

論文94-31B-7-18

# UNIX-TUTOR : UNIX 교육을 위한 지능형 개인교사 시스템

## (UNIX-TUTOR : Intelligent Tutoring System for Teaching UNIX)

鄭木童\*, 金容蘭\*, 金英成\*, 辛教善\*\*

(Mok-Dong Chung, Yong-Ran Kim, Yung-Sung Kim and Kyo-Sun Shin)

### 要約

본 논문에서는 UNIX 초심자들을 위하여 UNIX의 기본 개념과 명령어를 학생과의 대화에 의해서 교육하는 개인교사 시스템:UNIX-TUTOR를 구현한다. 제안된 시스템은 학습자가 직접 참여할 수 있도록 하여서 보다 상호주도적인 교육이 되도록 하고 있고, 교육 방법으로서 소크라테스 방법과 코치 방법을 병행하고 있으며, 학습자 모델은 오버레이 모델과 버그 모델을 함께 사용하고 있다. 따라서 UNIX TUTOR는 학습자 모델의 정보에 따라 다양한 설명 기능을 구현 함으로써 학습성취도가 서로 다른 여러 사람들을 적절히 교육하는 것을 목표로 하고 있다. 지식은 프레임과 생성 규칙을 이용하여 표현함으로써 복잡한 구조의 지식을 표현하는데 용이하도록 한다. 현재 이 시스템은 UNIX의 명령어 중에서 10개를 대표적으로 선택했으며, 이 중 cp 명령어의 level-1, level-2, level-3에 대해서는 완전하게, 나머지 명령어는 개략적으로 X/MOTIF와 C를 이용하여 SPARC Station에서 구현하고 있다.

### Abstract

In this paper, we develop a prototype of ITS(Intelligent Tutoring Systems) system:UNIX TUTOR. It is designed for the purpose of teaching the UNIX beginners the principal concepts of UNIX and the shell commands using the communication between the student and the system. UNIX TUTOR engages the student in a two-way conversation, that is mixed-initiative dialogue, and attempts to teach the student UNIX via the Socratic method of guided discovery and the Coaching method interchangeably. And the student model is based on both the overlay model and the buggy model together. Thus TUTOR aims at teaching the students effectively whose levels of learning are different using various explanations which are determined by the student model. Because the knowledge representation for UNIX-TUTOR is based on the frame structure and the production rules, it is easy to represent the complicated constructs. UNIX TUTOR is implemented on the SPARC station using X/Motif and C for cp command among 10 ones which were selected.

\*正會員, 釜山外國語大學校 컴퓨터工學科  
(Dept. of Computer Eng. Pusan Univ. of Foreign Studies)

\*\*正會員, (株)聯合情報  
(United Information Systems, Inc.)  
接受日字 : 1993年 8月 15日

1. 서론

인공 지능을 교육 공학에 응용하는 ICAI(Intelligent Computer Assisted Instruction) 혹은 ITS(Intelligent Tutoring System)의 개발은 세계적인 추세이며 우리의 교육 현실에 비추어 볼때에도 그 필요성은 매우 크다고 하겠다. ITS 이전에 개발된 CAI시스템들은 전문가 지식을 전하는데만 관심을 가졌을 뿐이며 학습자가 배운 내용을 얼마나 알고 있으며, 또 학습자의 능력 정도에 따라 차등을 두어 교육하는 효과적인 면을 고려 하지는 못했다. CAI가 교육학자들에 의해서 설계되고 구현된 것에 반해서, ITS는 인공지능 학자들에 의해서 시도된 시스템이다.<sup>1,3,4,5,6,7</sup> ITS의 목적은 효율적인 교육보다도 인공지능 기법을 교육적인 측면에 적용하고자 하는것이 우선이었지만 CAI에서 발생한 여러문제를 극복하기 위한 노력을 하게 되었다. 그림 1은 ITS 시스템의 예이다.

시스템	제작자/년도	주제 영역	지식 표현	학습자 모델	교사 모델
SCHOLAR	Carbonell 1970	지리	의미 네트워크	Overlay	Socratic 대화
WIF	Stevens 외 1982	강우량의 한인	Scripts	Misconceptions	Socratic 대화
SOPHIE	Brom 외 1982	전자회로 고정진단	의미 네트워크	Overlay	Guided Interaction
BRGGY	Brown & Burton 1978	신수 설명	절차 네트워크	Misconceptions	조언자
RUSOR	Goldstein 1982	Logical Relations	Genetic Graph Net	Overlay	Coaching
UIS	Sleeman 1982	대수	Rule & mal-Rule	진단 규칙 기반	tutoring 기능 없음
QUILDON	Clancey 1982	MYCIN 교육	규칙 기반	Overlay	구조적 상호 작용
PROKST	Soloway 외 1983	PASCAL 프로그래밍	의미 네트워크	Misconceptions	조언자 (tutoring 기능 없음)

그림 1. ITS 시스템의 예  
Fig. 1. An example of ITS systems.

국내에서도 ITS의 중요성을 깨닫고 최근 몇년동안에 몇개의 시제품들이 만들어지고 있다.<sup>12,13,14,15,16</sup> 그러나 ITS와 관련한 국내의 연구로서는 아직까지도 주로 기초 연구에 머물고 있으며, 영역 전문가들도 쉽게 교재(course ware)를 제작할수 있게 해주는 저작 시스템이나, 영역의 전문가인 교사들에게서 전문 지식을 쉽게 획득할 수 있는 지식 획득 시스템<sup>5</sup> 등의 연구는 이제 시작 단계에 불과하다. 또한 학습자와 자연스럽게 대화할 수 있는 자연언어나 그래픽 등 멀티 미디어에 의한 학습자 인터페이스의 개발도 매우 필요하다 하겠다.

본 논문에서는 ITS의 특징중 상호주도적인 측면과

학습자 모델을 사용하여 일대일 교육을 수행하는 UNIX교육을 위한 지능형 개인교사 시스템 UNIX TUTOR를 구현하고 이를 위한 지식을 제시한다. 본 논문에서 제시하는 시스템은 시스템이 계획하고 진행하는 교육과정에 학습자가 수동적으로 교육을 받는 것이 아니라, 학습자가 교육과정에 적극적으로 참여하여 자신에게 적절한 교육을 요구하며 교육 단계까지 학습자가 주도할 수 있도록 한다. 즉 학습 정도가 낮고 수동적 교육이 적절한 학습자에게는 시스템이 제시하는 교육과정에 따라 교육 받도록하고, 학습 성취도가 높고 적극적인 학습자에게는 학습자의 요구에 따라 원하는 교육을 받도록 함으로써, 보다 효과적인 교육을 수행할 수 있도록한다. 학습자가 틀리는 근본 원인을 오버레이 모델과 버그 모델<sup>7</sup>에 함께 돕으로써 각 모델의 정보에 따라서 소크라테스법과 코치법, 그리고 재교육을 학습 전략으로 적절히 선택할 수 있도록 하였다. 즉 오버레이 모델로 저장된 오류에 대해서는 소크라테스법을 학습 전략으로 선택하고 버그 모델로 저장된 오류에 대해서는 코치법과 재교육을 학습 전략으로 선택하도록 하여서 학습자의 상태에 대해서 보다 적절히 판단하여 다양한 설명을 해 줄 수 있게 한다 본 논문은 II에서 제안하는 시스템의 전체 구성도를 보이고, 각 모듈의 기능과 특성에 관해서 논한다. III장에서는 UNIX TUTOR의 지식 표현에 대해서 논하며, IV장에서는 교육과정의 예와 구현 결과를 보이고, V장에서 결론을 내린다.

II. UNIX TUTOR의 설계

UNIX TUTOR는 기능에 따라 여러 모듈로 나뉘어져 있다. 그림 2는 UNIX TUTOR의 전체 구성도이다. 단, 여기서는 전체 구성도만 설명하고 구체적인 지식(영역, 교육, 진단 지식 등)은 3장에서 상세히 표현하고 있다.

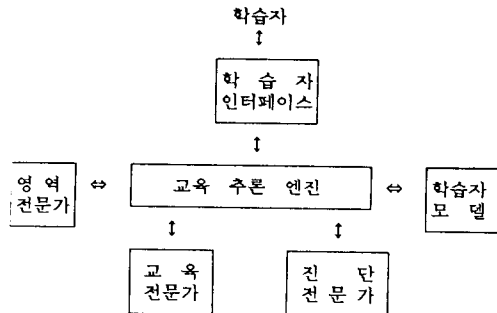


그림 2. UNIX TUTOR의 전체 구성도  
Fig. 2. The overall structure for UNIX TUTOR.

영역 전문가(DOMAIN EXPERT)는 학습자에게 가르치고자 하는 지식과 추론 기능으로 구성되어 있다. 지식은 전체 교육의 목표와 방법에 관한 것과 교육할 내용과 그리고 학습자에게 주어질 질문과 해답 그리고 오류 규칙(mal-rule)을 포함하고 있다. 오류 규칙은 질문에 대한 공통적인 실수와 오류를 오류 형태에 따라 생성 규칙으로 표현한 것이다.

교육 전문가(TUTORING EXPERT)는 학습 전략에 관한 지식과 추론 기능으로 구성되어 있다. 다음 학습 수행을 위한 학습전략의 선택은 계획의 수행 결과에 따른다. 이 계획은 2가지 행위를 허용한다. 한 가지는 표준 행위로서 계획 생성 과정에서 생성된 교육 계획에 따라 수행하는 것이다. 다른 하나는 예외적인 행위로서 학습자의 반응에 따라 결정된다. 계획 수행 과정에서 질문에 대한 학습자의 대답이 틀린 것으로 나타났을 때, 소크라테스법과 코치법중 어느것을 선택할 것인지 결정한다. 학습자의 대답이 틀린 경우에는 먼저 소크라테스법을 선택한다. 그러나 반복하여 같은 오류를 범했을 때, 시스템은 학습자가 잘못된 지식을 갖고 있다고 판단하여 코치 법에 따라 교육을 실시한다. 즉, 틀린 부분에 대한 개념과 문제 해결 방법을 설명하고 정답에 가까운 더 많은 힌트를 준다.

진단 전문가(DIAGNOSTIC EXPERT)는 학습자의 반응을 분석하는 모듈이다. 이 모듈은 진단지식과 추론 기능으로 구성된다. 이 모듈에서는 학습자의 대답과 영역 전문가가 제시하는 여러 해답과 서로 비교함으로써 학습자의 오류를 찾아낸다. 우선 학습자의 답을 분석하고, 분석된 학습자의 답과 전문가의 답을 비교한 다음, 필요한 설명을 한다. 학습자의 대답과 가장 유사한 전문가의 해답을 비교하여 불일치하는 부분을 학습자의 오류로 인식하여 학습자 모델에 저장한다.

학습자 모델(STUDENT MODEL)은 UNIX TUTOR에 의해 제공되어진 지식을 학습자가 어느정도 인식하였는가 하는것과 교육과정이 어느 정도 진행되었는가를 나타낸다. 학습자 모델은 오버레이 모델과 버그 모델을 함께 사용하여 학습자가 알되 불완전하게 아는것과 잘못 알고 있는 것을 표현한다. 학습자가 틀리는 근본 원인을 오버레이 모델과 버그 모델 둘다에 둬으로써 각 모델의 정보에 따라서 소크라테스법과 코치법, 그리고 재교육을 학습 전략으로 적절히 선택할 수 있도록 하였다. 학습자가 1-2회 틀린 경우에는 학습자는 전문가의 지식 중 일부만 익혔기 때문이라고 판단하여 소크라테스법에 의하여 틀린 것을 설명하고 유사질문을 2회 까지 반복하고,

Over\_Bug\_Error에 오류 회수를 1-2회로 기록하고 유사한 오류를 3-4회 까지 틀린 경우에는 학습자는 전문가의 지식에는 없는 엉터리 지식을 습득 했는것으로 간주하여 교육 방법을 코치 방법으로 변경한다. 즉 교육 전문가에서 학습자가 틀린 이유를 영역 전문가에 있는 오류 규칙에 의해서 추론해서 근본적인 오류 원인과 정답에 가까운 힌트를 준다. 이들의 힌트는 범하기 쉬운 공통적인 실수와 오류를 오류 규칙(mal-rule) 형태로 생성 규칙으로 영역 전문가에 저장되어 있다.

교육 추론 엔진(TUTOR INFERENCE ENGINE)은 전체 교육 과정을 통제하는 모듈이다. 또한 각 모듈과 정보를 교환함으로써 교육 과정을 수행한다. 이 모듈은 학습자의 반응을 다른 모듈에 전달하고 그 결과를 학습자에게 제공하여 준다. 그리고 오류가 발생했을 때에는 그 오류를 학습자 모델에 저장한다.

학습자 인터페이스(STUDENT INTERFACE)는 학습자와 교육 추론 엔진과의 대화를 담당하는 모듈이다. 이 모듈은 ProblemWindow, Message Window, UserInputWindow등 3가지의 윈도우를 통해서 대화를 진행한다. 그림 3은 X 환경위에서 수행되는 학습자 인터페이스를 보여주고있다. Problem Window는 학습을 위한 문제를 제출하는 윈도우이다. MessageWindow는 학생의 대답에 대한 시스템의 반응을 출력한다. UserInputWindow는 학생의 대답을 입력받는데 사용되는 윈도우이다.

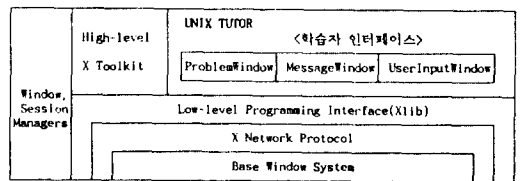


그림 3. UNIX TUTOR 학습자 인터페이스  
Fig. 3 The interface for UNIX TUTOR.

그림 4는 지능형 개인교사 시스템 UNIX TUTOR의 전체 수행 알고리즘이다.

ALGORITHM : UNIX TUTOR

1. Introduction 인쇄 : 영역 전문가 지식 이용
2. Student\_Name 입력 : 학습자 인터페이스 이용
3. Standard\_Plan을 보여주고, 필요한 경우 : 학습자 인터페이스 이용  
학습자로 부터 입력을 받아 표준계획 수정
4. Student\_Model을 초기화

5. Teaching\_Plan의 끝 혹은 학습자로 부터의 종료 인터럽트를 받을때 까지 다음을 반복
  - 1) Teaching\_Plan에 의해 교육 : 영역 전문가 이용
  - 2) 학습자의 응답을 진단 : 진단 전문가 이용
  - 3) Student\_Model 수정 : 교육 계획 수정 혹은 오류를 학습자모델에 수록
  - 4) 학습자 모델의 결과에 따라 학습 : 교육, 영역 전문가 이용 전략을 적용하고, 필요한 경우 재계획

그림 4. UNIX TUTOR의 전체 수행 알고리즘  
Fig. 4. The overall algorithm for UNIX TUTOR.

UNIX TUTOR는 먼저 가르칠 내용을 간단히 소개 하고 나서 학습자의 이름과 필요한 경우 학습자가 원하는 교육 계획을 입력 받는다. 학습자가 교육 계획을 입력 하지 않을 경우에는 표준 계획에 의하여 교육한다. 다음으로 학습자 모델을 생성하여 교육 계획을 저장하고 학습자 모델의 교육 계획에 따라 교육을 진행한다. 학습자의 반응을 입력받고 진단 과정을 수행한다. 결과를 학습자 모델에 저장한후 이 결과에 따라서 교육 전문가는 다음 교육 과정의 학습 전략을 결정한다. 이때 필요하다면 교육 계획을 수정한다. UNIX TUTOR는 더 이상 교육할 내용이 없든지 혹은 학습자로부터의 종료 인터럽트가 발생할때 까지 위의 과정을 반복 수행한다.

### III. UNIX TUTOR의 지식 표현

계획 수립은 ITS에서의 주요 관심사 중의 하나이다.<sup>[8,10,11]</sup> 본 논문에서는 개인에게 적절한 계획을 생성하여 수행하며 필요한 경우 시스템이나 학습자가 직접 계획을 수정할 수 있도록 하였다.

#### 1. 표준 계획 및 계획의 수정

그림 5는 UNIX TUTOR의 표준 계획이다. 학습자는 교육 과정의 생략과 단계의 변경과 순서의 변경을 요구할 수 있다. 표준 계획은 교육 과정 수행 중에도 변경가능하다. 그러나 학습자로부터 아무런 요청이 없으면 TUTOR는 표준 계획에 기초하여 교육 계획을 수립한다. 지식은 프레임으로 표현되고 있으며 각 노드는 한개의 프레임으로 표시된다. 표준 계획은 위에서 부터 단계1 - 단계4 까지 나누어 지고 있다. 단계 1은 전체 교육의 목표와 방법에 관한 지식이며, 단계 2,3은 중간 단원의 목표와 방법에 관한 지식이고 각각 장(chapter)과 절(session)로 부르

고, 마지막 단계 4는 최종 교육할 내용에 관한 지식과 학습자에게 주어질 질문과 해답 그리고 오류 규칙을 포함하고 있다.

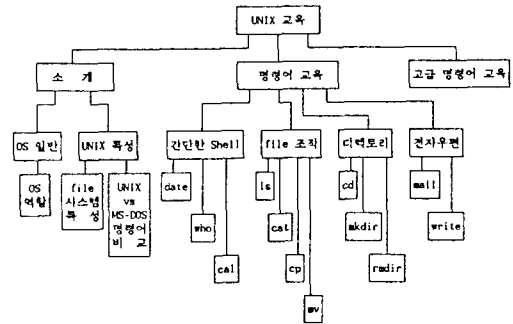


그림 5. UNIX TUTOR의 표준 계획  
Fig. 5. The standard plan for UNIX TUTOR.

계획의 수정은 학습자의 요구에 의해서도 이루어질 수 있다. 학습자가 교육 과정을 변경하면 현재 교육 중이던 명령어에 해당하는 학습자 모델 정보를 화일로 저장해 두고 변경된 교육 과정으로 간다. 이때 분기하고 싶은 명령어와 관련된 정보가 이미 화일에 있는 경우는 화일로부터 정보를 학습자 모델에 적재한다. 또한 학습자 모델의 오류에 따라 시스템이 개인의 교육 계획을 교육 전문가의 교육 규칙에 따라서 수정할 수 있다. 대표적인 메타 규칙은 그림 6과 같다.

#### TM04-Unix vs Dos : a Tutoring meta rule

IF

UNIX 명령어를 DOS명령어와 2번 이상 혼동하는 경우

Then

"UNIX vs DOS 명령어비교" 과정을 추가로 교육한다.

#### TM05-File system : a Tutoring meta rule

IF

path, source 그리고 destination에 대해 2번 이상 오류를 범할 때

Then

"File System" 과정을 교육한다.

그림 6. 교육 계획의 수정을 위한 메타 규칙의 예  
Fig. 6. An example of meta rules for modifying the standard plan.

UNIX TUTOR가 주도하는 계획이나 학습자의 요

구에 의해 변경된 계획은 학습자 모델에 저장되고, 변경된 교육 계획에 따라 교육이 계속 진행된다.

2. 지식 표현

영역 전문가 모듈의 지식은 프레임 구조로 나타난다. 그림 7은 cp 명령어에 관련된 프레임 시스템을 보여주고 있다.

< UNIX 명령어 >

```
description : 교육 소개
Chapter : file 조작 명령어
          dir 명령어
          기타 명령어
```

< file 조작 명령어 >

```
isa : < UNIX 명령어 >
description : global explanation
Session : cp
          mv
          ls
          cat
```

< cp >

```
isa : < file 조작 명령어 >
name : cp
description : cp 소개
synopsis :
associated command :
similiar command :
re-teach range :
level-list : level_1
              level_2
              level_3
```

< level-*i* >

```
instance : < cp >
description :
question :
solution :
mal-rule :
current directory :
```

그림 7. 프레임 시스템으로 표현된 cp 명령어에 관련된 영역 지식

Fig. 7. The domain knowledge for <cp> command which is represented by a frame system.

그림 7에서 보면 프레임 <cp>의 슬롯 <isa>는 상위 프레임 <file 조작 명령어>를 가리킨다. 그리고 슬롯

<name>은 이 프레임으로 표현하는 객체의 이름을 나타낸다. 슬롯 <synopsis>는 일반적인 cp 명령어의 형식이다. 슬롯 <associated command>는 이 명령어와 관계가 있는 다른 명령어를 나타낸다. 슬롯 <similar command>는 이 명령어와 유사한 다른 시스템의 명령어를 나타내고, 슬롯 <re-teach range>는 교육 과정 중에서 일정 횟수 만큼의 오류를 범했을 경우 재교육 하는 범위를 나타낸다. 여기서 프레임 <level1>, <level2>, 그리고 <level3>는 학습자에게 교육할 내용과 그 내용에 필요한 질문등을 포함하는 프레임이다. <level-*i*> 프레임의 슬롯 <mal-rule>은 학습자가 범하기 쉬운 공통적인 실수와 오류를 bug-catalog 형태로 생성 규칙으로 표현 한것이다. 이는 교육 방법이 소크라테스법에서 코치법으로 변경될때 참조된다. 프레임 구조의 접근 방법은 일반적인 프레임 시스템에서의 속성 상속 알고리즘<sup>[2,9]</sup>을 따른다.

학습자 모델의 지식 표현도 프레임 구조이다. 학습자에 대한 정보를 나타내는 모델로서는 오버레이 모델과 버그 모델 등이 있는데, 본 논문에서는 이 두 모델을 모두 사용하여 Over\_Bug\_Error 모델을 사용한다. 그림 8은 한 학습자가 cp 명령어를 학습하고 있는 상황에서 학습자 모델에 나타난 정보를 보여 주고 있다.

<student model : Hong, Kil-Dong>

```
identification :
plan :
taught plan : 현재까지 교육 받은 명령어 id
current command :
current level :
Over-Bug-Error : [Over-Bug-1],[Over-Bug-2],...,[Over-Bug-16]
```

< Over-Bug-Error-*i* >

```
instance : <Hong, Kil-Dong>
identification : Over-Bug-3
count :
```

그림 8. 프레임 형태로 나타낸 학습자 모델

Fig. 8. A student model which is represented by a frame structure.

TUTOR와 학습장에 의해 생성된 교육계획은 슬롯 <plan>에 저장된다. 이 슬롯은 시스템 또는 학습자의 요구에 의해 변경된 계획을 저장하기 위해 사용된다.

진단 전문가는 오류를 인식하기 위한 진단 지식이 존재한다. 오류는 구문 오류를 비롯해서 모든 명령어 사용에 있어서 공통적으로 발생할 수 있는 일반 오류와, 특정 명령어에서만 발생할 수 있는 영역 오류로 나누어진다. 그림 9는 오류의 종류에 관한 설명이다. 진단 지식은 생성 규칙으로 나타낸다.

	오류내용
0	최적 해(best solution)가 아닌 해(solution)인 경우
1	명령어의 형식이 잘못된 경우
2	교육중인 명령어의 유사 명령어 사용
3	다른 명령어의 유사 명령어 사용
4	표준 계획에 없는 명령어 및 유사 명령어에 없는 명령어 사용 (해당 명령어가 없는 경우)
5	명령어의 대, 소문자를 구분하지 않고 사용한 경우
6	option 사용시 '-'를 생략한 경우
7	option 사용시 '-'가 아닌 다른 문자를 사용
8	필요한 option 생략
9	불필요한 option 사용
10	명령어와 원시 파일/option을 분리하지 않고 사용한 경우
11	'/' 대신 '\'를 사용한 경우
12	원시 파일/디렉토리의 명을 잘못 적은 경우
13	목적 파일/디렉토리의 명을 잘못 적은 경우
14	원시 파일/디렉토리를 생략한 경우
15	목적 파일/디렉토리를 생략한 경우
16	원시 파일/디렉토리과 목적 파일/디렉토리가 바뀐 경우

그림 9. 오류의 종류  
Fig. 9. The types of errors.

교육 전문가는 학습 전략에 관한 지식을 가지고 있다. 학습 전략에 관한 지식은 학습자의 다음 교육 과정을 어떤 학습 전략에 따라 할 것인지, 그리고 무엇을 교육할 것인지에 관한 지식이다. 학습 전략에 관한 지식은 생성 규칙 시스템으로 나타난다. 학습 전략은 크게 소크라테스법, 코치 법, 그리고 재교육 등으로 구성된다. 소크라테스법은 학습자의 오류에 대해 현재 명령어의 슬롯 <description>을 다시 설명해 주고 어떤 새로운 문제를 제시 할 것인가를 나타낸다. 그리고 코치법은 학습자의 오류에 대해 정확한 개념과 문제를 해결하는 방법에 대한 힌트를 가지고 있다. 관련된 학습 전략 규칙은 T01-Socratic - T06-Help(그림 10) 이다. 또한 학습자 모델의 성취도에 따라서 표준 계획을 수정해주는 메타 규칙을 가지고 있다(TM04, TM05 메타 규칙 (그림 6)).

T01-Socratic : a Socratic tutoring rule  
IF  
    <cmd-i level-j>가 Over\_Bug\_Error 슬롯의 값이 1 이면  
THEN  
    현재 level의 <description>을 설명한다

T02-Socratic : a Socratic tutoring rule  
IF  
    <cmd-i level-j>가 Over\_Bug\_Error 슬롯의 값이 2 면  
THEN  
    a) 현재 노드의 상위 노드인 현재 명령어의

<description>을 설명하고 AND  
b) 앞선 단계가 있으면 이 단계(들)의 <description>을 설명하고 AND  
c) 현재 단계의 <description>을 설명한다

T03-Coach : a Coaching tutoring rule  
IF  
    <cmd-i level-j>가 Over\_Bug\_Error 슬롯의 값이 3 이면  
THEN  
    a) 영역 전문가에 있는 오류 규칙(mal-rule)에 의거 오류 원인을 설명하고 AND b) 정답에 관련된 힌트를 준다

T04-Coach : a Coaching tutoring rule  
IF  
    <cmd-i level-j>가 Over\_Bug\_Error 슬롯의 값이 4 면  
THEN  
    a) Domain Expert에 있는 mal-rule에 의거 오류 원인을 설명하고 AND  
    b) 현재 노드의 상위 노드로 가서 현재 명령어의 <description>과 관련 명령어(associated command)의 <description>을 비교 설명하고 AND  
    c) 정답을 가르쳐준다

T05-Re-Teach : a Re-Teaching tutoring rule  
IF  
    <cmd-i level-j>가 Over\_Bug\_Error 슬롯의 값이 5 면  
THEN  
    a) Over\_Bug\_Error 슬롯의 값을 0으로 하여 오류를 삭제하고 AND b) Domain Expert의 re-teach range 슬롯의 값에 따라서 plan 슬롯의 값을 재조정(replan)하여 재교육을 한다

T06-Help : a Help tutoring rule  
IF  
    학습자로 부터 개념이나 주요어(keyword)에 대한 질문, 현재의 질문에 관한 도움 요청, 혹은 임의 계획 변경 요청(help)이 있으면  
THEN  
    a) 계획 변경인 경우 학생 모델 내용 변경 명령을 내리고  
    b) Tutoring Expert의 planner가 관련된 프레임을 선택하고  
    c) 그 프레임의 상위 노드인 장(chapter) 까지 각각의 <description>을 설명한다

그림 10. 학습 전략 규칙의 예  
Fig. 10. An example of tutoring rule.

그림 11은 오류 규칙에 관한 메타 규칙의 대표적인 예를 보여 주고 있다.

TM02-Meta-rule : a meta rule for mal-rules

IF

2번 발생한 오류의 종류가 2개 이상일때

THEN

- a) 현재 적용 가능한 규칙을 적용하고 AND
- b) 적용 가능 규칙이 T01 - T02 이면 오류 규칙에 의거 오류 원인을 설명한다

TM03-Meta-rule : a meta rule for mal-rules

IF

서로 다른 유형의 오류가 3번 이상 발생하면

THEN

- a) 현재 적용 가능한 규칙을 적용하고 AND
- b) 적용 가능 규칙이 T01 - T02 이면 오류 규칙에 의거 오류 원인을 설명한다

그림 11. 오류 규칙에 관한 메타 규칙의 예  
Fig. 11. An example of meta rules for mal rules.

교육시 다음 단계로 넘어가는 규칙은 다음과 같다.

- 1) 기본 점수 5점을 준다.
- 2) 최적 해답을 한 경우에는 +2점을, 최적 해답이 아닌 경우에는 +1점을, 그리고 틀린 경우에는 -1

점을 준다.

3) 8점 이상은 다음 단계로 넘어간다.

4) 8점 미만은 4문제 까지 반복 출제한다.

#### IV. UNIX TUTOR의 예와 구현

시험적으로 10개의 명령어를 선별하여 구현한다. 구현을 위한 언어로는 SPARC의 X/Motif와 C를 사용한다. UNIX TUTOR는 계획의 수행에 앞서 전체 과정의 목표를 보여준다. 그리고 표준 계획을 제시하며 학습자로 부터 표준 계획을 수정시켜 교육 계획을 생성한다. 생성된 교육 계획에 따라 세부 과정을 수행한다. 먼저 각 명령어의 개념을 설명하며 학습자에게 준비된 질문을 한다. 이 질문에 대한 학습자의 문제 해결과정이 잘못되었으면 시스템은 그것을 지적하고 학습자가 잘못된 것을 인식하고 고치도록 다른 문제를 제출한다. 만약 학습자가 도움을 요청하면 문제 해결 방법을 설명하고, 동일한 주제에 관한 다른 문제를 제출하여 문제 해결 능력을 갖추도록 돕는다.

그림 12는 UNIX TUTOR의 초기 화면으로서 루트 윈도우는 ProblemWindow, MessageWindow, UserInputWindow로 구성된다. ProblemWindow는 학습을 위한 문제를 제출하는 윈도우이다. MessageWindow는 학생의 대답에 대한 시스템의 반응

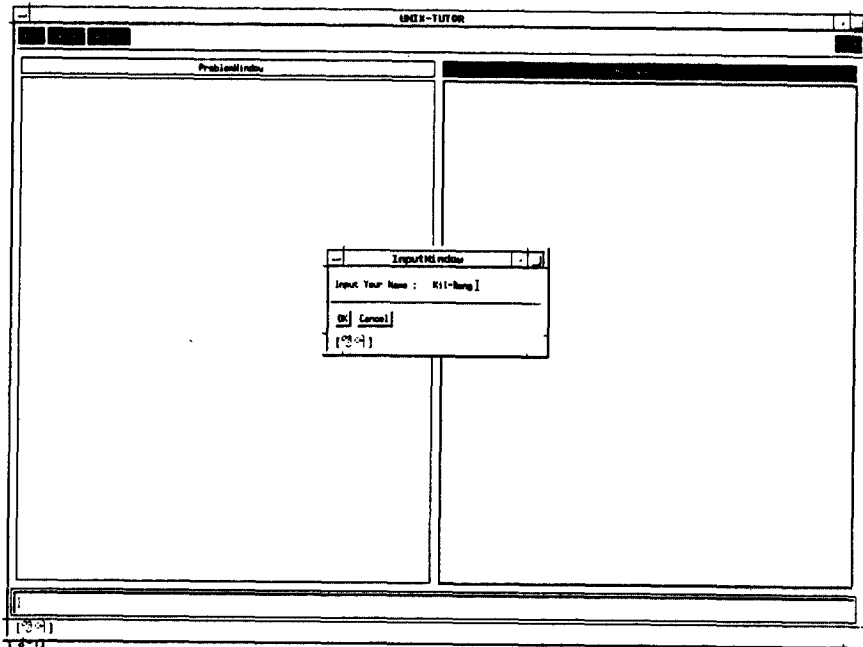


그림 12. UNIX TUTOR의 초기 화면  
Fig. 12. The beginning Window of UNIX TUTOR.

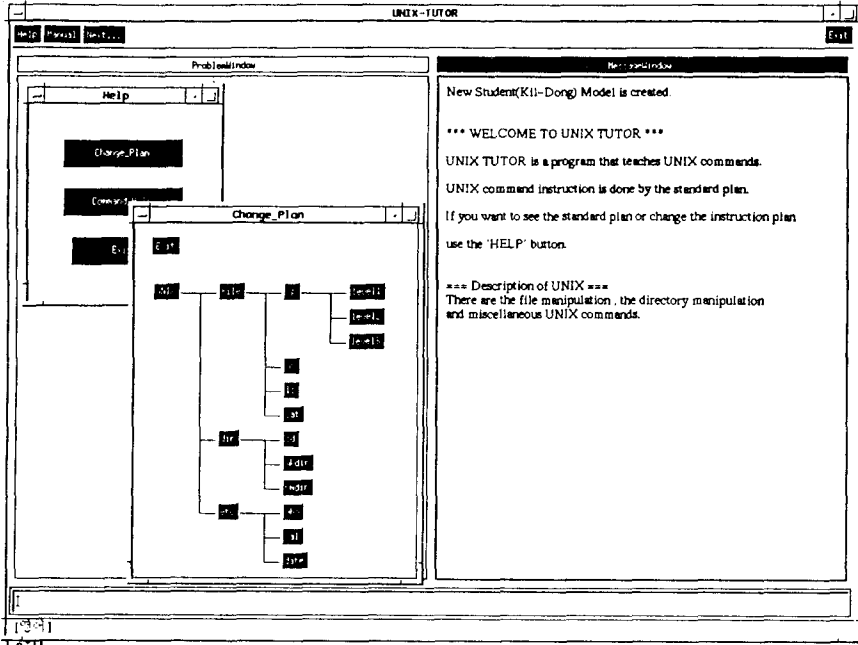


그림 13. Change Plan 윈도우  
Fig. 13. The Window for Change\_Plan.

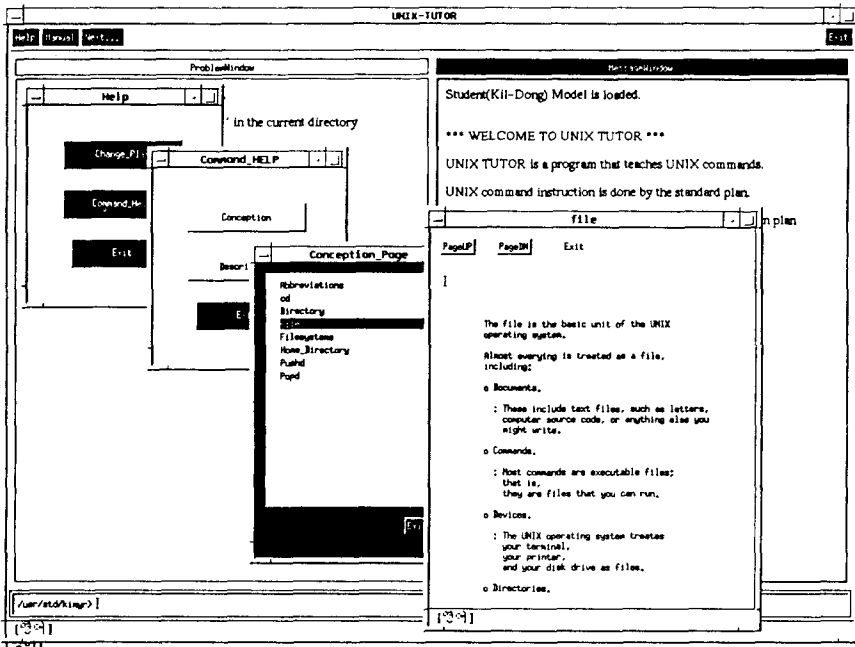


그림 14. 주요어 "File" 에 관한 윈도우  
Fig. 14. The window for the keyword "File".



을 출력한다. UserInputWindow는 학생의 대답을 입력받는데 사용되는 윈도우이다. 제일 아래에 위치한 UserInputWindow의 title bar는 생략되었다. 처음으로 UNIX TUTOR가 기동되면 InputWindow가 생성되고, 여기서 학습자의 이름을 입력받아서 학습자 모델을 만들고, 앞으로의 학습 과정에서의 학습 내용에 관한 정보를 기억하게 된다.

그림 13은 학습자의 이름을 입력받고 난 다음, MessageWindow로 부터의 설명에 의해서 Help 버튼을 누른 다음, Help 윈도우에서 Change\_Plan을 선택한 경우이다. Change\_Plan 윈도우는 표준 계획을 보여주고 학습자로 부터 계획 수정이 필요한 경우에는 마우스를 통해서 입력 받는다. (규칙 T06-Help)

그림 14는 학습자로 부터 개념이나 주요어에 관한 질문을 받았을때의 화면이다. 이 경우에는 Help::Command-Help::Conception-Page::File 윈도우를 차례로 선택한 경우인데, File에 관한 설명을 학습자에게 설명해 주고 있다.

```

U-TUTOR01> Welcome to UNIX TUTOR!!!
          Log in your name, please.
student01> Kil-Dong Hong      :: Student Model 작성
U-TUTOR02> :: 교육 내용 설명, 표준 계획 설명
          :: 시작 하고 싶은 번호 ? DEFAULT : #1
student02> :: file 조차의 cp는 선택한 경우 기법
U-TUTOR03> :: cp 명령어에 대한 <description> 설명
          Your current directory is "/users/mdch"
          Copy "/users/mdch/exam/ex1.c" onto "t1.c" in the current directory
student03> cp ex1.c
U-TUTOR04> It is wrong.
          Found syntax error.
          Synopsis : cp option from_file_to_file
          Try again!
student04> cp ex1.c t1.c
U-TUTOR05> It is wrong.
          There is no path.          :: 12 번 오류
          :: T01-Socratic 적용
          The path of "ex1.c" is "/users/mdch/exam" or "exam"
          The path of "t1.c" is "/users/mdch" or "."
          :: 현재 명령어와 현재 단계의 <description> 설명
          Copy "/users/mdch/exam/ex3.c" onto "/users/mdch/its/t3.c"
student05> cp exam/ex3.c t3.c
U-TUTOR06> It is wrong          :: 12번 오류를 3번 범했다고 가정
          :: 교육 규칙 T03-Coach 적용, mal-rule에 의거 오류 원인 설명
          :: 정답과 관련된 힌트를 준다.
          You must know the source and the destination.
          The source is "/users/mdch/exam/ex3.c" or "exam/ex3.c".
          The destination is "/users/mdch/its/t3.c" or "its/t3.c".
          Copy the source onto the destination.
          Copy "/users/mdch/exam/ex4.c" onto "/users/mdch/its/t4.c"
student06> help
  
```

그림 15. <U-TUTOR>와 <student> 사이의 학습 내용  
Fig. 15. A tutoring scenario between <U-TUTOR> and <student>.

그림 15의 과정은 cp 명령어에 대한 개념을 설명한 후에 진행되는 과정이다. 여기서 U-TUTOR는 문제를 제출한다. U-TUTOR는 학습자로 부터 질문에 대한 답을 학습자 인터페이스를 통해 가져 온다. 그 답을 TUTOR가 명령어 형식에 맞는지 조사한다. <student03>에서는 구문 오류가 발생했기 때문에 오류를 지적하고 조금전의 과정을 반복한다. <student04>에서는 학습자의 답이 구문 오류가 아니므로 U-TUTOR의 답과 비교하여 가장 유사한 것을 선택하여 비교한다. 일치하지 못하는 부분이 원시 화일/디렉토리를 잘못 지정한 오류(12번 오류)임을 발견한다. 이 과정은 진단 전문가에 의해서 이루어진다. 이 결과를 교육 전문가가 받아서 다음 과정을 결정한다. U-TUTOR는 이것을 학습자가 완전하지 못한 지식을 가졌기 때문이라고 판단한다. 그래서 학습자가 이 오류를 인식하고 해결할 수 있도록 메타 규칙 T01-Socratic에 의해서 현재 명령어와 현재 단계의 <description>을 설명하고 유사한 문제를 제시한다. 12번 오류를 3번째 범했다고 가정하면 T03-Coach에 따라 U-TUTOR는 학습자가 잘못된 지식을 가지고 있다고 판단하여 코치법에 의해 문제의 개념과 해결 방법에 관한 힌트를 준다. <student06>에서 제출된 문제에 대해서 정답을 한 경우에는 유사한 문제를 다시 출제하여 최종 점수가 8점이 될때까지 반복하지만 help를 요청했으므로 T06-help의 규칙에 따라서 교육 전문가가 학습자 인터페이스를 통하여 도움 요청의 범위를 학습자로 부터 확인하여 관련 <description>을 설명한다. 만약 학습자가 유사한 오류를 5번 반복 하는 경우에는 현재 명령어의 재교육 범위에 따라서 재교육 한다.

본 논문에서 제시하는 시스템은 cp date, who, cal, ls, 등 5개의 명령어에 대해서 교육을 하도록 830라인의 SCLISP으로 구현되었었는데, 그후 X/Motif와 C(약 3800라인)를 사용하여 cp 명령어의 level-1, level-2, level-3에 대해서 구현하고 있다. 나머지 명령어에 대해서는 영역 지식을 아직 구축하고 있지 않으나, 개략적인 구현은 되어있다.

현재 구현된 시스템의 학습자 인터페이스는 UNIX 명령어와 같이 형식이 분명한 경우에만 적용할 수 있다. 그래서 전체 교육 과정 중 명령어 교육 과정만을 구현 하였다. OS일반과 화일 시스템 교육 과정은 구현 범위에 넣지 않고, 단지 명령어 교육을 하는 중에 OS 일반과 화일 시스템과 관계되는 부분만 독립적인 프레임으로 구현하고 있다. 그러나 앞으로 O.S나 화일시스템과 같이 형식이 불분명한 경우의 교육도 수행할 수 있도록 시스템을 확장 하고자 한다.

본 시스템에서 사용한 교육 방법을 기존의 CAI(e. g. CAROL-C [17]) 등과 비교해보면, 기존의 CAI는 대개 문제 제출, 학습자 응답, 힌트, 정답으로 이어지는 일률적인 교육 방법인데 비해서 UNIX-TUTOR는 학습자 모델을 효과적으로 이용하여 다양한 교육 전략을 구사할 수 있다. 또한 기존의 ITS와의 차이점은 기존의 ITS에서는 학습자 모델을 주로 오버레이 모델이나 버그 모델 등 한가지 방법에 둬으로써 학습 전략을 다양하게 구사하기 힘들고 학습 전략의 융통성이 부족했는데 UNIX TUTOR에서는 학습자가 틀리는 근본 원인을 오버레이 모델과 버그 모델 둘다에 둬으로써 각 모델의 정보에 따라서 소크라테스법과 코치법, 그리고 재교육을 학습 전략으로 적절히 선택할 수 있도록 하였다. 즉 오버레이 모델로 간주되는 오류에 대해서는 소크라테스법을 학습 전략으로 선택하고 버그 모델로 간주되는 오류에 대해서는 코치법과 재교육을 학습 전략으로 선택하도록 하여서 학습자의 상태에 대해서 보다 적절히 판단할 수 있도록 하였다. 단, 본 논문에서는 교육 공학적인 측면에서의 연구는 수행하지 못했음을 밝혀둔다. 현재 구현된 cp 명령어의 level-1, level-2, level-3에 대해서 시험 수행 해본 결과 이들 level에 대해서 발생할 수 있는 16개의 오류에 대해서는 진단을 하고, 학습자 모델에 발생한 오류를 저장하였다. 그리고 이 내용에 따라서 학습 전략을 소크라테스법, 코치법, 재교육 등으로 바꾸어가며 교육할 수 있었다.

## V. 결론

본 논문에서는 ITS의 특징중 상호주도적인 측면과 통합된 학습자 모델을 사용하여 일대일 교육을 수행하는 UNIX교육을 위한 지능형 개인교사 시스템 UNIX TUTOR를 구현하고 이를 위한 영역 지식, 교육 지식 등을 제시하였다. 제시하는 시스템은 시스템이 계획하고 진행되는 교육과정에 학습자가 수동적으로 교육을 받는 것이 아니라, 학습자가 교육과정에 적극적으로 참여하여 자신에게 적절한 교육을 요구하며 교육 단계까지 학습자가 주도할 수 있도록 하였다.

기존의 CAI나 ITS 시스템과 비교해 보면, 기존의 CAI는 일률적인 교육 방법인데 비해서 UNIX-TUTOR는 학습자 모델을 효과적으로 이용하여 다양한 교육 전략을 구사할 수 있다. 또한 기존의 ITS에서는 학습자 모델을 주로 오버레이 모델이나 버그 모델 등 한가지 방법에 둬으로써 학습 전략을 다양하게 구사하기 힘들었는데, UNIX-TUTOR에서는 학습자 모델을 오버레이 모델과 버그 모델을 하나의 모델

Over\_Bug\_Error로 통합하여 이 모델의 정보에 따라서 소크라테스법, 코치법, 재교육 방법을 학습 전략으로 적절히 선택할 수 있도록 하였다. 이에 따라 학생의 답이 틀린 경우 통합된 학습자 모델의 정보에 따라서 다양한 설명 기능을 제공 해줌으로써, 학습 성취도가 서로 다른 여러 사람들을 적절히 교육하는 효과를 얻을 수 있다. 지식은 프레임과 생성 규칙을 이용하여 표현하였으므로 복잡한 영역의 구조 표현이 용이해졌다.

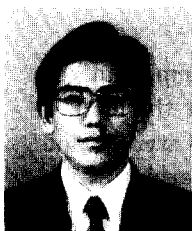
현재 이 시스템은 UNIX의 명령어 중에서 10개를 대표적으로 선택하여 이 중 cp 명령어의 level-1, level-2, level-3에 대해서 SPARC STATION의 X/Motif 와 C를 이용하여 구현 하고 있으며, 나머지 명령어에 대해서는 개략적인 구현을 하고있다. 앞으로 인터페이스 부분에 하이퍼텍스트 개념을 추가하면 보다 편리한 학습자 인터페이스가 될것이다. 본 시스템의 구현을 완료한 다음에는 UNIX와 같이 문법이 분명한 영역이 아닌 문법이 불분명한 영역으로도 확장할 수 있을것이다.

## 參 考 文 獻

- [1] A. Barr and E.A. Feegenbaum, *The Handbook of Artificial Intelligence*, Vol. 2, William Kaufmann, CA, 1982.
- [2] P. Jackson, *Introduction to Expert Systems* 2nd ed, Addison-wesley, Reading, Mass., 1990.
- [3] W.L. Johnson and E. Soloway, "PROUST : An Automatic Deebugger for Pascal Programs," in Chap 3, G. Kearsley, *Artificial Intelligence and Instruction : Application and Method*, Addison Wesley, Reading, Mass., 1987.
- [4] G. Kearsley, *Artificial Intelligence and Instruction : Application and Method*, Addison Wesley, Reading, Mass., 1987.
- [5] T. Murray, et al, "A Knowledge Acquisition Tool for Intelligent Computer Tutors," *SIGART*, Vol. 2, No. 2, 1991, pp. 9-21.
- [6] O. C. Park, "Intelligent CAI: Old Wine in New Bottles, or a New Vintages?," in Chap 2, G. Kearsley, *Artificial Intelligence and Instruction : Application and Method*, Addison

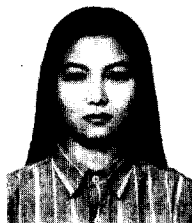
- Wesley, Reading, Mass., 1987.
- [7] D.Sleeman and J.S.Brown, *Intelligent Tutoring Systems(Eds)*, Academic Press, New York, N. Y., 1982.
  - [8] T.Sokolnicki, "Intelligent Tutoring Systems - Craft or Technology?," in *Technical Report LiTH-IDA-88-33*, 1988.
  - [9] E.Rich and K.Knight, *Artificial Intelligence*, McGraw-Hill, NY, 1991.
  - [10] B.Woolf and D.D.McDonald, "Building a Computer Tutor:Design Issues," *IEEE COMPUTER*, Vol.17, No. 9, 1984, pp. 61-73.
  - [11] B.P.Woolf, "Theoretical Frontiers in Building a Machine Tutor," in Chap 10, G.Kearsley, *Artificial Intelligence and Instruction : Application and Method* G.Kearsley(eds), Addison Wesley, Reading, Mass., 1987.
  - [12] 김 문 규, "컴퓨터 학습용 코스 웨어 개발 현황과 발전 방향," 정보과학회지, 7권, 3 호, 1989, pp.36-44.
  - [13] 김 진 형, "지능적 교사 시스템," 한국과학재단 학.연.산 연구 교류회 발표 요약문, Apr. 1992.
  - [14] 신 교 선, 정 목 동 "UNIX 교육을 위한 지능형 개인교사 시스템의 설계," 한국인지과 학회 춘계발표 논문집, 1992, pp.232-242.
  - [15] 이 옥 화, "교육용 소프트웨어(CAI)의 개발과 인공지능의 역할," 전산교육 연구회 춘 계 전산 교육 심포지움 발표 논문집, 1989, pp. 1-13.
  - [16] 장 경 식, 박 충 식, 김 재 희, "MS-DOS 학습용 지능적 교육 시스템," 전자공학회논문지, 제28권 B편, 제3호, pp. 143-150, 1991년 3월.
  - [17] 코리아 컴퓨터(주), "CAI CAROL, CAROL-C 조작 안내서," 코리아 컴퓨터(주), 1993.

著 者 紹 介



鄭 木 董(正會員)

1981年 2月 경북대학교 전자공학과 전산전공 (공학사), 1983年 2月 서울대학교 대학원 컴퓨터공학과 (공학석사), 1990年 8月 서울대학교 대학원 컴퓨터공학과 (공학박사), 1985年 3月 ~ 현재 부산외국어대학교 컴퓨터공학과 전임강사, 조교수, 부교수. 주관심 분야는 Intelligent Tutoring System, 지식 기반 시스템, HCI 등임.



金 容 蘭(正會員)

1991年 2月 부산공업대학교 전자계산학과 (공학사), 1994年 2月 부산외국어대학교 컴퓨터공학과 (공학석사), 현재 동명공업전문대학 전산학과 강사.



金 英 成(正會員)

1989年 2月 부산외국어대학교 컴퓨터공학과 (공학사), 1994年 2月 부산외국어대학교 컴퓨터공학과 (공학석사), 1994年 ~ 현재 소프트웨어하우스 근무 등임.



辛 敦 善(正會員)

1991年 2月 부산외국어대학교 컴퓨터공학과 (공학사), 1993年 2月 부산외국어대학교 컴퓨터공학과 (공학석사), 1993年 2月 ~ 현재 (주) 연합정보 연구원