전문가시스템 구축을 위한 지식레벨 지원도구에 관한 연구

김 은 경[†] · 김 성 훈[††] · 박 충 식[†††]

요 약

심볼레벨에서 개발된 1세대 전문가시스템의 문제점을 해결하기 위하여 최근 지식레벨에서 2세대 전문가시스템을 개발하기 위한 여러 가지 방법들이 제안되었으나, 대개 개념적인 모델링 기법을 제시하는 수준에 머무르고 있다. 본 논문에서는 이러한 개념적인 모델링 기법이 실용 가능한 개발기법이 될 수 있도록 기존의 연구 내용을 수정, 보완한 태스크 객체 모델링(TOM) 기법을 제시하였다. 이 기법에서 태스크 객체(TO)는 자신의 목표와 수행 조건, 실행할 행위 지식 등을 포함하는 하나의 지식 단위로 정의하고, TO의 구조적, 동적, 기능적 측면을 쉽게 도식화할 수 있는 태스크 객체 다이어그램(TOD)을 정의하였다. 또한, 각 TO의 행위 지식을 표현하는 기본 단위로서 추론유형(Inference Type)들을 정의하였다. 본 논문에서 제안한 TOM 기법이 단순히 개념적인 모델링 기법이 아니라 실용적인 2세대 전문가시스템 개발기법으로 활용될 수 있도록 TOD 에디터와 TO 에디터를 개발하고, TO의 상태 변화에 기반을 둔 TO 처리 알고리즘을 개발하였다. 또한, 정의된 추론유형들이 대표적인 전문가시스템 개발도구인 IRE(Intelligent Rules Element)의 메소드로 자동 변환될 수 있도록, 각각의 추론유형과 IRE의 메소드를 1:1로 대응시킨 추론유형 라이브러리를 구축하였다.

A Study on a Knowledge-level Supporting Tool for Building Expert Systems

Eun Gyung Kim[†] · Seong Hoon Kim[††] · Choong Shik Park[†††]

ABSTRACT

In order to overcome the problems with first generation expert systems at the symbol level, recently various knowledge level development techniques of second generation expert systems have been proposed. But, these techniques are conceptual modelling techniques. This paper modifies and complements these conceptual modelling techniques and proposes a Task Object Modelling (TOM) technique as a practical knowledge level expert system development technique. This paper defines a Task Object(TO) as a knowledge unit consisted of a goal, execution conditions, behaviour knowledge, and so on. And, we define a Task Object Diagram(TOD) to depict structual, dynamic, and functional aspects of TO easily. We also define Inference Types as basic units to describe behaviour knowledge of TOs. In order to utilize the proposed TOM technique as not a simple conceptual modelling technique but a practical second generation expert system development technique, we implement a TOD editor, a TO editor, and TO processing algorithm based on the state of TOs. Also we implement a Inference Types Library, in which each inference type is corresponded to an IRE(Intelligent Rules Element) method, to transform the defined inference types into IRE methods automatically.

† 종신회원: 한국기술교육대학교 컴퓨터공학과 부교수
†† 정 회 원: 영동대학교 전자공학부 전임 강사
††† 정 회 원: 영동대학교 전자공학부 조교수
논문접수: 1998년 1월 24일, 심사완료: 1998년 3월 3일

1. 서 론

기존의 지식표현 방법은 문제풀이 지식과 그 지식에 대한 처리가 분리되어 있는 심볼레벨(symbol level)의 표현 체계를 가지므로 전문가의 복합적인 지식체계와 사고과정을 기술하기에 적절하지 않으며, 결과적으로 지식습득이 전문가시스템 개발시에 병목 현상을 유발하는 주요 원인이 되고 있다[1]. 한편, 전문가시스템은 단순히 전문가로부터 획득된 지식을 보관하는 데이터베이스가 아니라, 실세계의 현상으로 묘사되거나 관찰될 수 있는 어떤 적절한 행위를 나타내는 조작 모델(operational model)이라고 볼 수 있다. 따라서 전문가시스템의 개발과정을 모델링 행위로 보는 것이 바람직하며, 지식 공학자의 가장 중요한 역할은 문제분야의 지식을 습득하여 문제해결에 적합한 시스템 모델을 개발하는 것이라고 할 수 있다. 따라서 전문가의 지식표현 체계와 일치하면서 지식 공학자의 모델링 행위가 자연스럽게 시스템 개발과 연결될 수 있는, 지식레벨(knowledge level)의 전문가시스템 개발 도구가 절실히 필요하다. 지식레벨이란 용어는 Newell이 자신의 논문에서 가장 먼저 사용하였으며, 심볼레벨 위에서 기존의 지식표현 방법으로 기술된 독자적인 목표와 문제풀이 방법을 포함하고 있는 지식의 한 단위라고 정의할 수 있다. 또한, 심볼레벨에서 개발된 기존의 전문가시스템을 1세대 전문가시스템이라고 칭하는 반면, 지식레벨에서 개발된 차세대 전문가시스템을 2세대 전문가시스템이라고 칭한다[2, 3, 4].

본 논문에서는 2세대 전문가시스템 개발도구의 필요성을 인식하고, 현재 가장 널리 사용되고 있는 전문가시스템 개발도구 가운데 하나인 뉴론 데이터(Neuron Data)의 IRE(Intelligent Rules Element)[5] 상에서 지식레벨의 전문가시스템 개발이 가능하도록 연구하였다. 먼저 성취할 목표와 그 목표를 성취하는데 필요한 지식 및 처리기능을 "태스크 객체(Task Object: TO)"라는 하나의 지식 단위로 정의하고, TO의 구조적, 동적, 기능적 측면을 쉽게 도식화할 수 있는 "태스크 객체 다이어그램(Task Object Diagram: TOD)"을 정의함으로써, 인간의 작업 단위와 마찬가지로 지식레벨에서 전문가시스템을 개발할 수 있는 "태스크 객체 모델링(Task Object Modelling: TOM)" 기법을 제안하였다. 또한, 이 TOM 기법이 단순한 모델링 기법에 그치지않고 실용적인 개발기법으로 활용될 수 있도록, TOD 에디터와 TO 에디터를 개발하였다.

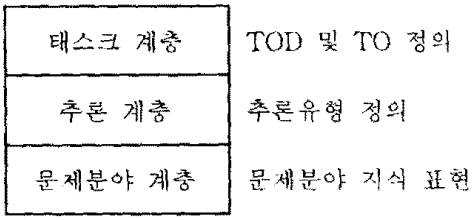
한편, KADS[6, 7]나 SKE(Structured Knowldege Engineering)[8, 9] 등과 같은 기존의 연구에서도 지식레벨에서의 전문가시스템 개발기법들이 제안되었으나, 대부분 개념적인 수준의 기법들이므로 실제로 전문가시스템을 개발할 때 상용화된 전문가시스템 개발도구에서 적용하기에는 많은 어려움이 있다. 그러나 본 논문에서는 각종 에디터의 개발과 더불어 TO의 상태 변화에 기반을 둔 TO 처리 알고리즘을 개발하였다. 또한, 각 TO의 행위 지식을 표현하는 기본 단위로서 추론유형(Inference Type)들을 정의하고, 이들 추론유형이 IRE의 메소드로 자동 변환될 수 있도록 각각의 추론유형에 대응하는 함수와 IRE 메소드를 1:1로 대응시킨 추론유형 라이브러리를 구축하였다. 마지막으로 TOM 기법을 적용하여 통신망 설계 전문가시스템의 프로토타입 시스템을 개발하고, 그 성능을 분석하였다.

2. 태스크 객체 모델링(TOM) 기법

2.1 TOM의 개념

기존의 지식레벨 개발기법은 대부분 문제분야 계층(domain layer)과 추론 계층, 태스크 계층, 전략 계층(strategy layer)의 4층 구조로로 이루어져 있다. 문제분야 계층은 문제분야의 개념과 개념의 속성, 개념간의 관계 등을 기술하는 부분이며, 추론 계층에는 해석모델(interpretation model)을 이용하여 구체적인 추론내용을 기술한다. 태스크 계층에는 태스크 단위의 문제풀이 업무을 상세히 기술하며, 이는 추론제어 지식에 해당한다. 전략 계층에서는 문제풀이 업무의 순서를 순서 결정과 같은 문제풀이 전략을 기술하게 된다. 심볼레벨의 전문가시스템 개발도구는 태스크 계층과 추론 계층이 혼합되어 있는 형태로, 지식습득 및 지식베이스 구축 과정에서 추론내용 지식과 추론제어 지식을 분리시키는 것이 어려우며, 결과적으로 전문가시스템 개발을 매우 어렵게 한다. 따라서 추론내용 지식에 해당되는 추론 계층과 추론제어 지식에 해당되는 태스크 계층이 명확히 분리되는 지식레벨의 개발기법이 절실히 요구된다. 본 논문에서 제안한 TOM

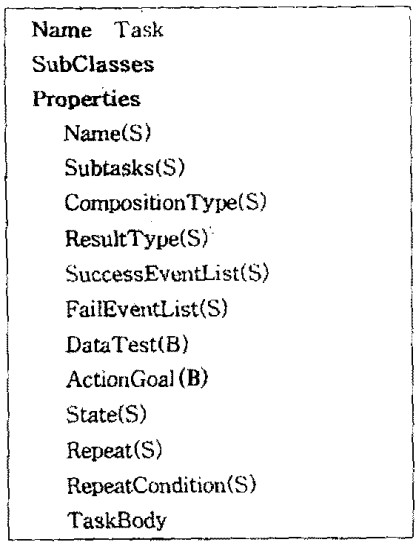
기법에서는 태스크 계층을 TOD로 표현하고, 전략 계층은 별도로 설정하지 않고 태스크 계층의 TOD에서 AND/OR 그래프 형태로 표현되도록 함으로써 (그림 1)과 같이 3층 구조로 구성하였다. 추론 계층에서는 고정된 해석모델을 제시하는 대신 기본적인 추론유형만을 정의하였다. 대표적인 지식레벨 개발기법인 SKE와 KADS에서는 모니터링, 설계, 예측 등과 같은 몇가지 고정된 해석모델을 제시하고 있으며, 시스템 개발시 해결하려는 문제에 적합한 한가지 이상의 해석모델을 선택하여 통합, 수정해야 하므로, 오히려 시스템 개발이 어렵고 융통성도 떨어지게 된다. 한편, 문제분야 계층에서는 IRE의 클래스와 객체 개념을 이용하여 문제분야의 지식을 표현한다.



(그림 1) TOM 기법의 3층 구조
(Fig. 1) Three-layer framework of the TOM technique

2.2 태스크 객체(TO)의 정의

TO는 실제로 전문가가 수행하는 한 단위의 일이나, 지식 공학자가 판단한 일의 단위가 될 수도 있다. 한편, 기존의 지식표현 방법은 지식표현과 그 지식을 처리하는 기능이 완전히 분리되어 있기 때문에 이러한 지식이 종합되어 있는 인간의 지식을 자연스럽게 표현하기가 매우 어렵다. 따라서 TO는 인간의 작업 단위와 일치하면서, 독자적인 목표와 목표를 성취하는데 필요한 지식 및 처리 기능이 하나의 단위가 되도록 정의되어야 한다. 또한, TO는 전체 시스템을 표현하는 TOD의 구성 요소이므로, 전체 시스템내에서 각 TO가 수행될 시기, 수행에 필요한 조건, 구체적인 수행 내용 등이 표현될 수 있어야 한다. 또, TO의 조합으로 전문가시스템을 모델링하기 위해서는 TO의 구조적, 동적, 기능적인 측면을 정의할 수 있어야 하며, 본 논문에서는 이러한 요구 사항을 종합하여 (그림 2)와 같이 Task 클래스를 정의하였다. 구조적인 측면은 한 TO를 구성하는 부태스크 객체(SubTask Object: STO)들의 구성방법을 의미하며, TO를 정의할 때 (그림 2)의 CompositionType과 Result-Type에 기술된 정보가 종합되어 STO의 구성방법이 정해진다. 즉, 한 TO의 모든 STO의 상태가 성공(success)이 되어야 하는 경우, CompositionType은 'and'로, ResultType은 'success'로 정의해야 한다. 동적인 측면은 활성(active) 상태가 될 TO로 선정되는 조건을 의미하며, STO들 사이의 수평적인 활성화 관계를 기술하는 것으로, SuccessEventList와 FailEventList에 기술한다. 기능적인 측면은 DataTest와 TaskBody에 기술되며, DataTest는 TO 처리에 필요한 자료를 확인하는 항목이며, TO가 활성화되었을 때 목표를 성취하기 위해 실제로 수행할 내용은 TaskBody 부분에 기술한다.



(그림 2) Task 클래스의 정의
(Fig. 2) Definition of task class

(1) Name
TO를 구분할 수 있도록 이름을 기술한다.
(2) Subtasks
TO를 구성하는 STO들의 이름을 기술한다.
(3) ResultType
CompositionType과 종합되어 STO들의 구성 방법이 정해지며, 다음과 같은 값을 가질 수 있다.
① success : STO의 상태가 성공이 되어야 함
② fail : STO의 상태가 실패가 되어야 함

③done:STO의 상태와 상관없이 모든 부태스크들을 수행해야 함

(4)CompositionType

①and:모든 STO가 ResultType을 만족해야 한다.

②or:STO 가운데 하나만 ResultType을 만족하면 된다.

단, ResultType이 'done'인 경우에는 선택할 필요가 없다.

(5)SuccessEventList

"성공" 상태일 때 자신을 활성화시킬 수 있는 TO들의 이름을 기술한다.

(6)FailEventList

"실패" 상태일 때 자신을 활성화시킬 수 있는 TO들의 이름을 기술한다.

(7)DataTest

TO 처리에 필요한 자료형과 자료값을 기술한다.

(8)State

TO의 상태를 나타내며, 다음과 같은 값을 가질 수 있다.

①활성(active):TO가 실행될 수 있는 상태

②대기(wait):STO들의 실행이 완료되기를 기다리는 상태

③성공(success):수행 결과 TO의 목표가 성공적으로 성취된 상태

④실패(fail):수행 결과 TO의 목표가 성공적으로 성취되지 못한 상태

⑤미정(unknown):TO의 상태가 정해지지 않은 상태

각 TO가 생성되는 초기상태에서는 TOD의 루트 TO의 상태만 'active'이고, 나머지 TO는 'unknown' 상태이다.

(9)Repeat

TO를 1회만 수행할 지 또는 조건이 만족될 때까지 반복 수행할 지를 기술하며, 각각 'oneshot'과 'cycle'로 구분한다.

(10)RepeatCondition

TO의 Repeat 항목을 'cycle'로 설정한 경우, 반복이 종료될 조건을 TO의 상태, 즉 'success'나 'fail'로 기술한다.

(11)TaskBody

TO가 활성화되었을 때 수행할 행위를 기술하며, 구체적인 기술 방법은 뒤에서 자세히 설명하겠다.

2.3 태스크 객체 다이어그램(TOD)의 정의

본 논문에서 제안한 TOM 기법에서는 전체 시스템이 TO들에 의해서 계층 구조로 모델링되는데, 전문가시스템의 최종 목표를 갖고 있는 루트 TO와 여러 개의 STO들로 구성된다. 본 논문에서는 이러한 계층적인 구조를 그림으로 쉽게 표현할 수 있도록 태스크 객체 다이어그램(Task Object Diagram:TOD)을 정의하였다. 이는 소프트웨어 공학에서 사용되는 객체지향 개발기법의 다이어그램을 변형한 것으로, TOD는 크게 다음 2가지 요소로 구성된다.

(1)노드(node):TO의 구조적 측면을 표현할 수 있도록 다음과 같이 구분하였다.

①CompositionType AND/OR:육각형과 원으로 구분한다.

②ResultType SUCCESS/FAIL/DONE:노드 안에 각각 S, F, D로구분한다.

③Repeat ONESHOT/CYCLE:반복 화살표의 유무로 구분한다.

(2)링크(link):TO와 STO들간의 관계 뿐만 아니라, TO의 동적인 측면을 표현할 수 있도록 다음과 같이 구분하였다.

①검은색 화살표 실선:TO와 STO들간의 관계 표시

②푸른색 화살표 실선:성공 이벤트 관계 표시

③붉은색 화살표 실선:실패 이벤트 관계 표시

TOD의 작성 예는 4장의 (그림 7)로 대신하겠다.

2.4 TO 처리 알고리즘

TOD의 루트 TO에 명시된 최종 목표를 성취하기 위해서는, TO의 구조적인 측면과 동적인 측면을 고려하여 각 TO의 상태를 적절히 변경하면서 전체 TO를 실행해야 한다. 또한, 각 TO가 활성화되면 자신의 상태가 성공 또는 실패로 판단될 때까지 자신의 목표를 성취하는데 필요한 행위를 1회 또는 반복 수행해야 한다. 본 논문에서는 각 TO의 상태를 기반으로 하는 TO 처리 알고리즘을 개발하고, 이를 IRE의 API 함수로 구현하였으며, (그림 3)은 TO 처리 알고리즘을 개략적으로 표현한 것이다.

2.5 추론유형 정의

추론유형(inference type)이란 추론 단계에서 하나의 행위를 수행하는 최소 단위로, 어떤 입력 데이터에 대

```
Procedure Task_execute()
Begin
  task : active 상태인 태스크 이름 /* task.State가 "active"인 task선택 */
  Success_event_check(task);
    /* SuccessEventList에 기술된 task들의 State가 "success"인지 체크 */
  Fail_event_cist(task);
    /* FailEventList에 기술된 task들의 State가 "fail"인지 체크 */
  If task.subtasks = NULL  /* subtask가 없는 경우 자신을 처리 */
   Then  Run_taskbody(task); /* task의 TaskBody 부분 실행 */
         State_modify(task);  /* task의 상태 수정 */
         Repeat_check(task);  /* task의 반복 실행 여부 판단 */
   Else  task.state := "wait";  /* task의 상태를 "wait"으로 변경 */
     Case task.ResultType Is  /* ResultType에 따라 처리 구분 */
       [DONE] :
         모든 subtask들의 상태가 "unknown"이 아닐 때까지
         Task_execute()와 Repeat_check()를 반복 수행한다.
       [SUCCESS] :
         Case task.CompositionType Is
           [OR] : subtask 중 하나의 State가 "success" 이거나
                  모든 task의 State가 "unknown"이 아닐 때까지
                  Task_execute()와 Repeat_check()를 반복 수행한다.
           [AND] subtask 중 하나의 State가 "fail"이거나
                  모든 task의 State가 "unknown"이 아닐 때까지
                  Task_execute()와 Repeat_check()를 반복 수행한다.
         EndCase
       [FAIL] :
         Case task.CompositionType Is
           [OR] : subtask 중 하나의 State가 "fail"이거나
                  모든 task의 State가 "unknown"이 아닐 때까지
                  Task_execute()와 Repeat_check()를 반복 수행한다.
           [AND] subtask 중 하나의 State가 "success"이거나
                  모든 task의 State가 "unknown"이 아닐 때까지
                  Task_execute()와 Repeat_check()를 반복 수행한다.
         EndCase
     EndCase
     State_check(task);  /* subtask들의 상태나 자신의 TaskBody 부분의
                            실행 결과에 따라 task의 State를 변경한다. */
End Task_execute;

/* Main module */
Begin
   Root := active 상태인 태스크의 이름; /* TOD의 root task */
   Task_execute();  /* 태스크 실행 */
   Repeat_check(Root);  /* 반복 수행 여부 판단 */
End;
```

(그림 3) TO 처리 알고리즘
(Fig. 3) Algorithm for TO processing

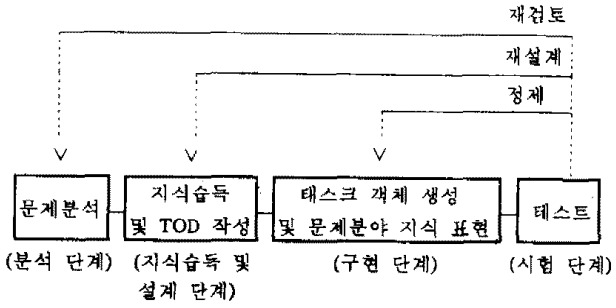
해서 행위를 수행한 결과로 새로운 지식 또는 정보를 출력하게 된다. 즉, 특정 개발도구의 지식 표현 방법과는 무관한, 각 TO의 행위 지식을 표현하는 기본 단위이다. 본 논문에서는 SKE의 추론유형을 참조하여 〈표 1〉과 같이 추론유형을 정의하였으며, 필요한 경우 새로운 추론유형들이 계속 추가될 수 있다. 추론유형의 정의에 사용된 용어는 다음과 같이 정의한다. 개념(concept)은 문제분야 지식 가운데 핵심이 되는 객체(object)로, 이름으로 구분되며, 하나 이상의 속성(property)을 갖는다. 속성은 이름과 속성이 가질 수 있는 값(value)으로 정의된다. 구조(structure)란 하나 이상의 개념이나 그들간의 관계로 구성되는 복잡한 객체를 의미하며, 전체 시스템을 하나의 구조로 볼 수 있다.

〈표 1〉 추론유형의 정의
〈Table 1〉 Definition of inference types

연산유형	추론유형	인수(arguments)
Change Concept	assign	property → property-value
	compute	structure → property-value
Generate New Concept	instantiate	concept → instance
	identify	instance → concept
	generalize	set of instances → concept
	abstract	concept → concept
	specify	concept → concept
	select	set of concepts → concept
	heuristic-match	concept → concept
Similarities /Differences	compare	value + value → value
	match	structure + structure → structure
	confirm	value → value
Structure Manipulation	compose	set of instances → structure
	decompose	structure → set of instances
	transform	structure → structure

2.6 문제분야 지식(Domain Knowledge)

문제분야 지식은 전문가시스템을 개발하려는 분야의 전문지식(expertise)을 의미하며, 각 TO가 수행할 몸체(TaskBody) 부분을 작성할 때 나타나는 자료에 해당한다. 이 지식의 표현은 실제로 사용할 전문가시스템 개발도구에 따라 달라지며, IRE를 사용하는 경우 클래스와 객체, 프로퍼티 등으로 문제분야 지식을 표현하게 된다. 이 부분에 대해서는 4장에서 예를 보였다.



(그림 4) TOM 기법을 이용한 전문가시스템의 개발 단계
(Fig. 4) Development phases of expert systems with the TOM technique

2.7 TOM 기법을 이용한 전문가시스템 개발 단계

TOM 기법을 기반으로 전문가시스템을 개발하는 단계를 도식화하면 (그림 4) 와 같다. TOM 기법에서 각 TO는 지식공학자의 업무처리 단위와 일치하므로 지식습득 단계에서 TOD를 작성하는 것이 매우 용이하며, 결과적으로 지식습득 단계에서의 병목현상을 최소화할 수 있다.

3. TOM 기법의 지원 도구 구현

본 논문에서는 TOM 기법이 현재 가장 널리 사용되고 있는 전문가시스템 개발도구 가운데 하나인 IRE에서 실용화될 수 있는 실질적인 지식레벨 개발기법이 될 수 있도록, IRE 상에서 활용할 수 있는 각종 개발 지원 도구들을 개발하였다.

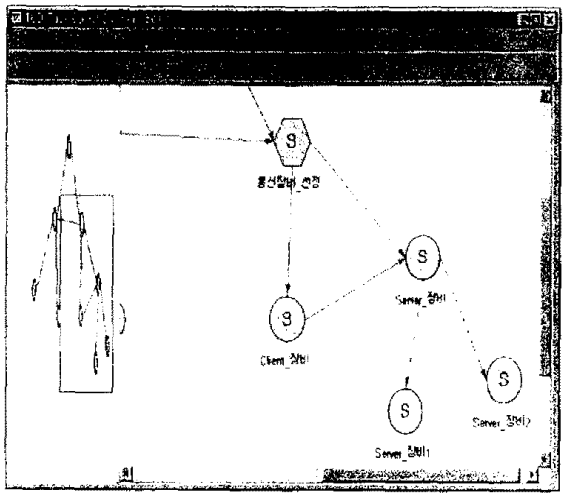
먼저, IRE에서 TOD를 쉽게 구성할 수 있는 TOD 에디터를 개발하고, TOD를 구성하는 TO들을 편집할 수 있는 TO 에디터를 개발하였으며, TO 에디터에서 편집된 각각의 TO는 IRE의 Task 클래스의 하나의 객체로서 생성된다. 또, 앞에서 정의한 추론유형을 각각 대응하는 함수로 정의하고, 이들 함수와 IRE 메소드를 1:1로 대응시켜 구현한 추론유형 라이브러리를 구축하였다. 따라서 사용자는 TO의 행위 지식을 IRE의 규칙이나 메소드로 직접 표현하는 대신, TO 에디터에서 적절한 추론유형 함수를 선택하여 필요한 인수만 기술하면 된다. 즉, 사용자가 에디터를 이용하여 TOD를 작성하고, TOD를 구성하는 각각의 TO를 편집한 다음, 추론유형 라이브러리 및 IRE의 API 함수로 구현된 TO 처리 알고리즘과 함께 컴파일하면, IRE 상에서 수행될 수 있는 실행 모듈이 생성된다.

3.1 TOD 에디터

2.3절에서 설명한 TOD의 작성이 용이하도록 지원하는 도구로서 먼저 (그림 5)와 같은 TOD 에디터를 구현하였다.

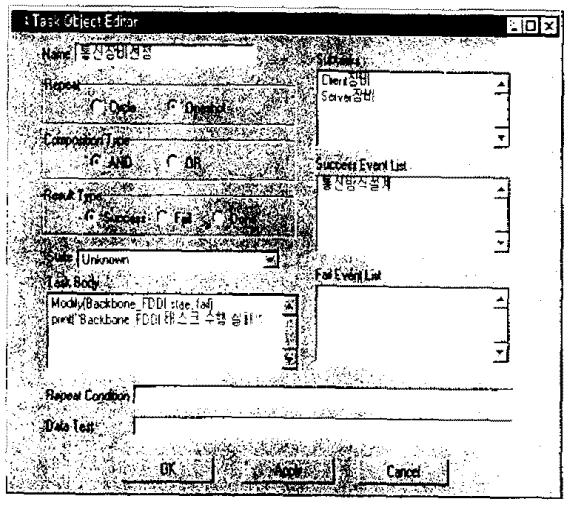
3.2 TO 에디터

TOD의 각 TO를 2.2절에서 정의한 Task 클래스의 객체로 쉽게 생성할 수 있도록, (그림 6)과 같은 TO 에디터를 개발하였다. TOD 에디터에서 작성한 TOD



(그림 5) TOD 에디터
(Fig. 5) TOD editor

의 한 노드를 마우스로 클릭하면, 해당 TO를 편집할 수 있는 TO 에디터가 표시된다. TO의 몸체(TaskBody) 부분에는 추론유형들을 리스트박스 형태로 표시하여 사용자가 쉽게 선택할 수 있도록 하였으며, 현재 인수는 사용자가 직접 입력해야 한다. 앞으로 인수의 지정이 보다 용이한 별도의 태스크 몸체 에디터를 개발하여, TO 에디터에서 TaskBody 부분을 마우스로 클릭하면 태스크 몸체 에디터가 표시되도록 구현함으로써 사용자가 보다 쉽게 TO를 편집할 수 있도록 확장할 계획이다.



(그림 6) TO 에디터
(Fig. 6) TO editor

3.3 추론유형 라이브러리

〈표 1〉에서 정의한 추론유형들을 각각 IRE의 함수로 정의하고, IRE의 메소드와 1:1로 대응시켜 구현한 추론유형 라이브러리를 구축하였다. 따라서 사용자가 TO 에디터에서 TaskBody 부분을 편집할 때 적절한 추론유형을 선택한 다음 인수를 지정하면 되므로, IRE의 지식표현 방법을 모르고도 지식베이스 구축이 가능하게 된다. 〈표 2〉는 추론유형 라이브러리에 저장된 일부 추론유형 함수와 그에 대응하는 IRE의 메소드 내에서 기술된 규칙 형태를 보여준다.

〈표 2〉 추론유형 라이브러리
〈Table 2〉 Inference types library

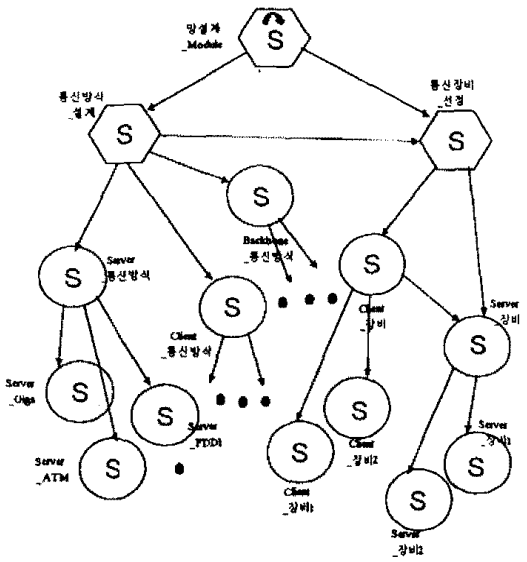
추론유형 함수	인수	IRE 규칙 형태
assign (A,B,C,D)	A:수행조건(Boolean) B:slot|value|expression C:slot D:수행결과(Boolean)	(IF (Yes A)) (THEN (Assign B C) (Assign True D))
compare (A,B,C,D,E)	A:수행조건(Boolean) B:relational operator C:slot|value|expression D:slot|value|expression E:수행결과(Boolean)	(IF (Yes A)) (B C D) (THEN (Assign True E)) (ELSE (Assign False E))
confirm (A,B,C,D)	A:수행조건(Boolean) B:Yes|No C:slot|value|expression D:수행결과(Boolean)	(IF (Yes A) (B C)) (THEN (Assign True D)) (ELSE (Assign False D))
specify (A,B,C,D,E,F)	A:수행조건(Boolean) B:class C:property D:value E:specified object F:수행결과(Boolean)	(IF (Yes A) (= (|B|).C D)) (THEN (CreateObject B E) (Assign True F)) (ELSE (Assign False F))
decompose (A,B,C,D)	A:수행조건(Boolean) B:object C:part-of object D:수행결과(Boolean)	(IF (Yes A) (CreateObject (B)|C|)) (THEN (Assign True D)) (ELSE (Assign False D))
match (A,B,C,D,E,F)	A:수행조건(Boolean) B:class C:property D:class E:property F:수행결과(Boolean)	(IF (Yes A) (Assign (|B|).C str1) (= str1 (|D|).E)) (THEN (Assign True F)) (ELSE (Assign False F))

4. 적용 사례: 통신망 설계 전문가시스템의 프로토타입

본 논문에서 제안한 TOM 기법의 적용 사례로서 통신망 설계 전문가시스템의 전체 기능 가운데 통신방식과 통신 장비를 선정하는 부분만을 선택하여 프로토타입 시스템을 개발하였다.

4.1 TOD 작성

통신망 설계 전문가로부터 습득한 지식으로 모델링된 TOD는 (그림 7)과 같다. 프로토타입에서 망설계_Module TO는 통신방식설계와 통신장비선정이라는 두 개의 STO로 구성되고, 통신방식설계 TO가 성공적으로 수행된 다음에만 통신장비선정 TO가 수행될 수 있으므로, 통신방식_설계와 통신장비_선정 TO 사이에 푸른색 화살표를 사용하여 성공 이벤트 관계를 표현하였다. 또한, 통신방식설계 TO는 Server와 Client, Backbone의 통신방식을 선정하는 세 개의 STO로 구성되며, 이들간의 수행 순서는 중요하지 않지만 모두 성공적으로 수행되어야 통신방식설계 TO가 실행될 수 있으므로, success와 and를 의미하는 S-육각형 노드로 표시하였다. Server_통신방식 노드는 여러 개의 말단 TO들로 구성되며, 이들 가운데 하나만 성



(그림 7) 통신망 설계 전문가시스템의 TOD
(Fig. 7) TOD of the communication network design expert system

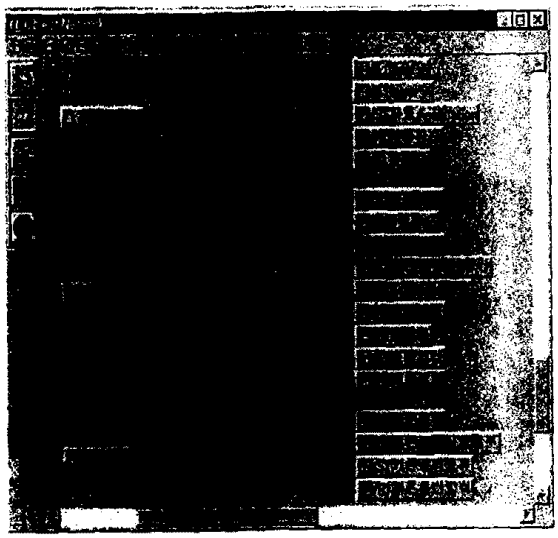
공 상태가 되면 Server_통신방식 TO가 실행될 수 있으므로, success와 or를 의미하는 S-원 노드로 표시하였다. 또한, 망설계_Module TO는 성공 상태가 될 때까지 반복 수행되어야 하므로, 노드에 반복 화살표를 표시하였다.

4.2 TO 생성

(그림 8)은 (그림 7)의 TOD 가운데 Server_Giga 노드의 TO를 생성한 예이다.

```
Task  Giga
 Classes  Task
 Properties
  Name: Giga
  Subtasks:
  CompositionType:
  ResultType:
  SuccessEventList:
  FailEventList:
  DataTest:
  ActionGoal:
  State: unknown
  Repeat: one
  RepeatCondition:
  TaskBody:
    Compare(True, =, 논리적요소.고속성_Server,
                "Dedicated 100M 이상", Rtn_1)
    Compare(Rtn_1, =, 논리적요소.경제성, "미고려", Rtn_2)
    Assign(Rtn_2, "Giga Base", 통신방식.Server방식,    )
    Assign(Rtn_2, 7, 통신방식.Server방식_우선순위,    )
```

(그림 8) Server_Giga 노드의 TO
(Fig. 8) TO of the Server_Giga node



(그림 9) 문제분야 지식의 객체 네트워크
(Fig. 9) Object network of the domain knowledge

4.3 문제분야 지식

(그림 9)는 통신망 설계와 관련된 전문지식을 표현하기 위해서 IRE에서 정의한 일부 객체와 프로퍼티의 관계를 보여주는 IRE의 객체 네트워크(object network)이다.

4.4 분 석

위의 전문가시스템을 IRE만을 이용하여 개발하는 경우, 우선 IRE를 사용해 본 경험이 없는 개발자는 IRE의 사용법을 배우기 위해서 많은 시간과 노력을 투자해야 한다. 또한, IRE를 사용해 본 경험이 있다 해도 전문가로부터 지식을 습득하여 지식베이스를 구축할 때, 실제 문제해결 지식과는 무관한 추론제어 지식까지도 고려해야 하므로 전문가시스템 개발은 매우 어려운 작업이 된다. 그러나 본 논문에서는 TOM 기법을 이용함으로써 위의 전문가시스템 개발과정을 크게 단축할 수 있었다. 즉, 지식 습득시에 TOD 에디터를 사용하여 TOD를 작성하고, TO 에디터와 추론유형 라이브러리를 사용하여 TO들을 생성하였으며, 결과적으로 IRE의 규칙이나 메소드로 직접 지식을 표현하지 않고도 IRE 상에서 수행되는 전문가시스템을 쉽게 개발할 수 있었다.

5. 결 론

지금까지 지식레벨에서 2세대 전문가시스템을 개발하기 위한 노력으로 여러 가지 개발기법들이 연구되었으나, 대부분 개념적인 모델링 방법을 제안하는 수준이었다. 즉, 실제 상용화된 개발도구에서 활용할 수 있는 구체적인 방법 및 지원 도구를 제공하지 않으므로, 실제로 전문가시스템을 개발할 때 적용하기에는 많은 어려움이 있다. 본 논문에서는 기존의 개념적인 수준의 개발기법들을 수정, 보완하여, 실제로 2세대 전문가시스템을 개발할 수 있는 TOM 기법을 제안하였다. 또한, 대표적인 전문가시스템 개발도구인 IRE에서 실제로 이 기법을 적용할 수 있도록, TOD 에디터와 TO 에디터 및 TO 처리 알고리즘을 개발하고, 추론유형 라이브러리를 구축하였다. 한편, 추론유형 라이브러리에서 언어 레벨의 표현만 변경하면 어떤 개발도구에서도 TOM 기법을 적용하는 것이 가능하다. 즉, CLIPS[11]라는 개발도구에서 적용하기 위해

서는 각 추론유형과 대응하는 IRE 메소드를 CLIPS의 규칙 형태로 바꾸어주면 된다. 따라서 본 논문에서 제안한 TOM 기법은 단순히 개념적인 모델링 기법이 아니라 어떤 개발도구에서도 적용 가능한 실질적인 2세대 전문가시스템 개발기법이라고 할 수 있다.

앞으로 태스크 몸체 에디터를 구현하여 사용자가 보다 편리하게 TO를 편집할 수 있도록 확장하고, TO를 기반으로 하는 설명 기능 및 TO 단위의 테스팅 기법 등에 대해서 연구할 계획이다. 또한 보다 다양한 응용 분야에 적용해 봄으로써 추론유형 라이브러리에 보다 다양한 추론유형들을 추가하여, 모든 응용분야에서 폭넓게 활용할 수 있도록 확장할 계획이다.

참 고 문 헌

[1] D. A. Waterman, A Guide to Expert System, Addison-Wesley, 1986.

[2] A. Newell, "The Knowledge Level," Artificial Intelligence 18, 1992, pp. 87-127.

[3] 박충식, 태스크 개념을 이용한 전문가시스템 개발기법, 박사학위논문, 1992.

[4] 박충식, 김재희, "태스크 개념을 이용한 전문가시스템 개발기법," 지능정보시스템 제1권 제1호, 1992. 8., pp. 33-48.

[5] NeuronData, IRE(Intelligent Rule Element) manual, 1995.

[6] G. Schreiber, et al, CommonKADS : A Comprehensive Methodology for KBS Development, IEEE Expert, December 1994, pp. 28-37.

[7] KADSII Technical Reports, SWI Home Page, http://swi.psy.uva.nl/swi.html.

[8] K. Gardner, "Designing Knowledge Systems with Objects," AI Expert, Sept. 1991, pp. 32-39.

[9] Bechtel AI institute and BOLESIAN, Structured Knowledge Engineering Tutorial, 1990.

[10] J. Rumbaugh, M. Blaha, W. Premerlani, F. Eddy and W. Lorensen, Object-Oirented Modelling and Design, Prentice-Hall, 1991.

[11] Johnson Space Center, CLIPS Reference Manual, 1993.

김 은 경

1984년 숙명여자대학교 물리학과 졸업(이학사)
1987년 중앙대학교 대학원 전자계산학과 졸업(이학석사)
1991년 중앙대학교 대학원 전자계산학과 졸업(공학박사)
1992년 3월~현재 한국기술교육대학교 컴퓨터공학과 부교수
관심분야 : 전문가시스템, 분산인공지능, 지능형에이전트 등임

김 성 훈

1988년 서강대학교 전자공학과 졸업(공학사)
1990년 연세대학교 대학원 전자공학과 졸업(공학석사)
1996년 연세대학교 대학원 전자공학과 졸업(공학박사)
1996년 3월~현재 영동대학교 전자공학부 전임강사
관심분야 : 사용자 인터페이스, 패턴인식, 서명 검증, 필기인식 등임

박 충 식

1985년 한양대학교 전자공학과 졸업(공학사)
1987년 연세대학교 대학원 전자공학과 졸업(공학석사)
1992년 연세대학교 대학원 전자공학과 졸업(공학박사)
1993년 3월~현재 영동대학교 전자공학부 조교수
관심분야 : 전문가시스템, 에이전트이론, 컴퓨터 비전 등임