCase to:메시지를 사용한다. 검색에서는 occurrencesOf:메시지를 사용하며, 삭제에서는 remove:메시지를 이용하여 처리한다. (그림 4)의 CRC 클래스에서 이 기능을 지원한다.

3) 관리 시스템

구성요소 관리 프로그램은 Object 클래스의 하위 클래스로서 구성요소 관리 프로그램 처리 클래스((그림 4)의 CMC)를 가진다.

이 클래스의 하위 클래스로서 라이브러리 관리 클래스((그림 4)의 LMC 클래스)와 새로운 구성요소 작성 클래스((그림 4)의 NCMC 클래스)가 구성되며, 라이브러리 관리 클래스는 라이브러리 정의 클래스와 라이브러리 작업 클래스를 가진다. 새로운 구성요소 작성 클래스의 하위 클래스로는 새로운 구성요소 정의 클래스와 새로운 구성요소 작업 클래스가 있다.

전체 시스템의 시작은 Object 클래스와 상위 클래스의 속성을 상속 받은 라이브러리 작업 클래스와 새로운 구성요소 작업 클래스로 실행한다.

4) 질의어 처리

각 처리 방식에서는 Smalltalk 윈도(window) 중의 하나인 Prompter와 menu, Display를 사용하여 사용자가 필요로 하는 질의 사항을 처리하며, 사용자에 의해 작성된 질의어의 오류 발생시에는 오류 메시지를 출력한다.

4.2 프로그램 및 실행

검색 시스템의 초기화와 구성요소 추가 프로그램은 (그림 5), (그림 6)과 같다. (그림 5)는 라이브러리 관리 시스템의 초기화 수행시에 한번만 수행되며, Smalltalk상의 Set과 Dictionary 클래스를 이용한다.

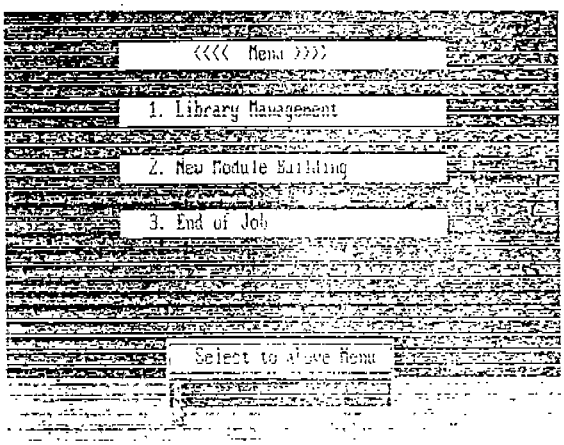
```
initialize
   documents := Set new.
   words := Dictionary new.
```

(그림 5) 구성요소 저장을 위한 초기화 프로그램 부분
(Fig. 5) The part of initialize program for component storage

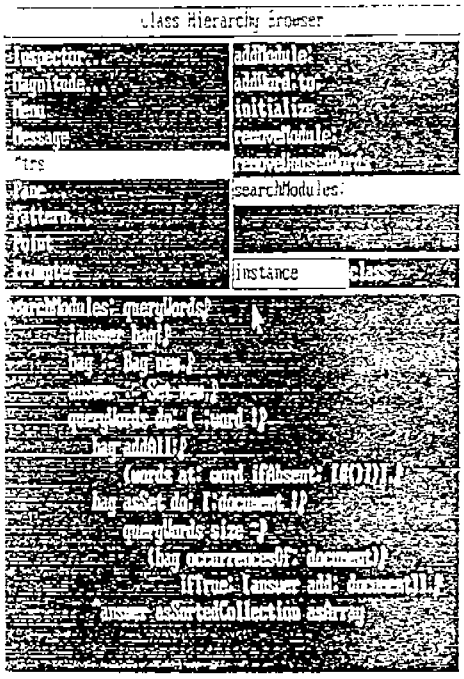
```
addModule: pathName
  | word wordStream |
  (document includes: pathNmame)
     ifTrue: [self removeModule: pathName].
   wordStream := File pathName: pathName.
   documents add: pathName.
   [(word :=wordStream nextWord) == nil]
      whileFalse: [
         self addWord: word asLowerCase to: pathName].
   wordStream close.
```

(그림 6) 구성요소 추가 프로그램
(Fig. 6) Component add program

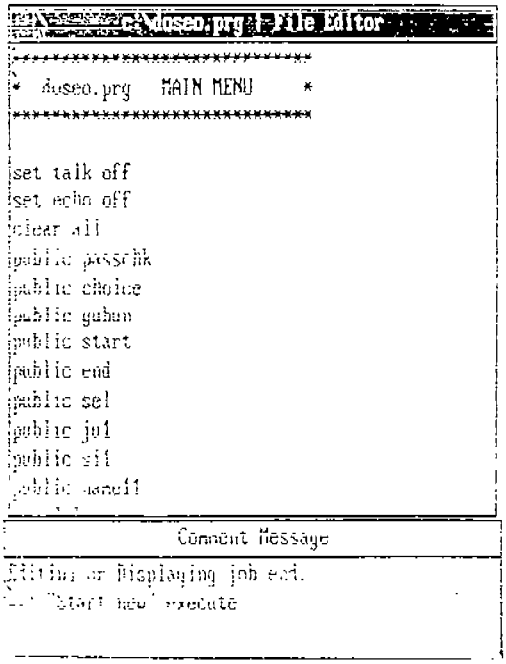
초기 실행 화면(그림 7)은 구성요소 관리 프로그램 시작 클래스(Start 클래스)를 실행하며, 화면상에서 Smalitalk 시스템에 실행 메시지((그림 4)의 Start 클래스를 가지고 "Start new"와 같이 실행한다)를 전송하여 처리한다.



(그림 7) 시스템의 실행 화면
(Fig. 7) Execution screen of the system



(그림 8) 라이브러리 검색 메소드
(Fig. 8) Library retrieval method



(그림 9) 사용자 인터페이스 브라우저
(Fig. 9) User interface Browser

(그림 7)의 각 메뉴에서는 화면에 출력되는 질의어 메시지와 마우스를 사용하여 사용자는 라이브러리에서 소프트웨어 구성요소를 관리할 수 있다.

(그림 8)은 Smalltalk 시스템 상에 구현된 라이브러리 검색 메소드를 시스템 메뉴에서 "browse disk"를 선택, 실행한 다음에 화면상에 제시된 검색 프로그램이다.

5. 결 론

버전 제어를 위해서는 라이브러리에 데이터를 어떻게 구성하여 저장, 관리할 것인가의 문제가 제시되며, 이를 위해 효율적인 검색 시스템이 제공되어져야 한다[5, 9, 14, 15].

본 논문은 Smalltalk 언어를 사용하여 소프트웨어 구성요소를 저장 및 검색할 수 있는 검색 시스템과 질의어를 구현하였으며, 검색 시스템의 지원으로 관리되는 라이브러리와 사용자 인터페이스를 제공하는 시스템을 구현하였다. 이 시스템은 버전 관리를 위한 라이브러리로 사용될 수 있다.

검색 시스템은 많은 구성요소를 저장하고, 빠른 시간 안에 키워드를 이용하여 원하는 구성요소를 검색할 수 있어야 한다. 기존의 검색 방법은 대부분 키워드를 이용하거나 내용을 기반으로한 역 파일 개념 등이 사용되고 있으나, 본 논문에서는 첫째, 객체지향 프로그래밍 언어인 Smalltalk 언어를 이용하여 키워드를 사용하면서도 내용을 기반으로 구성요소를 찾을 수 있는 검색 시스템을 제안하였으며, 이것은 키워드와 내용을 기반으로한 검색 방법의 장점을 제공한다. 또한, Smalltalk에서 제공하는 강력한 사용자 환경을 사용하여 사용의 용이성을 제공하며, 본 검색 시스템에서는 메뉴와 화면에서의 도움말을 제공하였다. 둘째, 시스템의 확장을 제공한다. 시스템이 추가, 변경되는 경우 부분 수정으로 시스템을 변경할 수 있다. 셋째, 시스템의 재사용성을 제공한다. 사용자가 라이브러리에 등록한 구성요소들을 사용자가 원할 때마다 사용함으로써 사용자의 작업 시간 단축과 일관성 있는 작업을 유도하여 품질 향상을 가져올 수 있으며, 본 논문의 검색 시스템은 다른 언어와의 인터페이스를 통해 사용될 수 있다.

앞으로의 연구 과제는 현재 시스템의 제약 사항을 개선 및 보완해 나가며, 소프트웨어 개발 과정의 생산성을 더욱 향상시키기 위한 도구에 대한 추가적인 연구와 효율적으로 구성요소를 정량적으로 측정하여 사용 빈도수가 높은 구성요소를 파악 및 관리하기 위한 방법 등이 연구되어야 한다. 또한, 이들의 시각 프로그래밍과 연관된 아이콘화 및 사용자 인터페이스에 대한 추가 연구가 필요하며, 개발된 라이브러리를 기반으로한 버전 제어 시스템을 구축해야 한다.

참 고 문 헌

[1] Adele Goldberg and David Robson, "Smalltalk-80:The Language and Its Implementation", Addison-Wesley, 1988.

[2] Bernhard Westfechtel, "Revision Control in an Integrated Software Development Environment", ACM, pp. 96-105, 1989.

[3] Bruce A. Burton, Rhonda Wienk Aragon, Stephen A. Baily, Kenneth D. Koehler, and Lauren A. Mayes, "The Reusable Softwar Library", IEEE Software, pp. 25-33, July 1987.

[4] George Copeland and David Maier, "Making Smalltalk Database System", ACM SIGMOD, pp. 316-325, 1985.

[5] Ian Thomas, "Version and Configuration Management on a Software Engineeing Database", ACM, pp. 23-25, 1989.

[6] Jakob Nielsen and John T. Richards, "The Experience of Learning and Using Smalltalk", IEEE Software, pp. 73-77, May 1989.

[7] Lewis J. Pinson and Richard S. Wiener, "An Introduction to Object-Oriented Programming and Smalltalk", Addison-Wesley, 1988.

[8] J. Diederich and J. Milton, "Experimental Prototyping in Smalltalk", IEEE Software, pp. 50-64, May 1987

[9] MARC J. ROCHKIND, "The Source Code Control System", IEEE trans. on software eng., vol. 1, no. 4, pp. 364-370, DECEMBER 1975.

[10] Ruben Prieto-Diaz and Peter Freeman, "Classifying Software for Reusability", IEEE Software, pp. 6-16, January 1987.

[11] Smalltalk/V, digitalk inc., 1986.

[12] Scott A. Kramer "History Management System", ACM, pp. 140-143. 1991.

[13] Susan P. Arnold and Stephen L. Stepoway, "THE REUSE SYSTEM:CATALOGING AND RETRIEVAL OF REUSABLE SOFTWARE", Proceedings of COMPCON S'87, pp. 376-379, 1987.

[14] Tichy, W. F., "RCS-A System for Version Control", Software Practice & Experience, Vol. 15, No. 7, pp. 637-654, 1985.

[15] Vincenzo Ambriola, Lars Bendix, Paolo Ciancarini, "The evolution of configuration management and version control", Software Engineering Journal, pp. 303-310, November 1990.

[16] W. B. Frakes and B. A. Nejmeh, "An Information System for Software Reuse", Proceedings of the Tenth Minnowbook Workshop on Software Reuse, 1987.

[17] 장명섭, 정인상, 권용래, "재사용을 위한 소프트웨어의 분류 및 검색 방법", 한국정보과학회논문지 제19권 6호, pp. 602-613, 1992.

[18] 김덕현, 박성주, "확장된 객체지향 데이터 모형을 이용한 소프트웨어 변경 관리 시스템", 한국정보과학회논문지 제 22권 2호, pp. 249-260, 1995.

[19] 김상근, 홍찬기, 이경환, "소프트웨어 재사용 시스템을 지원하는 사용자 인터페이스 구축기의 설계 및 구현", 한국정보처리학회논문지 제 2권 3호, pp. 324-334, 1995.

장 덕 철

1974년 고려대학교 대학원 경영정보학 석사
1982년 고려대학교 대학원 경영정보학 박사
1981년~1982년 버클리 대학교 객원 교수
1976년~현재 광운대학교 전자계산학과 교수
관심분야: 소프트웨어 재사용성 CASE Tool 소프트웨어 형상 관리

김 홍 진

1983년 광운대학교 전자계산학과 졸업(이학사)
1985년 광운대학교 대학원 전자계산학과(이학석사)
1993년 광운대학교 대학원 전자계산학과(이학박사)
1975년~1983년 민컴 근무
1984년~현재 경원 전문대학 교수
관심분야: 소프트웨어공학(특히, 소프트웨어 재사용) 시스템 소프트웨어

오 상 엽

1989년 경원대학교 전자계산학과 졸업(이학사)
1991년 광운대학교 대학원 전자계산학과(이학석사)
1992년~현재 광운대학교 대학원 전자계산학과 박사 과정
1992년~현재 경원 전문대학 교수
관심분야: 소프트웨어공학(특히, 소프트웨어 버전관리, 객체지향 소프트웨어)