대화형 지능시스템을 위한 WWW 인터페이스의 구현

김 창 민† · 김 용 기††

요 약

WWW(World Wide Web)의 비약적인 성장과 더불어 각종 인터넷 서비스 공급자(ISP, Internet Service Provider)의 기능과 역할이 크게 부각되고 있다. WWW을 인터페이스로 사용하는 대화형 지능시스템은 사용자와 대화를 통하여 보다 정확히 사용자의 요구를 인식할 수 있으며, 특히 전문가시스템은 사용자에게 일선의 전문기와 같은 양질의 정보를 공급할 수 있다. 본 연구는 WWW를 사용자 인터페이스로 사용하는 기존의 지능시스템이 가지고 있는 문제점을 해결하기 위하여 개발 환경에 독립되고, 클라이언트의 부담을 최소로 하며, 텍스트 기반의 통신을 지원하는 대화형 지능시스템 사용자 인터페이스를 제안한다.

Dept. of Computer Science, Information and Communication Research Center, Kyungsang National University

Chang-Min Kim† · Yong-Gi Kim††

ABSTRACT

With the rapid increase of WWW uses, the role and function of ISP(Internet Service Provider) becomes more important. Interactive intelligent system using WWW as its user interface recognizes a user's intention more correctly. Expert system on WWW, therefore, can provide users with better information like a human expert working on the site. The paper suggests a technique of user interface for interactive intelligent systems to solve the problems of current systems. The new method is independent on the development environment, minimizes the load of client, and supports the communications based on texts.

1. 서 문

WWW은 1990년 첫발을 내디딘 후 기하학적 성장을 거듭하고 있다. 1996년 7월에 이미 인터넷에 연결된 호스트 수가 천 2백만 대를 넘어섰으며 사용 인구는 대략 1억 3천만 명 정도 된다[23]. 이를 통해 우리는 현재 인터넷의 확산 정도를 쉽게 짐작할 수 있다. 이뿐만 아니라 WWW은 VRML(Virtual Reality Modeling Language), JAVA[6], TCL(Tool Command Language)[8], 각종 플러그인 소프트웨어의 등장과 같은 소프트웨어 기술의 점진적 발전과 네트워크 저반시설의 급격한 성장으로 인해 과거에는 볼 수 없었던 새로운 서비스들이 속속 등장하고 있다.

지능시스템(intelligent system)은 전문가시스템의 보다 진전된 형태로 통합된 전문가시스템을 특정 하드웨어에 내포하고 이를 통해 보다 향상된 성능을 보유한 시스템이다[2]. 전문가시스템(expert system)은 "전문

※ 본 연구는 경상대 부설 정보통신 연구센터의 연구비 지원에 의한 것임.
†준 회 원 : 경상대학교 컴퓨터과학과 및 정보통신연구센터
††종신회원 : 경상대학교 컴퓨터과학과 및 정보통신연구센터
논문접수 : 1997년 8월 26일, 심사완료 : 1998년 3월 25일

가와 같은 지적 능력을 갖는 소프트웨어 체계"의 개발을 목표로 시작된 연구이다. 그리고 실제 성공적으로 개발된 전문가시스템은 일선에서 인간 전문가를 대체한다. 전문가시스템은 인간 전문가에 비해 많은 장점을 가지고 있다. 전문가시스템은 인간 전문가와는 달리 지식을 일관성 있게 적용할 수 있고, 사람과 달리 수명의 한계가 없으며, 지식을 쉽게 복사·이동시킬 수 있을 뿐만 아니라 전문가시스템 자체도 쉽게 복제할 수 있다[22]. 따라서, 전문가시스템으로부터 획득한 정보는 다른 서비스로부터의 정보와는 질적으로 다를 수밖에 없다.

전문가시스템은 어떤 특정 분야의 인간 전문가들의 전문 지식들을 수집 정리하여 주어진 특정 전문 영역에 관한 문제를 컴퓨터의 어떤 추론 능력을 이용하여 해결하고자 하는 대화형 컴퓨터 자문 시스템을 말한다. 전문가시스템은 질문 및 응답을 위주로 한 대화형 컴퓨터 시스템이며, 특정 문제의 해결을 위한 시스템이며, 컴퓨터에게 지능화된 다양한 지식을 이식시켜 컴퓨터가 인간처럼 지능적인 행동을 하도록 만든 인공지능시스템이다[21].

WWW을 사용자 인터페이스로 이용하는데 있어 데이타베이스시스템과 전문가시스템에는 큰 차이점이 있다. 데이타베이스시스템은 사용자로부터 질의어를 입력받아 그에 합당한 서비스를 해주고 이후에 요청되는 질의어는 이전의 질의어와 아무런 관계를 가지지 않는 비대화형 시스템이다. 이는 서비스가 요청되면 연결하고 해당 결과가 돌려지면 연결을 단절하는 WWW의 성격과도 잘 맞는다. 그러나 전문가시스템은 사용자와 지속적인 연결을 유지하며 자료를 교환해야하는 대화형 시스템이므로 비연결적인 특성을 지닌 WWW과는 잘 맞지 않는다. 따라서 데이타베이스시스템과는 달리 전문가시스템을 위한 WWW의 인터페이스 구현은 지속적 연결을 유지하기 위한 사용자 검정, 사용자와의 대화를 유지하기 위한 사용자별 상태 유지 등을 고려해야 한다.

본 연구는 JAVA나 TCL과 같은 도구를 사용하지 않음으로써 WWW의 하부구조 대한 의존성을 줄이며, 서버 측에서 대부분의 처리를 제공함으로써 클라이언트의 부담을 최소화하며, 텍스트 기반의 통신을 지원함으로써 웹브라우저에 독립되고 대부분의 웹 자원에 접근 가능한 전문가시스템의 WWW 인터페이스를 개발하는 것이다.

2. 연구 배경

전문가시스템의 사용자 인터페이스를 WWW을 이용하여 구현하고자 함은 다음과 같은 의미를 가진다. 첫째, 이미 대중화되어 있는 WWW을 통하여 보다 많은 사람들에게 쉽고 빠르게 전문가시스템을 사용하게 할 수 있다. 둘째, 다양한 매체를 훌륭히 처리할 수 있는 WWW을 인터페이스로 사용하면 보다 쉽고 간결하게 멀티미디어 전문가시스템을 구현할 수 있다.

전 세계적으로 볼 때 WWW을 전문가시스템의 사용자 인터페이스로 이용하기 위한 다양한 연구들이 있었다. 대표적으로 JAVA나 TCL과 같은 도구를 이용한 방법과 Fuzzy Show Control[4] 시스템과 같은 환경변수를 이용한 방법이 있다. 그러나 국내에서는 그에 관한 연구가 거의 없는 실정이다. 본 장에서는 기존의 연구들을 분석해 보고 서로의 장단점을 살펴본다.

2.1. A Land Rover Identifier Expert System with a World Wide Web Interface

'A Land Rover Identifier Expert System with a World Wide Web Interface'[7]는 TCL CGI(Common Gateway Interface)[16]을 이용하여 구현된 전문가시스템이다. TCL은 어플리케이션의 확장과 제어를 편리하고 간단하게 해주는 스크립트 언어이다. TCL은 변수 및 제어구조를 갖추고 있어 프로그래밍을 위한 도구가 되는 동시에 C Library Interface를 통하여 언어의 확장을 매우 편리하게 할 수 있다는 장점을 가지고 있다. TCL은 WWW 페이지 내에 자신만의 창을 열어 사용자와 지속적인 연결을 유지하면서 상호 대화적으로 정보를 처리할 수 있는 간단한 방법을 제공해 줄 뿐만 아니라 부프로그램을 실행하는 다양한 방법을 가지고 있다. 특히 TCL은 자식프로세스를 생성한 다음 프로그램의 표준입출력을 파이프로 방향전환(red- irection)하여 실행시키는 방법을 제공해 준다. 이 방법을 사용하면 CGI는 전문가시스템과 간단히 통신할 수 있다[8].

TCL CGI를 이용하면 이미 구축되어 있는 라이브러리를 사용하여 비교적 쉽게 WWW용 전문가시스템을 구현할 수 있다. 그러나, 이 방법에서 전문가시스템은 CGI의 자식프로세스로서 실행된다. 그리고 CGI는 웹서버 데몬(web server daemon)의 자식프로세스로 실행되므로 프로세스 관리를 전적으로 웹서버에게 의존하게 된다. 이로부터 여러 부작용이 있을 수 있다. 특히 이 방법은 사용자가 서비스를 완결할 때까지 CGI는

항상 대기해고 있어야 하므로 동시에 실행할 수 있는 CGI 수는 WWW 서버가 동시에 실행할 수 있는 CGI의 프로세스 수를 초과할 수 없다.

2.2. JESS

JESS(JAVA Expert System Shell)[5]는 전문가시스템 도구인 CLIPS(C Language Integrated Production System)[1]를 JAVA 언어로 다시 쓴 전문가시스템 쉘이다. CLIPS는 미항공우주국(NASA)에서 만든 전문가시스템 도구이다. 1980년대 중반까지 NASA에서는 전문가시스템 도구 작성을 위해 LISP을 사용해 왔다. 그러나 LISP으로 작성된 전문가시스템 도구는 상용 컴퓨터에서의 성능 저하, 고가의 하드웨어와 소프트웨어 그리고 다른 언어와의 낮은 통합성과 같은 문제점을 안고 있었다. 이러한 문제점 때문에 NASA는 C 언어로 전문가시스템 도구를 작성하기로 하였다. 1985년 처음으로 CLIPS의 프로토타입이 완성된 후 CLIPS는 성능 향상과 확장을 거듭하였다. 현재 CLIPS는 이식성, 확장성, 기능성에서 높은 성능을 보유하고 있으며 저가로 공급되므로 정부, 학계는 물론 산업계에서도 두루 쓰이고 있다[14].

JAVA 언어는 중간코드 단계에서 다양한 플랫폼에 이식될 수 있을 뿐만 아니라 WWW의 응용성을 높이는 데도 크게 기여하고 있다. JESS는 JAVA언어로 WWW에서 전문가시스템 서비스하고자하는 사용자에겐 매우 유용하다. JESS를 사용하면 사용자의 애플릿에 쉽게 전문가시스템 기능들을 포함시킬 수 있다. 그 뿐만 아니라 JESS는 CLIPS를 그대로 JAVA로 바꿨기 때문에 기존의 CLIPS로 만들어진 전문가시스템을 그대로 쓸 수 있다[6].

불행하게도 JESS로 구현한 전문가시스템은 시동시키는데 지나치게 많은 시간을 소모한다. 그 원인은 두 가지가 있다. 첫째는 JESS가 JAVA로 씌어졌다는데 있다. JAVA 소스를 컴파일하면 중간코드 형태를 지닌 바이트 코드가 만들어진다. 이를 웹브라우저에서 실행할 때는 바이트 코드를 해석하여 실행하는 과정을 거치게 된다[3]. 그러나 이 작업은 상당한 CPU 시간을 요구한다. 그리고 전문가시스템의 실행코드는 패턴 매칭, 추론엔진과 같은 CPU 시간을 많이 요구하는 요소들을 가지고 있다. 따라서 전문가시스템을 JAVA 언어로 쓰면 실행속도가 느려질 수밖에 없다. 둘째는 JESS는 CLIPS를 다시 썼다는데 있다. CLIPS는 방대한 기능을 가지고 있으므로 그 실행파일도 엄청난 크기를 가지고 있다. 물론 JESS는 CLIPS의 핵심적인 기능만을 구현했으나 네트워크 환경에서 JESS로 구현된 전문가시스템을 실행해 보려는 사용자는 많은 시간을 기다려야 한다.

JESS를 사용한 전문가시스템은 사용자 측에 대부분의 시스템 운영 부담을 준다. 이는 곧 전문가시스템의 원활한 운영을 위해 사용자 측에 보다 높은 시스템 사양을 요구한다. 그러나, 이는 서버 시스템의 고기능-저가화와 NC(Network Computer)의 등장과 같은 클라이언트 경량화 추세에 역행한다. 그리고 아직도 JAVA를 사용할 수 없는 환경에서 웹브라우징(web browsing)을 하는 사용자가 많다는 점도 매우 중요하다.

2.3. Fuzzy Shower Control System

Fuzzy Shower Control 시스템은 Fuzzy CLIPS를 이용하여 제작된 간단한 전문가시스템으로서 IIT-NRC에서 NASA의 CLIPS에 퍼지논리를 지원할 수 있도록 확장한 것이다[4].

Fuzzy Shower Control 시스템은 시스템을 단순하게 하고 사용자와 논리적 연결을 유지하기 위해 전문가시스템 실행 후 냉수의 유속, 온수의 유속과 같은 프로세스의 상태와 사용자 검증 자료를 다음 웹 페이지 내에 포함시킨다. 주어진 웹문서에 사용자가 적당한 반응을 보일 때 Fuzzy Shower Control 시스템 이와 같은 값들을 재입력하여 전문가시스템을 초기화하고 요청되는 사용자의 응답을 이용하여 추론한 결과와 시스템 현 상태를 새로운 웹 페이지에 포함시켜 돌려준다[10].

웹서버는 환경변수를 이용하여 자신의 자식프로세스로 실행되는 CGI 프로그램에 사용자로부터 입력되는 정보나 시스템 정보를 전달한다. 예를 들어 환경변수 QUERY_STRING은 주로 CGI에 HTML form문에 의한 정보를 전송할 때 사용되고, 환경변수 PATH_INFO는 주로 CGI 프로그램에 파일의 위치를 전달하는데 사용된다. Fuzzy Shower Control 시스템은 이러한 환경변수를 다르게 이용한다. 환경변수 PATH_INFO는 사용자와 논리적 연결을 유지하기 위하여 사용자 동일성(identity) 검증에 이용되고, 환경변수 QUERY_STRING은 사용자의 메시지 전달을 위해 이용된다[10].

이 시스템은 클라이언트 측에 부담을 적게 주며 자

원의 낭비를 줄이면서 비교적 간단하게 WWW 전문가 시스템 구현한다. 그러나 일단 전문가시스템의 상태를 환경변수로 재저장하는 방식이므로, 전문가시스템이 커질 경우 상태변수가 많아져 시스템이 복잡해짐은 물론, 전문가시스템을 구축하기조차 힘들어질 것이다. 그리고 상태를 저장해야 하므로 기존의 전문가시스템을 재 이용하고자 할 때 시스템 구조를 크게 변경해야 하는 어려움이 뒤따른다. 그리고 이 방법은 사용자 요구가 한 번 있을 때마다 전문가시스템을 한 번씩 구동하는 구조로 되어 있다. 그러므로 이 시스템은 추론 도중 시스템이 사용자에게 질의를 요구하는 전문가시스템은 가동할 수 없다.

3. 시스템 제안 및 설계

앞장에서 WWW을 전문가시스템의 사용자 인터페이스로 사용한 기존한 방법들을 살펴보았다. 그러나 TCL을 이용한 방법과 JAVA를 이용한 방법은 WWW의 개발도구에 지나치게 의존하게 됨으로써 개발도구가 가지고 있던 근본적인 문제점을 그대로 내포하고 있으며, Fuzzy Shower Control은 문제를 지나치게 단순화하여 시스템의 응용성을 제약하는 결과를 낳았다. 본 연구는 이러한 문제점들을 제거하고 보다 현실성 있는 시스템을 구현하기 위해 다음과 같은 세 가지 항목에 초점을 맞추어 설계하였다.

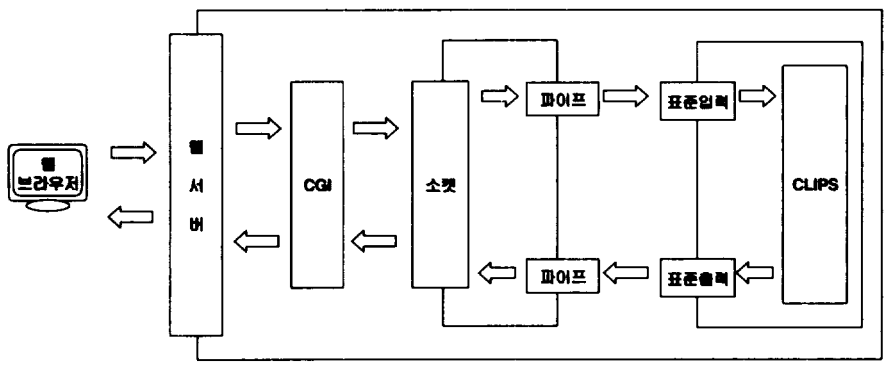
첫째, 본 시스템은 WWW의 개발환경에 종속되지 않는다. TCL이나 JAVA와 같은 도구에 의존하지 않고, CGI에 대한 전문가시스템의 의존성도 제거한다.

둘째, 본 시스템을 운영하는 데 있어 소요되는 비용은 서버 측에서 전격 부담한다. 대부분의 처리는 서버 측에서 해결해 주고 클라이언트나 네트워크는 간단한 질의와 응답이 오가는 정도의 기본적인 부담만 가지게 한다.

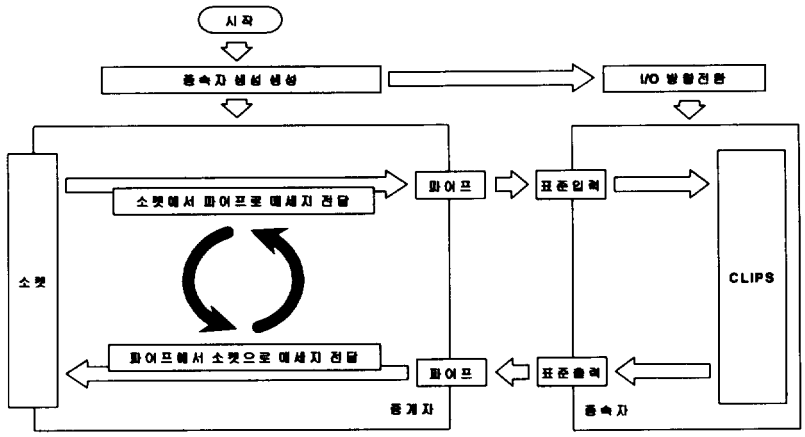
셋째, 본 시스템은 사용자와의 대화를 전적으로 텍스트에 기초한다. 따라서 최근의 웹브라우저는 물론 초기의 웹브라우저에서도 접근할 수 있을 뿐만 아니라, 모든 HTML[11] 문법은 물론 특정 웹브라우저가 지원하는 다양한 기능도 이용할 수 있다.

3.1 시스템의 구조

(그림1)은 본 논문에서 제안하는 시스템의 흐름 및 구성도이다. 본 시스템은 다음과 같은 시나리오를 따른다. 먼저 서버시스템이 시동되면 먼저 종속자와 중개자가 생성되고 상호 통신을 위하여 파이프를 설치한다. 중개자는 CGI의 접속에 대비하여 소켓을 설치하고 메시지 전달에 대비한다. 종속자는 사용자의 응답을 요구하는 질문을 중개자로 전달하고 중개자는 다시 CGI로 질문을 전달한다. CGI는 질문을 전달받아 웹문서를 구성한 후 웹서버를 통하여 사용자에게 전달하고 종료한다. 이에 사용자는 적절한 응답을 하게되면 응답은 다시 CGI를 통하여 중개자로 전달되고 이는 다시 종속자로 전달되어 실제 전문가 시스템에 전달되게 된다. 이에 전문가시스템은 적절한 처리를 하게 되고 최종결과나 적절한 질문을 중개자와 CGI를 거쳐서 사용자에게 전달하게 된다.



(그림 1) Server-Client 구성도
(Fig. 1) the Server-Client mechanism



(그림 2) 중개자와 종속자의 구조도
(Fig. 2) the structure diagram of Intermediator and slave

3.2 중개자(intermediator)와 종속자(slave)

중개자는 CGI와 종속자의 중간에서 메시지를 안전하게 중개하는 역할을 한다. (그림2)는 중개자와 종속자의 절차를 도식화하여 보여준다. 먼저 중개자는 사용자의 요구가 있기 전에 파이프를 두 개 설치하고 종속자 프로세스를 자식프로세스로 생성하는 초기화 작업을 시작한다. 이때 종속자는 부모프로세스인 중개자와 파이프를 공유하게 된다. 두 개의 파이프 중 하나는 중개자에서 종속자로 다른 하나는 종속자에서 중개자로의 메시지 전달에 쓰이게 된다. 이후 중개자는 CGI의 접속에 대비한다. 중개자는 CGI로부터 메시지가 전달되면 파이프를 이용하여 종속자에게 전달하고 종속자로부터 돌아오는 메시지는 CGI에게 전달한다[20].

중개자의 자식프로세스로 생성된 종속자는 무엇보다 먼저 자신의 입출력을 파이프로 방향전환하고 중개자와의 통신에 대비한다. 이후 중개자로부터 메시지 전달이 있으면 이를 읽어서 처리한 후 해당 출력을 중개자에게 전달한다.

3.3 CGI

본 시스템에서 사용된 CGI는 우선 웹문서의 선언부를 출력하고 소켓을 통하여 중개자에 연결된다. 사용자의 메시지를 적절한 형태로 변환하고 이를 소켓을 통하여 중개자로 전송하고 결과가 되돌아올 때까지 대기한다. 결과가 되돌아오면 이를 변환하여 웹문서를 적절히 구성하고 사용자로부터 입력을 받기 위해 FORM 태그를 구성한 후 문서의 종결부를 출력하고 종료한다[20].

4. 시스템 구현

본 시스템 유닉스 기반, 서버/클라이언트 시스템의 형태로 구현하였다. 이를 위해 TCP/IP 소켓을 이용하며 CGI는 이를 통하여 서버와 통신한다. CGI는 사용자와 본 시스템을 연결해 주는 부분으로서 웹서버의 자식프로세스로 실행된다. 마지막으로 본 시스템에 실제 전문가시스템을 장착하여 실험한다. 본 시스템의 서버와 CGI는 SUN Solaris 2.4에서 C 언어를 사용하여 구현하였고 클라이언트로는 Netscape 3.04[12]를 이용하였다.

4.1 서버 구현

종속자와 중개자 구현에 있어 프로세스간 통신과 프로세스 제어가 핵심이 된다. (그림 3)은 (그림 2)에서 보여주는 종속자와 중개자의 원시코드이다.

중개자는 먼저 소켓을 설치하는데, 소켓은 BSD 4.1.c에서 소개된 프로세스간 통신(IPC, Interprocess Communication)을 위한 한 방법으로서 TCP/IP 환경에서 서버/클라이언트 환경을 이루는 데 있어 유용히 이용된다. 소켓은 다중 프로세스 접근에 대한 직렬화(serialization) 기능과 선택되지 못한 프로세스를 위한 대기행렬(queue) 기능을 가지고 있다[15]. 중개자는 파이프를 설치하고 종속자를 자식프로세스로 실행하는데 이는 간단히 파이프를 이용하여 프로세스간 통신을 할 수 있는 이점을 가진다. 이후 중개자는 메시지 전달을 위한 무한루프에 들어간다.

```
1:  ID="test.socket";                                   /* CGI와 접속할 소켓의 이름 */
2:  sock=socket(AF_UNIX,SOCK_STREAM,0);                 /* 소켓 생성 */
3:  if (sock<0)                                         /* 소켓 생성의 실패여부 조사 */
4:      exit(0);                                        /* 시스템 종료 */

5:  cn.sa_family=AF_UNIX;                               /* 구조체에 소켓의 프로토콜 지정 */
6:  strcpy(cn.sa_data,ID);                              /* 구조체에 소켓 이름 지정 */
7:  addrlen=strlen(cn.sa_data)+sizeof(cn.sa_family);    /* 구조체의 크기를 구함 */

8:  unlink(ID);                                         /* 이전의 소켓이 남아 있다면 제거 */
9:  if (bind(sock,&cn,addrlen)<0)                       /* 소켓에 이름을 붙이고 공개 */
10:     exit(0);                                        /* 시스템 종료 */

11: if (listen(sock,1)<0)                               /* 동시접근을 위해 소켓에 큐를 설치 */
12:     exit(0);                                        /* 시스템 종료 */

13: if (pipe(Ps2p)<0 || pipe(Pp2s)<0)                   /* 종속자와 통신을 위해 파이프를 생성 */
14:     exit(0);                                        /* 시스템 종료 */

15: signal(SIGCHLD,SIG_IGN);                            /* 종속자로 부터 오는 시그널은 무시함 */

16: switch ((pid = fork())){                            /* 종속자를 자식프로세스로 생성 */
17:     case -1:                                        /* 자식프로세스 생성에 실패하였음 */
18:         exit(0);                                    /* 시스템 종료 */
19:     case 0:                                         /* 종속자의 코드 */
20:         close(Ps2p[1]);                             /* 쓰지 않는 파이프는 폐쇄 */
21:         close(Pp2s[0]);                             /* 쓰지 않는 파이프는 폐쇄 */
22:         clips(Ps2p[0],Pp2s[1]);                     /* 전문가시스템의 실행 */
23:         exit(0);                                    /* 종속자의 종료 */
24: }
    /* 중개자의 코드                                    */
25: close(Ps2p[0]);                                     /* 쓰지 않는 파이프를 폐쇄 */
26: close(Pp2s[1]);                                     /* 쓰지 않는 파이프를 폐쇄 */

27: while(1){                                           /* 중개자의 메시지 처리 루프 */
28:     if ((nsock=accept(sock,&cn,&addrlen))<0)        /* CGI의 소켓 접근 감시, 접속 */
29:         exit(0);                                    /* 시스템 종료 */

30:     while (fgetstr(nsock,buf)){                     /* CGI로부터 메시지를 읽어 들임 */
31:         if (ismsg(mOver,buf))                       /* 메시지의 끝인지 검사 */
32:             break;
33:         fputstr(Ps2p[1],buf);                       /* 메시지를 종속자로 전달 */
34:     }
35:     while (fgetstr(Pp2s[0],buf)){                   /* 종속자로부터 메시지를 읽어 들임 */
36:         fputstr(nsock,buf);                         /* 메시지를 중개자로 전달 */
37:         if (ismsg(mQuit,buf)){                      /* 서비스 종료 메시지인지 검사 */
38:             fputstr(nsock,mQuit);                   /* 종료 메시지를 CGI로 전달 */
39:             QuitClipsServer();                      /* 종속자 프로세스의 종료 */
40:             exit(0);                                /* 시스템 종료 */
41:         }
42:         if (ismsg(mOver,buf))                       /* 메시지 끝인지 종료 */
43:             break;                                  /* while 루프 탈출 */
44:     }
45: }

46: void ClipsCore(int in,int out,char *GName)          /* 전문가시스템 */
47: {
48:     if (in != STDIN_FILENO){                        /* I/O 방향전환 */
49:         dup2(in, STDIN_FILENO);
50          close(in);
51:     }
52:     if (out != STDOUT_FILENO){                      /* I/O 방향전환 */
52:         dup2(out, STDOUT_FILENO);
53:         close(out);
54:     }
55:     setvbuf(stdin,NULL,_IOLBF,0);                   /* 줄 단위 입력을 위한 라인규약 설정 */
56:     setvbuf(stdout,NULL,_IOLBF,0);                  /* 줄 단위 출력을 위한 라인규약 설정 */

57:     InitializeCLIPS();                              /*                                   */
58:     RerouteStdin(1,"clips");                        /* CLIPS 호출                        */
59:     CommandLoop();                                  /*                                   */
60:     exit(0);
61: }
```

(그림 3) 중개자의 원시코드

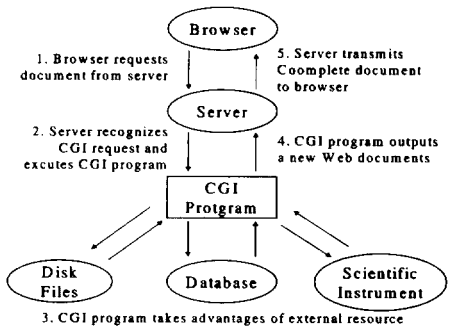
(Fig. 3) the source codes of Intermediator

종속자는 표준입출력을 파이프로 방향전환하고 입출력 장치의 버퍼링 방식을 블럭 단위 처리 방식에서 라인 단위 처리 방식으로 바꾼 후, CLIPS를 실행한다[20].

4.2 CGI 구현

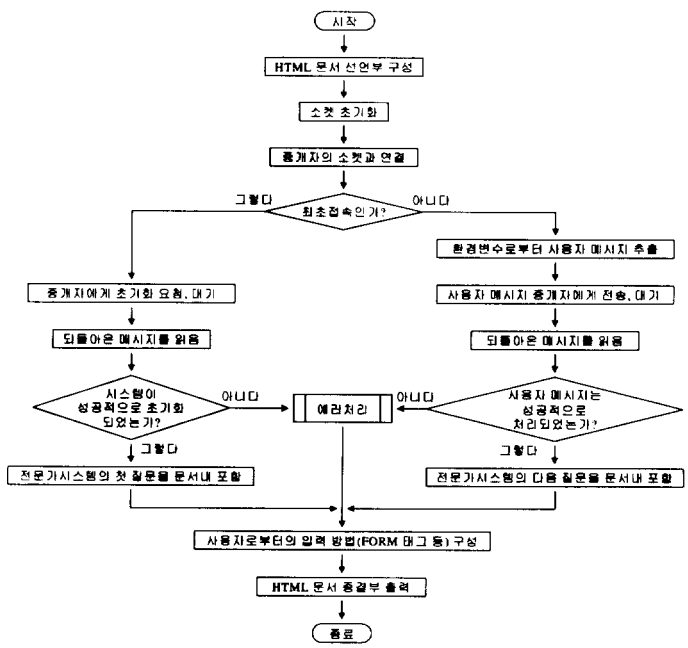
CGI는 동적 웹문서를 제공하기 위해 서버 시스템에 설치되어 있는 외부 프로그램을 이용하여 해결하기 위한 표준화된 방법이다[19]. CGI를 사용하지 않는 기존의 HTML문서는 문서 공급자의 정적인 의도만을 표현할 수 있었으나 CGI를 사용한 문서는 사용자의 의도를 동적으로 처리할 수 있으며 서버에 정보를 저장하는 것도 가능하다. 뿐만 아니라 CGI를 이용하면 데이터베이스뿐만 아니라 시스템 내의 다른 서버의 프로그램들과도 통신이 가능하다.

(그림4)는 CGI의 수행절차를 보여준다. 웹브라우저가 웹문서를 요구하면 서버는 CGI 자식프로세스로 실행한다. CGI는 일반 파일에 접근한다든지 데이타베이스에 접근한다든지 혹은 다른 기구에 접근하여 적당한 처리를 한 후 결과를 출력하고 웹서버는 CGI로부터 출력된 결과를 웹브라우저에게 전달한다. CGI는 환경변수나 표준입력을 이용하여 자료를 전달받고 표준출력을 이용하여 자료를 전달한다. CGI는 C, Perl[17], TCL 등 환경변수에 접근 가능하고 표준입력과 표준출력이 지원되는 대부분의 언어를 이용하여 작성할 수 있다[16].



(그림 4) CGI 구성도
(Fig. 4) the CGI mechanism

본 시스템에서 작성된 CGI는 (그림5)와 같이 수행된다. 먼저 웹문서의 선언부를 출력하고 소켓을 중개자의 소켓과 연결한다. 해당 사용자가 처음으로 시스템에 접속했는지 파악한 후 만약 첫 접속이면 중개자에게 초



(그림 5) CGI 프로그램의 흐름도
(Fig. 5) the flowchart of the CGI program

기화를 요구한다. 시스템이 적절히 초기화되면 전문가 시스템으로부터 첫 질문이 전달되고 CGI는 이를 웹문서에 포함한다. 만약 해당 사용자가 접속한 적이 있으면 CGI는 환경변수 QUERY_STRING으로부터 사용자의 메시지를 추출하고 이를 중개자에 전달한다. 성공적으로 수행되면 전문가시스템은 적절한 질문을 전달하고 CGI는 이를 웹문서에 포함한다. 이후 질문에 대한 사용자의 응답을 얻기 위해 FORM 태그나 아이콘을 이용한 Anchor 태그를 구성하고 웹문서를 종결한다.

4.3 I/O 방향전환

본 시스템은 전문가시스템의 사용자 인터페이스와 연결을 위해 I/O 방향전환과 입출력 동기화 기법을 이용한다. 이는 기존의 텍스트 기반 전문가시스템을 본 시스템에 연결함을 의미하는 것으로써 이미 개발되어 있는 전문가시스템을 최소의 수정을 통하여 재활용할 수 있게 한다. 본래 CLIPS는 표준 입출력을 사용하여 사용자와 통신을 하도록 되어있으나 본 시스템에서는 종속자의 표준 입출력을 파이프로 방향전환하게 하여 상호 정보 교환을 가능하게 한다.

종속자의 표준 입출력을 파이프로 치환하기 위해서는 I/O 방향전환이 있어야 한다. UNIX C 프로그래밍 에선 dup2(int filedes, int filedes2) 함수를 사용해서 특정 파일 디스크립터를 다른 파일 디스크립터로 치환할 수 있는 방법을 제공해 준다[13]. (그림6)은 dup2함수의 사용 예를 보여준다.

```
if (in != STDIN_FILENO) {
    if (dup2(in, STDIN_FILENO) != STDIN_FILENO)
        exit(0);
    close(in);
}
if (out != STDOUT_FILENO) {
    if (dup2(out, STDOUT_FILENO) != STDOUT_FILENO)
        exit(0);
    close(out);
}
```

(그림 6) 파일 디스크립터 치환
(Fig. 6) replacing file descripters

그러나 표준 입출력을 단순히 파이프로 치환한다고 하여 모든 문제가 해결되는 것은 아니다. 대부분의 유닉스 입출력은 파일로 추상화되었다. 그러나 파일의 실제 입출력 장치가 달라짐에 따라 처리방식이 달라지고 그로 인해 전혀 예상치 못한 경우가 발생한다. 따라서 특별한 조치 없이 파일 디스크립터를 치환하는 방법은 신뢰할 수 없다[13]. 그러므로 보다 안정성 있는 프로그램을 위해서는 다른 추가적인 기술이 필요하다.

UNIX 프로세스는 터미널 장치를 표준입출력 파일로 사용한다. 모든 터미널 장치에는 라인규약이라는 것이 있어서 라인단위로 입출력하는 것이 초기상태로 설정되어 있다. 그러나 파이프와 기타 다른 파일은 블럭단위로 입출력하므로 터미널을 사용하는 입출력을 파이프나 다른 파일로 치환하면 블럭단위의 입출력으로 변경되어 라인단위의 입출력함수를 사용한 프로그램에서는 예상치 못한 결과가 발생할 수 있다[13]. CLIPS에서는 printf()와 같은 버퍼링을 하는 표준입출력 함수를 사용하고 있다. 이는 프로세스가 출력하는 시간과 실제 매체에 저장되는 시간이 일치하지 않아 신뢰할 수 없다. 이를 해결하는 방법으로 약속된 특정 규정을 약속하여 실제 매체에 기록하는 시기를 정하는 방법이 있다. 예를 들어 "new line 문자가 입력될 때 버퍼를 비운다"는 방식의 규칙을 정하는 것이다. 본 시스템에서 필요하는 방법도 이와 같이 라인단위로 입출력을 행하는 것이다. (그림7)은 버퍼 비우는 시기를 변화시키는 예를 보여준다.

```
if (setvbuf(stdin,NULL,_IOLBF,0)!=0)
    error("Clips> Setvbuf Stdin Error");

if (setvbuf(stdout,NULL,_IOLBF,0)!=0)
    error("Clips> Setvbuf Stdout Error");
```

(그림 7) 버퍼 비우는 시기 변경
(Fig. 7) decision of the time to flush buffer

4.6 입출력 동기화(synchronization)

본 시스템은 입력을 받고 처리한 후 처리결과를 돌려주는 방식으로 입력과 출력이 순서를 바꿔가며 진행하는 동기화된 방식을 이용한다. 이는 곧 입력과 출력은 동시에 일어날 수 없으며 입력이 끝나야 곧이어 출력이 가능하다는 것을 의미한다. 소켓이나 파이프는 입력과 출력을 비동기적으로 행한다. 본 시스템은 비동기적인 소켓과 파이프를 사용하여 동기적인 프로세스간 통신을 구현해야하기 위해 특별한 방법을 필요로 한다.

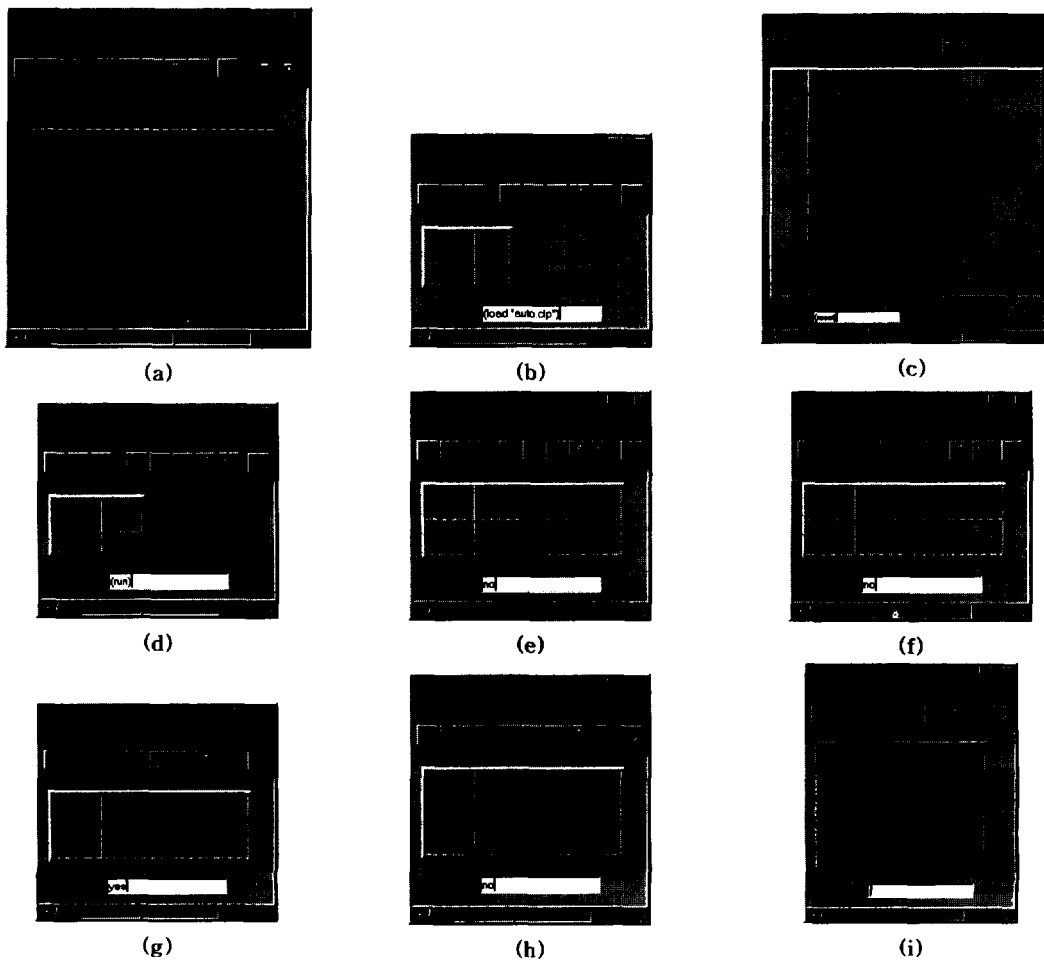
입출력을 동기화하려면 자료를 읽고 쓰는 처리과정에 있어 그 종료시점을 알아야 한다. 여러 방법이 있을

수 있으나 본 시스템에서는 프로세스가 출력할 때 출력의 끝을 알리는 특정 메시지를 보내는 방법을 이용하여 구현한다.

먼저. 본 시스템에서 CLIPS의 명령어 최상위 단계(top level)에서 쉘 명령어 동기화를 이루기 위해. (그림 8)과 같이 CLIPS 6.0의 'commline.c' 중 prompt를 출력하는 코드 다음에 특정 메시지를 출력하는 코드를 삽입하였다. 여기서 mOVER는 입출력의 끝을 알리는 메시지이다.

```
if ((CompleteCommand(CommandString) != 0) &&
(CLIPSInputCount > 0)){
    FlushPPBuffer();
    SetPPBufferStatus(OFF);
    CLIPSInputCount = -1;
    RouteCommand(CommandString);
    FlushPPBuffer();
    SetHaltExecution(CLIPS_FALSE);
    SetEvaluationError(CLIPS_FALSE);
    FlushCommandString();
    FlushBindList();
    PeriodicCleanup(CLIPS_TRUE,CLIPS_FALSE);
    PrintPrompt();
    printf("%s",mOver);      /* 동기화를 위한 메시지 출력 */
}
```

(그림 8) 동기화를 위해 CLIPS에 코드 추가
(Fig. 8) inserting a code into the source code of CLIPS for synchronization



(그림10) 실행과정
(Fig. 10) a sequence of the execution

CLIPS에 장착될 전문가시스템에도 입출력의 동기화가 이루어져야 한다. 전문가시스템은 사용자에게 질문을 보낸 후 즉각 동기화를 의미하는 특정 메시지를 전달하여야 한다. (그림9)는 자동차 고장진단 전문가시스템[9]의 원시코드의 일부이다.

```
(deffunction ask-question (?question $?allowed-values)
    (printout t ?question crlf)
    (printout t <OVER> crlf)     /* 동기화를 위한 메시지 출력 */
    (bind ?answer (read))
    (if (lexemep ?answer)
        (bind ?answer (lowcase ?answer)))
    (while (not (member ?answer ?allowed-values)) do
        ntout t ?question)
        (printout t <OVER> crlf)/* 동기화를 위한 메시지 출력 */
        (bind ?answer (read))
        (if (lexemep ?answer)
            then (bind ?answer (lowcase ?answer))))
    ?answer)
```

(그림 9) 동기화를 위해 전문가시스템에 코드 추가
(Fig. 9) inserting a code into the source code of expert system for synchronization

4.5 자동차 고장 전문가시스템

본 시스템에 실제 전문가시스템을 시험운영하기 위해 이미 개발되어 있는 자동차 고장진단 전문가시스템[9]을 수정하여 적용하였다. 자동차 고장진단 전문가시스템은 CLIPS를 도구로 이용하여 개발된 간단한 시범용 전문가시스템이다. (그림10)은 실제로 본 시스템에 자동차 고장진단 전문가시스템을 장착한 후 Netscape 3.04를 사용하여 운영한 예이다. 본 시스템은 서버에 실제 실행되고 있는 CLIPS와 직접 통신하기 때문에 CLIPS의 모든 명령을 웹브라우저에서 직접 사용할 수 있다. 그리고 CLIPS를 도구로 쓰는 전문가시스템은 자동차 고장진단 전문가시스템 뿐만 아니라 다른 전문가시스템도 입출력을 동기화 시켜주는 최소의 수정으로 본 시스템에 장착되어질 수 있다. (그림 10-a)는 본 시스템의 시작 WWW 페이지이다. (그림 10-b)에서는 본 시스템에 자동차 고장진단 전문가시스템을 장착할 것을 요청하고 (그림 10-c)에서는 전문가시스템을 초기화하며 (그림 10-d)에서 전문가시스템을 구동시킨다. (그림 10-e)에서 (그림 10-h)까지는 결과를 도출하기 위한 전문가시스템과 사용자와 대화하는 과정이다. 마지막으로 (그림 10-i)는 주어진 질문들에 대한 사용자의 응답을 토대로 도출된 결과를 보여준다.

5. 평 가

본 시스템과 소개되었던 세 가지 방법들은 서로 다른 방법으로 전문가시스템 WWW 인터페이스를 구현하였고 제각기 장단점을 가지고 있다. 〈표 1〉은 이들 네 가지 방법의 장단점을 여러 가지 측면에서 서로 비교하여 보여준다. 본 시스템은 클라이언트에 대한 부담은 최소화하고 서버에서 대부분은 운영 부담을 가지는 중앙 집중적 시스템이다. JAVA를 이용한 방법은 실행코드의 전송도 포함해야하므로 네트워크에 대한 부담이 크다. 환경변수를 이용한 방법은 시스템의 상태를 WWW 페이지 포함시키고 환경변수를 이용하여 시스템에 재저장해야하므로 전문가시스템이 커지면 복잡해져

〈표 1〉 전문가시스템 WWW 인터페이스 간 비교
〈Table 1〉 the comparison of WWW interfaces for expert system

분류 / 항목	TCL을 이용한 방법	JAVA를 이용한 방법	환경변수를 이용한 방법	본 시스템
클라이언트에 대한 부담	중	대	소	소
서버에 대한 부담	중	소	소	대
네트워크에 대한 부담	소	대	소	소
멀티미디어 처리 능력	중	중	대	대
개발자에 대한 부담	중	중	대	소
HTML 지원 능력	소	소	대	대
다중사용자 접근	가능	가능	가능	미결

개발자에게 큰 부담을 주게 된다. 본 시스템은 다중 사용자 접근에 해결책을 가지지 제시하지 못한다. 이에 관한 차후 연구가 요구된다.

6. 결론 및 향후과제

제안하는 방법은 기존의 방법과 비교해 볼 때 다음과 같은 점에서 개선되었다. 첫째, JAVA나 TCL을 이용하여 많은 문제점을 가지고 있던 개발환경에 대한 의존성을 그와 같은 도구를 사용하지 않음으로써 근원적으로 제거했다. 둘째, 서버 측에서 대부분의 부담을 지게 함으로써 실행프로그램 전송과 같은 클라이언트와 네크워크에 부담을 주지 않게 되었다. 셋째, HTML과 부합하는 텍스트에 기초한 통신을 함으로 현재는 물론 미래의 여러 인터넷 자원의 접근을 가능하게 한다.

본 연구에 이어서 이루어져야 할 연구로는 WWW 환경에 적합한 전문가시스템의 개발이 이루어져야하고 전문가시스템의 제작에 WWW의 멀티미디어 자원을 효율적으로 사용하기 위한 통일된 규약의 설정이 연구되어야 한다. 그리고 다수 사용자 동시 접근을 위해서는 본 시스템의 구성요소 중 중개자와 종속자의 재 설계가 요구되고 사용자들의 접근 상태를 관리하기 위한 관리기구가 필요하므로 이에 관한 차후 연구가 요구된다.

인터넷에서의 전문가시스템의 활용연구는 지식공유(knowledge sharing) 연구의 일환으로 전문가시스템의 새 장을 열게 될 것으로 기대된다. 대행자로 발전된 전문가시스템은 지식관리 기능뿐만 아니라, 문제해결 방법 선택기능과 대화기능를 갖추고 보다 양질의 정보제공에 기여하게 될 것이다[18].

참 고 문 헌

[1] CLIPS Home Page, [URL:http://www.jsc.nasa.gov/~clips/CLIPS.html]

[2] Donald A. Waterman, A Guide to Expert Systems, Addison-Wesley Publishing Company, 1986.

[3] Edith Au, Dave Makower, JAVA Programming Basics, MIS Press, 1996.

[4] Fuzzy CLIPS Home Page, [URL:http://ai.iit.nrc.ca/fuzzy/fuzzy.html]

[5] JESS Home Page, [URL:http://herzberg.ca.sandia.gov/jess]

[6] JESS Manual, [URL:http://herzberg.ca.sandia.gov/jess/README]

[7] Jan A. Barglowski, "Identa-Rover : A Land Rover Identifier Expert System with A World Wide Web Interface", [URL:http://www.offroad.com/RoverWeb/IDAR/project.html]

[8] John K. Ousterhout, Tcl and the Tk Toolkit, Addison-Wesley, 1994.

[9] Joseph Giarratano, Gary Riley, Expert Systems Principles and Programming Second Edition, PWS Publishing Company, 1994.

[10] Marvin Zaluski, "Implementing Fuzzy Shower Control on the WWW", e-mailed, 1996.

[11] NCSA, "The NCSA Beginner's Guide to HTML", [URL:http://www.ncsa.uiuc.edu/General/Internet/WWW/HTMLprimer.html]

[12] Netscape Communications Corporation, "Netscape Home Page", [URL:http://home.netscape.com/]

[13] Richard Stevens, Advanced Programming in the UNIX Environment, Addison-Wesley, 1992.

[14] Software Technology Branch Lyndon B.Johnson Space Center,CLIPS Reference Manual Volume I Basic Programing Guide CLIPS Version 6.04, 1997.

[15] Stevens, W, R, UNIX Network Programming, Prentice-Hall, 1990.

[16] Thomas Boutell, CGI Programming in C & Perl, Addison-Wesley, 1996.

[17] Tom Christiansen, "Perl 5 References",[URL:http://www.perl.com/]

[18] Utilities, "test SA *VANT for turbine troubleshooting", EPRI Journal, 17(3), 1992.

[19] W3 Consortium, "Overview of CGI", [URL:http://www.w3.org/hypertext/WWW/CGI/Overview.html]

[20] 김창민, 김용기, "전문가시스템 쉘을 위한 WWW 인터페이스의 구현", 정보과학회 '97 봄 학술발표논문집(B), 1997.

[21] 김화수, 고순주, 인공지능의 이론과 실제, 집문당,

1993.
[22] 이재규, 최형림, 김현수, 서민수, 주석진, 주원철, 전문가시스템 원리와 개발, 법영사, 1996.
[23] 한국전산원/Krnic, "인터넷 통계", [URL:http://www.nic.or.kr/net/net_m6.html]

김 용 기

1978년 서울대학교 공과대학 졸업(공학사)
1985년 University of Montana (전산학석사)
1992년 Florida State University (전산학박사)
1982년~1984년 KIST시스템공학연구소 연구원
1992년~현재 경상대학교 컴퓨터과학과 부교수
관심분야 : 인공지능, 전문가시스템, 퍼지시스템, 자동화 추론

김 창 민

1997년 경상대학교 컴퓨터과학과 졸업(이학사)
1998년~현재 경상대학교 컴퓨터과학과 석사과정
관심분야 : 인공지능, 퍼지시스템, 전문가시스템, 자동화 추론