# 소프트웨어 유지보수를 위한 형상변경통제절차

최 광 준[†] · 김 기 봉[††] · 진 성 일[†††]


## 요 약

현재까지 복잡하고 대규모인 업무체계를 가진 조직체들은 통합정보관리체계를 구축하기 위해 업무시스템 개발을 추진하여 왔다. 그러나 조직체내의 사용부서마다 개별적으로 업무전산화계획을 수립하여 개발을 추진해 온 결과 업무시스템간의 호환성을 확보하지 못하여 통합시스템을 구축하는 데 커다란 장애가 되고 있다. 이러한 시스템간의 호환성과 통합성을 이루기 위해서는 소프트웨어시스템의 생명주기 전반에 걸쳐 형상관리를 실시해야 한다. 효율적인 형상관리를 위해서는 생명주기 단계들에서 생성되는 문서와 자료, 프로그램코드 등의 형상물들을 일관성 있게 저장하고 상호 공유하도록 해주는 역할을 효과적으로 지원하는 모델과 이를 지원할 수 있는 통합 정보저장소가 필요하다. 본 논문에서는 소프트웨어 형상관리의 가장 중요한 부분인 형상변경통제활동을 생명주기단계간 상호참조와 자료공유에 중점을 둔 모델링을 제시하고, IRDS를 이용하기 위한 ER모델 및 IRD스키마를 정의하였다.


# A Configuration Change Control Procedure for Software Maintenance

Kwang J. Choi[†] · Ki. B. Kim[††] · Seong. I. Jin[†††]


## ABSTRACT

Many large-scale enterprises have developed software systems to build management information system for their business. However, it is hard to maintain shareability of data, compatibility of operation methodology, and interoperability among subsystems when the development project progresses since each distributed team prepares a development plan for its subsystem and must have poor communication problem among those teams. We must apply software configuration management to the whole life cycle of the software system in order to solve those problems. We need a model and a repository in order to execute configuration management on configuration products like document, data, and source code which are generated through the life cycle of software development. In this paper, we suggest a model for configuration change control where cross-referencing among life cycle phases and data-sharing are highlighted, and define ER model to use IRDS(Information Resource Dictionary System) and IRD schemas.


## 1. 서 론

기업체나 대규모 조직들은 효율적인 통합정보관리체계 구축을 목표로 각 업무기능별 데이터베이스 설치와 응용프로그램 개발을 추진해 왔다. 그러나 사용부서마다 독립적으로 개발을 추진해온 결과 업무개발환경과 운영환경 등이 상이하여 시스템간의 호환성을 확보하지 못하고 통합시스템을 구축하는 데 어려움을

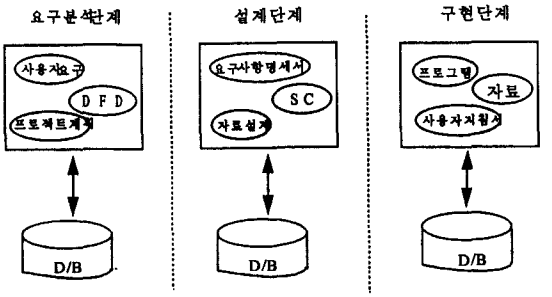
† 정 회 원:육군전산소
†† 정 회 원:대전보건전문대학 정보처리학과
††† 종신회원:충남대학교 컴퓨터과학과
논문접수:1997년 6월 16일, 심사완료:1997년 10월 12일

겪고 있다. 이러한 시스템개발 과정과 개발후의 유지보수과정에서 시스템간의 호환성과 통합성을 이루기 위해서는 소프트웨어 형상관리가 필수적이다[12, 14].

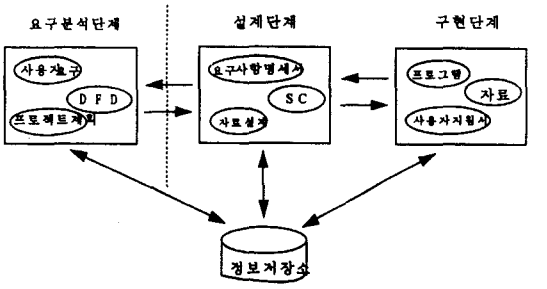
현재까지의 형상관리 도구들은 화일 또는 데이타베이스 시스템을 기반으로 개발되어 왔다[10, 15]. 그러나 이러한 형상관리 도구들은 생명주기 단계별로 특정 형상관리기능을 구현했으며 단계간 정보자원 공유나 상호참조 등의 통합 정보자원 관리에는 그 기능을 제공하지 못하는 제한점을 가지고 있다.

소프트웨어개발 생명주기단계는 크게 계획, 사용자요구분석, 설계, 구현, 그리고 운영유지 등 5단계로 나눌 수 있다. 본 논문에서는 사용자요구분석단계, 설계단계, 그리고 구현단계 등 3개 단계에서의 자료저장 및 접근 방식을 나타내었다. 요구분석단계에서는 시스템에 대한 사용자들의 요구를 수렴하고 분석하여 요구사항명세서와 자료흐름도를 작성한다. 설계단계에서는 시스템의 구조와 절차 등을 설계하며 구조도를 산물로 생산한다. 구현단계에서는 설계단계의 구조도를 기반으로 프로그램을 생산한다[6].

소프트웨어개발 생명주기에 있어서 사용자요구분석단계 및 설계단계 별로 사용자요구서나 자료구조설계서 등이 화일이나 수작업문서로 저장되어 있다. 이러한 구조가 (그림 1)에 표시되어 있다. 이러한 체계에서는 여러 팀의 개발자들이 자료를 공동으로 이용하기 어려운 단점이 있다. 이러한 문제점을 해결하기 위해서는 생명주기의 단계간에 형상자료를 일관성 있게 저장하고 공유하도록 해주는 기능을 지원하는 통합 정보저장소가 필요하다. 이러한 개선된 개념이 (그림 2)에 나타나 있다.




(그림 1) 기존 형상관리의 자료관리방식
(Fig. 1) Existing information management method in configuration management




(그림 2) 개선된 형상관리의 자료공유방식
(Fig. 2) Improved information sharing method in configuration management

본 논문에서는 소프트웨어 형상관리 네 가지 활동인 형상인식, 형상변경통제, 형상상태회계, 형상감사 중에서 가장 중요한 부분인 형상변경통제활동에 대해 생명주기 단계들간의 자료공유방식을 기반으로 하는 모델링 방안을 제시하였다.

## 2. 배경 및 관련연구

### 2.1 소프트웨어형상관리 정의

소프트웨어형상관리는 특정 시스템의 개발과정에서 발생하는 각종 결과물(소프트웨어, 하드웨어, 문서)들의 계획, 개발 및 운용 등을 종합하여 시스템 형상을 형성하고, 이들에 대한 변경을 체계적으로 관리하는 활동이다[6, 8, 9].

소프트웨어형상관리는 하나의 소프트웨어시스템 생명주기 전반에 걸쳐 정의되는 형상항목(configuration item)과 그에 관련되어 생성되는 형상물(configuration product)을 종합하여 시스템 형상을 구성하고 이들에 대한 변경을 체계적으로 관리하기 위한 활동이다. 형상관리를 이루는 네 가지의 주요 활동은 시스템의 형상을 완성하는 형상인식(Configuration Identification), 형상변경에 관한 제어를 실시하는 형상변경통제(Configuration Change Control), 형상이 형성되고 변경되는 내용을 기록하고 보고하는 형상상태회계(Configuration Status Accounting), 형상 형성과 승인된 형상변경요구가 적절하게 산물에 구현되었는지를 확인하는 형상감사(Configuration Auditing) 등이다[6, 8, 11, 16].

형상은 생명주기의 각 단계에 따라 만들어진 문서

와 프로그램을 말한다. 형상을 이루는 형상항목에는 정의단계의 문서, 개발단계의 문서와 프로그램, 시험계획과 절차, 사용자 매뉴얼, 유지보수단계의 변경사항, 표준 및 절차 등이 있다[6, 7, 10].

### 2.2 기존의 소프트웨어 형상관리도구 연구

초기의 형상관리방안은 운영체제에 포함된 버전제어기능(예:UNIX의 SCCS)과 BUILD 기능(예:Make) 등으로 이루어졌다. 형상변경통제기능은 일부만 자동화되었고 통제정책이나 절차 등은 수작업으로 문서화되어 있었다. 이 시기에는 버전제어, 코드컴파일, 버그 추적 및 해결 등에 주력했으며 단순한 버전제어로는 형상관리의 요구를 충족시키지 못함을 인식했다.

현재의 형상관리연구는 기술적인 면에서 많은 진보가 있었으며, 현재 상용화 또는 연구되어 있는 제품들은 정보저장소(Repository), Distributed component, 시스템 모델링, 변경요청, Change set, Consistency maintenance, Workspace, Transparent view, Transaction과 같은 기능들을 가지고 있다. 〈표 1〉은 현재 상용화된 주요 형상관리도구들를 나타내고 있다[10, 13].

이러한 형상관리도구들은 형상관리의 기능들을 독립적으로 몇 가지씩 특징적으로 구현하였으나 생명주기단계 개념에 의한 형상물 관리, 생명주기단계의 형상물간의 자료공유와 상호참조 기능, 응용정보 이외의 메타정보관리 기능 등의 지원은 제한되어 있다[7, 10].

### 2.3 정보자원사전시스템(IRDS)

IRDS는 정보자원과 메타 정보관리를 위한 핵심이 되는 소프트웨어 도구로서 정보환경을 표현하거나 모델링할 수 있도록 지원하는 특별한 응용시스템이다[1, 3]. IRDS는 소프트웨어 형상관리를 위한 메타정보를 Information Resource Dictionary(IRD)라고 불리우는 공유가능한 정보저장소에 저장하고 IRDS를 이용하여 IRD로의 접근을 관리하며, 버전제어, 이름부여, 사용자에 의한 스키마 확장 및 전체 생명주기 지원기능 등을 제공한다. 이러한 IRDS 표준을 이용함으로써 서로 다른 시스템 간의 메타정보의 교환이 가능하고 다수의 사용자가 생명주기 전반에 걸쳐 정보저장소의 정보관리 및 유지기능을 지원받을 수 있다[1, 3].

소프트웨어시스템 개발비용면에서 보면 기존의 DBMS를 정보저장소로 사용하는 시스템에서는 단순

〈표 1〉 주요 상용 형상관리도구

〈Table 1〉 Commercial configuration management tools

| 제 품 명 | 제 작 자 | 자료저장방법 | 기 능 | 특 징 |
|---|---|---|---|---|
| Adele | University of Grenoble | Database (Entity Relationship) | · Data modelling<br>· Interface checking<br>· Configuration building<br>· Workspace control | 불완전한 형상기술 점검 가능 |
| ADC (Aide-De-Camp) | Software Maintenance & Development Inc | Database (File 관리) | · Change set 관리<br>· Version tree 관리<br>· 다수 File 동시접근 가능<br>· 형상내용 보고서 출력 | |
| CMA (Configuration Management Assistant) | Tartan Laboratory | Database (Entity Relationship Attribute) | · 형상 구성<br>· Version 관리<br>· 동일 명칭 검사 | |
| DSEE(Domain SW Engineering Environment) | Apollo Inc. | File repository | · Version control<br>· 시스템 모델링<br>· 작업리스트 관리(진행, 완료 여부) | 각 항목이 configuration thread를 가짐 |
| RCS(Revision Control System) | W. Tichy | File repository | · Version tree 관리<br>· Change log 기록/관리 | version들간의 차이점만 기록해서 버전관리 |

정보관리 이외의 메타정보관리와 버전제어 등의 기능을 모두 프로그램 작성을 통해 구현해야 하나 정보자원사전시스템(IRDS)을 정보저장소로 사용하는 시스템에서는 이러한 기능들이 정보저장소 자체환경 안에 포함되어 지원되므로 형상관리시스템 구현시에 개발인력 및 시간을 대폭 절감할 수 있는 장점이 있다.

## 3. 소프트웨어 형상변경통제 모델

### 3.1 개 요

형상관리중 형상형성은 소프트웨어 형상기준선(Configuration Baseline) 이전의 생명주기 전 단계에 걸쳐 이루어지며 각 단계에서 생산된 형상자료는 정보저장소에 저장된다. 생명주기의 각 단계의 개발자 및 사용자는 정보저장소에 저장되어 있는 자료를 공유하고 재사용할 수 있으며 정보저장소 자료에의 접근은 IRDS와 같은 정보저장소지원 시스템을 통해 이루어진다.

IRDS를 통해서 저장 및 관리할 수 있는 자료로서는 프로그램, 데이타, 문서 등 소프트웨어 생명주기 단계에서 생성되는 모든 종류의 형상물들을 대상으로 한다. IRDS는 통합환경에서 이러한 자료들을 서로 다른 단계들에서 공유 및 상호참조할 수 있도록 도와준다[1, 3, 4].

정보저장소의 메타모델은 정보저장소 전체 내용에 대한 전반적이고 개념적인 관점에 입각하여 저장될 정보의 종류, 이들 정보의 형태와 성질, 그리고 이들 정보에 접근하고 정확성을 증명하는 방법 등을 정의한다. 메타모델이 표현되는 형태는 크게 E-R 모델과 객체지향모델이 있다. ANSI표준을 따르는 IRDS는 E-R모델을 사용하여 표현되는데, IRDS의 구조에서 메타모델은 IRD스키마에 해당하고 실제 인스턴스인 저장정보는 IRD메타데이타에 해당된다[1, 3, 4].

본 장에서는 소프트웨어개발 생명주기 단계들 중에서 사용자요구분석, 설계, 구현 등 3개의 단계들에서의 형상형성에 관한 메타모델을 정의한 다음 단계간 상호참조와 형상변경통제활동의 절차를 분석하여 통합메타모델을 구성하여 제시한다.

### 3.2 생명주기단계의 메타모델

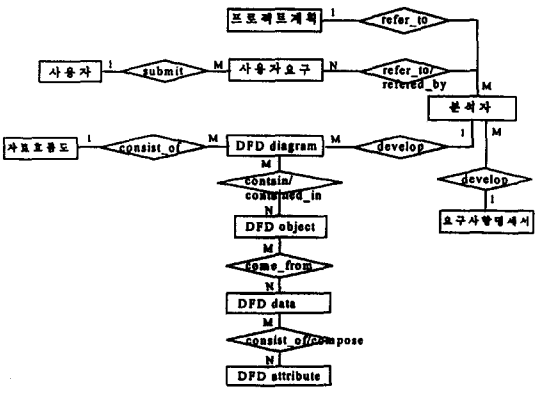
#### 3.2.1 통합메타모델의 생성

본 절에서는 소프트웨어개발 생명주기 단계들 중에서 사용자요구분석, 설계, 구현 등 3개의 단계들에서의 형상형성에 관한 메타모델을 정의한 다음 단계간 상호참조가 이루어지는 절차를 분석하여 통합메타모델을 구성한다.

소프트웨어 개발이 진행중일 때는 형상이 구성되어 가는 형상형성의 단계이며, 개발이 완료되어 형상이 동결된 후부터는 형상변경통제 활동이 적용될 수 있다. 따라서 요구분석, 설계, 구현의 3개 단계에서의 메타모델을 작성한 후 3개의 메타모델을 합쳐서 통합메타모델을 구성한다.

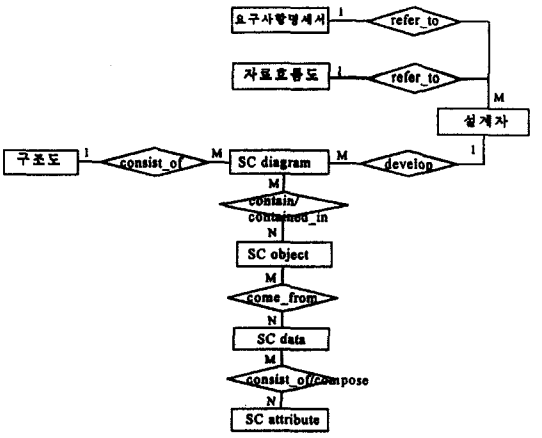
(그림 3)은 요구분석단계에서 이루어지는 형상형성의 메타모델을 기술한 것이다. 이 단계에서는 프로젝트계획과 사용자 요구사항 등을 기반으로 해서 시스템분석을 실시하며 이 단계의 주요 산출물은 사용자 요구사항명세서와 자료흐름도(Data Flow Diagram)이다. 자료흐름도의 객체(DFD object)에는 프로세스와 자료저장소 등이 포함된다[2, 4, 5].

(그림 4)는 설계단계에서 이루어지는 형상형성의 메타모델을 기술한 것이다. 이 단계에서는 요구분석단계에서 작성된 요구사항명세서와 자료흐름도 등을 기반으로 하여 시스템설계를 실시하며 주요 산출물은 구조도(Structure Chart)이다. 구조도의 객체(SC object)에는 루트모듈, 표준모듈, 처리모듈, 라이브러리모듈, 데이타베이스모듈 등이 있다[2, 4].




(그림 3) 요구분석단계의 메타모델

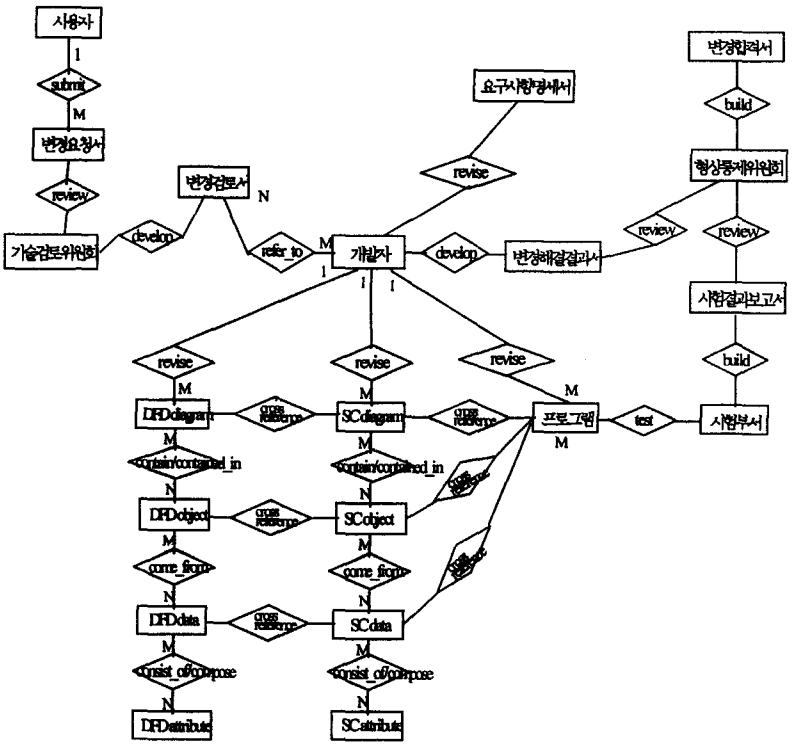
(Fig. 3) Meta-model of requirements analysis phase




(그림 4) 설계단계의 메타모델
(Fig. 4) Meta-model of design phase

구현단계의 주요 산출물은 시스템을 구성하는 프로그램이며 사용자와 시스템 관리자, 그리고 프로그래머 등이 연관된다.

위에 나타낸 기존 메타모델들은 메타모델들은 생명주기의 각 단계별 형상물처리를 위한 기반을 제시해 주지만 다른 단계의 형상물들과의 메타정보 관리를 할 수 있는 방법은 제시하지 못했다. 이러한 문제를 해결하기 위한 방법으로서 위의 3개 단계의 메타모델을 통합하여 기술한 것이 (그림 5)에 표현된 통합 메타모델이다. 여기서 DFDobject는 외부실체, 프로세스, 자료저장소 등을 나타내고, SCobject는 루트모듈, 표준모듈, 라이브러리모듈, 데이터베이스모듈을 포함하는 객체이다. 또한 DFDdata와 SCdata는 자료흐름을, attribute는 자료사전을 나타낸다. 여기에서는 3개 단계들에서 취급되는 메타데이타들과 그들간의 관계, 단계간의 상호관련성, 형상변경통제 등을 E-R




(그림 5) 통합메타모델
(Fig. 5) Integrated meta-model

모델을 통해 나타내고 있다. 이 모델을 사용하여 시스템 개발 후 형상이 동결된 후에 일정한 문서나 프로그램의 기능에 대한 형상변경요청이 있을 경우 이 변경요청과 관련된 프로그램과 시스템설계단계 및 요구분석단계의 산출물들을 상호참조관계를 이용해 쉽게 찾아서 변경을 수행할 수 있는 기반을 제공할 수 있다. 통합메타모델에서는 통합에 따른 특징을 간결하게 표현하기 위해서 3개 단계의 대표되는 Entity들과 Relationship들을 위주로 나타내었다.

프로그램은 설계단계의 산출물인 SC diagram, SC object, SC data와 cross_reference 관계로 연결되어 있다. 따라서 개발자는 프로그램의 설계문서인 구조도의 해당되는 도표와 데이터들을 추적하여 변경작업을 수행할 수 있다.

또한 설계단계의 산출물인 구조도의 계층별 Entity들과 요구분석단계의 산출물인 자료도의 계층별 Entity들간에도 cross_reference 관계로 연결되어 있으므로 DFD diagram, DFD object, DFD data 등도 추적하여 변경이 가능하다.

본 모델에서는 시스템분석, 설계 및 구현 단계에서 발생한 산출물들을 대상으로 각 단계간의 상호참조를 하기 위해 cross_reference 관계를 가질 수 있도록 하였다. 이러한 상호참조가 가능한 관계를 통해서 개발자는 프로그램이 작성되기 전단계인 요구분석 및 설계단계에서의 문서를 신속하게 접근하고 이해함으로써 변경요구에 관련된 사항을 쉽게 이해함으로써 변경작업 시간을 단축할 수 있다.

### 3.2.2 소프트웨어 형상변경통제절차

형상변경통제활동을 수행하기 위해서는 형상변경통제를 수행할 조직과 이를 처리하는 체계적인 통제절차, 그리고 절차수행에 소요되는 공식문서양식 등이 필요하다[7].

형상변경통제절차는 앞 절에서 기술한 형상변경통제모델에 근거하여 모델을 구성하는 개체들과 관계들을 이용하여 이루어지는 처리절차를 규정한다. 형상변경통제에 관련되는 상설기구는 사용자, 개발자, 시험부서, 기술검토위원회 등이 있으며 임시기구로서 형상통제위원회가 있다. 형상통제위원회는 형상관리관 등의 관리자들을 위원으로 포함하고 있으며, 기술검토위원회는 실무자들을 위원으로 가진다. 소프트웨어 형상변경통제절차는 다음과 같이 크게 일곱 단계로 나누어진다.

(1) 시스템 사용자나 부서에서 기능향상이나 오류수정을 위해 시스템형상에 대한 변경요청서를 기술검토위원회로 제출한다. 변경요청서에 기재하는 변경의 대상은 프로그램, 설계문서, 시스템분석문서, 요구사항명세서 등이 될 수 있다.

(2) 기술검토위원회에서는 제출된 변경요청서의 타당성을 검토하여 해결책임자 및 부서를 결정하고 변경검토서를 작성한다. 이 변경검토에는 변경요청서를 기반으로 하여 변경해야 할 대상과, 변경을 구현하는 방법을 수록한다. 변경검토서가 완성되면 형상관리관은 관련된 개발자에게 할당하여 변경작업을 준비하도록 한다. 이때 개발자에는 시스템분석관과 프로그래머 등이 포함된다. 작성된 변경검토서는 형상관리관에게 제출한다.

(3) 형상관리관은 변경검토서를 관련 개발자에게 할당해 주며, 개발자는 검토가 끝난 변경요청서에 대한 해결방안을 마련하고 문제를 해결한다. 이때 문제해결에는 자체적인 시험도 포함해서 실시한다. 시험에는 단위기능시험과 통합시험이 있는데, 프로그램의 단위기능시험은 시스템을 구성하는 기능의 기본단위인 블록에 대하여 시험을 실시하는 것이며 통합시험은 단위시험이 완료된 블록들간의 상호관계와 관련하여 오류를 점검하기 위한 시험이다. 개발자는 문제해결이 완료되면 형상변경해결결과서를 작성하여 개발실장의 승인을 받은 후 형상관리관에게 제출한다.

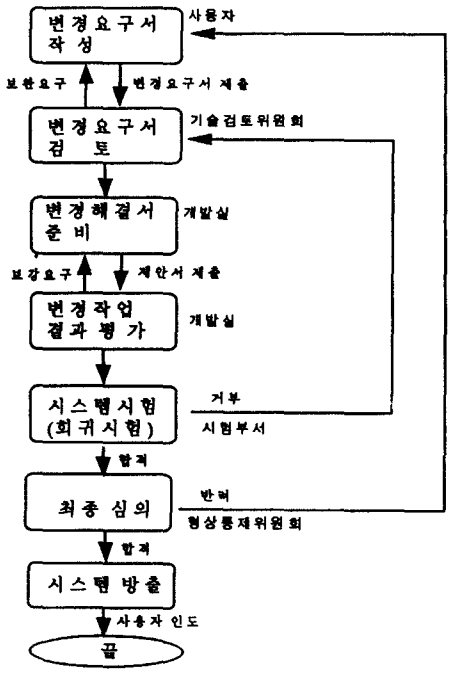
(4) 형상관리관은 형상변경해결결과서를 접수하면 시험부서에 시스템시험을 의뢰한다. 시스템시험에서는 통합시험을 완료한 시스템을 대상으로 요구사항정의서에 제시된 기능들을 만족하는지 여부를 시험하고 기타 시스템의 성능향상을 위한 제반 활동을 수행한다. 시험부서는 시험이 끝난 후 형상변경시험결과서를 작성하여 형상관리관에게 제출한다.

(5) 형상관리관은 시스템시험이 완료된 변경요청서에 대한 최종심의를 위해 형상통제위원회를 개최한다. 형상통제위원회는 그동안 수행된 변경작업결과 및 시험결과를 기반으로 심의를 실시하여 최종승인을 내리며 형상관리자는 형상변경합격서를 작성한다.

(6) 형상관리관은 합격된 형상변경요청에 관련된 변경형상항목을 등록하고 시스템형상변경에 반영하

며 변경된 시스템을 사용자에게 인도한다.

(그림 6)에는 이러한 일련의 절차가 표시되어 있다.




(그림 6) 소프트웨어 형상변경통제 절차
(Fig. 6) Software configuration change control procedure

### 3.2.3 통합메타모델의 분석

#### 3.2.3.1 변경요구 경우에 따른 분석

형상변경통제는 소프트웨어의 개발이 완료되고 형상이 동결된 후부터 시작된다. 본 절에서는 통합메타모델과 형상변경통제절차를 기반으로 실제 형상변경요구가 처리되는 절차를 세 가지 경우의 변경요구에 대해 분석한다.

(1) 프로그램모듈에 대한 변경요구

개발된 시스템의 사용자 또는 개발자가 시스템의 기능향상이나 오류수정을 위해 시스템형상에 대한 변경요구서를 제출한다. 이 때 한 명이 여러 건의 변경요구서를 작성할 수 있다.

변경요구서는 형상통제위원회와 review relationship으로 연결되어 있어 형상통제위원회의 검토를 받는다. 형상통제위원회에는 다수의 관리자들이 attend relationship으로 연결되어 참여한다. 검토결과 제출된 변경요구서의 타당성이 인정되면 개발을 담당한 개발자에게 할당하여 변경작업을 수행하도록 한다. 이 때 변경요구서와 개발자는 assign_to relationship으로 연결되어 있다. 임무를 할당받은 개발자는 시스템의 모듈을 생산했던 제도분석관과 프로그래머를 포함한다. 개발자는 변경요구서에 관련된 모듈에 대한 변경작업을 준비한다. 이때 개발자와 모듈간에는 develop relationship으로 연결되어 있다.

모듈은 블록, 서브시스템, 시스템과 consist_of relationship으로 연결되어 있고 또한 설계단계의 산출물인 SC diagram, SC object, SC data와 cross_reference relationship으로 연결되어 있다. 따라서 개발자는 변경대상인 모듈의 설계문서인 구조도의 해당되는 도표와 데이터들을 추적하여 변경작업을 수행할 수 있다.

또한 설계단계의 산출물인 구조도의 계층별 Entity들과 요구분석단계의 산출물인 자료도의 계층별 Entity들간에도 cross_reference relationship으로 연결되어 있으므로 DFD diagram, DFD object, DFD data 등도 추적하여 변경이 가능하다.

이러한 상호참조가 가능한 관계를 통해서 개발자는 모듈이 작성되기 전에 요구분석 및 설계단계에서의 문서를 신속하게 접근하고 이해함으로써 변경요구에 관련된 사항을 쉽게 이해하고 모듈변경작업 시간을 단축할 수 있다.

(2) 설계문서에 대한 변경요구

설계단계의 산출물인 구조도에 대한 설계상의 변경요구가 제출된 경우이다. 이때 형상통제위원회의 검토를 받는 상황은 (1)의 경우와 동일하다. 검토결과 제출된 변경요구서의 타당성이 인정되면 개발을 담당한 개발자에게 할당하여 변경작업을 수행하도록 한다. 임무를 할당받은 개발자는 시스템의 설계를 실시했던 제도분석관이 포함된다. 제도분석관은 변경요구가 제기된 SC diagram과 그 안에 포함된 SC object와 SC data에 대한 변경을 준비하며 해당 SC diagram에 관련된 모듈들을 추적해서 모듈생산에 관련된 프로그래머에게 모듈에 대한 변경작업을 할당한다. 이때 SC diagram과 모듈간에는 cross_reference relationship으로 연결되어 있어서 관련되는 모듈을

용이하게 추적하여 변경대상이 되는 모듈들에 대한 변경작업을 신속히 수행할 수 있다.

SC diagram은 또한 설계단계의 산출물인 DFD diagram과 cross_reference relationship으로 연결되어 있고, SC object는 DFD object와, SC data는 DFD data와 각각 상호참조 관계로 연결되어 있다. 따라서 개발자는 변경대상인 구조도와 대칭되는 요구분석단계의 산출물인 자료흐름도의 각 계층별로 해당되는 도표와 데이터들을 추적하여 변경작업을 수행할 수 있다.

(3) 요구분석문서에 대한 변경요구

요구분석단계의 산출물인 구조도에 대한 변경요구가 제출된 경우이다. 이때 형상통제위원회의 검토를 받는 상황은 (1)의 경우와 동일하다. 형상통제위원회의 검토 결과 제출된 변경요구서의 타당성이 인정되면 요구분석을 실시한 개발자에게 할당하여 변경작업을 수행하도록 한다. 임무를 할당받은 개발자는 시스템의 요구분석을 실시했던 제도분석관과 관리자를 포함한다. 개발자는 변경요구가 제기된 DFD diagram과 그 안에 포함된 DFD object와 DFD data에 대한 변경을 준비한다.

DFD diagram은 또한 설계단계의 산출물인 SC diagram과 cross_reference relationship으로 연결되어 있고, DFD object는 SC object와, DFD data는 SC data와 각각 상호참조 관계로 연결되어 있다. 따라서 개발자는 변경대상인 자료흐름도와 대칭되는 설계단계의 산출물인 자료흐름도의 각 계층별로 해당되는 도표와 데이터들을 추적하여 변경작업을 수행할 수 있다.

변경대상이 된 SC diagram과 연관된 모듈들에 대해서는 프로그래머에게 변경작업을 지시한다. 이때 SC diagram과 모듈간에는 cross_reference relationship으로 연결되어 있어서 관련되는 모듈을 용이하게 추적하여 변경대상이 되는 모듈들에 대한 변경작업을 신속히 수행할 수 있다.

### 3.3 IRD스키마 ER모델

메타모델이 표현되는 형태는 크게 E-R모델과 객체지향모델이 있다. ANSI IRDS는 E-R모델로 표현되며, IRDS자료구조상 메타모델은 IRD스키마에 해당하고 실제 저장정보는 IRD 메타데이타에 해당한다[17]. IRDS 모델링의 대상이 되는 주제에 대한 일반적인 E-R모델이 IRDS모델을 염두에 둔 모델이 아닌 경우 곧바로 IRDS 모델로 전환하기가 어렵다. 따라서 메타모델과 IRDS 모델에서 개체형과 관계형이 직접 일대일 관계로 대응되지 않을 수 있기 때문에 다음과 고려할 사항들이 있다:

첫째, IRDS 모델링을 통해 의미있는 결과를 갖기 위해 반드시 정의되어야 할 개체형 및 관계형을 추출하고

둘째, 전체 시스템을 관리하는 생명주기지원을 고려한 일관성있는 이름부여가 보장되어야 하고

셋째, IRDS의 생명주기 단계를 응용하기 위해서는 생명주기단계내 또는 단계간의 정보무결성 및 상호참조를 지원할 수 있는 스키마가 되도록 각 단계의 개체들을 연관시키는 관계형을 정의하여야 하며

넷째, 핵심 IRDS에 이미 정의된 스키마의 활용가능성과 부가적인 스키마를 정의하여 사용할 것인가를 고찰하며

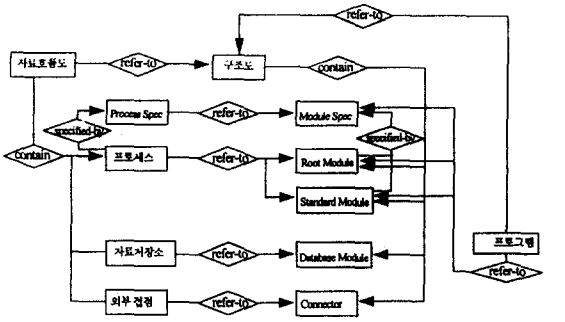
다섯째, IRDS에서는 개체간의 관계를 정의한 관계형에 대해 자동적으로 양방향으로 지원하지 않기 때문에 양방향 관계가 필요한 관계형은 반드시 두 개체형에 대해 두 개의 관계형을 별도로 정의할 필요가 있다.

메타모델로부터 새로운 IRD스키마를 E-R모델로 나타내면 다음과 같다. 먼저 전체 시스템에서 자료흐름도(DFD)와 구조도(SC), 프로그램이 차지하는 관계를 분석한다. 자료흐름도에 포함되는 개체들은 프로세스, 자료저장소, 외부개체, 외부접점 등이 있으며 이러한 개체들은 자료흐름도와 contains 관계로 연결되어 있다. 프로세스는 process_spec에 의해 더욱 자세히 설명되는데 두 개체들은 specified_by 관계로 연결된다. 자료저장소에는 DFD_data가 저장되는데 이 DFD_data는 ATTRIBUTE들로 구성된다. 프로세스는 외부개체나 외부접점으로부터 시작되어 표현되는데 source_of relationship으로 나타내었다. 자료흐름도와 ATTRIBUTE는 자체적으로 반복적인 표현(nesting)이 가능하므로 parent_of 관계로 자기 자신에게 연결되었다.

구조도에 포함되는 개체들은 Connector, Root Module, Standard Module, Trans Module, Library Module, DB Module 등이 있으며 이러한 개체들은 구조도와 contains 관계로 연결되어 있다. 각 모듈들은

Module_Spec에 의해 더욱 자세히 설명되는데 두 개체들은 specified_by 관계로 연결된다. 각 모듈에는 SC_data가 사용되는데 이 SC_data는 ATTRIBUTE들로 구성된다. 구조도와 ATTRIBUTE는 자체적으로 반복적인 표현(nesting)이 가능하므로 parent_of 관계로 자기 자신에게 연결되어 있다.

(그림 7)은 시스템 분석단계와 설계단계, 구현단계 산물들인 자료흐름도, 구조도, 프로그램간의 상호참조 추적을 위한 관계를 나타내기 위한 모델이다. 요구분석단계의 산물인 자료흐름도의 개체인 프로세스는 설계단계의 산물인 구조도의 개체인 Root Module 및 Standard Module과 참조관계를 가지고, 자료저장소는 Database Module과, 외부접점은 Connector와 참조관계를 가진다. 구조도의 각 모듈은 구현단계의 산물인 모듈과 참조관계를 가진다. 이와 같이 3개 단계의 산출물들이 서로 상호참조관계를 유지하는 관계를 나타내고 있다.




(그림 7) 자료흐름도, 구조도, 프로그램간 관계를 표현한 IRD 스키마 모델

(Fig.7) IRD schema model of relationship among DFD, SC, and program

### 3.4 IRD스키마 정의

일단 E-R모델이 완성되면 사용자는 IRDS명령어를 사용해서 자신의 IRD 응용에 적합한 스키마 구조를 정의할 수 있다. IRDS에 이미 기본적으로 내장된 스키마만을 사용할 경우에는 IRD스키마를 별도로 정의

〈표 2〉 IRDS명령어를 사용한 IRD스키마 정의 예

〈Table 2〉 Examples of IRD schema definition using IRDS commands

| Command Group | IRDS Commands |
|---|---|
| Lifecycle Phase 정의 | Add meta-entity SYSTEM-ANALYSIS meta-entity-type = IRD-PARTITION; |
| Entity Type | Add meta-entity PROJECT meta-entity-type = ENTITY-TYPE;<br>Add meta-entity DFD meta-entity-type = ENTITY-TYPE;<br>Add meta-entity PROCESS meta-entity-type = ENTITY-TYPE; |
| Attribute Type | Add meta-entity CONTEXT-DFD meta-entity-type = ATTRIBUTE-TYPE with SYS-GENED = YES, PURPOSE = CONTEXT DIAGRAM; |
| Entity Type과 Attribute Type의 결합 | Add meta-relationship DFD entity-type-contains-attribute-type DFD-NO; |
| Relationship Class Type | Add meta-entity CONTAINS meta-entity-type = RELATIONSHIP-CLASS-TYPE; |
| Relationship Type | Add meta-entity PROJECT-CONTAINS-P-DFD meta-entity-type = RELATIONSHIP-TYPE; |
| Relationship Type과 Relationship Class Type의 결합 | Add meta-relationship PROJECT-CONTAINS-P-DFD relationship-type-member-of-relationship-class-typeCONTAINS; |
| RelationshipType 위치정의 | Addmeta-relationshipPROJECT-CONTAINS-P-DFDrelationship-type-connects-entity-type PROJECTposition = 1;<br>Addmeta-relationshipPROJECT-CONTAINS-P-DFDrelationship-type-connects-entity-type P-DFDposition = 2; |

할 필요없이 바로 메타데이타를 입력할 수 있으나 사용자가 자신이 정의하는 스키마를 추가로 사용하려면 IRDS명령어에 따라 스키마를 정의해야 한다[17].

〈표 1〉은 IRD스키마를 위한 E-R모델을 IRDS명령어 문법을 따라 IRD스키마로 정의한 것의 일부를 예로서 기술한 것이다. Lifecycle Phase와 Entity Type, Attribute Type, Relationship Type 등을 갖는 meta-entity를 정의하는 때는 'Add' 명령문 뒤에 정의하고자 하는 meta-entity의 이름과 meta-entity type을 명시한다. Entity Type과 Attribute Type, Relationship Type과 Relationship Class Type을 결합시킬 경우는 meta-relationship의 앞뒤에 관련되는 meta-entity들을 위치시켜 정의한다. Relationship Type을 갖는 meta-entity는 Relationship Class Type을 갖는 meta-entity의 앞뒤 부분에 관련되는 두 개의 meta-entity들이 접합된 형태로 정의된다.

본 연구에서는 이러한 IRDS명령어들을 사용해서 형상변경통제 통합모델 스키마를 정의하고 형상변경통제절차에 필요한 기능들을 구현할 예정이다.

## 4. 결 론

본 논문에서는 대규모 소프트웨어시스템의 형상관리를 수행함에 있어서 필수적으로 요구되는 정보관리를 위해서 소프트웨어형상관리중 가장 중요한 기능인 형상변경통제활동의 모델을 정의하였다. 소프트웨어 생명주기단계들중에서 요구사항분석, 시스템설계, 프로그램 구현의 3개 단계의 메타모델을 각각 정의한 후 통합하여 형상이 동결된 이후의 형상변경통제에 있어서 생명주기단계간 상호참조와 자료공유가 가능한 모델을 정의하고 IRD스키마로 변환하는 과정을 보였다.

각 단계별 상세한 모듈에 대한 모델링을 통하여 완벽한 형상변경통제를 위한 시스템을 확장하는 것이 필요하다. 이 부분의 구현이 완료되면 통합된 정보자원관리를 필요로 하는 CALS 시스템의 정보저장소와 연결하여 사용될 수 있을 것으로 기대된다.

## 참 고 문 헌

[1] Robert E. Hodges, Information Resource Dictionary Services Architecture Technical Report, X3H4/93-048R1, 1993.

[2] Roger S. Pressman, Software Engineering: A Practitioner's Approach 3rd Edition, McGraw-Hill Inc., 1992.

[3] 김기봉, 박중기, 조유희, 진성일, "Design and Implementation of IRDS Service and User Interface for Information Resource Management," 한국정보과학회 논문집, 22(11), 1995.

[4] 김기봉, 최광준, 박중기, 진성일, "A Design and Implementation of an Integrated CASE Tool Repository using IRDS," The International Association of Science and Technology for Development, 14, 1996.

[5] J.S. Lee, D.H. Suh, J.W. Park, 천유식, "TCMS Tool for TICOM Configuration Management," 한국정보과학회 논문집, 9(2), 1991.

[6] 천유식, 소프트웨어 개발방법론, 대청미디어, 1995.

[7] Software Technology Support Center, Software Configuration Management Technology Report, Sep. 1994.

[8] IEEE Std 828-1990, IEEE Standard for Software Configuration Management Plans, 1990.

[9] H. Ronald Berlack, Software Configuration Management, Wiley Professional Computing, 1992.

[10] Susan Dart, Concepts in Configuration Management Systems, Carnegie-Mellon University, 1991.

[11] 정호원, 양해술, ISO 9000 시리즈와 소프트웨어 품질 시스템, 하이테크정보, 1993.

[12] 한국전자통신연구소 컴퓨터연구단, 고속중형컴퓨터 개발체계, 한국전자통신연구소, 1992.

[13] Terry Moriatry, Are You Ready for a Repository, Database Programming and Design, March 1990.

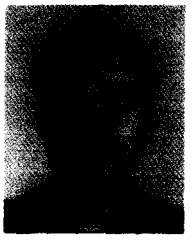
[14] 시스템공학연구소, 국산 주전산기용 소프트웨어 개발지원도구 개발, 소프트웨어 정보저장소 개발, 과학기술처, 1994.

[15] 양해술, I-CASE의 도구통합과 방법, 정보과학회지 12권 2호, 1994. 3.

[16] Ben-Menachem, Software Configuration Management Guidebook, McGraw-Hill Book Company,

1994.

[17] 김기봉, 이혜란, 진성일, CASE 정보저장소를 위한 IRDS 모델링, 정보과학회논문지 24권 2호, 1997. 2.

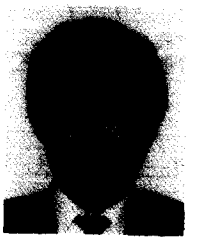


**최 광 준**

1978년 육군사관학교 전자공학과 졸업(학사)
1985년 미해군대학원 전산과학과(이학석사)
1991년~현재 육군전산소 개발과장
1993년~현재 충남대학교 컴퓨터과학과 박사과정
관심분야: 데이터베이스, 정보저장소, SW형상관리

**김 기 봉**

1991년 충남대학교 전산학과 졸업(학사)
1993년 충남대학교 대학원 전산학과 졸업(석사)
1996년 충남대학교 대학원 박사과정 수료
1994년~1996년 혜전전문대학 전산정보처리과 교수
1997년~현재 대전보건전문대학 전산정보처리과 교수
관심분야: 데이터베이스, CALS, 정보저장소시스템



**진 성 일**

1978년 서울대학교 계산통계학과 졸업(학사)
1980년 한국과학기술원 전산학과 졸업(석사)
1994년 한국과학기술원 전산학과 졸업(박사)
1983년~1989년 충남대학교 전산학과 조교수
1987년~1989년 Northwestern대학 객원교수
1989년~1994년 충남대학교 전산학과 부교수
1994년~현재 충남대학교 컴퓨터과학과 교수
1996년~현재 충남대학교 소프트웨어연구센터 소장
관심분야: 병렬데이터베이스, CALS, 멀티미디어