韓國經營科學會誌
第23卷 第2號
1998年 6月

수리 모형의 표현과 관리를 위한 다면적 접근법

김종우* · 김형도** · 박성주***

Multi-facetted Approach to Mathematical Model Representation and Management

Jong Woo Kim* · Hyung Do Kim** · Sung Joo Park***

Abstract

One of the essential issues in model systems is how to represent and manipulate mathematical modeling knowledge. As the bases of integrated modeling environments, current modeling frameworks have limitations: lack of facility to coordinate different users perpectives and lack of mechanism to reuse modeling knowledge. In this paper, multi-facetted modeling approach is proposed as a basis for the development of integrated modeling environment which provides facilities for (1) independent management of modeling knowledge from individual models; (2) object-oriented conceptual blackboard concept; (3) multi-facetted modeling; and (4) declarative representation of mathematical knowledge. The proposed multi-facetted approach is illustrated using multicommodity transportation models.

1. 서 론

수리적 모형화 과정은 지식집약적이고 시간이 많이 소비되는 작업이다. 이러한 모형화와 관련된 행위들을 지원하기 위해서 모형화 환경(modeling environment)[16,37]과 모형 관리 시스템 (Model Management System, MMS)[2,4,23,24,29,36]에 대한 연구가 의사결정 지원 시스템 (Decision Support System, DSS) 분야에서 활발히 진행되고 있다. 모형 관리 시스템과 모형화 환경이란 용어는 때로 혼용되어 사용되기도 하는데, 모형 관리 시스템 또는 모형베이스 관리 시스템이란 용어는 데이타베이스 관리 시

* 충남대학교 통계학과
** 데이콤 종합 연구소 멀티미디어통신연구팀
*** 한국과학기술원 테크노경영대학원

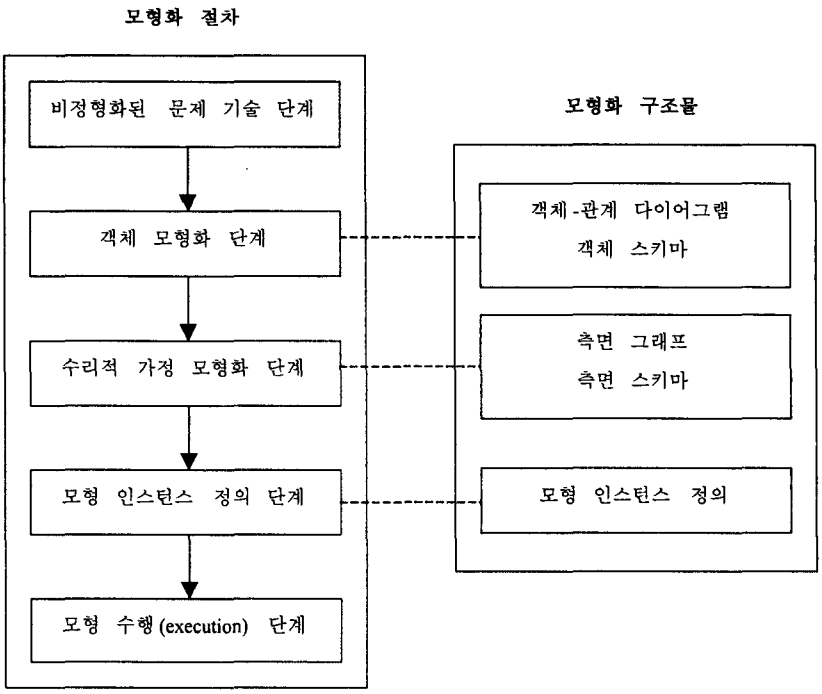
스템에 대응되는 용어로 데이타베이스 관리 시스템의 개념과 기능을 모형 관리에도 적용하고자 하는 의도에서 출발하였다[11,41]. 이에 반해서 모형화 환경은 수리적 모형화와 관련된 일련의 작업들을 포괄적으로 지원하기 위한 컴퓨터 도구들의 집합으로 모형 수명 주기 (model life cycle) 전 과정을 지원하는 환경을 지향하는 용어이다[16]. 모형화 환경이나 모형 관리 시스템을 구현하기 위해서는 수리적 모형을 체계적으로 표현하고 관리하기 위한 개념적 틀이 필요하다. 이러한 틀을 모형화 틀 (modeling framework)이라 하는데, 그 대표적인 예로는 구조적 모형화 (structured modeling)[14], 논리 기반 모형[2,28], 그래프 문법 (graph grammar)[24], 객체지향 모형[23,36], 프레임 (frame) 기반 모형[3,31,32] 등이 있다.

Geoffrion이 제시한 구조적 모형화는 의존도 그래프 개념을 중심으로 한 모형화 틀로서, 광범위한 수리 모형들을 표현할 수 있을 뿐만 아니라 형식성(formality), 수행가능성(executability), 구현가능성(implementability) 등의 특성을 갖는 유용한 틀로 인정받고 있다[14,16]. 다면적 접근법(multi-facetted approach)은 모형화 지식의 재사용과 공유의 두 가지 측면에서 구조적 모형화를 개선하기 위한 시도이다. 수리적 모형화 지식은 여러 가지 다른 유형의 지식을 포함하는데, 이에는 문제 영역의 객체, 관계, 속성에 대한 정의 및 기술, 모형에 포함된 수리적 가정, 모형 관리를 위한 지식들이 포함된다. 하지만 구조적 모형화를 비롯한 대부분의 모형화 틀에서는 수리적 모형화 지식의 관리 단위가 개별 수리 모형으로 되어있다. 이러한 상황에서 새로운 모형을 개발하는 경우, 기존의 모형에 포함된 지식을 재활용하기 위해서는 기존에 작성된 개별 모형들을 하나하나 재검토해야 한다. 또한 기존의 모형들이 서로 다른 도구나 모형화 환경을 활용하여 개발된 경우에는 모형화 지식의 재사용은 더욱 어려워진다. 모형화 환경 내의 모형화 지식을 여러 사용자가 공유해야 할 필요성이 점차 증가하고 있다. 하지만 기존의 모형화 틀에서는 여러 사용자간의 다른 추상화 관점을 지원하고 모순성을 갖는 수리적 가정을 수용하기 위한 고려가 부족하다.

본 논문에서는 모형화 지식의 재사용과 공유를 원활히 지원하기 위한 다면적 접근법을 제시한다. 여기서 다면적 접근법은 다면적 모형 표현과 모형 절차를 포괄하는 의미로 사용하기로 한다. 본 논문의 구성은 아래와 같다. 제 2장에서는 다면적 접근법의 구성과 기본 개념을 소개한다. 제 3장에서는 다면적 접근법에서의 모형화 절차와 모형 표현을 설명한다. 제 4 장에서는 모형 추가 시나리오를 중심으로 모형 추가시의 모형화 절차를 설명한다. 제 5장에서는 다면적 접근법에 기반하여 개발된 프로토타이프 모형화 환경을 소개한다. 제 6장에서는 다면적 접근법의 의의를 제시하고 제 7장에서는 결론을 제시한다.

2. 다면적 접근법의 구성과 기본 개념

다면적 접근법을 구성하는 모형화 구조물(construct)과 모형화 절차는 그림 1과 같다. 다면적 접근법에서의 모형화는 (1) 비정형화된 문제 기술, (2) 객체 모형화, (3) 수리적 가정 모형화, (4) 모형 인스턴스 정의, (5) 모형 수행(execution)의 5단계를 순차적으로 거친다. 모형 표현을 위한 모형화 구조물은 객체 모형화를 위



[그림 1] 모형화 구조물과 모형화 절차

한 객체-관계 다이어그램과 객체 스키마, 수리적 가정 모형화를 위한 측면 그래프와 측면 스키마, 모형 인스턴스 정의를 기술하기 위한 모형 인스턴스 정의를 포함한다.

다면적 접근법에서는 모형화 지식의 재사용성을 증진시키고 모형화 지식의 공유를 지원하기 위해서 다음과 같은 개념들을 포함하고 있다.

(1) 객체 지향 개념 칠판 (object-oriented conceptual blackboard)

다면적 접근법에서는 문제 영역의 구조적 지식 (즉, 문제 영역의 객체, 관계, 및 속성)을 개별 수리 모형과 독립적으로 관리하도록 하였다. 이러한 구조적 지식의 집합을 다면적 접근법에서는 객체 지향 개념 칠판 (object-oriented conceptual blackboard), 또는 개념 칠판이라 부른다. 객체 지향 개념 칠판은 구조적 지식들을 여러 사용자가 공유하도록 함으로써 사용자들간의 용어 사용의 일관성을 유지시키고, 수리적 가정 등 모형화 지식을 접근하는 출발점으로 활용된다.

(2) 다면적 개념 (multi-facetted concept)

수리적 모형화는 문제 영역에 구조적 지식들을 규명하고 이들간의 수리적 가정들을 수립하는 과정으로 이해할 수 있다[44]. 다면적 접근법에서는 측면 (facet) 개념을 제시하여 수리적 가정을 구조적 지식과 구분하여 관리할 수 있도록 한다. 다면적 접근법에서 측면은 객체 지향 개념 칠판의 객체 및 관계에 포함된 속성들의 한 일면 (aspect)으로 정의된다. 측면 개념을

활용하여 동일한 속성에 대한 다른 여러 수리적 가정들을 한 모형화 환경 안에서 체계적으로 관리하고 구조적 지식과 연계하여 사용할 수 있도록 한다.

(3) 단일 인덱스 집합을 활용한 수식 표현

수리적 모형화 언어에서 인덱스 집합 (index set)은 수리적 지식을 대수적으로 표현하는데 유용한 수단이지만[20,40], 수리적 표현에 약한 사용자들에게는 모형을 이해하고 다루는 데 장애 요인이 된다. SML[14,18], GAMS[5], AMPL[13] 등 대부분의 수리적 모형화 언어들에서는 사용자가 임의적으로 인덱스 집합을 정의하여 사용할 수 있도록 허용하고 있는데, 이러한 임의적 인덱스 집합 (artificial index set)을 많이 사용하게 되면, 복잡한 모형의 경우는 모형의 이해성 (understandability)이나 가독성 (readability)을 떨어뜨리게 된다. 다면적 접근법에서는 사용자가 임의적으로 인덱스 집합을 정의하지 못하도록 하고, 객체명을 모형화 환경 내의 단일한 인덱스 집합으로 사용하도록 제한하므로써 모형화 지식의 일관성과 가독성을 높이고자 하였다. 수식을 표현함에 있어서도 선언적 표현 방식을 제공하므로써 수리적 표현에 약한 사용자들도 수리적 지식을 쉽게 이해할 수 있도록 하였다.

3. 모형화 절차와 모형 표현

본 절에서는 다면적 접근법의 모형화 절차와 모형 표현을 전형적인 다상품 운송 모형 (multicommodity transportation model)을 사용하여 설명하기로 한다.

3.1 비정형화된 문제 기술 단계

초기에 문제를 분석하기 위해서는 문제에 대한 비정형화된 기술을 획득해야 한다. 이러한 비정형화된 문제 기술의 획득은 관련 업무 담당자와의 면담, 관련 문서 검토 등을 통해서 가능하다. 일반적으로 이러한 문제 기술에는 문제의 범위, 구성 요소, 가정, 제약 조건, 목적함수에 대한 내용들이 혼재되어 포함된다. 그림 2는 예제 다상품 운송 모형에 대한 문제 기술서의 일부이다.

문제 기술서

A회사는 band, coil, plate를 생산하는 철강회사이다. A사는 3개의 공급지에서 보관 중인 제품을 7개 지역의 수요지 물류 창고로 수송하고자 한다. 3개의 공급지는 각각 3 제품에 대하여 공급 능력의 한계를 가지고 있으며, 7개 지역에 위치한 수요지 물류 창고들도 각 제품별로 최소한 만족시켜야 하는 수요량 요구 조건을 가지고 있다. 또한 운송 경로별로 수송 가능한 운송량의 제한을 가진다.

A회사의 물류부서에서는 전체 운송비를 최소화하도록 운송량을 결정하는 운송 모형을 수립하고자 한다. 운송 단가는 공급지, 수요지, 제품에 의해서 결정되며, 한 공급지에서 한 수요지까지 운송되는 특정 제품의 운송비는 운송단가와 운송량의 곱으로 계산된다. 전체 운송비는 이러한 운송비들의 총합으로 계산된다. 단, 특정 수요지의 특정 제품에 대한 수요량이 100 단위 미만인 경우는 운송 모형에서 제외하기로 한다.

[그림 2] 다상품 운송 모형의 문제 기술서

3.2 객체 모형화 단계

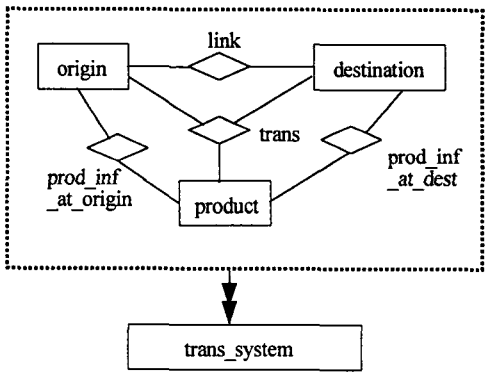
객체 모형화 단계는 문제 내에 포함된 개념적 객체들과 그들간의 관계들을 추출하는 단계이다. 기 정의 되어 있는 객체 지향 개념 칠판

을 참조하여 문제 영역에 관련된 객체, 관계, 속성들을 추출하고, 필요한 경우 모형 개발자는 새로운 객체, 관계 또는 속성을 개념 칠판에 추가한다. 예제인 다상품 운송 모형의 문제 기술서로부터 product (제품), origin (공급지), destination (수요지), trans_system (운송 시스템)의 객체를 개념 칠판에 추가한다. 객체간의 관계로는 link (공급지와 수요지간의 연결), 특정 공급지에서 특정 수요지로의 특정 제품의 운송에 대한 정보를 담은 trans (운송) 등이 정의된다.

개념 칠판은 객체-관계 다이어그램 (object-relationship diagram)과 객체 스키마 (object schema)에 표현된다. 객체-관계 다이어그램은 확장된 개체-관계 다이어그램 (Extended Entity-Relationship Diagram)[6,12]과 유사한 형태로 구조적 지식의 개략적 관점을 제공한다. 객체, 관계에 대한 자세한 정보들은 객체 스키마에 기술된다. 객체-관계 다이어그램의 구성요소들은 객체, IsA, IsAggregateOf, 사용자 정의 관계이다. 객체와 사용자 정의 관계는 각각 직사각형과 마름모꼴로 표시한다. IsA와 IsAggregateOf는 각각 화살표와 머리가 둘인 화살표로 표시한다. 그림 3은 예제 모형의 객체-관계 다이어그램이다. 점선으로 된 사각형은 사각형 내의 객체 및 관계들과 trans_system 객체와는 IsAggregateOf 관계를 가짐을 의미한다.

객체 스키마는 객체명, 객체가 가지는 관계, 속성의 세 부분으로 이루어진다. 사용자 정의 관계인 경우, 객체 스키마의 "객체가 가지는 관계" 부분에는 IsA 정보와 관계에 참여하는 객체명과 참여 객체의 역할 (role)이 기술된다. 예를 들어, 그림 4에서 사용자 정의 관계인 link 객체의 객체 스키마에서 origin_part와 dest_part는 각각 origin, destination 객체의 참여 역할을 의미한다. 사용자 정의 관계가 아닌 객체의 경우는 IsA, IsAggregateOf 정보가 기술된다. 단, IsAggregateOf 정보는 해당사항이 없는 경우 생략이 가능하다.

사용자 정의 관계가 아닌 객체의 속성 중에 객체의 키 (primary key) 역할을 하는 속성에는 "〈ID〉"라는 기호를 속성에 표시해준다. 사용자 정의 관계가 아닌 객체 중에 객체 인스턴스의 갯수가 항상 1을 넘지않는 경우는 키를 명세할 필요가 없다. 예를 들어, trans_system은 문제 영역의 다른 객체들을 포괄하는 객체로서, 하나의 인스턴스만을 가지는 객체로 정의되었기 때문에 키에 대한 명세가 없다. 사용자 정의 관계의 경우는 키를 명세하지 않아도 되는 데, 이 경우 사용자 정의 관계의 키는 참여 객체 키의 합집합으로 정의된다. 객체 스키마의 BNF는 부록 1과 같다.



[그림 3] 다상품 운송 모형의 객체-관계 다이어그램

객체 모형화 단계에서는 객체, 관계의 추출 및 추가와 함께 객체 인스턴스들의 입력 또는 수정 작업도 함께 수행된다. 모형 개발자가 객

OBJECT origin	
IsA(OBJECT)	
name	: STRING 〈ID〉

OBJECT destination	
IsA(OBJECT)	
name	: STRING 〈ID〉

OBJECT product	
IsA(OBJECT)	
name	: STRING 〈ID〉

OBJECT trans_system	
IsA(OBJECT) IsAggregateOf (origin, destination, product, prod_inf_at_origin, prod_inf_at_dest, link, trans)	
total_cost	: REAL

OBJECT prod_inf_at_origin	
IsA(RELATIONSHIP) origin_part(origin) prod_part(product)	
supply supply_balance	: REAL : BOOLEAN

OBJECT prod_inf_at_dest	
IsA(RELATIONSHIP) dest_part(destination) prood_part(product)	
demand demand_balance	: REAL : BOOLEAN

OBJECT link	
IsA(RELATIONSHIP) origin_part(origin) dest_part(destination)	
limit link_limit	: REAL : BOOLEAN

OBJECT trans	
IsA(RELATIONSHIP) origin_part(origin) dest_part(destination) prod_part(product)	
trans_cost trans_qty unit_price	: REAL : REAL : REAL

[그림 4] 다상품 운송 모형의 객체 스키마

체 및 관계를 모형화 환경에 추가하면, 객체 및 관계의 인스턴스를 넣을 수 있는 테이블이 모형화 환경에 의해서 자동적으로 생성된다.

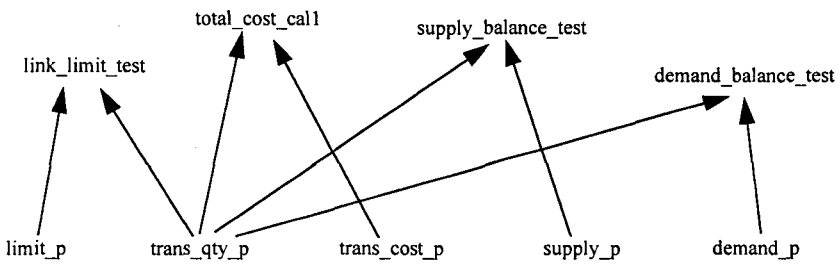
3.3 수리적 가정 모형화 단계

수리적 가정 모형화 단계에서는 앞 단계에서 정의된 객체의 속성들간의 수리적 가정들을 측면 (facet) 개념을 활용하여 정의한다. 모형 개발자는 사용하고자 하는 수리적 가정이 기정의되어있는 경우에는 이를 재사용하고, 필요한 경우에는 속성간의 새로운 가정을 추가로 정의한다. 다면적 접근법에서 수리적 가정 모형화는 측면들간의 정의적 의존성 (definitional dependency) [14]와 계산 규칙 (computational rule)을 정의하는 것을 의미한다. 측면간의 정의적 의존성은 측면 그래프 (facet graph) 형태로 표현되며, 계산 규칙을 포함한 측면 각각의 상세 정보는 측면 스키마 (facet schema)에 기술된다.

측면 그래프는 측면들로 구성된 비순환 그래프 (acyclic graph)이다. 측면들간의 정의적 의존성이 방향있는 아크 (directed arc)로 표현된다. 각 아크의 머리 노드 (head node)를 호출 측면 (calling facet), 꼬리 노드 (tail node)를 호출된 측면 (called facet)이라 부른다. 측면은 본원 측면 (primitive facet)과 유도 측면 (derived facet)으로 나누어진다. 본원 측면은 측면 그래프의 근원 노드 (root node)에 위치한 측면이고, 유도 측면은 측면 그래프의 중간 노드 (intermediate node)나 하위 노드 (leaf node)에 위치한 측면이다.

예제인 다상품 운송 모형의 측면 그래프는 그림 5와 같다. 측면 그래프 multi__prod__trans__model__1은 9개의 측면으로 구성되는데, 총 운송비에 대한 수리적 가정을 담고 있는 total__cost__call과 공급지 능력 제약 조건, 수요 요구 조건, 운송경로별 운송량 제한 제약 조건을 각각 담고 있는 supply__balance__test, demand__balance__test, link__limit__test 등을 포함한다. 그림 5에서 total__cost__call과 trans__qty__p, trans__cost__p간의 아크들은 측면 trans__qty__p, trans__cost__p 값을 사용하여 total__cost__call을 계산함을 의미한다.

각 측면의 자세한 정보들은 측면 스키마 (facet schema)에 기술된다. 측면 F가 객체 O의 속성 P의 한 측면으로 정의되었을 때, O, P를 각각 측면 F의 근원 객체 (origin object), 근원 속성 (origin property)이라 하고, 측면 F는 객체 O의 속성 P에서 근원한다고 부른다. 측면



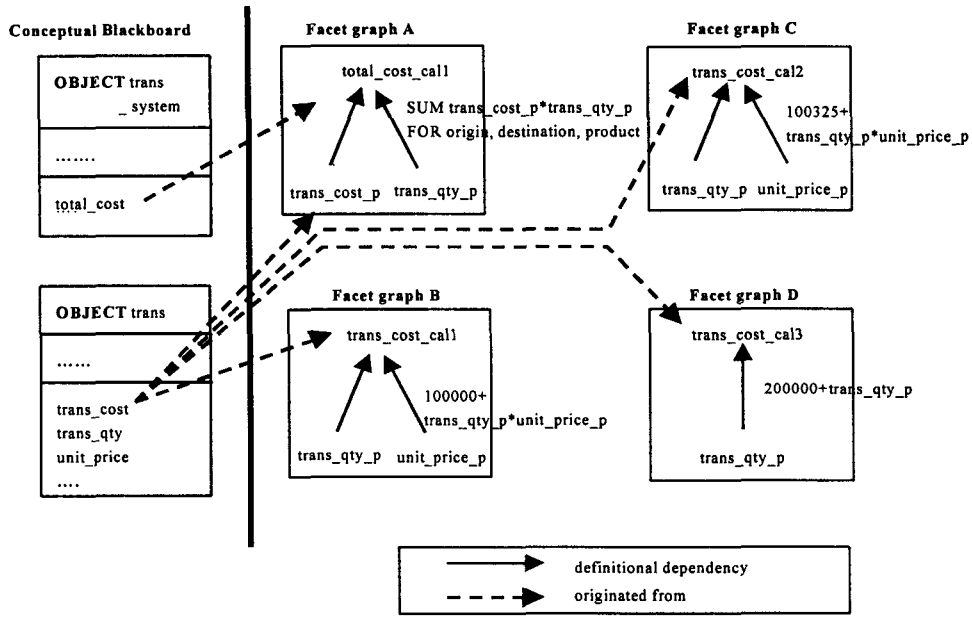
[그림 5] 측면 그래프 "multi__prod__trans__model__1"

스키마에는 측면명, 근원 객체명, 근원 속성명, 측면의 유형 (본원 측면인지 유도 측면인지 여부)에 대한 정보가 포함된다. 유도 측면인 경우에는 호출된 측면명의 집합과 계산 규칙이 추가된다. 측면 스키마의 BNF는 부록 2와 같다. 예제인 다상품 운송 모형의 측면 그래프에 포함된 측면 total_cost_cal1은 trans_system 객체의 total_cost 속성에서 근원한 측면이고, 측면 trans_cost_p, trans_qty_p는 각각 trans 객체의 속성 trans_cost와 trans_qty에서 근원한 측면이다. 그림 5에 포함된 측면들의 측면 스키마는 부록 3과 같다.

다면적 접근법의 측면 개념은 문제 영역 내의 속성들에 대한 다른 관점들을 수용하고, 속성들을 모형에 따라 입력 변수 또는 출력 변수로 다르게 활용할 수 있는 유연성을 제공한다. 예를 들어, 그림 6에서는 trans 객체의 trans_cost 속성에서 근원한 4개의 측면의 예를 보여주고 있다. 측면 그래프 A에서는 trans_cost로부터 근원한 측면 trans_cost_p가 입력 변수로 사용되고 있으나, 측면 그래프 B, C, D에서는 trans_cost에서 근원한 측면 trans_cost_cal1, trans_cost_cal2, trans_cost_cal3가 출력 변수로 사용되고 있다. 측면 그래프 A에서 trans_cost_p는 총 운송비인 total_cost_cal1을 계산하기 위해서 사용되고 있다. 하지만 측면 그래프 B, C, D는 모두 특정 공급지에서 특정 수요지로의 특정 제품의 운송 비용 (trans_cost)을 계산하기 위한 모형이다. 측면 그래프 B와 C에서는 모두 운송량 (trans_qty_p)과 운송 단가 (unit_price_p)를 입력 변수로 운송비용 (trans_cost_cal1, trans_cost_cal2)을 계산하고 있다. 하지만 모형 개발자의 추상화 관점에 따라서 다르게 계산 규칙이 정의되어 있다. 즉 측면 그래프 B에는 고정비를 100000 단위로 가정하였으나, 측면 그래프 C에서는 고정비를 100325 단위로 보다 정밀하게 추정하고 있다. 측면 그래프 D는 운송 비용 (tran_cost_cal3)을 계산하는데, 단지 운송량 (trans_qty_p)만을 입력 변수로 사용하고 있다. 다면적 접근법에서는 이와 같이 어떤 속성 값을 결정하기 위해 사용되는 속성들의 집합이나, 결정을 위해 사용되는 계산 규칙의 형태가 모형의 사용 목적, 모형의 추상화 정도 (abstraction level), 모형 개발자의 관점에 따라서 다르게 정의할 수 있고, 또한 상이하게 정의된 측면들을 개념 칠판과 연결하여 관리할 수 있도록 한다.

계산 규칙을 기술하는 데 있어서는 개념 칠판의 객체명을 모든 계산 규칙에 공통으로 사용하는 인덱스 집합으로 사용하여 수리적 지식을 표현한다. 임의의 측면이 사용자 정의 관계가 아닌 객체로부터 근원한 경우, 그 측면의 인덱스 집합은 해당 측면의 근원 객체가 된다. 임의의 측면이 사용자 정의 관계로부터 근원한 경우, 인덱스 집합은 사용자 정의 관계에 참여한 객체들의 곱집합 (Cartesian Product)이 된다. 예를 들어, 그림 5에서 trans_cost_p는 trans 객체를 근원 객체로 가지고, trans 객체는 origin, destination, product간의 사용자 정의 관계이므로, trans_cost_p 측면의 인덱스 집합은 origin×destination×product가 된다. 하나의 객체간의 사용자 정의 관계인 경우는 역할명을 이용하여 참여 객체를 구분한다.

다면적 접근법에서는 기본 연산자 집합을 제공하여 계산 규칙을 이들 기본 연산자들의 합성함수 (composite function) 형태로 표현하도록 한다[25]. 예를 들어, 다상품 운송 모형에서 총 운송비용 계산규칙의 대수적 표현 (1)은 다면적



[그림 6] 측면의 예

접근법에서는 (2)의 형태로 표현된다. (2)식에서 trans__cost__p, trans__qty__p의 인덱스 집합은 origin×destination×product이고, (2)식은 두 개의 기본 연산자, '*'와 'SUM ... FOR'가 차례로 적용된 형태이다.

$$\sum_{for\,all\,i,j,k} c_{ijk} \times q_{ijk}$$

$$where\ i = origin,\ j = destination,\ k = product,$$
$$c_{ijk} = trans\_cost\_p,\ and\ q_{ijk} = trans\_qty\_p \qquad (1)$$

SUM trans__cost__p*trans__qty__p FOR origin, destination, product (2)

3.4 모형 인스턴스 정의 단계

모형 인스턴스 정의 단계는 앞에서 정의된 측면 그래프를 바탕으로, 어떤 해법자 (solver)와 객체 인스턴스를 사용하여 문제의 해를 생성할 지에 대해 정의하는 단계이다. 모형 인스턴스 정의 (model instance definition)는 모형 인스턴스 정의명, 측면 그래프명, 해법자에 대한 기술, 선택 규칙 (selection rule)으로 이루어진다. 해법자에 대한 기술은 문제의 해를 구하기 위해 사용할 해법자명과 해당 해법자의 수행을 위해 필요한 정보들에 대한 할당이 포함한다. 예를 들어, LP 해법자 수행을 위해 필요한 정보에는 목적 함수, 의사결정 변수, 제약식이 되는 측면들의 집합과 목적 함수를 극대화 할지 극소화 할지 여부가 포함된다. 모형 수행에 참여하는 객체는 모형 인스턴스 정의에 기술된 측면 그래프의 측면들이 근원한 객체들이 된다. 선택 규칙은 모형 수행에 참여할 객체 인스턴스의 범위를 SQL 형태로 지정하게 된다. SQL 형

태로 명세되는 않은 객체는 모든 객체 인스턴스가 모형 수행에 참여하게 된다. 그림 7은 다상품 운송 모형에서 demand가 100이상인 destination과 prod_inf_at_dest의 객체 인스턴스만이 모형 인스턴스에 포함되도록 표현하고 있다. 모형 인스턴스 정의의 BNF는 부록 4와 같다.

다면적 접근법에서는 모형 인스턴스 정의와 모형 인스턴스를 구분하고 있다. 모형 인스턴스 정의는 객체 인스턴스의 상태에 따라서 다른 모형 인스턴스를 생성할 수 있다. 따라서, 모형 인스턴스 정의는 객체 인스턴스의 상태가 계속적으로 변화하는 실시간 모형화 환경에서 유용하게 사용될 수 있다.

3.5 모형 수행 단계

모형 인스턴스 정의를 수행하도록 모형화 환경에 명령함으로써 모형 인스턴스가 생성되고 관련된 해법자를 수행시킨다. 모형 인스턴스 생성을 위해서는 모형화 환경이 자동적으로 모형 인스턴스 정의의 내용을 해석하여 객체 인스턴스 데이타베이스로부터 필요한 객체 인스턴스들을 추출하여 해법자가 처리가능한 형태의 모형 인스턴스를 생성한다. 해법자가 수행한 결과는 사용자에게 출력되거나 객체 인스턴스들에 반영된다. 모형화 환경에서의 상세한 모형 수행 절차는 아래와 같다.

[STEP 1] SQL 형태로 기술된 선택 규칙의 정보를 사용하여, 필요한 객체 인스턴스의 존재 여부를 점검하고 모형 수행에 필요한 객체 인스턴스를 추출하여 모형 인스턴스를 생성한다.

[STEP 2] 해법자 기술을 사용하여, 해법자를 선정하고 모형 인스턴스가 해법자가 풀 수 있는 조건들을 만족하는지를 검증한다.

[STEP 3] 모형 인스턴스를 해법자가 처리 가

```
MODEL INSTANCE DEFINITION LP_multi_prod_trans_model_ins
        Facet Graph        : multi_prod_trans_model_1
        Solver             : LP
                             (Objective_Function = {total_cost_call},
                              Decision_Variable = {trans_qty_p},
                              Constraint = { link_limit _test, supply_balance_test,
                                             demand_balance_test },
                              MIN_OR_MAX = MIN)
        Selection Rule     :
                SELECT name, demand_balance, demand
                FROM destination, prod_inf_at_dest
                WHERE prod_inf_at_dest.dest_part = destination.name
                AND demand >= 100
```

[그림 7] 모형 인스턴스 정의 "LP_multi_prod_trans_model_ins"

능한 형태로 변환한다.

[STEP 4] 해법자를 수행한다.

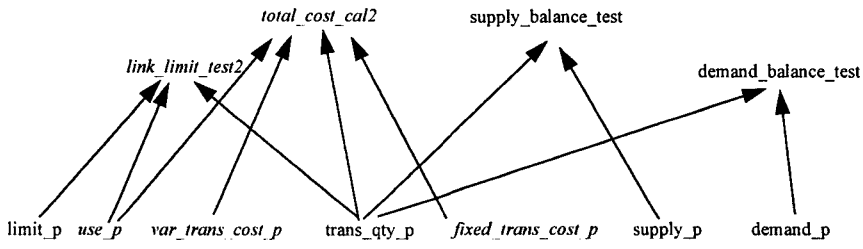
[STEP 5] 해법자가 수행한 결과를 사용자에게 출력하고, 필요한 정보들은 객체 인스턴스에 반영한다.

4. 모형 추가 시나리오

다면적 접근법에서 모형화 지식의 재사용성을 보이기 위해서, 모형 추가의 시나리오를 소개하도록 한다. 예제의 A회사 물류부서에서는 앞에서 소개한 다상품 운송 모형을 변형하여 새로운 다상품 운송 모형을 모형화 환경에 추가하고자 한다. 새로운 다상품 운송 모형에서는 운송 비용을 고정비와 변동비를 포함하는 형태로 수식화하고자 한다. 다시 말해서, origin O와 destination D사이에 운송량이 존재하는 경우에만 운송비에 고정비를 포함하고 운송량에 따라 변동비가 늘어나는 형태로 모형을 개발하고자 한다. 이러한 경우에 정수 계획법의 이진 변수(binary variable)를 사용하여 모형화가 가능하다[10].

위와 같은 모형을 추가하는 경우, 먼저 위의 내용들을 중심으로 문제 기술서를 작성하고, 객체 모형화 단계를 수행한다. 기존에 작성된 개념 칠판을 검토한 결과 추가되어야 할 객체나 관계는 없고 기정의된 객체, 관계들을 그대로 재사용할 수 있다. 하지만 고정비와 변동비를 반영하기 위해서 3개의 속성의 추가가 필요하다. 추가되는 속성은 객체 trans에 변동비를 나타내는 var_trans_cost와 객체 link에 고정비를 나타내는 fixed_trans_cost, 이진 변수인 use이다. 속성 use의 값은 운송량이 0보다 큰 link 객체 인스턴스에 대해서는 1이 되고, 그렇지 않은 경우는 0이 된다.

다음 단계는 수리적 가정 모형화 단계이다. 수리적 가정 모형화 단계에서 작성되는 측면 그래프는 고정비와 변동비를 고려한 형태로 그림 8과 같다. 그림에서 이탤릭체로 된 측면들이 추가된 측면들이고 나머지는 앞에서 정의된 측면들이 재사용되었다. 그림 8에서 추가된 측면 use_p는 객체 link의 새로 추가된 속성 use로부터 근원한 본원 측면이다. total_cost_cal2는 앞의 total_cost_call과 동일하게 trans_system 객체의 total_cost 속성에서 근원한 측면이나, 호출된 측면들도 다르고 계산 규칙도 다르게 정의되었다. 추가된 facet에 대한 자세한 내용은 부록 5와 같다. 그림 9는 측면 그래프 multi_prod_trans_model_2와 정수 계획법 해법자를 활용하도록 선언한 모형 인스턴스 정



[그림 8] 측면 그래프 "multi_prod_trans_model_2"

```
MODEL INSTANCE DEFINITION IP_multi_prod_trans_model_ins
        Facet Graph      : multi_prod_trans_model_2
        Solver           : IP
                           (Objective_Function = {total_cost_cal2},
                            Decision_Variable = {trans_qty_p},
                            Binary_Variable = {use_p},
                                    Constraint = { link_limit_test2,
                                    supply_balance_test,
                                        demand_balance_test },
                            MIN_OR_MAX = MIN)
        Selection Rule   :
                SELECT name, demand_balance, demand
                FROM destination, prod_inf_at_dest
                WHERE prod_inf_at_dest.dest_part = destination.name
                AND prod_inf_at_dest >= 100
```

[그림 9] 모형 인스턴스 정의 "IP_multi_prod_trans_model_ins"

의이다.

5. 프로토타이프 모형화 환경

5.1 개요

본 절에서는 다면적 접근법을 지원하기 위하여 개발된 프로토타이프 모형화 환경인 HIME (Hypertext-based Integrated Modeling Environment)[1]에 대해 소개하기로 한다. HIME은 Microsoft Windows™에서 작동하는 하이퍼텍스트 저작도구인 Toolbook™[45]과 내장 스크립 언어인 OpenScript™[46]를 사용하여 개발되었다. HIME에서는 하이퍼텍스트의 비선형 링크 (nonliner link)[9]를 사용하여 개념 칠판, 측면 스키마, 측면 그래프, 모형 인스턴스 정의 내의 관련 개념들간의 동적인 링크를 제공한다. 예를 들어, 측면 스키마에 기술되어 있는 근원 객체명, 근원 속성명, 호출된 측면명 중에 선택하여 더블 클릭하면 해당 개념으로 이동하도록 링크를 시스템이 자동적으로 생성하고 관리한다. HIME은 여러 종류의 Book (매캔토시의 HyperCard™에서 Card에 해당한다)으로 이루어진다. HIME은 객체 스키마 Book, 객체 계층도 Book, 객체 관계 다이어그램 Book, 객체 인스턴스 Book, 측면 그래프 Book, 측면 스키마 Book, 모형 인스턴스 정의 Book, 계산규칙 Book, 선택 규칙 Book으로 이루어진다. 계산 규칙 Book과 선택 규칙 Book은 보조적인 Book으로 측면 스키마 Book이나 모형 인스턴스 정의 Book에서 각각 계산 규칙과 SQL 형태의 선택 규칙의 입력을 도와준다. 이러한 보조적 Book은 모형 개발자가 정확한 문법을 기억하지 않고도 쉽게 입력할 수 있도록 도와주고 문법적, 어의적(semantic) 오류들을 입력 시에 점검하여 모형화 지식의 일관성과 정확성을 증진시킨다. HIME은 통합된 모형화 환경을 지향한다. 즉

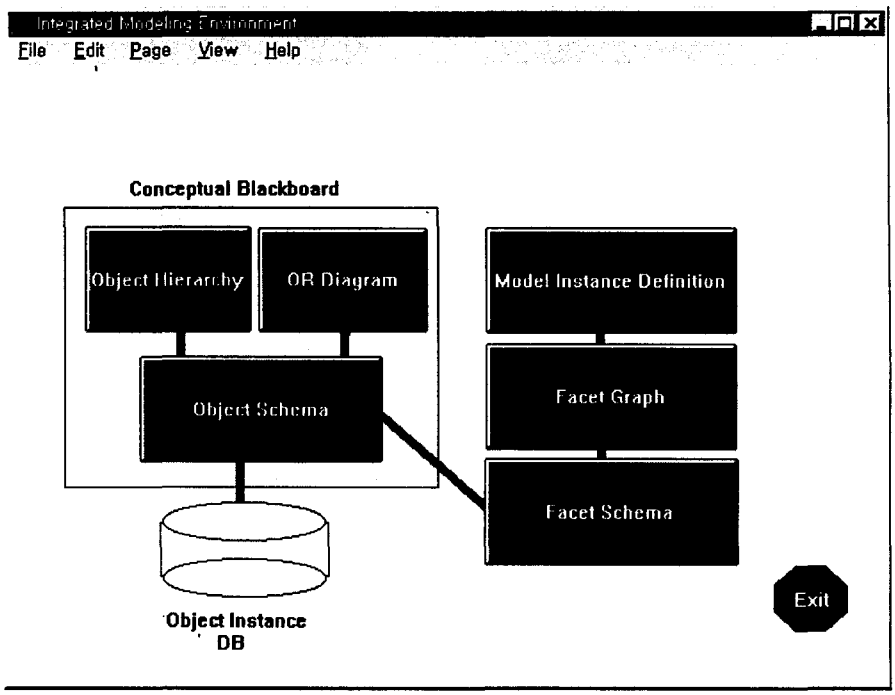
모형 수행과 관련된 객체 인스턴스의 관리나 모형 수행 등의 모형화 관련 행위들을 별도의 소프트웨어를 활용하지 않고 HIME 내에서 처리할 수 있도록 하였다. 예를 들어, 객체 인스턴스는 관계형 데이타베이스 내에 저장되지만 객체 인스턴스의 생성, 삭제, 수정 작업은 모두 객체 인스턴스 Book에서 이루어진다. 그림 10은 HIME의 초기 화면이다.

5.2 모형 명세 기능

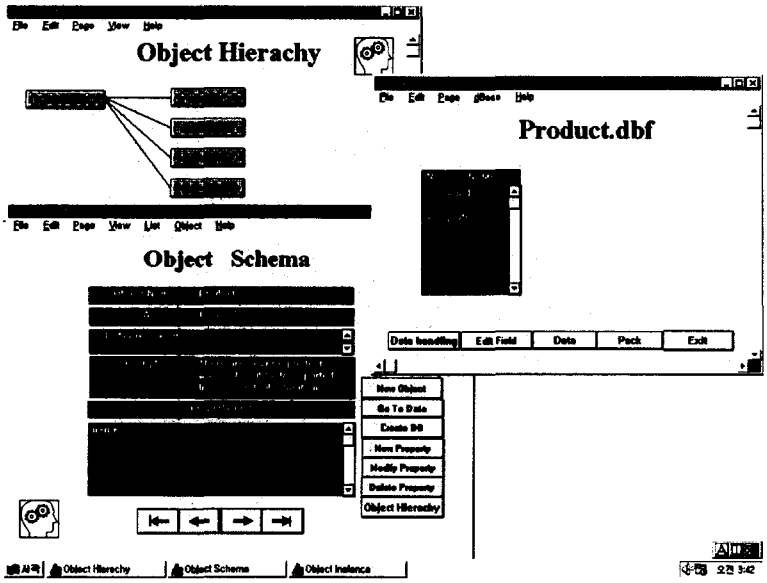
HIME의 주요 기능은 크게 (1) 모형 명세 (specification) 지원과 (2) 모형 수행 기능으로 구분할 수 있다. 모형 명세 지원 기능은 모형 개발자가 객체, 객체 인스턴스, 측면, 측면 그래프, 모형 인스턴스 정의 등을 입력, 수정, 삭제하도록 지원하는 기능이다.

(1) 객체와 객체 인스턴스

그림 11에서는 개념 칠판과 관련된 세 개의 윈도우를 보여준다. 가장 뒷편에 존재하는 윈도우는 객체 계층도 Book으로서 사용자로 하여금 개념 칠판에서 필요한 객체를 추출하는 기본 윈도우 역할을 한다. 사용자가 하나의 객체를 선택해서 더블클릭하면 해당 객체에 대한 객체 스키마 Book이 나타난다. 사용자가 객체를 생성하면 자동적으로 객체 인스턴스의 저장을 위한 테이블이 생성된다. 사용자가 객체 스키마 Book에서 "Go To Data"버튼을 누르면 객체 인스턴스 Book이 나타난다. 객체 인스턴스 Book에서 모형 개발자는 객체 인스턴스의 추가, 삭제, 수정 등 모든 자료 관리 작업을 수행할 수 있다.



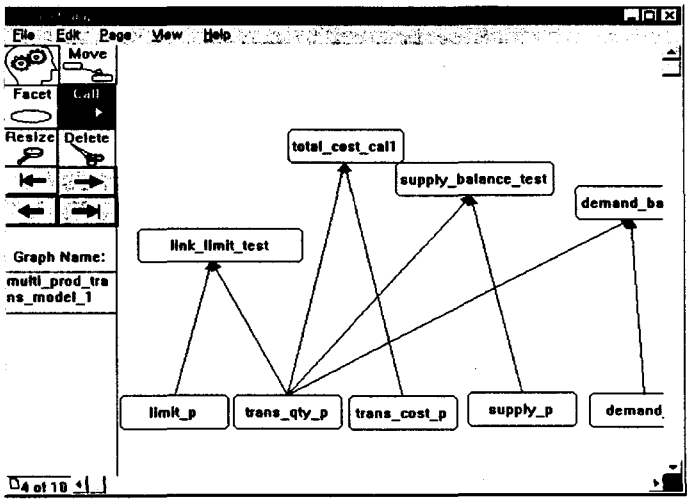
[그림 10] HIME의 초기 화면



[그림 11] 객체와 객체 인스턴스 명세를 위한 화면

(2) 측면 그래프와 측면 스키마

측면 그래프 Book은 그래픽 모형화 도구이다. 사용자는 측면 그래프 Book을 사용해서 측면 그래프를 생성하거나 전후로 이동하면서 측면 그래프들을 검토할 수 있다. 측면 그래프 Book에서 측면을 선택해서 더블 클릭하면 측면 스키마 Book이 나타나고 해당 측면의 측면 스키마의 내용을 보여준다. 측면 그래프 Book에서 측면 간의 정의적 의존성을 설정하면, 이 정보들은 측면 스키마 Book의 호출된 측면 정보에 반영된다. 그림 12는 그림 5의 측면 그래프를 편집하는 화면이다.



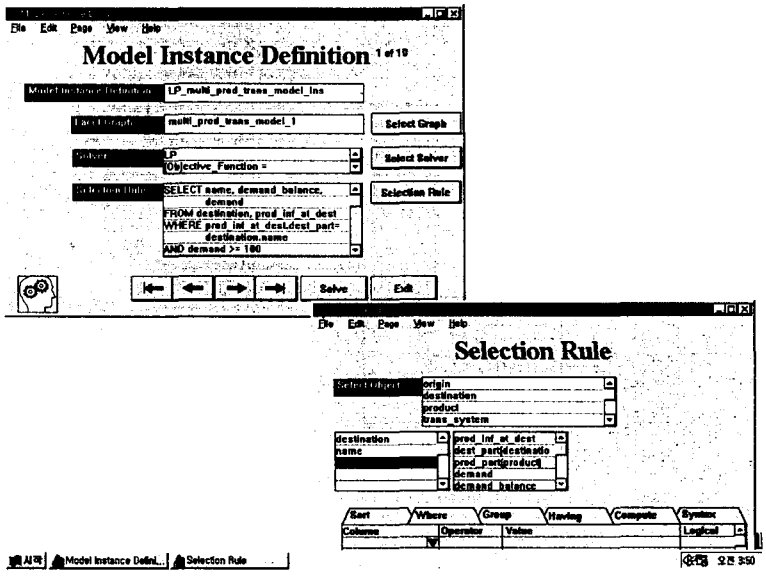
[그림 12] 측면 그래프 Book

(3) 모형 인스턴스 정의

그림 13은 모형 인스턴스 정의를 위한 모형 인스턴스 정의 Book을 보여준다. 모형 인스턴스 정의 Book에서, 측면 그래프의 선택을 위해서는 "Select Graph" 버튼을 누르면 현재 모형화 환경 내의 측면 그래프 리스트와 각각에 대한 요약 정보가 제공되어 측면 그래프의 선택을 도와준다. 해법자에 대한 기술을 위해서 "Select Solver"버튼을 누르면 모형화 환경에서 제공하는 해법자 리스트가 나타나고 그 중 하나를 선택하면 해당 해법자 수행을 위해 필요한 정보를 입력 받기 위한 다이얼로그 박스가 나타난다. 선택 규칙의 입력을 위해 제공되는 선택 규칙 Book은 PowerBuilder™[47]등의 4GL (4 Generation Language)에서와 같이 선택 규칙의 SQL 문장을 가시적 (visual)으로 입력할 수 있도록 설계되었다. 모형 인스턴스 정의 Book에서 "Selection Rule" 버튼을 누르면 선택 규칙 Book이 그림 13과 같이 나타난다. 선택 규칙 Book이 나타날 때 해당 모형 인스턴스 정의의 측면 그래프와 관련된 객체들을 시스템이 자동으로 추출하여 나열해준다. 또한 모형 개발자가 모형 인스턴스 생성에 참여를 제한하고자 하는 객체를 선택하면, 해당 객체의 속성들이 나열되어서 모형 개발자가 관련된 객체와 속성들을 다시 찾아보는 수고를 줄여준다.

5.3 모형 수행 기능

모형 수행은 모형 인스턴스 정의 Book에서 "Solve" 버튼을 눌러서 시작한다. HIME에서는 필요에 따라서 해법자를 모형화 환경에 추가가 가능하도록 설계하였다[7]. 즉 새로운 해법자가 획득된 경우, 새로운 해법자명을 등록하고 "해법자 입력 양식 명세 (Solver Input Format Specification)"와 "해법자 출력 조작 명세 (Solver Output Handling Specification)"를 작성하므로써 새로운 해법자를 HIME 내에서 사용할 수 있도록 하였다. HIME에서 확장 가능



[그림 13] 모형 인스턴스 정의 Book과 선택 규칙 Book

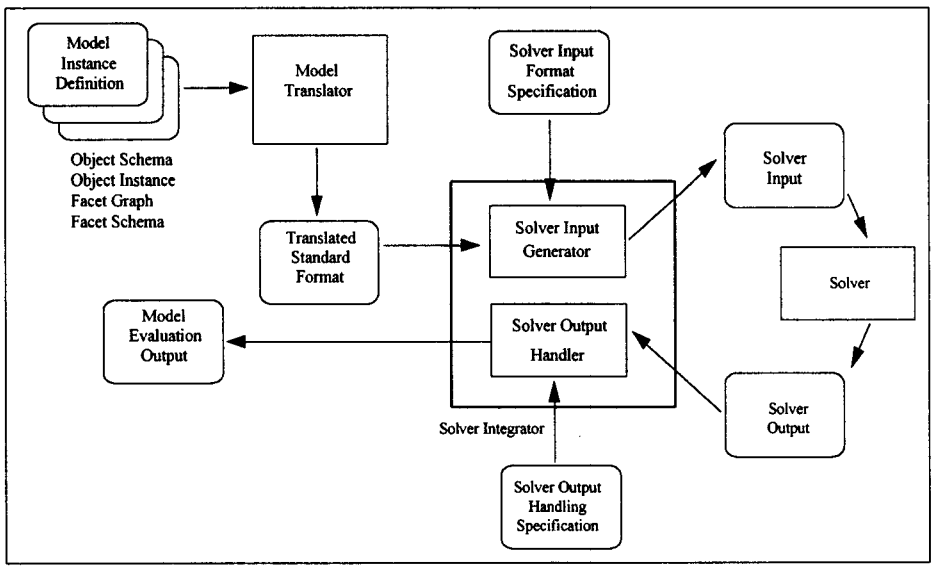
한 해법자 관리를 위한 접근 방법을 도식적으로 나타내면 그림 14와 같다. 해법자의 확장이 가능하도록 하기 위해서 모형 인스턴스 표준 양식(Model Instance Standard Format), 해법자 입력 양식 명세 언어, 해법자 출력 조작 명세 언어가 설계되었다[7]. 모형 인스턴스 표준 양식은 해법자에 독립적인 형태로 3.5절의 모형 수행의 4 단계 중 첫번째 단계에서 생성된다. 모형 인스턴스 표준 양식에 맞추어 생성된 모형 인스턴스와 해당 해법자의 해법자 입력 양식 명세를 사용하여 해법자가 처리가능한 형태의 모형 인스턴스가 생성된다. 모형 수행 후 결과는 해당 해법자의 해법자 출력 조작 명세를 참조하여 객체 인스턴스에 반영된다.

6. 다면적 접근법의 의의

다면적 접근법이 갖는 의의를 모형 재사용, 모형 공유, 모형 통합의 관점에서 정리하면 다음과 같다.

6.1 모형 재사용 (model reuse)

모형 재사용의 어려움은 이론적으로는 모형화 지식을 표현하는 틀들의 이질성[16]에, 실제적으로는 모형 개발 환경의 이질성[10]에 기인한다. RMT[29]에서는 모형화 지식의 재사용을 위해서 여러 다른 모형화 시스템 내의 지식들을 역공학 방법 (reverse engineering)을 활용하여 객체지향의 틀로 변환하여 관리하는 방안을 제시하고 있다. 반면에, 다면적 접근법에서는 객체지향 개념 칠판을 중심으로 개별 모형에 독립적인 지식 관리가 가능한 지식 표현 틀을 제공하므로써 모형 재사용을 도모하고 있다. 구조적 지식, 수리적 가정을 개별 모형으로부터 독립적으로 관리함으로써 모형화 지식을 목적에 따라 쉽게 재구성할 수 있도록 한다. 모형을 추가하는 경우에는 기존의 모형화된 개념 칠판과 측면들을 검토하여 필요한 내용들을 추출하고, 추가적으로 필요한 지식만을 추가하는 모형화 절차



[그림 14] 확장가능한 해법자 관리

를 제공함으로써, 모형화 지식의 재사용을 지원한다.

6.2 모형 공유 (model sharing)

일반적으로 기존의 모형화 틀 및 소프트웨어들이 단일 사용자 (single user) 중심으로 개발되어 왔으나 최근의 그룹 공동 작업을 강조하는 추세에 따라 모형화 지식을 개인적 차원을 넘어서, 조직적 차원에서 관리해야 할 필요성이 제기되고 있다[8,36,38,42,43]. 모형 공유를 위해서는 지식의 다양한 추상화 수준의 지원, 다른 관점의 수용, 지식의 문제 중심적 표현 등이 필요하다. 이러한 의미에서 다면적 접근법에서 제공하는 구조적 지식과 수리적 지식의 구분, 측면 개념, 수리적 지식의 선언적 표현 등이 모형 공유에 기여한다. 다면적 접근법의 측면 개념은 여러 다른 모형 개발자가 정의한 수리적 가정들을 공유하며 모형화 작업을 수행할 수 있도록 한다. 수리적 지식을 표현함에 있어서 단일 인덱스 집합의 활용이나 선언적 표현은 수학적 지식이 부족한 모형화 환경 사용자들도 다른 모형 개발자가 작성한 수리적 지식을 가능하면 쉽게 이해할 수 있도록 하기 위한 배려이다. 모형 공유와 관련된 기술적 문제 중에 하나는 동의어 (synonym), 동음이의어 (homonym), 별칭 (alias)의 처리이다[43]. 다면적 접근법의 개념 칠판은 모형화 환경 사용자간의 공동 용어 사전 역할을 함으로써 부분적으로 동의어, 동음이의어의 문제를 해결할 수 있다. 또한 개념 칠판은 여러 사용자가 기술한 수리적 모형들에 쉽게 접근할 수 있도록 도와준다.

6.3 모형 통합 (model integration)

모형 관리 시스템에서의 모형 통합은 두 가지 다른 의미, 깊은 통합 (deep integration)과 기능적 통합 (functional integration)으로 사용된다[21]. 깊은 통합은 두 개 이상의 모형을 결합하여 하나의 새로운 모형을 만드는 것으로, 새로운 모형은 기존의 모형 표현방식으로 표현되는 형태이다[15,34,35]. 기능적 통합은 새로운 모형이 동일한 표현방식으로 표현되지 않고 모형들간의 계산 순서, 입출력 연결 관계 등을 표현하는 것으로 모형 연결 언어 (model interconnection language), 또는 모형 기술 언어 (model description language) 의 형태로 제시되었다[11,36].

다면적 접근법에서는 모형의 깊은 통합에 대한 고려만이 되어 있다. 다면적 접근법에서 깊은 통합은 두개의 측면 그래프를 결합하여 하나의 새로운 측면 그래프를 생성하는 것이다. 다면적 접근법에서의 깊은 통합 절차는 (1) 결합될 두 측면 그래프에서 합쳐지는 노드 역할을 하는 측면의 규명, (2) 합쳐지는 노드들 쌍에 대해서 두 측면 중 어떤 측면이 새로운 측면 그래프에 남아야 하는지 결정, (3) 생성된 측면 그래프에 포함된 측면들의 계산 규칙을 검토하여, 계산 규칙간의 단위 (unit of measure)가 일관성을 유지하고 있는지를 검토하는 절차로 구성된다. 모형의 기능적 통합에 대한 부분은 다면적 접근법의 확장된 틀인 통합 모형화 틀 (Integrated Modeling Framework, IMF)[25,38]에서 사건/처리과정 망 (event/process net)의 형태로 기술된다.

7. 결 론

본 논문에서는 수리적 모형화 지식의 재사용성을 증진시키고 여러 사용자간의 공유가 가능하도록 표현하고 관리하기 위한 다면적 접근법을 제시하였다. 다면적 접근법은 구조적 모형화(structured modeling)의 의존도 그래프 개념을 기반으로 하고 있다. 다면적 접근법에서는 객체지향 개념 칠판을 활용하여 문제 영역의 객체, 관계, 속성 등의 구조적 지식을 모형화 환경 내의 공통의 개념 칠판으로 활용하고, 수리적 가정들은 측면 (facet) 개념을 활용하여 측면 그래프 형태로 표현하고 관리하도록 하였다. 다면적 접근법의 개념들을 구체화하기 위해서 하이퍼텍스트 저작 도구를 사용하여 프로토타이프 모형화 환경인 HIME (Hypertext-based Integrated Modeling Environment)을 개발하였다.

다면적 접근법과 관련된 향후 연구 과제는 다음과 같다. 첫째, 현재까지는 다면적 접근법의 유용성 및 의의가 실증적으로 검증된 상태는 아니다. 실증적 검증을 위해서는 인지 과학적 실험 등을 통해 기존의 다른 모형화 틀과 비교해서 모형 표현과 관리 측면에서의 장단점을 분석 평가하는 것이 필요하다. 두번째로 다면적 접근법에서는 수식을 표현하기 위한 계산 규칙의 문법을 설계함에 있어서 임의적 인덱스 집합을 활용하지 않고 객체명을 인덱스 집합으로 활용하도록 제한하고 있는데, 이러한 표현 기법으로 다양한 형태의 수식들을 표현할 수 있는지에 대한 검증이 필요하다. 또한 수리적 지식의 표현에 있어서는 선언적 표현 방식을 채택하고 있는데, 대수적 표현에 익숙한 사용자를 위해서는 대수적 표현으로 사용자가 인터페이스할 수 있도록, 모형화 환경이 자동적으로 수식을 대수적 표현과 선언적 표현간의 상호 변환하는 기능이 필요하다. 마지막으로, HIME을 실제적으로 활용할 수 있는 형태로 보완하고, 분산 환경에서 모형에 대한 공동작업 (CSCW, Computer Supported Cooperative Work)을 지원할 수 있도록 확장하는 것이 필요하다.

참 고 문 헌

[1] 박성주, 김형도, 김종우, "A Hypertext-based Integrated Modeling Environment (HIME)," 한국 경영과학회/대한 산업공학회 '93 춘계 공동 학술 대회, 1993.

[2] Bhagava, H.K. and S.O.Kimbrough, "On Embedded Languages for Model Management," *Decision Support Systems*, Vol.10, No.3(1993), pp.277-299.

[3] Binbasioglu, M and M. Jarke, "Domain Specific DSS Tools for Knowledge-based Model Building," *Decision Support Systems*, Vol.2, No.3(1986) pp.213-223.

[4] Blanning, R.W., "A Relational Framework for Join Implementation in Model Management Systems," *Decision Support Systems*, Vol.1, No.1, January (1985).

[5] Brooke, A.D., D.Kendrick and A.Meeraus, *GAMS: A Users Guide*, *Scientific Press*, Redwood City, CA, 1988.

[6] Chen, P., "The Entity-Relationship Model-Toward a Unified View of Data," *Transac-*

tion of Data Systems, Vol.1, No.1(1976), pp. 9-36.

[7] Cho, K.H., An Approach to Solver Integration for Modeling Environment, *MS Thesis*, KAIST, 1994.

[8] Choobineh, J., "SQLMP: A Data Sublangauge for Representation and Formulation of Linear Mathematical Models," *ORSA Journal of Computing*, Vol.3, No.4(1991) pp. 358-375.

[9] Conklin, J., Hypertext: An Introduction and Survey, Computer, Sept. (1987) pp.17-41.

[10] Cragg, P.B. and M.King, "Spreadsheet Modeling Abuse: An Opportunity for OR?," *Journal of Operational Research Society*, Vol.44, No.8(1993), pp.743-752.

[11] Dolk, D.R. and J.E.Kottemann, "Model Integration and a Theory of Models," *Decision Support Systems*, Vol.9(1993), pp.51-63.

[12] Elmasri, R. and S.B.Navathe, *Fundamentals of Database Systems*, Behjamin/Cummings, CA, 1991.

[13] Fourer, R., D.M.Gay and B.W.Kernighan, *AMPL : A Modeling Language for Mathematical Programming*, The Scientific Press, 1993.

[14] Geoffrion, A.M., "An Introduction of Structured Modeling," *Management Science*, Vol.33, No.5(1987), pp.547-588.

[15] Geoffrion, A.M., "Reusing Structured Models via Model Integration," *Proceedings of the Twenty-Two Annual Hawaii International Conference on System Sciences*, IEEE Computer Society, 1989, pp.601-611.

[16] Geoffrion, A.M., "Computer-based Modeling Environments," *European Journal of Operational Research*, Vol.41(1989), pp.33-43.

[17] Geoffrion, A.M., "FW/SM : A Prototype Structured Modeling Environment," *Management Science*, Vol.37, No.12(1991), pp. 1513-1538.

[18] Geoffrion, A.M., "SML : A Model Definition Language for Structured Modeling: Level 1 and 2," *Operations Research*, Vol. 40, No.1(1992), pp.38-57.

[19] Geoffrion, A.M., "SML : A Model Definition Language for Structured Modeling: Level 3 and 4," *Operations Research*, Vol. 40, No.1(1992), pp.58-75.

[20] Geoffrion, A.M., "Indexing in Modeling Languages for Mathematical Programming," *Management Science*, Vol.38, No.3, March(1992), pp.325-344.

[21] Geoffrion, A. M., "Structured Modeling: Survey and Future Research Directions," *ORSA CSTS Newsletter*, Vol.15, No.1, Spring, (1994), pp.11-19.

[22] Hudge, J. G., *Object-oriented Databases*, Prentice Hall, 1991.

[23] Huh, S.Y, "Modelbase Construction with Object-Oriented Constructs," *Decision Sciences*, Vol.24, No.2(1993), pp.409-434.

[24] Jones, C.V., "An Introduction to Graph-based Modeling Systems, Part I: Over-

view," *ORSA Journal of Computing*, Vol. 2, No.2(1990), pp.180-206.

[25] Kim, J.W., A Modeling Framework for Integrating Decision Models and Information Systems, *Ph.D. Dissertation*, KAIST, 1995.

[26] Klein, G., "Developing Model String for Model Management," *Journal of Management Information Systems*, Vol.3, No.2 (1986), pp.94-110.

[27] Kottemann, J.E. and D.R.Dolk, "Model Integration and Modeling Languages : A Process Perspective," *Information Systems Research*, Vol.3, No.1(1992), pp.1-16.

[28] Krishnan, R., "PDM: A Knowledge-based Tool for Model Construction," *Decision Support Systems*, Vol.7, No.4(1991), pp.301-314.

[29] Kwon, O.B. and S.J.Park, "RMT: A Modeling Support System for Model Reuse," *Decision Support Systems*, Vol.16, No.2(1996), pp.131-154.

[30] Lazimy, R., "A Deductive Approach for Problem Representation and Modeling Support: Conceptual Schema and Object-Oriented Models," *Proceedings of the Twenty-Four Annual Hawaii International Conference on System Sciences*, IEEE Computer Society, 1991, pp.291-305.

[31] Lee, J.K. and M.Y.Kim, "Knowledge-assisted Optimization Model Formulation: UNIK-OPT, *Decision Support Systems*, Vol.13, No.2(1995), pp.111-132.

[32] Lee, J.S. "Structure Frame Based Model Management System", *Unpublished Doctoral Dissertation*, University of Penn., 1989.

[33] Lenard, M. L., "Representing Models as Data," *Journal of Management Information Systems*, Vol.2, No.4(1986), pp.36-48.

[34] Liang, T. P., "Reasoning for Automated Model Integration," *Applied Artificial Intelligence*, Vol.4(1990), pp.337-358.

[35] Liang, T. P. and B.R.Konsynski, "Modeling by Analogy-Use of Analogical Reasoning in Model Management Systems," *Decision Support Systems*, Vol.9, No.1(1993), pp.113 -125.

[36] Muhanna, W.A., "An Object-Oriented Framework for Model Management and DSS Development," *Decision Support Systems*, Vol.9, No.1(1993), pp.217-229.

[37] Park, S. J. and H.D.Kim, "Constraint-Based Metaview Approach for Modeling Environment Generation," *Decision Support Systems*, Vol.9, No.4(1993), pp.325-248.

[38] Park, S.J., J.W.Kim, and H.D.Kim, "IMF: A Modeling Framework for Integrating Decision Models and Information Systems," *Working Paper*, KAIST, 1996.

[39] Park, S.J., J.W.Kim, and H.W.Kang, Heuristic Knowledge Representation of Production Scheduling: An Integrated Modeling Approach, *Expert Systems with Applications*, Vol.10, No.3(1996), pp.325-339.

[40] Ramirez, R.G. and E.Lin, "Subscript-free Indexing in a Mathematical Programming Language," *Proceedings of the Twenty-Sixth Annual Hawaii International Confe-*

rence on System Sciences, IEEE Computer Society, 1993, pp.424-433.

[41] Sprague, Jr.R.H., E.D.Carlson, *Building Effective Decision Support Systems*, Prentice Hall, NJ, 1982.

[42] Stohr, E.A. and B.R.Konsynski, *Information Systems and Decision Processes*, IEEE Computer Society Press, 1992.

[43] Tung, L., R.G.Ramirez, R.D.St.Louis, "Model Integration in an Object-Oriented Model management System," *Proceedings of the Twenty-Four Annual Hawaii International Conference on System Sciences*, IEEE Computer Society, 1991, pp.284-290.

[44] Turban, E. *Decision Support and Expert Systems(2nd Ed.)*, Maxwell Macmillan International Editions, 1990.

[45] __, *ToolBook User Manual*, Asymetric™, 1994.

[46] __, OpenScript Reference Manual, Asymetric™, 1994.

[47] __, PowerBuilder User Manual, PowerSoft™, 1993.

부록 1. 객체 스키마의 BNF

ObjSch = **OBJECT** <ObjName>
IsA (SupObj)
[(IsAggregateOf (<PartObjName1> {, <PartObjName2>}*))|
{<RoleName> (<ParticiObjName>)}+]
{PropSpec}*

SupObj= [OBJECT|RELATIONSHIP|<SupObjName>]

PropSpec= <PropName>:DataType (<ID>)

DataType= [STRING|INT|REAL|BOOLEAN|SetValue]

SetValue= { <EleName1> {, <EleName2>}* }

Legend

=	definition of nonterminal token
...	reserved token
<...>	value-bearing
{...}+	one or more iterations of contents limited by the braces
{...}*	zero or more iterations of contents limited by the braces
[...]	alterntive items separated by vertical bars
(...)	optional item

부록 2. 측면 스키마의 BNF

FacetSch= **FACET** <FacName>
Origin Object :<ObjName>
Origin Property : <PropName>
Type : [PRIMITIVE|DERIVED]
(**Called Facets** : { <CallFacName1> {,<CallFacName2>}* }
Computational Rule : CompRule)

CompRule= EltSpec (WHERE WhereCond)

EltSpec= [ConExp|UniExp|Bi1Exp|Bi2Exp|NExp|ForExp]

ConExp= [<Constant>[+|*|-|/]Argument|Argument**<Constant>]

UniExp= [LN|EXP|LOG|ABS|SIN|COS|TAN|ROUND|SQRT]
(Argument)

Bi1Exp= Argument[+|*|-|/|=|<>|<=|<|>|>=]Argument

Bi2Exp= [PSUM|PPROD|PSUB|PDIV](Argument,Argument)

NExp= [MIN|MAX](Argument{,Argument}+)

ForExp= [SUM|PROD] Argument FOR [IndexList|ALL]

Argument= [<Constant>|<FacName1>|(CompRule)|CompRule]

IndexList= [<ObjName1> {, <ObjName2>}*|
<RoleName1>(<ParticiObj1>){,<RoleName2>(<ParticiObj2>)}*]

WhereCond= [BoolTerm|WhereCond OR BoolTerm]

BoolTerm= [BoolFactor|BoolTerm AND BoolFactor]

BoolFactor= (NOT) BoolPrimary

BoolPrimary= [BoolOperator|WhereCond]

BoolOperator= ScalarExp Comp ScalarExp

Comp= [=|<>|<=|<|>|>=]

ScalarExp= [Term|ScalarExp[+|-]Factor]

Term=[Factor|Term[*|/]Factor]

Factor= [+|-]Primary

Primary= [<Constant>|<FacName1>|<ObjName1>.<PropName1> .<FacName2>]

부록 3. 다상품 운송 모형의 측면 스키마

FACET limit_p	
Origin Object	: link
Origin Property	: limit
Type	: PRIMITIVE

FACET Trans_qty_p	
Origin Object	: trans
Origin Property	: trans_qty
Type	: PRIMITIVE

FACET trans_cost_p	
Origin Object	: trans
Origin Property	: trans_cost
Type	: PRIMITIVE

FACET trans_supply_p	
Origin Object	: prod_inf_at_origin
Origin Property	: supply
Type	: PRIMITIVE

FACET demand_p	
Origin Object	: prod_inf_at_dest
Origin Property	: demand
Type	: PRIMITIVE

FACET link_limit_test	
Origin Object	: link
Origin Property	: link_limit
Type	: DERIVED
Called Facets	: {limit_p, trans_qty_p}
Computational Rule	: (SUM trans_qty_p FOR product) <= limit_p

FACET supply_balance_test	
Origin Object	: origin
Origin Property	: supply_balance
Type	: DERIVED
Called Facets	: {supply_p, trans_qty_p}
Computational Rule	: (SUM trans_qty_p FOR destination) <= supply_p

FACET demand_balance_test	
Origin Object	: destinate
Origin Property	: demand_balance
Type	: DERIVED
Called Facets	: {demand_p, trans_qty_p}
Computational Rule	: (SUM trans_qty_p FOR origin) <= demand_p

FACET total_cost_call	
Origin Object	: trans_system
Origin Property	: total_cost
Type	: DERIVED
Called Facets	: {trans_qty_p, trans_cost_p}
Computational Rule	: SUM trans_qty_p*trans_cost_p FOR origin, destination, product

FACET total_cost_cal1	
Origin Object	: trans
Origin Property	: total_cost
Type	: DERIVED
Called Facets	: {trans_qty_p, unit_price_p}
Computational Rule	: 100000+trans_qty_p*unit_price_p

FACET total_cost_cal2	
Origin Object	: trans
Origin Property	: total_cost
Type	: DERIVED
Called Facets	: {trans_qty_p, unit_price_p}
Computational Rule	: 100325+trans_qty_p*unit_price_p

FACET total_cost_cal3	
Origin Object	: trans
Origin Property	: total_cost
Type	: DERIVED
Called Facets	: {trans_qty_p}
Computational Rule	: 200000+trans_qty_p

FACET unit_price_p	
Origin Object	: trans
Origin Property	: unitl_price
Type	: PRINITIVE

부록 4. 모형 인스턴스 정의 (model instance definition)의 BNF

ModInsDef= **MODEL INSTANCE DEFINITION** <ModInsDefName>
Facet Graph : <FacGraphName>
Solver : SolverSpec
Selection Rule : SQLStatement

SolverSpec= <SolverName>
((<SolverPara1> = SolverAssignedValue
{, <SolverPara2> = SolverAssignedValue}*))

SolverAssignedValue=
[<SingleValue>|SetValue]

SetValue= {<SingleValue1> {,<SingleValue2>}*}

*SQLStatement는 Standard SQL Syntax에 준함.

부록 5. 추가된 측면 스키마

FACET var_trans_cost_p	
Origin Object	: trans
Origin Property	: var_trans_cost
Type	: PRIMITIVE

FACETfixed_trans_cost_p	
Origin Object	: link
Origin Property	: fixed_trans_cost
Type	: PRIMITIVE

FACET use_p	
Origin Object	: link
Origin Property	: use
Type	: PRIMITIVE

FACET link_limit_test2	
Origin Object	: link
Origin Property	: link_limit
Type	: DERIVED
Called Facets	: {limit_p, trans_qty_p, use_p}
Computational Rule	:(SUM trans_qty_p FOR product)<=limit_p*use_p

FACET total_cost_cal2	
Origin Object	: trans_system
Origin Property	: total_cost
Type	: DERIVED
Called Facets	: {trans_qty_p, var_trans_cost_p, fixed_trans_cost_p, use_p}
Computational Rule	: SUM var_trans_cost_p*trans_qty_p FOR origin, destination, product + SUM fixed_trans_cost_p*use_p FOR origin, destination