설계정보 구축 및 재사용을 위한 지식베이스 시스템

이 수 홍*

1. 서 론

국제경쟁력의 증대와 시장수요의 빠른 변화에 대해 새로운 제품을 빠른 시간내에 제공해야 하는 오늘날의 세계 시장은 점점 그 경쟁이 복잡해지고 있다. 이 경쟁에서의 승리는 적절한 새모델을 원가절감과 설계오류 등을 최소화하면서 적절한 시기에 제시하느냐에 달려있다. 불행히도 제품설계 및 제조사이클에는 종종 많은 시간이 요구되는 루프가 포함되어 있다. 즉 설계변경과 재작업 사이클 때문에 시장출하시간이 지연된다. 이러한 변화를 동적으로 취급하고 적절한 시점에 적절히 이동할 수 있는 전략을 개발하는 것이 오늘날 세계경쟁에서의 성공에 필수적이다. 이러한 문제점들의 해결방안으로 동시공학, 혹은 지식베이스 시스템을 포함하는 지식베이스엔지니어링(Knowledge-Based Engineering, KBE)이 소개되는데 이러한 지식베이스엔지니어링은 기존의 제품개발 방법론을 개선하기 위한 지능적인 설계도구로서 컴퓨터를 사용하는 새롭고 흥미로운 가능성을 제시한다. 이러한 설계도구는 조직적이고 체계적으로 지식을 수집하여 활용함으로써 동시공학(Concurrent Engineering)방법론을 효율적으로 수행하는 데에 매우 유용하다.

이러한 컴퓨터기술을 제품개발과정에 응용하는 것이 분명히 새로운 아이디어는 아니다. 대부분의 회사들은 이미 어떠한 형태로든 CAD시스템을 실행하여 왔다. 비록 자동화 도구들이 각 부서에서 생산성을 향상시켜 왔지만, 총괄적이고 체계적 이익을 얻기가 어렵다는 것이 종종 입증되어 왔다. 일부 관리자들은 기술의 변화로 인해 단순히 한 부서로부터 다른 부서로 비용과 시간, 애로공정이 이전되는 것으로 생각한다. 또한 우리가 경쟁하는 환경은 항상 그 변화속도가 빨라지고 있다. 새로운 아이디어나 새로운 제품기술은 종종 개발과정을 통해 발견되는 수가 많다. 사용자는 자기들의 요구에 대해 즉각적으로 대응해 줄 것을 요구하며, 개발과정의 마무리 단계에 가서 제품에 대한 변경을 요구하는 경우도 있다. 대부분의 회사들은 이러한 변화들을 미리 파악하여 경쟁에서 승리하는 방법을 모색하고 있다.

1.1 지식베이스엔지니어링-규칙베이스시스템-전문가시스템-객체지향프로그래밍

이 용어들은 컴퓨터기술이 점점 발전되어 환경 및 기술 변화를 다루는 능력이 크게 개선됨을 일컫는다. 이 기술의 핵심은 "설계보정"이라 불리는 총괄적이고 체계적인 설계철학에 있다. 이 "설계보정"철학은 동시공학의 조직철학과 유사하다. 설계사이클에서의 각 의사결정은 관련된 모든 요소들을 고려하여 여러번의 "설계보정" 혹은 "설계수정"을 통해 이루어지기 때문이다. 그림 1에서 보여진 것과 같이 기존의 제품 설계 제작 과정은 단계적으로 이루어지므로 어느 한 단계에서라도 일의 진행이 마무리 되지 않으면 다음 단계로 넘

* 연세대학교 기계공학과 교수

어 가지 않으므로 설계에서 제품 생산에 이르기까지 상당한 시간이 요하며, 만의 하나 어느 공정이 잘못되어 수정하고자 할 때도 역시 상당한 시간의 공정을 요한다. 반면 그림 2에서 보여진 것과 같이 중앙에 있는 제품 사양들을 중심으로 각 분야별 단계들이 분산되어 구성되어 있으면, 앞서 언급한 문제들이 쉽게 해결될 수 있다.

고전적인 제품개발과정은 "수행-평가-개선(변경)"의 접근방법에 기초를 두고 있다. 이것은 공학에서 어떤 설계를 제안할 때 시작된다. 다음에는 다른 것들이 그 설계를 평가하고 변화를 제안하며 문제점을 지적한다. 이 사이클에는 상당한 시간이 소모되며, 이 과정에서 설계 변경이 늦게 제안되는 경우에는 아주 많은 비용이 소요될 수도 있다.

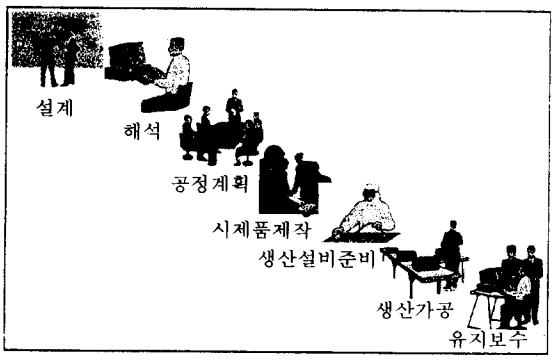

그림 1. 단계적으로 이루어지는 기존의 제품 설계 제작 과정

지식베이스엔지니어링의 툴로 사용되는 전문가 쉘(Expert-Shell, 전문가 시스템과 구별됨.)은 이 사이클을 시스템내의 객체지향형프로그램 환경의 특성(속성전이, 정보 공유 등.)들을 이용하여 총괄적이고 체계적인 지능적 제품모델을 생성하도록 돕는다. 이 지능적 모델에서는 기존 조직의 데이타베이스와 전문 기술진들의 경험적 지식을 사용하여 동시공학을 위한 지원환경을 구축한다.

전문가 쉘에서는 제품모델의 각 구성요소가 그것의 구축 및 다른 객체들과의 상호작용에 필요한 모든 규칙들을 가진 하나의 지능형 객체(Object)로 정의된다. 또한 전문가 쉘은 전문 기술인의 설계규칙을 보강해 주며, 규칙의 적합여부를 검색한다. 파라메트릭(Parametric) CAD/CAM 시스템과 달리 이들 규칙은 부품의 기하학적 형상과 관련될 필요가 없다. 즉 전문가 쉘에서는 부품의 기하학적 형상이 단지 제품모델의 또 다른 하나의 속성일 뿐이다. 일반적으로 전문가쉘이 갖추어야 할 기본환경으로 다음의 5가지를 들 수 있다. 구비조건 (3), (4), (5)는 대부분의 전문가쉘이 갖고 있으나 조건 (1)과 (2)는 고가의 제품에만 장착되어 있다.(예 : Concentra사의 IDL 시스템)

(1) CAD와의 인터페이스
(2) DB와의 인터페이스
(3) 속성전이
(4) 메시지 전송
(5) GUI구축의 용이성

1.2 지식베이스엔지니어링의 활용

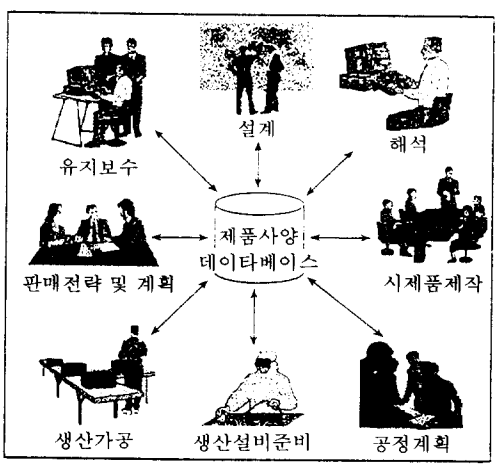

그림 2. 중앙에 있는 제품사양들을 중심으로 각 분야별 단계가 분산되어 구성되는 병행 엔지니어링 시스템의 구조도

전문가 쉘에서는 규칙을 갖는 지능형 객체와 부품을 정의하는 것이 가능하다. 이들 부품은 서로 결합되어 하나의 지능형 제품모델로서 상호작용한다. 각 객체는 모델 내에서의 자신의 위치, 다른 객체와의 연관 관계 및 기능을 알고 있다. 지능형 부품들의 상호작용은 기존의 도구와 기술로는 얻을 수 없는 동적인 차원을 제공한다. 또한 지능형 제품모델은 각각의 독립적인 그룹의 지식과 지시를 받아, 그것을 사용자의 전체 설계 및 제조팀에서 사용할 수 있도록 한다. 공간적으로 떨어진 장

소에 있더라도 제품 개발팀은 동일한 지식베이스를 사용하여 동시에 작업을 수행한다. 이러한 기능이 전문가 쉘에서는 쉽게 구현될 수 있는데 이는 동시공학 접근방법의 필수적인 피드백을 이용할 수 있기 때문이다.

지식베이스엔지니어링 도구는 기존의 자동화도구와 공동으로 작업을 수행하고, 이미 사용되고 있는 기술을 활용해야 한다. 설계는 어떻게 만들어지고 그것이 왜 특정의 방법으로 생성되어야 하는 가에 대해 완전히 이해하게 되면, 전문가 쉘은 이 지식을 적용하여 비용을 최소화하고 사용자의 요구에 대해 손쉽게 대처할 수 있다.

생산 제약조건이 공학적 활동 중에 자동적으로 고려될 수 있도록 모델에 추가할 수 있다. 이렇게 함으로써 생산의 문제점으로 인한 재설계의 필요성을 최소화한다. 설계과정을 생산 능력에 맞춤으로써 사용자의 필요조건에 신속하게 대응하고 시장출하시간을 빠르게 할 수 있다. 경쟁적인 게임에 있어서 변화는 불가피하다. 설계 변경은 이전에 어떤 일이 발생하였는가와 앞에 무엇이 놓여 있는가에 근거를 둔다. 지식을 획득하고 그것을 변화하는 환경에 적용하는 것은 어떠한 상황이든 그것에 대처하기 위해서 필수적이다. 사용자는 지식베이스엔지니어링 활용으로 변화를 예측함으로써 적절한 때에 적절히 대치 할 수 있다.

1.3 동시공학(Concurrent Engineering)

동시공학 철학은 명확한 하나의 목표이지만, 단지 예로부터 실시되어 오던 상식이며 팀웍일 뿐이다. 분명 제조는 어떤 설계의사 결정이 실현불가능한 제품을 초래할 때에 공학부서에 경고를 주어야 한다. 마찬가지로, 재무에서는 어떤 설계의사 결정이 너무 값비싼 제품을 초래할 때에 공학부서에 경고를 주어야 한다. 설계과정에 포함된 모든 사람들로부터 피드백을 적절한 시기에 획득하는 것이 재작업을 피하고 제품설계사이클을 단축하는 핵심이다. 기업에서는 동시공학의 이러한 이점을 인식하고 있다. 동시 공학의 개념을 효율적으로 병합하는 설계과정의 실행은 현실적으로 해 볼만한 가치있는 일이다. 설계의사 결정들을 주고받고 여러 원칙들의 현장 지식을 얻는 일은 어렵다. 그럼에도 불구하고 설계의사 결정은 제품개발 사이클 도중에 계속적으로 이루어진다. 지식베이스엔지니어링은 이러한 동시공학 철학을 실행하

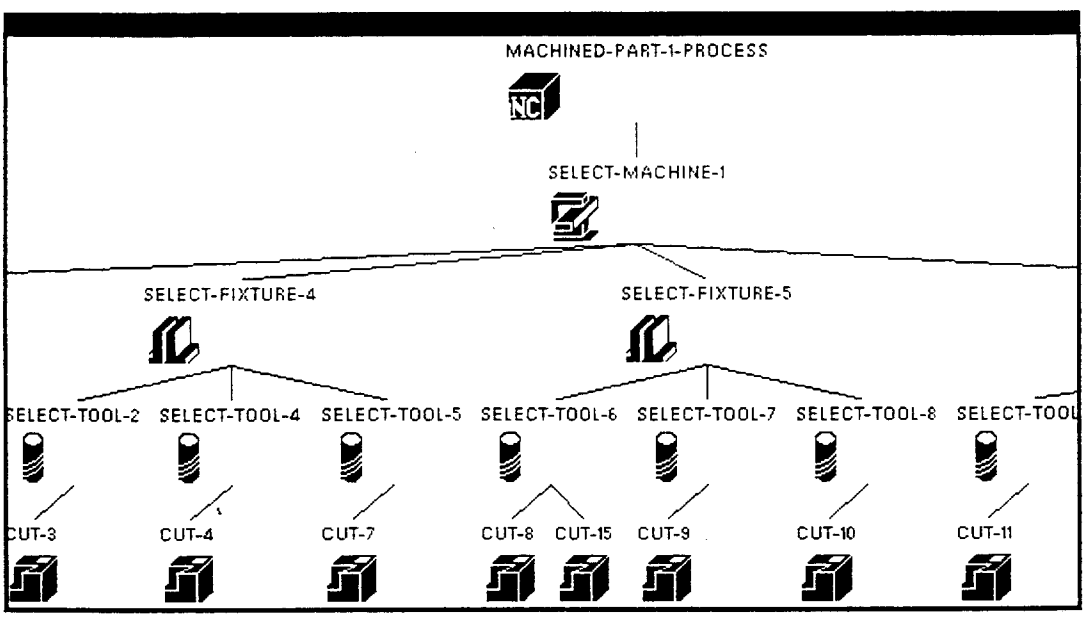

그림 3. 절삭가공용 지식베이스엔지니어링 시스템[NextCut,|이수흥 1991]에서 중앙지식베이스 내에 저장되어 있는 공정 정보의 연관관계 및 가공순서 정보를 보여준다. 이와같이 지식베이스시스템에서는 사용자에게 설계 정보의 연관관계를 보여주며 보다 편리한 작업 환경을 제공한다.

는 기초이다. 지식베이스엔지니어링이 어떻게 동시공학을 가능하게 하는 가를 보다 잘 이해하기 위해서는, 설계과정과 설계과정에 있어서의 컴퓨터의 이용을 보다 더 연구하는 것이 효과적일 것이다.

2. 설계 및 제조에 있어서의 컴퓨터

어떤 기술이 생성으로부터 소멸되기까지의 사이클을 예측하기는 매우 쉽다. 일반적으로 기술이란 문제를 해결하려는 노력으로부터 시작된다. 다음 단계로 현재의 기술을 사용하여 하나의 해결안을 고안해 낸다. 마지막 단계에 이르러 이 기본 해결안이 장시간의 지속적인 개선과 새로운 기술의 응용으로 들어간다. 이 과정이 이익의 감소점에 도달할 때 또는 문제에 관한 보다 나은 이해를 얻게 될 때 새로운 사이클이 시작된다. 또한 이 기술의 수명사이클을 뒤이어 컴퓨터를 설계과정에 응용한다. 컴퓨터 프로그램은 다양한 설계작업을 모방하도록 작성되어 설계 과정을 제시하거나 또는 제안된 설계의 분석에 초점을 둔다.

지식베이스엔지니어링은 설계에 대한 컴퓨터 응용의 새로운 적용 분야를 제시한다. 지식베이스엔지니어링 시스템은 설계 의사결정 과정에 컴퓨터를 도입하여 제안된 설계를 단순히 평가만 하지 않고 설계 생성 과정을 지원한다.

2.1 설계 과정

설계는 보통 물리적 제품을 생산하는 것을 의미한다. 그러나 여기서는 이 용어를 보다 넓은 의미로 사용할 것이다. 예를 들어, 공정산업에 속하는 기업들은 새로운 공정을 설계하며, 이 새로운 공정이 그들의 최종제품이 된다. 또한 설계는 새로운 제품을 생성하는데 필요한 공학적 노력만을 의미할 수도 있다. 나아가 이 용어를 전체적인 생성 및 실행과정(예를 들면, 조립, 제조)을 포함하는 보다 폭넓은 내용으로 사용할 수도 있다.

임의의 설계나 또는 발명은 상당히 유사한 생성으로부터 소멸되기 까지의 수명사이클 진화를 따른다. 새로운 발명은 문제의 보다 나은 이해와 기존 해결안에 대한 새로운 기술 도입에 기초를 두어 이전 발명의 확장으로 나타난다. 다음은 임의의 설계 과정을 보여준다.

단계 1 : 모방(Emulation)

임의의 제품이나 공정은 어떤 제기된 문제에 대한 해결안으로서 시작한다. 이 초기의 해결안은 보통 현재의 해결안 또는 유사한 문제에 대한 해결안의 직접적인 모방에서 비롯된다. 예를 들어, 최초의 비행기는 펄럭이는 날개를 사용하여 나는 새를 모방하려고 시도하였다. 현대의 컴퓨터의 시초인 Babbage의 계산기는 주판의 직접적인 기계적 모방이었으며, 이것은 다시 인간의 손가락 셈을 모방한 것이었다.

단계 2 : 적용(Adaptation)

간혹 모방과정은 Babbage의 계산기와 같은 쓸만한 결과를 산출한다. 보통 그 결과들이 실패하는 경우가 대부분이지만 문제를 규명하고 개선하는데 도움이 된다. 새를 모방한 비행기의 초기 실험은 비행이라는 개념에 대한 보다 나은 이해를 유발하였다. 이후 실현 가능한 해결안이 개발되었음은 우리가 잘 알고 있다.

단계 3 : 확장(Extension)

일단 초기의 문제가 해결되면 사용자는 그 해결안을 적용하기 시작하고, 개선을 위한 대안을 제시한다. 최초의 워드프로세서는 타이프라이터를 모방하였다. 이후에 사용자는 보다 많은 것을 원하였다. 만약 그것이 컴퓨터라면 스펠링을 체크할 수는 없는가? 활자체를 보다 융통성있게 변화시킬 수는 없는가? 등. 이 단계는 기술의 수명사이클에 있어서 가장 긴 시간을 요구한다. 자동차는 수년의 기간에 걸쳐 개발되었다. 그 기술의 발전은 수십년동안 계속되었으며, 아마도 수십년 더 계속될 것이다.

단계 4 : 재설계

어떤 기술도 결국에는 한계에 도달할 것이며, 그 과정이 되풀이된다. 새 해결안이 모방을 위한 또다른 기술 또는 다른 어떤 것의 모방을 근거로 하여 제시된다. 어떤 경우에는 기존 기술의 새로

운 분야가 성장을 시작하는 동안, 기존 기술은 계속해서 조금씩 발전할 것이다. 한 예로서, 헬리콥터와 로켓은 비행에 대한 대안의 해결안으로서 발명되었지만, 고전적인 비행기는 그들의 기술적 확장을 계속하였다.

이와같이 임의의 설계는 사용자 요구를 보다 잘 이해하는데에 기초한 이전 설계의 변형 또는 기초 기술에 관한 지식베이스의 확장으로 나타난다. 이를 효율적으로 유지관리하기 위해 도입된 컴퓨터 기술은 30년이 되지 않았지만, 컴퓨터는 이제 거의 모든 곳에서 찾아볼 수 있다. 예를 들어, 개인용 컴퓨터는 16년이 되지 않았지만, 세계에서 컴퓨터를 사용하는 사람은 8천만명이 넘는다. 오늘날 기업들이 자신의 경쟁력을 유지하기 위해서는 시장 변화 또는 기술 변화에 신속히 대처할 수 있어야 한다.

2.2 설계의 컴퓨터 응용 역사

컴퓨터는 설계분석업무(CAE)에 기초로 적용되어 수치제어 기계 가공을 거쳐 가상현실에까지 이르게 되었다. 이러한 발전 단계를 도식화하면 그림 4를 얻을 수 있다. 그림 4에서 보여지는 것과 같이 모델링 기술은 계속해서 확장되었으며, 특히 사용자의 편의를 고려하여 보다 쉽게 만들고, 보다 사용이 용이하게 만드는 데에는 아직도 개선의 여지가 많다. 그러나 설계 그 자체를 볼 때 어떤 설계를 거의 완벽에 가까운 영상으로 표현하는 모델을 생성할 수 있다. 현재의 기술은 설계 결과의 거의 완벽한 영상을 표현할 수 있다. 그러나 설계 뒤에 감춰진 논리적 근거는 획득할 수가 없다. "설계의도"를 문서화하는 것은 설계에 대한 컴퓨터 응용에 있어서 다음 단계로 나아가는 데에 필수적이다. 설계의도와 그의 관련된 규칙들을 포함할 때 컴퓨터를 설계의사결정 과정에 통합할 수 있는 것이다. 이러한 환경하에서 설계조건이 만족되고 유사한 설계간의 일관성을 추가할 수 있게 된다.

2.3 향후 개선안

CAD/CAM/CAE의 발전은 단순한 제도 및 분석계산을 훨씬 넘어 서 있다. 모델링 도구들은 설계의 실물과 흡사한 영상을 획득한다. 분석도구들은 이동체계의 복잡한 시뮬레이션을 허용한다. NC 프로그램은 몇가지의 간단한 명령들이 주어지면, 완벽에 가까운 부품을 가공할 수 있다. 이들 도구는 설계과정의 필수적인 부분이 되었으며, 또한 원칙들간의 정보 흐름과 같은 내부적 과정은 사용자가 알 필요가 없게 만들었다.

이러한 극적인 개선이 주어진다고 하더라도, CAD/CAM/CAE는 주로 설계과정에 속하는 제한된 수의 업무 자동화일 뿐이다. 그리고 컴퓨터는 설계의 부분들을 자동적으로 생성하는 것이 아니라 거의 독자적으로 다른 파트에 의해 사용되어 설계의 결과만을 제시하는 데에 초점이 맞춰져 있다.

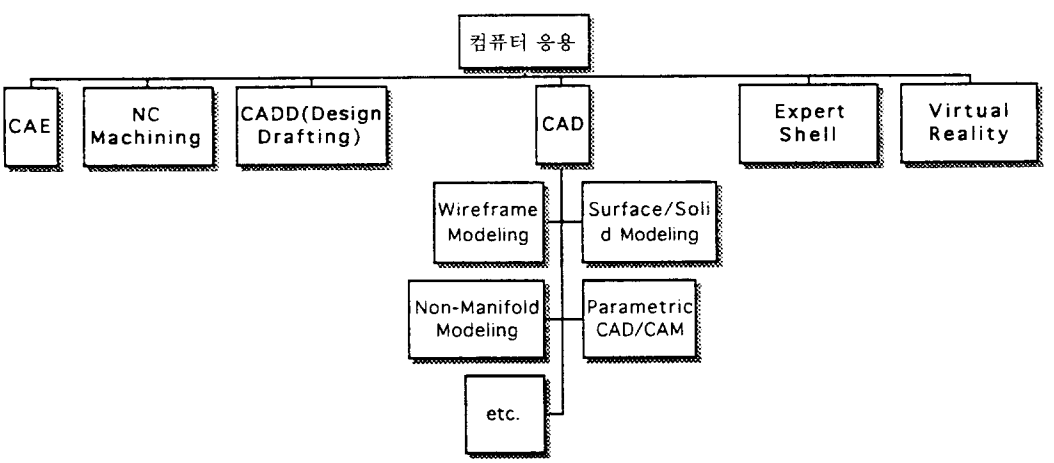

그림 4. 설계에 관련한 컴퓨터 응용 분야

2.3.1 설계 변경 이력의 필요

수세기동안 엔지니어들은 도면을 사용해 왔고, 이제 자신들의 설계를 문서화하기 위해 컴퓨터에서 생성된 도면들을 사용한다. 그러나 각 의사결정을 수행하는 설계자의 마음 속을 제외하고는 어디에도 획득된 '설계의사결정'에 대한 이유들이 존재하지 않는다. 왜 그 사무실 의자는 바퀴가 3개가 아니고 4개인가? 왜 그 받침대에는 구멍들이 존재하는가? 어떤 의사결정들은 공학의 법칙에 근거를 두고, 어떤 의사결정들은 산업법규에 근거를 두고 있다. 또한 의사결정들이 시장필요조건에 근거를 두고 있기도 하다. 모든 제약조건들, 혹은 공학법칙들 조차도 시간에 따라 변화할 것이다. 제품의 수정이나 업그레이드를 위해 설계의도를 재발견하는 데에 소비되는 시간비용은 원가 상승의 요인이 될 수도 있다. 또한 어떤 부분은 오류가 발생하기 쉬운 곳이며, 원래의 의도가 발견되지 않을 수도 있으므로 부적절한 설계변경이 이루어질 위험성도 내포하고 있다.

차체 모델 디자인을 전문으로 하는 자동차공학회사 예를 들면, 기업들이 이 문제를 어떻게 취급하는가에 관한 고전적인 예를 발견한다. 이 회사는 설계에 최신의 컴퓨터 도구들을 사용하고 있었으나 그들이 방금 완성하고 있던 부품들 중의 다소 이상한 형상의 돌기물에 관해 질문하자 그들은 다소 당황해하며 실제로 그 이유를 모른다고 고백하였다. 그러나 이 설계는 이전의 한 설계를 기본으로 하고 있었으며, 그것에 돌기물이 포함되어 있었다. 그들은 그 돌기물이 왜 그곳에 존재하는지를 모르기 때문에 돌기물의 제거를 꺼렸던 것이다.

이전 설계의 부분이기 때문에 설명하기 힘든 형상들이 포함되는 경우는 아주 흔하다. 설계자들은 그것이 왜 거기에 존재하는지 이해하지 않고는 형상들을 제거하기가 쉽지 않다. 이것이 설계 변경 이력을 갖추고자 하는 시스템의 근본적인 이유들이다. 일반적인 기하학적 모델링시스템은 결과가 어떻게 나왔는지에 대한 설계의도, 설계를 정규화

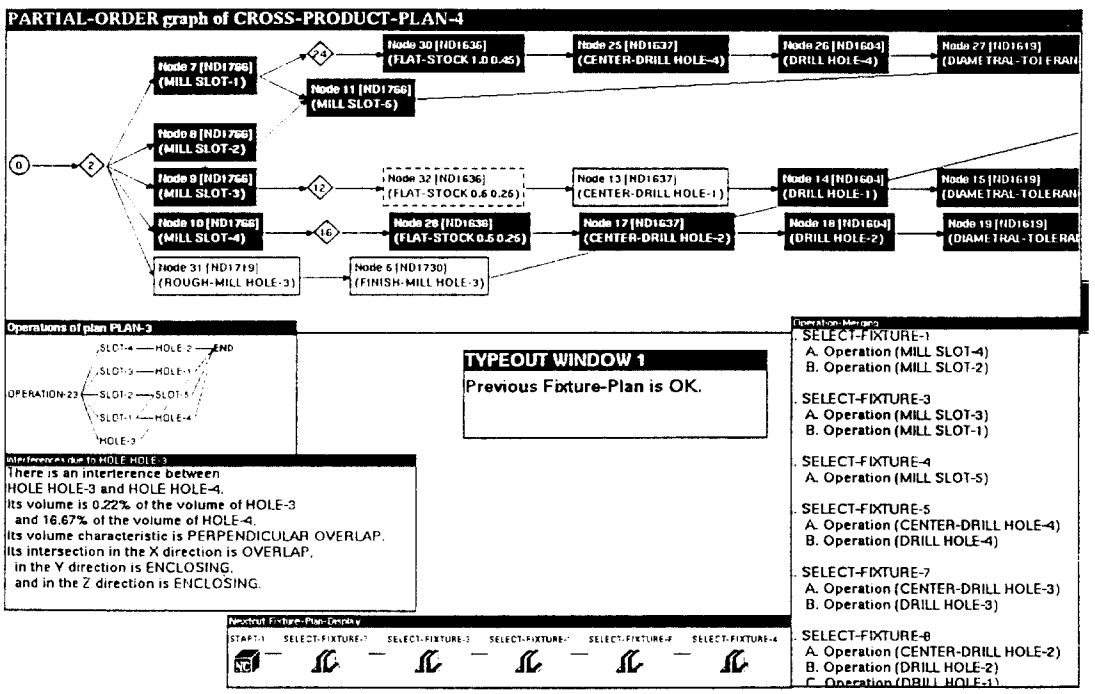

그림 5. 최종 설계를 결정하기 까지는 여러 단계를 가진다. 기본적인 설계에서 두개의 특징형상(HOLE)을 변경하였을 때 이전 정보의 재사용으로 사용자에게 설계 변경의 이력을 제공해 주며 이로 인한 생산 단계의 변경된 공정설계를 제시해 준다. 물론 변경전보다 신속히 새 정보를 제공해 주는데 이는 이전 결과를 재사용했기 때문이다. 그림의 예에서는 가공 허용 오차의 증대로 센터-드릴 공정이 추가되었고 지름의 변경으로 마땅한 드릴공구가 없어 밀링으로 대체 되었다.

하는데 사용된 설계규칙 또는 사용된 임의의 설계 절충조건을 제시함이 없이 설계의 결과만을 제시한다.

2.4 지식베이스엔지니어링

지식베이스엔지니어링은 임의의 기술적인 발전을 위해 앞서 언급한 문데들의 해결 방안으로 제시된다. 현재의 시스템들은 설계과정 중의 한 개인에 의해 수행된 업무들을 모방함으로써 개발되었다. 한 대안으로 설계과정 자체를 모방하는 것을 고려한다.

지식베이스엔지니어링 해결안은 3가지의 주요 구성요소로 구성된다. [1] 지식베이스들의 집합, [2] 부품객체들의 집합, [3] 부품객체들이 제품 또는 공정을 달성하기 위해 결합되는 방법의 개념적 모델 등이 그것이다. 지식베이스엔지니어링 시스템은 보통 기존의 CAD/CAM/CAE 환경과

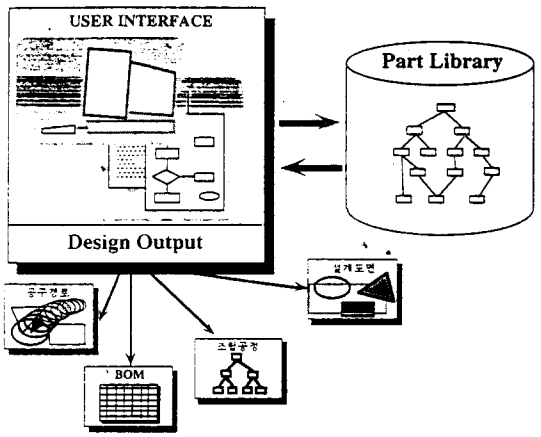

그림 6. 전형적인 지식베이스엔지니어링 시스템의 구성도

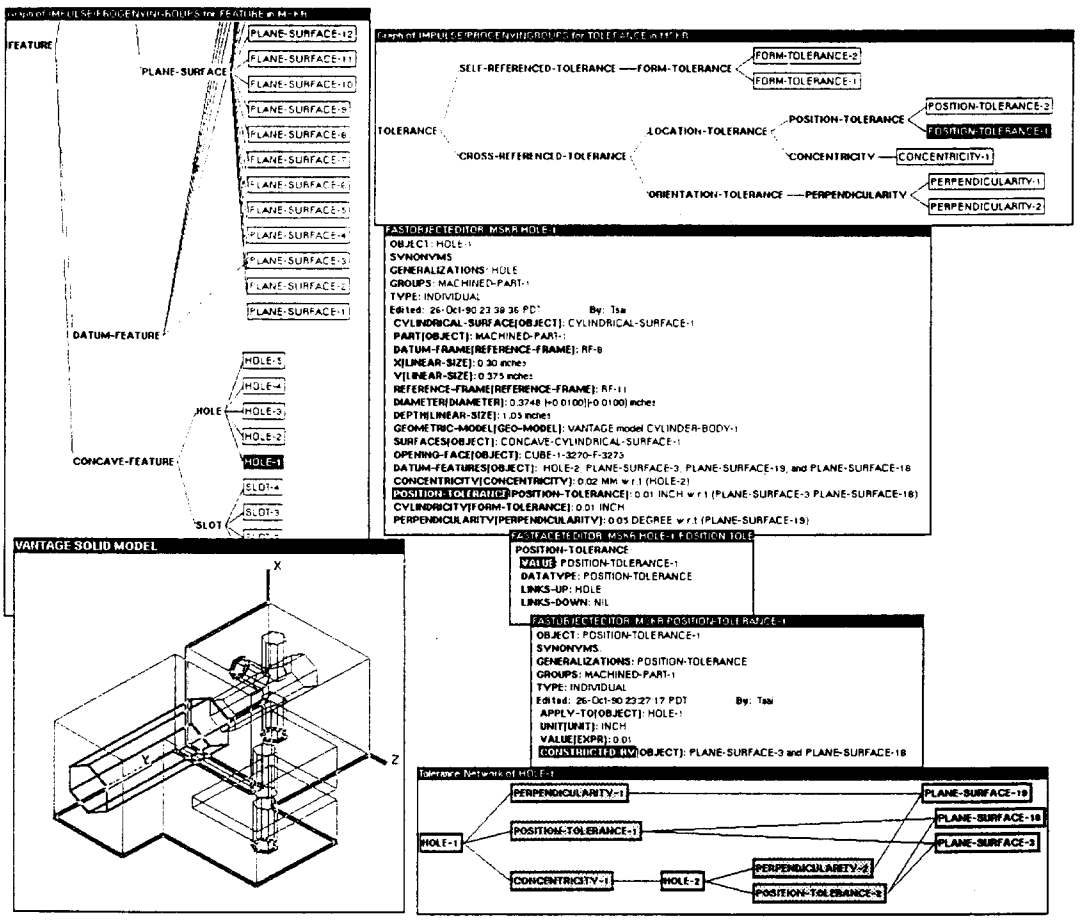

그림 7. 절삭가공용 지식베이스엔지니어링 시스템[NextCut, 이수홍 1991]내에 정의된 부품 객체들의 일반적 계층 구조도

통합되어 설계 결과를 얻고 다시 일반적인 형태로 분해될 수 있게 된다. 주된 차이는 지식베이스엔지니어링 시스템에서는 설계에 영향을 주는 지식베이스가 모든 단계에서 고려된다는 것을 보장한다는 점이다. 이것은 설계를 수행하고 그 결과를 얻는 과정에서 만족해야 할 여러 제약 조건들을 이미 만족했기 때문이다. 이미 얻는 결과를 다시 평가하며 검색하는 필요성을 없앤다. 이는 지식베이스엔지니어링 시스템이 "설계보정"된 결과를 만들어내기 때문이다.

2.4.1 지식베이스

지식베이스는 폭넓은 일반적인 지식으로부터 그 제품 특유의 지식에 이르는 모든 영역을 포함한다. 이 지식은 앞 절에 설명한 바와 같이, 여러 가지 형태로 존재한다. 지식베이스엔지니어링 시스템은 설계과정을 개선하기 위해 이 공동지식베이스에 대한 체계화된 접근을 통해 필요 정보를 축출한다. 지식베이스를 사용함으로써 설계의 수행/평가/수정 루프를 최소화 또는 제거할 수 있다.

2.4.2 부품객체

부품객체는 지식베이스엔지니어링 해결안의 기본골격을 형성한다. 그 명칭이 의미하는 바와 같이, 객체는 흔히 실제 개체(entity)에 해당한다. 예를 들어, 보울트는 기계설계에 보편적으로 사용되는 하나의 객체이다. 부품 객체는 그들 자신을 모델화하는데 필요한 모든 정보를 포함한다.[은닉(Excapsulation)으로 알려져 있는 개념].

객체는 다양한 지능을 가진다. 기본 객체는 단지 그들 자신을 그리는 방법만을 알고 있다. 보다 지능적인 객체는 지식베이스를 사용하여 크기, 형상, 위치, 제조공정 등에 관한 특정의 설계의사결정까지도 포함한다. 어떤 객체는 다른 객체들이 이 설계에 적용될 수 있는 지도 결정한다. 또한 객체는 다른 객체들을 그룹화함으로써 형성할 수 있다. 예를 들어, 조임구(Fastener) 객체는 보울트, 와셔 및 너트 객체들의 그룹핑이라 볼 수 있다. 그 핵심은 이들 객체들이 하나의 제품을 설명하기 위해 "레고블록(Lego Block)"형식으로 서로 결합된다는 것이다.

2.4.3 개념적 제품모델

개념적 제품모델은 특정의 제품이 적절한 부품객체들을 조합함으로써 형성되는 방법을 설명한다. 즉 Part-Subpart의 관계를 보여준다. 이것은 제품의 주요 구성요소와 중요하지 않은 구성요소들 사이의 관계를 설명하기도 한다. 개념적 제품모델은 종종 나무 구조 형태로 표현되며, 제품의 조직구조를 표현한다. 개념모델은 또한 객체들이 자신에게 할당된 업무를 수행하는데 필요한 정보를 획득하는 방법과 다른 객체들에게 제공하기 위해 어떤 정보를 요청받을 것인가를 결정한다. 최종 객체로서의 제품을 고려하는 경우에는, 그 제품을 설계하기 위해 사용자가 제공할 제품사양과 결과로서의 필요조건 등을 결정해야 한다. 이 정보는 이 제품라인에 대해 제품 특유의 사용자 인터페이스를 구축하는데 사용될 것이다.

3. 지식베이스엔지니어링 시스템의 선택, 통합 및 실행

지식베이스엔지니어링 시스템은 기존의 기술을 교체하는 것이 아니라 보완하기 위해 계획되었다. 따라서 이 시스템이 기존의 설계환경에 어떻게 통합될 것인가를 고려하는 것이 중요하다. 지식베이스엔지니어링 시스템의 선택과 그것을 설계 및 제조환경으로 통합하는 것은 분리될 수가 없다. 어떤 의미에서 구축된 시스템의 성공여부는 시스템의 선택에 좌우한다고도 할 수 있다. 또한 이러한 기술의 도입은 고가인만큼 신중을 기해야 하며 최고관리자의 적극적인 지원이 절대적으로 필요하다. 다음의 절들에서는 지식베이스엔지니어링 시스템을 선택할 시 갖추어야 할 조건들을 살펴보기로 한다.

3.1 개방형 시스템 구조(Open System Architecture)

(1) 적응성 : 초기의 지식베이스엔지니어링 시스템은 "블랙박스" 설계시스템을 구축하기 위한 소프트웨어 도구였다. 그 아이디어는 사용자로부터 데이타를 수집하고 필요한 설계과정을 자동적으로 생성하는 독립적인 응용을 구축하려는 것이

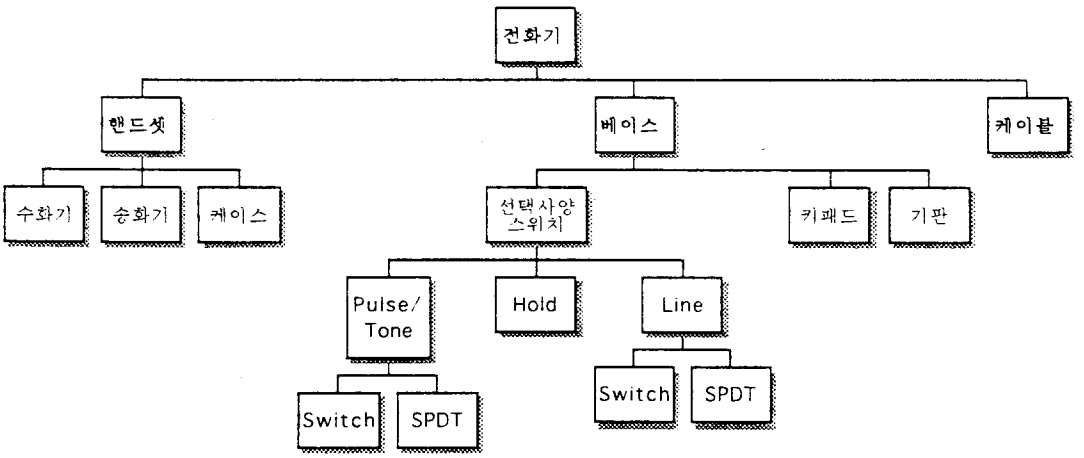

그림 8. 개념적 제품모델을 형성하는 Part-Subpart의 계층적 관계

었다. 이것은 타당성있는 목표인 듯 보이지만, 실제로는 주어진 제품에 대한 모든 단알 규칙 또는 제약 조건을 예상한다는 것이 거의 불가능하다. 또한 설계라는 것이 계속해서 발전하고 변화하는 것이고, 컴퓨터가 인간 설계자만큼 유연하지 않다는 점도 있어, 예상되지 않거나 또는 자연발생적인 변화를 수행하는 데에는 그다지 적합하지 않다.

사용자들은 종종 우리가 예상하지 못하는 제품에 대한 변화를 요구한다. 조그마한 설계 사양 변경에 대해 얼마나 신속히 새로운 결과를 제시해 주느냐가 시스템의 적응성을 가늠한다. 일부 시스템은 설계 변경이 용이하도록 자동화 과정에 대화형식을 포함하는 변동이 잦은 부품(Mutable Part)을 정의하기도 한다. 이렇게 함으로써 시스템내 객체들의 기본정의를 변경시키지 않고 또한 시스템을 재구축하지 않고서도 어떤 객체의 성질이나 또는 서브-어셈블리 부품들을 추가하는 것이 가능하다. 즉 설계자는 시스템 기본 틀을 수정하지 않고 객체의 특정 사례에 대한 성질을 수정하거나 추가할 수도 있다.

(2) 호환성(Portability) : 컴퓨터, 특히 워크스테이션의 발전은 느낄 수 있을 정도로 계속되고 있다. 장래 어떠한 플랫포옴이 가장 비용 효과가 클 지 또는 가장 속도가 빠를 지를 예상하는 것은 불가능하다. 그래서 지식베이스엔지니어링 시스템을 다양한 플랫포옴 상에서 운용가능하게 하는 것이 필수적이다. 또한 임의의 시스템과의 향후 호환가능성을 평가하는 것도 중요하다. 오늘날 다양한 플랫포옴 상에서 호환되는 시스템이 먼 훗날에도 계속해서 호환 가능하리라는 것을 보장하지는 못한다. 업체들간의 표준(예를 들면 X-Window, Motif, UNIX지원 등)을 따라가는 것이 장래의 호환성을 보장하는 최선의 선택이라 하겠다.

(3) 표준화 : 컴퓨터 산업에 대해서는 아주 많은 표준들이 생겨나고 있다. 이들 표준에 기초하여 구축된 시스템은 많은 장점(예, 호환성의 증대, 훈련의 감소, 통합의 용이성)을 제공한다. 대부분의 벤더들은 그들이 지원하거나 또는 지원하려고 계획하는 표준리스트를 제시하여 판매 전략 측면으로 표준화를 강조한다. 그러나 이것은 흑백의 상황이 아니다. 지원되어야 하는 표준들의 최소의 집합이라는 것은 존재하지 않는다. 임의의 특정 표준 지원이 표준화의 명확한 증거 제시가 아니다. 벤더들이 표준의 부분집합만을 지원하는 것은 아주 흔한 일이다.

3.2 통합성(Integration)

지식베이스엔지니어링 시스템은 일반적으로 기존 시스템에 대한 부속물로서 실행될 것이기 때문에 그러한 기존 시스템들과의 통합능력이 중요하다. 표면적으로는 지식베이스엔지니어링 시스템의 모든 것들이 기존 환경으로 쉽게 통합될 것으로 보인다. 그러나 이 통합이 수행되는 방법은 종

종 달성할 수 있는 결과에 중요한 영향을 미칠 것이다. 통합성의 정도를 판단하는 기준으로 다음의 사항들을 고려한다.

(1) 데이타베이스 관리시스템(Data Base Management Systems)과 기하학적 모델러(Geo- metric Modeller)와의 통합

지식베이스엔지니어링 시스템과의 통합이 요구되는 기존 시스템으로는 데이타베이스 관리시스템과 기하학적 모델러를 들 수 있다. 특히 기하학적 모델러와의 통합이 매우 중요한 데, 이는 지식베이스엔지니어링 시스템에서 기하학적 모델들이 다른 CAD 시스템의 Application 프로그램에 의해서도 사용될 수 있도록 보다 대화식 방법으로 기존의 CAD 시스템과 통합될 필요가 있기 때문이다. IGES, DXF 또는 기타의 표준화일 전송형식들은 이 목적에는 적용할 수가 없다. 지식베이스엔지니어링 벤더들은 모두 CAD 시스템 모델러와의 통합을 선언한다. 그렇지만 이러한 통합은 임의의 어떤 상호연결만을 말하는 것일 수도 있다. 이 용어는 거의 의미가 없어질 정도로 과용되고 있다. 그 핵심은 데이타가 CAD 시스템과 어떻게 공유되는 가를 조사하는 것이다.

(2) 객체와 CAD 데이타간의 인터페이스

지식베이스엔지니어링 시스템내의 객체와 CAD 데이타간의 인터페이스를 활용하는 시스템들은 그들간에 데이타를 전송하는 2개의 다른 모델러(CAD 시스템에 있는 것과 지식베이스엔지니어링 시스템에 있는 것)를 가진다. 이 전송을 지식베이스엔지니어링 시스템 사용자는 쉽게 알 수 있다. 그러나 설계과정에 복잡한 곡면모델이나 솔리드모델이 포함된다면, 이들 객체와 CAD 데이타간의 인터페이스에는 상당히 많은 양의 수학적 데이타번역이 포함된다. 하나의 예로서 자동차 차체 판넬이나 플라스틱 몰드부품과 같은 자유형상표면을 포함하고 있는 부품을 고려해 보자. 이들 복잡한 표면을 모델링하는 데에는 여러가지의 기법들이 사용된다. 일부의 시스템들은 NURBS (Non-Uniform Rational B-Spline)으로 알려진 수학적인 형식을 사용하고, 어떤 것들은 Bezier (Higher Order Bezier Polynomials)로 알려진 수학적 형식을 사용한다. 여기서 이들의 상대적인 장점을 가지고 논쟁하지는 않는다. 그들이 서로 다르다는 것을 아는 것만으로 충분하다. NURBS에 의해 설명되는 곡면은 Bezier에 의해 설명되는 곡면으로 번역이 가능하고, 이 역도 마찬가지이다. 그러나 이들 두 곡면은 대부분의 경우 동일하지 않다. 이런 경우 응력을 해석하거나, 어떤 조립과정 혹은 부품을 가공하고자 할 때 매우 중요한 문제로 대두된다.

(3) 실시간(Real Time) 프로그래밍 연결

일부 지식베이스엔지니어링 시스템에 의해 사용되는 또다른 접근방법은 CAD 시스템 모델러와 지식베이스엔지니어링 시스템 사이의 "실시간(Real Time)" 프로그래밍 연결을 구축하는 것이다. 이것은 지식베이스엔지니어링 시스템이 직접 CAD 시스템의 기하학적 추론기관의 기하학적 형상의 생성, 조작 및 관리기능을 호출한다는 것을 의미한다. 이 경우에는, 단지 하나의 모델러만 존재하기 때문에 데이타 변환이 없다. 만약 CAD 시스템이 Bezier 표현을 사용한다면, 지식베이스엔지니어링 시스템은 CAD 시스템의 Bezier 표현을 직접 사용할 것이다. 다른 한편으로, 만약 CAD 시스템이 NURBS 표현을 사용한다면, 지식베이스엔지니어링 시스템도 그것의 NURBS 표현을 직접 사용할 것이다.

이러한 형태의 통합은 산업표준 RPC(Remote Procedure Call) 설비에 의해 가능해진다. 이러한 구체적 통합의 개념을 사용하여 구축된 시스템들만이 지식베이스엔지니어링 시스템과 CAD 시스템 사이의 데이타 무결성을 보장할 수 있다.

(4) 데이타 공유와 전달

실시간(Real Time) 프로그래밍 연결은 단순한 데이타 공유의 개념만은 아니다. 이것은 또한 지식베이스엔지니어링 시스템이 CAD 시스템에서의 기하학적 형상의 생성에 참여할 수 있다는 것을 의미한다. 예를 들어, CAD 시스템 사용자 인터페이스 앞에서 사용자가 CAD 시스템 사용자 인터페이스에서의 메뉴선택을 통해 임의의 표면(예, 비행기 날개)에 대해 기계적 하부구조의 전문가적 지원을 제공하는 임의의 지식베이스엔지니어링 응용 프로그램을 호출할 수 있다. 응용 프로그램이 시작되면 CAD 시스템이 대화식 선택방

법에 의해 추가의 매개변수들을 메뉴 형태로 사용자에게 요구함으로써 문제를 해결해 나간다. 지식베이스엔지니어링 응용 프로그램이 완료되면 최종 결과의 기하학적 형상이 CAD 스크린 상에 나타난다.

(5) 사용자 인터페이스

그래픽 사용자 인터페이스(GUI)의 이점은 대부분의 시스템에 의해 잘 받아들여지며, 또한 제공된다. 최선의 선택은 그 시스템이 수행되는 워크스테이션에 대한 표준(일반적으로 MOTIF)을 고수하는 인터페이스이다. 이것은 인터페이스 기능들이 표준의 방법으로 수행되기 때문에 훈련시간이 최소화된다는 것을 의미한다. 지식베이스엔지니어링 시스템은 일반적으로 사용자가 전문화된 응용 인터페이스를 정의하는 것을 가능하게 한다. 대부분의 지식베이스엔지니어링 시스템은 완전한 그래픽 인터페이스를 쉽게 구축할 수 있는 툴을 제공한다. 판넬, 아이콘(ICON) 등을 스크립트(Script) 언어로 사용하여 정의하는 것은 단조로운 과정이므로 엔지니어들이 직관적으로 느낄 수 있는 부분에 대해 지식베이스엔지니어링 툴들이 이들 인터페이스를 구축하기 위한 대화식 그래픽 도구를 제공한다.(객체 지향형 프로그램 환경이라 불리는 대부분의 시스템이 이와 같은 기능을 제공한다.)

(6) 객체 정의

시스템 사용을 쉽게 하는 또하나의 특기 사항은 부품 또는 객체의 구축 및 정의에서 비롯된다. 지식베이스엔지니어링 시스템의 모든 정보들을 객체로 정의하기 위해서 LISP와 유사한 언어의 형태를 사용한다. 지식베이스엔지니어링 시스템구축에 필요한 대부분의 과정은 이들 객체정의의 구축을 포함한다. 이 과정이 단순할수록 사용자는 지식베이스엔지니어링 시스템을 보다 빨리 개발할 수 있다. 객체정의를 손쉽게 할 수 있는 추가의 도구들이 제공된다면 시스템 개발 생산성은 크게 증가될 수 있다. 일반적으로 지식베이스엔지니어링 시스템은 새로운 객체들을 대화식으로 정의한다. 이러한 객체 정의의 모든 부분들이 메뉴로서 제공되고, 필요에 따라 정의된 속성치들만을 가지고 그래픽 윈도우상에 나타내기도 한다.

3.3 지식베이스엔지니어링 시스템의 구축

지식베이스엔지니어링 시스템의 구축 또는 기타의 컴퓨터를 기초로 한 설계도구들의 구축은 많은 위험 부담을 수반한다. 사용자가 선택하는 시스템은 성공의 정도에 분명하게 영향을 줄 것이지만, 일반적으로 시스템의 고장은 사용된 시스템의 어떤 특정 약점들의 결과라기 보다는 잘못된 실행 접근 방법의 결과이다.

지식베이스엔지니어링 도구들은 설계과정 자체의 조직구조에 영향을 주기 때문에 추가의 위험 부담을 갖는다. 명확하게 정의된 계획없이 시스템 구축을 시작하는 것은 효율적이지 못하다. 또한 사용자는 지식베이스엔지니어링 시스템을 구축하는 과정에서 여러번의 시행 착오를 거치는 과정에 부산물로 시스템 구축 계획도 변경 발전될 것이라는 것을 인식해야 한다.

3.3.1 장기적인 목표의 설정

첫번째 단계는 지식베이스엔지니어링 기술에 대한 장기적인 목표를 규명하는 것이다. 이 기술이 보다 빠른 속도로 보다 고품질의 제품을 생산하기 위해 조직을 재구성할 수 있다는 것을 이해해야 한다. 지식베이스엔지니어링 목표는 사업 목적을 표현해야 한다. 기술적 목적만을 설정하는 실수를 해서는 안된다.

일단 목표들이 설정되면, 사용자는 조직 내의 다른 부서들과 이 목표들을 명확하게 나누어야 한다. 목적을 잘못 이해하는 조직 내의 일부 인원들에 의해 실행이 연기되는 일은 아주 빈번하다. 그리고 어떤 개인들은 새로운 기술의 적용에 대해 위협을 느낀다. 목표 설정을 잘 이해시키면 모든 사람들을 보다 장기적인 목표에 초점을 두게 하여 일의 활력을 불어 넣을 수도 있다.

3.3.2 초기 목표

종종 지식베이스엔지니어링 자동화를 위해 가장 어려운 신제품을 선택하는 경향이 있다. 이것은 아마 적절한 장기목표일지 모르지만, 그 기술의 초기도입을 위한 최선의 선택이 아닐 수도 있다. 새로운 제품 고유의 미지 사항들은 새로운 기술의 미지 사항들과 함께 섞이게 마련이다. 따라서 사용자가 피할 수 없는 장애와 접하게 될 때, 그 출처를 결정하기가 어려워진다. 그러므로 기존

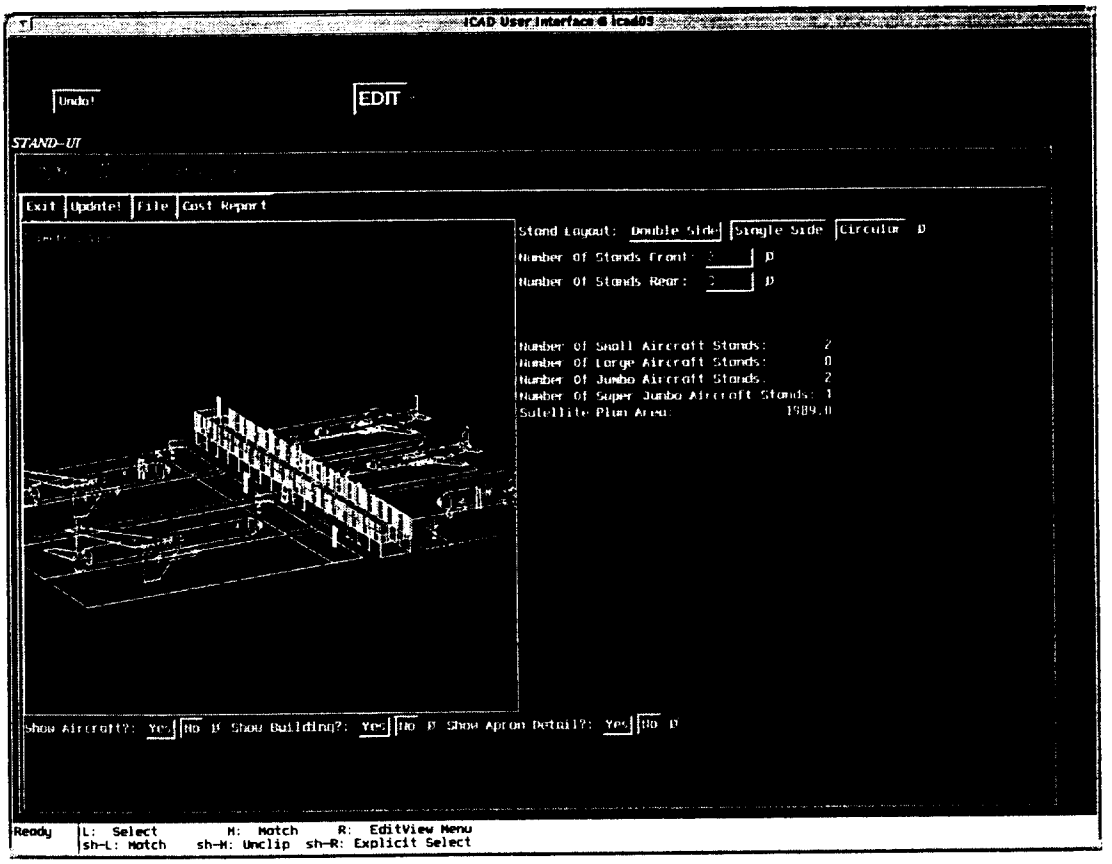

그림 9. 객체 지향형 프로그램 환경이 제공하는 대화식 그래픽 도구를 이용하여 구축한 시스템 레이아웃의 한 예

의 잘 알려져 있는 제품을 선택하는 것이 최선이다.

다음으로, 일의 추진은 이 새로운 기술을 이해하는 데에 초점이 맞추어질 것이다. 초기의 성공을 바탕으로 일을 계속적으로 추진해 나가면 장기적으로 활기를 주며 이러한 초기의 성공은 아주 작은 프로젝트에 대한 것이라고 할지라도, 조직내의 여론 일치를 조장하게 되어 기술개발이 가속화 될 수 있다.

3.3.3 프로젝트 리더의 선택

시스템을 구축하는 프로젝트의 리더를 선택하는 것은 거의 지식베이스엔지니어링 시스템 자체를 선택하는 것만큼이나 중요하다. 가장 성공적인 경우는 내부의 우수 연구 집단이 그 길을 이끌어 가는 경우이다. 기술은 대부분의 공학 및 제조 기술진들에게 새로울 것이다. 새로운 기술을 포용하는 것을 꺼리는 것은 당연한 일이다. 성공적인 리더는 인력 자원 및 판매상과의 섭외로서 작용할 수 있는 사람이다. 공학 및 제조기술진이 자신의 회사내에 자신들을 도울 수 있고 그들의 문제를 접근하는 방법에 제안할 수 있는 누군가가 존재한다는 것을 느낄 때, 그들은 훨씬 쉽게 적극적으로 프로젝트를 진행할 수 있다.

3.3.4 사용의 용이성

사용하기가 쉽다는 것은 말하기가 쉽다는 것이라고 누군가가 말한 적이 있다. 실제로 그것을 사용해 볼 때까지는 어떤 시스템이 얼마나 사용하기가 쉬운 지를 측정하는 것은 어렵다. 시스템을 작동하는 개인들은 전문가이고, 항상 그것을 쉽게 여기기 때문에 벤치마크(Benchmarks) 또는 증명이 설득력이 없다. 사용 용이성에 기여하는 인자들을 고려하여 그들이 지식베이스엔지니어링 시

스템에 포함되는 정도로 전체적인 사용 용이성의 등급을 결정할 수 있다.

4. 결 론

이상으로 지식베이스엔지니어링이 기존의 제품 개발 방법론을 개선하기 위한 지능적인 설계도구로서 컴퓨터를 사용하는 새롭고 흥미로운 대안임을 살펴보았다. 전문가의 설계 및 현장 경험을 조직적이고 체계적으로 수집하여 활용하는 것은 동시공학(Concurrent Engineering)방법론의 실행을 위한 도구로서 매우 중요함을 살펴 보았다. 또한 시스템의 효과적인 구축을 위해 어떠한 것들을 고려해야 하는 지도 살펴보았다. 이러한 일련의 과정을 통해 설계 분야가 미약한 국내 환경에 지식베이스엔지니어링이 하루 속히 정착 되기를 희망한다. 예를 들어 기계류제품의 제조과정을 살펴보면 쉽사리 설계 분야의 취약성을 알아볼 수 있다. 즉 기계류제품의 제조원가중 10% 내외가 설계 비용이지만 설계 단계에서 이미 제조원가의 70% 정도가 결정된다는 사실이다. 즉 요즈음 생산기술에서 많이 시도되는 자동화, 관리기법의 선진화, 전산기이용 제조기술 등 수많은 생산기술상의 노력은 나머지 30% 내외의 원가를 줄이기 위한 노력이다. 지금까지 우리는 제품의 성능개선, 신제품 개발의 관점에서 설계 기술의 중요성을 인식하고 있다. 그러나 본질적에서의 예와같이 자국의 설계 기술 향상에 많은 노력을 기울일 필요가 있다. 국내에서 추진중인 기술 개발 사업들의 대부분도 생산기술상의 나머지 30% 내외의 원가를 줄이기 위한 노력이 주류를 이루는 듯한 감이 없지 않으나, 국제 경쟁력 강화 및 고품질 제품 개발의 효율적인 운용을 위해 설계 기술 개발에도 많은 투자가 이루어졌으면 하는 바램이다.

참 고 문 헌

1. 이수홍, "병행 설계 공정 시스템," **대한기계학회 학술지,** Vol.34, NO.3, March 1994, pp.193-204.
2. 이수홍, "자동차 전장용 에이전트 기반 시스템 연구," **한국 자동차 공학회 논문집,** Volume 1, Number 3, Novemver, 1993, pp.83-94.
3. 이수홍, "동시 공학 설계 생산 시스템," **한국 정밀 공학회,** 10권, 제3호, Sep. 1993, pp.5-20.
4. 이수홍, "설계의 병행 엔지니어링(Concurrent Engineering) 기법(Ⅱ)-케이블 디자인 시스템," **기계와 재료,** Vol.5, No.3, Oct, 1993, pp. 115-130.
5. Breuer, M.A., *Design Automation of Digital Systems,* Englewood Cliffs, N.Y., Prentice-Hall, Chapter 6, 1972.
6. Cutkosky, M.R., and Tenenbaum, J.M., "A Methodology and Computational Framework for Concurrent Product and Process Design," *Mechanism and Machine Theory,* Vol.25, April, 1990, pp.365-381.
7. Park, H.S., Conru, A.B., Cutkosky, M., and Lee, S.H., "An Agent-Based Approach to Concurrent Cable Harness Design," *Artificial Intelligence for Engineering Design, Analysis and Manufacturing(AI EDAM),* NO.8, April, 1994, pp.45-61.
8. Park, H.S., Lee, S.H., and Cutkosky, M., "Computational Support for Concurrent Engineering of Cable Harnesses," *Proceedings of the 1992 ASME International Computers in Engineering Conference,* San Francisco, August, 1992, pp.261-268.
9. Lee, Soo-Hong, Cutkosky, M., and Kambhampati, S., "Incremental & Interactive Geomitric Reasoning for Fixture and Process Planning," Issues in *Design/Manufacture Integration,* 1991, ASME Winter Annual Meeting, Atlanta, Georgia, December 1991, pp.7-13.
10. Shoham, Y. "Agent 0 : A Simple agent language and its interpreter," *Proceedgins Ninth National Conference on Artificial Intelligence,* IEEE, 1991, pp.704-709.
11. Kambhampati, S., Lee, S.H., Cutkosky, M., and Tenenbaum, J.H., "Integrating General Purpose Planners and Specialized Reasoners : Case study of a Hybrid Planning Architecture," *IEEE-Transactions on SYSTEMS, MAN, and Cybernetics,* VOL.23, No.6, November / December, 1993, pp.1503-1518.
12. Sycara, K.P., "Cooperative negotiation in concurrent engineering design," *Computer-Aided Cooperative Product Development,*" Springer-Verlag, 1989, pp.269-297.