CASE 도구활용과 소프트웨어 개발 프로세스 수준이 정보시스템 개발 생산성에 미치는 영향

박중영,김영걸

한국과학기술원 경영정보공학과

요약문

CASE 도구의 활용과 소프트웨어 개발 프로세스 수준과의 관계를 규명하기 위하여 국내 57개 기업의 91개의 정보시스템 프로젝트를 대상으로 한 실증적 연구를 수행하였다. 기업의 일반적인 특징과 CASE 도구의 활용 정도, 소프트웨어 개발 프로세스 수준에 관한 자료를 획득하기 위하여 설문조사를 실시하였고, 연구 초점은 CASE 도구의 활용에 있어서 중요한 요인을 찾는데 있다. CASE 도구의 도입 형태와 소프트웨어 개발 프로세스가, 개발 생산성에 미치는 영향에 대하여 4가지 가설을 검정하였다. 검정결과 두 독립변수가 모두 개발 생산성에 유의한 영향을 미치는 것으로 나타났다. 또한 CASE 활용과 소프트웨어 개발 프로세스간의 교호작용이 확인되었다. 본 연구는 또한 CASE 도구의 활용과 소프트웨어 개발 프로세스 수준사이의 전략적 관계 격자를 제시하였다. 전략적 관계 격자는 CASE 도구 도입 형태와 소프트웨어 개발 프로세스 수준의 정도에 따른 4가지 셀로구성되어 있는데 개별 셀마다 각 기업의 정보시스템 환경을 고려한 효과적인 전략을 제시한다. 마지막으로 CASE 도구의 활용을 위한 추후 연구방향이 제시된다.

제1장 서론

오늘날, 기업은 경제 및 경쟁의 범 세계화라는 큰 압력에 직면하고 있다. 이러한 경쟁압력을 극복하기 위해 기업은 비지니스 프로세스 조정, 조직의 많은 계층을 축소하거나, 또는 기업의 전략적 목적을 달성하기 위하여 정보기술을 도입하고 있다 [Drucker, 1988; Keen, 1991]. 도입된 정보기술은 조직의 경쟁 효과와 경쟁 우위를 결정하는 중요한 요인으로 제시되고 있다 [Singh, 1993]. 도입된 정보 기술은 기업과 정보시스템 조직에 큰 영향을 미치고 있는데, 기업은 중요한 비지니스 프로세스에 정보 기술을 신속하게 투입하여 생산과 서비스로 통해서 기업목적을 달성하고자 한다. 위와 같이 확장된 정보시스템의 역할을 지원하기 위하여 정보시스템 조직은 보다 신속하게, 보다 높은 품질의 시스템을 개발하고자 하는 추세에 있다 [Chen and Norman, 1992].

그러나 기업이 새로운 정보시스템을 구축하는데 소프트웨어 개발의 지연과 사용자 요구사항의 불만족, 소프트웨어 개발의 스케쥴 및 비용 추정이 부정확하며 소프트웨어 개발 생산성이 사용자에 대한 만족을 충족시키지 못하고 있고, 소프트웨어 개발에 있어 품질 문제 등이 장애요인이 되어 왔다 [Press, 1987]. 또한 소프트웨어 크기 증가, 소프트웨어 기능의 다양화, 통합 데이타베이스와 네트웍킹, 분산처리시스템, 사용자와의 인터페이스 관계 등과 같은 소프트웨어 환경 변화로 정보시스템 환경 변화에 맞는 소프트웨어 개발이 점점 더 어려워 지고 있다 [박 성주, 1991]. 이에 대한 해결책으로 많은 기업은 소프트웨어 개발에 대한 생산성 및 품질 문제를 해결하기 위해 소프트웨어 엔지니어링 기법을 자동화한 CASE 도구를 도입하고 있다. CASE 도구는 시스템 분석 및 구조적 방법론 같은 기능을 통해 정보시스템 개발 계획에서 유지보수의 모든 단계를 지원함으로써 기업의 소프트웨어 개발 과정을 자동화하여 신속한 소프트웨어 개발을 가능하게 한다.

기업은 CASE 도구를 도입할 때 막대한 비용을 투자하면서도 그에 상응하는 효과를 직접적으로 보지 못하는 실정이다[Huff, 1992]. 그 원인으로는 여러 가지가 있지만 그 중에서도 CASE 도구의 사용 방법 및 절차와 관리 등에 대한 정확한 이해와 지식의 결여와 막연한 기대감에 의한 CASE 도구의 도입 추진 등을 들 수 있다 [전응섭, 남상조, 1993]. 그러나 이러한 CASE 도구의 도입에 대한 부정적인 효과에도 불구하고 CASE 도구의 도입은 소프트웨어 개발에 관련된 프로세스 절차 및 표준화를 정립시키고, 사용자의 정보시스템 변경요구에 대한 신속한 대응을 가능하게 한다. CASE 도구를 도입하고자 하는 기업은 CASE 도구의 공급자 명성, 사용자 교육 지원 여부, 제공되는 CASE 도구의 기능, 기존 어플리케이션과의 호환성 여부 등을 고려하고 있다. 이런 요인들은

실제 CASE 도구의 도입에 있어서 중요하지만 여러 환경 요인과의 연관성을 고려해 볼 때 도입 목적을 달성하는데 있어서 충분하지는 않다. 이는 많은 기업들이 CASE 도구의 도입을 추진함에 있어 이런 요인들을 고려하긴 하지만 소수 기업을 제외한 대다수의 기업은 CASE 도입 효과를 보지 못하는 사례를 통하여 알수 있다. 따라서 본 연구는 기업의 CASE 도구의 도입에 있어 최적 도입 및 활용 전략을 수립하기 위하여 기업의 소프트웨어 개발 프로세스 수준과 CASE 도구의 도입형태가 정보시스템 부서의 소프트웨어 개발 생산성에 미치는 영향에 대하여 연구하고자 한다.

제2장 문헌연구

2.1 CASE 도구의 도입배경

CASE (Compute-Aided Software Engineering) 도구란 원래의 의 미는 컴퓨터를 통하여 소프트웨어를 개발하기 위하여 사용하는 방법론을 제공하는 것을 말한다 [Yourdon, 1986]. 또는 "정보시스템 개발의 모든 단계 즉, 계획에서 유지 보수단계의 모든 단계를 지원하기 위한 도구"를 말한다 [Everst and Alanis, 1991]. 기업의 정보시스템 조직은 소프트웨어 개발 환경 변화로 인해 소프트웨어 개발 문제가 발생하고 이러한 문제를 해결하기 위하여 구조적 방법론 및 정보 공학 개념을 도입하기 시작하였고 정보공학 개념은 정보시스템 개발의 표준화, 절차 확립, 관리 체계의 정립등으로 나타나고 있다. 소프트웨어 개발 생산성 및 품질 향상, 문서화등을 제공하는 정형화된 소프트웨어 개발 방법론과 CASE 도구는 기업의 비지니스에 새로운 기회를 제공한다 [Winant, 1992].

이러한 CASE 도구는 도입시 단기적인 생산성 향상보다는 조직의 변화를 유도하고 기업의 비지니스와 소프트웨어 개발 프로세스를 변화시켜 장기적인 생산성, 품질 향상을 가져다 줌과 동시에 소프트웨어 개발 비용도 감소시키고 [McClure, 1989; Kemerer, 1992], 또한 CASE 도구의 도입 효과에 대한 학습 곡선(Learning Curve) 은 도입후 최소한 6개월에서 1-2년 경과 후에 생산성 효과가 나타나는 것으로 연구되었다 [Kemerer, 1992]. 이런 CASE 도구의 특성으로 인하여 기업은 소프트웨어 개발에 대한 생산성 향상, 품질관리, 효과적인 정보 자산 관리를 하기 위하여 CASE 도구의 도입을 추진하고 있다.

2.2 CASE 도구 분류 및 특징

1) CASE 도구분류[McClure,1989; Chen & Norman,1992; Everest & Alanis, 1992]

소프트웨어 개발의 지원 단계에 따라 CASE 도구를 분류하면 다음과 같이 3가지로 구분할 수 있다. 첫째, 소프트웨어 개발의 최초 계획 단계에서 분석, 설계, 코딩, 시스템 구축 및 유지보수의 모든단계를 지원하는 통합 CASE(ICASE) 도구. 둘째, 소프트웨어 개발에서 계획 및 요구 분석, 디자인 단계를 지원하는 상위 CASE (Upper CASE) 도구. 세째, 소프트웨어 개발에서 코딩 및 유지보수의 단계까지 지원하는 하위 CASE (Lower CASE) 도구로 분류할 수 있다.

2) CASE 도구 기술[Chen & Norman, 1992]

CASE 도구의 기술은 관리, 도구, 방법론의 3요소로 구성되어 있다. 이 3 가지의 구성 요소를 간단히 설명하면, 첫째, 관리는 소프트웨어 개발에서 사용하는 도구나 방법론을 선택하는데 있어서 적절하게 선택되었는지 또는 정보시스템 조직의 개발 및 유지보수 업무를 지원하는데 사용되는지를 판단하거나, 정보 시스템 계획 및 정책의 결정, 정보시스템에 대한 투자 및 표준을 결정하는 것을 말한다. 둘째, 방법론은 소프트웨어 개발에서 구조적 분석 및 설계, JAD (Joint Application Design) 등의 개발 방법에 관한 영역을 말한다. 셋째, 도구는 소프트웨어 개발 방법론을 지원하는 것으로 정보 저장소 도구, 역 공학 도구, 재 공학 도구, 정보시스템 계획 및 분석 도구 등으로 구성되어 있다.

3) CASE 도구 특징

(1) CASE도구장점및 성공요인[Everest & Alanis,1992 ; Chen & Norman 1992]

기업은 소프트웨어 개발시의 생산성 및 품질 문제를 해결하기 위하여 자동화된 도구를 필요로 하게 되었다. 자동화된 도구를 소프트웨어 개발을 위해 도입함으로써 얻을 수 있는 장점은 정보공학의 구조화 기법을 업무상에서 실제 적용할수 있고, 정보 공학 기술의 부분적인 지원, 프로토타이핑의 실용화, 소프트웨어 유지 보수의 용이성, 소프트웨어 개발 과정의 단축, 정보 자원의 효율적인 운영 관리가 용이,부품 재사용으로 인한 생산성과 품질 향상 등의 장점이 있다 [McClure,1989] . 도입에 대한 성공요인은 경영자의 적극적인 지원, CASE 도구의 도입에 따른 전략 여부, 사용자의 참여 및 커뮤니케이션, 기업내의 소프트웨어 개발에 있어서의 방법론에 대한 표준, 절차, 도구의 선정 및 기준 등을 들 수가 있다.

(2) CASE 도구 실패원인 [McClure, 1989 ; 남상조, 전응섭, 1992]

CASE 도구의 도입으로 인해 정보시스템 조직의 변화, 비지니스 프로세스 변화, 소프트웨어 개발 프로세스 변화등과 같은 긍정적인 효과를 주는데도 불구하고, CASE 도구의 도입후의 성공 사례는 일부 **Dupont, Touch- loss, Deer & co** 기업에 지나지 않는다. CASE 도구의 도입에 따른 실패 원인은 다음과 같다.

- ◆ CASE 도구의 기능에 대한 인식의 부족
- ◆ CASE 도구의 도입후 관리 미흡
- ◆ 개발 방법론 및 표준이 없고,개발 방법론 변경이 곤란
- ◆ CASE 도구의 도입을 소프트웨어 개발 프로세스 자체 변경에 대한 위험으로 인식하여 가격에 비해 투자 효과 미흡으로 인식
- ◆ CASE 도구의 기술 도입에 대한 기업의 계획 부족
- ◆ CASE 도구에 대한 훈련 및 교육 부족
- ◆ 경영층 지원 부족
- ◆ CASE 도구를 선정 및 평가하는 전문가의 부족 등.

CASE 도구의 도입 및 활용에 대한 성공 및 실패 요인들을 살펴보면, 기업이 새로운 정보 기술을 도입함에 있어서 기업 내적인 요인이 큰 영향을 주고 있는 것으로 나타났다. CASE 도구의 도입에 성공한 사례를 보면 명확한 전략과 관리 통제, 개발 방법론의 표준화, 교육 훈련 등의 절차 및 관리에 대한 명확한 정의가 된 후에 도입을 추진한 것으로 나타나고 있다 [박성주,1991; 경영과 컴퓨터, 1990]. CASE 도구의 도입 및 활용 차원에 대한 효과를 사례를 통하여 보면 일부 기업은 도입 효과를 보고 있지만 대부분 기업은 도입 효과를 보지 못하고 있는 실정이다. CASE 도구를 도입한 기업에서 도입 효과가 높거나 낮게 나타나는 원인을 **CASE 도구의 도입 실패 원인 8가지**를 통하여 보면 거의 전부 기업의 내부 문제에서 비롯되고 있음을 알 수 있다.

WestingHouse의 사례에서는 CASE 도구의 도입을 추진함에있어 CASE 도구에 관련된 프로젝트 관리, 개발방법론에 대한 교육 및 훈련, 개발방법론의 표준화, 관리 절차의 확립, 사용자 요구사항의 수행 여부, CASE 도구의 선정 및 평가 기준 등을 정립하기 위하여 7 년간의 노력 후에 CASE 도구를 도입하였고 그에 따라 **Westinghouse** 의 도입 목적을 달성할 수 있었다. 이것은 먼저 기업

의 내부 능력을 배양한 후에 CASE 도구의 도입을 추진하여야 도입 효과를 달성할 수 있다는 것을 의미한다 [Mosley, 1992].

제 3 장 프로세스 성숙 체계

3.1 정의[Humphrey, 1989 ; Bollinger, McGowan. 1991]

프로세스 성숙 체계(Process Maturity Framework)란 소프트웨어 품질 및 생산성을 개선하기 위한 목적으로 현재 기업의 소프트웨어 프로세스의 문제점을 분석하고 개선해야 할 요인을 제시하는 이론 체계이다. 소프트웨어 프로세스란 소프트웨어를 개발하는 업무에 사용하는 도구, 방법론, 정책등에 관련된 활동의 집합으로서 소프트웨어 개발 프로젝트에서 다루어져야하는 활동과 프로세스에 대한 현재 상태에 관한 정보를 알기 위하여 사용되는 것이다. 소프트웨어 프로세스는 보다 나은 소프트웨어를 개발하기 위하여 조직 능력을 개선하고 계획에 따라 소프트웨어를 개발하는 것을 그 관리목적으로 한다. 소프트웨어 개발에 있어 기본 원칙으로 제시되는 것이 통계적 품질 관리 및 통제이다 [Humphrey, 1989 ; Mi and Scacchi, 1992]. 본 연구에서는 소프트웨어 프로세스를 이해하기 쉽게 소프트웨어 개발 프로세스로 정의하여 사용하기로 하겠다. 프로세스 성숙 이론 체계는 5 단계로 초기 단계, 반복 단계, 정의 단계, 관리 단계, 최적 단계로 나누어진다. 그림3.1는 프로세스 성숙 체계의 진화단계를 보여주는 것으로, 각 개별 단계에 대한 내용을 자세히 살펴보면 다음과 같다.

3.2. 프로세스 성숙의 제 단계

1) 초기단계 (Initial Level)

소프트웨어 개발 프로세스 측면에서, 이 단계는 공식화된 소프트웨어 개발 절차나 비용 추정, 프로젝트 계획 없이 운영되고 있는 혼란한 조직이다. 또한 통합되거나 표준화된 도구가 없고, 관리 및 통제에 관한 측면에서 중간 관리자의 지원, 소프트웨어 개발상의 문제에 대한 이해가 없으며,이를 회피하거나 잊어버리는 경향이 있다. 소프트웨어 개발자 측면에서는 정보 시스템 요구 사항 및 기술 숙련도가 낮은 단계이다. 이 단계에서 조직이 진화하기 위해 프로젝트 관리, 프로젝트에 대한 검토, 품질 보증, 변화에 대한 관리 등에 관심을 가져야한다.

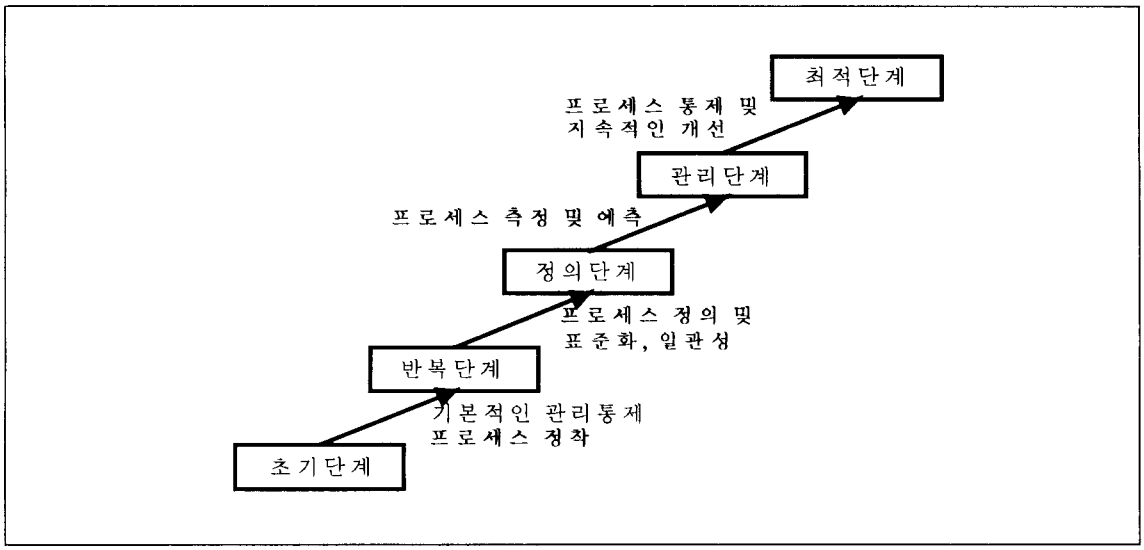

그림3.1 프로세스성숙진화 단계[Humphrey, 1989; Paulk and Curtis, Chrissis and Weber, 1993]

2) 반복단계 (Repeatable Level)

초기 단계에서 숙련된 업무를 반복적으로 수행함으로서 인적 자원의 경험이 축적된 단계를 말한다. 즉, 프로젝트, 비용, 스케쥴 및 변경 관리등의 반복적인 업무를 수행하여 어느정도의 통계적 관리가 실시되고, 소프트웨어 개발 프로세스는 안정적이지만 문서화는 부족한 단계이다. 프로세스 개선을 위한 순차적인 체계가 부족한 단계이며, 따라서 이 단계에서는 개선에 필요한 우선 순위를 분야별로 확인하고 평가하기 위한 체계가 필요하다. 다음 단계로 진화하기 위해 관심을 가져야하는 요인은 소프트웨어 개발 프로세스를 정의하고 개선을 위한 프로세스 그룹 설립하고, 개발 업무 및 기능의 기술, 절차의 기준 등을 위한 소프트웨어 개발 프로세스 아키텍처의 수, 정보공학의 개념과 기술을 도입 등과 같은 관심 요인에 대한 전사적인 지원과 지속적인 교육 및 훈련이 실시되어야 한다.

3) 정의단계 (Defined Level)

소프트웨어 개발 프로세스는 정성적이고, 프로 세스의 효과적인 진화를 위한 지표가 되는 데이타가 없고, 프로세스에 대한 정의가 되어 있지 않는 단계이다. 이 단계에서 초점은 구체적인 개발 프로세스 업무에 대한 측정에 있고, 효과적인 측정을 위하여 프로세스 아키텍처가 필수요인이다. 다음 단계로 진화하기 위해 개별 프로세스의 비용 및 품질 수준을 확인하기 위하여 최소한의 프로세스를 측정, 소프트웨어 개발에서 생성되는 자원을 관리하거나 유지하기 위해 프로세스 데이

타베이스 유지, 개발된 개별 소프트웨어의 상대적인 품질을 평가하고 , 품질목적이 달성되지 않은 프로세스를 분석 등에 대한 관심을 가져야 한다. 정의 단계에서는 프로세스에 대한 정의가 명확하고 정보 기술이 도입된 후 전반적으로 사용하기가 쉽고, 개발 절차 개선이 가능하며 새로운 정보 기술의 도입이 가능한 수준이다.

*Yourdon*에 의하면 , 기업이 CASE 도구를 도입하는 시기는 프로세스 성숙 이론 체계의 정의 단계(Defined Levl)에 있을때 CASE 도구를 도입을 해야 도입 목적을 성공적으로 달성할 수 있다고 한다 [Humphrey, 1989; Paulk,etc, 1993: 경영과 컴퓨터, 1992].

4) 관리단계 (Managed Level)

관리 단계의 초점은 개발된 데이타의 유지 및 관리에 있으며, 이로 인해 축적된 데이타는 소프트웨어 개발 및 지원시 중요한 참고 데이타로 사용된다. 또한, 소프트웨어 개발 프로세스 수준이 개별적으로 측정이 되는 단계이며, 소프트웨어 개발 프로세스 관리 요소가 정성적인 단계에서 정량적인 단계로 전환되는 시기로 제한된 범위내에서 소프트웨어 개발 시간, 스케쥴, 에러 측정 및 품질 수준 등에 대한 정성적 예측이 가능하다.

관리단계에서 최적화 단계로 진화하기 위하여 자동화된 프로세스 데이타의 축적을 위한 지원이 필요하고, 프로세스 데이타는 소프트웨어 개발에 관련된 문제 및 효율적 개선을 위한 예방, 분석 및 변경을 위해 사용 등에 관심을 가져야 한다.

이 단계에서는 기업의 전반적인 소프트웨어 개발 프로세스의 측정과 품질 개선이 시작되는 단계이다. 소프트웨어 개발과정에서 각 개발 프로세스별 측정이 가능하고 진보된 기술 도입이 가능한 수준을 말한다.

5) 최적단계 (Optimizing Level)

최적화 단계는 소프트웨어 개발 프로세스 개선에 관심을 집중하고 , 관리자는 소프트웨어 개발에 대한 프로세스 개선에 직접 관계된 데이타를 분석 및 축적하는 단계를 말한다. 이 단계에서는 부분적으로 소프트웨어 개발 프로세스를 개선하기 위해 지속적인 노력이 필요한 단계이다. 프로세스 성숙 이론 체계는 기업이 가지고 있는 소프트 웨어 개발 능력을 보여주고 능력에 따라 개선하고자 하는 요인을 제시하므로, 기업이 지향하는 목표를 효과적으로 달성할 수 있도록 지원한다. 기업은 소프트웨어 개발 프로세스가 적절한 수준으로 증가하면 CASE 도구의 도입을 검토하

고 추진해야 한다. 기업이 원하는 목표를 효과적으로 달성하기 위해서는 프로세스 성숙 체계의 정의 단계(Defined level)에서 CASE 도구를 도입해야만 한다 [경영과 컴퓨터 , 1992].

기업은 소프트웨어 개발 프로세스의 관리 통제 및 절차, 전략, 방법론, 비용, 교육 및 훈련, 형상 관리, 프로젝트 관리, 품질 보증, 개발 방법론의 표준화, CASE 도구의 공급자의 명성, 교육 및 컨설팅에 대한 지원 여부, 기존 어플리케이션과의 호환성, 조직의 도입 필요성 및 효과 등의 요인은 CASE 도입 및 활용을 위한 의사 결정 기준으로 사용되어야만 한다.

단계	특 징	관심사항	결과
최적 단계	- 개발 프로세스의 개선 - 자료수집의 자동화 / 프로세스 요소의약점을 개선하는 단계 - 개선업무에 정보기술이 필요 단계 - 오류원인 분석 및 예방단계	- 인적자원최적화 - 조직의 최적화	생산성 및 품질
관리 단계	- 정량적인 단계 - 프로세스의 측정 - 프로세스DB의 데이타분석 / 유지단계	- 기술변화 - 문제분석 - 문제예방	
정의 단계	- 정성적인 단계 - 프로세스 정의 및 제도화 단계 - 프로세스그룹을 구축하는 단계	-프로세스 측정 -프로세스 분석 -정량적품질계획	
반복 단계	- 인지단계 - 개별프로세스는 종속적 단계 - 반복작업의 강점 - 신기술 /방법론에대한 위험노 출단계 - 개선을 위한 순차적인 프레임웍이 부 족한 단계	- 훈련 - 기술적실행방침(검토, 시험) - 프로세스 초점 (프로세스그룹 표준)	
초기 단계	- 혼란단계 - 공식적인 절차 및 비용평가, 프로젝트 계획이 없음 - 절차에 대한관리기준이 없고 도구 통합되어 있지않다	- 프로젝트 관리 - 프로젝트 계획 - 형상 관리 - 소프트웨어 품질 보증	위험

표3.1 프로세스 성숙체계의특징 및 단계별 관심사항 [Humphrey, Snyder, and Willis, 1991]

제 4 장 연구 방법

4.1 연구 범위

정보시스템 조직 능력을 소프트웨어 개발 프로세스로 정의하고 소프트웨어 개발 프로세스가 CASE 도구의 도입 목적 달성에 중요한 요인으로 고려되어야 한다는점을 실증적으로 보이고자 한다. 그림 4.1은 연구 범위를 보여주는 것으로 연구 환경에는 조직 환경과 정보 시스템 조직 환경으로 구분할 수가 있다. 본 연구에서는 정보시스템 조직 환경에 국한하며 정보시스템 환경에서 CASE 도구의 도입 형태, 소프트웨어 개발 프로세스를 독립변수로 설정하였다. 정보시스템 조직의 2가지 요인을 분석함에 있어서 첫째, CASE 도구의 도입 형태에 따라 분류하여 소프트웨어 개발 생산성과의 관계를 분석하고, 둘째, 소프트웨어 개발 프로세스 수준을 측정하기 위하여 소프트웨어 개발 프로세스를 측정하여 조직의 소프트웨어 개발 수준과 생산성과의 관계, 셋째, CASE 도입형태와 소프트웨어 개발 프로세스간의 교호작용이 생산성에 미치는 영향을 연구하고자 한다.

4.2 연구 모형

위의 두 변수와 생산성과의 관계를 분석하기위한 연구모형은 그림 4.2 와 같다. 연구모형에는 2 개의 독립변수(Independent Variable)로 구분할 수가 있는데, 첫째, CASE 도구의 도입 형태로 이 독립 변수를 측정하기 위해 다음과 같이 분류한다.

- 통합 CASE 도구를 사용하는 기업
- 상위 CASE 도구와 4세대 언어를 사용하는 기업
- 상위 CASE 도구와 3세대 언어를 사용하는 기업
- 하위 CASE 도구를 사용하는 기업
- CASE 도구의 미 보유기업

둘째, 소프트웨어 개발 프로세스로서 정보시스템 조직 능력을 평가하는 것으로 소프트웨어 개발 프로세스에 관련된 하부변수에는 조직, 프로젝트관리, 프로세스 관리, 기술의 4가지 요인에 대한 기업의 소프트웨어 개발 프로세스 수준을 분류하여 각 수준별 생산성을 측정하였다 [Humphrey, 1989].

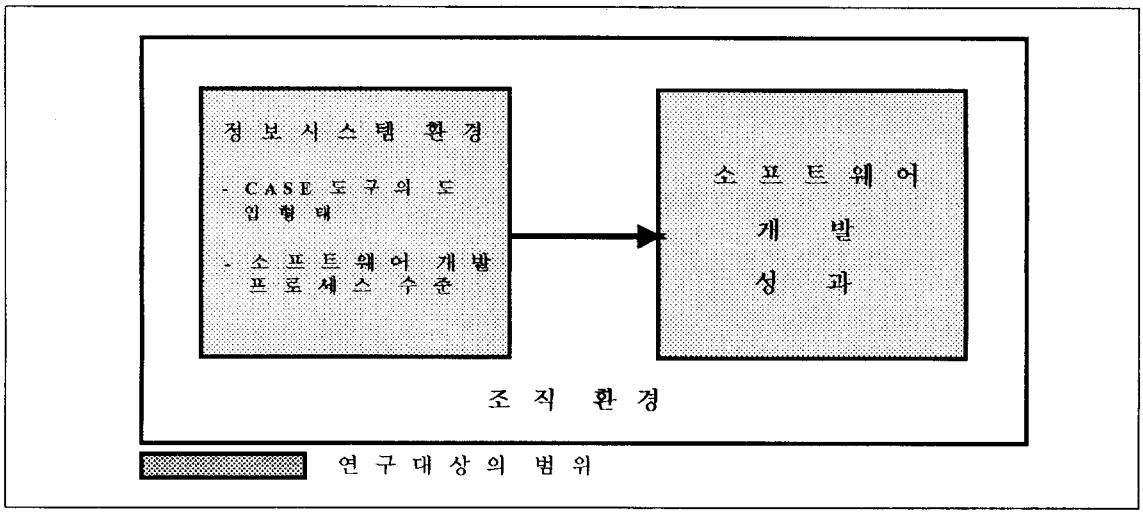

그림 4.1 연구 범위

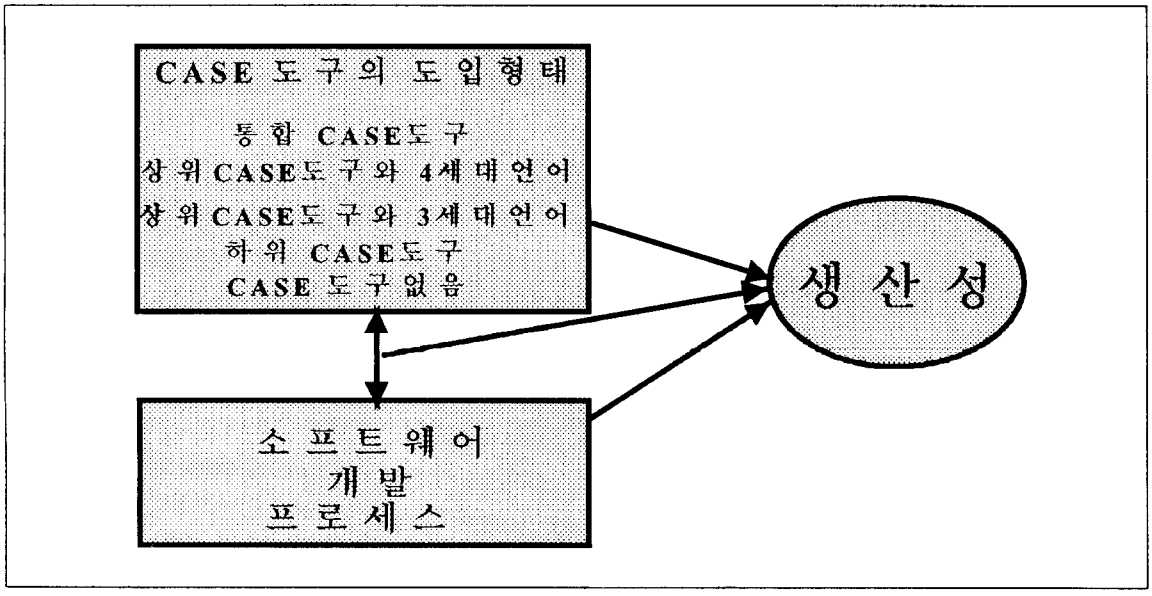

그림 4.2 연구모형

종속변수 (Dependent Variable) 에는 소프트웨어 개발 생산성으로 보았다. 소프트웨어 개발 생산성을 측정하기 위한 요인은 기업의 소프트웨어 개발에 있어서 사용하는 기능 점수(Function point)를 사용하였는데 기능 점수(Function Point)에는 입력 화면수, 출력화면수, 마스터 파일수, 조회화면수, 외부시스템 접속용 파일수의 다섯가지의 요인이 있다 [Lowand and Jeffery, 1990; Arthur,1984]. 생산성 항목을 측정할 때 개발된 소프트웨어의 기능 점수 5가지 항목의 합을 개발 기간과 투입 인원(Man-Month)으로 나누어 생산성 측정지표로 사용하였다.

4.3 가설

[가설1] CASE 도구의 도입 형태에 따라 생산성 차이가 있다.

정보공학의 차원에서 소프트웨어 개발 각 단계의 지원 여부에 따라 CASE 도구의 도입 형태가 분류되기 때문에 각 단계 활동에서 자동화된 도구의 지원에 따라 생산성이 다를 것이다. 즉 소프트웨어 개발 단계의 자동화 지원 수준이 높으면 생산성이 높고 자동화 지원 수준이 낮으면 생산성이 낮을 것이다.

[가설2] 소프트웨어 개발 프로세스 수준에 따라 소프트웨어 개발 생산성 차이가 있다.

기업의 소프트웨어 개발에 관련된 기업능력을 평가하는 것으로 소프트웨어 개발 프로세스 수준이 높으면 생산성이 높고 소프트웨어 개발 프로세스 수준이 낮으면 생산성이 낮을 것이다.

[가설3] CASE 도구의 도입 형태와 생산성의 관계는 소프트웨어 개발 프로세스 수준에 영향을 받는다.

CASE 도구와 생산성의 관계에서 소프트웨어 개발 프로세스의 수준에 따라 CASE 도구의 도입으로 인한 생산성 효과에 차이가 있을 것이다. 다시말해 CASE 도구의 도입이 생산성에 미치는 영향은 소프트웨어 개발 프로세스 수준이 높으면 더욱 강하게 나타날 것이고 소프트웨어 개발 프로세스 수준이 낮으면 미미할 것이라 기대된다.

[가설4] CASE 도구의 도입한 기업과 CASE 도구의 미보유 기업간의 생산성 차이가 있다.

CASE 도구는 정보 공학 개념을 소프트웨어 개발 방법론 전 단계를 자동화하였기 때문에 CASE 도구를 도입한 기업은 CASE 도구를 도입하지 않는 기업보다 소프트웨어 개발에 있어서 생산성이 높을 것이다.

4.4 표본선정

설문 조사의 표본 집단은 상장기업을 대상으로 하였으며 CASE 도구를 보유한 기업은 상장기업에 포함되어 있는 기업도 있으나 대다수가 상장 기업에 포함되어 있지 않는 기업이 많았다. 상장 기업에서 CASE 도구를 도입한 기업수가 적기때문에 CASE 도구를 보유한 기업에 대한 목록은 경영과 컴퓨터 '91, 7월호와 하이테크 컴퓨터 '94, 9월호의 CASE 도구를 도입한 기업 목록에서 64개 기업을 추출하였다.

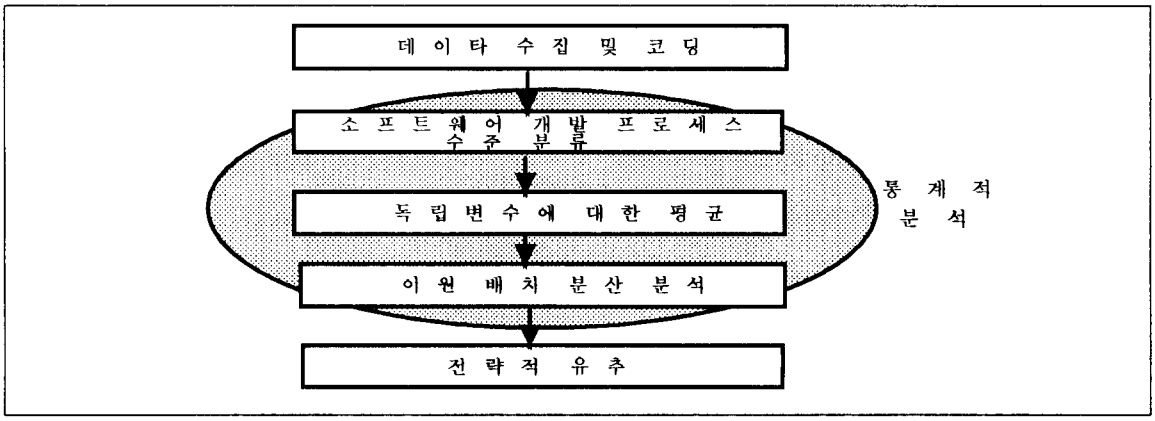

그림 4.3연구절차

4.5 연구절차

연구모델의 각 변수들에 대한 데이타의 수집 및 분석의 절차는 그림 4.3과 같이 5가지 절차로 구분할 수 있다. 1) 데이타 수집 : 개발된 설문지를 통하여 개별 기업체를 방문하거나 우편을 통하여 수집, 2) 그룹핑 : 개별 기업의 소프트웨어 개발 프로세스 수준을 5단계로 분류 (기업의 소프트웨어 개발 프로세스 수준을 5 단계로 분류하는 방법은**Bollinger**와 **MacGowan**의 SEI 분류 방법론으로 기업을 분류), 3) 독립 변수에 대한 평균 생산성 비교 , 4) 이원배치 분산분석 : 두 독립변수간의 교호작용을 분석, 5) 전략적인 의미 도출의 연구절차로 데이타를 수집 및 분석하였다.

4.6 자료의 정리 및 1차 분석

1) CASE 도구의 도입 형태와 생산성 분석

표 4.2을 보면 실제 기업의 도입 형태에서 통합 CASE 도구가 상위 CASE도구와 CASE 도구의 미 보유 기업보다 평균 생산성이 높으나 하위 CASE 도구보다 낮다. 하위 CASE 도구는 코드 생성에 대한 생산성만 중점을 두고 있으나 통합CASE 도구, 상위 CASE 도구는 문서화, 개체관계 모형도, 자료흐름도, 프로젝트 관리등의 소프트웨어 품질에 중점을 두고 있기 때문으로 해석된다.

구 분	프로젝트 수	평 균	전체비율
통합 CASE 도구	22	8.68	24
상위 CASE 도구와 4세대언어	23	6.73	25
상위 CASE 도구와 3세대언어	7	5.23	8
하위 CASE 도구	18	14.59	20
CASE 도구없음	21	3.75	23
합 계	91	7.95	100

표 4.2 CASE 도구의 도입 형태에 따른 평균 생산성 분석

2) 소프트웨어 개발 프로세스 수준과 생산성 분석

소프트웨어 개발 프로세스 수준에 따라 평균 생산성의 차이를 보이고 있는데 이는 소프트웨어 개발 프로세스 수준이 높을수록 생산성이 높고 소프트웨어 개발 프로세스 수준이 낮을수록 생산성이 낮다는 것을 알수 있다. 그러나 관리단계의 생산성이 높은 것은 이 수준에 해당되는 기업은 거의 하위 CASE 도구를 사용하는 기업이기 때문이다.

구 분	프로젝트 수	평 균	전체비율
초기단계	8	3.98	9
반복단계	13	4.65	14
정의단계	43	6.42	47
관리단계	18	14.86	20
최적단계	9	9.79	10
합 계	91	7.95	100

표 4.3 소프트웨어 개발 프로세스 수준에 따른 평균 생산성 분석

3) 업종별 평균 생산성 분석

업종별 평균 생산성을 분석한 결과를 보면 정보서비스, 전자 업종이 생산성이 높고 제조, 금융, 기타 업종은 생산성이 상대적으로 낮게 나타났다. 이것은 정보 관련 업종은 도구 사용과 우수한 인적자원으로 다른 업종보다 평균 생산성이 높고 새로운 정보시스템을 도입 및 운용하는데 상당한 능력을 보유하고 있기 때문으로 해석할 수 있다.

업 종	평 균	프로젝트 수
정보서비스	13.64	29
전 자	9.6	8
제 조 업	5.28	20
금 융 업	4.39	11
기 타	4.23	23

표 4.4 업종별 평균 생산성 분석

4) CASE 도구의 사용정도에 따른 생산성 분석

CASE 도구의 사용정도에 대한 평균 생산성 분석표에서 CASE 도구의 사용 정도가 높을수록 생산성이 높게 나타나고 사용정도가 낮을 수록 생산성이 낮게 나타났다. 기업은 CASE 도구의 사용 정도을 제고하기 위하여 정보공학 개념에 대한 방법론, 프로젝트 관리, 관리 절차에 관한 정책 수립 및 인적 자원에 대한 지속적인 교육 및 훈련을 통하여 최대한 CASE 도구를 활용할 수 있도록 하여야 한다.

CASE 도구의 사용정도	평 균	프로젝트 수
100 %	15.95	20
75 %	10.10	14
50 %	6.21	15
25 %	4.09	19
0 %	4.03	23

표 4.5 CASE 사용정도에 따른 평균 생산성 분석

제 5 장 연구분석 및 결과

5.1 연구분석

1) CASE 도구의 도입 형태와 생산성 차이에 관한 가설검정

표 5.1에서 보는 바와같이 분산 분석을 한 결과를 보면 CASE 도구의 도입 형태에 따른 소프트웨어 개발 생산성 차이는 F값을 보면 $F_0 = 2.171 > F_{(0.1, 4, 86)}$ 로 CASE 도구의 도입 형태와 생산성에 대한 [가설1]이 유의한 것으로 나타났다.

2) 소프트웨어 개발 프로세스 수준과 생산성 차이에 관한 가설 검정

표 5.1에서 보는 바와같이 소프트웨어 개발 프로세스 수준과 생산성에 대한 분산 분석을 한 결과를 보면 소프트웨어 개발 프로세스 수준에 따라 생산성 차이는 F 값을 보면 $F_0 > F_{(0.1, 4, 86)}$ 로 CASE 도구의 도입 형태와 생산성에 대한 [가설2]가 유의한 것으로 나타났다.

3) CASE 도입형태와 생산성 관계에 소프트웨어 개발 프로세스의 교호작용에 관한 가설검정

표 5.1에서 보는 바와같이 CASE 도구의 도입 형태와 소프트웨어 개발의 생산성에 소프트웨어 개발 프로세스의 영향에 대한 분산분석을 한 결과 를 보면 CASE 도구의 도입 형태와 생산성 관계에서 소프트웨어 개발 프로세스 수준에 따라 교호작용(Interaction) 은 F 값에서 $F_0 = 2.86 > F_{(0.1, 12, 70)}$ 로 소프트웨어 개발 프로세스 수준과 CASE 도구의 도입 형태에 따른 생산성의 관계에 대한 [가설3]이 유의한 것으로 나타났다.

4) CASE 도구 유무에 따른 생산성에 관한 가설검정

표 5.1에서 보는 바와같이 CASE 도구의 유무에 따른 생산성관계를 분산분석을 한 결과 CASE 도구의 유무와 생산성 관계에 차이는 F 값을 보면 $F_0 = 3.234 > F_{(0.1, 4, 89)}$ 로 CASE 도구의 도입 형태와 생산성의 관계는 [가설4]가 유의한 것으로 나타났다.

가설	분석 결과
가설 1	$F_0 = 2.171 > F_{(0.1, 4, 86)}$, $p = 0.079$*
가설 2	$F_0 = 2.163 > F_{(0.1, 4, 86)}$, $p = 0.080$*
가설 3	$F_0 = 1.87 > F_{(0.1, 12, 20)}$, $p = 0.003$**
가설 4	$F_0 = 3.234 > F_{(0.1, 4, 89)}$, $p = 0.076$*

표 5.1 가설에 대한 결과 ($p < 0.1$*, $p < 0.05$**)

5.2 교호작용(Interaction) 분석

[가설 3]의 이원배치 분산분석에서 교호작용(Interaction) 매트릭스는 어떤 CASE 도구를 도입할 것이며, 소프트웨어 개발 프로세스 어떤 수준에서 CASE 도구를 도입할지를 결정하는 두 독립변수간의 최적 전략을 설명하는 요인으로 이를 최적조합(Optimal Combination)이라 한다. 개별 교호작용(Interaction)을 분석하기 위하여 두 독립변수의 간의 교호작용 매트릭스에서 각 셀에 대한 평균을 산출하여 보면 표 5.2과 같다. 여기서 최적 조합은 하위 CASE 도구와 소프트웨어 개발 프로세스의 제 4 수준으로 나타났다. 표 5.2는 전체 교호작용을 개별 교호작용을 CASE 도구의 도입 유무에 따라 구분하고, 소프트웨어 개발 프로세스 수준 1,2를 한 셀로, 3, 4, 5를 다른 한 셀 구분한 것으로 소프트웨어 개발 프로세스 구분 기준은 ***Humphrey와 Yourdon*** 기준으로 CASE 도입이 가능한 제 3 단계를 기준으로 하였다.

기업수준 / 도입형태	1	2	3	4	5
CASE 도구 없음	3.78	2.62	3.77	3.98	5.18
통합 CASE 도구	0.13	9.14	6.26	8.76	13.84
상위 CASE 도구와 4세대언어	5.08	2.62	3.38	13.63	4.28
상위 CASE 도구와 3세대언어	0	0	6.48	4.29	0
하위 CASE 도구	5.17	2.7	11.89	**57.9**	0

표 5.2 개별 교호작용에 대한 생산성 매트릭스

기업 수준 / 도입 형태	S/W 개발 프로세스 수준 1	S/W 개발 프로세스 수준 2
CASE 도구 없음	**셀 1** 평균 : 3.19	**셀 2** 평균 : 3.97
CASE 도구 있음	**셀 3** 평균 : 4.88	**셀 4** 평균 10.40

표 5.3 교호작용 생산성 매트릭스 셀의 평균 생산성

위 기준에 따라 구분된 각 셀의 평균을 구하면 표5.3과 같다. 각 셀의 평균에 대하여 다음과 같은 2 가지의 가설을 세워 개별 셀에 대한 평균 생산성 차이를 t-테스트검정 기법을 통하여 가설 검정하여 전략적 의미를 파악하고자 한다.

첫째, [가설3 - 1] CASE 도구의 유무에 따른 생산성의 차이는 소프트웨어 개발 프로세스 수준에 따라 다르다.

표5.8_A는 위의 가설을 검정한 T-검정표로서 소프트웨어 개발 프로세스 수준 1에 대한 셀 1과 셀 3 사이에 평균 생산성 차이가 없는 것으로 분석되고, 소프트웨어 개발 프로세스 수준 2는 셀 2와 셀 4 사이는 평균 생산성 차이가 있는 것으로 나타났다.

둘째, [가설 3-2] 소트웨어 개발 프로세스 수준에 따른 생산성의 차이는 CASE 도구의 유무에 따라 다르다.

표5.8_B는 위의 가설을 검정한 T-검정표로서 CASE 도구가 없는 경우에는 셀 1과 셀 2 사이에 평균 생산성 차이가 없고, CASE 도구가 있는 경우에는 셀 3과 셀 4 사이에 평균 생산성 차이가 있는 것으로 나타났다.

표 5.4는 [가설 3-1,2]를 검정한 결과로이것은 다음과 같이 해석할 수 있다.

- 소프트웨어 개발 프로세스 수준이 낮으면 CASE 도구의 도입 형태와는 생산성 향성에 아무런 교호작용이 없다.
- CASE 도구가 생산성 증대 목적으로 도입한다 할지라도 소프트웨어 개발 프로세스 수준이 낮으면 투자 효과가 거의 없고 막대한 투자 비용만 초래한다는 것을 의미한다.
- 소프트웨어 개발 프로세스 수준이 높으면 CASE 도구의 도입 형태에 따라 평균 생산성이 다르게 나타났고, CASE 도구가 있으면 더 큰 생산성 향상 효과가 있다고 할 수 있다.
- CASE 도구가 없을때 소프트웨어 개발 프로세스 수준에 따른 생산성 향상과는 아무런 교호작용이 없다.
- CASE 도구가 있을때 소프트웨어 개발 프로세스 수준에 따라 평균 생산성이 다르게 나타났다. 이것은 CASE 도구가 있으면 소프트웨어 개발 프로세스 수준이 높을수록 생산성 향상 효과가 크다.

구분	S/W 개발 프로세스 1	S/W 개발 프로세스 2
CASE 무	t-값 - 0.81 신뢰구간 (- 4.792, 1.423) H_0 : 채택 ($u_0=u_1$)	t-값 - 2.97 신뢰구간 (- 9.580, - 3.267) H_0 : 기각
CASE 유		

A) 소프트웨어 개발 프로세스에 따른 생산성 비교 분석 (P < 0.1)

구분	S/W 개발 프로세스 1	S/W 개발 프로세스 2
CASE 무	t - 값 - 0.59 신뢰구간 (- 2.538, 0.975) H_0 : 채택 ($u_0=u_1$)	
CASE 유	t-값 - 2.32 신뢰구간 (- 8.605, - 2.436) H_0 : 기각	

B) 소프트웨어 개발 프로세스과 CASE 형태 유무와 생산성 비교 분석(P < 0.1)

표 5.8 개별 셀에 대한 평균 생산성 T- test 분석

5.3 최적 CASE 도구의 도입 및 활용 전략과 교호작용

앞에서 교호작용 메트릭스를 통한 4가지의 셀로 구분하였는데 개별 셀에 포함된 기업은 생산성 효과를 달성하기 위하여 기업 외부 환경(공급자 지원 여부, 도구의 기능, 호환성)과 기업 의 목적인 고품질의 소프트웨어 개발, 생산성 향상과 정보자산관리를 위하여 CASE 도구를 도입할 때 기업의 현재 소프트웨어 개발 프로세스 수준에 따라 최적 CASE 도구의 도입 및 활용 전략을 수립하여야 도입 효과를 제고 할 수 있다. 그림 5.1는 교호작용과 생산성과의 관계를 나타낸 매트릭스로 해당 셀에 그 특징을 나타낸 것이다. 그림 5.2에서 보면 셀 1, 2, 3은소프트웨어 개발 생산성이 낮고 셀 4는 소프트웨어 개발 생산성이 높게 나타났다. 그림 5.2은 개별 셀에 대한 생산성 향상을 위한 전략적 의미을 나타낸 것으로 해당 셀에 대해기업이 취해야 할 구체적인 전략은 다음과 같다.

1) 전략 I

- 소프트웨어 개발 프로세스 수준을 높이기 위하여 정보공학 방법론에 대한 교육, 관리절차 수립, 개발 프로세스 정립과 CASE 도구의 기술을 습득하여 소프트웨어 개발 생산성 효과를 위하여 노력하여야 한다.
- 소프트웨어 개발 프로세스 수준이 증가되고 난 후 기업의 도입 및 활용목적을 효과적으로 달성할 수 있는지 여부와 CASE 도구의 도입 형태를 결정하여야 한다.

2) 전략 II

- ◆ 소프트웨어 개발 프로세스 수준을 유지하기 위하여 부분적인 프로세스 개선을 통하여 현 수준을 유지하여야 한다.
- ◆ 소프트웨어 개발 프로세스 수준은 CASE 도구를 도입하더라도 충분히 운용할 수 있는 능력을 보유한 기업으로 소프트웨어 개발에 있어 CASE 도구의 도입은 기업 도입 및 활용 전략에 따라 CASE 도구를 선정하거나 도입 형태를 결정해야 한다.

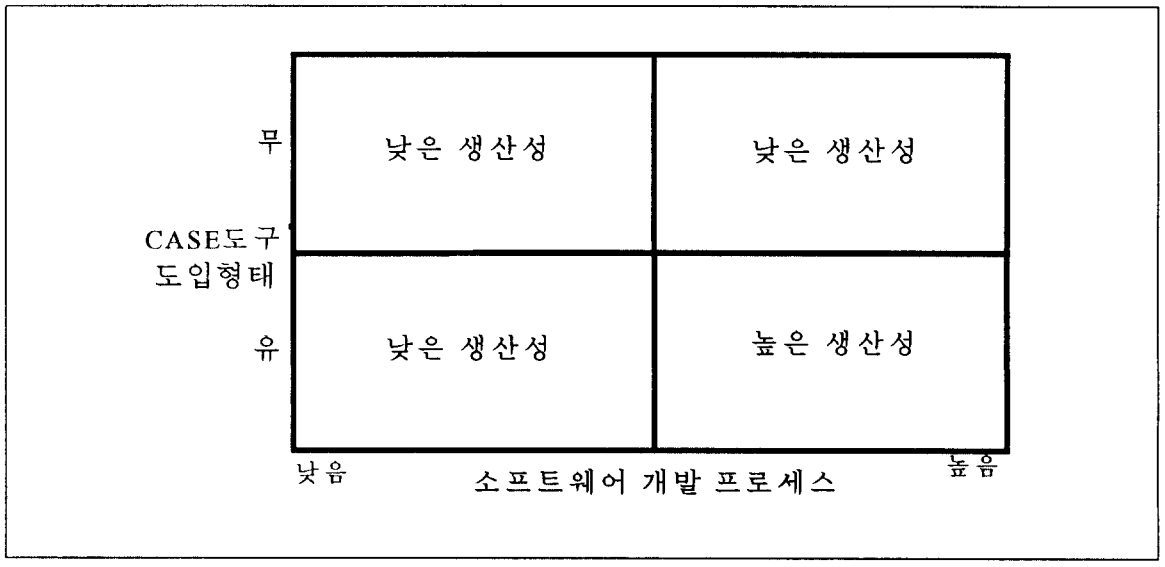

그림 5.1 교호작용(Interaction)과 생산성 관계의 메트릭스

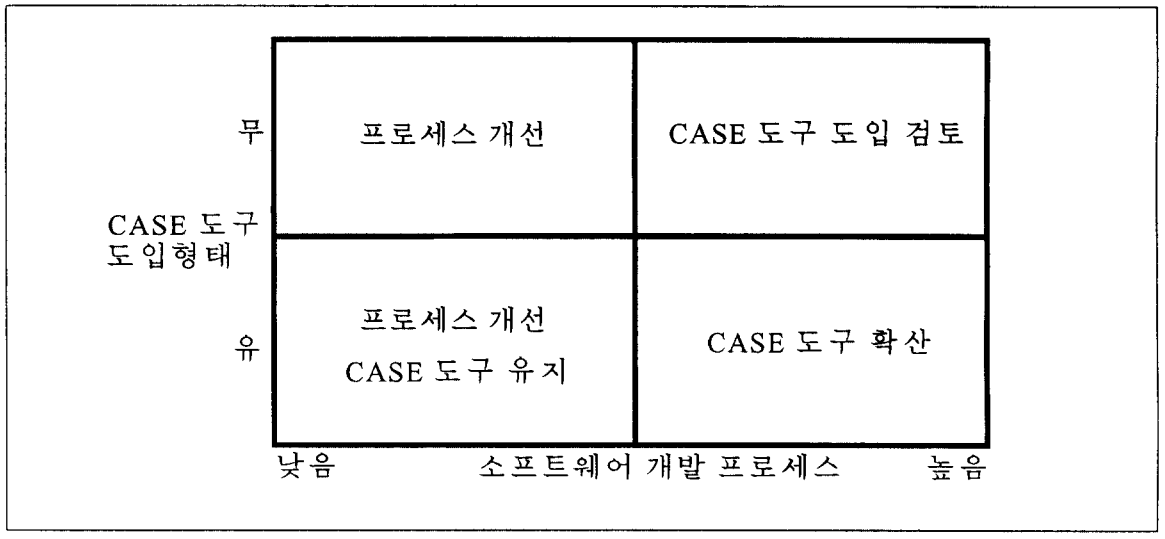

그림 5.3 최적 CASE도구의 도입 및 활용 전략

3) 전략 III

- ◆ 소프트웨어 개발 프로세스 수준을 높이기 위하여 정보공학 방법론에 대한 교육, 관리절차 수립, 개발 프로세스 정립과 CASE 도구의 기술을 습득하여 소프트웨어 개발 생산성 향상을 위하여 노력하여야 한다.
- ◆ CASE 도구의 활용을 위하여 CASE 도구의 운영 기술 습득 및 생산성과 품질, 정보자산관리 차원에서 활용 방안을 수립하여 CASE 도구를 사용을 유지하거나 촉진시키기 위해 노력해야 한다.

4) 전략 IV

- ◆ 소프트웨어 개발 프로세스 수준의 부분적 개선을 통하여 현재의 소프트웨어 개발 프로세스 수준을 유지하여야 한다.
- ◆ CASE 도구의 활용을 극대화 하기 위하여 인적자원에 대한 지속적 교육 훈련과 CASE 도구의 운영 기술의 습득에 많은 투자가 필요한 시기이다.
- ◆ CASE 도구의 사용은 정보시스템 부서에 국한 하지말고 사용자 부서까지 확산되도록 노력하여야 한다.

위 4가지 전략을 수행하기 위하여 기업은 현재 기업의 소프트웨어 개발 프로세스 수준을 분석하고 프로세스에 대한 문제점을 해결하기 위한 정책, 관리 절차, 정보공학 방법론 교육 및 훈련을 통하여 수준을 높이는데 노력을 해야 하며 소프트웨어 개발 프로세스가 적절한 수준이 되면 CASE 도구의 도입을 통하여 소프트웨어 개발 생산성 향상을 달성할 수 있을 것이다.

마지막으로 기업의 최적 CASE 도구의 도입 및 활용전략을 수립하기 위하여 기업 외부 환경에 대한 분석과 기업 정보시스템 조직의 정보공학 방법론, 관리 방침, 개발 도구에 대한 표준화, 사용정도에 관한 조직, 프로젝트 관리, 개발 프로세스 관리, 기술 수준등을 분석하여 소프트웨어 개발 프로세스 수준을 측정한 후 적절한 수준이 되면 CASE 도구의 도입 여부를 검토하고 CASE 도구의 도입 형태를 결정한 후 CASE 도구의 도입을 추진하여야 한다. CASE 도구의 도입 형태와 활용에 있어서 기업의 소프트웨어 개발 프로세스 수준에 따라 효과가 다르기 때문에 기업은 외부 환경과 기업 내부 환경을 이해하여 CASE 도구의 도입 및 활용에 대한 전략을 수립하여 기업의 중장기 전략에 반영하고 도입 효과를 얻을 수 있도록 노력해야 한다.

제 6 장 결론

본 연구는 기업이 언제, 어떻게 CASE 도구를 도입 및 활용하고 있는가 대한 실제 자료를 수집하였고 CASE 도구의 도입형태와 소프트웨어 개발 프로세스 수준이 기업의 소프트웨어 개발 생산성에 미치는 영향에 관한 분석을 통하여 최적 CASE 도입 및 활용전략 수립을 하였다. 최적 CASE 도구의 도입 및 활용에 대한 연구결론은 다음과 같다.

- 현재 기업의 소프트웨어 개발 프로세스 수준을 고려해야 한다.
- 기업 전략에 따른 CASE 도구의 도입 형태 및 활용을 결정해야 한다.
- CASE 도구에 대한 정보공학적 방법론에 대한 교육 및 훈련, 관리 절차 수립, 기술 습득, 도구의 선정 및 평가 기준을 수립하고 인적자원에 대한 지속적인 교육 및 훈련이 필요하다
- 기존 도입된 CASE 도구의 활용을 위한 촉진 전략이 필요한다.

최적 CASE 도구의 도입 및 활용 전략 수립에 있어서 본연구 기여도는 아래와 같다.

- 최적 도입 및 활용 전략 수립을 위한 새로운 모델 제시
- CASE 도구의 도입 및 활용에 대한 기업 능력의 중요성 인식
- 연구모델의 교호작용에 대한 전략적 의미를 부여

이러한 최적 CASE 도구의 도입 및 활용 전략 은 기업의 중장기 전략 계획에 포함되어야 하며 이 계획은 기업 목적을 달성할 수 있도록 합리적이고 유용성을 제공할 것이다. 연구 한계는 CASE 도구를 도입한 기업수가 적고, 도입 기간이 짧아 설문 조사가 곤란하고, 생산성만으로 CASE 도구의 도입 및 활용 효과를 측정한다는 것은 단면만을 고려한 것으로 여겨진다. 기업 능력을 소프트웨어 개발 프로세스 수준으로 측정된 것은 전체 조직의 능력에서 극히 부분적이라고 할 수 있다.

이러한 연구 한계를 극복하기 위하여 추후 연구 방향은 다음과 같다.

- CASE 도구의 도입과 활용에있어 생산성과 품질, 정보 자산 관리측면에서 연구되어야 한다.
- 사용자 컴퓨팅, 다운사이징, 분산 시스템, 클라이언서버 환경 등의 환경 변화에 알맞는 CASE 도구의 도입 및 활용에 대한 연구가 있어야 한다.
- 정보시스템 조직이 아닌 조직 전체의 능력을 대상으로 측정 할 수 있도록 해야 한다.

참고 문헌

1 박성주, " CASE " 한국경영정보학회 춘계학술대회, 1991. 5, pp 143 - 172.

2 박성현, " 회귀 분석 ", 1986, pp 371 - 395.

3 오택섭, " 사회과학 데이타 분석법 SPSS/PC+", 1990, pp 97 - 115.

4 이종수, " CASE는 왜 필요한가 ? " 경영과 컴퓨터, 1990. 11, pp 120 - 126.

5 전응섭, 남상조, " 효율적인 S/W 개발을 위한 CASE 도구 활용의 실 증적 연구 " 경영정보학연구, 1993. 6, Vol. 3, No. 1, pp 31 - 52.

6 채서일, " Marketing Research ", 3th Edition, 1992, pp 397 - 444.

7 한국과학기술원, " SAS를 이용한 다변량 통계분석 ".

8 허명회, " SAS 분산분석 ", 1986, pp 45 - 61.

9 허문열, 송문섭 " 수리통계학 ", 1991, pp 389 - 405.

10 Applegate, L.M. "Technology Support for Coopratiwework : A Framework for Studying Introduction and Assimilation in Organizations" Journal of Organizational Computing, Vol.1. No.1, pp. 11- 39.

11 Arthur, L. J "Measuring Programmer Productivity and software Quality " John Wiley & Sons, 1984, chapter 2.

12 Bollinger, T. B. and McGowan, C. " A Critical Look at Software Capability Evaluation " IEEE Software, July 1991, pp 25 - 41

13 Chen, M. and Norman, R. J. " Integrated Computer - Aided Software Engineering (CASE):Adoption,Implementation,and Impacts " IEEE Software, 1992, pp 362 - 373.

14 Chen, M. and Norman, R. J. "A Framework for Integrated CASE" IEEE Softare, 1992, pp 18 - 22

15 Chen, M., Nunamaker, J. F. and Weber, S. " Computer - Aided Software Engineeirng : Present Status and Future Directions " DATABASE , Spring 1989, pp 7 - 13.

16 Curtis, B., krasner, H. and Iscoe, N. " A Field Study of the Software Design Process for Large Systems " Communication of the ACM, November 1988, Vol. 31, No. 11, pp 1268 - 1287.

17 Drucker, P. F. " The Coming the New Organization " Harvard Business Review , January / February 1988, pp 45 - 53

18 Everest, G. C. and Alanis, M. " Assessing User Experience with CASE Tools : An Exploratory Analysis " Proceeding of 1992 HICSS, Volumn II, PP 343 - 352. IEEE Software, 1992, pp 343 - 352.

19 Fuggetta, A. and Milano, P. D. and Cefriel.," A Classification of CASE Technology " IEEE Computer, December 1993, 25 - 38.

20 Green, P. E., Krieger, A. M. and Schaffer, C. M. " Quick and simple Benefit Segmentation" Journal of Advertising Research, June/ July, 1985, Vol. 25, No. 3. pp 9 - 17.

21 Hartwick, J. and Barki, H. " Explaing the Role of User Participation in Information System User " Management science, April 1994, Vol. 40, No. 4, pp 440 - 465.

22 Henderson, J. C. and Cooprider, J. G. " Dimension of IS Planning and Design Aids : A Functional Model of CASE technology " Information System Research, September 1990, 227 - 254.

23 Huff, C. C. " Elements of a Realistic CASE tool Adoption Budget " Communication of ACM, April 1992, Vol 35, No 4, pp 45 - 54.

24 Humphery, W. S. " Characteriing the Software Process : A Maturity Framework " IEEE Software, March 1988, pp 73 - 79.

25 Humphey, W. S. and Curtis, B. " Comments on ' a Criticle Look' " IEEE Software, July 1991, pp 42 - 46.

26 Humphrey, W. S. " Managing the Software Process " Addition-Wesley , 1989.

27 Humprey, W. S., Snyder, T. R. and Willis, R. R. " Software Peocess Improvement at Hughes Aircraft " IEEE Software, July 1991, pp 11 - 23.

28 Keen, P. G. W. " Shaping the Future : Business Design through Information technology " Harvard Business School, Boston, MA, 1991.

29 Kemerer, C. F. " How the Learing Curve Affects CASE Tool Adoption " IEEE Software, May 1992, pp23 - 28.

30 Low, G. C. and Jeffery, D. R. "Function Point in the Estimation and Evaluation of the Software Process " IEEE Software Engineering, January 1990, Vol. 16, No 1, pp 64 - 71

31 Mi, P and Scacchi, S. "Process Integration in CASE Environments" IEEE Software, March 1992, pp 45 - 53.

32 Mosley, V. " How to Assess Tools Efficiently and Quantitatively " IEEE Software, May 1992, pp 29 - 32.

33 Neuamnn, S. and Ahituv, N. " A Measure for Determining the Strategic Relevence of IS to the Organization " Information & Managment, 1992, pp 281 - 299.

34 Orlikowski, W. J. " CASE Tools as Organizational Change : Investigating Incremental and Radical Changes in Systems Development " MIS Quartly, September 1993, pp309 - 340.

35 Paulk, M. C., Curtis, B. and Chrissis, M. B. Charles.V.Weber "Capability Maturity Model Version 1.1," IEEE Software, July 1993, pp. 18-27.

36 Poston, R. M. and Sexton, M. P. " Evaluating and Selecting Testing Tools " IEEE Software, May 1992, pp 33 - 42.

37 Prell, E. M and Sheng, A. P. " Building Quality and Productivity into a Large System " IEEE Software, July 1984, pp 47 - 54.

38 Singh, S. K. " Using Information Technology Effectively Organization Preparedness Models " Information & Management , 1993, pp 134 - 146.

39 Sommer, R., Chen, M. and Sibley, E. H. " Computer - Aided Software Engineering Standards and Integration " IEEE Software, 1992, pp320 - 330.

40 Tate,G., Verner,J. and Jeffery, R. " CASE : A Testbed for Modelling , Measurement and Management " Communication of the ACM, April 1992, Vol. 35, No. 4, pp 65 - 72.

41 Vessey, L., Jarvenpaa, S.L. and Tractinsky, N. " Evaluation of Vendor Products : CASE tools as Methodology Companions " Communication of ACM, April 1992, Vol 35, No 4, pp 90 - 105.

42 Winant, R. " Establishing an Implementation Plan for Software Methods and CASE " Software Engineering Strategies, pp 18 - 29.

43 Yourdon, E. " 소프트웨어 미래, 이것이 승부수다 " 경영과 컴퓨터, 1992. 1, pp 165 - 169.