'98 International Conference Digital Library & Knowledge
September 22, 1998, Seoul Korea

시스템 개발 프로세스 관리 능력의 향상을 위한 방안: 지식관리적 접근방법

김성근, 이진실 (중앙대)
원은희 (LG-EDS)

요 약

정보시스템 개발노력의 상당수는 실패로 끝나고 있다. 최근 통계에 따르면 정보시스템 개발 프로젝트의 반은 실패로 끝난다고 한다[Kaplan, 1998]. 이와 같은 높은 실패율은 시스템 개발을 위한 노력을 체계적으로 투입하지 못하고 개발 프로젝트를 관리하기 위한 노력을 단위 프로젝트 차원에서만 집중시키는데서 연유한다고 생각된다. 다시 말해 장기적인 관점에서 개발조직의 역량 향상이라는 보다 근본적인 목표를 간과하고 있는 것이다. 이러한 점에 착안하여 소프트웨어 엔지니어링 분야에서는 정보시스템 개발과 관련한 개발 조직의 능력을 향상시키기 위한 다양한 접근방법이 제시되고 있다. 개발조직의 개발 프로세스 성숙도를 진단하기 위한 측정도구로 개발된 카네기멜론대학의 CMM(Capability Maturity Model)과 ISO에서 정의한 표준인 SPICE (Software Process Improvement and Capability dEtermination) 모델이 그 대표적인 예에 속한다. 그러나 이와 같은 모델들은 개발조직의 프로세스 개선을 위한 방향과 요건은 제시하고 있지만, 이를 조직 내에서 구현하기 위한 구체적인 방법이나 수단은 제시해주지 못하고 있다. 따라서 이러한 접근방법 역시 소프트웨어 엔지니어링 역량이나 개발경험이 일천한 우리 현실에서는 부분적인 성과 이상을 기대하기는 어려웠다. 본 연구에서는 이와 같은 문제점이 개발 프로젝트와 관련된 경험이나 지식을 효과적으로 추출하고, 획득하고, 체계화하고, 시스템화하여 조직 내에서 활용하려는 노력이 부족했기 때문이라고 본다. 이에 본 연구에서는 개발조직의 역량 향상을 위한 지식관리적 접근 방법의 세 가지 유형을 제시하기로 한다.

1. 서론

정보시스템 개발노력의 상당수는 실패로 끝난다. 프로젝트가 실패하게되는 경우는 다양하다. Laudon & Laudon[1998]은 시스템실패의 원인으로 네 가지 영역을 지적한다. 설계, 자료, 비용, 운용이 바로 그에 해당한다. 이와 같은 문제점은 단지 정보시스템의 기술적 차원에서만 발생하는 것이 아니라 기술외적 문제로도 기인한다.

이와 같이 정보시스템 개발프로젝트에 관련된 불확실성과 위험을 줄이기 위해 개발 프로젝트를 세심하게 관리하고 조정할 필요가 있다. 그리하여 사전에 정의해놓은 일정한 개발계획을 기초로 진행상황을 평가분석하고, 필요에 따라서는 투입자원의 조정 등과 같은 조치를 취하기도 한다. 이와 같은 프로젝트 관리는 대개 시간, 예산, 품질이라는 세 가지 차원에서 이루어진다. 그러나 이 모두는 서로 상충관계에 놓여있기 때문에 어느 한 목표를 달성하기 위해서는 다른 두 가지를 희생해야하므로 세 가지 모두를 달성하려는 시도는 가능성 없는 욕망에 불과하다는 것이 일반적인 시각이었다[Redmill, 1997]. 특히 시간이 촉박한 경우에는 프로젝트 관리자와 팀원들이 최선의 노력을 경주한다고 하더라도 프로젝트가 성공적으로 수

행되지 못하는 경우가 많다[Ward, 1995].

그러나 문제의 더 근본적 원인은 정보시스템 개발을 이처럼 단위 프로젝트 차원에서 계획하고 수행하려고 시도해왔기 때문이라고 생각된다. 프로젝트관리의 속성상 프로젝트관리의 목표가 쉽게 달성될 수 없을 뿐만 아니라, 이전의 프로젝트를 수행하는 과정에서 축적된 경험과 지식이 활용되지 못해 프로젝트관리 능력의 향상을 불러일으키기가 어렵게 되어 있다. 네덜란드의 연구에 따르면 대략 50% 정도의 소프트웨어 엔지니어링 조직이 자신들의 엔지니어링 프로세스에 대해 어떠한 자료도 수집하지 않고 있다고 한다[Michiel, 1993].

프로젝트의 성공적 수행을 촉진하고 프로젝트관리 능력의 향상을 위해서는 개발조직 전체의 공동노력이 필요하다. 즉, 보다 장기적이고 전사적인 관점에서 개발 조직의 역량을 향상시킬 수 있는 프로세스 개선 방안을 마련하고 이를 조직내에 구현해야만 개발 프로젝트의 실패율을 낮추고 프로젝트 관리의 세 가지 목표간에 균형을 유지할 수 있을 것이다. 이러한 요구에 따라 현재 정보시스템 개발 프로세스와 개발 조직의 능력을 향상시키기 위한 다양한 접근방법이 제시되고 있다. 대표적인 예로는 표준적인 개발 프로세스를 제시하고 있는 ISO/IEC 12207과 DoD의 MIL-STD-498, 그리고 프로세스 개선 방향 및 평가 방법을 제시하고 있는 카네기멜론대학의 CMM과 ISO의 SPICE(ISO 15504) 등을 들 수 있다[김성근 외, 1998].

그러나 이들 접근방법의 대부분은 개발조직의 프로세스 개선을 위한 방향과 요건은 제시하고 있지만, 이를 조직내에서 구현하기 위한 구체적인 방법이나 수단은 제시해주지 못하고 있다. 따라서 이러한 접근방법 역시 소프트웨어 엔지니어링 역량이나 개발경험이 일천한 우리 현실에서는 부분적인 성과 이상을 기대하기는 어려웠다.

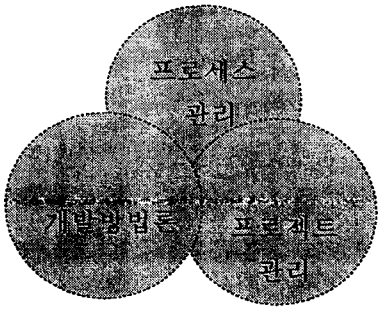
이에 본 연구에서 개발 프로젝트와 관련된 경험이나 노하우를 체계적으로 축적하고 활용하려는 노력이 부족했음에 기인했다고 보고, 이와 같은 지식을 효과적으로 추출하고, 획득하고, 체계화하고, 시스템화하여 조직내에서 활용하려는 노력을 기울일 필요가 있다고 본다. 이에 본 연구에서는 개발조직의 역량 향상을 위한 지식관리적 접근 방법의 세 가지 유형을 제시하고자 한다.

2절에서는 프로젝트 관리능력의 향상을 위한 통합적 모형을 제시하고, 특히 프로세스관리를 위해 전개되어온 노력을 설명한다. 3절에서는 개발방법론 및 프로젝트관리 도구의 관점에서 추진될 필요가 있는 노력을 제시하고, 4절에서는 이와 같은 개발프로젝트 수준 향상을 보다 효과적으로 구현하기 위한 지식관리적 접근방법을 제시한다. 마지막으로 요약 및 향후 연구방향이 5절에 묘사되어 있다.

2. 프로세스 관리의 향상을 위한 노력

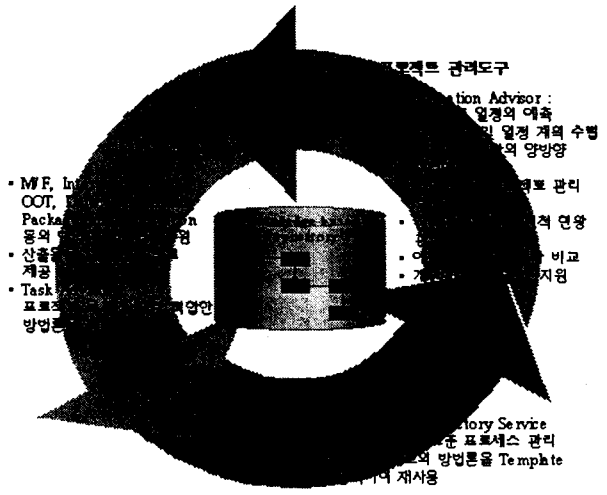
정보시스템 개발프로젝트에 관련된 불확실성과 위험을 줄이고 개발될 정보시스템의 품질을 높이기 위해서는 다양한 노력이 통합적으로 수행될 필요가 있다. 정보시스템을 효과적으로 개발하기 위한 개발방법론, 이 개발 프로젝트의 관리, 그리고 개발에 관련된 프로세스의 관

리 노력이 통합적으로 전개될 필요가 있다.(그림 2 참조)



[그림 1] 프로젝트 수준 향상을 위한 통합 프레임워크

방법론을 도입하여 개발 조직이 이를 제대로 활용하기 위해서는 상당한 노력과 시간이 필요하다. 왜냐하면 방법론은 각각 그 자체가 개발조직의 특성과 기술수준, 그리고 생산성을 충분하게 반영하고 있어야 하며 프로젝트의 적용사례를 통해 부단하게 보완해야 하기 때문이다. 따라서 어떤 의미에서 특정 개발조직에 가장 적합한 방법론은 개발조직 스스로가 경험을 통해 만들어낸 것이라 할 수 있다. 그렇지만 이러한 방식은 필연적으로 많은 개발경험과 시간을 필요로 하므로 정보시스템 개발경험이 적거나 노하우가 축적되지 않은 조직에서는 수용하기 힘든 방식이다. 따라서 우리 나라의 경우 해외의 상용 방법론을 도입하여 활용하는 경우가 많은데, 이러한 방식은 개발 조직이 시스템 개발 프로세스를 정의하기 위해 투입해야 하는 시간과 노력을 단축시켜 준다는 장점이 있다. 하지만 방법론을 도입했다는 것은 이제 겨우 향후 개발 조직에 가장 적합한 프로세스를 완성하기 위한 출발점에 섰다는 것으로 그 자체로 조직에서 사용할 방법론이 완성된 것이 아니라는 사실을 인식하는 것이 중요하다. 따라서 방법론을 도입했다고 하더라도 개발 조직 전체가 프로세스를 완성하는 주체가 되어 관심과 노력을 기울이지 않으면 프로세스 관리는 불가능하다. 이를 프로젝트 관리도구와 개발방법론 측면에서 살펴보면 다음과 같다.



[그림 2] 개발프로젝트 관리능력 향상 프레임워크

아래에서는 프로세스관리 수준을 향상시키기 위해 전개되어온 노력을 간략하게 소개하기로 한다. 전체 소프트웨어 라이프사이클 프로세스를 위한 개념적인 틀을 제공하기 위한 라이프사이클 참조모형과 개발조직의 개발프로세스 성숙도를 진단하기 위한 기준 등에 관한 노력이 소개된다.

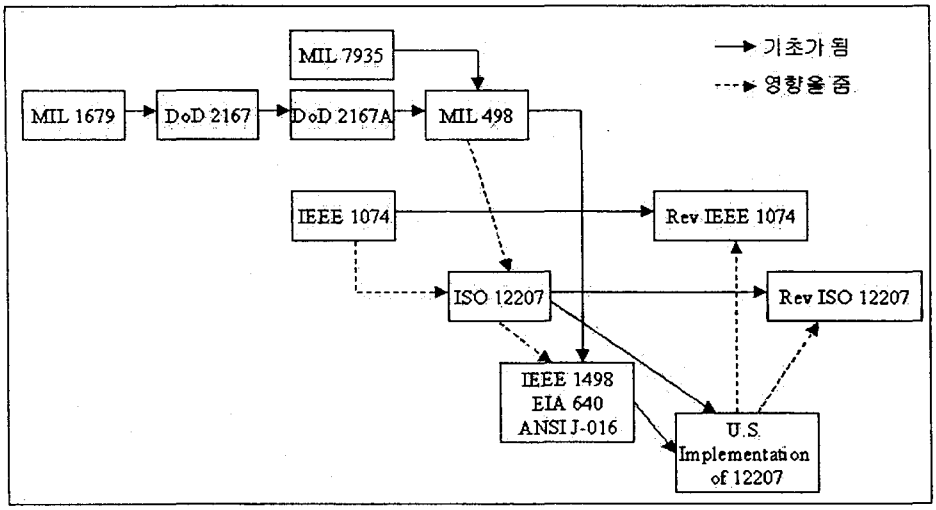
2.1 개발 프로세스 라이프사이클의 표준화

ISO 12207과 MIL-STD-498과 같은 개발 프로세스 라이프사이클 표준은 개념화 단계부터 퇴거에 이르는 전체 소프트웨어 라이프사이클 프로세스를 위한 개념적인 틀을 제공한다. 따라서 이러한 표준안은 특정한 라이프사이클 모델 혹은 소프트웨어 개발 기법이라기보다는 서로 납득할 수 있는 용어를 정의하기 위해 사용되는 프로세스의 골격을 제공하는 것으로 이해하는 것이 바람직하다.

먼저 ISO 12207 표준은 획득, 공급, 개발, 유지보수, 운용이라는 다섯 가지 "주요 프로세스"로 구성된다. 그리고 각각의 프로세스는 다시 "활동"들로 나뉘어지고, 각 활동은 실행을 위한 구체적 요구사항이 부여된 "태스크"들로 분할된다. 이외에도 ISO 12207 표준에는 문서화, 형상관리, 품질보증, 확인, 타당성 검증, 합동검토, 감리, 그리고 문제해결과 같은 여덟 가지 "지원 프로세스"와 관리, 하부구조, 개선, 교육훈련이라는 네 가지 "조직 프로세스"를 규정하고 있다.

MIL-STD-498은 소프트웨어 개발 프로세스를 위한 표준안으로 전체 시스템 라이프사이클에 적용할 수 있으며, 소프트웨어의 획득, 개발, 수정, 그리고 문서화를 위한 통합된 요구사항을 제시한다. 또한 표준화된 용어를 정의하고 아울러 소프트웨어 개발 및 유지보수 프로젝트를 위한 활동, 태스크 및 산출물에 대해서도 정의하고 있다. 이러한 MIL-STD-498은 어플리케이션 소프트웨어, 운영체제 소프트웨어, 펌웨어의 소프트웨어 부분, 재사용 소프트웨어, 그리고 소프트웨어 개발을 위해 도입된 소프트웨어 등 모든 종류의 소프트웨어에 적용할 수 있다.

이러한 시스템 개발 프로세스 라이프사이클 표준안의 발전과정은 다음과 같다.



[그림 3] 개발 프로세스 라이프사이클 표준의 발전과정[MOORE]

2.2 개발 조직의 프로세스 개선 방향

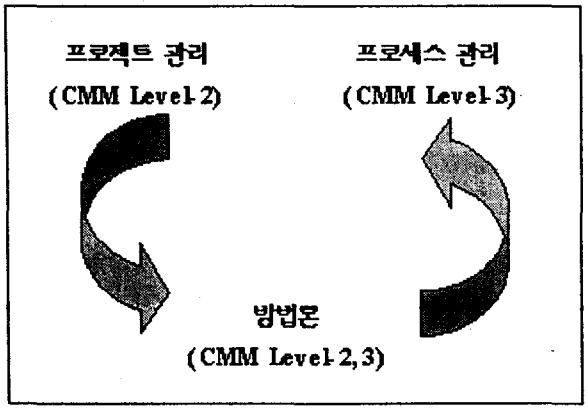
시스템 개발의 최종산출물인 정보시스템의 품질관리는 일반 제품의 그것과는 다르다. 정보시스템의 품질은 일반 제조공정과는 달리 개발자의 능력과 조직의 개발 프로세스의 안정성 및 그 수준에 따라 최종산출물의 품질이 좌우되는 특성이 있다. 이에 본 연구에서는 위에서 제시한 모델중 대표적인 사례인 CMM을 사용하여 개발 프로세스의 계층별 내용과 그 특징을 살펴보고자 한다. CMM(Capability Maturity Model)은 개발 조직의 개발 프로세스의 성숙도를 진단하기 위한 측정도구로 카네기멜론 대학의 소프트웨어 공학 연구소(Software Engineering Institute)에서 개발하였다. CMM은 소프트웨어 개발 조직이 자신들의 개발 노력을 얼마나 일관성 있고 효율적으로 투입하고 있는가를 평가할 수 있는 다섯 단계의 성숙도 계층(maturity level)으로 구성되어 있다. CMM의 목적은 효율적인 소프트웨어 개발 프로세스를 확립하는 것이다.

CMM에 따르면 성숙하지 못한 조직과 성숙한 소프트웨어 개발 조직간의 차이점은 자체적인 소프트웨어 프로세스의 존재여부와 성숙도에 따라 달라진다. 성숙하지 못한 조직은 산출물의 품질 혹은 개발상의 문제점을 평가하기 위한 목표와 기준이 없는 반면에 성숙한 조직은 관리자들이 소프트웨어의 품질과 고객 만족도를 계량적인 방식으로 측정하고, 개발 프로세스를 지속적으로 개선하려고 시도한다는 것이다. 아래의 표에서는 CMM의 프로세스 성숙도 계층과 계층별 특성을 제시하고 있다[Paulk, 1993].

	개발 프로세스가 거의 정의되어 있지 않고, 프로젝트의 성공여부는 주로 개인적 노력에 의해 좌우되는 단계(1995년의 경우 전체 정보시스템 개발의 대략 76% 정도가 이 단계인 것으로 추정됨)	프로세스를 예측할 수 없고 관리활동도 거의 이루어지지 않음
	비용, 일정 및 기능을 추적하기 위한 기본적인 프로젝트 관리 프로세스가 마련되어 있는 단계(1995년의 경우 전체 정보시스템 개발의 대략 15% 정도가 이 단계인 것으로 추정됨)	이전에 성공적으로 끝난 태스크를 다른 프로젝트에서 반복적으로 적용함
	모든 프로젝트가 전사차원에서 적용하기 위해 체계적으로 구성되고, 문서화되고, 표준화된 일련의 활동에 따라 수행된다. 이러한 활동에는 상세한 검토, 제품 공학, CASE 도구의 활용, 시험 표준, 그리고 전체 라이프사이클에 대한 형상 관리 등이 포함된다.(1995년의 경우 전체 정보시스템 개발의 대략 8% 정도가 이 단계인 것으로 추정됨)	프로세스의 내용이 상세하게 정의되고 팀원들이 이를 잘 이해하고 있음
	시간, 비용 및 기타 지표들에 대한 상세한 자료를 수집하고, 소프트웨어 개발 프로세스를 계량적으로 관리하기 위해 이를 사용한다. 이러한 조직은 품질 목표와 함께 개발활동을 지원하기 위한 도구와 교육훈련 프로그램을 가지고 있다.(1995년의 경우 전체 정보시스템 개발의 대략 1% 정도가 이 단계인 것으로 추정됨)	프로세스를 계량적인 지표를 사용하여 측정하고 통제할 수 있음
	계량적 관리를 통해 지속적으로 프로세스 개선이 이루어지는 단계이다. 소프트웨어 감사, 코드 워크스루, 자동화된 평가지표 수집 및 기술 검토 등과 같은 프로세스들이 표준적인 개발방법론의 일부를 형성한다.(1995년의 경우 전체 정보시스템 개발의 대략 1% 미만이 이 단계인 것으로 추정됨)	프로세스 개선에 초점을 맞추어 관리활동이 이루어짐

<표1> CMM 계층 및 계층별 특성

CMM에 따르면 소프트웨어 개발 프로젝트를 프로세스 개선이라는 관점에서 접근하기 위해서는 방법론, 프로젝트 관리, 프로세스 관리라는 세 가지 차원에서 적절한 활동이 필요한데 CMM의 각 계층과 이들 활동간의 관계를 도시해보면 다음과 같다.



[그림 4] CMM과 프레임웍간의 관계

CMM의 장점은 소프트웨어를 개발 및 유지보수를 담당하는 조직이 자신들의 프로세스를 어떤 방식으로 통제하고, 소프트웨어 엔지니어링 및 관리 수준을 어떻게 최고수준으로 개선할 것인지에 관한 지침을 제공하고 있다는 점이다. 그러나 현실적으로 CMM과 같은 참조모델들은 개발조직의 프로세스 개선을 위한 방향과 요건은 제시해주고 있지만, 구체적으로 이를 조직내에서 구현하기 위한 방법이나 수단은 제시해주지 못하고 있다. 따라서 이러한 접근방법은 개발 프로젝트와 관련된 경험이나 노하우를 체계적으로 축적·활용하려는 노력이 뒷받침되지 않은 상태에서는 충분한 성과를 기대하기 어려운 것이 현실이다. 이러한 문제를 인식하고 일부 개발 조직에서는 많은 비용을 투자하여 프로젝트 관리도구와 개발방법론을 도입하여 이러한 문제를 해결하려고 시도했지만 이 역시 단위 프로젝트(project by project)에 국한된 것으로 전사적인 관점에서의 개발 프로세스 개선이라는 본질적인 문제를 충분히 고려하지 않고 개발과 관련한 지식을 체계적으로 활용할 수 있는 수단이 제대로 없는 우리나라의 현실에서는 부분적인 해결책이 될 수밖에 없었다.

3. 프로젝트관리 및 개발방법론의 향상 측면

여기서는 개발프로젝트 수준의 향상을 위해 전개될 필요가 있는 개발방법론과 프로젝트관리의 측면을 소개하기로 한다.

3.1 프로젝트 관리도구 측면

전사적인 프로세스 개선 차원에서 단위 프로젝트를 관리하기 위해서는 먼저 단위 프로젝트를 수행하기 위한 최적화된 태스크를 의미하는 WBS(Work Breakdown Structure)를 바탕으로 프로젝트 계획 과정을 체계적으로 진행할 필요성이 있다. 이때 WBS는 유사한 프로젝트에서 수행된 태스크들을 통합하여 최적의 경험을 도출한 것으로 일정을 계획하고, 개발자들을 할당하고, 비용을 추정하기 위한 기준이 된다. 따라서 기존 프로젝트 계획서를 근간으로 작성되거나 방법론 레포지토리에서 추출된 WBS가 반드시 필요하다.

두 번째로 추진일정과 비용관리를 위한 프로젝트 관리시스템이 필요하다. 프로젝트 추진일

정은 문서로서만이 아니라 실제 개발자들이 이를 참조하여 작업을 진행할 수 있도록 가급적이면 개인별로 상세하게 제시되어야 한다. 물론 소규모 프로젝트에서는 개략 일정만으로도 충분하지만 규모가 크고 복잡한 프로젝트에서는 팀 또는 팀원별로 일정을 상세하게 관리해야 하므로 이러한 관리도구를 활용해야할 필요성이 크다. 따라서 단위 프로젝트의 수행경험을 프로세스 개선으로 연결시키기 위해서는 프로젝트 관리자가 개인별 또는 팀별로 상세한 일정계획을 수립하고 이를 토대로 프로젝트를 관리하고 보완하여 차후 유사한 프로젝트에서 이를 활용하는 과정을 반복함으로써 일정 및 비용예측의 신뢰도를 높여나가야 한다.

세 번째로 프로젝트의 추진현황을 파악하고 이에 따라 필요한 통제활동을 수행하기 위한 도구가 필요하다. 실제 프로젝트가 시작되면 관리자는 계획 대비 실적을 고려하여 필요한 관리활동을 수행해야 한다. 따라서 프로젝트 관리도구는 개발자별로 할당된 태스크의 수행상황과 문제점을 파악하여 대책을 수립할 수 있도록 필요한 정보를 제공하고 가능하다면 이러한 정보를 프로젝트에 참여한 모든 개발자들이 참조하여 업무에 반영할 수 있도록 정보를 공유할 수 있는 수단이 필요하다. 이러한 과정을 통해 어떤 태스크가 지체된다면 이를 전체 일정과 연결시켜 전체일정을 조정하거나 우선순위를 변경하여 문제를 해결하고 차후의 프로젝트에서는 이러한 정보를 고려하여 프로젝트를 계획함으로써 전체 조직의 개발 프로세스를 개선할 수 있다.

2.2 개발방법론 측면

정보시스템이 유사한 하드웨어, 운영체제, 그리고 데이타베이스를 바탕으로 개발되고 운영되던 시절에는 여러 개발 조직들의 작업방식이 서로 유사하였기 때문에 다년간의 개발 경험과 사례를 바탕으로 표준적인 프로젝트 라이프사이클을 정의하는 것이 가능하였다. 그리고 기술력이 있는 일부 개발 조직에서는 이러한 사례들을 체계적으로 수집하고 표준화하여 나름대로의 개발방법론을 정립하기도 하였으며, 이를 더욱 발전시켜 상용화된 제품으로 출시하는 조직도 생겼다. 그리고 이러한 역량을 구비하지 못한 조직은 막대한 비용을 투자하여 상용화된 개발방법론을 도입하여 자사의 환경에 맞도록 내용을 수정하는 접근법을 취하기도 하였다. 사실상 소프트웨어 개발 프로세스가 제대로 정립되지 못하고 필요에 따라 주먹구구식으로 작업하던 국내 실정에서 이러한 상용 방법론의 도입은 소프트웨어 공학의 개념을 확산시키는데 커다란 기여를 했다고 할 수 있다.

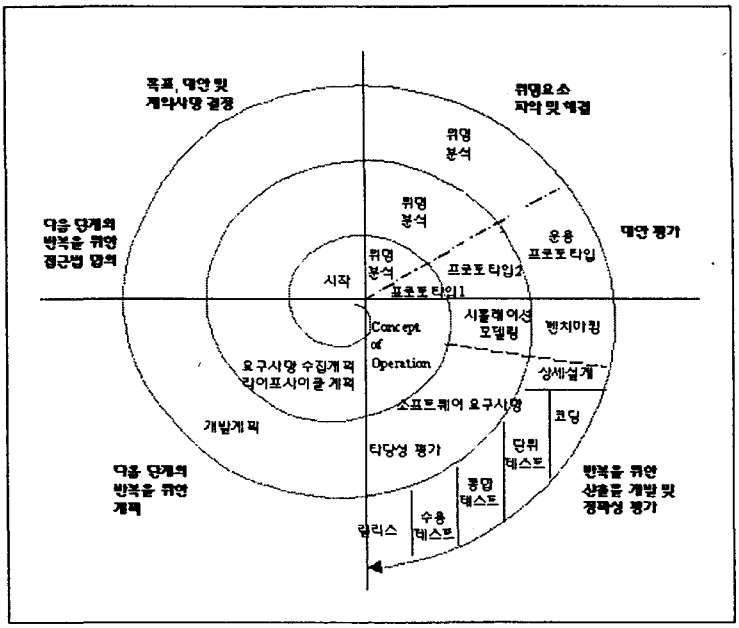
그러나 정보기술환경이 새로운 패러다임으로 급격하게 전환됨에 따라 이에 걸맞은 방법론을 도입할 필요성이 그 어느 때보다 절실하지만 조직내부의 경험과 역량이 뒷받침되지 않는 한 어느 방법론도 근본적인 해결책을 제시하지 못하고 있는 것 또한 현실이다. 이러한 혼란이 거듭됨에 따라 일부 조직에서는 이제 더 이상 방법론은 필요치 않다고 주장하며 그나마 체계적으로 접근하려던 노력들을 포기한 채 쏟아지는 신기술을 도입하는데 급급하여 다시 구심점이 없어지는 혼란이 초래되었다.

몇 년간의 혼란기를 거치면서 이제는 프로젝트의 규모나 시스템 또는 네트워크 아키텍처의 유형, 그리고 적용할 기법에 따라 전략적으로 프로젝트 라이프사이클을 결정해야 한다는 사실을 벤더와 개발 조직 모두가 이해하게 되었으며, 프로젝트의 초기단계에서 벤더로부터 제공되는 방법론에 대한 커스터마이징 작업을 수행해야 한다는 사실을 인식하게 되었다.

따라서 새로운 방법론은 가능한 다양한 유형의 프로젝트를 지원할 수 있도록 다양한 개발경

로를 제시하고 정형화된 태스크와 기법의 적용을 요구하기보다는 상황에 따라 최대한 융통성을 발휘할 수 있도록 세부 태스크보다는 전체적인 프레임워크만을 다룸으로써 상세한 부분은 필요에 따라 개발 조직이 재구성하여 사용하도록 하는 경향을 띄고 있다.

또한 방법론의 라이프사이클 패러다임 역시 많은 변화가 있었는데 최근의 경향은 앞 단계의 산출물이 뒷 단계의 입력물이 되는 순차적인 구조로 진행되는 폭포수 모형 대신에 철저한 계획과 검토작업을 거쳐 시스템 개발과정에서 발생할 수 있는 각종 위험요소들을 체계적으로 관리할 수 있는 나선형(Spiral) 패러다임으로 발전되었다[McConnel, 1996].



[그림 5] 나선형 모형

4. 프로세스 개선을 위한 지식관리적 접근방법

위에서 제시한 프로젝트 관리도구 및 개발방법론과 관련한 요건이 충족되었다 하더라도 정보시스템 프로젝트와 관련된 경험이나 노하우를 체계적으로 축적하고 활용할 수 있는 수단이 존재하지 않는다면 이를 프로세스 개선으로 연결시키기는 어렵다. 따라서 필연적으로 지식관리 개념을 도입해야 하는데, 지식관리란 기업이 가지고 있는 유·무형의 정보자산을 관리하기 위한 정형화되고 통합된 접근방법으로 기업이 보유하고 있는 다양한 형태의 지식을 공유하고 재사용하기 위한 여러 시도들을 조합한 것이라 할 수 있다. 이러한 노력을 조직의 경쟁력으로 연결하기 위해서는 구축된 지식을 실제 업무에 응용하고 또한 이러한 과정을 통해 새로운 지식이 창조되는 순환과정을 만들어 가야 한다. 즉 이러한 순환과정이 기업의 지적능력이며 이것이 축적된 지식자본과 서로 보완적이며 상승효과를 발휘할 때 비로소 전체적인 조직의 경쟁력을 향상시킬 수 있는 것이다[강구영].

따라서 이러한 문제점들을 근본적으로 해결하고 프로세스를 개선하기 위해서는 필연적으로 지식관리적 개념이 구현된 프로젝트 하부구조(Project Infrastructure)를 구축해야 한다. 프로젝트 하부구조의 목적은 단위 프로젝트 차원이 아니라 장기적이고 전사적인 관점에서 개발

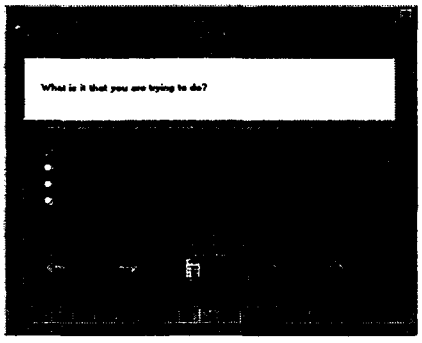
조직의 역량을 향상시킬 수 있는 방안을 구현하기 위한 것이다. 프로젝트 하부구조를 구축하기 위한 접근법으로는 전문가 시스템을 이용한 어드바이스를 사용하는 방법, 개발 조직에서 발생하는 모든 유형의 정보를 저장할 수 있는 전사적 정보저장소(Corporate Repository)를 이용하는 방법, 그리고 인트라넷 시스템을 이용하여 통합 프로젝트 관리환경을 구축하는 방법 등이 있다. 지식관리적 접근방법의 세 가지 유형과 각 대안별 장단점을 살펴보면 다음과 같다.

4.1 전문가 시스템을 이용한 어드바이스를 사용하는 방법

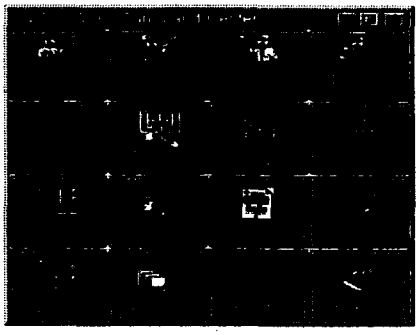
정보시스템 개발을 위해 WBS를 만들고, 기간을 예측하여 비용을 산정하고, 프로젝트 진척상황을 체계적으로 관리하고, 방법론에서 정의한 산출물을 제대로 작성하기 위해서는 오랜 경험과 방대한 지식이 필요하기 때문에 개발경험이 적은 조직의 경우 단시간 내에 이러한 요건을 구비하여 개발 프로세스를 정립하기는 현실적으로 어렵다. 따라서 이러한 경우에는 필요한 기능에 대해 외부업체의 컨설팅을 받는 것이 일반적이지만 이 역시 비용문제 때문에 현실적으로 쉽게 수용할 수 있는 대안은 아니다. 이 때문에 전문가의 도움이 필요할 경우 이와 유사한 기능을 제공해주는 전문가 시스템으로 구축된 어드바이스 기능을 이용함으로써 비교적 적은 투자로 유사한 서비스를 제공받을 수 있는 시스템이 새로운 대안으로 부상하였는데, CSCL(Client/Server Connection, Ltd)사에서 개발한 CS/10,000이 대표적인 예이다.

4.1.1 CS/10,000의 예

CS/10,000은 시스템이 사용자에게 질문을 던지면 사용자는 자신의 요구를 가장 잘 반영하고 있는 대답을 선택하는 과정을 통해 사용자들의 요구사항을 수집하고, 이러한 과정이 완료되면 전문가 시스템인 Advisor가 추론엔진을 작동시켜 최적의 대안을 선정해주는 방식으로 작동된다.



[그림 6] 요구사항 수집과정



[그림 7] CS/10,000의 전문가 시스템

CS/10,000에서 제공하는 핵심기술과 그 내용은 다음과 같다.

	템플릿 기반의 "가상 방법론(Virtual Methodology)"이 태스크 관리자, 문서화 도구, 개발 도구, 프로젝트 예측 및 전사차원의 표준을 관리하는 하부시스템과 통합되어 있다. 또한 프로젝트 관리를 위한 그룹웨어 형식의 인터페이스는 프로젝트 계획(Project Plan)에 프로젝트 수행을 위해 필요한 태스크와 태스크별로 할당된 개발자들의 작업시간을 예측하여 기록하고, 실제 프로젝트가 시작되면 자동적으로 타임쉬트를 통해 개인별로 보고하는 실제 작업시간을 바탕으로 통합해준다.
	전문가 시스템이 자동적으로 최적의 아키텍처를 선정하고, 방법론을 커스터마이징하고, 프로젝트의 수행기간을 예측해준다. 또한 프로젝트 예측을 위한 전문가 시스템은 업무상의 실제적인 경험과 프로젝트 수행결과에 따라 프로젝트에 대한 예측치를 개선하기 위해 자체적으로 학습하고 필요할 경우 이를 재구성할 수 있다.
	시스템의 주요 구성요소들에 대한 시각적인 다이어그램을 자동적으로 선정할 수 있고 필요할 경우 이를 쉽게 커스터마이징 할 수 있다. 잘 정의되고 문서화된 아키텍처는 프로젝트 조정(coordination)과 커뮤니케이션에 있어 핵심적인 요소이다.
	표준적인 인터페이스와 네트워크로 연결된 개방형 레포지토리를 통해 현재 수행하고 있거나 여태까지 수행한 모든 프로젝트에 대한 정보를 활용할 수 있다. 네트워킹이 가능하고 적절한 사용권한을 가진 모든 사용자들은 자신이 수행하고 있는 프로젝트에 관한 정보뿐만 아니라 다른 프로젝트와 프로세스 관리 도구들도 활용할 수 있다.
	태스크 수행과 관련한 표준은 한 번만 정의해두면, 전사차원의 표준을 관리해주는 모듈을 통해 모든 개발자와 관리자들이 이를 활용할 수 있다.

<표2> CS/10,000의 핵심 기술

5.2 전사적 정보저장소(Corporate Repository)를 이용하는 방법

개발 프로젝트와 관련한 개발 조직의 지식을 체계적으로 활용하기 위해서는 전사차원의 지식공유 체계(Enterprise Knowledge Architecture)를 도입하고, 이를 바탕으로 필요한 지식을 획득, 공유 및 활용할 수 있는 지식관리 프로세스가 필수적이다. 그러나 대개 정보시스템 개발 프로젝트 수행을 위해 필요한 지식은 사내외 응용시스템, 데이터베이스 및 파일, 일반문서 등 다양한 시스템에 이질적인 형태로 존재하며, 개인에게 체화된 지식(Tacit Knowledge)도 많아 공유하기 어려운 경우도 있다. 게다가 지식은 고정되어 있는 것이 아니라 환경변화에 따라 역동적으로 변화하는 특성을 갖고 있기 때문에 지속적으로 관리해야 한다는 문제점이 있다. 따라서 이러한 환경에서는 개인의 지식과 분산된 형태로 존재하는 다양한 지식을 통합하여 조직의 지적자산으로 변환하고 창조하는 지식관리가 경쟁력의 핵심이며, 지식경영의 성패는 바로 이러한 프로세스를 어떻게 잘 지원할 수 있는가에 달려 있다. 이러한 요건을 충족할 수 있는 대안이 바로 전사차원의 정보저장소, 즉 Corporate Repository인데 독일 Rochade사에서 개발한 Rochade가 그 대표적인 예이다.

5.2.1 Rochade의 예

전사적 정보저장소(Corporate Repository)는 소프트웨어 개발과 어플리케이션 개발 프로세스

와 관련된 모든 정보를 관리하기 위해 만들어진 실체이다. 여기서 중요한 것은 시스템 개발과 관련된 모든 정보를 저장하는데 그치지 않고 이러한 정보를 관리하고 공유함으로써 어떻게 가치를 부가할 것인가에 초점을 맞추고 있다는 점이다. 레포지토리에 저장할 수 있는 정보로는 기존 시스템 명세서와 CASE 도구에서 사용하는 오브젝트들로부터 프로젝트 계획서, 테스트 시나리오, 그리고 실제 운용중인 시스템에 대한 정보에 이르기까지 광범위한 내용을 포괄하고 있다. 이러한 정보와 이들간의 관계는 레포지토리 정보 모델(Repository Information Model; RIM) 내에 저장된다. 따라서 레포지토리의 핵심은 RIM이라 할 수 있다. RIM은 레포지토리에 접근하기 위한 매커니즘을 제공하고, Rochade 주제영역과 프로젝트, 그리고 CASE 도구 등과 같은 외부 도구들을 지원하는데 필요한 세부사항들을 정의한다. 레포지토리에 저장되는 정보의 내용, 구조, 그리고 문법(syntax)은 조직의 레포지토리 정보 모델을 생성하고 수정함으로써 결정된다. RIM은 무한히 확장시킬 수 있으며 조직이 레포지토리와 커뮤니케이션하는데 필요한 정확한 정보들을 반영하기 위해 적절한 권한을 가진 사람들이 이를 수정할 수 있다[Rochade].

Tannenbaum은 레포지토리가 갖추어야할 요소로서 다음과 같은 다섯 가지를 강조하고 있다 [Tannenbaum, 1994].

① 레포지토리는 통합된 자료보관 장소이다. 따라서 저장되는 모델의 형태와는 무관하게 어떤 유형의 모델이건 레포지토리 내에 저장된 다른 모델과 연계시킬 수 있어야 한다.

② 모델은 또한 레포지토리 외부에 존재하는 물리적 구현결과와 연계될 수 있어야 한다.

③ 레포지토리는 개방적이어야 한다. 즉 레포지토리로의 접근과 자료입력, 그리고 레포지토리의 구조 및 내용물들은 이를 제공하는 특정 벤더와 독립적이어야 한다. 또한 레포지토리의 메타모델은 확장성이 있어야 한다.

④ 레포지토리에 저장된 내용물들은 사전에 정의한 뷰 또는 템플릿을 통해 조회할 수 있어야 한다. 레포지토리에는 DBMS의 구현내용을 설명하는 메타데이타로부터 특정부서의 목표를 묘사한 메타데이타에 이르기까지 조직의 메타데이타와 관련된 다양한 내용들이 포함될 수 있으므로 메타데이타에 대한 접근은 목적에 따라 재구성할 수 있어야 한다. 또한 데이터베이스의 뷰와 마찬가지로 레포지토리 뷰 역시 정의하고 유지보수할 수 있어야 한다.

⑤ 레포지토리의 내용물들에 대해 버전 관리를 할 수 있고, 여타의 기업 DBMS와 마찬가지로 레포지토리 역시 보안기능이 필요하다. 버전 관리는 바로 모델, 하부모델, 그리고 모델 구성요소 차원의 보안관리를 위해 필요하다.

Rochade는 다음과 같은 기능을 통해 이러한 개방성과 확장성을 지원하고 있다.

① 완벽하게 재구성이 가능한 레포지토리 정보 모델(레포지토리 내에 모델로 저장된 실세계의 정보구조)

② 외부 도구들과의 인터페이스(API 제공)

③ 최종사용자 인터페이스

④ 레포지토리 내에서 규칙(rules)과 행위(behaviors)의 구현을 위한 내장 프로시저 랭귀지

이를 통해 Rochade Corporate Repository는 조직의 어플리케이션 시스템, 데이터베이스, 그리고 개발도구들간의 데이터와 프로세스에 대한 집중화된 정보관리를 위한 단일의 통제포인트를 제공한다. 이를 위해 Rochade는 정보관리기능뿐만 아니라 어플리케이션 개발 프로세스와 같이 저장된 정보와 관련된 프로세스 관리기능도 제공한다.

5.3 인트라넷 시스템을 이용하여 통합 프로젝트 관리환경을 구축하는 방법

지식관리(Knowledge Management)와 관련하여 중요한 이슈중의 한 가지는 지식의 획득방법뿐만 아니라 이렇게 얻어진 지식을 어떠한 방식으로 배포 또는 공유할 것인가 하는 문제이다. 이 외에도 기존 정보시스템의 이질성, 상이한 데이터와 지식의 표현, 데이터 시멘틱(semantic)과 어플리케이션 인터페이스가 비호환성 등이 지식관리시스템의 아키텍처를 정의하는데 있어 가장 큰 문제점으로 나타나고 있다[김기섭]. 따라서 대부분의 기업들이 최근 인트라넷 환경으로 옮겨가는 것에 가장 높은 우선순위를 부여하고 있는데 이는 기존 지식의 출처와 별도의 도구를 연결하고 다양한 형태의 데이터와 텍스트, 그리고 다양한 어플리케이션에 엑세스할 수 있는 단일 창구를 제공하자는 목적과 지식관리시스템의 구조를 정의하는 과정에서 발생하는 어려운 문제들을 손쉽게 해결하려는 목적도 내포하고 있다. 인트라넷 시스템으로 구현된 통합 프로젝트 관리환경의 예로는 LG-EDS의 MethodPlus와 SDS의 Innovator가 있다.

5.3.1 LG-EDS의 MethodPlus의 예

인트라넷 시스템을 이용한 통합 프로젝트 관리환경은 지식의 활용이라는 측면을 강조하고 있는 접근방법이다. MethodPlus 역시 기존 프로젝트를 수행하는 과정에서 도출된 경험을 집대성하여 이를 쉽게 활용할 수 있도록 다양한 개발경로를 제시하고 이에 필요한 프로젝트 관리 도구 등을 지원하고 있다. MethodPlus의 특징은 다음과 같다.

방법론 지원 도구를 장착
Task가 구현되는 기술별로 그룹핑되어 프로젝트에 맞게 조합하여 사용할 수 있음
프로토타이핑 라이프사이클 지원
다양한 개발 경로 지원 (DW, OO, Internet/Intranet, Client/Server, Package Implementation 등)

<표3> MethodPlus의 특징

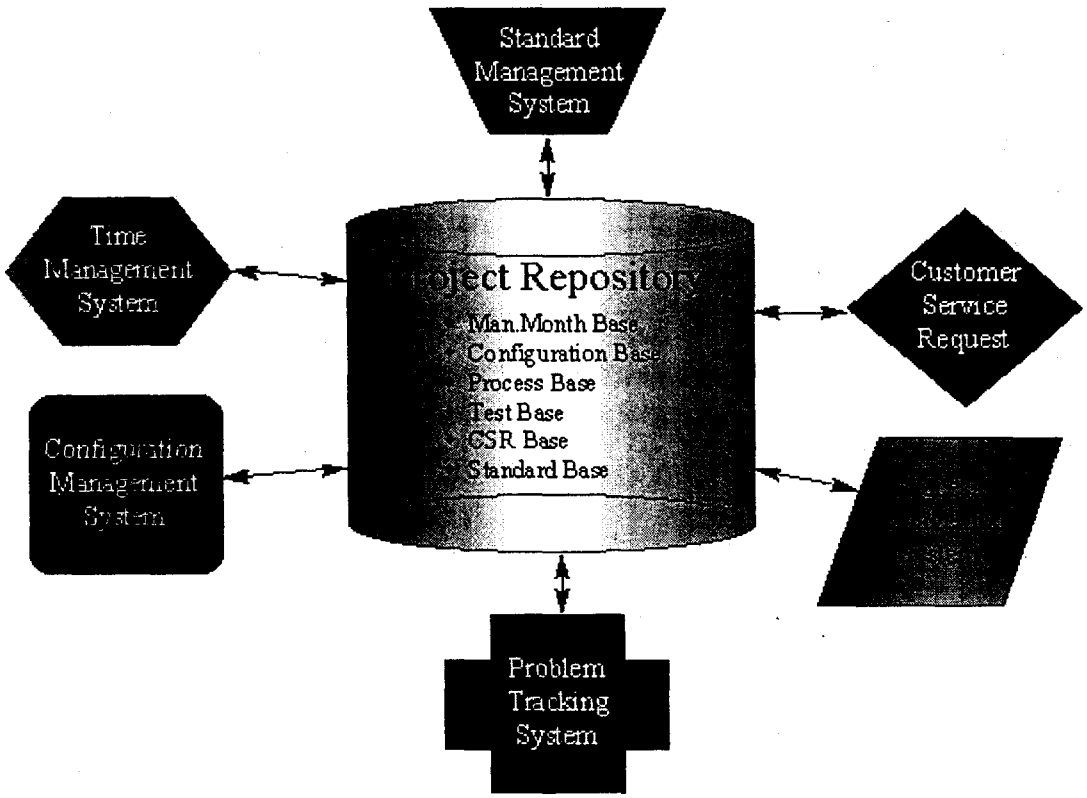
5.3.2 대법원 부동산 등기부 전산화 프로젝트의 프로젝트 워크벤치의 예

프로젝트의 규모가 커질수록 프로젝트의 성패를 가르는 중요한 요소로 작용하는 것이 프로젝트 관련자들간의 커뮤니케이션이다. 대법원 부동산 등기부 전산화 프로젝트팀에서는 커뮤니케이션의 활성화를 목표로 함은 물론 프로젝트팀의 프로세스 성숙도를 높이기 위하여 다음과 같은 서브 시스템을 구축, 단계적으로 통합 시켜나가고 있다.

시스템명	지원 기능	관련 CMM KPA
시간 추적 시스템	프로젝트의 개발 단계별 투입된 공수를 관리하여 프로젝트내에서는 물론 사내에서 과학적인 공수 산정에 필요한 정보를 제공함.	Software Project Planning
산출물 버전관리 시스템	전 개발 단계에서 생성된 제반 상위단계 산출물들에 대하여 버전관리를 해주고 프로젝트내에서 쉽게 공유가 가능하도록 지원함.	Software Configuration Management
구성관리 시스템	실행 모듈과 연관된 오브젝트들의 정보를 관리해주고 이들간의 관계를 쉽게 파악하도록 하여 하위 단계의 오브젝트들에 대한 버전관리를 지원함.	Software Configuration Management
오류 추적 시스템	기능 시험시 발견된 오류가 조치될 때까지의 일련의 라이프 사이클을 추적하여 신속하게 오류가 처리되도록 도와 주며 오류의 발생원인에 대한 통계자료 제공으로 근본적인 오류 예방책을 수립하도록 지원함.	Software Quality Assurance
교육 일정 안내	프로젝트팀내에서 실시되는 업무 및 시스템 교육 일정을 알려준다.	Trianing Program

< 표4> 대법원 부동산 등기부 전산화 프로젝트의 프로젝트 워크벤치 구성 시스템별 주요 기능

프로젝트 워크벤치는 향후 추가적인 서브 시스템 개발로 궁극적으로는 프로젝트내 통합 레파지토리의 역할을 수행할 것이며, 이를 통해 각종 프로젝트 매트릭스정보를 추출하게됨으로써, 매트릭스를 통한 과학적인 프로젝트 관리가 가능하도록 하며, 프로세스 관리도 가능하게 할 예정이다.



[그림 8] 대법원 프로젝트 레파지토리 개념도

6. 결론 및 향후 연구방향

위에서 개발조직의 프로세스 개선을 위한 지식관리적 접근방법의 세 가지 유형을 제시했다. 이들 각각의 대안들은 나름대로의 특징이 있는데 첫 번째 대안인 전문가 시스템을 이용한 어드바이스를 사용하는 방법은 개발경험이 일천한 개발 조직이 전문가 시스템을 이용하여 비교적 적은 투자로 신속하게 개발 프로세스 관리를 위한 환경을 구축할 수 있다는 장점이 있다. 반면에 이러한 접근방법을 사용할 경우 조직의 성숙도나 생산성이 시스템에서 가정하고 있는 내용과 차이가 있을 경우 이를 자사의 환경에 맞도록 근본적으로 재구성하기가 어렵다는 문제점이 있다. 또한 전문가 시스템이 제시해주는 대안의 신뢰성을 검증할 수 있는 방법이 없다는 점과 대안을 적용하는 과정에서 필요한 상세한 요구사항들에 대해서 제대로 지원 받을 수 없다는 한계점이 있다.

두 번째 전사적 정보저장소를 활용하여 개발 조직에서 발생하는 모든 유형의 정보를 저장하여 활용하는 방법은 첫 번째 대안에 비해 보다 근본적인 해결책이라 할 수 있다. 전사적 정보저장소를 이용할 경우 개발 조직에서 필요한 다양한 유형의 정보를 유연하게 저장할 수 있고 이를 신속하게 추출할 수 있는 수단을 제공해준다는 측면에서 보면 장기적인 관점에서 가장 바람직한 해결책이라 할 수 있다. 그렇지만 현실적으로 정보저장소에 저장할만한 지식의 내용과 양이 충분하지 못할 경우에는 적용하기가 어렵고 이를 구현하는 과정도 다른 대

안에 비해 많은 준비와 시간이 필요하다는 단점이 있다.

마지막으로 인트라넷 시스템을 이용하여 통합 프로젝트 관리환경을 구축하는 방안은 정보의 활용 측면을 강조한 접근방법으로 이러한 접근법을 따를 경우 비교적 손쉽게 조직에서 보유하고 있는 자원을 공유할 수 있는 체계를 마련할 수 있고 구축 역시 비교적 간단하다는 장점이 있다. 반면에 프로세스 개선에 필요한 여러 기능들을 제공하기가 어렵고, 전담조직이 프로젝트 수행과정에서 발생하는 방대한 자료를 신속하게 수집하여 내용을 갱신해야 하는 **부담이 있고, 내장된 지식의 품질을 사용자들이 신뢰하지 않을 경우 활용도 역시 아주 낮을 수 있다는 단점이 있다.**

따라서 바람직한 대안은 개발 조직이 처한 상황에 따라 단기적으로는 첫 번째 대안과 세 번째 대안을 선택하여 적용하고 장기적으로는 근본적인 지식공유 및 활용체계를 구축할 수 있도록 두 번째 대안을 고려하여 지식관리체계를 정립해 나가는 것이 바람직하다고 생각한다.

참 고 문 헌

강구영, "지적 자본의 구성과 측정에 대한 접근", 포스코경영연구소, 1998

김기섭, "인포믹스의 KMS 요소기술 및 솔루션 소개, *KMS 기획특집*

김성근, 이주헌, 최성운, 조창현, *정보시스템 객체지향 개발감리지침 연구 중간보고서*, 한국전산원, 1998

Moore, Jim, "ISO 12207 and Related Software Life-Cycle Standards", ACM(www1.acm.org:81/tsc/lifecycle.html)

Kaplan, Jeffery "Raising your network management project batting average", *Network World*, 1998, Aug. 17

King, Julia "Poor planning kills projects, pushes costs up", *Computerworld*, 1997, Sep. 22

Laudon, K. and J. Laudon, *Management Information Systems: New Approaches to Organization & Technology*, Prentice-Hall, 1998.

McConnell, Steve. *Rapid Development*, Microsoft Press, 1996

Paulk, M., B. Curtis, Chrissis, M. and Weber, C. "Capability Maturity Model for Software, Version 1.1, SEI, 1993

PRIMAVERA *News Letter*, 1995. 7, Vol.2, Number 3

Redmill, Felix *Software Projects*, Wiley, 1997

Rochade, *Rochade and AUTOPILOT Concepts*, Rochade, 1996

Tannenbaum, Adrienne *Implementing A Corporate Repository*, John Wiley & Sons, 1994

van Genuchten, Michiel, "Analysis and improvement of software engineering processes", *Information and Management*, 1993(vol. 25)

Ward, J. "Project Pitfalls", *Information Systems Management*, 1995(Vol 12) No.1-4