소프트웨어 개발관리를 지원하기 위한 프로세스 모델 기반 CASE 도구 구축방법의 제시

조 병 호† 김 태 달††

요 약

IPSE(Integrated Project Support Environment) 도구는 언어 중심적이고, 개발방법론에 근거한 툴셋 형태로 제공되는 현재의 CASE 도구들의 주요 기능들을 하나로 통합하고자하는 노력의 결과로 볼 수 있다. 프로세스 모델을 기반으로 한 IPSE 접근방법이 통합 CASE 구현을 위한 효과적인 방법으로 여겨진다. PM-CASE(Process Model based CASE) 도구는 새로운 프로세스 모델링 기법에 의해 프로세스를 표현한 다이아그램을 작성하기 위한 시제품으로서, 프로세스내의 태스크 관련 속성들을 정의하고 데이터베이스에 저장한다. 이들 속성들은 태스크 수행 중에 만들어진 산출물에 대한 정보의 검색 및 태스크와 연관된 도구를 호출하는데 사용된다. 본 논문에서는 PSEE(Process centered Software Engineering Environments) 도구들을 비교 분석하고, PM-CASE 도구의 기본개념, 구조, 설계에 대한 기술을 통해 효과적인 소프트웨어 개발관리를 지원하는 프로세스 모델 기반 CASE 도구의 구축방법을 제시한다.

A Method of Building an Process Model-based CASE Tool to Support Software Development and Management

Byungho Cho† and Taidal Kim††

ABSTRACT

The IPSE(Integrated Project Support Environment) tool can be seen as a result of an attempt to synthesize the key aspects of language-centered, specific methodology-based and toolkit oriented environments, which are current CASE tools into an organic whole. The IPSE approach based on a process model is regarded as an effective way to implement integrated CASE. The PM-CASE (Process Model based CASE) tool is currently a prototype which draw diagrams describing processes by using a new modeling technique. Attributes related with a task of withen the process model should be defined and saved the database. These attributes are used to retrieve the information of products, and to call the tool related with the task. In this paper, PSEE(Process centered Software Engineering Environment) tools are compared and analyzed. By describing the basic concept, architecture and design of PM-CASE tool, a method of building an process model-based CASE tool is proposed to support an effect software development and management.

1. 서 론

소프트웨어 개발관리를 위한 통합적인 도구를 소프트웨어 품질향상 및 생산성을 향상하는데 활용하고자 이들 통합 CASE 도구 및 개발지원 환경을 제공하는 도구의 구축에 대한 관심이 고조되고 있다. 아직까지 소프트웨어 개발환경 지원을 위한 효과적인 통합도구로서 상용화된 도구는 없고, 단지 소프트웨어 생명주기 단계의 분석 및 설계 혹은 코딩단계의 일부분만을 지원하는 도구가 대부분이다.

초창기의 개발환경을 지원하는 도구들은 언어 중심적인 개발지원도구이거나, 주로 툴킷(Toolkit) 방식으로서 분석도구, 프로그래밍, 테스트 등을 위한 툴킷 형태의 도구들이다. 또한 응용시

† 정 회 원 : 한국과학기술원 교통산업연구센터 연구원
†† 정 회 원 : 도로교통안전협회 전산실장
논문접수 : 1995년 4월 25일, 심사완료 : 1995년 11월 9일.

스템의 발전에 따라 여러 가지 방법론이 제안되었고 이들 방법론을 지원하는 CASE 도구들이 개발되었다.

현재까지 상용화된 대다수의 CASE 도구들은 주로 특정 방법론의 일부 기능을 지원하는 단순 그래픽 기능이나 도큐멘테이션의 기능을 가진 도구들이 대부분이다. 이와 같은 방법론 중심적인 CASE 도구들의 예로서 Martin의 정보공학 계열의 IEF, IEW가 있고, 구조적 방법론을 지원하는 StructSoft사의 TurboCASE, Excel Software사의 MacDesigner와 객체지향방법론을 지원하는 BNR사의 Object Time 등을 들 수 있다[1].

이에 비해 소프트웨어 개발지원 환경을 제공하는 통합 CASE 도구의 구축은 소프트웨어 생명주기 모든 단계의 프로세스를 지원하는 도구들이 사용자의 작업진행에 맞추어서 필요로 하는 도구들을 마음대로 사용할 수 있도록 각 생명주기 단계에 필요한 도구들을 틀셋 형식으로 제공하고, 정보저장소를 활용하여 자료의 공유 및 서로 다른 형태의 도구들 간의 정보 교환이 가능하도록 하는 것이다.

사용자 측면에서 보면 단일 사용자가 이용하는 CASE 도구에서 여러 사용자가 동시에 사용이 가능하거나 팀단위의 작업이 가능하며 자료의 공유 및 여러가지 기능을 지원하는 도구를 함께 사용이 가능한 통합 CASE로 발전하는 추세이다.

이와 같이 통합 CASE 도구의 구축의 필요성은 소프트웨어의 개발이 점차 복잡해지고 여러 사람이 공동으로 참여함에 따라 개인 중심적이고 단순히 프로그래밍을 위한 개발지원 도구보다는 자료의 공유 및 도구를 함께 이용할 수 있는 통합적인 개발환경을 지원하는 도구의 필요성을 인식하게 되었다.

따라서 통합 CASE의 구축은 소프트웨어 생명주기 전 단계의 프로세스를 지원하는 통합 프로젝트 지원환경을 제공하기 위한 도구의 구축을 목표로 하는 IPSE(Integrated Project Support Environment) 방향으로 나아가고 있다. 이에 대한 연구로는 단순한 프로그래밍 환경을 지원하는 도구인 APSE[5], InterLisp[20]가 있고, 프로젝트 개발과정을 커다란 작업단위로 나눈 개념의 프로세스를 지원하는 통합 개발지원도구인 ISTAR [6]와 DOD-STD-2167A의 소프트웨어 생명주기 프로세스를 중심으로 한 통합도구인 SLCSE [19]가 있다. 최근에는 프로세스 모델을 중심으로 자료 및 도구통합을 이루는 PSEE(Process centered Software Engineering Environment) 도구[11, 12, 13, 16]가 IPSE 도구를 구축하기 위한 주요한 방법으로서 연구되고 있다.

이와 같이 소프트웨어 개발관리를 위한 통합적인 CASE 도구의 구축은 프로세스 모델을 중심으로 한 PSEE 도구형태가 IPSE 구축을 위한 효과적인 방법으로 연구되고 있다. 따라서 본 논문에서는 새로운 소프트웨어 프로세스 모델링 방법에 의해 설계된 소프트웨어 프로세스 모델을 기반으로 하여 현재 프로토타입으로 개발하고 있는 프로세스 모델 기반 CASE 도구인 PM-CASE(Process Model based CASE)를 이용하여 효과적인 통합 CASE 구축하기 위한 방법을 제시하고자 한다.

본 논문의 구성은 2장은 프로세스 중심의 CASE 도구들에 대한 조사 및 비교분석에 대한 것을 기술하고, 3장은 PM-CASE 도구의 구축에 대한 것으로서 새로운 프로세스 모델 표기법을 소개하고, 이 프로세스를 모델을 중심으로 한 PM-CASE 도구의 기본원리 및 구조, 시제품 설계에 대한 기술을 통해 효과적인 통합 CASE 구축을 위한 방법을 제시한다. 또한 4장에서는 시제품으로서 설계된 PM-CASE의 발전 방향을 제시하며, 5장은 결론 및 연구방향에 대해 기술한다.

2. 프로세스 중심의 CASE 도구들에 대한 조사 및 비교분석

소프트웨어 개발관리를 효과적으로 지원하기 위한 통합도구로서의 CASE 도구의 역할은 두 가지 면에서 정보를 다룰 수 있는 능력이 있어야 한다. 즉, 정적인 정보는 요구사항명세서, 설계문서, 테스트사례, 프로그램 등과 같은 자료나 산출물을 의미하며, 동적인 정보는 산출물이 어떻게 개발되고 유지되는지, 사용자간의 프로세스에 대한 공유와 통기를 이루기 위한 방법, 도구간의 통합, 관리, 절차나 표기의 표준제정 등에 대한

정보를 의미한다.

현재의 CASE의 형태는 단순히 독립적인 도구에 의존적인 커널을 가진 시스템이거나, 미리 정해진 정책에 의해서 상용화된 데이터베이스 및 화일 시스템을 이용하여 자료의 저장 및 검색이 가능하고 몇까지 도구만이 제공되는 형태로 현재 상용화된 대부분의 CASE 도구들이 이 범주에 속한다. 앞으로의 CASE 도구는 정적인 면과 동적인 면을 다 지원할 수 있는 통합 CASE 도구의 방향으로 나아가고 있으며, 이와 같은 효과적인 통합 CASE 도구를 구축하기 위한 방법으로서 PSEE(Process centered Software Engineering Environment) 도구들이 새로운 방법으로 제안되고 있다. 다음은 이러한 프로세스 중심의 CASE 도구들에 대해 알아보고 그 특징을 비교분석하였다.

2.1 프로세스 중심의 CASE 도구

프로세스를 중심으로 한 소프트웨어 개발지원 환경을 제공하는 도구들이 통합 CASE 도구의 효과적인 방법으로 제시되고 있다. 다음은 프로젝트 수행을 위한 프로세스를 contract 단위의 커다란 작업으로 나누어서 그 프로젝트의 작업을 지원하는 도구들을 여러 그룹으로 나누어진 툴셋으로 지원하는 IPSE(Integrated Project Support Environment) 도구인 ISTAR[6], 생명주기 표준인 DOD-STD-2167A[14]의 프로세스들을 지원하는 개발지원 환경 도구인 SLCSE[18]와 프로세스 모델링에 의해 상세한 프로세스를 표현하고 이를 지원하는 도구들의 통합을 이룬 PSEE 도구인 Adele[11], ALF[12], Weaver[13], Matisee[16]에 대해 조사하여 그 특징을 알아본다.

(1) ISTAR

ISTAR는 통합적인 소프트웨어 개발환경을 지원하는 초창기의 IPSE(Integrated Project Support Environment) 도구로서, 프로젝트를 수행하기 위한 프로세스를 contract라는 여러 가지 작업으로 나누어서 이것들과 관련된 도구들을 통합적으로 사용이 가능하도록 만든 것이다. contract는 subcontract의 계층구조로 구성되며 contract의 저장을 위해 데이터베이스를 사용한다. 이 시스템은 프로세스와 연관된 도구들을 여러 개의 집단으로 묶어서 contract의 사용자가 원하는 작업을 하고자 할 때 호출해서 사용이 가능하다. 또한 도구의 통합을 위해 공통된 텍스트나 폼(form) 편집기 등의 사용자 인터페이스를 통해서 도구들을 함께 이용할 수 있고, 프로세스에 관련된 자료의 저장을 위하여 contract 단위로 독립적인 데이터베이스를 사용하고 이들 데이터베이스간의 정보교환을 위하여 LAN과 같은 물리적 통신 시스템을 이용한다.

(2) SLCSE

SLCSE는 DOD-STD-2167A 생명주기 표준에 근거한 생명주기의 프로세스를 중심으로 자료 및 도구의 통합이 이루어지도록 만든 통합적인 소프트웨어 개발환경 도구이다. 이는 생명주기 프로세스 관련 활동과 산출물에 대한 정보를 E-R 데이터 모델링 기법을 이용해 정의한다. 이 시스템은 프로세스 관련 정보를 데이터베이스에 저장하고, 검색 및 갱신이 가능하도록 상용화된 DBMS를 사용한다. 사용자는 생명주기 단계에서 필요한 도구들이 툴셋으로 만들어져 있어서 역할(예 : 프로젝트 관리자, 시스템 분석가, 프로그래머)에 따라 필요한 도구들의 선택이 가능하고, 데이터베이스에 저장된 프로세스에 관련된 활동 및 산출물을 이 도구들을 사용하여 작업을 하게 된다. 사용자 인터페이스는 공용 인터페이스를 사용하지만 VT-100 터미널을 쓰던 시대이므로 마우스 방식의 그래픽 사용자 인터페이스 방식을 제공하지는 못한다.

(3) Adele

Adele은 정적인 면에서 기록문서, 코드 등과 같은 자료의 통합을 이루기 위한 데이터 모델로서 객체지향개념을 가진 ERA(Entity Relation Attribute)모델을 사용한다. 또한 동적인 면의 지원을 위한 행위 모델로서 event와 action 형태(예 : ON event DO action)의 언어로 표현한다. 또한 트리거를 이용한 표기법으로 행위가 수행되기 위한 사전조건(pre-condition)과 사후조건(post-condition)을 표현한다.

자료의 저장을 위하여 Adele은 Adele-DB를

사용하는데 프로젝트가 큰 경우 프로젝트를 분할하여 부트리 형태로 구성한다. 루트가 프로젝트가 되고 여러 계층의 부프로젝트로 나뉘게 되며, 이들은 partition이란 용어를 사용하여 분할된 구성요소의 스키마를 기술하게 된다. 이때 partition은 ERA 데이터모델로 표현되고 상속성(inheritance), 연관성(association), 집단화(aggregation) 형태의 표현도 가능하다. 자료 및 프로세스의 구성관리를 위한 버전관리 시스템이 있으며 Adele의 OMS(object Management System)는 PCTE(Portable Common Tool Environment)의 OMS를 지원할 만큼 잘 만들어져 있고, BMS(Broadcast Message System)에 의해 CASE 도구간의 통신이 이루어진다.

(4) ALF-based IPSE

ALF-based IPSE(Integrated Project Support Environment) 시스템은 유럽지역의 ESPRIT 프로젝트의 일환으로 진행되고 있는 프로세스 중심의 통합 프로젝트 소프트웨어공학 환경(IPSE) 도구의 구현을 목적으로 한다. 이 시스템은 자료의 통합을 위한 방법으로 PCTE의 OMS를 근간으로 하고 프로세스의 행위적인 면의 표현을 위해 트리거 기법을 사용한다. 프로세스 모델인 MASP(Model for Assisted Software Processes)는 MASP/DL 언어에 의해 프로세스를 표현한다.

이 시스템의 특징은 소프트웨어 개발자가 자기의 해야할 작업이 어떻게 진행되는지 모르는 경우에 그 프로세스에 대한 설명이나 지침을 제공하는 기능을 가지고 있다. 또한 여러 명의 개발자, 프로젝트 관리자, 혹은 품질관리자들의 공동작업 및 팀단위 작업이 가능하도록 하는 MASP 관리 시스템이 있다.

(5) Weaver

이 시스템은 페트리네트를 이용한 프로세스 모델링 방법을 제공하며, 사용자는 그가 수행해야 할 다른 태스크를 접근하기 위해 Agenda 불리는 인터페이스를 통해서 작업공간에 있는 프로세스와 관련된 도구와 객체(자료나 산출물)를 사용할 수 있게 만든 것으로 프로세스모델 기반 개발환경(PSEE) 도구이다.

프로세스 모델의 표현에 사용되는 언어는 CP(Cooperative Procedure)이며 이는 CP 번역기에 의해 해석된다. 서로 다른 사용자를 위한 작업공간과의 통신을 통하여 여러 사람이 이용이 가능하도록 BMS(Broadcast Message System)에 의한 통신채널을 이용한 방법을 사용하고 고유의 사건(event) 표기방법에 의한 협조체계가 이루어지고 있지만 데이터 공유에 대한 지원체계가 없는 것이 단점이다.

(6) Matisee

이 시스템은 팀단위의 협력작업을 할 수 있는 프로세스 중심의 개발환경지원 도구로서 작업, 사람, 기록문서, 소스코드 등을 객체로서 데이터베이스에 저장하고 이들과 연관된 프로세스를 사용자 인터페이스를 통하여 저장 및 검색이 가능하도록 만든 도구이다. 또한 팀협력 작업과 구성관리를 위하여 규칙(rule)을 정의하고 이를 저장하기 위해 규칙베이스를 가지고 있다.

Matisee는 프로세스 모델로서 객체형태의 frame으로서 프로세스를 표현하고, frame은 객체지향 DBMS인 IRIS에 저장되고 객체편집기(object editor)와 객체주사기(object browser)를 통하여 수정 및 검색이 이루어진다. 규칙베이스인 workshop은 객체와 관련도구를 규정하고, 효율적으로 객체의 저장 및 검색을 위한 규칙이 들어있다. 이 시스템은 규칙에 의한 객체관련 도구들을 BMS(Broadcast Message System)에 의해서 호출하게 된다.

2.2 프로세스 중심의 CASE 도구들의 비교분석

통합 CASE 도구를 구축하기 위해서는 자료 및 도구의 통합이 이루어져한다. 즉 통합 CASE 도구는 정적인 정보 및 동적인 정보를 다룰 수 있어야 한다. 위에서 살펴본 프로세스 중심의 CASE 도구들의 특징으로부터 〈표 1〉과 같은 프로세스 중심 CASE 도구들의 비교분석표를 작성하였다.

〈표 1〉을 살펴보면 자료의 통합측면에서는 자료가 중복되는 것을 막고 도구들 간의 자료공유가 용이해지도록 정보저장소(Repository)를 구축하여 프로세스 관련 산출물을 저장하고 관리한

다. 대부분의 시스템이 정보저장소의 메타모델로서 ER(Entity Relation) 모델 혹은 확장된 ER 모델과 객체지향 모델을 사용한다. 또한 정보저장소간의 정보교환과 여러가지 도구를 함께 사용할 수는 도구의 통합이 가능하도록 표준화된 정보저장소를 이용하고자 하는 노력이 있다. 객체지향 모델을 이용하는 PCTE(Portable Common Tool Environment)와 확장된 ER 모델을 이용하는 IRDS(Information Resource Dictionary System) 등이 그 예이다.

〈표 1〉 프로세스 중심 CASE 도구들의 비교분석
table 1\ comparing & analizing of process-centered CASE tool

구분	ISTAR	SLCSE	Adele	ALF	Weaver	Matisse
메타모델	ER 모델	ER 모델	Extended ER 모델	PCTE (OMS)	X	OO 모델
프로세스 모델	coarse grain	coarse grain	fine-grain	fine grain	fine-grain	medium-grain
프로세스 표현방법	contract 계층 구조	X	Adele 언어 (event-trigger)	MASP /DL (event-trigger)	CP언어 (페트리 네트 기반)	규칙 (rule)
도구간의 통신	독립적 전달 (LAN)	X	BMS	X	BMS	BMS
DBMS	상용 DBMS	상용 DBMS	Adele-DB	X	X	OO DBMS (IRIS)
사용자 인터페이스	공통 (text, form)	공용 interface	GUI	GUI	Agenda	OO editor, browser
안내 및 설명기능	X	X	X	O	X	X
팀 단위 작업	X	X	X	O	X	O
구성관리 기능제공	O	X	O	X	X	O

동적인 측면에서는 프로세스들의 수행을 자동화하거나 사용자에게 설명해주는 기능을 갖기 위한 메커니즘으로서 트리거링 기법을 이용하는데 이는 프로세스의 행위를 잘 표현하기 위해 고안된 언어나 규칙기반(rule-based) 언어로 표현함을 알 수 있었다.

사용자 측면에서 살펴보면 단위 사용자의 개발관리를 지원하는 도구에서 다수 사용자가 동시에 사용이 가능하거나 팀단위로 협동해서 프로젝트 수행이 가능한 시스템, 더 나아가서는 BMS(Broadcast Message System)에 의한 CASE 도구간의 통신에 의한 정보를 공유하고, 서로 다른 제품의 도구들의 사용이 가능하게 하는 연합 개발지원환경 시스템을 구축하고 있다.

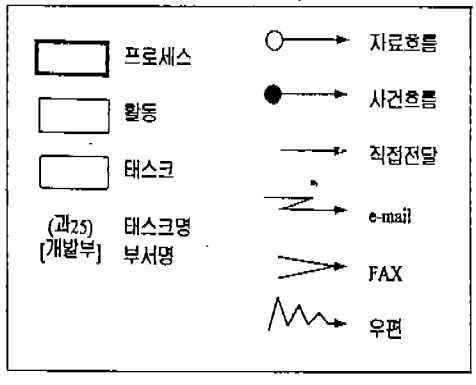
3. PM-CASE(Process Model based CASE) 도구의 구축

기존의 통합 CASE에 대한 연구는 주로 도구들의 통합과 자료에 대한 효과적인 통합이 연구되고 있지만, 이들 도구 및 자료들이 소프트웨어 생명주기 프로세스의 어떤 작업과 연관이 되어있는지 알 수 없기 때문에 각각의 독립적인 도구에 들어가서 작업을 하고 그곳에 결과를 저장하게 된다. 따라서 실제 프로세스 수행 중에 필요로 하는 도구의 호출 및 필요한 산출물(예를 들면 요구명세서, 프로그램 코드 등)을 참조하거나 수정 작성이 용이하지 못하다. 따라서 위에서 살펴본 여러 가지 프로세스 중심 CASE 도구의 비교분석을 통하여 효과적인 통합 CASE를 구축하기 방안으로 〈표 2〉와 같은 프로세스 모델 표기법에 의해 설계된 프로세스 모델을 중심으로 한 PM-CASE 도구를 구축하여 통합 CASE 도구로 발전시키고자 한다.

3.1 소프트웨어 프로세스 모델 표기법

프로세스 모델을 위한 프로세스의 구성요소는 ISO 소프트웨어 생명주기 표준안[28]에서 계층

〈표 2〉 소프트웨어 프로세스 모델 표기법
〈table 2〉 a legend of software process model



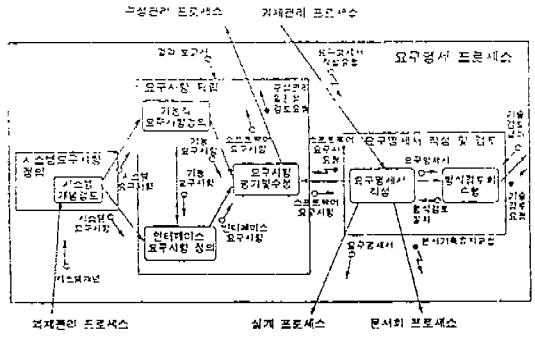
[illegible]의한 [illegible]스, 활동, 태스크를 구체로 [illegible]로 [illegible] 활동과 태스크의 관계 및 [illegible]의 흐[illegible] 는 표시한다. 이 모델의 표[illegible] 표 2 [illegible]다. 기본적으로 객체인 프[illegible] 활동스[illegible] 적으로 태스크는 표시되거[illegible] 현으[illegible] 다. 또한 단일 다이아그[illegible] 살표[illegible] 의 검은 원에 의하여 데이[illegible] 흐름[illegible] 하여 같이 표시할 수 있[illegible] 살표의 [illegible] 따라 프로세스의 전달 수[illegible] 하고 [illegible], 활동 및 태스크의 담당[illegible] [illegible] 도시하고, ( )내에 태스[illegible] 시한다.

[illegible])은 요[illegible] 프로세스와 설계 프로세[illegible]이스를 [illegible]는 다이아그램인데 한 프[illegible]는 [illegible] 로세스의 세부적인 활동과 [illegible] 없고 오직 인터페이스에 [illegible] 태스크[illegible] 달은 되므로 도형에는 외부 [illegible] 세서 보[illegible]sible) 태스크들은 경계선 [illegible], 인터[illegible]는 화살표로 표현하되 화살[illegible]출(invoke)의 의미를 가지고 그 외에 [illegible]나 사[illegible]ment)의 흐름을 표현한다. [illegible] 사료라 할[illegible] 소프트웨어 프로세스들에서 [illegible] 산출들은 [illegible]미한다. 또한 자료나 사건의 [illegible]산물를 다로[illegible] 도시함으로써 전달수단을 표시[illegible] 수 있고, 프로세스, 활동 또는 태스크들의 활[illegible]체를 구체적으로 모델에 명시하여 조직 내에서의 업무담당을 모델에 표현함으로써 조직적인 면에서의 프로세스를 표현할 수 있다.

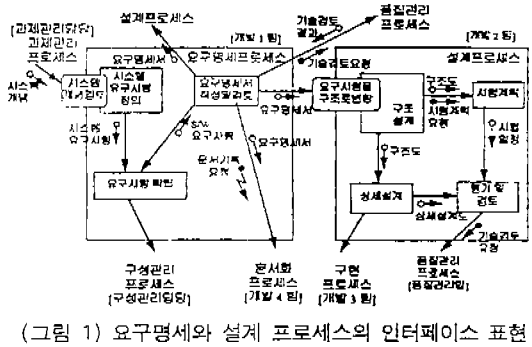
태스크의 표기를 나타내는 모서리가 둥근 사각형에 안에는 태스크 수행기능 및 상태를 표시하고, ( )에 태스크명을 표시함으로써 자세한 서술을 찾아볼 수 있도록 한다. 또한 자료흐름을 표시하여줌으로써 기능을 수행하기 위해 필요한 입력사항과 기능수행후의 산출물을 나타내고 있으므로 기능적인 관점에서의 프로세스를 잘 표현하고 있다. 또한 사건의 흐름의 표기방법에 의해서 어떤 요구가 있든지, 어떤 사건이 발생할 경우 태스크가 수행되는가를 알 수 있도록 함으로써 행위적인 관점에서의 프로세스 표현을 지원한다. 이와 같이 태스크는 기능 및 상태를 자료 및 사건의 흐름을 나타내는 두 가지의 구분되는 화살표에 의한 표현이 가능하도록 하였다. (그림 2). (그림 3)은 요구명세 프로세스 및 설계 프로세스를 프로세스 모델로 표현한 예이다.

3.2 PM-CASE 도구의 기본원리 및 구조

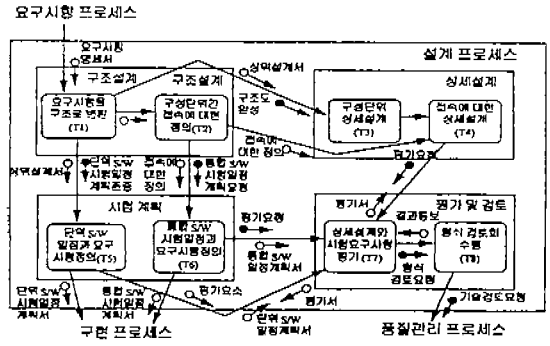
논문에서 제시한 표기법에 의한 프로세스 모델은 소프트웨어 생명주기 각 단계에서 이루어지는



(그림 2) 요구명세 프로세스
(Fig. 2) Requirement Specification Process



(그림 1) 요구명세와 설계 프로세스의 인터페이스 표현
(Fig. 1) an expression of the interface between Requirement Specification Process and Design Process



(그림 3) 설계 프로세스
(Fig. 3) Design Process

프로세스에 대한 이해와 관리를 위해 잘 활용될 수 있다. 개발자는 프로세스 모델을 보고 자기가 수행할 태스크와 그와 관련된 태스크의 인터페이스를 이해하고, 관리자는 전체 프로세스를 관리하고 통제하는데 활용이 가능하다.

PM-CASE 도구는 프로세스 모델을 중심으로 소프트웨어 생명주기 전반에 걸친 프로세스의 각 태스크를 수행하는데 필요한 자원, 산출물, 도구, 적용할 방법론, 제약사항, 담당자, 역할, 태스크 수행의 예상되는 시작시점과 종료시점을 나타내고, 태스크의 수행진도 등의 태스크 관련속성을 이용하여 도구 및 자원의 통합을 이루고자 하였다.

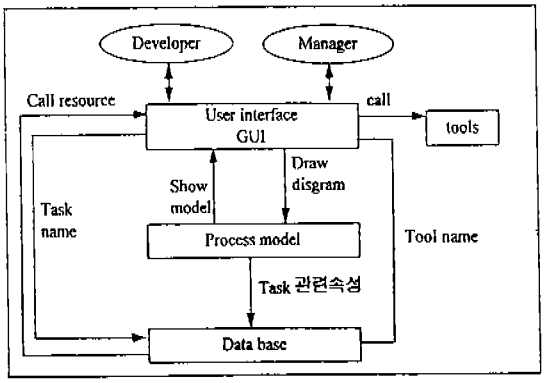
프로젝트 계획단계에서 우선, 프로세스 모델링에 의한 프로세스 모델을 구축하고, 이 모델을 다이아그램으로 작성할 때 프로세스의 각 태스크들에 대한 관련속성들을 정보저장소에 저장한다. 그런 후, 실제 프로젝트 수행시 태스크를 맡은 사람은 프로세스 모델을 보고서 자기가 지금 수행할 태스크를 선택하여 이 태스크에 연관된 도구, 필요한 자원, 산출물, 방법론, 제약사항, 기간 등을 참조할 수 있으며, PM-CASE 도구를 통해 태스크 수행자는 필요한 도구를 호출해서 사용할 수 있다. 또한 필요한 자료를 호출해서 참조할 수 있으며, 산출물을 작성할 수 있다. 이와 같이 태스크 수행자는 필요한 도구들을 따로 사용하거나 필요한 자원이 어디에 위치해 있는지 찾을 필요없이 도구들을 통합적으로 사용이 가능하며, 자료들을 정보저장소에 저장하고, 검색 및 수정이 용이하도록 설계하여 자료의 일관성 및 사용의 편리성, 공유가 가능하도록 하였다.

또한 프로젝트 관리자는 프로세스 모델을 통해서 프로세스를 이해하고 관리가 용이하며 PM-CASE 도구로부터 태스크 수행을 위한 담당자, 태스크 시작시점 및 종료시점, 진행 상태 등을 참조하므로 써 프로젝트 관리에 도움을 받을 수 있다.

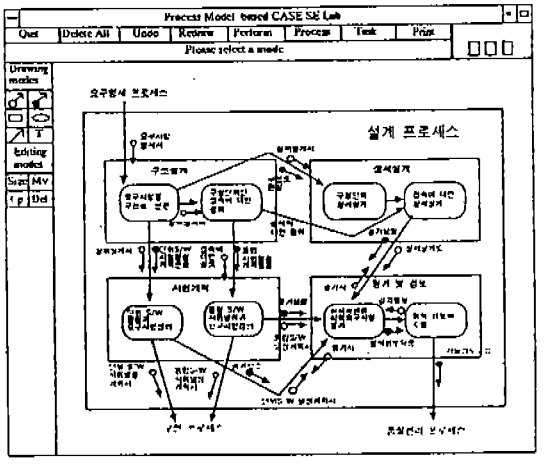
이 PM-CASE 도구의 구조는 (그림 4)와 같은데 이를 구현하기 위하여 그래픽 사용자 인터페이스를 설계하고 태스크 관련속성들의 저장을 위해 데이터베이스를 구축하여 정보저장소로 이용한다. 또한 태스크 수행시 필요한 도구 및 자료의 호출이 가능하도록 사용자 인터페이스 상에서 버튼방식으로 호출이 가능하도록 설계하여 필요한 자료는 데이터베이스로부터 가져오고 도구들은 이름에 의한 다른 도구의 호출하여 사용이 가능하도록 한다. 각 태스크 수행에 필요한 도구들은 PM-CASE 도구와 같은 그래픽 사용자 인터페이스를 사용하는 도구를 독립적으로 개발하여 PM-CASE 도구 상에서 통합하여 사용할 수 있다.

3.3 PM-CASE 도구 구축을 위한 시제품 설계

(그림 4)와 같은 구조를 가진 PM-CASE(Process Model based CASE) 도구의 구축을 위한 시제품을 설계하였다. 프로세스 모델에 대한 다



(그림 4) PM-CASE 도구 구조
(Fig. 4) PM-CASE tool architecture

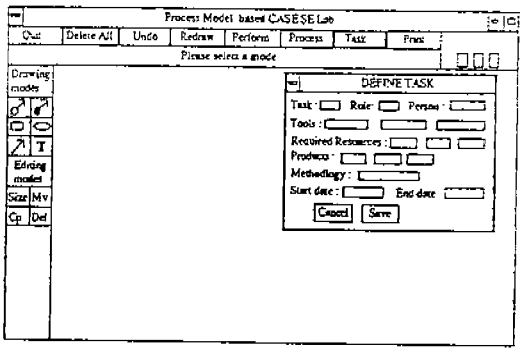


(그림 5) PM-CASE 주화면
(Fig. 5) PM-CASE main screen

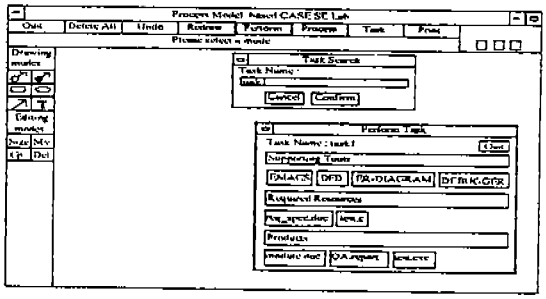
이아그램 작성 및 태스크 관련속성의 입력과 태스크 수행시 필요한 태스크 관련 도구 및 자료의 호출을 위한 사용자 인터페이스 부분을 (그림 5)와 같이 설계하였다.

이와 같이 설계된 PM-CASE는 (그림 5)의 화면 좌측에 있는 drawing 모드의 아이콘이나 editing 모드 버튼을 이용하여 (그림 3)의 설계 프로세스 모델 예와 같은 프로세스 모델을 설계할 수 있다. 또한 프로세스내의 각 태스크에 대하여 태스크 관련사항을 입력하기 위해서 (그림 4)에서 윗부분에 있는 관넬의 Task를 선택하게 되면 (그림 6)와 같은 pop-up 화면이 나타나게 되며, 여기서 입력하게 되는 것들은 정보저장소로 저장된다.

프로세스 모델을 작성하면서 태스크 관련사항은 함께 입력이 되며, 후에 개발자가 작업을 하고자 할 때는 (그림 4)의 윗부분 관넬에 있는 Process를 선택하면 이미 작성된 프로세스 모델의 해당 프로세스에 대한 다이아그램을 불러오게 되



(그림 6) Task 입력화면
(Fig. 6) Task input screen



(그림 7) Task 수행화면
(Fig. 7) Task execution screen

며, 이때 개발자는 이 프로세스 모델을 보고 자기가 수행해야 할 태스크를 파악하여, 위 관넬의 Perform 이라고 하는 버튼을 선택해서 태스크 이름을 입력하면 이 태스크 이름에 의해서 정보저장소내의 태스크 관련사항을 검색하게 되며 (그림 7)과 같은 화면이 나타나게 된다.

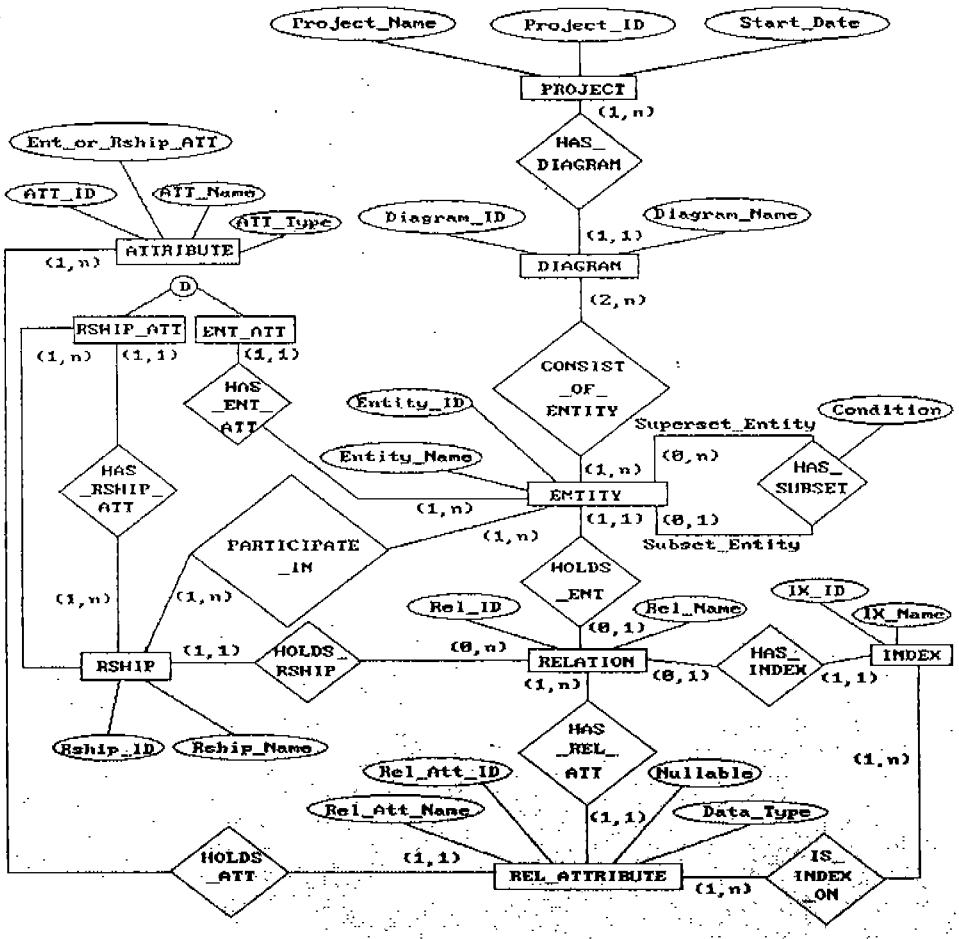
(그림 7)에서 보듯이 태스크 수행자는 태스크 수행시 필요한 도구 및 자원에 관한 이름이 버튼으로 만들어져 있어 화면상에서 쉽게 마우스로 호출이 가능하며, 개발방법론 및 태스크 수행기간을 참조하면서 작업을 할 수 있고, 작업 종료시 자기가 한 태스크 작업의 진도를 입력하여 줌에 관리자가 각 태스크의 진도사항 파악이 용이할 수 있도록 한다. 여기서 필요한 도구 호출은 기존에 있는 도구들을 연결해서 사용이 가능하고, 없는 도구들은 독립적으로 만들거나 구입해서 이 PM-CASE 도구에 연결해서 사용한다. 또한 데이터의 일치성 및 공유를 위하여 각 태스크에서 만들어진 산출물은 정보저장소에 보관된다.

3.4 PM-CASE 도구의 정보저장소(repository)의 설계

PM-CASE 도구는 소프트웨어 생명주기 전반에 걸친 프로세스와 연관된 도구, 필요한 자원, 산출물, 방법론, 제약사항, 기간 등의 정보를 정보저장소에 저장하고, 이를 활용하여 태스크 수행시 필요한 도구나 자료 등을 호출하는데 이를 위한 데이타베이스 구축은 관계형 데이타베이스를 사용하여 시제품을 제작하였는데 보다 효과적인 소프트웨어 개발관리 지원 도구가 되기 위해서는 정보저장소의 구축이 필수적이라 할 수 있다.

현재 정보저장소의 구축방법으로 많이 연구되고 있는 것중에는 PCTE(Portable Common Tool Environment)와 EER(Extended Entity Relation) 모델을 이용한 IRDS 등이 있다. PCTE는 UNIX 환경 하에서 PCTE 모듈이 서로 호환성이 있고 용이하게 결합해서 사용할 수 있는 것으로서 이는 객체지향 관리모델인 OMS(Object Management System)을 이용한다. 이때 객체는 소스코드 화일, 실행 프로그램, 문서 등과 같이 개발에 필요한 자료 등이 되며 이들을 공유하고 쉽

게 사용할 수 있도록 OMS는 데이타 정의 기능과 데이타 관리 기능이 있는데 이를 이용해서 효과적인 데이타 관리를 위한 개념적인 단계의 스키마를 제공하게 되며 이들 정의된 내용이 C 코드와 함께 UNIX 환경에서 수행이 가능하게 된다. PCTE는 PACT, EAST(Eureka Advanced Software Technology)[8] 등과 같은 유럽의 통합적인 개발관리 환경 구축을 위한 도구를 만드



(그림 8) 정보저장소를 구축을 위한 EER 다이아그램
(Fig. 8) EER diagram for a repository

는 프로젝트에서 정보저장소 구축 수단으로 만들어진 것으로 CASE 도구간의 자료 공유 및 교환을 위한 표준인 정보저장소 구축 방법으로 효과적이라 할 수 있다.

또 다른 방법으로 EER 모델을 활용한 관계형 데이타베이스를 이용하는 방법[7]으로 PM-CASE 도구는 ORACLE 관계형 데이타베이스를 이용하여 시제품을 제작하였는데 정보저장소의 구축방법으로서 이 방법을 이용한다. 자료의 효과적인 공유와 검색을 위해서는 도구에서 사용하는 자료의 개념적인 단계에서 정의로서 (그림 8)과 같은 EER 다이아그램을 이용하고 〈표 3〉은 그 데이타베이스 테이블에 대한 정의로서 도구에서 사용하는 자료가 이 테이블에 저장되어 관계형 데이타베이스의 장점인 자료의 공유 및 검색이 가능해진다. 이와같이 PM-CASE 도구 시제품은 효과적인 정보저장소의 구축을 위해 도구상에서 그리는 다이아그램이 정의된 관계 및 속성으로 데이타베이스에 저장되도록 한다.

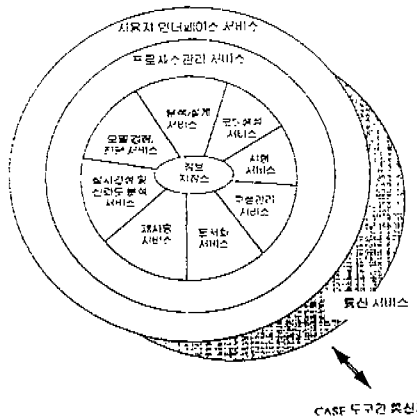
〈표 3〉 정보저장소 구축을 위한 관계 및 속성 정의
〈table 3〉 a relation & attribute definition for repository

PROJECT={Project-ID, Project-Name, Start-Date}
DIAGRAM={Diagram-ID, Diagram-Name, Project-id}
ENTITY={Entity-ID, Entity-Name, Diagram-ID}
RSHIP={Rship-ID, Rship-Name}
ATTRIBUTE={Att-ID, Ent-or-Rship-Att, Att-Name, Att-Type, Ent-or-Rship-ID}
PARTICIPATES-IN={Rship-ID, Entity-ID, Role-Number, Role-name, Min-Card, Max-Card, Weakness}
HAS-SUBSET={Superset-Ent-ID, Subset-ID, Subset-Ent-ID, Condition}
RELATION={Rel-ID, Rel-Name}
INDEX={IX-ID, IX-Name, Rel-ID}
KEY-INDEX={IX-ID, Rel-Att-ID}
REL-ATTRIBUTE={Rel-Att-ID, Rel-Att-Name, Nullable, Data-Type, Rel-ID, Att-ID}
HOLDS-ENTITY={Rel-ID, Entity-ID}
HOLDS-RSHIP={Rel-ID, Rship-ID}

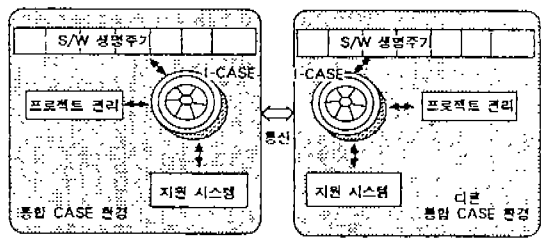
4. 시제품으로 개발한 PM-CASE 도구의 발전방향

시제품으로 만들어진 PM-CASE 도구가 효과적인 소프트웨어 개발관리를 지원하는 통합 CASE 도구가 되기 위해서는 몇까지 점을 보안하여 PM-CASE 도구는 궁극적으로는 (그림 9)와 같은 구조를 가지는 통합 CASE 및 (그림 10)과 같은 통합적인 소프트웨어 개발관리 지원 환경을 제공하는 발전된 시스템으로 나아가야 한다.

이와 같이 PM-CASE 도구를 활용하여 발전된 통합 CASE 도구를 구축하기 위해서는 표준화된 정보저장소를 이용하여 다른 형태의 CASE 도구들과도 자료를 공유할 수 있도록 PCTE나 IRDS를 이용한다. 그리고 각 생명주기 단계의 프로세스를 지원하는 도구들을 점차적으로 개발하여 툴셋으로 제공하고, 다른 정보저장소를 가진 개발환경 도구들의 호출이 가능하도록 BMS (Broadcast Message System) 기능을 추가한다. 나아가서는 분산 시스템에 의해 정보저장소를 분



(그림 9) 통합 CASE의 구조
(Fig. 9) an architecture of integrated CASE



(그림 10) 소프트웨어 개발관리를 지원하는 통합 CASE 구축 환경
(Fig. 10) a building environment of integrated CASE tool to support software development and management

리 저장 및 공통 이용이 가능한 시스템을 구축한다.

또한 팀단위 작업이 가능하도록 팀의 역할에 따라 필요한 도구들을 묶어서 함께 이용이 가능하도록 한다. 그리고 다수의 사용자가 동시에 자료의 검색 및 수정이 가능하고 도구를 같이 사용하기 위한 협동체계 시스템을 구성한다. 다른 기능으로는 산출물의 버전관리 기능을 갖추어서 구성관리가 가능하도록 하며, 프로세스의 행위를 잘 표현할 수 있는 규칙(rule)이나 event-trigger 방식의 언어를 고안하여 향후 프로세스의 자동화 및 안내, 설명기능을 갖추도록 한다.

5. 결 론

PM-CASE 도구는 개발자가 자기가 수행해야 할 태스크 및 다른 인터페이스가 되는 태스크를 프로세스 모델을 통하여 쉽게 알 수 있으며, 태스크 수행시 필요한 도구 및 자료를 따로 찾아서 수행할 필요없이 사용자 인터페이스상의 버튼방식에 의하여 호출이 가능하도록 설계하였다.

본 논문에서는 여러 가지 프로세스 모델 중심의 CASE 도구에 대한 조사 및 비교분석을 통하여 PSEE 도구들이 효과적인 통합 CASE 도구의 구축을 위해 어떤 기능을 갖추어야할지를 알아보았고, PM-CASE 도구의 기본개념, 구조 및 동작원리에 대한 기술을 통하여 효과적인 통합 CASE 도구의 구축을 위한 방법을 제시하였다.

PM-CASE 도구는 프로세스 모델을 설계하고 프로세스 수행을 지원하는 도구로서 소프트웨어 생명주기 전 단계를 지원하는 도구의 효과적인 통합과 정보저장소의 사용으로 자료 및 산출물의 통합이 가능하도록 설계된 시제품으로서 향후 통합 CASE가 지향하는 IPSE(Integrated Project Support Environment) 도구 구축을 위한 연구에 초석이 되리라 본다.

시제품으로 만든 PM-CASE 도구가 실제적인 통합 CASE가 되기 위해서는 여러가지로 아직 구현되지 못한 부분이 많이 있는데, 이는 PM-CASE 도구를 통합 CASE 도구 발전시키기 위한 방안에서 제시된 방향으로 계속적인 연구와 구현을 위한 노력이 경주되어야 할 것이다.

참 고 문 헌

[1] 정기원, 이광용, "통합 CASE의 기술현황 및 발전방향", 정보과학회지, 제1권 제2호, 1994.3.

[2] 정기원, 조병호, "객체중심 소프트웨어 프로세스 모델링 방법의 제시", Journal of the Industrial Technology Research Institute, Soongsil University, Vol.23, 1993., pp.223-233.

[3] Alan W. Brown, Peter H. Feiler and Kurt C. Wallnau, "Past and Future Models of CASE Integration", Proceedings Fifth International Workshop on Computer-Aided Software Engineering, July. 1992, pp.36-45.

[4] Bill Curtis, Marc I. Kellner and Jim Over, "Process Modeling", Communication of the ACM, Sep. 1992, pp.75-88.

[5] DoD-STD-1838A, Common Ada Programming Environment(APSE) Interface Set(CAIS), Revision A. U.S. Department of Defence, 1988.

[6] Dowson, M., "ISTAR An Integrated Project Support Environment", Proceedings of 2nd ACM SIGSOFT/SIGPLAN Software Engineering Symposium on Practical Software Environments, December. 1986.

[7] Eko K. Budiardjo, "A Custodial Application Generator for use in a CASE Tool Environment", Proceedings Sixth International Workshop on Computer-Aided Software Engineering, pp.326-347, 1993.

[8] European Computer Manufacturer's Association(ECMA). A Reference Model for Computer Assisted Software Engineering Environment Frameworks. ECMA/TC33/TGRM, 1999.

[9] IEEE STD 1074-1991,IEEE Standard for Developing Software Life Cycle Processes, 1992.

[10] ISO/IEC JTC1/SC7/WG7 N1, Information Technology Software Life Cycle Processes, Aug. 1991.

[11] Jacky Estublier, Noureddine Belkhatir, "Process centered SEE and Adele", Proceedings Fifth International Workshop on Computer-Aided Software Engineering, July. 6-10, 1992, pp.156-165.

[12] J. C. Derniame, C. Godart, V. Gruhn, J. Lonchamp, "Process-Centered IPSEs in ALF", Proceedings Fifth International Workshop on Computer-Aided Software Engineering, July. 6-10, 1992, pp.179-187.

[13] Maryse Bourdon, "Building Process Models using PROCESS WEAVER : a Progressive Approach", Proceedings of the 8th International SOFTWARE PROCESS WORKSHOP, March. 2-5, 1993, pp.40-45.

[14] Military Standard DOD-STD-2167A. Defence System Software Development, Feb. 1988.

[15] Minder Chen, Ronald J. Norman, "A Framework for Integrated CASE", IEEE Software, Vol 9, No 2, March. 1992, pp. 9-13.

[16] Pankaj K. Garg and Thuan Q. Pham, "Process Modeling in Matisee : A Team Programming Environment", Proceedings of the 8th International SOFTWARE PROCESS WORKSHOP, March. 2-5, 1993, pp.81-84.

[17] Peiwei Mi, Walt Scacchi, "Process Integration in CASE Environments", IEEE Software, Vol. 9, No. 2, March. 1992, pp.45-53.

[18] Steve Sibbald, Terry Shepard and Colin Wortley, "Software Process Enaction with VPL", Proceedings Fifth International Workshop on Computer-Aided Software Engineering, July. 1992, pp.191-199.

[19] Surlich, T. The Software Life Cycle Support Environment(SLCSE) Computer-Based Framework for Developing Software Systems, Proceeding of ACM SIGSOFT/SIGPLAN Software Engineering Symposium on Practical Software Engineering Environments, Boston, MA, 1988.

[20] Teitelman, W., Masinter, M. "The Interlisp Programming Environment", IEEE Computer , 1981.

[21] William Harrison and Harold Ossher, "Integrating Coarse-Grained and Fine-Grained Tool Integration", Proceedings Fifth International Workshop on Computer-Aided Software Engineering, July. 1992, pp.23-35.

조 병 호

1983년 인하대학교 전자공학과 졸업
1989년 뉴욕공대 전산과학과 석사학위 취득
1990년~92년 한국 오라클 근무
1995년 숭실대학교 전자계산학과 박사과정 수료
1994년~95년 한국과학기술원 교통산업연구센터 연구원
관심분야 : 소프트웨어 공학, 실시간 시스템, 데이타베이스, 인공지능임.

김 태 달

1973년~79년 숭실대학교 공과대학 전자계산학과 졸업(학사)
1990년~92년 숭실대학교 정보과학대학원 전자계산학과 졸업(석사)
1993년~95년 현재 숭실대학교 전자계산학과(박사과정)
1986년 국가기술자격고시합격(정보처리기술사)
1978년~89년 쌍용컴퓨터 근무(시스템프로그래머, GIS)
1989년~91년 현대전자 근무(시스템소프트웨어 개발부장)
1991년~94년 도로교통안전협회 교통과학원(수석연구원)
1995년~현재 도로교통안전협회(전산실장)
관심분야 : 과제관리, 시스템엔지니어링, 시스템통합, 실시간 처리 및 시뮬레이션