

경영정보학연구
제3권 1호
1993년 6월

효율적 S/W 개발을 위한 CASE도구 활용의 실증적 연구

전 응 섭 , 남 상 조

An Empirical Study on the Use of CASE Tools for Efficient Software Development

Computer Aided Software Engineering(CASE) tools are suggested as an automated method for the efficient development of softwares. This study are intended to measure the use level of CASE tools and to provide the meaningful information on the whole software development environment supported by CASE tools. Therefore, the results of a survey on CASE environment in Korea are empirically reported and analyzed. The use of CASE tools is measured; purposes purchasing CASE tools and achievement levels are investigated. Implications from the comparative analysis of the utilization of software engineering methodologies between the CASE user group and the non-user group are described. A proposition is suggested to check the request frequency for program change. Further studies on the reverse engineering, program reuse and the use of CASE tools under distributed processing systems such as client-server environments are suggested.

I. 서론

1960년대 부터 줄곧 제기되어온 소프트웨어 위기(Software Crisis)는 최근까지도 모든 MIS 부서마다 공통으로 안고 있는 중요한 문제가 되고 있다.

이는 하드웨어의 급가속적인 기술 진보에 따른 상대적인 소프트웨어에 대한 요구는 증폭되고 있고, 이를 충족시켜줄 만한 소프트웨어 제품이 생산되지 못하는 데에 그 원인을 찾을 수 있다. 뿐만 아니라 조직을 둘러싸고 있는 제반 환경이 급변하고 이에 따른 경쟁전략이 중요해짐에 따라 사용자 부서에서의 요구도 점차 다양해지고 있다. 이처럼 고도의 소프트웨어 개발의 필요성은 점점 증되는 반면 여기에 부응할 MIS조직의 인력과 예산, 그리고 개발 기술 수준등은 미약하기 때문에 소프트웨어의 생산성이나 품질 등의 향상에 어려운 요소로 작용하고 있는 것이 사실이다. 이러한 문제를 해결하기 위한 방안으로써 소프트웨어공학(Software Engineering)적 기법이 도입되었고, 이 개념을 이용하여 소프트웨어 개발의 자동화를 실현시킬 목적으로 자동화 도구인 CASE(Computer-Aided Software Engineering)도구를 도입 사용하는 것이 전반적인 추세이

자 활로라고 볼 수 있다.

1980년대 중반부터 등장한 CASE 도구는 선진국에서는 이미 적용업무 개발에 효율적으로 활용하기 위한 방안과 절차들에 관한 연구가 활발히 진행되고 있고, 도구의 도입 효과에 대한 실증적 효과를 발표하고 있다.

국내에서도 여러 종류의 CASE 도구들이 들어와 주로 소프트웨어 개발 회사와 연구소등을 중심으로 부분적인 업무개발에 적용되고 있고, 사용방법 및 관리의 정립과 지원기능의 효율적 이용방안에 관한 연구에 많은 관심을 가지고 있는 실정이다. 그러나 대다수의 일반 기업체에서는 아직도 CASE 도구의 사용방법 및 절차와 관리등에 대한 정확한 이해와 지식이 없거나, 막연한 기대감으로 도입사용을 추진함으로써 CASE 도구 도입을 통한 효율적인 소프트웨어 개발이라는 목적달성에 혼란을 야기하고 있다.

따라서 본 연구에서는 국내의 CASE 도구의 도입 및 사용 실태와 이에 대한 문제점을 분석하여 향후 효율적인 활용방안에 관한 토대를 제시하고자 한다.

본 연구는 제 II장에서 연구의 목적에 부합되는 이론적 배경을 설정하고 제 III장에서는 연구방법으로써 국내의 사용

실태에 관한 설문지조사 및 수집과 연구의 제약성을 논한다. 제 IV장에서는 연구 결과로써 통계처리된 데이터를 토대로 국내에서의 CASE도구의 활용현황을 제시하고 제 V장에서는 연구결과의 분석 및 토의로써 CASE 도구의 사용방법 및 절차, 소프트웨어 개발 수명주기, 사용자 만족도, 소프트웨어 개발 생산성등의 범주로 실증 분석을 한다. 그리고 마지막으로 제 VI장에서는 결론으로써 실증결과를 요약하고 향후 효율적 이용방안에 관한 제언을 한다.

II. 소프트웨어 개발 환경과 CASE 도구

소프트웨어 개발에 있어서 가장 어려운 점은 주어진 납기일 안에(Within Time), 주어진 예산(Within Budget)으로 최적품질의(High Quality) 제품을 생산하는 것이다. 그러나 대부분의 소프트웨어 개발 프로젝트에 임하는 MIS 부서는 투입인력의 부족과 개발비용의 초과, 시스템에 대한 사용자요구의 다양성과 복잡성, 최종 산출 소프트웨어 제품의 품질에 대한 요구조건, 그리고 납기일 준수와 비용절감등에 대한 도전을

계속해서 받고 있는 상황이라고 할 수 있다. 이같은 현상은 소프트웨어 개발을 수행하고 있는 거의 모든 조직체가 공통적으로 겪고 있는 현상으로 [소프트웨어 개발의 주요 장애요인]을 요약하면 다음과 같다.

- 예상비용 초과
- 개발 관리의 어려움
- 수정변경시의 어려움
- 개발 납기준수
- 제품의 신뢰성 결여
- 조악한 품질

이러한 문제를 해결하기 위한 방안으로 소프트웨어 공학이 탄생하였고, 소프트웨어개발 수명주기에 바탕을 둔 여러 개발방법론과 기법 그리고 관리절차등이 사용되기 시작 했다.

1. CASE도구의 도입배경

1970년대초에 소프트웨어 위기에 대한 대처방안의 하나로 구조적 기법이 도입되었고 이에 따른 정형화된 공학적 기법에 의한 접근방법으로 정보시스템 개발의 표준화와 절차수립 그리고 관리체계가 정립되었다. 이러한 개념은 소프트웨어의 생산성과 품질향상은 일련의

규칙과 정형화 그리고 표준화를 통해서만이 가능하다고 생각한 것이다.

1970년대 후반과 1980년대 초반에 와서는 이러한 구조적 분석과 설계기법을 소프트웨어 개발 수명주기의 요구분석 단계에 적용하여 프로그래밍 작업을 용이하게 했다. 이러한 방법들은 수작업으로 수행되기 때문에 일반적으로 사용하는 데 어려움이 많았으며 손으로 그린 자료흐름도와 같은 도큐먼트들을 수정관리하기가 어려워지기 시작하였다.

1980년대 중반부터는 소프트웨어 개발과정에 대한 새로운 시각으로써 소프트웨어 위기를 해결하려는 생각을 갖게되었다. 즉, 소프트웨어개발의 전공정을 단순화 시키고 자동화 시킴으로써 마치 제조업에서의 대량 생산체제에 의한 생산성향상과 같은 효과를 기할 수 있다는 생각이다. 이러한 개념은 지난 25년간의 정의된 소프트웨어공학의 기본적인 원칙들을 재 확인하는 것으로, 소프트웨어 개발의 생산성과 품질을 혁신적으로 증대시킬 수 있는 방안인 것이다. 여기에는 소프트웨어 분석과 설계 단계에서 부터 유지, 보수활동까지 전체 공정을 자동화하고, 검증과 확인도 자동적으로 수행하는 통합적기능이 포함되는 것이다.

CASE는 소프트웨어개발시 다음과 같은 효과를 제공한다.

- 구조화 기법의 실제적용
- Information Engineering기술의 부분적 지원
- 프로토타이핑의 실용화
- 유지보수의 용이성
- 개발과정의 단축
- 개발자원의 효율적 운영관리
- 부품의 재사용으로 인한 생산성과 품질의 향상

반면에 이러한 CASE 도구의 도입효과에 대해서는 실증적 분석을 통해 연구된 결과가 상반되게 나오고 있어 주목되고 있다. 먼저 CASE 도구가 생산성을 향상시켜 준다는 견해가 있고, 이에 반박하는 견해도 있다. 그러나 일반적으로는 CASE 도구의 사용으로 단기적인 생산성보다는 조직의 변화를 유도하고 질적인 향상을 통해 장기적으로 생산성이 증대되는 효과를 가져오고 있다는 견해가 지배적이다.

2. CASE 도구의 개념

소프트웨어 개발에 있어서 생기는 주요 문제점을 해결하는 방식중의 하나

가 과거의 수작업에 의한 개발방식에서 탈피하여, 개발과정을 자동화시켜고 높은 수준의 품질을 갖는 소프트웨어 제품을 만드는 것이다. 이러한 환경의 중심이 CASE에 의한 기술적 지원환경이다. CASE 도구는 소프트웨어개발을 자동화하는 것인데, 소프트웨어 개발 수명 주기의 전 단계를 연결하고 자동화함으로써 보다 잘 통합되게 하는 도구이다.³⁾ 따라서 CASE 도구는 소프트웨어 개발 수명주기의 자동화 기반을 강조하면서, 잘 통합된 기법과 방법론의 결합으로 이루어 진다. 즉, 기존의 소프트웨어 방법론인 구조화된 분석 및 설계와, 프로그래밍, 소프트웨어를 개발하기 위한 구체적이고 정형화된 절차를 포함하게 되며, 단지 구축에만 중점을 두는 것이 아니라, 소프트웨어 개발 수명주기 전체에 걸쳐 소프트웨어 개발 생산성에 중점을 두는 것이다.

이를 요약하면 CASE는 첫째, 소프트웨어 개발시 요구분석 및 설계단계의 수단을 제공하고 설계 명세서에의한 코드작성과 D/B Schema생성을 자동적으로 할 수 있도록 한다. 둘째는 프로그램의 코드로 부터 응용프로그램의 설계를 도

출할 수있다. 세째는 구조적 소프트웨어 공학의 개념과 기법들을 자동적으로 적용할 수 있도록 패키지화 되어있다.

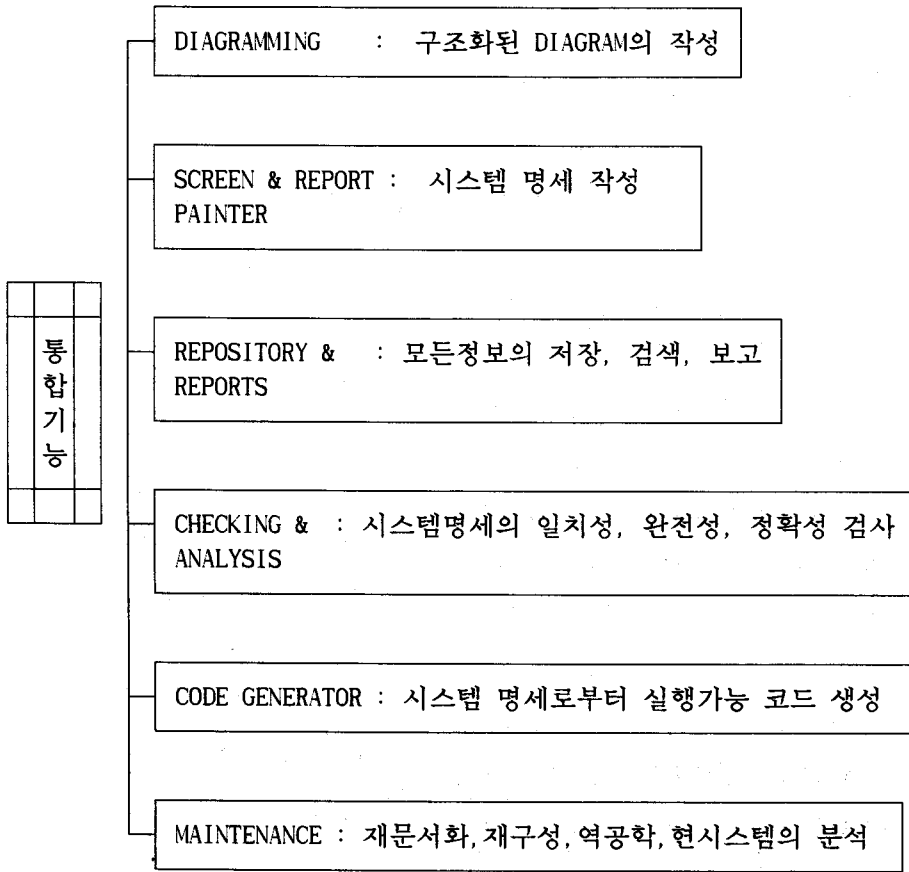
3. CASE도구의 지원기능 및 분류

3.1 지원기능

CASE는 시스템분석가로 하여금 초기의 사용자 요구에서부터 설계 구현 단계에 이르기까지 정보시스템을 문서화하고 모델링하는데 도움을주며, 또한 그들로 하여금 일치성, 완전성 그리고 표준에 맞는지의 여부에 대한 기능을 지원해준다. 이들 기능을 세분화하면 다음 장의 그림과 같다.

3.2 분류

CASE 도구는 조직의 정보시스템을 구축하기위해 필요로하는 계획, 분석, 설계, 구현등의 단계적 기능을 컴퓨터로 지원 활용할 수가 있다. 이러한 단계별로 적용가능한 도구를 크게 계획과 설계 지원 도구인 상위 CASE(UPPER CASE)와 프로그래밍 지원도구인 하위 CASE(LOWER CASE)로 구분하는데 최근에는 소프트



웨어 개발 수명주기의 전 단계를 통합적으로 지원하는 통합 CASE(INTEGRATED CASE)도구의 사용이 증대하고 있다.

본 논문에서는 CASE의 종류를 크게 3가지로 구분하기로 한다.

첫째, 소프트웨어 개발 수명주기상에서 소프트웨어 개발의 계획 및 요구분석 단계를 지원하는 기능인 상위 CASE

둘째, 설계 및 코딩의 구현단계를 지원하는 하위 CASE

셋째, 소프트웨어 개발의 최초 계획 단계에서부터 시스템의 구축 및 유지보수의 전 단계를 지원하는 기능인 통합CASE

이하에서는 위와 같은 이론적 배경하에 국내의 실증적 자료분석의 기준이 되는 변수로 설문지 항목을 구성하였다.

Ⅲ. 연구 방법

1. 연구 대상

본 연구에서는 CASE 도구의 사용실태를 파악하기 위해 국내에 도입되어 있는 CASE제품별 판매회사를 직접 방문하여 그들의 고객 명단을 입수하고 직접방문 또는 우편으로 설문서를 작성하게 하였다.

설문서 작성대상자는 크게 연구소, 국영기업체, 일반기업, S/W개발업체로 구분하여 CASE를 사용하여 개발하고 있거나 개발한 경험이 있는 그룹과 CASE를 사용하지 않고 S/W를 개발한 경험을 가진 사람을 선별하여 설문서에 응답하게 하였다.

2. 가설 설정

본 연구에서는 S/W개발시 CASE도구의 효율적 활용정도를 파악하기 위해서 CASE도구 사용자 집단과 비사용자 집단으로 구분하여 그 차이를 실증적으로 비교하는 방법을 취하였다. 이러한 실증적 분석을 위해 가설을 다음과 같이 설정하였다.

가설 1 : CASE 사용자군과 비사용자군간에 소프트웨어 개발기법의 활용에

차이가 있을 것이다.

가설 2 : 소프트웨어 개발단계에 있어 사용자군과 비사용자군간에 수행난이도의 인지에 차이가 있을 것이다.

가설 3 : 개발 소프트웨어의 최종사용자(End User)들의 만족도에 있어서 CASE 사용자군과 비사용자군간에 체감하는 정도에 차이가 있을 것이다.

가설 4 : CASE 도구의 기능별로 사용자 만족도에 차이가 있으며, 각 개발단계에서 활용에도 차이가 있을 것이다.

가설 5 : CASE도구 사용자와 비사용자간에 소프트웨어 개발생산성에 차이가 있을 것이다.

3. 설문지 구성과

가설 검증 절차

설문지의 구성은 크게 공통문항, CASE도구 사용자에 대한 문항, CASE도구 비사용자에 대한 문항의 세 부분으로 분류하였다.

공통문항은 설문 응답자의 조직에 관한 1 문항 및 소프트웨어공학 방법론에 대한 활용 및 인지도에 관해 2 문항으로 구성되어 있다.

CASE 도구사용자에 대한 문항 부분에서는 사용 CASE 도구에 관한 2 문항,

CASE 도구와 소프트웨어 공학 방법론의 접목에 관한 2 문항, CASE 도구의 선택 기준과 구입목적 및 만족도에 관한 5 문항, CASE 도구 판매자의 지원정도에 관한 1 문항, CASE 도구의 현 사용정도에 관한 4 문항등 총 14 문항으로 구성되어 있다.

CASE 도구의 비사용자에 대한 문항 부분에서는 소프트웨어 개발 현황에 관한 4문항, CASE도구 사용의지 및 기대활용성에 관한 4문항으로 이루어져 있다.

본 연구에서 설정한 가설을 검증하기 위한 절차는 다음과 같다.

첫째, CASE 도구 사용자의 선택이유, 만족도, 사용상황, 판매자의 지원정도등을 빈도수(Frequency)로 제시한다.

둘째, CASE 사용자군과 비사용자군간에 소프트웨어 개발기법의 활용에 차이가 있는가를 t-test를 통하여 검증하였다.

셋째, 소프트웨어 개발 단계에 있어 CASE사용자군과 비사용자군이 수행난이도의 인지에 있어 차이의 여부를 t-test를 통하여 검증하였다.

넷째, 소프트웨어 이용자들의 만족도에 있어 사용자군과 비사용자군이 체감하는 정도에 차이가 있는 지를 t-test를 통하여 검증하였다. 또한 CASE 도구의 기능별(UPPER, LOWER, INTEGRATED)로 각 개

발단계에서 활용에 차이가 있는지 또한 CASE 도구의 기능별로 사용자의 만족도에 차이가 있는 지를 DUNCAN의 multiple range test를 통하여 검증하였다.

다섯째, CASE도구의 사용자와 비사용자간에 소프트웨어 개발생산성의 차이가 있는지를 검증하기 위해 실제 투입인력(MAN-MONTH)과 개발규모(LINE OF CODE, FUNCTION POINT)를 산출하여 비교한다.

4. 자료수집

국내 도입된 각종 CASE 도구의 보급자를 통해 이미 설치된 120여업체의 사용자명단을 입수하여 총 300여부의 설문지를 주로 방문 배포하였다. 그 중 약 43%에 달하는 133부를 현장수거 또는 우편으로 회수하였다.

설문 조사 및 회수 기간은 1992년 8월 부터 1993년 6월 까지로 약 10개월 간에 걸쳐 수행되었다.

회수된 설문지중 CASE도구를 현재 사용하고 있는 개발자의 설문서 응답은 69부이고 현재 사용하지않는 개발자의 설문서 응답은 64부로 나타났다.

CASE 도구의 사용자인 경우에 평균 사용기간은 7개월이며 평균 교육기간은 2개월로 나타났다. 분석 대상자의 업종

에따른 분류는 <표1>과 같고 CASE 도구의 기능별 업종분류는 <표2>와 같다.

되어 있다고 볼 수 없으며, 따라서 미국 등 선진국에서 수행된 유사연구와 비교 시 신중한 해석이 요구된다.

5. 연구의 제약성

국내에 CASE 도구가 들어온지 얼마 되지 않았고, 도구를 사용한지가 평균적으로 10개월에 불과하여 CASE 도구의 활용이 제대로 수행되지 못하고 있는 상황이므로 답변의 신뢰도가 완전히 성숙

IV. 연구 결과

1. 소프트웨어 공학의 인지도

<표 3>에서 나타난 바와 같이 CASE 도

<표1> 분석 대상자 개요

업종	사용자	비사용자	총계
연구소	9 (13%)	9 (14%)	18 (13%)
S/W 개발업체	35 (51%)	17 (26%)	52 (40%)
국영기업	5 (7%)	5 (8%)	10 (7%)
일반기업	20 (29%)	33 (52%)	53 (40%)
계	69	64	133

<표2> CASE 기능별 사용자 분류

업종	UPPER CASE	LOWER CASE	INTEGRATED CASE
연구소	3(4%)	4(6%)	2(3%)
S/W 개발업체	4(6%)	10(15%)	22(34%)
국영기업	2(3%)	0(0%)	4(6%)
일반기업	3(4%)	3(4%)	12(17%)
계	12(17%)	17(25%)	40(58%)

구를 사용하고 있는 연구대상자들은 DFD (Data Flow Diagram)나 Data Dictionary, Mini Spec, Structure Chart, Pseudo Code, E-R Diagram과 같은 보편화된 소프트웨어 공학 기법을 실제 사용하고 있다.

CASE 도구 사용집단은 제반 기법의 사용비율이 62% 인데 반하여 비사용집단은 소프트웨어 공학기법의 인지 및 교육

에도 불구하고 제반 기법의 사용비율이 48%로서 CASE 도구를 사용하는 집단이 좀더 활발하게 업무에 적용하고 있음을 알 수 있다.

<표 4>에서 보는 바와 같이 S/W 개발 단계중 가장 수행하기 어려운 단계는 프로젝트 계획 및 관리와 시스템 요구분석단계로서 시스템계획을 효율적으로 지원할 수 있는 방법론의 보급이 중요함을

<표 3> S/W 개발시 사용하는 기법 (괄호안의 숫자는 응답자수)

<보기>
 현재 사용시효과 (1:우수, 2:적절, 3:비약)
 현재 미사용시 (4:추후사용예정, 5:시도후 유보, 6:잘모름)

기법	CASE 도구 사용집단 (N = 69)						CASE 도구 비사용집단 (N = 64)					
	사용시효과			미사용시			사용시효과			미사용시		
	1	2	3	4	5	6	1	2	3	4	5	6
DFD	14	34	11	2	0	1	7	26	10	14	3	3
DATA DICTIONARY	17	26	10	8	1	1	9	20	10	15	3	6
MINI SPEC	12	28	12	4	0	4	4	28	8	10	4	8
DECISION TREE/TABLE	3	8	18	12	4	6	1	5	7	15	4	22
JACKSON 방법	2	8	5	6	7	18	1	2	3	15	3	28
WARNIER-ORR방법	1	7	4	8	5	20	0	2	2	6	5	34
플로우 차트	1	14	19	3	9	8	18	10	4	8	10	3
STRUCTURE CHART	15	23	8	8	2	8	24	9	3	12	5	3
PSEUDO CODE	13	21	7	8	2	5	12	8	3	8	13	8
N-S CHART (나씨-쉬나이어만)	0	8	7	5	4	21	3	5	1	12	5	24
E-R DIAGRAM	25	17	6	9	0	9	11	8	5	12	1	11
HIPO	3	11	11	4	8	10	10	4	1	11	11	13
SADT	0	6	4	7	5	21	0	4	3	6	3	33

〈표 4〉 S/W 개발단계중 어려운 단계순위(1:가장어려움...7:가장쉬움)

개발단계	사용자 집단 (N = 69)							비사용자 집단 (N = 64)						
	1	2	3	4	5	6	7	1	2	3	4	5	6	7
프로젝트계획및관리	30	18	5	4	4	3	2	24	7	15	6	4	3	4
시스템요구분석	25	18	12	6	3	2	0	26	14	10	7	3	3	0
시스템 설계	1	13	26	7	13	3	3	7	19	12	12	9	4	0
코딩	0	0	0	8	8	18	32	1	0	0	6	8	16	33
테스트	1	6	7	9	13	23	7	0	5	7	11	11	23	7
유지보수	5	6	7	16	15	11	6	4	11	8	14	11	10	6
문서화	3	7	9	15	9	7	16	1	7	11	11	17	5	12

시사하고 있다.

2. CASE 도구의 선택기준 과 목적 및 만족도

CASE 도구의 선택기준은 〈표 5〉와 같이 주로 기능의 우월성이나 적합성이 가장 빈도가 높았으며 다음으로는 조직의 정책적 선택이 차지하는 반면에 보급자의 지원이나 도구의 사용경험, 가격측면은 선택시 상대적으로 덜 중요시됨을 알 수 있다.

특정 도구를 검토해서 도입하기까지 소요되는 기간이 평균적으로 6 개월이 소요됨에 따라 검토 및 도입에 있어서

〈표 5〉 CASE 도구의 선택기준

저렴한 가격	(9)
사전에 그 TOOL에 대한 지식이 많음	(8)
VENDOR의 지원이 우월	(12)
여타 TOOL보다 기능의 우월성, 적합성	(35)
조직의 정책상 선택	(31)

효율적이고도 전문적인 지원기능의 필요성이 대두되고 있는데, 특히 초기의 CASE도구의 적절한 도입이 개발 프로젝트의 성공과도 직결되므로 이에 대한 요구가 중요함을 알 수 있다.

CASE 도구가 사용자 인터페이스에 있

어 획기적인 서비스를 제공하는데도 불구하고 사용기간이 짧아 아직도 <표 6>에서 보는 바와 같이 CASE 도구의 사용이 상대적으로 어려움을 느끼고 있다.

이에 대한 원인으로서는 <표 7>과 같이 사용미숙이나 S/W 공학적 개념의 미비와 아울러 보급자의 지원부족으로 야기된 것임을 알 수 있다.

CASE 도구의 도입이유 및 실제개선된 분야는 <표 8>과 같이 개발의 체계화와 개발 S/W의 질적향상이 가장 중요한 도입이유였으며, CASE 사용후 실제로 개선이 많이 이루어진 것으로 나타났다.

그러나 개발시간 단축 및 비용절감, 업무적체해소, 인원절감등의 주요 도입이유들은 별 실효를 못 거둔 것으로 나타났다. 이는 CASE 도구 도입이 단기적 생산성을 목적으로 해서는 안된다는 사실과 부합한다고 볼 수 있다.

<표 6> CASE 도구의 사용상의 친숙도

매우어렵다(0)	어렵다(33)
쉽다(32)	아주 쉽다(0)

<표 7> CASE 도구사용이 어려운 이유

TOOL 사용경험 미숙	(28)
VENDOR의 지원 부족	(22)
S/W공학적 개념미비	(32)
적용대상 업무의 곤란성	(18)

CASE 도구의 도입 및 사용에 관한 경영층의 의지는 <표 9>와 같이 적극적 지원 및 장려가 압도적으로 많고 무관심의 경우는 극히 적어 CASE와 같은 고가의 개발환경 도입에 있어 경영층의 이해 및 지원이 매우 중요함을 보여준다.

<표 8> CASE 도구의 도입이유 및 실제개선분야

항목	도입이유	실제 개선
. 개발업무 적체 현상(BACKLOG) 해소	(22)	(5)
. 납기의 지연 해소	(11)	(3)
. 개발인원의 부족 해소	(22)	(6)
. 개발비용의 절감	(23)	(5)
. 개발시간 단축	(34)	(13)
. 개발 S/W 질적향상	(40)	(22)
. 개발내용변경시 즉시처리	(17)	(11)
. 개발의 체계화	(42)	(33)
. 신기술 경험축적	(33)	(27)

〈표 9〉 CASE 도구 도입에 대한 경영층의 의지

경영층의 적극적 지원 및 장려환경	(36)
부서장의 재량에 맡김	(13)
관심은 있으나 비용상 도입사용을 꺼려함	(7)
경영층의 전혀 무관심	(3)

CASE 도구를 사용하여 개발한 S/W를 최종사용자 부서에 인도한 후 사용자의 만족도는 아주 만족 또는 대체로 만족이 응답자의 64%에 이르고, 별차이 없음이 24%, 불만족이 12%에 이르러 최종사용자의 만족도는 대체적으로 만족한 수준임을 나타내고 있다. 그 이유로서 〈표 8〉에서 나타난 바와 같이 개발이 체계화되고 개발내용의 변경시 대처가 용이하며 개발 S/W의 질적인 향상등이 주요한 요인임을 알 수 있다.

3. 보급자(vendor)의 지원정도

보급자의 지원에 대한 불만은 〈표 7〉과 같이 CASE 도구의 사용이 어려운 이유중의 하나로 지적되는데, 이는 CASE 도구에만 국한된 것이 아니라 어떤 형태의 구매에도 발생하는 문제이다. 하지만 보급자의 지원이 어느 분야에 집중되는가를 분석하면 지원의 실태에 관해 대단히 중요한 정보를 제공해준다.

〈표 10〉에 의하면 보급자의 지원이 시스템 설치에 평균 11일이 지원되고 S/W 개발용역에 평균적으로 107일을 지원하는 반면 운영교육에는 평균 12일, CASE 개념교육에는 불과 9일의 지원이 이루어지고 있다. 여기서 운영교육 및 개념교육은 별도의 교육비를 부과하는 것으로 나타나고 있어 보급자의 역할이 주로 판매 및 설치 또는 개발용역에 치중되고 있다는 문제점을 시사하고 있다.

운영교육 및 개념교육에 교육비를 부과하는 것은 사용상의 애로 및 CASE 도구의 확산에 큰 장애 요인임에 틀림이 없지만 사실상 지적자산에 대한 당연한 대가라고 할 수 있다. 국내에서는 별도의 교육비에 대한 인식이 안되어 있어 충분한 교육투자가 부족하고 이것은 전문 운용교육의 개설을 저지하는 악순환으로 나타나는 것으로 생각된다.

〈표 10〉 보급자의 지원정도

시스템 설치 소요	(11)일
CASE 개념교육	(9)일
운영 교육	(12)일
S/W 개발업무 용역	(107)일

4. CASE 도구의 사용정도

<표 11>에서 볼 수 있듯이 수명주기 (life cycle) 지원분야중에서 가장 많이 쓰이는 분야는 요구분석 및 설계기능과 코딩분야라고 볼 수 있다. CASE 도구중에서도 요구분석이나 설계등의 소위 상위CASE와 코딩과 유지보수의 하위CASE까

지 통합기능을 가진 도구의 상품화가 활발할 것으로 예상된다.

프로젝트 계획 및 관리기능과 유지 보수기능은 현재 지원이 미약하나, 추후 지원되기를 희망하는 기능으로 나타나고 있다. 선진국에서는 관리, 유지보수, 테스트를 위한 전문 도구의 개발이 상당

<표 11> CASE 도구의 현 이용기능 및 희망사용기능

<LIFE CYCLE 지원 기능>	이용기능	사용희망기능
프로젝트 계획 및 관리	(20)	(33)
요구분석	(49)	(10)
설계	(52)	(7)
CODING	(44)	(11)
TEST	(28)	(18)
유지보수	(32)	(23)
<DIAGRAM 지원 기능>	이용기능	사용희망기능
DFD	(49)	(7)
DECISION TREE TABLE	(11)	(19)
STRUCTURE CHART	(41)	(7)
ACTION DIAGRAM	(35)	(9)
WARNIER-ORR	(6)	(14)
PSEUDO CODE	(28)	(9)
SCREEN LAYOUT	(49)	(5)
REPORT LAYOUT	(42)	(8)
DATA STRUCTURE	(48)	(8)
ENTITY-RELATIONSHIP	(53)	(6)
<기타기능>	이용기능	사용희망기능
DATA DICTIONARY	(46)	(8)
ERROR CHECKING	(36)	(11)
역공학(REVERSE ENGINEERING)	(13)	(29)
코드 생성자동화(PROGRAM SKELETON)	(40)	(15)
프로그램 재사용(REUSE)	(23)	(21)
형상관리(CONFIGURATION MANAGEMENT)	(10)	(23)
DOCUMENTATION 관리	(44)	(13)

히 진척되어 있으나 국내에는 아직 미진한 실정이다.

DIAGRAM 기능은 DFD, STRUCTURE CHART, SCREEN 및 REPORT LAYOUT, DATA STRUCTURE, E-R DIAGRAM등이 잘 활용되고 있음을 보여준다. 그러나 역공학이나 프로그램 재사용등의 S/W 자원화와 관련된 기능의 사용이 미약하고 사용하기를 희망하는 의지도 미약한 것은 아직까지도 국내 S/W 산업이 비효율적이고 비관리적임을 단적으로 나타내고 있다.

CASE 도구의 사용이 가장 어려운 S/W 개발단계는 프로젝트 계획 및 관리

단계로 나타났으며 요구분석 단계도 적용이 어려운 단계로 나타났다. 요구분석단계에서는 적용은 어렵지만 효율적으로 활용되고 있는 반면 프로젝트 계획 관리단계는 가장 활용이 안되고 있는 단계인 것으로 나타나 이분야에 대한 연구가 집중적으로 이루어져야 할 것으로 보인다.

코딩단계와 문서화단계는 적용도 쉽고 활용도 효과적으로 나타나 인적자원의 질감에 큰 역할을 기대할 수 있다고 볼 수 있다.

<표 12> CASE 도구의 적용난이도 및 활용효과

개발단계	적용 난이도 (1: 가장 쉬움, 7: 가장 어려움)							활용 효과 (1: 가장 효과적, 7: 가장 비효과적)						
	1	2	3	4	5	6	7	1	2	3	4	5	6	7
프로젝트 계획 관리	1	2	4	7	12	9	24	9	6	9	1	8	6	19
요구분석	6	2	5	2	7	20	16	21	14	8	5	1	7	5
시스템설계	2	8	7	7	19	9	7	5	17	17	7	11	1	2
코딩	23	10	5	9	5	1	3	18	4	7	14	4	4	3
테스트	4	16	11	6	7	6	7	1	7	4	6	17	8	11
유지보수	1	8	17	10	7	11	3	3	6	8	6	9	17	7
문서화	18	9	8	17	2	3	1	4	6	6	20	5	9	7

〈표 13〉 S/W 개발기법의 활용도

기법	mean(비사용자)	mean(사용자)	t-value
DFD	2.06977(N=43)	1.94915(N=59)	0.93733
DATA DICTIONARY	2.02564(N=39)	1.86792(N=53)	1.05699
MINI SPEC	2.10000(N=40)	2.00000(N=52)	0.77884
DECISION TREE/TABLE	2.46154(N=13)	2.51724(N=29)	-0.24952
JACKSON 방법	2.33333(N=6)	2.20000(N=15)	0.35435
WARNIER-ORR방법	2.50000(N=4)	2.25000(N=12)	0.73552
플로우 차트	2.05556(N=36)	2.52941(N=34)	-3.08915*
STRUCTURE CHART	2.02439(N=41)	1.84783(N=46)	1.21993
PSEUDO CODE	2.16667(N=24)	1.85366(N=41)	1.74457
N-S CHART (나찌-쉬나이더만)	2.44444(N=9)	2.46667(N=15)	-0.08039
E-R DIAGRAM	1.96429(N=28)	1.60417(N=48)	1.98700*
HIPO	2.05882(N=17)	2.32000(N=25)	-1.23690
SADT	2.42857(N=7)	2.40000(N=10)	0.10999

(* : 유의수준 $p < 0.05$)

V. 연구 결과의 분석 및 토의

1. S/W 개발기법의 활용

〈표 13〉에서 보면 소프트웨어 개발 기법들중 CASE 도구 사용집단은 E-R Diagram의 활용에 있어서, 그리고 비사용집단은 Flow Chart의 활용면에서 통계적으로 유의한 활용우위를 보이고 있다. 이는 CASE 도구사용집단이 Data model-

ing을 중심으로 개발에 임하고 있는 반면, 비사용집단은 아직도 Process 위주의 개발에 치우치고 있음을 암시하고 있다.

기타의 기법들에서는 두 집단간에 통계적으로 유의한 차이를 보여주지 못하고 있어 CASE 도구 사용자들이 아직 CASE 도구에서 제공되는 여러 기법들을 적극 활용하지 못한다는 의미로도 해석된다.

〈표 14〉 S/W 개발 단계중 수행이 어려운 단계

단계	mean(비사용자) (N = 63)	mean(사용자) (N = 66)	t-value
프로젝트계획및관리	2.70603	2.25758	1.57352
시스템요구분석	2.30159	2.24242	0.23918
시스템설계	3.02286	3.69091	-1.98458*
코딩	6.12500	6.12121	0.01927
테스트	4.95312	4.87879	0.28087
유지보수	4.10937	4.31818	-0.69534
문서화	4.54687	4.59091	-0.14297

(* : 유의수준 $p < 0.05$)

2. S/W개발 단계별 수행 난이도

〈표 14〉에서는 소프트웨어 개발단계 중 CASE 도구 사용자와 비사용자간에 수행시 느끼는 난이도에 대한 차이가 있는가에 관한 결과를 보여주는 데, CASE 도구 사용자가 비사용자에 비해 시스템 설계단계를 쉽게 느끼고 있다는 것이 통계적으로 검증되었다.

하고 또 개발시 pioneer의 시행착오들이유로 생각할 수 있다. 이에 대한 연구는 CASE 도구의 활용이 전반적으로 확산되고 숙달되었을 때 이루어져야 하리라 본다.

〈표 15〉 최종사용자 만족도

mean(사용자) (N = 58)	mean(비사용자) (N = 42)	t-value
2.36207	2.38095	-0.13074

3. CASE기능별 사용자 만족도

〈표 15〉에서 보면 개발 소프트웨어를 최종사용자 부서에 인도한 후 사용자의 만족도에 있어 CASE 도구 사용자와 비사용자간에 차이가 통계적으로 유의하지 않다.

이것은 전반적으로 CASE 도구의 활용기간이 짧아 유지보수까지 이루어지지 못

CASE의 기능별 만족도

class	mean	F-value	Pr>F
INTEGRATED CASE	2.455	0.25	0.7823
UPPER CASE	2.364		
LOWER CASE	2.222		

상위CASE, 하위CASE, 통합 CASE 별로 사용자의 만족도는 ANOVA에 의한 결과로 상호간에 차이가 없음을 나타내고 있다.

4. S/W 개발 생산성

본 연구에서는 생산성을 측정하기 위한 척도로써 투입노력 대비 생산된 제품의 규모(코드라인수나 function point)를 비율로 계산하려 했다. 따라서 본 설문서에서는 이러한 항목을 측정하기 위해 문항을 다음과 같이 설정했다.

- 1) 투입노력 측정 변수 : 각 단계별 월 투입 인원(MAN-MONTH)
- 2) 생산제품의 측정변수 : 코드 라인수 또는 기능수(FUNCTION POINT)

그러나 실제 설문지를 회수하여 본 결과 2)에 관련된 정확한 자료를 수집하기가 어려워서 처음 설정한 방법의 생산성 측정보다는 사용자의 수정변경비율과 각 단계별 노력배분 현황으로 이를 대체하였다.

<표 16> 수정 변경 비율

mean(사용자) (N = 39)	mean(비사용자) (N = 14)	t-value
28.01	43.3081	2.13*

(* : 유의수준 $p < 0.05$)

먼저 현업에서의 사용자들의 변경요구에 대한 반응 척도로써 수정변경 노력 / 총개발 노력의 비율로써 척도를 측

정했는데 이에 대한 CASE사용자 집단과 비사용자 집단의 평균값은 위의 <표 16>과 같다.

위의 데이터는 최종사용자의 잦은 변경 요구에 대한 투입노력이 CASE의 비사용자보다는 사용자에게서 투입노력이 적게 들어 감을 보여주고 있다.

소프트웨어개발시 투입되는 개발수명주기상에서의 단계별 노력 배분현황을 보면 다음 <표 17>과 같다. 이 데이터는 표본수가 적어서 평균차이에 관한 t-value를 검증하지 못하고 단지 단계별 배분의 우선순위만을 보기로 한다.

비사용자 그룹에서는 프로그램코딩 부분에 관한 투입노력과 유지보수에 대한 투입노력이 다른 단계보다도 높게 나타남으로써 생산성 향상에 관한 전형적인 문제점을가지고 있다고 보아야 할 것이다.

사용자 그룹에서는 시스템분석과 설계부분에 많은 노력을 배분함으로써 S/W 제품의 품질과 신뢰성 향상에 진력하고 있음을 보여준다.

여기서 코딩 부분에 아직도 17.4%의 노력이 배분되고 있는 것은 CASE를 분석과 설계를 위한 문서화 작성등에 부분적으로 이용하고 있거나, 본격적인 교육과 훈련의 미비로 실제로 잘 활용하지 못하

고 있는 것으로 생각되어진다.

〈표 17〉 CASE 사용자와 비사용자간의
투입노력 배분비율

단계	투입 노력 배분 비율(%)	
	비사용자	사용자
프로젝트계획및관리	7.4%	9.4%
시스템요구분석	8.2%	21.3%
시스템설계	13 %	19.1%
코딩	23.6%	17.4%
테스트	10.7%	13.2%
유지보수	26 %	12 %
문서화	10.5%	7.5%

Ⅵ. 결 론

본 연구에서는 S/W개발시 CASE도구 활용의 정도를 알아보기 위해 사용자 집단과 비사용자 집단으로 구분하여 가설을 설정하였고 이에 대해 다음과 같은 결과가 유도되었다.

첫째, 가설 1의 검증에서 나타난 결과를 보면 CASE 도구의 사용집단이 비사용집단에 비해 월등히 소프트웨어 공학 기법을 많이 이용하고 있음을 알 수 있다.

둘째, 가설 2의 검증에서 나타난 결과를 보면, 도구 사용자 집단이 비사용자 집단에 비해 시스템 설계단계에서 수행한 이도가 낮고 다른 개발단계에서는 유의

하게 차이를 나타내지 않았다.

셋째, 가설 3의 검증에서 나타난 결과를 보면, 최종산출물에 대한 사용자의 만족도에서 도구 사용자집단과 비사용자 집단에서 유의한 차이를 나타내지 않고 있다.

넷째, 가설 4의 검증 결과에서는 CASE의 각 기능별 활용에 따른 최종사용자 만족도에 유의한 차이가 나타나지 않았다.

다섯째, 가설 5에 대한 검증 결과에서는 도구 사용자와 비사용자간의 S/W 개발 생산성의 측정 척도로 투입노력과 생산코드 라인수에 관한 데이터대신에 수정변경 비율로 대체하여 측정하였는데 도구 사용자와 비사용자간의 유의한 차이가 있었다.

최종사용자의 잦은 변경요구에 대한 투입노력이 도구 사용자에게서 보다 효율적으로 수행되고 있음을 보여주었다.

본 논문에서 조사된 국내의 CASE 도구사용에 대한 전반적인 환경을 요약하면 CASE도구의 사용시 요구되는 방법론이 사용자를 보다 이론적으로 체계화하는 기회를 제공하고 있음을 알 수 있다. 그러나 보다 효율적인 활용을 한다는 측면에서는 통계적 유의성이 검증되지 않아 좀더 경험의 축적이 요구됨을 알 수 있다.

E-R Diagram과 Flow Chart의 활용항목의 비교를 통해서 볼때, CASE 도구가 Process 위주의 개발보다 Data modelling 위주의 개발을 유도함을 알 수 있다.

CASE 도구의 도입이유는 개발의 체계화나 S/W 개발의 질적향상, 개발시간단축이 중요한 이유로 부각되며 개발의 체계화나 질적향상 측면에서는 실효를 거두고 있는 것으로 보인다.

개발시간 단축은 큰 효과가 없는 것으로 조사결과가 나타났는데 이는 CASE 도구 도입의 타당성을 규명하는 데 있어 단기적 생산성 보다는 질적인 향상과 장기적 안목에 초점을 맞추어야 할 것임을 시사한다.

CASE 도구의 도입에는 경영자의 이해 및 장려가 필수적인 것으로 나타난바 경영자에 대한 지속적인 Information Technology 개념 주입 및 신기술 소개 프로그램이 활성화되어야 할 것이다.

CASE 도구의 사용을 통한 제품의 인도시 최종사용자들의 만족도는 비사용 개발자들이 체감하는 최종사용자의 만족도와 통계적 차이가 검증되지 못하는데 이는 아직까지 CASE 도구의 사용경험이 짧은 것에 기인한다고 볼 수 있다.

CASE 도구의 이용자가 사용이 어렵다고 답한 경우가 쉽다고 답한 경우보다

많은데 이것 역시 사용경험이 증가할 수록 쉽게 느껴질 것으로 판단되고 소프트웨어 공학의 개념도 점차 확산될 것으로 보인다.

보급자의 지원현황은 아직도 설치 및 개발용역에 크게 치우치고 있으며 운영 교육 또는 CASE 개념교육은 미진한 것으로 나타나고 있다. 교육의 경우 별도의 교육비를 부담하여야 하나 지적자산에 대한 인식이 부족하고 교육을 담당할 수 있는 능력이 국내 보급자에게도 결여되어 있다는데 그 원인이 있다. 따라서 선진국 수준의 교육이 이루어 지려면 교육비 부담에 대한 인식 제고와 함께 교육여건의 확립이 시급하다고 할 것이다.

현재 CASE 도구 사용실태는 기본적인 운용정도에 그치고 있는 것으로 판단되고 있으며 소프트웨어의 자원화 개념이 성숙되어 있지 않아서 선진국에서 활발히 논의되고 있는 역공학이나 프로그램 재사용을 통한 생산성 및 품질의 향상이 시급한 과제로 이에 대한 활용방안이 요청된다.

국내에서는 아직 CASE사용에 의한 개발환경이 성숙되지 않은 상황에서 최근의 정보화 기술이 다운사이징과 분산처리 및 CLIENT-SERVER환경으로 변화하게 됨에 따라 대형 HOST 중심의 CASE

기술을 어떻게 적용해야 할 것인가도 중요한 문제로 부각되고 있다. 따라서 향후 연구에서는 분산처리 및 다운사이징과 같은 환경하에서 CASE기술이 어떻게

활용되고 있으며, 기존 HOST중심의 CASE 사용시와의 비교연구도 병행되어야 할 것이다.

참 고 문 헌

- Minder Chen, Jay F. Nunamaker Jr., E.S. Weber, "Computer-Aided Software engineering: Present status and Future Directions," DATA BASE, Spring, 1989, pp 7-13.
- Gordon C. Everest, Macedonio Alanis, "Assessing User Experience with CASE Tools: An Exploratory Analysis," Proceedings of 1992 HICSS, Vol II, pp 343-352.
- Carma McClure, CASE is S/W Automation, Prentice Hall International Editions 1989.
- Ncco C.R., N.W. Tsai & K. W. Holgeson, "Current Usage of CASE Software," Journal of Systems Management, 1989 May, pp 6-11.
- Statland N., "Payoffs Down Pike : CASE," Datamation(35:7),1989, April 1, p 32.
- Norman Ronald J. and Jay f.Nunamaker , "An Emirical Study of Information Systems Professionals' Productive Perceptions of CASE Technology," Proceedings of the Ninth ICIS Conference, Mineapolis, MN,1988
- Loh M. and R. R. Nelson , "Reaping CASE Harvest," Datamation(35:13) 1989, July 1, pp 31-34.
- Glass M. C., J.G. Hughes, W. Johnston and I. McCheseny, "Critical Analysis of Tools for Computer-Aided Software Engineering," Information and Software Technology(31:9) 1989, Nobemver, pp 486-496.
- Gibson M. L., "The CASE Philosophy," Byte(14:4),1989, April, pp 209-218.
- McClure, Carma, "The CASE Experience," Byte(4:14),1989 April, pp 235-244.

- Everest Gordon C., Presentation for Index Group, Cambridge, MA, 1990 February and May.
- M. Chen, R. J. Norman, "A Framework for Integrated CASE," IEEE SOFTWARE, 1992.3., pp 18-22.
- M. Jarke, "Strategies for Integrating CASE Environment," IEEE SOFTWARE, 1992.3., pp 54-61.
- I. Thomas, B. Nejme, "Definitions of Tool Integration for Environment," IEEE SOFTWARE, 1992.3., pp 29-35.
- R. M. Poston, M. P. Sexton, "Evaluating and Selecting Testing Tools," IEEE SOFTWARE, 1992.5., pp 33-42.
- P. Oman, "Maintenance Tools," IEEE SOFTWARE, 1990.5., pp 59-64.
- S. Reisman, "Management and Integrated Tools," IEEE SOFTWARE, 1990.5., pp 71-76.
- M. E. Porter, V. E. Miller, "How Information Gives You Competitive Advantage," Harvard Business Review, July-August 1985.
- P. Samuelson, "Reverse-Engineering Someone Else's Software: Is it Legal?," IEEE SOFTWARE, 1990.1, pp 90-96.
- P. Freeman, "A Perspective on Reusability, in Tutorial: Software Reusability," IEEE Computer Society Press, 1987.11.
- Leland L. Beck & Thomas E. Perkins, "A Survey of Software Engineering Practice : Tools, Methods, and Results," IEEE Transaction On Software Engineering, VOL. SE-9, NO. 5, 1983.9 pp 541 - 561.

◇ 저자소개 ◇



공동저자 전웅섭은 성균관
대학교 대학원 경영학석사,
KAIST 경영정보공학과 박사
과정, 한국과학기술연구원
시스템공학센터연구원을 거

쳐 현재, 인덕전문대학 사무자동화과 교
수로 재직하고 있다. 관심분야는 전문가
시스템, 소프트웨어공학, 지식베이스를
이용한 Object-Oriented 분석 및 설계
등이다.



공동저자 남상조는 State
University of New York at
Buffalow 경영학석사, KAIST
경영정보공학과 박사과정을
거쳐 현재, 신구전문대학

경영과 교수로 재직하고 있다. 관심분야
는 전문가시스템, 신경망이론, 주식투자
자문 시스템 등이다.