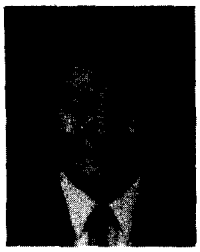


CASE 道具를 이용한 情報시스템 具現 事例

Case study on implementation of information system using CASE tool

金 麗 腥*
Kim, Ryo Sung

1. 머리말

사회 각분야에 컴퓨터가 많이 활용되면서 소프트웨어의 중요성이 날로 증대되고 있다. 그리고 정보시스템에 대한 사용자의 요구가 다양해지고 방대해지고 있어 소프트웨어에 투자되는 비용이 급격히 증가하고 있다. 따라서 이제는 어떻게 하면 개발비용과 유지보수 비용을 절감하면서 양질의 소프트웨어를 활용할 수 있는가 하는 것이 중요한 문제로 부상하고 있으며, 이에 따라 소프트웨어를 개발지원하는 도구에 대한 관심이 점차 높아지고 있다.

이러한 인식하에 필자는 증권감독원 업무전산화 초기단계부터 소프트웨어의 개발 및 유지보수를 능률적으로 해결하는 방안으로 한국전산원이 보급하고 있는 관리기법/1을 적용하는 한편 이를 지원하는 소프트웨어 개발도구(CASE Tool)를 도입하여 정보시스템 구축에 적용하여 보았다.

관리기법/1과 CASE 도구같은 새로운 기술과 방법론은 여러가지 상황에서 시행착오를 거쳐가면서 보완적용해야만 그 성과를 신뢰할 수 있을 것이므로 이런 면에서 보면 아직 업무개발이 진행중이고 또 개발된 소프트웨어를 유지보수해 본 경험이 적은 현재개발 적용사례를 소개하기에는 아직 때 이른 감이 없지는 않다.

그러나 다른 어떤 기관보다도 증권감독원이 먼저 전산업무개발에 관리기법/1과 개발도구를 적용하고 있고 그 동안의 개발과정으로 보아 나름대로는 성공적이라고 평가하고 있어서, 소프트웨어 개발도구(CASE Tool)와 같은 새로운 기술과 방법론에 관심을 가지고 있는 모든 분에게 조금이나마 도움이 되기를 기대하며 이글을 쓴다.

2. CASE의 개요(概要)

가. CASE는 이제까지 노동집약적인 수준에 머물러 왔던 소프트웨어 개발방식을 자동화함을 목표로 삼는다. CASE는 소프트웨어 개발전공정(Life Cycle)의 상당한 부분을 지원할 수 있도록 개발기법과 개발도구를 하나의 패키지(package)로 구성하여 개인용 컴퓨터나 워크스테이션에 탑재하여 제공하고 있다. CASE도구들은 기존의 소프트웨어 도구와는 달리 그래픽기능을 기본으로 하고 시스템 분석과 설계, 문서화 도구들과 부분적으로 코드를 생성하는 도구들을 포함한 것도 있다. 그리고 일부 CASE도구들은 개방방법론과 개발관리를 위한 도구도 제공하는 것이 있다.

나. CASE는 Computer Aided Software Engineering의 약어로서 소프트웨어 개발의 자동화라고 정의할 수 있다. 이는 CASE가 의

* 電子計算組織應用技術士. 證券監督院 電算業務室 室長. 情報處理分會 會長.

미하는 말 그대로 컴퓨터의 도움을 받아 소프트웨어 개발과정을 지원하기 위한 자동화된 도구들을 통합적으로 제공하는 것이라고 할 수 있다. CASE기술은 소프트웨어 도구와 방법론의 결합체이며 분석과 설계 작업뿐만 아니라 구현과 유지보수까지를 자동화함으로써 소프트웨어개발 생산성과 신뢰성을 제고시키고자 한다.

다. CASE 도구와 환경

CASE=소프트웨어공학+컴퓨터
=소프트웨어 공학기법의 컴퓨터화
CASE도구=CASE를 위한 소프트웨어 개발 도구
=소프트웨어개발/유지보수를 지원하는 소프트웨어
CASE환경=복수의 CASE기능이 사용되는 것처럼 기반이 정비된 상태, 또는 그 소프트웨어 전체

3. CASE의 목표(目標)

- 통합화된 도구에 의해 생명주기(Life Cycle) 전체의 자동화
- 소프트웨어개발·유지보수의 생산성 향상
- 소프트웨어 제품의 품질 향상

CASE의 의의는 소프트웨어 개발도구와 방법론을 결합시켜 [그림 1]과 같이 자동화하는데 있다. 기존의 도구들이 주로 프로그래밍 단계에 중점을 둔 반면, CASE에서는 지원단계를 소프트웨어 생명주기 전과정으로 확대하여 설정하고 있다. 즉, 소프트웨어 개발초기의 분석·설계과정에서, 개발·유지보수단계에 이르기까지 확대하여 일관성있는 도구와 방법론을 통합적으로 지원하자는 것이다.

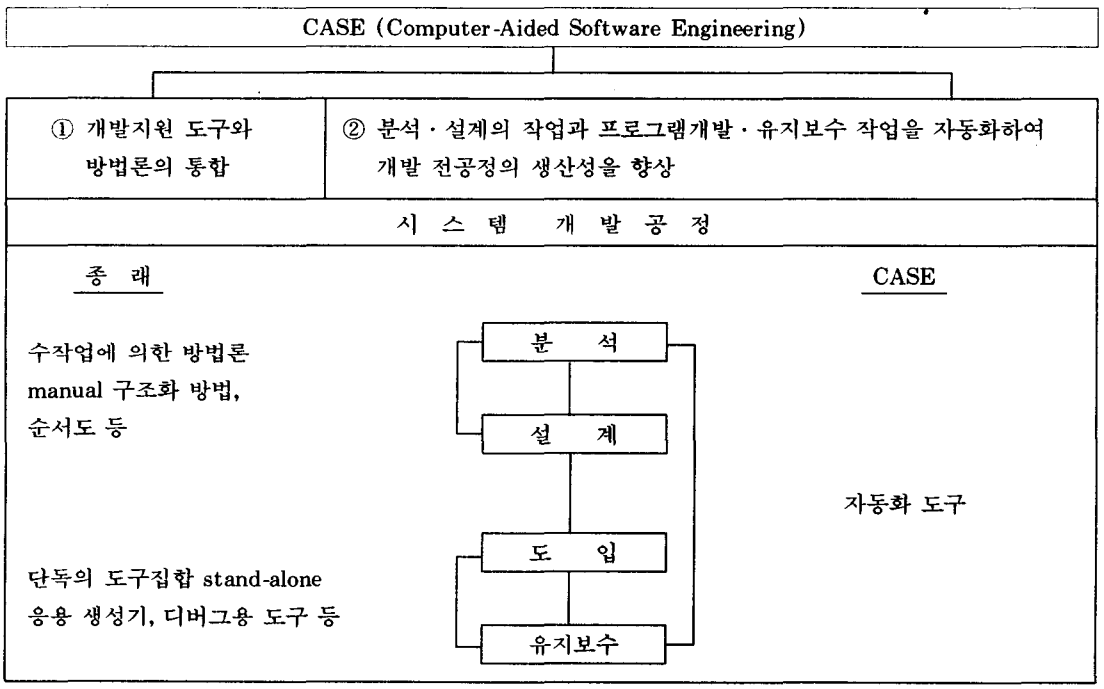


그림 1. 소프트웨의 자동화

4. CASE의 원천기술

CASE 도구에 직접적인 영향을 준 소프트웨어 기술은 다음과 같은 것이다.

가. 구조적 기법

구조적 분석기법은 하향식(Top-down)과 기능분할(functional decomposition)방법 등을 이용하여 시스템의 요구를 정의하는 모델링 기법이다. 개발하고자 하는 정보시스템을 자료흐름도(Data Flow Diagram), 자료사전(Data Dictionary) 및 소분류 명세(minispec) 등을 이용하여 명세한다. 구조적 분석기법은 도형적 모델을 이용하고 있기 때문에 사용자가 시스템을 쉽게 이해할 수 있어서 분석가와 사용자간에 의사소통을 원활히할 수 있다는 장점이 있다. 구조적 분석기법은 로스(D. T. Ross)에 의하여 개발되었으며 게인/사슨(Gane/Sarson)과 디 마코/요든(DeMarco/Yourdon)에 의하여 발전된 형태가 널리 보급되어 있으며 Constantine/Jackson/Warnier Orr의 구조적 설계기법과 함께 널리 알려지고 있다.

나. PROTOTYPE 기법

이 기법은 試製品(prototype) 형태의 소프트웨어를 단기간에 제작하여 사용자에게 제시함으로써 사용자가 원하는 소프트웨어의 요구사항을 조기에 확정하고 개발을 진행하여 비용절감을 꾀하려는 시도이다. 이를 위하여 프로토타입용 프로그래밍언어, 스크린 layout, 보고서 설계를 위한 도구 등이 개발되고 있다.

다. End User Computing

시스템이 복잡화·거대화하므로 개발기간의 지연뿐만 아니라 개발도중에도 빈번히 발생되는 요구사항의 변경은 개발팀에 업무를 가중시켜 사용자가 필요한 시기에 소프트웨어 납품이 이루어지지 못하는 사례가 많아졌다. 따라서 사용자가 직접 참여하여 개발하는 방법인 Application Generator, non-procedural language, 제 4 세대 언어 및 데이타베이스를 쉽게 access하는 Query language 등이 개발·보급되고 있다. 이는 전문개발인을 통하지 않고 사용자가 직접 응용시스템을 개발함으로써 요구정의 과정에서 흔히 발생될 수 있는 정보교환의 오류를 배제하자는 것이 주목적이다. 그러나 간편하다는 장점은 있으나 자원의 효율적인 이용이나 사용에 한계가 있다는 단점을 가지고 있다.

라. Software Factory

소프트웨어 공장은 다양한 프로젝트에 적용하게 될 표준화된 도구, 개발기법 등 재사용을 전제로 한 mechanism의 하부구조이다. 소프트웨어 공장은 소프트웨어 제품을 제작하기 위한 조직이나 관리체계를 정비하고, 공장내에서 제작에 사용될 도구나 기법을 표준화한 다음 제작공정을 점진적으로 자동화해나가는 순서로 발전해 나아간다. 소프트웨어 공장의 효시는 일본의 SIGMA 프로젝트이며 그 이후 ESPRIT 프로젝트의 SFINX(Software Factory Integration and Experimentation), EAST(EUREKA Advanced software Factory), ESF (EUREKA Software Factory) 등이 그 뒤를 잇고 있다.

마. Sofware Reuse

품질이 입증된 소프트웨어를 재사용하면 중복 개발을 피할 수 있고 이로써 비용절감도 꾀하고 소프트웨어의 신뢰도도 높일 수 있다고 하겠다. 그러나 재사용이 효과적으로 이루어지기 위해서는 사전에 소프트웨어 제조방식을 표준화하여 필요한 부품들로 조립하는 소프트웨어 공장의 개념을 도입해야 하며 이는 막대한 초기 투자가 필요하다. 효과적인 재사용을 위하여는 소프트웨어 부품의 분류기술, 검색기술 그리고 부품간의 인터페이스를 단순화 할 수 있는 설계기술이 계속 연구되어야 한다.

바. 소프트웨어 개발환경개선

소프트웨어 개발환경은 개발기법, 개발도구 및 개발관리절차를 총괄하는 방법론을 핵심으로 하고 최적의 컴퓨터시스템, 주변환경 및 외적요인을 고려한 종합적인 개발지원 시스템을 뜻한다. 따라서 소프트웨어 개발환경에 관한 연구는 소프트웨어 개발작업의 자동화, 소프트

웨어 자체의 구조와 구성요소의 연구 그리고 형상관리(Configuration Management) 등 개발관리절차의 제반 기술적 지원을 연구하여 개선하는 것이다.

5. CASE의 구성요소

- 분석 · 설계도구(Diagramming tool)
- 정보저장도구(Centralized Repository)
- 사용자 작업도구(User Interface Generator)
- 코드 생성도구(Code Generator)
- 프로젝트 관리도구(Management Tool)

CASE는 생명주기(SDLC)의 모든 부분에 대해 프로세스와 개발도구들을 제공한다. CASE의 기술은 개발과정의 전반부, 즉 타당성 조사, 요구분석, 시스템 설계를 상위(upper) CASE 그리고 구현, 테스팅, 유지보수 등 후반부를 하위(lower) CASE라고 부르며 정보 저장소를 repository라고 한다. CASE 제품들의 구성요소나 도구들은 다음과 같다.

가. Diagramming 도구

다이어그램은 소프트웨어공학의 발전과 함께 그 궤(軌)를 같이해 왔다고 해도 과언이 아니다. 다이어그램은 S/W 명세서를 정의하고 설계를 표현하는 데 사용되며 설계된 결과를 코드로 구현하기 위한 청사진으로 제공된다. 또한 시스템을 분석 · 설계하는 경우에 명료하게 사고할 수 있도록 도움을 주며 여러 사람이 개발하는 경우에는 정보전달의 도구로도 사용된다. 그러나 이러한 다이어그램을 손으로 직접 그리는 것은 매우 지루하며 시간이 많이 소요되는 귀찮은 작업이다. 더구나 완성된 시스템에 변경요구가 있을 경우에는 이미 그려놓은 다이어그램을 다시 그려야만 하는 방대한 작업이 필요하기 때문에 실질적으로는 다이어그램이 갖는 장점을 완전히 살리지 못하고 있는 실정이다. 따라서 CASE 도구는 이러한 문제점을 해결하기 위하여 시스템 설계 다이어그램을 생성하는 데 있어 CAD 기법을 사용하여 다이어그램 생성의 자동화를 지원한다. CASE 도구들은 Yourdon/DeMarco, Gane/Sarson, Jackson, Warnier-Orr같은 수작업 지향적인 구조적인 기법뿐만 아니라 정보공학에 기반을 둔 James Martin 등 최근 기법에 이르기까지 다양한 개발방법론에 연관되어 있는 다이어그래밍 기법들을 지원하고 있다. 다이어그래밍도구는 CASE 구성요소 중 제일 먼저 관심이 모아져서 개발된 부분이며 문서화도구와 더불어 CASE라는 분야를 탄생시킨 모태의 역할을 했다고 말할 수 있다.

나. 설계분석기

소프트웨어 시스템의 품질은 명세서(Specification)에 의해서 좌우되며 명세서의 질은 그것의 완전성, 일치성 등에 따라 좌우된다. 따라서 소프트웨어 명세서의 오류를 검사하고 분석함으로써 품질이 우수한 소프트웨어를 생산할 수 있다. 설계분석기의 기능은 설계명세에 있어서 이와 같은 내부 불일치성, 모호성, 불완전성 등을 자동적으로 검사하는 것이다. 대부분의 CASE 제품들은 아직까지 극히 한정된 검증능력을 제공하고 있으나 앞으로는 이 검증능력이 빠른 속도로 개선될 것으로 예상된다.

다. Code Generator

코드생성기는 다양하고 일관성있는 명세서로부터 고수준의 프로그래밍 언어로 쓰여진 모듈화된 코드를 생성한다. 이미 수년 전부터 독립적으로 사용되던 코드 생성기를 많은 CASE 도구들이 다른 구성요소와 함께 통합시켜 나가고 있다. 이때 제기되는 중요한 기술적 사항은 upper CASE와 코드생성기간의 밀접한 연결(Inter Connection) 기술이라 하겠다. 이것은 개발과정의 모든 정보가 집약되어 있는 정보저장소인 repository와의 관계 그리고 Software specification과 관련된 각종 형식과 정의에 따른 자세한 사항들이 일치해야 하는 문제를 해결해야하는 것이다.

라. 정보저장소(Centralized Repository)

정보저장소는 SDLC 전 기간동안에 모아진 모든 시스템 정보가 집약되어 저장되는 곳으로서 각기 다른 개발팀 구성원에 의한 시스템 작업이 결합되고 분석되어 하나의 완전한 시스템으로 보관되는 곳이다. 정보저장소는 자료흐름도, 구조도, ER Diagram 등과 같이 추상화된 명세서를 저장하는 데이타베이스의 역할을 할 수 있어야 한다. CASE 정보저장소는 논리적으로 프로젝트 라이브러리와 사스템 모델로 구분할 수 있으며 그 내부의 자료가 개발단계별로 서로 이동할 수 있도록 연결이 필요하며 아울러 LAN과 같은 네트워크(network)에 의해서 편리성이 부가된다.

마. 試製品(prototype)제공

Prototyping 도구는 사용자의 요구사항을 조기에 확정하는 데 아주 유용한 역할을 한다. 따라서 대부분의 CASE tool은 prototype을 제공하는 방식을 택하고 있으며 이의 접근방식으로는 Rapid throughway prototyping 방식과 Evolutionary prototyping 방식이 활용되고 있다.

바. 프로젝트 관리 지원도구

시스템을 개발하는 데 있어 실제 기술적인 측면의 지원뿐만 아니라 프로젝트 관리자를 지원하는 관리도구의 자동화도 지원하는 CASE 도구가 있다. 이는 프로젝트 관리자로 하여금 S/W 프로젝트에 관하여 작업계획, 작업할당, 일정관리, 진도관리, 변경관리, 보고 및 통제 등의 관리작업을 향상시키도록 도와주며 CASE의 정보저장소를 Access하여 개발상태를 적기에 파악함으로써 프로젝트가 효율적으로 진행되도록 지원하는 역할도 한다. Foundation의 Method/1, CADRE의 Teamwork, Softlab의 MAESTRO와 AGS의 MULTI/CAM 등이 이런 류에 속한다.

사. 유지보수 도구

기존의 시스템을 새로이 문서화하거나 분석, Re-engineering 또는 Reverse-engineering을 하기 위한 도구이다. 이미 작성되어 있는 코드를 표준양식에 맞추어 재구성해주고, 이름을 통일하거나 프로그래밍 스타일을 다듬어 주는 역할을 하는 도구이다. 아직은 대부분의 CASE 도구들이 이 부분이 취약하므로 앞으로 많은 연구가 있을 것으로 기대된다.

6. CASE의 기능

- -그래픽(Graphic)편집기능
- -요구분석기능
- -정보저장기능(Centralized Repository 구성)
- -프로토타입(PROTOTYPE) 생성기능
- -코드 생성기능(Code Generator)
- -문법 검증기능
- -문서 작성기능
- -사용자 인터페이스(Interface)기능
- -프로젝트 관리기능(Management Tool)

7. CASE의 분류

CASE는 일반적으로 통합여부에 따라 부분 CASE와 통합 CASE (I-CASE : Integrated CASE)로 분류되고 소프트웨어 생명주기(SDLC)단계에 따라 상위, 하위CASE로 분류된다.

1. 부분CASE(Component CASE)
   - -상위(upper)CASE : 기획수립, 분석과 설계단계 지원
   - -하위(lower)CASE : 프로그래밍과 테스트 지원
2. 통합CASE(I-CASE : Integrated CASE)

8. CASE 도입배경

CASE 도입의 필요성을 인식하기 시작한 것은 1990년 7월 증권감독원의 전산화 중 장기계획 수립시 부터이다. 당시 국내에서 통합 CASE를 사용하고 있는 사례가 전무하고 CASE 도입환경이 전혀 준비되어 있지 아니한 상황에서 왜 CASE를 도입하려고 하였는지 살펴보고

자 한다.

그 당시 증권감독원은 증권업계의 공동 온라인 업무를 운영하고 있는 증권전산(주)로부터 전산업무책임자를 스카웃한 바로 직후였으며 동 책임자가 제일 먼저 착수하여야 할 일은 전산업무 중장기 개발계획을 수립하는 일이었다. 다시 말하면 증권감독원 전산화에 대한 청사진을 제시하고 소정의 기간과 예산범위 내에서 감독원이 요구하는 전산업무를 차질없이 수행할 수 있는 프로젝트 추진계획을 작성하여 제시하는 일이었다.

중장기 계획의 주요 내용을 간단히 소개하면, 주요 구성항목은 업무 개발에 대한 방향, 개발업무, 인력운영, 추진조직, 전산기기도입, 통신, 전산설비, 교육, 소요비용 기타 주요 고려사항 등으로서, 시스템 구축의 목표를 첫째, 자본시장 정책결정을 위한 종합 정보체제의 구축 둘째, 효율적인 증권관리·감독시스템의 구축 셋째, 자본시장 국제화에 따른 외국인 투자관리를 위한 시스템 구축 등으로 하고 업무특성과 긴급도 및 예산 등을 감안하여 3단계로 구분하여 개발을 추진하되, 자체인력 등을 고려하여 외주용역개발을 주로 하고 일정부분은 감독원이 자체 개발함을 원칙으로 결정하였다.

개발단계별 업무현황은 다음과 같다.

구 분	대상업무	프로그램 본수	개발기간	비 고
1단계	증권회사관리 시장관리 국제업무 검사지원 인사·급여	996	'91. 1－'92. 5 (17개월)	(증권전산)
2단계	상장법인관리 등록법인관리 외감법인관리 감리지원업무	836	'91. 11－'93. 2 (16개월)	(포스데이타)
3단계	조사·통계 내부관리	597	'92. 4－'93. 4 (13개월)	

계획수립 당시의 상황은 다행스럽게도 1989년도에 전산전문회사에서 작성한 기본설계서가 있었으므로 업무량 파악이나 개발기간 및 소요공수 등의 파악은 비교적 수월하였으나, 문제는 업무량이 프로그램 본수상으로 약 2,700여본으로 방대하고, 조기전산화의 필요성에 따라 이를 적어도 3년 이내에 개발하여야 하며 동 기간에 투입할 직접 개발요원은 무려 2,700MM에 상당하고 개발 완료 후 유지보수를 위하여서는 적어도 90여명 이상의 상주인력이 필요한 대규모 프로젝트를 어떻게 추진하느냐 하는 것이었다.

이와 같은 대규모의 프로젝트를 단기간 내에 수행하기 위하여는 용역회사에 용역을 하면 소프트웨어 개발은 할 수 있으나, 문제는 개발 후 유지보수를 위해서 필요한 전산인원을 충원하여야 하는 데 있었다. 왜냐하면 동인원수는 감독원 전체인원이 450여명임을 고려하면 엄청난 비중을 차지하는 인원이었으며, 한편 그 많은 인원을 2-3년내에 채용한다는 것은 감독원 인사 정책상 추진하기 매우 어려운 일이었기 때문이다.

결국 업무개발은 외주용역에 의하여 개발토록 하고 용역관리는 최소한의 경력직원을 채용

하여 해결토록 하였으나 유지보수를 위한 전산인력 90명을 줄이는 방안에 대하여는 별다른 대안을 찾을 수 없어서 결국 CASE 도입을 통해서 해결하고자 생각하게 되었다. 왜냐하면 전통적인 방법으로는 도저히 이 문제를 해결할 수 없다고 판단하고 당시에는 아직 잘 알려지지 않았지만 생산성을 높이기 위한 CASE를 이용하기로 결심하였다.

당시 중장기 계획에 자동화 도구에 관하여 언급한 부분이 있었는데 그 부분을 잠시 회상함으로써 보다 구체적인 도입배경을 설명하고자 한다. 감독원 중장기계획을 보면 CASE의 개요와 도입의 필요성, 구현방향에서 참고로 개발공수의 추이와 국내 판매업자의 주요제품을 기술하고 있는데 이를 소개하면 다음과 같다.

개요에서 'CASE'란 Computer Aided Software Engineering의 약어로서 개발방법론과 도구(TOOL)의 결합체이며, S/W개발 전 공정의 자동화를 실현하는 기술이라고 하였고 부문에서는 다음과 같이 기술하였다.

첫째 대기하고 있는 전산화 요구에 대한 미개발분(Back-log)을 해결하고, 둘째 개발비용 절감 및 개발기간을 단축하며, 셋째 유지보수가 용이하고 효율적인 시스템을 개발하고, 넷째 표준화를 통한 개발자 상화간의 의사소통을 강화하며, 다섯째 표준 모듈 재사용 및 축적을 통한 생산성 향상에 기여할 수 있다고 기술하고 있다.

또 적용방침에 대하여는 동 CASE를 이용한 S/W개발이 국내에서는 아직까지 보편화되지 않은 점을 감안하여 우선 시범적으로 특정 업무를 개발한 후, 테스트를 거친 다음 전 업무에 확대 적용함으로써 동 개발방법에 따른 위험부담을 최소화하기로 하였다.

CASE 도입시 환경

(1) 정보시스템이 전혀 없는 상태
(2) 독자적인 개발방법론도 없음
(3) 정보시스템 개발경험이 없는 전산요원
(4) 컴퓨터 도입이 안 된 상태
(5) 정보시스템 개발 요구는 있음 ☜ 최고경영자
(6) 전산 전문가를 스카웃한 직후
(7) 전산화 중장기 계획 수립후
(8) 소프트웨어 개발방법론과 개발도구를 물색 중

9. CASE 추진전략

국내에서는 처음으로 대형 프로젝트에 새로운 개발방법론과 개발도구를 적용하여 정보시스템을 구축해 보기 때문에 성공에 대한 보장이 없다고 하겠다. 따라서 이 프로젝트의 총책임을 맡은 필자는 대단히 조심스럽게 진행할 수 밖에 없었다. 실패할 요인을 최대한 줄이고 프로젝트를 성공적으로 이끌기 위해서 세심한 주의를 기울여야 했던 것으로 기억된다. 이 프로젝트의 단계별 추진전략은 다음과 같다.

• 도입(1차 적용)단계

- CASE 제품선정 및 도입
- CASE 전담팀 구성
- 1차 교육 및 환경구축
- 1차 방법론 조정
- 시범 프로젝트수행 및 1단계 업무개발적용

• 발전(2차 적용)단계

- 2차 개발방법론 조정
- 1단계 개발업무 운영
- 2단계 업무개발 적용
- 2차 교육
- CASE 통합 데이타베이스 기반조성

• 확산(3차 적용)단계

- 2단계 개발업무 운용
- 3단계 업무개발 적용
- 3차 개발방법론 조정 및 보완

· 전 업무에 대한 CASE 적용의 본격화
· 연속적 교육
· 통합 데이타베이스 구축
· 시스템 정보통합
· CASE 버젼 향상에 따른 유지관리

● 정착단계

· 개발방법론을 전산업무 규정에 반영
· 개발방법론 최적상태 유지
· 연속적인 교육
· CASE 버젼 향상에 따른 유지관리

10. 환경구축

추진전략에 따라 프로젝트를 원만히 진행하고자 다음과 같은 하드웨어 환경에 CASE 도구를 설치하고 새로운 개발방법론과 개발도구의 교육을 실시하면서 개발환경을 구축하기 시작했다.

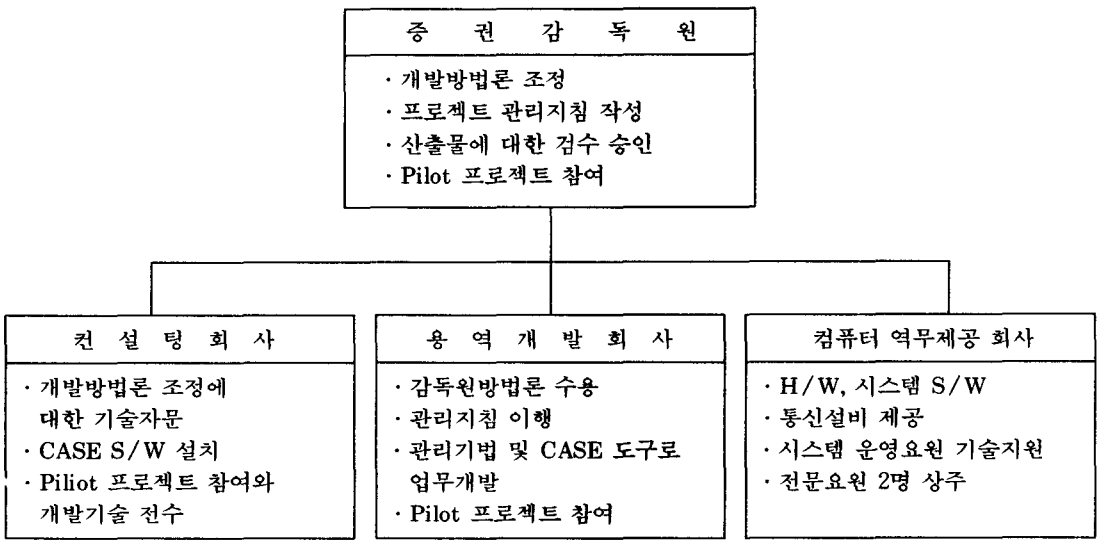
가. CASE 도구
· 관리기법/1 및 Design/1 설치
· DB2 및 CICS 환경검증
· Install/1 설치 및 한글지원 테스트

나. H/W
· W/S(386 PC) 및 LAN 구축 : 32대 설치
· File server, Gate way(386 PC) : 각 1대 설치
· 주전산기 : 타기관 전산기기 사용(삼성데이타시스템(주))
-기 종 : IBM 4381-T92

다. 전산교육 실시
· 추진방향
-대상자별 교육과정 설정
-실기위주의 OJT 교육
-자체전담 교육요원 양성을 통한 전달교육 강화
· 교육실시
-업무분석 기법교육 : 구조적 분석 및 설계
-Install/1 구현에 필요한 시스템 교육 : OS, CICS, DB2, JCL
-CASE 교육
┌ 관리기법/1 : 한국전산원
└ Design/1, Install/1 : 앤더슨컨설팅(유)

11. 프로젝트 추진조직

프로젝트를 추진하는 조직은 다음과 같이 구



성하여 상호 유기적인 관계를 유지하며 개발을 진행하도록 계획하였으며 각 팀마다 품질보증 조직을 두도록하여 생산성과 정확한 품질에 중점을 두어 추진하도록 하였다.

12. 개발평가

가. 관리적 측면
· 표준화된 개발방법론의 사용으로 프로젝트 관리자와 개발자간, 개발자 상호간의 의사소통을 원활하게 하여 팀의 결속을 강화

나. 생산성
· 개발과정에서 작성되는 문서를 기계(W/S)화하고 이를 자료보관소에 저장하여 공동으로 활용함으로써 문서의 상호참조를 용이하게 하여 문서의 중복작성을 배제할 수 있었음
· 모델 프로그램(Shell)의 사용 및 시스템 테스트 환경을 용이하게 지원함으로써 개발기간을 단축
· 프로그래머에게 시스템 환경에 대한 지식습득의 부담을 줄여줌으로서 프로그램 작성시 Logic(기능)에만 집중할 수 있도록 하여 품질향상에 기여

다. 운영적 측면
· 기 작성된 산출물(S/W)들의 재사용을 용이하게 함으로써 시스템변경사항이 발생시 유지·보수를 신속히 할 수 있도록 함

라. 개발기술적 측면
· 방법론에 입각하여 개발함으로써 개발자로 하여금 체계적이고 과학적인 개발 기술습득과 합리적인 개발태도를 함양시킴

13. CASE도구 적용시 성공적인 구현요건

CASE도구 적용시 성공적인 구현요건을 나열하면 다음과 같다.

- -경영층의 확고한 지원
- -관리자 및 개발자의 적극적인 참여
- -CASE에 대한 지나친 기대는 배제
- -충분한 사전교육 실시(Learning Curve 단축)
- -조직에 맞는 자체개발 방법론 설정
- -CASE에 대한 전담요원 육성
- -시범사업(PILOT PROJECT) 추진
- -시범사업(PILOT PROJECT) 경험자 현업에 배분 투입
- -하드웨어 기술발전 추이 감안
- -단계별 투자 및 효과분석
- -통합 CASE(Integrated CASE) 활용

14. 결 론

CASE 도구의 유용성에 대한 사용자의 인식이 제고됨으로써 앞으로 CASE 도구의 수요는 증가할 것이며 CASE tool은 소프트웨어 개발의 보편적인 도구로 정착될 것이 예상된다. 그러나 필자가 CASE 도구를 활용하여 업무를 개발하면서 당면했던 문제와 이를 어떻게 해결하였는지를 개발단계별로 구분하여 15개 항목을 기술하면서 본고를 마치고자 한다.

① 현업 요구사항 파악에 대한 중요성을 충분히 인식할 것
② 개발자는 관리기법/1에 따른 문서작성 요령에 대하여 예비지식을 충분히 가질 것
③ 관리기법/1의 사용목적과 효과에 대하여 사전에 충분히 인식할 것
④ 작업 착수시 개발자에게 개발지침을 제공할 것
⑤ 관리기법/1 및 CASE 도구에 대한 교육을 지속적으로 실시하고 교육상태를 반드시 확인할 것
⑥ 개발시스템 환경에 대하여 조기에 교육을 실시하도록 할 것
⑦ 사용자 요구사항 파악 및 시스템 설계시 수행되는 프로토타이핑은 현실적으로 실감나게 수행할 것
⑧ 관리기법/1 및 CASE를 이용해 S/W 구현중 개발자들의 강력한 저항과 반발이 있는

경우에도 이들을 설득시키고 소신과 확신을 가지고 지속적으로 추진할 것

⑨ 표준화를 잘 설정하여 이의 이행여부를 철저히 확인·점검하도록 할 것

⑩ 표준화는 조직적으로 추진할 것

⑪ 데이타는 CASE 도구의 자료저장소(Repository)에 통합 유지관리하도록 할 것

⑫ DBA, SP 등의 기술지원요원을 업무개발 추진과정에서 적기에 투입시킬 것

⑬ 관리기법/1 및 CASE 도구에 의한 소프트웨어 개발은 기존의 개발방법에서 새로운 개발방법으로의 전환, 즉 개발문화의 전이를 의미하고 있음을 개발자에게 주지시킬 것

⑭ 프로그램 모듈표준에 대한 검증을 강화할 것

⑮ 관리기법/1 및 CASE 도구에 의한 소프트웨어 개발에 있어서는 개발자의 중요한 관심사항이 프로그램보다는 설계, 설계보다는 사용자 요구사항임을 개발자에게 주지시킬 것

◆ CASE의 종류

제 품 명	개발회사	방 법 론	공급업체
IEW(Information Engineering Workbench)	Knowledge Ware	James Martin	언스트영 자문
Foundation(Method/1, Design/1, Install/1)	Arthur Anderson & co.	Method/1	앤더슨 컨설팅
EXCELERATOR	Index Technology	Yourdon/DeMarco	한국아이비엠
TEAMWORK	Cadre Technologies, Inc.	Yourdon/DeMarco	제니시스
POSE(Picture Oriented Software Engineering)	Computer Systems Advisers	Yourdon, Gane/Sarson, Constan tine, Finkistein	삼보컴퓨터
TELON	PANASOPHIC		지구어시스트
VAX set/DEC design	DEC	Yourdon, Mellor, Gane & Sarson	한국디지탈
PRIDE	M. Bryce and Associate. Inc.	Pride-ISEM (자체방법론)	벽산정보산업
CASE STATION/CODE LINK STATION	Mentor Graphics	Yourdon/DeMarco. Mellor	맨코그래픽스
CASE * dictionary CASE * designer CASE * generator	Oracle	CASE * Method (자체방법론)	오라클코리아
Softbench	HP	구조적 방법론 * 객체지향방법론	삼성 HP
IEF	TI(Texas Instrument)	James Martin	TI
YPS/APG	후지쯔(日)		한국후지쯔
LDA	유니시스		한국유니시스

참 고 문 헌

1. 양해술, CASE 개발기술과 고도이용, 정보산업 '93. 11～'94. 4. 한국정보산업연합회.
2. 권용래, CASE의 목표 및 발전방향, 정보과학회지 제9권 2호, '91. 4.
3. 이형원 · 우치수, CASE시스템의 구성요소, 정보과학회지 제9권 2호, '91. 4.
4. T. G. LEWIS, CASE, VAN NOSTRAND REINHOLD. 1991.
5. Chris Gain. CASE. Pretice-Hall International, 1990.
6. Carma McClure CASE is Software Automation. Pretice-Hall International, 1989.
7. 김려성저, 정보처리기술사 Course Work, 1993. 1. 20. 집문당.
8. 한국전산원, 관리기법/1 및 CASE 도구 적용사례, 1992. 6.

寄 稿 要 領

1. 一般要領

1) 投稿者의 資格은 本會 會員으로 한다. 다만 弘報委員會에서 特히 必要하다고 인정할 때에는 例外로 한다.
2) 本紙에 投稿되는 揭載內容으로서 技術解說, 技術資料, 時事性이 있는 論說(論壇, 提言, 建議) 現場工事報告, 紀行文, 社會相 또는 見聞記, 生活科學技術, 感想文, 研究論文, 研究報文, 其他 趣味, 體驗記, 分野別, 職場別, 懇談會 等等.
3) 本紙에 揭載키로 採擇된 原稿中 弘報委員會는 字句의 修正加減을 할 수 있다.

2. 投稿要領

1) 投稿는 200字 或은 400字 原稿紙를 반드시 使用하고, 題目과 姓名은 國漢文 및 英文으로 記載하여야 한다.
2) 採擇된 原稿에 對해서 所定의 稿料를 支拂한다.
3) 提出期間 : 投稿는 隨時로 한다.
4) 提出處 : 韓國技術士會 事務局(弘報委員會)

서울特別市 江南區 驛三洞 635－4
科學技術會館 401號 TEL : 566－5875, 557－1352