동시공학적 전문가시스템 개발방법론[1)]

박광호*

A Concurrent Engineering Approach to Expert Systems Development

요 약

전문가시스템이 등장한지도 근 20년이 흘렀고 다양한 분야에서 기업의 중추적 시스템으로 운영되고 있다. 이렇게 전문가시스템이 연구단계에서 벗어나 실용화됨에 따라 전문가시스템의 개발방법에 대한 연구가 학계와 산업계에서 새로운 연구과제로 인식되고 있다. 지금까지 전문가시스템의 개발에 대한 연구는 주로 지식획득 기술의 개발에 치중되어 왔다. 본 논문에서는 전문가시스템의 개발에 대한 방법론적 접근방법이 제시된다. 이 접근방법은 시간과 인력 등 자원의 제약을 받는 전문가시스템 개발프로젝트에 적합한 것으로 동시공학(Concurrent Engineering) 개념을 채택하고 있으며 개발 라이프사이클 모델, 프로젝트팀 구성방법 등을 다루고 있다. 또한, 동시공학적 접근방법을 지원하기 위한 분석과 설계방법으로는 객체지향기술을 사용하며 개발단계별로 해당 모델이 산출물로 구축된다.

Ⅰ. 서 론

전문가시스템은 기존의 정보시스템에 비해 내용면에서 지식이 차지하는 비중이 더 높다. 이것은 전문가로부터 지식을 도출하는 지식획득(Knowledge Acquisition)이 전문가시스템의 개발에 있어서 얼마나 중요한 위치를 차지하고 있는지만 봐도 잘 알 수 있다 (Buchanan et al., 1983), (Scot et al., 1991). 전문가시스템은 지식에 관련된 특성때문에 기존의 정보시스템과 다음과 같이 대별된다. 첫째, 전문가와 최종사용자가 프로젝트 전 과정을 통해 적극적으로 관여해야 한다. 또한, 개발자와 업무담당자 사이의 긴밀한 협조가 요구된다. 둘째, 전문가의 지식을 획득하기 위해서는 반복적인 지식도출 작업이 요구된다. 사실 전문가의 지식을 단시일에 모두 도출하는 것은 불가능하며 따라서 개발 기간이 장기화되는 것이 일반적이다 (Waterman, 1986). 세째, 획득된 지식에 대한

* 한양대학교 경영학부

1) 한양대학교 1994년도 교내 연구지원과제임.

타당성(Validation) 평가와 검증(Verification)이 지식 자체의 정성적 특성으로 인해 매우 어렵다. 네째, 지식과 정보는 특정 지식표현 구조, 예를 들어 규칙, 프레임 등을 중심으로 지역화된다.

한편, 이상의 특성에도 불구하고 전문가시스템 개발을 기존의 정보시스템 개발과 같이 취급하려는 경향이 있는데 이는 첫째, MYCIN(Shortliffe, 1976)의 출현으로 시작된 전문가시스템이 이제 연구실을 벗어나 AA(Authorizer's Assistant)와 같이 기업의 중추적 시스템으로 전문가를 대신하여 운영되고 있으며 (Dzierzanowski et al., 1989), 전문가시스템이 초기에는 LISP, PROLOG 등으로 개발되어 독자적인(Stand-alone) 환경에서 운영되었으나 최근 전문가시스템 개발도구가 C 언어를 기반으로 하기 때문에 기존의 정보시스템에 포함되어(Embedded) 사용되는 추세에 있다는 것이다 (Wolf et al., 1991), (Weitzel, 1989). 둘째, 전문가시스템이 정보시스템화됨에 따라 데이타베이스, 그래픽 사용자 인터페이스 등 일반적인 정보시스템의 구성요소도 전문가시스템 개발에 중요한 역할을 차지하게 되었다. 즉, 전문가시스템의 개발과정은 지식획득과정을 제외하고는 기존의 정보시스템과 동일하다고 볼 수 있다. 세째, 객체지향패러다임이 경영정보시스템, 의사결정지원시스템, 멀티미디어, 클라이언트-서버 시스템, 전문가시스템 등의 소프트웨어에 대한 차세대 통합기술로 등장함에 따라 (Martin and Odell, 1992) 단일화된 분석, 설계가 가능하게 되었기 때문이다.

전문가시스템의 개발방법론에 대한 연구는 대부분 지식획득 기술에 관련된 것들이다 (Wright and Ayton, 1987), (Kim and Courtney, 1988), (Abdul-Gader and Kozar, 1990), (Agarwal and Tanniru, 1990), (McGovern et al., 1991), (Byrd et al., 1992). 반면에 프로젝트 라이프사이클 관리, 프로젝트팀 조직, 분석, 설계기법 등 시스템 개발 전반에 걸친 방법론적 문제에 대해서는 연구가 빈약한 상태다. 그렇기 때문에 지금까지 전문가시스템 개발 프로젝트는 기존의 정보시스템 개발 방법론을 적용할 수 밖에 없었다. 그러나, 비록 전문가시스템이 정보시스템화됨을 인정할지라도 구조적 방법론(Structured Methodology), 폭포라이프사이클 모델(Waterfall Life Cycle Model) 등과 같은 기존의 방법론을 전문가시스템 개발에 직접 적용하는데는 무리가 있다. 따라서, 일반적으로 전문가시스템 개발이 장기화되고 초기개발단계에서 지연되어 시간과 자원 측면에서 문제가 발생하는 것을 감안할 때, 보다 효과적인 프로젝트 관리방법과 발생가능한 위험요인을 통제하기 위한 엔지니어링 접근방법이 요구되는 시점이다. 본 논문은 전문가시스템 개발에 대한 방법론적 문제를 다루고 있는데 전문가시스템 고유의 특성과 정보시스템적 특성을 통합적으로 고려한 프로젝트 진행관리, 프로젝트팀 구성, 모델 구축 등에 대한 접근방법을 제시하며 적용사례의 분석을 통해 실용성과 문제점을 분석해본다.

II. 프로젝트 진행관리

전문가시스템 개발과정은 지식획득과정의 연장선상에서 정의할 수 있다. 지식획득과정은 전문가로부터 지식을 도출, 표현, 테스트하는 일련의 과정으로 다음과 같이 공식적으로 정의되고 있다 (Buchanan et al., 1983) :

(1) 문제정의단계-개발계획 수립
(2) 개념화단계-분석
(3) 공식화단계-설계
(4) 구현단계-개발
(5) 테스트단계-테스트

2-1. 전문가시스템 개발 라이프사이클

이상의 지식획득과정에 사용자인터페이스, 데이타베이스 등 비지식적 요소를 첨가하면 전문가시스템 개발자체를 의미하게 된다. 그리고 일반적인 개발 라이프사이클의 정의(Berard, 1993)와 같이 전문가시스템 개발라이프사이클도 분석, 설계, 개발, 테스트의 4단계를 포함하는 것으로 보며, 따라서 개발계획수립 단계는 고려하지 않음을 밝혀 둔다.

전문가시스템 개발과정은 초기 분석단계가 기간면에서 상대적으로 길다. 이는 개발자가 업무에 대한 지식이 부족한 상태에서 전문가의 지식을 단시일에 이해하는 것이 불가능하기 때문이다. 이 때 분석의 대상이 되는 전문가의 지식과 업무정보는 다음과 같이 구분할 수 있다.

(1) 조직적 상황(Organizational Context)
(2) 의사결정과정
  a. 정보수집(Intelligence)
  b. 설계(Design)
  c. 선택(Choice)
(3) 문제해결 전문지식 및 모델
(4) 사용자인터페이스

조직적 상황은 개인의 의사결정에 중대한 영향을 주기 때문에 개발자는 우선 개발대상 업무의 조직적 상황이 의사결정에 미치는 영향을 파악해야 한다 (Dhar, 1987), (Simon, 1986). 업무에 관련된 주변상황을 불완전하게 분석했을 때, 개발자는 도출되지 않은 정보의 중요성을 제대로 평가할 수 없기 때문이다 (Berry, 1987). 조직적 상황을 분석한 결과는 개발대상 업무를 중심으로 한 도메인모델로 구축된다. 의사결정과정에 대한 분석은 전문가시스템이 무엇을 수행해야 하는지(What to do)를 결정하는 것으로 시스템이 제공해야 할 기능사양을 확정하게 된다. 의사결정과정은 Simon(1960)이 정의한 바와 같이 정보수집, 설계, 선택의 3 단계로 분석될 수 있는데 분석결과는 의사결정모델로 구축된다.

다음으로 문제해결방법(How to do)을 규명하여 전문지식과 모델을 도출하게 되는데 가장 긴 시간이 요구되는 단계라 할 수 있다. 경험지식과 모델은 문제특성에 따라 적절히 조합되는데, 예를 들어 스케쥴링 문제의 경우 경영과학이나 OR에서 제공하는 최적화 알고리즘과 업무담당자의 경험지식이 결합하여 문제를 해결하는 통합적 접근방법이 사용될 수 있다 (Lee, 1990). 도출된 경험지식과 모델은 규칙과 알고리즘으로 구성된 지식모델로 구축되는데 이 지식모델은 개발도구 선정의 기준이 된다. 패턴매칭위주의 규칙이 지식의 대부분이라면 전문가시스템 전용개발도구를 사용함을 고려해 볼 수 있으며 알고리즘위주라면 C, C++와 같은 일반 범용언어를 사용할 수도 있고 규칙과 알고리즘이 복합되어 있을 경우는 전용개발도구와 범용언어를 동시에 사용할 수 있을 것이다.

마지막으로 시스템의 효과적 운영을 위해서 사용자인터페이스에 대한 요구분석이 수반돼야 할 것이다. 사용자인터페이스에 대한 분석은 가장 단순하여 단기에 완료될 수 있으나 반복적인 수정작업이 요구되며 사용자인터페이스에 대한 분석결과는 인터페이스모델로 구축된다. 이상에서 분석단계는 도메인모델, 의사결정모델, 지식모델, 인터페이스모델이 구축되는 과정이라 요약할 수 있다 (그림 1).

설계단계에서는 분석단계에서 도출된 주요개념, 하

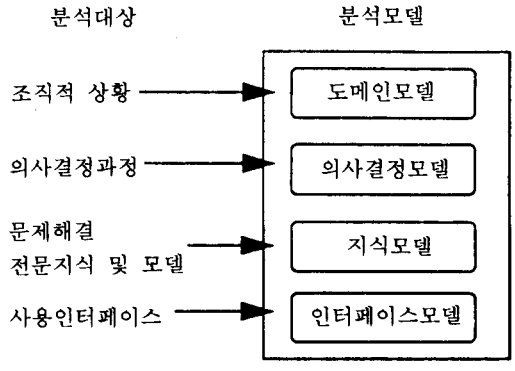

그림 1. 분석대상과 모델

부문제, 데이타흐름, 모델 등을 개발도구나 환경에서 제공하는 공식적 표현으로 전환하는 작업이 수행된다 (Buchanan et al., 1983). 분석결과에 대한 설계단계의 작업은 다음과 같이 세분화할 수 있다.

(1) 도메인모델의 설계
(2) 의사결정모델의 설계
(3) 지식모델의 설계
(4) 사용자인터페이스모델의 설계
(5) 통합모델의 설계

도메인모델에 대한 설계는 주로 주변 정보시스템과의 연계방법에 대한 것으로 의사결정과정의 정보수집단계와 중복되는 부분도 있다. 기존의 데이타베이스를 어떻게 사용할 것인지, 신규로 개발해야 할 기반시스템은 어떻게 개발할 것인지를 결정한다. 이때 신규개발이 요구되는 시스템은 별도로 개발을 추진하되 기존의 시스템과의 연계를 위한 설계는 통합모델의 설계에서 보다 구체적으로 수행된다. 의사결정모델의 설계는 전문가시스템 운영절차에 관한 것으로 시스템통제, 즉, 데이타 입력, 의사결정, 출력, 에러처리방법 등을 설계하게 된다. 의사결정모델에 대한 설계는 지식베이스, 사용자인터페이스, 네트웍인터페이스 등의 설계에 대해 매크로적 관계를 가지므로 이들에 대한 설계는 항상 의사결정모델 설계의 통제를 받아야 한다. 지식모델에 대한 설계는 도출된 지식에 대한 적절한 표현형태를 결정하는 것이다. 지식은 규칙, 케이스 등으로 표현되며 모델은 LP, 휴리스틱모델 등의 알고리즘으로 표현될 수 있다. 사용자인터페이스모델의 설계는 전문가시스템이 가장 효과적으로 사용될 수 있도록 텍스트, 그래픽, 오디오, 비디오 등을 사용하여 개발환경에 맞춰 최적의 인터페이스를 구상하는 작업이다. 마지막으로 통합모델의 설계는 시스템의 운영환경을 분석하고 설계된 여러 모델들을 시스템의 목적을 수행할 수 있도록 조직화하는 것으로 분석된 모델들을 통합하기 위한 구체적인 방법을 제시하는 것이다. 예를 들어, 클라이언트-서버 환경하에서 도메인모델, 지식모델, 인터페이스모델 등을 물리적으로 각각 어떤 하드웨어에 설치하고 이들사이의 인터페이스는 어떤 방법을 사용해야 하는지를 결정하는 것이다.

구현단계에서는 설계된 내용을 정해진 하드웨어에 정해진 개발도구로 구현하게 되는데 다음과 같이 세분화된다.

(1) 의사결정모델 구현
(2) 지식베이스 구현
(3) 사용자인터페이스 구현
(4) 데이타베이스인터페이스 구현
(5) 네트웍인터페이스 구현

테스트단계에서는 일차적으로 개발된 시스템의 각 모델이 정해진 사양대로 작동하는지를 테스트하게 된다. 따라서, 각 모델별로 해당 사용자나 전문가와 참여하여 다음과 같은 테스트작업이 진행된다 :

(1) 모델에 대한 테스트 사례 설계
(2) 테스트 기준, 즉, 성능, 해결안의 질, 시스템 운영 등에 대한 목표수준 결정
(3) 테스트 실행
(4) 차이분석및 피드백

모델별 테스트작업이 완료되면 모델들에 대한 통합테스트가 진행될 것이다. 통합테스트는 전문가시스템의 운영에 앞선 최종 테스트로 운영환경에서 사용자가 실사례를 가지고 전문가시스템의 기능을 테스트하는 것이다. 테스트단계에서는 프로젝트의 진행과 품질보증에 대한 관리도 수행된다. 예를 들어, 프로젝트 실행계획에 대비한 프로젝트 진행현황의 차이분석과 각 단계별로 작성된 문서에 대한 테스트가 실시된다.

2-2. 라이프사이클모델

정보시스템의 개발 라이프사이클 관리모델로는 순차적 폭포모델(Sequential Waterfall Model) (Benni-

ngton, 1956), (Boehm, 1986) ; b-모델 (Birrell and Ould, 1985) ; 나선형모델(Spiral Model) (Belz, 1986), (Boehm, 1986), (Boehm et al., 1984), (Boehm and Belz, 1986) ; 재귀적/병행적모델(Recursive/Parallel Model) (Berard, 1993) 등이 있다. 순차적 폭포모델은 가장 보편화된 라이프사이클 관리모델로 분석, 설계, 개발, 테스트의 개발 단계가 순차적으로 한 단계에서 다음 단계로 진행하는 방식을 취하고 있다 (그림 2). 이 모델의 특징은 어떤 단계가 착수되기 위해서는 반드시 사전단계 작업이 완전히 종료돼야 한다는 것이다. b-모델은 유지보수가 단순히 에러를 고치는 것이 아니라는 인식하에 구축되었다. 따라서 초기개발 완료 후에 발생하는 유지보수단계를 라이프사이클에 추가, 연장하고 있다 (그림 3). 이 때, 유지보수단계는 개발 전 단계를 포함하는 제 2의 개발로 해석된다. 나선형모델은 계획, 개발, 검토, 문제 규명 등 4 단계가 반복되어 최종시스템을 점진적으로 완성한다 (그림 4). 이 모델은 프로토타이핑방식을 사용하며 요구사항, 제약조건 등이 초기에 알려지지 않는 연구 프로젝트에 적합한 모델로 알려져 있다. 재귀적/병행적모델은 가능할 때 언제든지 분석, 설계, 개발을 진행한다. 시스템을 상위의 독립적인 구성요소로 분해하고 이 분해작업을 다시 분해된 하부 구성요소에 재귀적으로 적용하며 또한 재귀적 분해작업을 동시에 다수의 구성요소를 대상으로 수행한다.

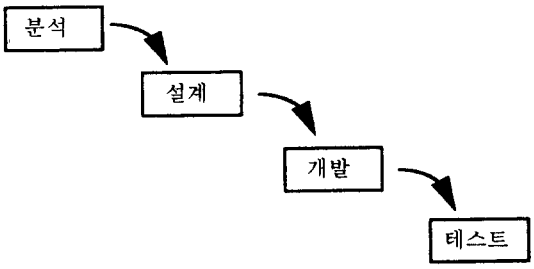

그림 2. 폭포모델

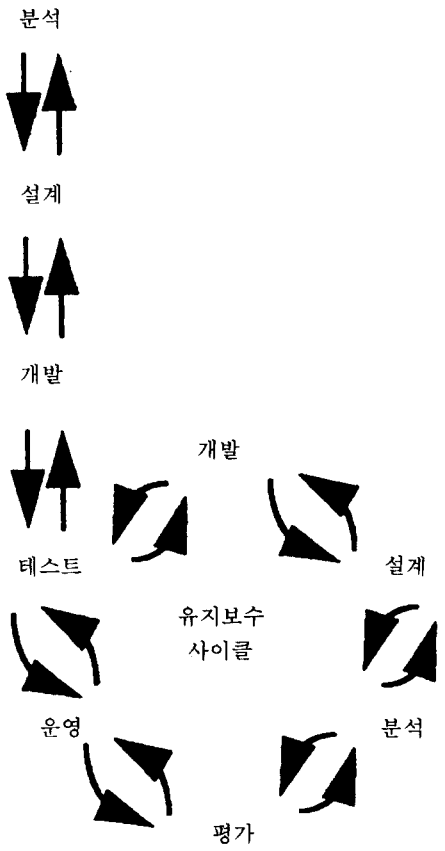

그림 3. b-모델

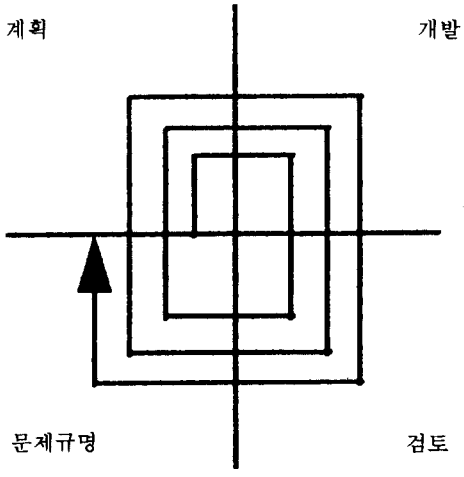

그림 4. 나선형모델

2-3. 동시공학적 모델

동시공학은 신제품개발에 대한 접근방법으로 관련된 작업들에 대한 통합을 의미하는데 (Nevins and Whitney, 1989), 제품설계를 팀웍 접근방법으로 보고 설계, 생산, 마케팅, 재무부서가 제품설계 초기부터 함께 참여하여 최종제품이 완성될 때까지 동시에 각자 작업을 수행하는 것이다. 우리는 동시공학적 접근방법을 전문가시스템 개발 라이프사이클 관리에 적용하고자 한다. 동시공학적 모델에서는 분업의 원칙이 가정되는데 분석, 설계, 구현, 테스트 등 각 단계별로 전담팀이 존재한다고 보는 것이다. 단계별 전담팀은 프로젝트 개발 초기부터 공동으로 참여하여 동시에 작업을 수행하게 된다. 또한, 동시공학적 모델은 개발단계내의 세부 작업에도 관련된다. 예를 들어 분석의 경우, 조직적 상황, 의사결정과정, 문제해결 전문지식및 모델, 사용자인터페이스 등 세부작업별로 전담팀을 구성하여 동시에 분석작업을 진행할 수도 있다. 따라서, 동시공학적 모델에서의 동시성은 그림 5에서와 같이 개발단계적 수준과 개발단계 내부적 수준의 두 가지 수준에 적용된다.

각 단계별 전담팀은 내부적으로 세 단계를 거쳐 작업을 수행하게 된다 : (1)준비(Preparation) ; (2)작업(Action) ; (3)정리(Wrap-Up). 분석팀은 프로젝트의 착수 시점에서 짧은 준비단계를 거쳐 바로 작업단계로 넘어갈 것이다. 그러나, 설계, 구현, 테스트팀은 일정기간의 준비작업 후에야 고유작업에 착수할 수 있을 것이다. 이런 준비기간을 고려한다해도 개발 초기부터 모든 팀이 함께 분석팀과 같이 분석작업에 참여해야 할 것이다. 개발대상 업무에 대한 이해는 모든 팀이 반드시 함께 수행해서 상호 개념적 오해가 없어야 하기 때문이다. 일단, 분석작업이 진행되면 다른 팀은 점차 팀 고유작업 영역이 명확해지고 이에 따라 고유작업에 대한 착수율이 증가하게 될 것이다. 따라서, 동시성 수준은 개발이 진행됨에 따라 예를

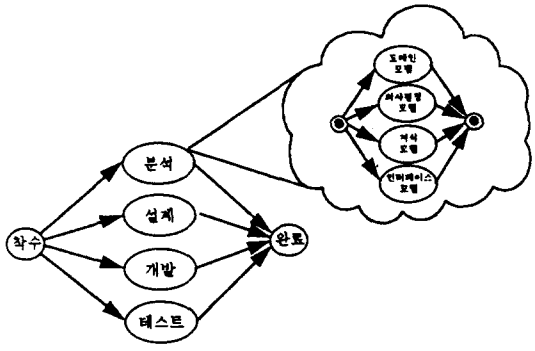

그림 5. 동시성

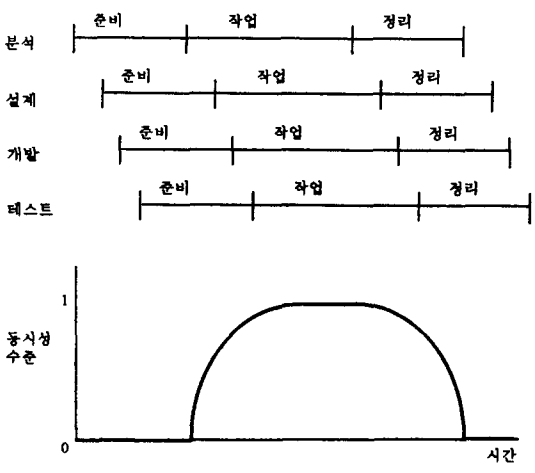

그림 6. 동시성 커브

들어 그림 6과 같이 변화할 것이며 프로젝트별로 변화과정에는 많은 차이가 있을 것이다.

각 단계별 전담팀이 개발 초기부터 모두 참여하는 것이 비효율적이라고 반박할 수 있다. 개발작업이 완전히 독립적인 하부작업으로 분해될 수 있고 이런 하부작업을 수행하는 과정에서 팀간에 대화할 필요가 전혀 없다면 인력을 단계별로 늘려 나가는 것이 효율적일 수 있다. 그러나, 전문가시스템의 개발단계는 상호 밀접한 관계를 맺고 있어 개발팀을 단계적으로 투입할 경우, 기존 팀은 신규 투입인원에게 업무를 설명하는데 많은 시간을 소비하게 되어 오히려 인력을 낭비하게 되는 결과를 낳게 될 것이다. Brooks(1976)

는 지연되고 있는 프로젝트에 신규 개발 인력을 투입하면 오히려 프로젝트를 더욱 지연시키게 된다고 지적한 바 있다. 동시공학적 접근방법은 전문가시스템 라이프사이클 관리에 있어 다음과 같은 장점을 가지고 있다. 첫째, 전체적인 시스템 무결성이 최소의 변동 수준에서 유지된다. 분석, 설계, 구현, 테스트팀이 동시에 작업함으로써 단계별 개념적 격차가 최소화될 수 있다. 둘째, 프로젝트 진행이 보다 효과적으로 관리된다. 세째, 개발진행상황이 최종사용자나 전문가에게 조기에 가시화된다. 마지막으로 각 단계별 오류가 조기에 발견되고 최소의 비용으로 수정될 수 있다. 결론적으로, 동시공학적 접근방법은 작업을 동시에 수행함으로써 개발기간을 효과적으로 단축하고 자원을 최소수준에서 낭비하는 목적을 달성할 수 있다. 그러나, 동시공학적 접근방법이 효과적으로 실현되기 위해서는 두가지 전제조건이 만족돼야 한다. 첫째, 프로젝트팀이 적절히 구성돼야 하고 둘째로, 시스템에 대한 개념적 무결성을 유지하고 프로젝트 관리자가 각 전담팀을 통제할 수 있도록 모델이 적절히 구축돼야 한다. 다음 두 장에서는 프로젝트팀 구성과 모델구축 방법에 대해 논의하게 된다.

III. 프로젝트팀 조직

동시공학적 접근방법은 분업을 원칙으로 함을 이미 지적한 바 있다. 전문가시스템을 개발하기 위한 개발팀은 이 분업을 원칙에 근거하여 적절히 조직돼야 한다. 개발팀은 그림 7과 같이 프로젝트 관리자를 중심으로 분석팀, 설계팀, 구현팀,테스트팀으로 구성된다. 분석팀은 내부적으로 이미 정의된 바와 같이 조직적 상황, 의사결정과정, 전문지식과 모델, 사용자 인터페이스 등 세부 분석작업별로 팀을 구성할 수 있다. 설계팀도 도메인모델, 의사결정모델, 지식모델, 사용자 인터페이스 모델, 통합모델 등의 세부 작업별로 인력을 구성하게 된다. 구현팀은 데이타베이스나 기존의 정보시스템과의 인터페이스를 위한 RPC(Remote Procedure Call), API(Application Programming Interface) 개발기술, MOTIF, VisualBasic 등의 사용자인터페이스 개발기술, C++, LISP, Prolog, ART-Enterprise, Nexpert, ProKappa 등의 지식베이스 개발기술별로 인력을 구성하게 된다. 동시공학적 접근방법에서 테스트팀의 역할은 매우 중요하다. 테스트팀은 단순히 개발된 모델에 대한 테스트작업뿐만 아니라 프로젝트 진행과 품질보증 등 프로젝트 관리자의 주요 기능을 지원하는 역할까지 담당하기 때문이다. 따라서 테스트팀은 모델에 대한 테스트 인력뿐만 아니라, 프로젝트진행관리, 품질보증 등에 대한 전문가도 포함해야 한다.

개발범위와 지식의 깊이에 따라 개발팀의 규모와 구조가 결정된다. 우선 개발범위가 클 경우에는 모든 작업이 비례적으로 증가한다고 볼 수 있으므로 프로젝트팀은 개발단계별, 또 세부작업별로 다수의 전담팀을 구성하는 경우도 있을 것이다. 반면에 지식의 깊이 수준은 지식과 모델에 관련된 작업에 대한 투입인력 규모에 영향을 준다.

개발인력은 앞에서 토론한 바대로 모두 개발초기부터 프로젝트에 참여해야 한다. 초기 준비작업을 거쳐 각 전담팀이 고유의 작업을 수행하게 되면 다수의 팀이 동시에 작업을 수행하는 현상이 당분간 유지될 것이다. 그러나, 동시성이 최고에 이른 후, 일정시간 흐른 후에는 선행 단계의 작업이 완료되는 시점에 다다르게 된다. 따라서, 개발 후기로 가면 개발팀은 점차적으로 정리 단계를 거쳐 철수하게 된다. 분석팀이 가장 먼저 철수하고 완료시점에는 구현팀 일부와 테스트팀이 철수하게 될 것이다.

다수의 개발인력이 동시에 작업에 착수하게 되면 프로젝트의 통제, 조정업무가 프로젝트 진행에 절대적 영향을 미치게 된다. 또한 각 개발팀간의 의사소통이 원활하지 못할 경우, 프로젝트의 지연, 품질문제가

예상된다. 따라서, 프로젝트 관리자는 각 전담팀의 효율적 작업진행을 지원해야 하며 이들 팀간의 갈등이나 불화, 의사소통문제 등을 최소화하고 조기에 이를 조정해야 한다. 가장 기초적으로 조직적 상황이나 의사결정과정에 대해서는 모든 팀이 동일한 개념을 가지고 있어야 할 것이다. 프로젝트 관리자가 이런 조정작업을 효과적으로 수행하기 위해서는 각 구성요소별 모델이 개념적 무결성이 유지되도록 구축돼야 한다. 모델은 참여한 모든 팀의 개발 진척상황을 대변하는 블랙보드다. 각 팀은 새로운 작업결과가 도출되면 즉시 이를 해당모델에 반영해야 한다. 이렇게 되면 프로젝트 관리자는 모델을 기준으로 프로젝트의 진행을 관리하고 갈등의 해결을 위한 조정회의의 대화도구로 사용하게 된다. 모델의 구조와 구축 방법에 대해서는 다음 장에서 토론하기로 한다.

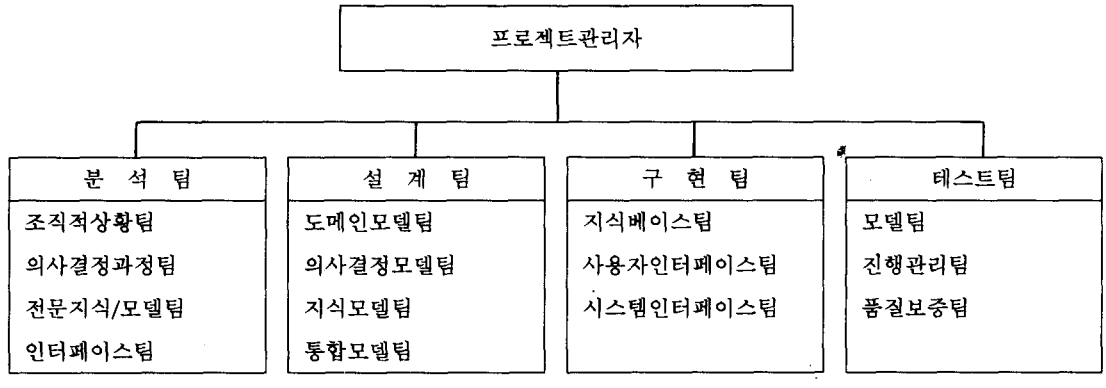

그림 7. 프로젝트팀 조직도

Ⅳ. 객체지향모델의 구축

전문가시스템의 개발과정에서 도출되는 모델은 도메인모델, 의사결정모델, 지식모델, 인터페이스모델, 통합모델 등이다. 본 논문에서는 이들 모델에 대한 구축방법으로 객체지향기술을 사용하는데 이에 대한 타당성은 다음과 같이 제시할 수 있다. 첫째, 도메인, 의사결정모델은 일반적인 MIS형 모델이라는 점에서 객체지향분석에 의한 모델구축이 무리가 없다고 볼 수 있다. 둘째, 사용자 인터페이스모델은 윈도우 환경하의 그래픽이나 멀티미디어 등이 객체지향개념에 기반을 두고 있으므로 객체지향 분석에 의한 모델구축이 요구된다. 세째, 통합모델은 클라이언트-서버와 같은 분산환경하의 통합이 객체지향 개념에 기반을 두고 있으므로 역시 객체지향분석에 의한 모델구축이 적합하다고 하겠다. 마지막으로, 지식모델은 과거 프레임(Frame) 구조가 객체 개념 속에 포함되며 규칙도 객체로 표현될 수 있다면 객체지향 분석에 의한 모델구축이 가능할 것이다. 본 연구의 목적중에 하나는 다양한 모델구축에 있어 통일된 구축패러다임의 가능성을 조명하는데 있다.

4-1. 객체지향 패러다임

객체는 현실세계에 존재하고 있는 실체(Entity)에 대한 추상체다. 실체는 그것에 관한 정보를 저장해야 할 필요가 있는 모든 것이며(Matin, 1989) 정보 공학에서는 이를 데이타로서 정의하고 있다. 반면에 객체지향에서는 실체를 데이타와 연산(Operation)을

결합한 객체로 정의한다 (Rumbaugh, 1992). 데이타는 보유하고 있는 속성을 의미하며 실체 자체가 가지고 있는 기본 특성뿐만 아니라 다른 실체와의 관계도 포함한다. 반면에 연산은 데이타를 이용하여 계산하는 함수(Function, Procedure)에 대한 정의(Definition)를 의미하며 구체적인 함수의 내용(Body)은 메소드(Method)라 칭한다. 과거에는 데이타와 분리되었던 연산이 객체라는 구조 속에 데이타와 결합된다는 개념이 객체지향과 정보공학을 위시한 기존의 방법론과의 근본적인 차이일 것이다.

사실 이런 광의의 객체의 개념은 엄밀히 클래스(Class)와 객체와의 관계 속에서의 협의의 객체 개념과 구별되어야 한다. 클래스는 추상데이타타입(Abstract Data Type)을 정의할 수 있도록 하는 객체지향의 중심개념으로 동일한 특성을 갖는 객체들을 대표하는 개념이다. 그리고 객체는 어떤 클래스의 구체적인 인스턴스(Instance)를 의미한다. 그러나, 일반적으로 객체를 광의의 개념으로 사용하는 것이 보통이며 따라서, 본 논문에서도 객체를 인스턴스보다는 객체지향을 대표하는 용어로 사용하기로 한다.

객체지향은 유전관계(Inheritance), 캡슐화(Encapsulation), 폴리모피즘(Polymorphism), 메시지 전달(Message Passing) 등의 개념으로 설명할 수 있다. 유전관계는 개념의 일반화(Generalization), 특수화(Specialization)를 지원하여 계층적 객체 구조와 이 구조상에서 상위 객체의 데이타와 연산이 하위 객체로 유전되는 것을 가능하게 한다. 캡슐화는 데이타와 연산의 외부적 관점인 정의와 내부적 관점의 구현을 분리하여 내부 구현의 변경이 외부 객체에게 영향을 주는 이른바 노콘 효과(Knock-on Effects)를 최소화 시키는 개념이다(Wilkie, 1993). 캡슐화는 정보은닉(Information Hiding)으로도 설명되는데 이는 데이타와 메소드를 외부로부터 은닉하고 오직 연산만을 외부 인터페이스방법으로 제공하기 때문이다. 폴리모피즘은 개념적으로 동일한 서비스를 수행하는 함수들을 하나의 이름을 사용하여 정의함으로써 객체에 대한 서비스 요청시 개념적 편리성을 제공한다. 즉, 하나의 연산(정의)에 대해 다수의 메소드(구현)가 존재하는 것을 의미한다. 객체지향시스템에서 객체 사이의 상호작용은 모두 메시지 전달을 통해 이루어진다. 메시지란 상대 객체가 제공하는 연산에 대한 수행요청으로 함수 호출(Function Call) 형태로 전달된다. 캡슐화된 상대 객체에게 서비스를 요청할 수 있는 유일한 방법은 메시지 전달 매커니즘 밖에는 없다.

객체지향의 기본사상은 정보가 객체를 중심으로 모여 지역화되는 것이라 볼 수 있는데 객체지향접근방법에는 다음과 같은 장점이 있다 (Rumbaugh et al., 1991), (Jacobson et al., 1992), (Martin and Odell, 1992), (Booch, 1991), (Wirfs-Brock et al., 1990), (Coad and Yourdon, 1991), (Taylor, 1992), (Berard, 1992), (Wilkie, 1992).

(1) 객체지향 모델은 분석하려는 문제와 매우 유사하다.
(2) 객체지향 접근방법은 분석된 개념에 대한 추적이 매우 용이하다.
(3) 개념적 무결성이 유지된다.
(4) 객체는 느슨하게 연결되며(Loosely Coupled) 매우 응집되어(Cohesive) 있으므로 동시에 개발될 수 있다.
(5) 구축된 모델의 재사용이 용이하다.

4-2. 객체지향 모델의 구축 절차

객체는 본질적으로 정적, 동적인 특성을 가지기 때문에 도출과정이 결코 단순하지 않다. 따라서, 객체의 도출을 위해서는 다양한 각도에서의 업무 분석이 수행되어야 한다. 객체의 도출을 위한 분석은 객체다이어그램(OD : Object Diagram), 인터액션다이어그램(ID : Interaction Diagram), 객체상태다이어그램(OSD : Object State Diagram)으로 구성된다.(그림 8)

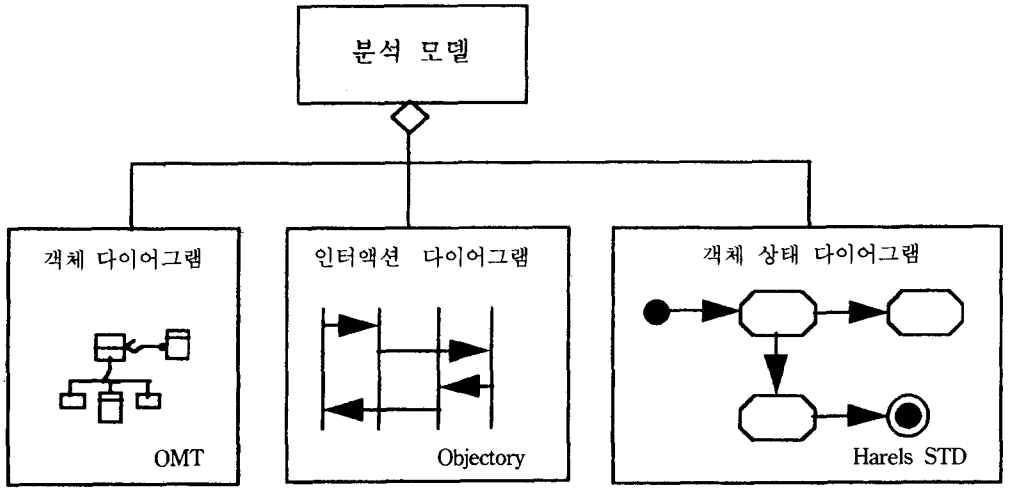

그림 8. 분석모델

객체다이어그램은 객체의 정적인 측면의 모델로 객체의 속성과 객체 사이의 관계 등을 정의하는데 객체를 공통적 특성 (Rumbaugh et al., 1991)－기능, 위치, 서비스 등이나 주제(Subjects) (Coad and Yourdon, 1991)에 따라 계층적 구조로 분해한다. 객체다이어그램은 OMT(Rumbaugh et al., 1991)에서 제공하는 개념과 표기법을 사용하여 구축될 수 있다. 인터액션 다이어그램은 객체간의 서비스 요청(연산호출)을 나타낸 것으로 객체의 연산도출을 위한 분석 정보를 제공한다. 인터액션다이어그램 상에 나타나는 연산은 각각 해당객체에게 할당된다. 인터액션다이어그램은 Objectory(Jacobson et al., 1992)에서 개발된 분석기법이다. 객체상태다이어그램은 내부, 외부에서 사건이 전달될 때 객체의 데이타 속성의 값이 어떻게 변화되고 사건이 어떤 순서로 전달되는지 나타내는데 연산을 메소드로 구현할 때 유용한 정보를 제공한다. 객체상태다이어그램은 Harel이 개발하였고 OMT에서 사용하고 있는 STD (State Transition Diagram) (Harel, 1987)를 사용할 수 있다.

객체지향모델은 단적으로 다수의 객체로 구성된 것이다. 따라서, 객체지향모델의 도출은 개별 객체의 속성과 연산을 정의하고 이에 대한 구현을 계획하는 작업을 반복하는 것이라 할 수 있다. 우선 도메인, 의사결정모델을 구축하는 단계는 다음과 같이 수행된다.

Step 1. 업무를 기술한다.

Step 2. 객체의 대상을 발견한다.

Step 3. 객체의 데이타 속성을 정의한다.

Step 4. 객체 사이의 일반관계, 구성관계, 유전관계를 발견한다.

Step 5. 사용케이스(시나리오)를 작성한다.

Step 6. 객체간의 상호작용을 인터액션다이어그램으로 작성한다.

Step 7. 인터액션다이어그램에서 발견된 연산들을 객체에게 할당한다.

Step 8. 완성된 객체를 질의(Query)로 테스트하고 세련시킨다.

Step 9. 도출된 객체를 콘트롤, 인터페이스, 실체 객체로 분류한다.

Step 10. 객체상태다이어그램을 작성하고 연산에 대한 구현을 계획한다.

사용자인터페이스, 통합모델은 유사한 과정을 거쳐 구축되나 객체상태에 대한 실시간 제어가 요구되므로 객체상태다이어그램의 비중이 크다. 지식모델은 지식의 표현 방법중 규칙과 같이 비객체적 부분을 객체로 표현하는 방법이 아직 정형화되지 않았으므로 가장 객체지향접근방법으로 구축하기 어려운 모델이다. 따라서 다음 장에서 이에 대한 접근방법을 모색하고자 한다.

4-3. 객체지향지식모델

도메인모델과 의사결정모델이 선언적 지식을 표현한다면 지식모델은 선언적 모델을 사용하여 문제를 해결하는 절차적 지식을 표현한 것이라 볼 수 있다 (Rich and Knight, 1991). 절차적 지식을 나타내기 위한 가장 보편적인 방법은 if-then 규칙이다. 예를 들어, 진단시스템은 규칙들이 전방 또는 후방으로 매치되어 긴 연쇄고리를 이루고 이 과정에서 고장의 원인과 이에 대한 대책을 제시하게 된다. 반면에, 설계나 스케쥴링시스템의 경우, 규칙으로 표현되지 않는 알고리즘이나 수학적 모델이 최적 해결안을 제시하게 된다. 따라서, 선언적 지식이 일반적으로 객체로 표현되고 있는 반면에 절차적 지식은 규칙, 알고리즘 등을 혼합한 형태로 표현되고 있으며 통합적 표현구조가 정형화되지 못한 실정이다. 한편, 최근 절차적 지식을 객체로 표현하려는 연구가 시도되고 있다. Rubin et al(1994)은 규칙을 객체의 행동적 측면에서 모델링하는 방법을 제시하였는데, 규칙의 조건부의 조건들과 결론부의 액션들에 대한 책임을 담당할 객체를 각각 찾고 이들의 연산으로 조건과 액션을 표현하고 있다. 또한 사례기반추론의 경우, 사례를 객체로 표현하고 유사도 계산에 따른 유사사례발견 알고리즘은 사례객체의 연산으로 정의하여 객체를 지식표현구조로 사용하고 있음을 알 수 있다. 따라서, 규칙이 객체로 표현될 수 있다면 객체는 모든 유형의 지식을 표현할 수 있는 통합적 지식표현구조가 될 것이다. 그러나, 규칙과 같은 절차적 지식을 객체로 반드시 표현해야 한다는 당위성은 아직 제시되지 않고 있다. 다만, 객체가 데이타와 연산을 결합한 것처럼 규칙도 함께 결합할 수 있을 때, 기존의 객체가 가지고 있는 장점을 확장할 가능성을 기대해 볼 수는 있을 것이다. 규칙을 객체구조로 표현할 수 있는 방법으로 몇가지 방법이 있을 것이다. 첫째, 규칙을 객체속에 결합하는 방법이다. 즉, 규칙과 같이 독자적인 표현구조를 사용하는 것이 아니라, Rubin et al.(1994)의 접근방법과 같이 규칙의 조건부와 결론부를 각각 해당 객체에 연산으로 표현한다. 둘째, 규칙을 규칙객체와 같은 새로운 형태의 객체로 표현하는 것이다. 이 방법은 규칙을 표현방법에만 차이가 있을 뿐 독자적인 지식으로 표현하게 된다. 규칙객체에 대한 클래스는 데이타 속성으로 조건부와 결론부를 정의할 수 있으며 클래스 연산으로 규칙객체를 전방 또는 후방으로 활성시키는 매칭알고리즘을 정의할 수 있다. 활성화연산에 대한 구현은 RETE (Forgy, 1982)와 같은 기존의 매칭 알고리즘을 사용할 수 있을 것이다.

V. 적용사례 분석

동시공학적 전문가시스템 개발방법론은 1992년에 처음으로 스케쥴링 전문가시스템 개발에 적용된 이후 지속적으로 전문가시스템 개발 프로젝트에 적용되고 있다. 아직 완벽히 적용된 사례가 없고 예상한 기대효과가 모두 검증되지는 않았지만 시간과 자원제약조건하의 프로젝트의 경우, 개발기간의 효과적 활용과 지식의 효과적 획득면에서 소기의 목적을 달성한 것으로 평가받고 있다. 본 논문에서는 제일모직의 스케쥴링전문가시스템인 ProSES(Production Scheduling Expert System)의 개발에 있어 동시공학적 개발방법론의 적용사례를 소개하고자 한다.

5-1. 프로젝트 개요

제일모직 화성사업부는 고객에게 납기정보를 주문시점에 제공하기 위해 여천공장의 스케쥴링 기능을 서울본사의 영업과 물류팀에서 수행할 수 있도록 스케쥴링전문가시스템의 개발을 계획하게 되었다. 이 프로젝트는 ProSES(Production Secheuduling Expert System)라는 이름으로 1992년 6월에 전체 10개월의 비교적 단기간내에 개발을 완료해야 하는 시간적 제약조건을 가지고 착수되었다. ProSES는 다품종 소량생산체제와 JIT(Just In Time) 자재관리를 지향하는 고객에게 정확한 납기일을 주문시점에 제공해줌으로써 고객과의 협상력에서 경쟁우위를 노리는 프로세스 재설계의 구현단계에서 계획된 프로젝트였으며, 따라서, 후발업체인 제일모직으로서는 가능한 한 빠른 시점에 개발을 완료해야하는 상황에 있었다. 개발 하드웨어는 VAX VLC400이 선정되었으며 전문가시스템 개발도구로는 ART-IM을, 사용자인터페이스는 MOTIF를 각각 사용하였다. 기존의 생산정보데이타는 ADABAS 데이타베이스에 저장되었으며 이에 대한 수주관리시스템, 생산관리시스템 등 관련 정보시스템은 Natural 4GL로 개발되어 있었다.

5-2. 프로젝트팀 구성

효과적인 프로젝트 진행관리를 위해 개발팀은 분석팀, 설계팀, 구현팀으로 세분화되어 구성되었다. 분석팀은 분석단계에서 수행돼야 할 조직적상황, 의사결정과정, 지식모델, 사용자인터페이스에 대한 분석을 담당하였으며 인원 구성상, 의사결정모델과 지식모델에 대한 설계와 구현도 실질적으로 분석팀에서 수행하였다. 설계팀은 사용자인터페이스에 대한 설계와 기존의 정보시스템과의 데이타 송수신을 위한 데이타베이스 및 네트웍 인터페이스의 설계와 그리고 이에 대한 구현을 수행하였다. 한편, 구현팀은 사용자 인터페이스에 대한 구현 작업을 전담하고 프로젝트 중반부터 지식모델의 구현과 테스트 작업을 지원하였다.

프로젝트팀 구성상의 난점은 제한된 투입인원으로 다수의 작업을 동시에 수행해야 한다는데 있었다. 결국 동시공학적 방법론을 현실적으로 변형하여 동일한 개발자가 다수의 작업을 동시에 수행하는 가상적인 동시성(Virtual Concurrence)이 발생하였다. 예를 들어 분석팀은 초기에는 분석에만 전담하였으나 프로젝트 중반이후에는 특히 지식모델에 대한 분석, 설계, 구현을 동일한 개발인원들이 동시에 수행하게 되었다. 한편, 테스트팀은 초기에는 구성하였으나 인터페이스 부분이 예상외로 난이하고 또, 작업범위도 확장된 이유로 설계팀과 구현팀에 재배치하였다.

이상에서 살펴본 바와 같이 ProSES의 개발은 투입인원의 제한으로 개발팀이 개발단계별로 역할이 분담된 형태로 구성되지 않고 결과적으로 개발팀이 모델별로 구성된 형태가 되고 말았다. 그러나, 충분한 개발인력이 확보되어 단계별로 팀이 구성되어야만 동시공학적 개발이 실현될 수 있음을 지적하며, 다만 분석팀의 예와 같이 선행되어 작업이 완료되는 팀은 구현이나 테스트 등에 일부 참여할 수 있다는 가상적 동시성 적용의 가능성을 현실적으로 받아 들일 수 있다고 본다.

5-3. 동시개발과정

ProSES는 그림 9와 같이 전체적으로는 분석, 개발, 구현이 동시에 진행되는 형태로 개발이 진행되었다. 그러나, 지식모델은 개발팀의 인원 구성상 개발 중반까지 동시에 진행되지 않고 부분적으로는 순차적으로 분석, 설계, 구현되었다. 초기 분석단계에서 스케쥴링 업무의 조직적상황은 모든 개발팀이 참여하여 수행되었는데, 기존의 수주관리시스템, 생산정보시스템에 대한 분석이 실시되어 기존의 정보시스템은 수정없이 사용하고 다만 이들 기존 시스템과의 인터페이스만

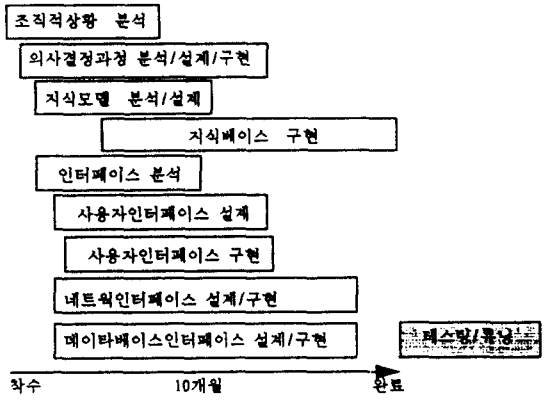

그림 9. ProSES의 동시개발과정

개발하면 된다는 결론을 얻었다. 그러나, 이 단계에서 서울 본사의 IBM4381의 ADABAS 데이타베이스에 개발된 수주관리시스템과 여천 공정의 VAX6210에 개발된 생산정보시스템, 그리고 ProSES가 개발될 VAXstation VLC400과의 사이에 네트웍 인터페이스가 한글 데이타 처리관계로 당초 예상보다 난이하게 분석됨에 따라 인터페이스 설계팀에 테스트팀과 구현팀의 일부를 재배치하였다. 조직적 상황에 대한 분석이 진행되는 동안 분석팀은 의사결정과정과 사용자인터페이스에 대한 분석도 병행해서 수행하였으며 특히 설계팀은 이 시점부터 데이타베이스와 네트웍인터페이스에 대한 설계작업의 일환으로 다양한 인터페이스기술을 시험하기 시작했다. 의사결정과정에 대한 분석으로 스케쥴링시스템의 기능과 스케쥴링 모델이 윤곽을 들어내자 사용자인터페이스에 대한 분석, 설계, 구현 작업이 거의 동시에 수행될 수 있었으며 ProSES의 사용자인터페이스는 개발착수 4개월후에 완전히 구현될 수 있었다.

사용자인터페이스의 구현이 진행되면서 데이타베이스와 네트웍인터페이스에 대한 설계와 구현 작업이 본격적으로 수행되어 본사 수주정보시스템과 생산정보시스템, 그리고 ProSES 간의 인터페이스가 구현될 수 있었다. 예를 들어, 수주데이타가 ProSES에 정확히 전송되고 또한 ProSES의 스케쥴링 결과가 수주관리시스템에 정확히 전송되는지에 대해 테스트 작업이 이루어질 수 있었다.

한편, 의사결정모델의 설계가 완성되어 납기를 목적함수로하고 생산성을 제약조건으로 하는 납기위주의 의사결정모델과 생산성을 목적함수로 하고 납기를 제약조건으로 하는 생산성위주의 의사결정모델이 도출되었다. 이에 프로토콜분석(Byrd et al., 1992) (Kim and Courtney, 1988)에 의한 전문지식획득작업이 본격화되었으며 특히 사용자인터페이스 개발이 완료됨에 따라 전문가의 시스템에 대한 이해가 효과적으로 향상되면서 지식획득이 용이하게 되었다. 지식모델이 개발 초기에는 순차적으로 진행되었음을 이미 지적한바 있지만 사실, 일부 선언적 지식, 예를 들어 라인특성, 제품 특성, 색상과 물성의 그룹 정보등은 절차적 알고리즘이나 휴리스틱을 분석, 설계하는 과정에서도 ART-IM의 스키마(Schema : 객체구조임)로 구현되어 이에 대한 조회나 규칙에 대한 테스트가 가능하였다. 구현팀은 사용자인터페이스에 대한 개발이 완료됨에 따라 지식모델 개발에 참여하였으며 이에 따라 개발착수후 5개월후에는 지식모델에 대한 분석, 설계, 구현이 어느정도 동시에 작업이 수행될 수 있었다. ProSES는 개발 착수후 10개월후 개발완료되었으나, 추가로 5개월간의 테스트와 튜닝작업을 거쳐 서울 물류팀에서 정상 운영되고 있다.

5-4. 객체지향 패러다임의 적용

ProSES의 개발계획을 작성할 시점에는 객체지향 분석/설계/구현을 계획하였으나 결론적으로 본문에 소개된 객체의 도출절차를 공식적으로 적용할 수 없었다. 다만 지식베이스의 선언적 지식인 스키마와 GUI의 그래픽 객체, 네트웍인터페이스에서 사용된 Socket 등 소프트웨어 구조가 객체개념에 근간을 둔 관계로 부분적인 측면에서 볼 때 객체지향 패러다임이

적용되었다고 볼 수 있겠다. 객체지향 패러다임의 적용 당위성에도 불구하고 실행되지 못하는 이유는 기존의 정보시스템이 구조적방법론과 관계형데이타베이스 등을 기반으로 하기 때문이며 따라서, 일부 기존의 정보시스템과 향후 개발될 정보시스템은 ProSES와 같이 객체구조와 비객체구조의 양면성을 유지할 수 밖에 없을 것이나 결국에는 객체구조로 통합될 것으로 예상된다.

5-5. 동시공학적 방법론의 적용을 통한 교훈

ProSES는 개발착수후 10개월후에 개발이 완료되고 추가로 5개월의 테스트와 튜닝작업을 거쳐 서울 물류팀에서 정상 가동되었으나 개발과정에서 드러난 문제점과 이에 대한 분석결과는 시사하는 바가 크다. 첫번째 문제점은 개발기간과 비용차원에서의 계획이 크게 잘못되었다는 것이다. 10개월은 시스템 개발을 위한 최소의 기간이었으며 투입인력도 개발범위와 난이도를 감안할 때 크게 부족하였으며 결국 5개월의 기간이 추가로 소요된 후에야 시스템이 적상적으로 가동될 수 있었다. 둘째, 시스템 인터페이스에 대한 기술적 분석의 미비로 인터페이스작업이 크게 늘어나 프로젝트팀 조직과 운영에 차질이 발생하였다. 세째, 테스트 사례에 대한 사전준비 부족으로 스케쥴링 기능에 대한 평가가 지연되었다. 개발이 완료되어 사용자인 서울 물류팀에 설치된 것은 스케쥴링 결과에 대한 부분적인 기능테스트를 마친 후였고 따라서, 스케쥴링에 대한 평가가 완전히 이루어진 상태가 아니었다. 네째, 개발 초기, 시스템 사용부서의 방관적 자세로 스케쥴링 의사결정과정과 사용자인터페이스 부분이 개발도중 변경되기도 했으며 따라서, 개발진행관리에 많은 문제를 안겨 주었다.

이상의 문제점을 고려할 때, 정상적인 개발방법으로는 개발을 계획대로 완료할 수 없을 것이라는 결론에 도달하게 된다. ProSES는 그러나, 이런 문제점에도 불구하고 현재 성공적으로 개발, 가동되고 있는데, 이는 동시공학적 개발방법론이 다음과 같이 기여했기 때문이다. 첫째, 분석, 설계, 구현의 동시작업으로 많은 위험요인이 조기에 발견, 조치될 수 있었는데 예를 들어, 인터페이스 작업의 난이성의 조기파악은 기술적 대체안에 대한 신속한 평가로 해결책을 효과적으로 마련하는 계기가 되었다. 둘째, 시스템의 조기 구현으로 개발진행이 가시화됨에 따라 개발과정을 단축하는 결과를 낳았는데 예를 들어, 사용자인터페이스를 포함한 인터페이스의 조기개발은 사용자의 시스템에 대한 이해를 향상시켜 지식모델의 설계와 구현과정을 효과적으로 단축시켰다.

한편, ProSES에 대한 동시공학적 개발방법론의 적용으로 얻은 교훈은 방법론의 개선에 많은 도움이 되었다. 첫째, 개발인력의 부족이나 예기치 않은 개발범위의 확대는 동시성 유지의 변경을 예고하나, 가상적 동시성 유지로 이를 해결할 수도 있다는 것을 확신했다. 둘째, 개발초기에 개발인력의 불충분한 투입으로 인한 동시성의 위반은 결국 장기적인 비용의 증가로 귀결된다는 결론을 얻었다. ProSES의 경우, 테스트팀의 정상적인 동시작업이 있었더라면 스케쥴링 결과에 대한 평가가 조기에 가능했을 것이며 개발중반의 지식모델의 설계나 구현에 있어 순차적인 작업이 동시에 수행되었더라면 지식베이스가 보다 신속하게 개발될 수 있었을 것이다. 세째, 다수의 개발팀이 동시에 작업하는 과정에서 많은 의사소통 문제가 발생하게 되었는데, 이는 구조적 분석, 설계방법의 단계별 개념의 불일치에 기인한 것으로 근본적으로 객체지향방법론의 적용의 필요성을 인식하게 된 계기가 되었다.

Ⅵ. 결론과 향후 연구

전문가시스템이 기업의 정보시스템에서 차지하는 비중이 증가하고 있는 상황에서 이제 전문가시스템 개발에 엄격한 공학적 원칙이 적용돼야 한다. 지식을 획득하고 이를 표현해야 한다는 당면과제는 전문가시스템을 아직도 위험이 내포된 시스템으로 인식하게 한다. 지금까지 전문가시스템의 개발에 관련된 연구는 주로 지식획득 기술에 관련된 것이다. 본 논문은 전문가시스템 개발 프로젝트에 있어 방법론적인 문제를 다루고 있으며 전문가시스템 개발방법론의 실현 뿐만 아니라 일반적인 정보시스템의 개발방법론의 발전에도 기여할 것이다. 본 논문이 제시한 동시공학적 전문가시스템 개발방법론은 다음과 같이 요약할 수 있다.

(1) 분석, 설계, 구현, 테스트 단계가 동시에 진행된다.
(2) 단계별 세부작업도 동시에 진행된다.
(3) 프로젝트팀 구성에 있어 전담팀 개념을 도입하였다.
(4) 개발단계별 모델을 정의하고 모델을 중심으로 프로젝트진행을 관리한다.
(5) 객체지향패러다임을 통합적 모델구축방법으로 사용하고 있다.

본 논문에서 제시된 접근방법이 실용적인 전문가시스템 개발방법론으로 구현되기 위해서 향후 다음 주제에 대한 연구가 요구된다.

(1) 동시성 수준을 나타내는 준거곡선(Reference Curve)에 대한 연구
   a. 동시작업의 발생시점에 대한 연구
      도메인모델과 의사결정모델의 구축에 있어서 전담팀별 역할 분담
   b. 동시작업과정중 동시성을 최적 수준으로 유지하기 위한 방법
      - 작업결과의 상호전달과 피드백
      - 문제와 갈등 발생시 조정방법
   c. 동시작업의 완료시점에 대한 연구
      - 단계별 작업이 동시에 완료되는 경우
      - 단계별 작업이 점진적으로 완료되는 경우
(2) 프로젝트팀 구성원의 역할 분담
(3) 객체를 통합적 지식표현구조 사용하기 위한 세부적인 구현방법
(4) 품질관리, 모델의 재사용, 문서화 기술, 프로토타이핑 등 방법론적 문제에 대한 추가적 연구

동시공학적 전문가시스템 개발방법론은 전문가시스템의 고유특성과 정보시스템적 특성을 모두 고려한 접근방법이라 할 수 있다. 개발방법론의 타당성 여부는 실사례 적용을 통한 실증적 분석으로 결정됨을 인정할 때, 동시공학적 접근방법을 적용한 성공적 전문가시스템 개발 사례를 기대해본다.

참고문헌

1) Abdul-Gader, A. H. and Kozar, K. A., "Discourse Analysis for Knowledge Acquisition : The Coherence Method", *Journal of Management Information Systems*, Vol. 6, No. 4, 1990, pp.61~82.
2) Agarwal, R. and Tanniru, M. R., "Knowledge Acquisition Using Structured Interviewing : An Empirical Investigation", *Journal of Management Information Systems*, Vol. 7, No. 1, 1990, pp.123~140.
3) Belz, F. C., "Applying the Spiral Model : Observations on Developing System Software in Ada", *Proceedings of the 1986 Annual Conference on Ada Technology*, Atlanta, GA, 1986.
4) Benington, H. D., "Production of Large Computer Programs", *Proceedings of the ONR Symposium on Advanced Program Methods for Digital Computers*, 1956.
5) Berard, E. V., *Essays on Object-Oriented Software*

Engineering, Prentice-Hall, Englewood Cliffs, NJ, 1993.

6) Berry, D. C., "The Problem of Implicit Knowledge", *Expert Systems*, Vol. 4, No. 3, 1987, pp.144~150.

7) Birrel, N. D. and Ould, M. A., *A Practical Handbook for Software Development*, Cambridge University Press, Cambridge, UK, 1985.

8) Boehm, B. W., Gray, T. L., and Seewaldt, T., "Prototyping Versus Specifying : A Multiproject Experiment", *IEEE Transactions on Software Engineering*, Vol. SE-10, No. 3, 1984, pp.290~302.

9) Boehm, B. W., "A Spiral Model of Development and Enhancement", *Software Engineering Notes*, Vol. 11, No. 4, 1986.

10) Boehm, B. W. and Belz, F. C., "Applying Programming to the Spiral Model", *Proceedings of the 4th International Software Process Workshop*, 1988.

11) Booch, G., *Object Oriented Design with Applications*, Benjamin/Cummings, Redwood City, CA, 1991.

12) Buchanan, B. G., Barstow, D., Bechtel, R., Bennet, J., Clancey, W., Kulikowski, C., Michell, T., and Waterman, D. A., "Constructing an Expert System", in F. Hayes-Roth, D. A. Waterman, D. B. Lenat, eds., *Building Expert Systems*, Addison-Wesley, Reading, MA, 1983, pp.127~167.

13) Brooks, F. P., *The Mythical Man-Month*, Addison-Wesley, Reading, MA, 1975.

14) Byrd, T. A., Cossick, K. L., and Zmud, R W., "A Synthesis of Research on Requirements Analysis and Knowledge Acquisition Techniques", *MIS Quarterly*, Vol. 16, No. 1, 1992, pp.117~138.

15) Coad, P. and Yourdon, E., *Object-Oriented Analysis*, 2nd Ed. Yourdon Press, Englewood Cliffs, NJ, 1991.

16) Dhar, V., "On the Scope and Plausibility of Expert Systems in Management", *Journal of Management Information Systems*, Vol. 4, No. 1, 1987, pp.25~41.

17) Dzieranowski, J., Chrisman, K., MacKinnon, G., and Klahr, P., "The Authorizer's Assistant-A Knowledge-Based Credit-Authorization System for American Express", *Proceedings of 1989 Innovative Applications of Artificial Intelligence Conference*, Palo Alto, CA, 1989, pp.28~30.

18) Forgy, C. L., "Rete : A Fast Algorithm for the Many Pattern/Many Object Pattern Match Problem", *Artificial Intelligence*, Vol. 19, No. 1, 1982, pp.17~37.

19) Harel, D., "Statecharts : A Visual Formalism for Complex Systems", *Science of Computer Programming*, Vol. 8, 1987.

20) Jacobson, I., Christerson, M., Jonsson, P., and Overgaard, G., *Object-Oriented Software Engineering*, Addison-Wesley, Reading, MA, 1992.

21) Kim, J. and Courtney, J. F., "A Survey of Knowledge Acquisition Techniques and Their Relevance to Managerial Problem Domain", *Decision Support Systems*, Vol. 4, No. 3, 1988, pp.269~284.

22) Lee, J. K., "Integration and Competition of AI with Quantitative Methods for Decision Support", *Expert Systems with Applications*, Vol. 1, No. 4, 1990, pp. 329~333.

23) Martin, J., *Information Engineering Book I*, Prentice-Hall, Englewood Cliffs, NJ, 1989.

24) Martin, J. and Odell, J., *Object-oriented Analysis and Design*, Prentice Hall, Englewood Cliffs, NJ, 1992.

25) McGovern, J. and Samson, D. and Wirth, A., "Knowledge Acquisition for Intelligent Decision Systems", *Decision Support Systems*, Vol. 7, 1991, pp.263~272.

26) Nevins , J. L. and Whitney, D. E., *Current Design of Products and Processes*, McGraw-Hill, New-York, NY, 1989.

27) Rich, E. and Knight, K., *Artificial Intelligence*, 2nd

ed., McGraw-Hill, New York, NY, 1991.

28) Rubin, K. S., McClaughry, P., and Pellegrini, D., "Modeling Rules Using Object Behavior Analysis and Design", *Object Magazine*, Vol. 4, No. 3, 1994, pp.63~67.

29) Rumbaugh, J., Blaha, M., Premerlani, W., Eddy, F., and Lorensen, W., *Object-Oriented Modeling and Design*, Prentice-Hall, Englewood Cliffs, NJ, 1991.

30) Scot, A. C., Clayton, J. E., and Gibson, E. L., *A Practical Guide to Knowledge Acquisition*, Addison-Wesley, Reading, MA, 1991.

31) Shortliffe, E. H., *Computer-Based Medical Consultations : MYCIN*, Elsevier, New York, 1976.

32) Simon, H. A., "What Have Computers to Do with Management ? ", in *Management Organization and the Computer*, Schultz, G. and Whisler, T., eds., Free Press, Glencoe, 1960, pp.39~60.

33) Simon, H. A., "Introduction to Empirical Research on Organizational Decision Making", Witte, E. and Zimmerman, H. J. eds., North-Holland Elsevier, Amsterdam, 1986.

34) Taylor, D. A., *Object-oriented Information Systems*, Wiley, New York, NY, 1992.

35) Waterman, D. A., *A Guide to Expert Systems*, Addison-Wesley, Reading, MA, 1985.

36) Weitzel, J. R. and Kerschberg, L., "Developing Knowledge-Based Systems : Reorganizing the System Development Life Cycle", *Communications of the ACM*, Vol. 32, No. 4, 1989, 482~488.

30) Wilkie, G., *Object-Oriented Software Engineering*, Addison-Wesley, Reading, MA, 1993.

31) Wirfs-Brock, R., Wilkerson, B., and Wiener, L., *Designing Object-Oriented Software*, Prentice Hall, Englewood Cliffs, NJ, 1990.

32) Wolf, M., Wenger, D., and Kirchmayr, K., "CUBUS－An Assistant for Fundamental Corporate Analysis", *Innovative Applications of Artificial Intelligence*, Vol. 3, R. Smith and C. Scott, eds, 1991, pp. 217~288.

33) Wright, G. and Ayton, P., "Eliciting and Modeling Expert Knowledge", *Decision Support Systems*, Vol. 3, No. 4, 1987, pp.13~26.