재사용에 기반한 객체들의 정보 분석과 자동화에 관한 연구

김 재 생† · 송 영 재††

요 약

재사용에 관한 연구는 여러 가지 방향에서 연구되어왔으나 실제로 객체들을 재사용하는 성공사례는 특히 드물다. 그 이유는 재사용을 지원하는 기술적인 환경이 부족하기 때문이다. 본 논문에서는 C++ 객체들의 재사용성에 관한 정보들을 분석하고, 추출된 정보들을 정보 DB에 저장하고 운영하는 시스템에 관하여 연구하였다. 이 개발 시스템에서 사용자들은 객체들의 명세서 정보와 객체적, 절차적 속성의 품질 정보들을 브라우징하여 볼 수 있고, 재사용 정보 DB에 없는 객체들은 사용자가 직접 정보 분석 환경을 이용하여, 자동으로 객체의 정보들을 분석 및 테스트 할 수 있다. 이 시스템은 C언어로 X-Window OSF 1/Motif 2.1.1 하에서 구현하였다.

A Study on Information Analysis and Automation of Objects based on Reusability

Jae Saeng Kim† · Young Jae Song††

ABSTRACT

Research about Reuse is developed in various directions, but the successful examples of the reusable objects are very rare in practice because technical environment to support reuse is short. In this paper we describe a system for analyzing the information about reusability of C++ objects and for storing and managing the extracted information in information DB. In this development system, user can see the specification information of objects and the quality information of the procedual and object oriented attibutes by browsing. And objects that are not in the reuse information DB is analyzed and is tested automatically using the information analyzer by user. We implemented this system using C program in X-Window OSF 1/Motif 2.1.1.

1. 서 론

1960 연대의 "소프트웨어 위기" 이후에 오늘날에 이르기까지 소프트웨어의 생산성을 향상시키기 위한 많은 노력이 요구되어, 소프트웨어 개발기술, 방법론, 파라다임, 환경에 관한 연구가 많이 진행되어 왔다. 그 중에서도 소프트웨어의 재사용은 소프트웨어 개발 비용이 감소하고, 시스템의 신뢰성이 증가하고, 소프트웨어 개발 기간과 개발 생산성이 감소하고, 소프트웨어 시스템의 설계 지식을 공유할 수 있다는 장점이 있다[8].

기존의 재사용 시스템들은 부품들을 분류하고 검

† 정 회 원:경희대학교 전자계산공학과 강사
†† 종신회원:경희대학교 공과대학장
논문접수:1996년 10월 14일, 심사완료:1997년 2월 10일

색하는 방법에 치중되어 재사용에 관한 정보들을 분석하고 보여주는 기능이 부족하여 효율성이 적었다. 대부분의 재사용 시스템들은 절차언어인 C 코드, Fortran 코드, Cobol 코드, 프로그램 기술서에서 정보들을 분석 및 추출하였기 때문에 객체지향언어의 특성 정보를 분석하지 못하였다[3][4][6][7][11][12].

또한, RSL 시스템은 ADA 원시코드와 설계서에서 정보를 추출하였으나 마찬가지로 OOP의 특성 정보를 추출하지 못하였다[2]. CIA++ 시스템은 C++ 코드를 분석하였으나 함수 관계와 상속 관계를 계층적으로 나타내고 정보들을 테이블 형식으로 표현하여 사용자 환경을 지원해 주지 못하였다[5].

RESOFT 시스템은 C++ 코드에서 객체 지향 정보들을 포함한 일반 기술서의 정보를 추출하였으나 정보 브라우징 기능과 부품들의 객체적인 품질 정보를 지원하지 않았다[8].

따라서 본 논문에서는 절차언어의 품질 정보들과 기존의 연구 결과에서 부족한 객체지향언어의 품질 정보들을 정량적으로 분석 및 추출하는 자동화 환경에 관하여 소개한다. 구현된 시스템은 C++ 언어의 재사용 가능한 부품인 클래스와 멤버함수를 재사용할 때 프로그래머가 클래스와 멤버함수의 정보를 이해하는 데에 도움을 줄 수 있고 클래스와 멤버함수의 품질을 자동으로 평가하여 볼 수 있다. 또한, 추출된 클래스 품질의 정량적인 값들은 유사한 후보자들이 여러 개 있을 경우 평가 정보로서 사용되었다[9].

2. 관련연구

2.1 기존의 재사용 정보 시스템

이절에서는 기존 재사용 시스템들의 정보 종류와 지원 환경들을 비교하고 재사용의 문제점들과 재사용을 지원하는 시스템이 갖추어야 하는 특성들을 살펴보기로 한다.

2.1.1 Prieto와 Freeman의 시스템

이 시스템[1]은 프로그램 기술서와 원시 코드로 부터 부품의 수행기능, 수행환경, 구현사항, 구현환경 등의 정보를 추출하였다. 재사용 결과의 가장 적절한 지시자로서 프로그램 크기, 구현언어, 프로그램 구조(순환수, 모듈의 수, 링키지 수), 프로그램 문서화 정도, 재사용자 경험 정도 등의 5가지 정보 속성들을 후보자 평가 정보로서 사용하였다. 이 정보들은 퍼지함수를 사용하여 값들을 측정하였기 때문에 정확한 정량적인 값들의 추출이 어렵다. 예를 들면, 문서화 정도가 좋다(good), 나쁘다(bad)와 같은 비정량적인 표현을 0.7 등의 수치값으로 정량화하기 위하여 퍼지함수를 사용하였다.

2.1.2 RSL 시스템

이 시스템[2]은 원시코드와 PDL 설계서로부터 unit name, category code(부품의 함수성 기술), 사용한 컴퓨터 기종, 컴파일러 종류, 키워드, 저자, 생성된 날짜, 갱신된 날짜, 버전, 요구사항서, 부품의 기술, 에러, 사용한 알고리즘, 문서화 정도, 테스팅 등의 속성 정보들을 추출하여 부품의 분류와 검색, 후보자 평가 등에 이용하였다.

2.1.3 Selby의 정보 분석

Selby[3]는 포트란 언어로 구현된 25개 시스템을 프로젝트 레벨, 모듈 설계 레벨, 모듈 구현 레벨 등의 3개 단계로 나누어 정보를 분석 및 추출하였다. 프로젝트 레벨에서는 프로젝트의 크기, 프로젝트의 시작 날짜 정보를 추출하였고, 모듈 설계 레벨에서는 함수 호출의 수, 모듈간의 인터페이스(입출력 문장의 수), 문서화 정도, 설계시의 노력도를 추출하였다. 모듈 구현 레벨에서는 새로 개발된 모듈의 수, 수정된 정도에 따른 모듈의 수, McCabe의 순환수, 비 제어흐름의 정도 등의 정보를 분석하였다.

2.1.4 CIA 시스템

이 시스템[4]은 C 원시코드에서 관계 정보들을 추출하였고, 파일 영역, 마크로 영역, 데이터형 영역, 글로벌 변수 영역, 함수 영역 등의 5 가지 객체 영역으로 나누어 정보를 운영하였다. 각 부품은 파일명, 데이터 타입, 함수 이름, 시작 라인과 끝 라인 등의 정보를 테이블 형식으로 관계 데이터 베이스에 저장하였다. 또한, C 원시코드의 내부구조인 함수들의 호출 관계를 보여주는 그래픽 뷰와 재사용 가능한 서브 시스템의 추출과 dead 코드 삭제, 바인딩 분석 등을 지원한다. 이 시스템은 사용자가 명령어(command)방식으로 질의하므로 사용자 인터페이스 환경 지원이 부족

하다.

2.1.5 CIA++ 시스템

이 시스템[5]은 CIA[2]의 정보 요약기를 기반으로 하여 C++ 언어를 지원하는 툴이다. 이 모델은 C++ 원시코드에서 파일, 마크로, 타입들, 변수들과 함수들 등의 5가지 개체들을 인식하여 정보를 추출 운영하였다. CIA와 마찬가지로 명령어 방식의 질의를 사용하여 사용자 지원 환경이 없고 재사용을 지원하는 객체적인 품질에 관한 정보를 지원하지 못하였다.

2.1.6 Arnold와 Frakes 의 시스템

이 시스템[6]은 역공학을 이용하여 원시 소프트웨어를 조사하여 수정, 변형시켜 부품을 재사용하였다. 재사용 효율성을 측정하기 위해서 추상화 계층성, 외부 저장소의 정의, "uses" 관계 정의, 각 부품의 이름, 코드, 추상화 레벨 등의 정보들을 사용하였다. C 언어로 구현된 시스템을 함수로 분해하여 재사용 레벨을 계산하였지만, 재사용 부품을 얻을려고 하는 정보에만 치중하여 정보 운영과 객체지향언어의 품질 정보가 없고 사용자 환경을 지원하지 않았다.

2.1.7 Code Miner 시스템

이 시스템[7]은 C 언어로 구현된 원시코드에서 재사용 부품들을 추출하였으며, 호출되는 함수들의 관계, 함수들간의 결합도, 전역 데이터 등의 정보들을 이용하여 재사용 가능한 부품들을 추출하는 데에 사용하였다. 지원 환경에서는 사용자들이 프로그램을 선택하고 분석 결과를 사용자에게 디스플레이하였다. 그러나, 함수들의 절차적인 정보들에만 치중하여 객체적인 정보들을 분석하지 못하였다.

2.1.8 RESOFT 시스템

이 시스템[8]은 부품들을 라이브러리에 저장, 검색을 지원하고 소프트웨어 명세서와 정보를 브라우징하여 볼 수 있다. 객체지향언어의 상속성과 클래스, 프로시듀어, 데이타의 명세서 정보를 이용하여 재사용 가능한 부품들을 탐색하는 데에 사용하였다. 명세서의 상세 정보들은 클래스 이름, 제한 사항, 일반적인 묘사, 오퍼레이션과 속성들, 키워드들, 슈퍼 클래스, 구현 언어, 작동 환경, 특성 명세, 저자 등이다. 정보들의 운영은 지원하지만 정보들의 자동적인 분석 과정은 지원하지 못하였다.

2.1.9 재사용 모듈 검색 시스템

이 시스템[11]은 코볼 언어를 지원하며 설계 모듈들을 재사용 대상으로 하여 원시 모듈, 제어 모듈, 원시코드 등을 선택하였다. 모듈의 외부 명세서에서는 모듈명, 모듈의 기술, 입출력 데이터 흐름도, 입출력 데이터를, 모듈의 내부 명세서에서는 모듈명과 알고리즘 과정을, 모듈 명세서에서는 정보은닉을 사용한 함수명과 기술 등의 정보들을 사용하였다. 또한, 품질 정보면에서는 모듈의 결합도, 응집도, 분해 가능성, 제한성, 포괄성, 정보은닉, 독립성, 기능, 인터페이스, 모듈의 이해성과 문서화, 수정 용이성, 신뢰성을 사용하였다. 이 시스템에서는 여러 가지 정보들을 부품의 분류와 검색에 사용하였지만, 객체지향언어면의 품질 정보들을 지원하지 않았다.

2.1.10 소프트웨어 부품들의 분류와 검색 시스템

이 시스템[12]은 부품의 정보, 원시코드, 문서화일을 저장 관리하는 라이브러리, 질의를 위한 키워드들을 저장 운영하는 시소러스 관리와, 질의어 처리, 가중치 계산, 브라우징을 허락하는 부품의 검색 및 이해 등의 3가지 요소로 구성되었다. 부품의 기능, 객체, 메디움, 언어, 운영체제, 가중치 등의 정보들은 질의와 검색에, 키워드들은 어구사전 운영에 사용되었다. 이외에도 부품명, 유사도, 원시코드명, 문서화명, 예제 프로그램 등의 정보들을 사용하였다. 이 논문에서는 부품들의 품질에 관한 정보들은 지원하지 않았다.

2.2 검토사항

소프트웨어 재사용이 많은 잇점에도 불구하고 실제로 재사용 가능한 부품들을 재사용하는 프로그래머들은 드물다. 그 이유는 다음과 같다.

· 대부분의 개발자들은 개발자 자신이 타격을 받기 때문에 일반적이고 재사용 가능한 소프트웨어를 개발하지 않을려고 한다.

· 재사용성에 기초한 툴들의 부족-기존의 재사용 시스템들이 정보의 검색과 분류에 치중되어재사용 정보들을 자동으로 분석하고 **추출해주는** 기능이 부족하다.

· 재사용 가능한 소프트웨어 부품들을 표현하는 정보들의 표준화가 부족하여 부품들을 분류하고 이해하기가 어렵다.

· 재사용 가능한 소프트웨어 라이브러리의 부족은 부품들의 재사용을 어렵게 만든다.

· 재사용 가능한 큰 라이브러리에서 재사용하고자 하는 부품들의 탐색과 검색을 지원하는 툴이 부족하다.

재사용 가능한 소프트웨어 부품들이 많이 있다고 하더라도 효율적인 검색 메카니즘과 이 메카니즘을 지원하는 재사용 정보 분석 툴이 없다면 재사용은 아직도 비 실제적이라고 말할 수 있다.

이와 같이 재사용 문제점들과 앞절에서 기술한 시스템들을 비교하여 본 결과, 재사용을 지원하는 정보 분석 환경은 다음과 같은 특성들을 가져야 한다.

· 정보들의 분석과 추출-재사용 가능한 부품들의 특성에 따라서 정보들을 자동으로 분석하고 추출하는 기능을 제공해야 한다.

· 정보들의 분류-명세서 정의서에서 사용되는 정보들과 부품의 품질 평가에 사용될 정보들을 구분하여 부품을 분석하여야 한다. 이러한 정보들은 재사용 가능한 부품들의 품질 평가 척도에 사용할 수 있다.

· 정보들의 운영-기존 시스템의 대부분이 정보 라이브러리를 갖고 있어서 다양한 정보들을 이용하여 부품들의 검색과 분류에 사용하고 있었다. 그러므로 정보를 저장, 삭제, 수정하는 기능이 있어야 한다.

· 브라우징 환경-사용자가 재사용하고자 하는 부품들의 정보들을 질의하여 결과를 볼 수 있는 브라우징 기능을 가진 그래픽 환경이 제공되어야 한다.

기존의 시스템들은 대부분이 정보 라이브러리를 갖고 있어서 여러 분야에 이용하고 있지만, 정보들을 자동으로 분석하고 그 정보를 사용자에게 보여주는 사용자 인터페이스 기능이 부족하였고[1]~[6][8], 사용된 정보들은 거의 절차언어의 속성 정보에만 치중되어 있었다[1]~[4][6][7][11][12].

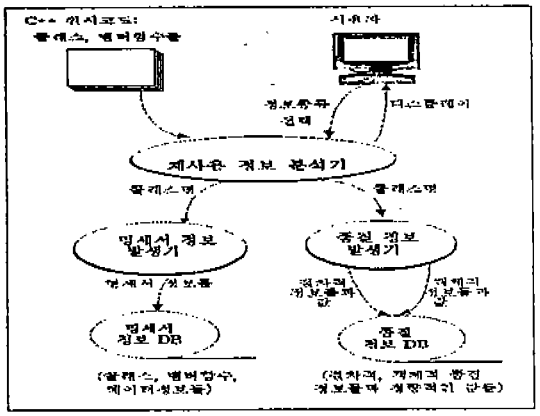
이상의 검토사항들을 감안하여, 본 논문에서는 C++ 언어의 재사용 가능한 클래스들과 멤버함수의 명세서를 지원하는 일반적인 정보들과 절차언어의 품질 정보들과 객체지향언어의 품질 정보들을 자동으로 분석하고 디스플레이 해주는 기능과 정보 운영을 지원하는 시스템을 설계 및 구현하였다. 클래스들의 품질 분석 정보들의 종류들과 정의식에 의한 정량적인 값 추출은 발표한 [9]의 논문을 참조하였다.

3. 시스템의 설계와 구현

3.1 시스템 구성과 구현환경

재사용 정보 분석기는 (그림 1)과 같이 2 가지 정보 발생기를 가지고 있다. 정보 분석 과정은 사용자가 원하는 클래스의 이름과 정보 종류를 선택하면, 선택된 정보의 종류에 따라 각 발생기는 원시코드를 읽고 분석하여 정보를 발생시킨다. 정보 분석기는 클래스와 멤버함수의 명세서 정보 발생기, 품질 정보 발생기의 2 가지 종류의 정보들을 발생시킨다.

명세서 정보 발생기는 클래스와 멤버함수의 재사용 이해를 돕기 위하여 클래스 명세서, 데이터 멤버 명세서, 멤버함수 명세서 등에서 사용하는 정보들을 발생하고, 품질 정보 발생기는 클래스들의 절차적인 품질 정보들과 객체적인 품질 정보들을 발생시킨다. 발생한 정보들은 재사용 정보 DB에 저장되고 사용자가 원할때는 필요한 정보를 사용자에게 디스플레이 해주며 새로운 부품일때는 그 부품들의 정보를 사용자가 직접 정보 발생기를 통하여 테스트해 볼 수 있다.



(그림 1) 재사용 정보 분석기의 구성도
(Fig. 1) Structure of reuse information analyzer

재사용 정보 분석기의 프로토타입은 SUN Sparc 10 워크스테이션에서 개발하였고 C 언어로 구현하였다. 이 정보 분석기는 KHR 재사용 시스템[10]에 부가

하였고, 시스템의 크기는 총 6000라인 이상으로 구성되었다. 프로토타입에서는 분석 및 추출된 정보를 데이터 베이스에 저장하는 대신 SUN 파일로 저장하였다. 사용자 인터페이스는 X win dow 환경에서 Motif 룰 이용하여 구성하였다.

3.2 재사용 정보 발생기와 명세서 운영

재사용 정보 발생기는 사용자가 원하는 정보의 형태에 따라 명세서 정보 발생기, 품질 정보 발생기를 선택할 수 있도록 되어 있다. 각 발생기는 멤버함수를 포함한 클래스들을 입력 및 분석하고 클래스와 멤버함수들의 품질 정보들을 발생하여 재사용 정보 DB에 저장하고 사용자에게 그 결과를 디스플레이 해준다.

재사용 정보를 분석 및 추출하기 위해서는 어휘 분석 단계, 정보 인지 및 추출 단계를 거쳐서 해당 정보를 저장하게 된다. (그림 2)는 심볼 테이블과 클래스 정보 테이블을 나타낸다. 어휘 분석 단계에서는 입력 스트림에서 공백 문자를 제거하고 키워드, 오퍼란드, 오퍼레이터 등의 기본 단위인 토큰을 생성한다. 특히 키워드와 사용자 정의 변수는 심볼 테이블에 저장하고 필요할 때 이용한다.

정보 인지 및 추출 단계에서는 발생된 토큰들이 명세서에 필요한 정보인가? 절차적 품질 정보인가? 객체적 품질 정보인가? 를 구별하여 각 해당 정보들을 클래스 테이블에 저장하고 그 결과를 사용자에게 디스플레이 해주며 발생된 정보들을 파일로 저장하였다.

```
  typedef struct token {
     char *tok;          // 토큰 저장
     int tok_type;       // 토큰의 형
  };
struct c_inf {
    char cname[40];       //클래스명
    char bname[50];       // 베이스클래스명
    char date[10];        // 생성된 날짜
    char version[3];
    char author[30];      // 저자명
    char category[30];    // 참조클래스
    char friend[5][40];   //프렌드클래스명
    struct data  dataDB[15]; // 데이터 객체들
    struct module moduleDB[15];// 멤버함수들

    char header[5][20];   // 헤더화일명
    char fname[20];       // 화일명
    char cpurpose[240];   // 클래스의 목적
};
```

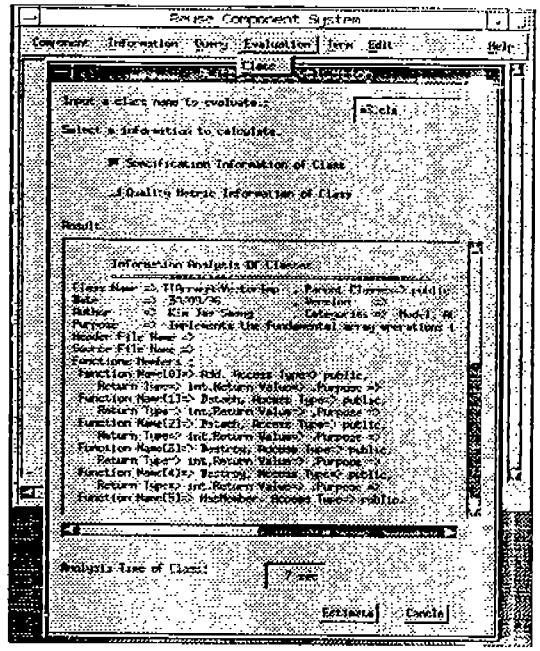
(그림 2) 심볼 테이블과 클래스 정보 구조

(Fig. 2) Structure of symbol table and class information

```
while(tok!=NULL){
     get_token(tok);              // 토큰을 얻는다.
     if(tok=="protected"||tok="private"
          ||tok=="public")        return(tok_type);
     else if(tok=="#include")     return(tok_type);
     else if(tok=="Filename")     return(tok_type);
     else if(tok=="Author")       return(tok_type);
     else if(tok=="Creation")     return(tok_type);
     else if(tok=="Categories")   return(tok_type);
     else if(tok=="Purpose")      return(tok_type);
     else if(tok=="class")        return(tok_type);
     else if(tok."friend")        friend_analysis();
     else if(tok."operator")      operator_analysis();
     else if(tok=="//"||tok=="/*") return(COMMENT);
     else if(tok=="{") brace++;
     else if(tok=="}") brace--;
     else if(tok."}:") cbrace--;
     else { /* datas and function */
       type=look_up(tok);
       if(type==KEYWORD)   keyword_analysis();
       if(type==IDSTRING)  idstring_analysis();
       else if(type==MODULE)
            module_analysis();
       else if(type==DATATYPE)
            datatype_analysis();
     } /* else */
}
```

(그림 3) 정보 인지 및 추출 알고리즘

(Fig. 3) Algorithm for information recognition and abstraction



(그림 4) 클래스의 명세서 정보 발생

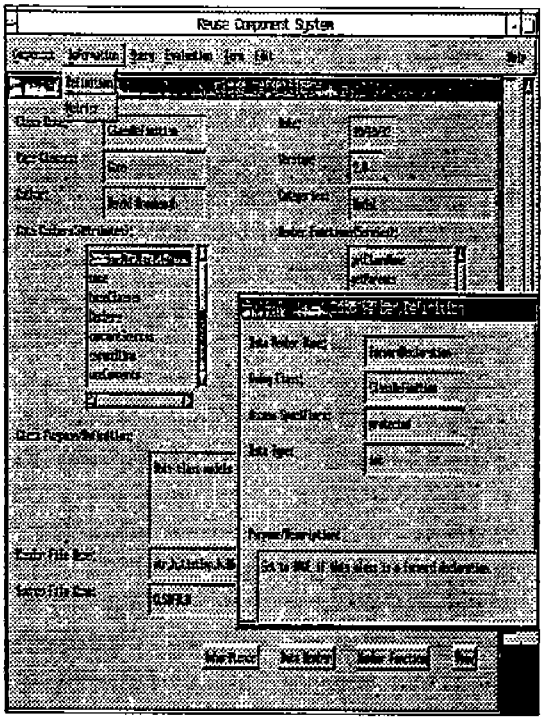
(Fig. 4) Generation of specification information about class

클래스의 목적이나 저자명 등은 원시 코드의 코멘트 부분에서 인식하여 처리하였다. 사용자가 선택하는 정보의 종류와는 상관없이 토큰은 발생되어 (그림 3)의 토큰의 형을 인지하는 알고리즘을 거친다. 토큰의 정보 인지에 따라 정보 발생기는 한 라인씩 스트림들을 읽어서 선택된 정보의 형에 따라 해당 정보를 클래스 정보 테이블에 저장하였다.

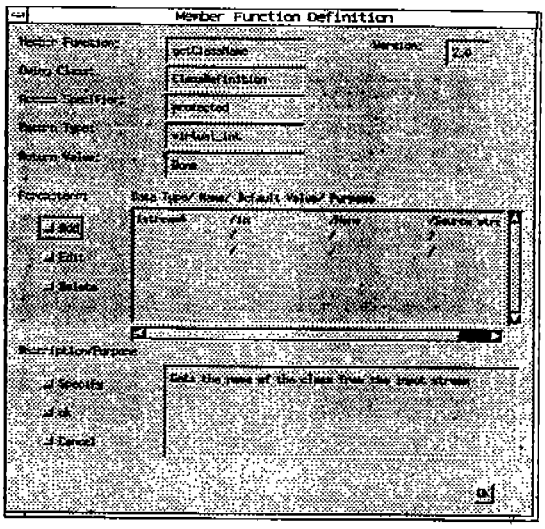
(그림 4)는 클래스의 명세서 정보 발생기이다.

사용자는 원하는 클래스 정보가 재사용 정보 DB에 없을 경우에는 클래스의 이름(예:B1.CLS)을 텍스트 박스에 입력하고 클래스 정보의 토글 버튼을 누르면, 클래스의 코드가 자동으로 분석 및 추출된 정보 리스트가 텍스트 영역에 나타난다.

사용자가 원하는 클래스의 정보가 재사용 정보 DB에 저장되어 있을 때에는 주메뉴인 정보(Info rmation 메뉴)에서 정의서(Definition) 서브 메뉴를 누르고 원하는 클래스의 이름을 삽입하면, 해당하는 클래스 명세서가 (그림 5(a))와 같이 디스플레이된다. 사용자가 데이터 멤버들의 리스트 박스에서 알고 싶은 데이타를 하나 선택하면, 데이터 멤버에 관한 정보 명세서가 (그림 5(a))의 오른 편에 나타난다.



(그림 5(a)) 클래스의 명세서 정보 운영
(Fig. 5(a)) Management of specification information about class

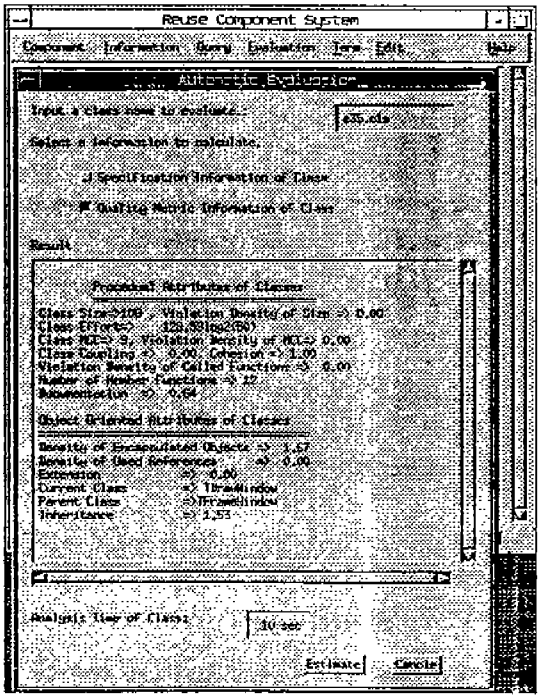


(그림 5(b)) 멤버 함수 명세서 정보 운영
(Fig. 5(b)) Management of specification information about member function

데이터 멤버의 정보 명세서를 알기 원할 때에는 멤버함수들의 리스트 박스에서 알기를 원하는 함수를 하나 선택하면, 해당하는 멤버함수의 명세서가 (그림 5(b))와 같이 나타난다. 클래스 명세서에서는 클래스 명, 버전 번호, 부모 클래스 명, 클래스의 목적, 데이터 멤버들의 리스트, 데이터명과 타입, 데이터의 목적, 멤버함수들의 명과 리턴값, 멤버함수의 목적 등의 정보들을 포함하고 있다. 그러므로 클래스의 명세서 정보들은 재사용성 속성 모델을 개선하기 위해서 사용한다.

3.3 클래스의 품질 정보 발생기와 정보 운영

우리는 재사용 가능한 클래스에 관한 품질 정보들의 정량적인 값을 추출하기 위해서 [9] 논문에서 제안한 함수들을 사용하여 계산하였다. 클래스의 품질 분석 및 추출은 2 가지 종류로 나누어서 절차언어의 특성 정보와 객체지향언어의 특성 정보 값들을 추출하였다.



(그림 6) 클래스의 품질 정보 발생
(Fig. 6) Generation of quality information about class

```
keyword_analysis(char *tok, int type)
{
if(type==EOL){
    line++; // 함수크기
    if(line>30 && MODULE_END) non_size++; // 크기위반함수
  }
 else if(type==NEW_OPERATOR){
   n1++; // 노력도의 n1 계산
   N1++; //    //    N1 계산
 }
 else if(type==OLD_OPERATOR) N1++; // 사용된 연산자 수
 else if(type==NEW_OPERAND){
   n2++; // 노력도의 n2 계산
   N2++; // 노력도의 N2 계산
 }
 else if(type==OLD_OPERAND) N2++;  // 사용된 피연산자 수
 else if(type==FOR||WHILE||IF||ELSE||CASE)
   mcc++;  // MCC 의 순환수 계산
 else if(type==ARGUMENT){
   arg_cnta++;  // 호출함수의 인수 수
 else if(type==DATA && COMMON_DATA) copling_cnt++;
 else if(type==FCALL){
   fcall_cnt++; // 호출되는 함수 수
   if(fcall_cnt>10) non_fcall++;//호출된함수들의 위반정도
 }
 else if(type==PRIVATE && type==PROTECTED){
   encap++; // 엑세스된 총수 - 캡슐화된 정도
   if(declare(tok)) declaer_enc++; // 선언된 수
 }
 else if(type==REFERENCE) {
   ref_cnt++; // 사용된 참조자 수
   if(declare(tok)) declare_ref++; // 선언된 수
 }
 else if(type==MEMBER){ // 상속도 계산
   member_cnt++;  // 클래스의 멤버들의 수
   if(type==PROTECTED||PUBLIC)
     current_inh++; // 보호부와 공용부의 멤버수
 }
 else if(type==OVERLOAD||VIRTUAL||FRIEND||INLINE||MACRO)
   extension++;  // 확장성
 else if(type==COMMENT||STUCTURED_STATEMENT)
   document++;  // 문서화 정도
}
```

(그림 7) 클래스의 품질 정보 인지와 정량화
(Fig. 7) Quantization and recognition of quality information about class

절차언어의 재사용 특성으로서는 한 클래스의 크기와 크기 위반 정도, 노력도, 순환수 정도, 함수들의 호출 위반 정도, 문서화 정도 등의 정보들을, 객체지향언어의 재사용 특성으로서는 클래스의 상속 깊이와 베이스 클래스의 수에 따른 상속 정도, 확장성, 정보은닉 정도, 참조자 사용 정도 등의 정보들을 [9] 논문에서 제안한 공식에 의거 정량적으로 추출하였다.

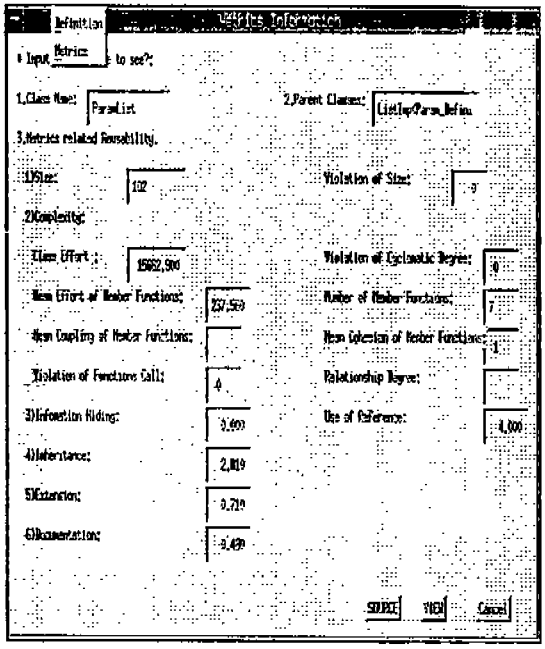
사용자가 어떤 클래스의 품질 정보를 알기 원할 때에 (그림 6)과 같이 클래스의 이름을 삽입한 후, 클래스의 품질 분석을 나타내는 토글 버튼을 선택하면, 재사용 정보 분석 발생기는 분석 추출된 결과 리스트를 텍스트 영역에 보여준다.

품질 정보 분석 추출기는 분석 과정에서 문장들의 키워드들을 구분하여 구문적인 분석을 하게 된다.

(그림 7)은 입력된 클래스에 [9]논문에서 정의한 정의식들을 적용하여 품질 정보의 값들을 정량적으로 계산해내는 알고리즘이다. 예를 들면, 오퍼란드와 오퍼레이터의 종류와 사용된 횟수를 카운터하여 클래스의 노력도와 순환수를 구할 수 있고, 클래스 크기는 실제의 코드 라인 수를, 정보 은닉 정도는 멤버들의 캡슐화된 정도와 참조자들의 사용된 정도를, 상속도는 현 클래스의 상속도를 먼저 구한 다음, 베이스 클래스의 상속도를 더하였다. 확장성은 확장 가능한 함수들의 수를 카운트하여 멤버들의 총 수로 나누어 계산하였다. 문서화 정도는 코맨트 수와 구조적인 문장 수를 카운트하여 구하였다.

원하는 클래스의 품질 정보가 재사용 정보 DB에 저장되어 있을때는 자동화 과정을 거치지 않고 재사용자는 직접 원하는 정보들을 (그림 8)과 같이 볼 수 있다.

4. 실험 및 평가



(그림 8) 클래스의 품질 정보 운영
(Fig. 8) Management of quality information about class

4.1 재사용 정보들의 분석 및 추출 결과

설계 구현된 재사용 정보 분석기의 성능을 실험하기 위하여 2개의 파일[13][14]과 볼란드 C++ 4.0의 예제 클래스들을 대상으로 하여 테스트하였다. 〈표 1〉은

〈표 1〉 재사용 정보들의 분석 및 추출 결과
〈Table 1〉 Result of analysis and extraction about reuse information

화일명		Windows	Case	Borand C++
클래스 총 라인수		2,786	3,385	7,017
클래스 수		58	35	104
멤버함수들의 수		422	425	1,124
클래스 1개당 평균 멤버함수들의 수		7.275	12.142	10.807
분석된 정보수	명세서 정보 수	638	385	1,144
	품질 정보 수	348	210	624
클래스의 명세서 정보 분석 평균 실행 시간		10분	15분	8.52초
클래스의 품질 정보 분석 평균 실행 시간		25분	30분	9.71초

3개의 파일들에서 클래스들을 추출한 다음, 2개의 파일(Windows, CASE)은 수동작으로 1개의 파일(Borland C++)은 프로토타입을 통하여 비교하였다.

클래스 1개당 평균 라인수는 48.034, 86.714, 67.471 라인이며, 멤버함수들의 수는 평균 7.275, 12.142, 10.807개를 가지고 있었다. 한 클래스는 7±2개의 멤버함수들을 다루는 것이 좋고[15], 한 멤버함수의 크기[16][17]는 보통 사람이 한눈에 볼 수 있는 한 페이지 정도가 좋으며, 실제 크기는 30라인 정도이다. 한 클래스는 7±2개의 멤버함수로 구성되어 있으므로 클래스의 실제 크기는 210라인 이하로 구성되어 있는 것이 좋다.

그러므로 Windows 파일의 클래스들은 멤버함수들이 알맞게 설계되었고, CASE 파일과 Borland C++ 파일의 클래스들은 멤버함수들이 7±2에 위반되므로 좋은 구성은 아니라는 것을 알 수 있었다. 또한, 3개 화일 클래스들의 평균 크기는 210라인 이하이므로 모두 적절한 크기를 갖고 있었다. 이와 같이 분석 및 추출된 재사용 정보들은 재사용자에게 직접 화면을 통하여 보여줄 수 있고, 재사용 부품의 선정시나 클래스 후보자들의 평가시에 이용 자료로서 많은 도움을 줄 것이다.

평균 실행 시간은 재사용자가 클래스명을 입력하고 난 후, 결과 리스트가 나오기까지의 시간을 측정하였다. 실행 시간이 짧은 이유는 큰 파일에서 추출된 클래스들과 멤버함수들만을 입력으로 했기 때문이며, 정보들을 2가지 종류인 절차적 정보와 객체적 정보로 구분하여 분석 추출했기 때문이다. 수동작으로 계산했을 때의 시간은 명세서 정보면에서는 각 정보들을 추출하여 기록하는데 시간이 많이 걸렸고 품질 정보면에서는 토큰의 종류를 식별하여 연산자와 피연산자를 구분하는데에 시간이 걸렸다. 수동작과 분석기를 통해 자동으로 동작할 때 걸린 시간을 비교해보면, 프로토타입을 통해 정보들을 분석 및 추출된 실행 시간이 수동작으로 실행된 시간보다 70배이상 빠르다는 것을 알 수 있었다.

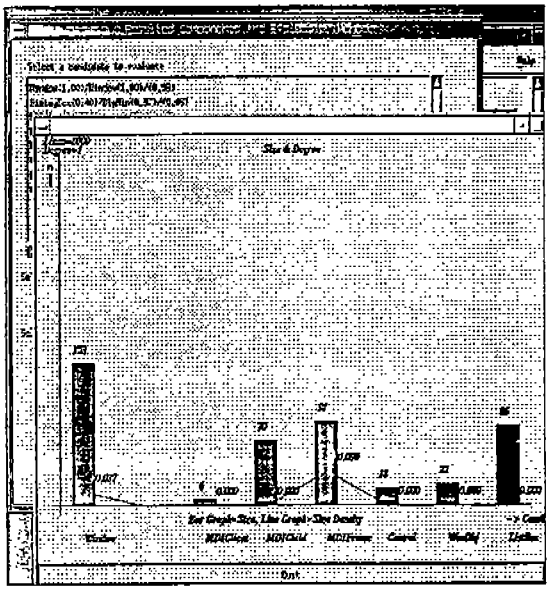
4.2 클래스 정보들의 사용 예

재사용 시스템의 질의 결과 유사한 클래스 후보자들이 여러개 나왔을 때, 재사용자는 이들 후보자들 가운데서 가장 적절한 후보자를 선택하기 위해서 (그림

5)와 (그림 8)의 명세서 정보와 품질정보들을 참조할 수 있다.

(그림 9)는 재사용자가 재사용 시스템[9]에 Window (0.10)/Window(0.17)//의 질의를 했을 때에 8개의 유사한 후보자들이 발생한 것을 나타낸다. 앞 화면은 클래스의 품질 정보들 가운데서 크기와 크기위반정도면[9]에서 7개의 후보자들을 비교한 것이며 1개 후보자는 정보에 관한 자료가 정보 DB에 없는 경우이다. 크기위반정도는 한 클래스의 멤버함수들의 크기위반정도를 계산한 것이다. 크기위반정도가 1이면, 크기면에서 가장 나쁜 후보자이다. (그림 9)에서 대부분의 후보자들의 크기는 다양하지만 실제 크기에 비해 크기위반정도가 낮으므로 크기면에서는 후보자들이 다 적절하는 것을 알 수 있었다.

(그림 9)와 같이 클래스의 품질 정보들을 이용하여 7개의 후보자들을 비교한 경과, Window 클래스가 가장 품질면에서 우수하였고 후보자들은 각각 품질 정보들의 종류에 따라 품질의 정도가 달랐다. 예를 들면, 노력도 정보에서는 Window 클래스와 MDIFrame 클래스가 가장 높았고, 상속도 정보에서는 8개의 후보자들이 모두 1.6배이상으로 높았다. 확장성 정보에서는 Window 클래스와 Control 클래스와 WinObj 클래스가 높았고, 문서화 정보면에서는 Window 클래스와 Control 클래스가 문서화 정도가 좋았다.



(그림 9) 크기에 의한 유사한 클래스 후보자들의 비교
(Fig. 9) Comparison of the similar classes candidates by size

이와같이 클래스에서 절차언어의 품질 정보들인 크기, 노력도, McCabe 순환수, 함수의 호출 위반 정도, 문서화 정도와 객체지향언어의 품질 정보들인 캡슐화 정도와 참조자 사용 정도, 상속도, 확장성 등의 정보들을 이용하여 각 후보자들 중에서 가장 적절한 후보자를 선정할 수 있었다.

4.3 기존 시스템들과의 비교

재사용 정보 분석 및 추출기와 기존 시스템들과의 비교는 지원 언어, 재사용 지원, 정보들의 종류, 정보 분석 및 추출기 등의 특성들을 기준으로하여 〈표 2〉와 같이 비교하였다.

코드 분석에 사용된 부품들은 대부분 절차언어로 구현된 원시코드들이었고, 정보들의 사용목적은 주로 재사용을 지원하는 부품의 분류와 검색, 평가, 부품의 정보 분석 등에 주로 사용되었다. 사용된 정보

〈표 2〉 기존 시스템들과의 비교
〈Table 2〉 Comparision of existing systems

	지원언어	재사용 지원	정보들의 종류				정보분석 및 추출 Viewer
			기능면 정보들	환경면 정보들	절차적 정보들	객체적 정보들	
제안한분석기	C++코드	O	O	O	O	O	O
Prieto 시스템	절차언어 기술서	O	O	O	O	×	×
RSL 시스템	ADA언어 설계서	O	O	O	O	×	×
Selby논문	Fortran 코드	O	O	O	O	×	×
CIA 시스템	C 코드	O	O	O	O	×	×
CIA++시스템	C++코드	O	O	O	O	△	×
Arnold시스템	C 코드	O	O	O	O	×	×
Code Miner	C 코드	O	O	O	O	×	×
RESOFT 시스템	C++코드	O	O	O	O	△	×
재사용 모듈시스템	쿄볼 언어	O	O	O	O	×	×
S/W분류 검색시스템	모든 언어	O	O	O	O	×	×

들은 크게 4가지의 종류로 나누어 비교하였다. 기능면의 정보들은 부품들의 이름, 종류, 구조 등이며, 환경면의 정보들은 프로그램의 수행 장소, 컴퓨터 기종, 저자, 구현언어, 키워드 등이다. 절차적 정보는 부품들의 평가에 사용된 부품의 크기, 복잡도, 모듈수, 문서화 정도, 가독성, 이식성, 코드의 표준화 정도, 함수의 호출 관계, 결합도 등이며, 객체적 정보는 부품들의 상속관계, 추상화 단계, 정보은닉정도 등의 정보들을 말한다.

〈표 2〉에서 CIA++ 시스템과 RESOFT 시스템이 스인 이유는 부품들의 객체적 정보들 중 부품의 상속관계 정보만을 계층적으로 보여주기 때문이다. 정보분석 및 추출 Viewer는 대부분의 시스템이 지원을 하지 않았다.

5. 결 론

본 연구는 C++ 언어의 재사용 가능한 클래스와 멤버함수들의 재사용 정보들을 자동으로 분석하고 추출하는 환경을 구축하기 위하여 그에 대한 시스템의 설계와 구현에 대해서 기술하였다. 특히 이 시스템은 기존 재사용 시스템들을 고려하였고 재사용자 인터페이스와 연결하여 재사용자가 분석 과정을 볼 수 있으며 분석기의 실행 시간을 측정할수 있도록 하였다. 또한 그래픽 환경을 기반으로 마우스와 브라우징 기술을 사용하여 재사용자가 쉽게 시스템에 액세스할 수 있도록 하였다.

앞으로의 연구 방향은 구현된 재사용 정보 분석기의 기능 확장과 정보 저장소를 구축하는 기술과 부품들의 분류 자동화 등이 주요 연구 과제이다.

참 고 문 헌

[1] R.Prieto and P.Freeman, "Classifying Software for Reusability," IEEE Software, Vol.4, No.1, Jan 1987, pp.6-16.

[2] B.A.Burton, R.W.Aragon and L.A. Mayes "The Reusable Software Library," IEEE Software, July 1987, pp.25-33.

[3] R.W.Selby, "Empirically Analyzing Software Reuse in a Production Environment," in Software Reuse: Emerging Technology, IEEE Computer Soc. Press. Los Alamitos, Calif., April 1988, pp. 176-189.

[4] Y.Chen, M.Y.Nishimoto and C.V.Ramamoorthy, "The C Information Abstraction System," IEEE Trans. on Software ENG. Vol.16, No.3, March 1990, pp.325-334.

[5] J.E.Grass and Y.F.Chen, "The C++ Information Abstractor," USENIX C++ Conference, 1990.

[6] R.Arnold and W.Frakes, "S/W Reuse and Reengineering," Final draft of paper that appeared in CASE Trends, 1991, pp.476-483.

[7] M.F.Dunn and J.C.Knight, "Automating the Detection of Reusable Parts in existing S/W," IEEE Proceedings of ICSE-15, 1993, pp.381-390.

[8] Jingwen Cheng, "A Reusability Based S/W Development Environment," IEEE ACM SIGSOFT Software Engineering, Notes, vol.19, No.2, 1994, pp.57-62.

[9] 김재생, 송영재, "재사용 가능한 클래스 후보자들의 품질 메트릭들에 관한 연구," 한국정보 처리학회지 제4권 제1호, Jan. 1997.

[10] Jaesaeng Kim and Youngjae Song, "Environment of the retrival system for C++ reusable components," Proceeding of the IASTED International Conference, July 1995.

[11] 변상용, "소프트웨어 설계 정보의 재사용에 관한 연구," 중앙대학교 대학원, 박사 학위 논문, 1990.

[12] 강문설, "재사용을 위한 소프트웨어 부품의 분류 방식 및 검색 모델," 전남 대학교 대학원, 박사 학위 논문, 1994.

[13] A. Porter, "C++ Programming for Windows," McGraw Hill, 1993.

[14] D.E.Brubaugh, "Object-Oriented Development," John Wiley & Sons, Inc., 1994.

[15] G.A.Miller, "The Magical Number Seven Plus or Minus Two: Some Limits on our Capacity to Process Information," Psychological Reviews, Vol.63, 1958, pp.81-86.

[16] 송영재, "C 언어로 구현한 소프트웨어 공학," 홍능 과학 출판사, 1991.

[17] 김형주, "객체 지향 시스템," 동아출판사, 1993, pp.229.

김 재 생

1988년 경희대학교 전자계산공학과(학사)
1990년 경희대학교 전자계산공학과(석사)
1995년 경희대학교 전자계산공학과 박사과정 수료
1993년~현재 경희대학교 전자계산공학과 시간강사
관심분야:소프트웨어공학, OOP, S/W 재사용

송 영 재

1969년 인하대학교 전기공학과(공학사)
1976년 일본 Keio University 전산학과(공학석사)
1979년 명지대학교 대학원 졸업(공학박사)
1971년~1973년 일본 Toyo Seiko 연구원
1982년~1983년 미국 Univ. of Maryland 전산학과 연구교수
1985년~1989년 IEEE Computer Society 한국지회 부회장
1984년~1989년 경희대학교 전자계산소장
1976년~현재 경희대학교 전자계산공학과 교수
1993년~1995년 경희대학교 교무처장
1996년~현재 경희대학교 공과대학장
관심분야:소프트웨어공학, OOP/S, CASE 도구, S/W 개발도구론, S/W 재사용