McCabe 및 BP/Win도구를 이용한 소프트웨어 역공학 사례연구

(Case Study of Software Reverse Engineering using McCabe and BP/Win Tools)

조 현 훈 † 최 용 락 †† 류 성 열 †††
(Hyun Hoon Cho)(Yong Lak Choi)(Sung Yul Rhew)

요 약 본 논문은 개발된 원시코드를 분석하여 단계별 효율적인 유지보수와 나아가 재공학 과정에서 재사용 가능한 모듈을 제공함으로써 생산성 있는 소프트웨어를 구축할 수 있는 지침을 제공하는 데 그 목적이 있다. 전체적으로 4개 흐름을 중심으로 첫번째는 개발된 원시코드를 분석하고, 두번째는 분석된 코드를 McCabe와 BP/Win도구를 이용하여 모듈 슬라이싱 및 클러스터링을 하고, 세번째는 비즈니스 모델링에서 추출된 산출물을 재사용 모듈로 변환하여 생성한 후, 마지막 네번째는 리포지토리 설계 및 시스템 구축으로 구성한다. 논문에서 제시하는 부분은 1~3번째를 세분화한 4단계 방법을 사례연구로 적용했으며, 세분화한 4단계는 리포지토리를 구축하는 데 필요한 제반사항을 포함하고 있다. 세분화한 4단계는 기존의 비 정형화되고, 비 구조화된 자료를 역공학 도구를 이용하여 재 분석함으로써 생산성 있는 소프트웨어 유지보수와 재공학에 효과적으로 지침을 제공한다.

Abstract This paper aims at providing guidelines enabling productive software construction by offering reusable modules which is used not only for effective maintenance for each step, but also for a re-engineering process after analyzing developed source code. There are four processing steps. The first is to analyze source code. The second is module slicing and clustering using McCabe and BP/Win Tools. The third is to transform the outputs extracted from the business model to reusable modules. The final step is to design repository and to construct a system. In this paper, we applied the fourth step to our case study, which was specified from the first step to the fourth. The specified fourth step contains various things for constructing repository. And the fourth step reanalyzes informal and unstructured information by using reverse engineering tools, in order to provide effective guidelines for productive software maintenance and re-engineering.

1. 서 론

현재 소프트웨어 산업계에서는 새로운 응용 시스템 개발에 집중을 하면서 품질 향상과 시장 출하 시간 단축을 통한 사용자의 욕구 충족 및 새롭고 효율적인 개발 프로세스를 만드는데 치중하고 있다. 반면에 개발된 소프트웨어에 대해서는 아직까지 체계적인 유지보수가 미흡하고 이에 따른 유지보수 방법론의 필요성이 절실히 대두되고 있다. 기업들은 새롭고 다양한 요구 사항들을 시스템에 반영하기를 원하며 이를 위한 정확한 분석 작업을 필요로 한다. 특히, 비 구조화되고 비 문서화된 노후 시스템을 새로운 시스템으로 교체 작업 또는 수작업으로 유지보수를 한다는 것은 매우 어려운 일이다. [4, 7] 이러한 소프트웨어의 유지 보수성을 향상 시키는 방법 중 하나는 역공학 기법을 통해서 가능하다.

본 논문에서는 이러한 필요성을 충족하고 문제점을 해결하기 위한 방안으로 시스템의 원시코드 분석을 위한 역공학 4단계 처리 방법을 제시하고 사례연구를 통

· 본 논문은 1998년 12월부터 1999년 3월까지 대신정보통신㈜의 한우리/S 리버스 엔지니어링 과제의 연구비 지원에 의한 연구 산출물임.
† 학생회원 : 숭실대학교 컴퓨터학부
hhcho@selab.soongsil.ac.kr
†† 비 회 원 : 숭실대학교 전자계산학과
ylch@hitel.net
††† 종신회원 : 숭실대학교 컴퓨터학부 교수
syrhew@computing.soongsil.ac.kr
논문접수 : 2000년 2월 14일
심사완료 : 2000년 8월 10일

해 소프트웨어 유지보수에 효과적으로 적용될 수 있음을 보였다.

2. 소프트웨어 역공학의 이론적 고찰

2.1 역공학의 이해

소프트웨어 역공학이란 자동화된 도구를 이용하여 소프트웨어의 물리적 계층 표현(예, 원시코드)을 소프트웨어 데이타 컴포넌트와 프로세스를 설명해 주는 명세 계층 표현으로 재발견하거나 재구축하는 과정을 말한다. 역공학을 통해서 얻을 수 있는 이점은 다음과 같다. [1, 5, 6, 8]

(1) 현재 시스템의 데이타와 논리(logic)를 분석하는 효율적인 방법을 제공함으로서 효율적인 유지보수와 처리를 용이하게 하며, 논리와 데이타 설계, 그리고 함수를 재사용할 수 있다.

(2) 새로운 환경에 데이타와 논리가 순공학에 사용될 수 있으므로 시스템의 변환(conversion)과 이식(migration)을 보다 쉽게 할 수 있다.

이러한 이점에 반하여, 소프트웨어 역공학에 나타나는 문제점으로는 다음과 같다.

(1) 노후 시스템의 경우, 하위 수준의 원시코드에서 상위 수준의 소프트웨어 구조를 파악하는 데 많은 시간이 걸리기 때문에 유지보수 단계에 많은 비용과 시간이 소요된다. [10]

(2) 구조화된 원시코드 부재와 불완전하고 미비한 설계 명세서, 존재하지 않는 낙후된 문서, 높은 중복성, 함수들의 복잡성 등의 바람직하지 않는 특성들이 나타나게 된다.

(3) 소프트웨어 자체의 이해도와 신뢰도를 낮추고, 소프트웨어 유지보수를 어렵게 한다.

이러한 문제점을 본 논문에서는 역공학 도구와 순공학 도구의 유기적인 사용을 통해 해결하고자 한다.

2.2 역공학 관련 기술

역공학은 현재 소프트웨어 부분에서 각광을 받고 있으며 프로그램 이해와 소프트웨어 유지보수, 그리고 재공학의 기술에 필수적인 활동으로 간주되고 있다. 다음 표1은 역공학에 관련된 기술을 영역 별로 정리하여 보여 주고 있다.

표 1 역공학 기술[9, 11]

기 법	영 역
문자적분석	라인을 구성하고 프로그램의 크기를 가늠하는 척도인 기호 추출
어휘분석	프로그램의 부호 배열을 프로그램의 성분 어휘구 단위로 분할
구문분석	어휘분석으로부터 추출된 토큰을 문장, 프로그램, 모듈로 정의
제어흐름분석	내부 절차 분석 및 상호간 절차 분석
데이타흐름분석	데이타의 정의와 참조에 의한 분석
프로그램 종속도형	데이타 흐름 분석 정제
조각화(Slicing)	자료 및 제어흐름을 분석하여 프로그램 행위를 관심있는 부분으로 분할
관용구 인식	일반 프로그래밍 패턴들에 대한 프로그램 문장 검색
추상해석	프로그램의 의미적 성질들을 기호 의미론인 수학적 기법으로 표현
동적분석	프로그램의 실행을 통한 실행성과 정확성 이해
부분평가	프로그램과 그 프로그램의 입력 인자를 받아 가능한 최대의 프로그램 수행을 유도하고 그 결과 출력

2.3 역공학 도구의 비교

2.3.1 McCabe Tool

역공학 도구들은 유지보수 행위동안 복잡한 소프트웨어 시스템들을 분석하고 이해하는 과정에 있어서 소프트웨어 엔지니어들을 지원한다. 그러한 도구들의 기능은 편집 및 브라우징(browsing) 기능들로부터 텍스트 및 그래픽 보고서들까지 다양하다. 역공학을 지원하는 제품으로는 표2에서 제공하는 여러 가지가 있으며, 이들은 특정 원시 코드의 언어들을 지원할 뿐 만 아니라, 서로 다른 기능을 제공한다. 그리고 역공학 도구들이 지원하는 기본 기능에서부터 핵심이 되는 기능을 중심으로 McCabe도구와 함께 비교 분석한 결과는 표2와 같다. 평가의 판단 기준은 4가지 형태의 표현으로 제공된다. [2, 3]

2.3.2 BP/Win

BP/Win은 구조적 방법론에 기반을 둔 업무 분석용 도구로써 IDEF(Integration DEFinition)방법 체계를 따른다. BP/Win이 제공하는 업무 분석 방법은 IDEF0(Function Modeling), IDEF1(Information Modeling), IDEF3(Work Flow Diagram/Process Description Capture), IDEF4(Object Oriented Design) 및 DFD(Data Flow Diagram)이다. 도구를 통해 나타나는 산출물들은 서로 호환하여 이용할 수 있으며, 다양한 형태의 분석 자료를 제공함은 물론 다른 CASE Tool과의 상호 호환 연결 사용도 가능하다. IDEF방법론은 기업, 조직의 실체를 추상화하여 모델링하고, 작성된 모델의 체계적인 분석을 통하여 문제점을

표 2 역공학 도구 비교 평가[3]

평가수치항목	McCabe	Logiscope	HindSight	Ensemble	Refine/C	Rigi
일반적인 수행 능력	+	O	O	O	O	O
지원 환경	Sun Sparc HP Digital AIX IRIX DGUX WindowsNT/98/95/3.x/OS2	Sun Sparc HP Digital Windows NT/95/3.x	UNIX	UNIX	Sun Sparc HP 9000/7xx IBM RS/6000	Sun Sparc IBM RS/6000
다중 사용자 지원	O	-	-	O	O	-
Toolset 확장성	+	-	-	+	+	+
저장 수행능력	++	+	+	+	O	O
산출물 수행능력(Printing)	+	+	++	+	+	-
산출물 수행능력(Exporting)	+	-	-	+	-	-
산출물 수행능력(Documentation)	++	+	++	+	-	-
분석	++	+	+	++	++	O
지원 언어	C(ANSI) C++ COBOL Visual BASIC etc.	C(ANSI) C++ ADA COBOL etc.	C(ANSI) C++ Fortran	C(ANSI) C++ ADA	C(ANSI) C(K&R)	C(ANSI) C++ COBOL PL/AS Latex
증량적 분석	-	-	-	+	-	-
재분석	+	+	+	+	-	+
내 증량성 분석기	+	+	+	+	-	-
분석 결과	++	+	+	+	++	O
표현	+	+	+	+	+	+
문서(통계적 표, 형식 등)	+	-	+	+	+	-
그래픽(2-, 3-차원)	+	-	-	+	O	O
일반적인 레포트 속성	+	+	+	+	+	+
문서 레포트 속성	+	+	+	+	+	-
그래픽 레포트 속성	+	+	+	+	O	++
Editing/Browser	-	-	-	+	-	-
통합문서 editor/browser	++	-	-	+	-	++
외부 editor/browser	+	-	-	+	+	+
문서 editor/browser의 조절	+	-	O	+	O	O
강조되는 부분	+	O	O	+	-	-
시각화 기능	+	O	O	+	-	-
문서 editor 속도	+	O	O	+	-	-
탐색기능	++	+	O	+	-	-

++: 매우 좋음 +: 좋음 O: 보통 -: 없음

추출하여 개선된 기업의 모델을 설계할 수 있도록 개발된 시스템 분석, 설계 방법론이다. 또한 IDEF는 시스템의 개발과 관련된 사람들간의 의사소통을 촉진하기 위한 언어로 개발되었으며 다음과 같은 목적으로 사용되고 있다.

(1) 시스템 분석, 설계, 교육, 문서화, 통합(2) 컨센서스(Consensus)를 위한 의사 소통 수단 지원(3) 기업의 정보 시스템 구축을 위한 업무 활동의 분석과 문제점 포착(4) 기업의 활동에 관한 업무 흐름의 명확한 표현

3. 원시코드 분석을 위한 4단계 방법

3.1 McCabe와BP/win도구를 이용한 4단계 역공학 세스

본 논문에서 제공하는 역공학 프로세스는 이미 개발된 원시코드 입력물을 받아 우선 수작업을 통한 기능 측면에서 모듈을 분할하고 CASE도구를 이용한 여러

가지 산출물을 재사용 가능한 모듈로 생성하여 리포지토리에 저장함으로써 향후 개발하고자 하는 관련 업무에서 효율적으로 검색하여 사용할 수 있도록 하는 데 주 목적이 있다. 그림1은 전반적인 프로세스 흐름도로써 본 논문에서 실제 사례로 보여주는 부분은 3단계를 포함한 리포지토리에 저장하기 위한 준비 단계까지이며, 분석된 재사용 가능한 모듈과 업무분석 산출물들을 실제로 저장하는 리포지토리 설계 및 구축은 향후 확장된 부분에서 제공하고자 한다.

기존의 원시코드를 입력으로 받아들여 추상화된 개념을 도출해 내는 역공학 프로세스와 추상화된 개념에서 순공학을 통해 새로운 산출물을 생산할 수 있는 프로세스 모델을 그림1에서 1~3단계 부분을 4단계로 세분화하여 그림2에서 태스크 네트워크로 보여주고 있다.

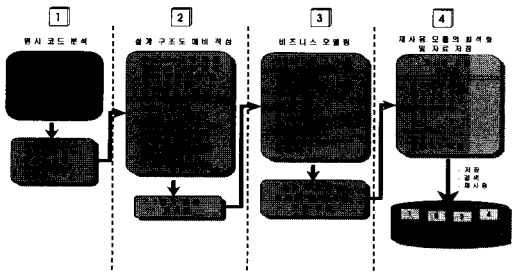

그림 1 프로세스 흐름도

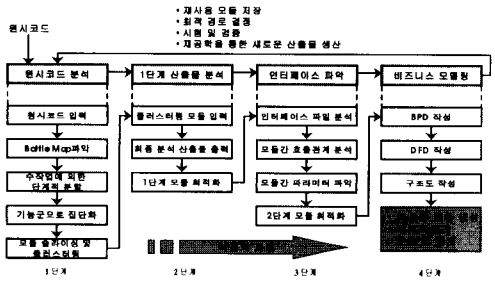

그림 2 역공학 프로세스 태스크 네트워크

3.2 프로세스 단계별 의미

1단계 : 원시코드 분석

고객으로부터 받은 많은 양의 소스 파일들이 어떠한 기능과 의미가 있는지를 McCabe도구를 이용하여 분석하고 수작업을 통해 단계적으로 분할한다. 이 때, 원시 파일들은 기능군으로 집단화한다.

2 단계 : 1단계 산출물 분석

기능군으로 클러스터링된 입력 원시코드들을 다시 분석하여 사용자가 필요로 하는 분석 산출물들을 출력하는 단계이다.

3 단계 : 모듈 인터페이스 파악

분석 산출물들을 기반으로 모듈간의 데이타 입,출력 관계와 제어흐름 인터페이스 관계를 분석, 파악 하는 단계이다.

4 단계 : 비즈니스 모델링

2,3단계의 분석 산출물들을 기반으로 *BP/win*도구를 이용하여 업무분석에 필요한 산출물들을 출력하는 단계이다. 앞 단계의 자동화된 산출물만으로는 시스템의 전체 기능들을 구체적으로 파악하기 어렵기 때문에 앞 단계의 출력물과 원시코드를 검토하여 수동으로 작성한다. 4단계까지 이루어진 후에는 재사용 가능한 모듈 추출 및 생성단계를 거쳐 리포지토리에 저장할 수 있는 구조로 모듈을 변환하게 된다.

3.3 프로세스 태스크별 의미

1단계 : 원시코드 분석

(1) 원시코드 입력

고객으로부터 받은 원시코드를 *McCabe*도구를 이용하여 입력하는 단계이다.

(2) *Battle Map*파악

원시코드 입력을 통해 출력되는 *Battle Map(구조도)*을 보고 전체 구조를 파악하는 단계이다.

(3) 수작업에 의한 단계적 분할

원시코드의 비구조적인 복잡한 프로그램을 수작업을 통해 구조적인 프로그램으로 재구조화하며 자동화 구축 도구를 사용하여 자동적으로 수행하는 단계이다.

(4) 기능군으로 집단화

모듈의 기능들을 분석하기 위해 모듈의 이름을 소프트웨어 기능과 연관해서 유추해 보고 소스의 주석들을 기반으로 파일들을 정리, 분석해서 원시코드 분석 테이블을 작성하는 단계이다. 분석테이블은 파일이름, 파일내용, 파일사이즈 3개의 필드로 구성된다.

표 3 파일 분석 표 구성

파일 이름	파일 내용	파일 크기
File 명	파일의 주석문을 중심으로 가급적 상세히 파일의 목적과 기능을정리한다.	(Mbyte)

(5) McCabe방법을 이용한 모듈 복잡도 시험

슬라이싱을 위하여 재구성된 원시 코드들을 *McCabe* 도구를 이용하여 모듈별로 시험하고 시험한 결과 기준으로 하위 모듈로 나눈다. 테스팅의 기준은 구조적 구조

를 모두 제거한 *Essential Complexity Metric*을 기준으로 *Cyclomatic Complexity*를 줄일 수 있도록 서브 모듈화 한다. 복잡도가 10보다 큰 경우 *subfunction*을 제거하고 대신 *subroutine*으로 재구성한다. *Subgraph*를 제거하고 *subroutine*으로 재 구성하는 경우, 시스템의 전체 복잡도가 낮아지는 것이 아니라 그림3과 같이 시스템이 좀 더 관리 가능한 형태로 되는 것이다.

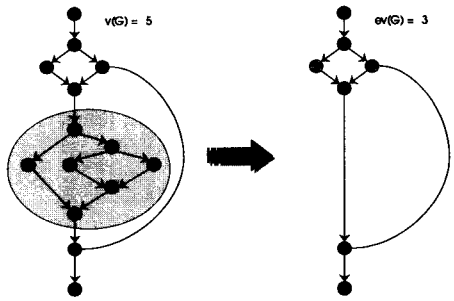

그림 3 *Cyclomatic Graph(v(G))*와 *Essential Graph (ev(G))* 비교

(6) 모듈 슬라이싱 및 클러스터링

소프트웨어 전체기능을 이해하기 쉽도록 하기 위해 하위단계의 추상화된 원시코드들을 분석된 결과를 기반으로 유사기능의 모듈별로 원시코드들을 클러스터링하는 상위단계 추상화를 유도하는 단계이다.

표 4 추상화 표 구성

그룹명	구성화일
유사 모듈들을 추상화한 그룹명	유사 파일들로 구성

2단계 : 1단계 산출물 분석

(1) 클러스터링 모듈 입력

1단계의 최종 클러스터링된 모듈들을 *McCabe*도구에 입력하는 단계이다.

(2) 최종 분석 산출물 출력

고객이 유지보수에 필요로 하는 여러 분석 산출물들을 출력하는 단계이다.

(3) 1단계 모듈 최적화

기존에 개발된 원시코드를 수작업을 통한 단계적 분할 및 기능군 집단화를 반복적으로 함으로써 최적 모듈을 구성하는 단계이다.

3단계 : 모듈 인터페이스 파악

(1) 인터페이스 파일 분석

*Battle Map*을 기반으로 모듈간 인터페이스 관계를 분석하는 단계이다.

(2) 모듈간 호출관계 분석

모듈간의 *Calling / Called*관계를 분석하는 단계이다.

(3) 모듈간 파라미터 파악

분석 산출물 *Code Breaker*를 이용하여 모듈간 파라미터 이동 관계를 파악하는 단계이다.

(4) 2단계 모듈 최적화

1차 분석된 산출물을 기반으로 모듈간의 흐름과 인터페이스 관계를 반복적인 검토를 통해 정제하는 단계이다.

4단계 : 업무분석 산출물 출력

(1) *BPD(Business Process Diagram)*작성

3단계의 인터페이스 분석을 기반으로 *BPD*를 작성하는 단계이다.

(2) *DFD(Data Flow Diagram)* 작성

작성된 *BPD*를 기반으로 *DFD*를 작성하는 단계이다.

(3) 구조도*(Structure Chart)* 작성

*BPD*와 *DFD*를 작성하고 나면 자동으로 구조도를 출력해 주는 단계이다.

4. 사례 연구

고객으로부터 받은 *60,000*라인에 해당하는 39개의 원시 파일들을 표5와 같은 환경에서 원시코드를 분석하고자 한다. 원시코드는 데이타 베이스의 자료를 관리 하는 하부 커널이다.

표 5 실무적용 환경

분석환경	분석 도구	원시 코드
Windows 95/98	McCabe, BP/Win 도구	60,000라인의 C 원시코드

4.1 단계별 적용

1단계 : 원시코드 분석

(1) 6만 라인의 원시코드를 McCabe도구 입력물로 지정한 후, 전체적인 구조도를 분석한다. 그리고 수작업에 의해 비구조화된 모듈을 분할 및 기능별로 집단화한다.

(2) McCabe방법을 이용한 모듈 복잡도 시험: v(G)와 ev(G)를 분석한 후, ev(G)를 기준으로 하위 모듈을 분할한다.

(3) 모듈 슬라이싱 및 클러스터링: 주어진 원시코드에 대한 분석을 통해 각 기능별로 재사용 가능한 모듈로 클러스터링한다. 표6은 주어진 원시코드에 대한 기능을 분석한 것이다. 모듈을 주어진 원시코드에서 그 기능 별로 잘라내서 추상화 레벨에서 클러스터링을 하여 표8과 같이 집단화 할 수 있다.

표 6 파일 분석

파일 이름	파일 내용	파일 크기
BLOB.C	Binary Large Object의 약자로 데이타베이스내에서 상당한 양의 Binary 데이타를 처리하기 위한 파일	12Mbyte
BTREE.C	데이타베이스내에서 자료의 추가, 삭제등을 수행하기 위해 B+트리 구조의 인덱스를 관리하는 파일	20Mbyte
SHM.C	데이타베이스 엔진에 의해서 공유되는 시스템 테이블을 관리한다.	8Mbyte
LK.C	다수의 트랜잭션 공유자료 접근을 순서화하는 동시성 제어를 관리한다.	6Mbyte
INTERFACE.C	사용자 인터페이스명을 정의 한다.	7Mbyte

표 7 v(G)와 ev(G)의 비교

파일 이름	Cyclomatic Complexity	Essential Complexity	파일 크기
BLOB.C	110	70	12Mbyte
DISK.C	60	40	17Mbyte
CLOB.C	120	60	14Mbyte
BF.C	100	55	16Mbyte
BTREE.C	135	75	20Mbyte
BTREE1.C	75	75	10Mbyte
INTERFACE.C	16	16	7Mbyte

표 8 그룹별 추상화

그룹명	구성 파일
BLOB 관리기	BLOB.C, BL_BUF.C, BL_DATA.C, BL_RECOVER.C
B+인덱스 관리기	BTREE.C, BT_SORT.C, BT_LOG.C, BT_MISC.C
정보검색 인덱스 관리기	IR_CURSOR.C IR_RECOVER.C, IR_SORT.C
디스크 관리기	DISK.C
트랜잭션 관리기	TRANS.C
잠금 관리기	LK.C, LK_LOCK.C
회복 관리기	LG.C, LG_log.C
파일 관리기	FILE.C

2단계 : 소스분석 산출물 출력

1단계의 산출물을 입력으로 하여 현 단계에서는 CASE도구를 사용한 각 태스크 실행화면을 보여주고 있다.

(1) 클러스터링 모듈 입력

그림4는 미리 컴파일된(pre-compiled) 원시코드를 모듈별로 클러스터링하여 입력하는 부분으로 분석의 기초적인 자료가 되는 항목이다.

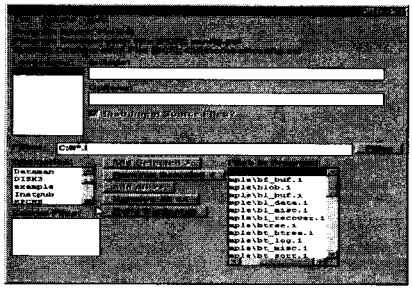

그림 4 모듈 입력 상자

(2) 최종분석 산출물 출력

그림5는 그림4를 통해 입력된 기초 자료를 바탕으로 전체 구조를 쉽게 이해할 수 있도록 시각적인 인터페이스로 모듈별 관계를 구조도의 형태로 나타낸 것이다.

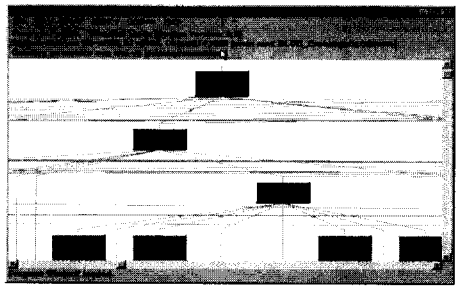

그림 5 역공학 구조도 (Battlemap)

3단계 : 모듈 인터페이스 파악

(1) 인터페이스 파일 분석

그림6은 분석된 프로그램의 각 모듈 메트릭에 대한 요약 정보를 나타내며, 코드의 품질을 구체적으로 제시한 것으로 향후 분석 결과를 정형화하는 데 영향을 미친다.

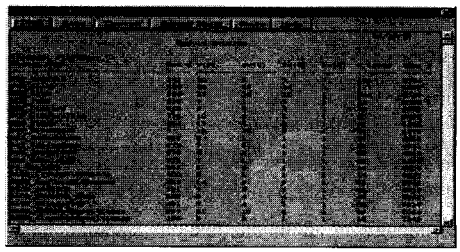

그림 6 모듈간의 메트릭 분석 산출물

(2) 모듈간 호출관계 분석

그림 7은 분석된 전체 구조에서 각 모듈별 호출관계를 팝업 메뉴를 통해 제공함으로써, 프로그램의 이해를 높인다.

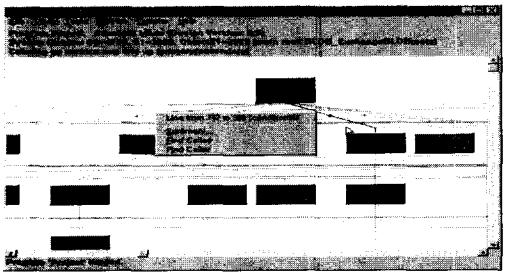

그림 7 모듈간 호출관계

(3) 모듈간 파라미터 분석

그림7에서 볼 수 있는 호출관계를 세분화하여 그림8에서 모듈 이름, 모듈간 관계의 형태, 사용되는 변수를 요약 정리하여 제공한다.

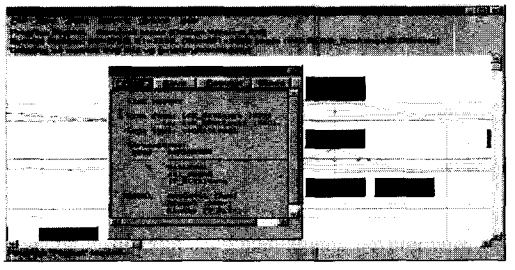

그림 8 모듈간 입·출력 파라미터

4단계 : 업무분석 산출물 출력

(1) BPD 작성

그림9는 분석 대상 모델에 대한 범위, 관점, 목적과 같은 전반적인 내용을 개략적으로 나타내고, 이를 바탕으로 기능적으로 세분화하여 분해한 분해도이다.

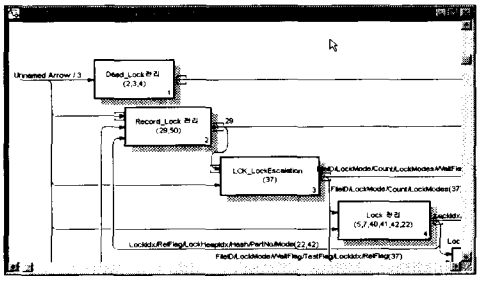

그림 9 BPD(Business Process Diagram)

(2) DFD 작성

그림10은 분석 자료를 저장하는 저장소와 기능적 업무처리의 파이프라인으로 연결한 네트워크로써 시스템을 단순하면서 간단하게 표현하여 이해도를 높일 수 있다.

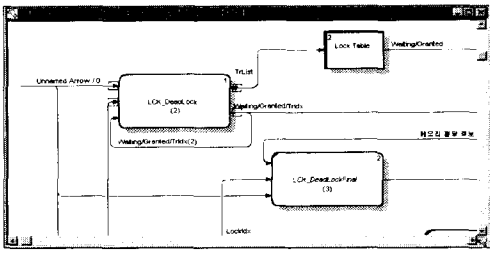

그림 10 DFD(Data Flow Diagram)

(3) 전체 업무 구조도

그림9에 표현한 다이어그램은 시스템의 분석대상을 개략적인 부분에서 상세화해 보여주는 것이며, 그림11에서는 기능 모델에 대한 구조를 계층적으로 표현하여 전체적인 모듈 기능 관계를 표현해 준다.

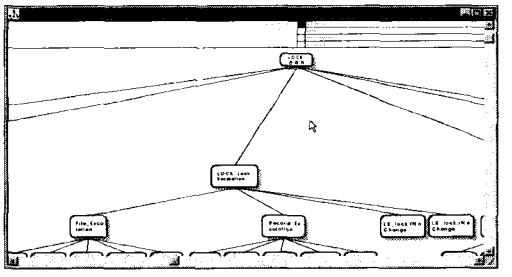

그림 11 Node Tree (Structure Chart)

4.2 평가

사례 연구를 통해 본 연구 결과에 대한 평가를 각 단계별로 정리해 보면 다음과 같다.

세부1단계에서는 고객과의 요구 분석 단계에서 발생됐던 문제점으로 원하는 주요 요구사항을 구두로 전달받았기 때문에 개발된 시스템에 대한 설명이 미비하여 프로젝트 수행 중에 수정 보완해야 하는 문제와 정확한 원시코드가 전달되지 않아서 재작업을 해야 하는 문제가 발생하였다. 그리고 개발된 원시코드 크기가 대규모이기 때문에 전체적인 구조를 수작업을 통해 단계적으로 분할해야 하는 문제점을 보여주었다. 이러한 문제점을 해결하는 과정에서 기존 개발에 대한 비정형화된 개발 체계를 발견할 수 있었다.

세부2단계에서는 고객으로부터 받은 원시코드를 역공학 도구의 입력값으로 여과 없이 적용하였을 경우 산출

물들이 너무 복잡하고 방대해 지므로 유지보수에 실질적으로 적용할 수 있는 기능단위 분할 및 클러스터링의 필요성이 나타나게 되었다. 이 과정을 통해 비구조화된 부분들의 모듈화가 어느 정도 이루어졌으며, 프로그램의 이해도를 높일 수 있었다.

세부3단계에서는 업무분석 산출물을 출력하기 위해서 원시코드 분석 단계의 산출물들이 자동적으로 연동되지 않기 때문에 인터페이스와 앞 단계의 산출물들을 파악해서 출력함으로써 추상화 단계의 수준을 높일 수 있었다.

세부4단계에서는 개발된 시스템이 추구하는 목적에 맞는 비즈니스 분석을 위한 접근 방법으로 분석 대상 시스템을 이해함으로써 비지니스 모델링이 가능했으며, 제안한 프로세스는 향후 유사한 소프트웨어의 개발에 있어서 효율적인 지침을 제공해 줄 수 있다. 이 외에 나타난 외적인 환경에서 나타나는 제약 사항으로는 효율적인 방법론과 도구의 선택에 있어서의 어려움과 선택한 도구에 대한 참여 연구원의 사용 미숙에 따른 시간 제약성이 있었다.

5. 결론 및 향후 연구과제

본 논문에서는 소프트웨어 유지보수에 효율적으로 적용할 수 있는 역공학 처리를 위한 정형화된 세부4단계 방법을 제시하고 사례 연구를 통해 나타나는 여러 가지 문제점을 정리함으로써 소프트웨어 유지보수와 재공학을 통한 새로운 시스템 개발 시 효과적인 지침으로 제공될 수 있음을 보였다. 향후 연구방향은 재공학 분야와 연계하여 분석된 산출물과 재사용 가능한 모듈을 생성하여 저장하는 리포지토리의 설계가 이루어질 것이며, 저장된 자료를 이용하여 재공학을 통한 새로운 시스템 개발에 도움이 되도록 설계 지침을 제공할 것이다. 또한, 분석 단계 중 원시 코드를 기능군으로 분할하여 시스템 유지보수에 적절하게 적용할 효과적인 모듈 클러스터링 방법에 대한 연구를 세부적으로 병행하여 진행할 것이다.

참 고 문 헌

[1] Rene R.Klsch, *Reverse Engineering: Why and How to Reverse Engineer Software*, Proceedings of the California Software Symposium, 1996.

[2] Berndt Bellay and Harald Gall, *An Evaluation of Reverse Engineering Tools*, Technique University of Vienna Information Systems Institute Distributed Systems Department, March 13, 1997.

[3] Berndt Bellay and Harald Gall, *A Comparison of four Reverse Engineering Tools*, Technique University of Vienna Information Systems Institute Distributed Systems Department, September 19, 1997.

[4] Harald Gall and Rene Klsh, *Balancing in Reverse Engineering and in Object-Oriented Systems Engineering to Improve Reusability and Maintainability*, Vienna University of Technology Institute of Information Systems, 1994.

[5] Scott Tilley, *A Reverse-Engineering Environment Framework*, Carnegie Mellon Software Engineering Institute, April 1998.

[6] Hausi A. Mller, *Reverse Engineering Strategies for Software Migration*, ACM, 1997.

[7] Yih-Farn Chen, Michael Y. Nishimoto, and C. V. Ramamoorthy, The C Information Abstraction System, IEEE, Vol.16, No. 3, March 1990.

[8] Elisa L. A. Baniassad and Gail C.Murphy, *Conceptual Module Querying for Software Reengineering*, IEEE, 1998.

[9] Mary Jean Harrold and Ning Ci, *Reuse-Driven Interprocedural Slicing*, IEEE, 1998.

[10] Nicolas Anquetil and Timothy Lethbridge, *Extraction Concepts from File Names; a New File Clustering Criterion*, IEEE, 1998.

[11] 권오천, 신규상, 역공학 및 재공학의 기술 동향, 한국정보과학회, 소프트웨어공학회지 pp6-21, March 1999.

조 현 훈

1995년 광주대학교 전자계산학과 학사(공학사). 1997년 숭실대학교 컴퓨터학과 석사(공학석사). 1997년 ~ 현재 숭실대학교 컴퓨터학과 박사과정. 관심분야는 소프트웨어 개발 방법론, 웹 어플리케이션 개발, 소프트웨어 유지보수, 소프트웨어 재사용, 리엔지니어링, 분산 객체 컴퓨팅 등.

최 용 락

1985년 숭실대학교 전자계산학과 학사(공학사). 1996년 숭실대학교 정보과학대학원 정보산업학과 석사(공학석사). 1997년 ~ 현재 숭실대학교 컴퓨터학과 박사과정. 관심분야는 데이타베이스 모델링, 소프트웨어공학, 정보공학, ERP 등.

류 성 열

1997년 2월 아주대학교 컴퓨터학부(공학박사). 1997년 3월 ~ 1998년 3월 George Mason University 교환교수. 1981년 3월 ~ 현재 숭실대학교 컴퓨터학부 교수. 1998년 3월 ~ 현재 숭실대학교 정보과학대학원 원장. 1998년 3월 ~ 현재 숭실대학교 전자계산원 원장. 관심분야는 리엔지니어링, 분산 객체 컴퓨팅, 소프트웨어 재사용.