

적응형 거북 세계의 설계

김성백[†] · 김철민[†] · 이정훈^{††}

요 약

교육용 프로그래밍 언어의 교육적 활용은 컴퓨터 자체를 쉽게 이해하고 활용하는 데 도움이 될 뿐만 아니라 학습자의 논리력과 사고력을 신장시켜 준다. 교육용 프로그래밍 언어의 이러한 역할에도 불구하고 그에 대한 연구나 개발은 범용의 프로그래밍 언어에 비해 매우 미비하다. 본 논문에서는 컴퓨터와 정보 기술의 발달을 반영하여 설계한 새로운 교육용 프로그래밍 환경 TWo (Turtle World)를 제안하고 고찰한다. 멀티미디어 프로그래밍 학습 도구이자 멀티미디어 작업 도구로서의 실용적 활용을 목적으로 개발 중인 TWo는 반영(reflection) 개념을 도입하여 수준별 프로그래밍 학습 환경을 효과적으로 구성할 수 있게 지원하는 적응성 높은 웹 기반 객체 지향 한글 프로그래밍 환경이다.

Design of the Adaptable Turtle World

Seong Baeg Kim[†] · Cheolmin Kim[†] · Junghoon Lee^{††}

ABSTRACT

Educational programming languages help learners to easily understand and use computer system itself. They also enhance the reasoning power and the contemplative faculty of learners. However, in spite of the importance of educational programming languages in education, there has not been much research and development on them compared with general purpose programming languages. In this paper, we propose and examine a new educational programming environment *TWo* (*Turtle World*) which is designed for learners to utilize the recent information technologies. Aimed at being used as a learning and working tool for multimedia programming and processing, *TWo* is a web-based reflective object-oriented Hangeul programming environment which provides each learner with the learning environment according to his intellectual ability to learn.

1. 서 론

컴퓨터 기술의 발달과 컴퓨터 보급의 대중화로 인해 모든 분야에서 컴퓨터 사용이 가능해짐에 따라 컴퓨터의 중요성은 매우 커지고 있다. 이처럼 컴퓨터가 필수 불가결한 요소로 인식되면서 사회 각 구성원들에게 컴퓨터에 대한 올바른 이해와 기

본 활용 능력이 요구되고 있으며, 이를 위해서는 체계적인 컴퓨터 교육이 필요하다. 컴퓨터 교육은 크게 컴퓨터 자체에 대한 교육과 컴퓨터 활용 교육으로 구분할 수 있는데, 컴퓨터 프로그래밍 교육은 컴퓨터 자체를 이해하는 데에 도움이 될 뿐만 아니라 자연스럽게 컴퓨터 활용의 체계성과 효율성을 제고시켜 준다. 이에 더하여 컴퓨터 프로그래밍이 인간의 논리력과 사고력을 증진시켜 준다는 점은 컴퓨터 교육에 있어 프로그래밍 교육의 효율

[†] 정회원: 제주대학교 컴퓨터교육과 조교수
^{††} 정회원: 제주대학교 전산통계학과 조교수
 논문접수: 2000년 8월 10일, 심사완료: 2000년 9월 22일

성과 중요성을 대변한다.

컴퓨터가 출현한 이후부터 수많은 컴퓨터 프로그래밍 언어가 개발되어 왔지만, 이들 대부분은 일반적 문제 해결을 지원하는 범용의 프로그래밍 언어들이다. 범용 프로그래밍 언어는 효과적 문제 해결에 초점을 맞추고 있기 때문에 컴퓨터에 대한 기본 지식이 제대로 갖추어져 있지 않은 초보자의 경우 능숙한 프로그래밍을 위해서는 상당한 교육 과정이 필요하다. 특히, 초보자로서 초등학생이나 중등학생이 교육 대상일 경우 일반적인 문제 해결 보다는 컴퓨터에 대한 이해 및 활용, 논리적인 사고력 등 교육적 측면이 더 중요한데, 이러한 교육적 요구를 충족시키기 위해 개발된 프로그래밍 언어가 교육용 프로그래밍 언어이다.

교육용 프로그래밍 언어로서 널리 사용되고 있는 대표적인 언어가 로고(Logo)[7]이다. 1960년대 미국 MIT에서 Seymour Papert에 의해 개발된 로고[14]는 발견 학습을 통해 학생들의 통찰력과 사고력을 증진시킨다는 특징이 있다[1,4]. 로고의 영어 명령에 부담을 갖는 학습자들을 고려해 국내에서 개발된 언어 피하미[2]는 로고 개념의 교육적 효용성을 살리면서 한글 문화권 학습자들도 쉽게 학습할 수 있도록 개발되었다. 피하미는 프로그래밍 환경을 한글화함으로써 프로그래밍에 사용되는 명령어뿐만 아니라 학습자를 위해 보다 편리한 상호작용 인터페이스를 제공한다. 그러나 피하미는 윈도우나 인터넷 환경을 지원하지 않고 단순 그림 그리기 기능 제공에 그치고 있어, 윈도우와 웹(Web), 멀티미디어 환경에 친숙해진 학습자들의 관심과 흥미를 불러일으키기에 부족한 면이 많다. 최근 개발된 EHLL[3]은 유지보수성과 확장성을 높이고 마우스만으로 프로그래밍 가능하도록 개발/유지보수 과정을 체계화하고 사용자 인터페이스를 개선한 한글 로고 언어이지만, 보다 차원 높은 프로그래밍 교육을 위해 로고 언어 자체의 개념적/기능적 확장을 추구하지는 않고 있다.

최근 학습 효과를 높이기 위해 널리 활용되고 있는 멀티미디어 기반 학습 환경은 텍스트, 그래픽, 사운드, 애니메이션, 영화 등 다양한 매체를 통합시킴으로써 여러 가지 정보를 복합적으로 학습자에게 제공한다. 멀티미디어 이용 학습은 학습 내용을 보다 현장감 있게 효과적으로 전달할 수 있

어 매우 효과적인 교육 매체로 인식되고 있으며, 앞으로도 멀티미디어 활용 교육의 형태는 더욱 다양해질 것으로 예측된다. 본 논문이 제안하고 있는 교육용 프로그래밍 언어 *TWo* (*Turtle World*)가 멀티미디어 기반 교육에 주안점을 두고 있는 이유도 거기에 있다. *TWo*는 피하미나 로고, EHLL보다 차원 높은 개념과 기능을 제공하는 교육용 프로그래밍 환경으로서, 그래픽이나 애니메이션, 사운드 등을 효과적으로 프로그래밍할 수 있는 다양한 기능을 추구한다. 또한, 객체(object) 개념[8]과 반영(reflection) 개념[5,15]을 도입하여 수준별 학습 환경을 효과적으로 구성할 수 있게 한다.

본 논문은 우선 *TWo* 설계시 고려한 사항을 설계 방향과 설계 원칙으로 구분하여 기술한 후, *TWo*의 기본 개념 및 기능을 바탕으로 주요 명령의 형식과 의미에 대해서도 논의한다. 이어서 관련된 프로그래밍 언어들과 비교해서 *TWo*가 지닌 장점과 특징을 설명하고, *TWo*의 목표를 기술하는 것으로 결론짓는다.

2. 설계 고려 사항

2.1 설계 방향

교육용 프로그래밍 언어 *TWo*는 컴퓨터 프로그래밍 교육이라는 기본 목적 이외에 멀티미디어 작업과 평생학습 도구로서의 활용을 그 목표로 하고 있다. 이러한 목표에 효과적으로 접근하는 기본 전략으로서 *TWo*의 설계 방향은 다음과 같다.

■ 수준별 학습 환경의 효과적 구성

범용 프로그래밍 언어와 다르게 교육용 프로그래밍 언어는 초등학생이나 더 나아가 유치원생까지 쉽게 익혀서 사용할 수 있어야 하고, 동시에 수준 높은 학습자를 위한 고급 프로그래밍 환경도 제공되어야 한다. 교육용 프로그래밍 언어 활용의 주안점이 언어 개념을 기반으로 문제 해결 과정에서 얻어지는 논리력과 사고력 배양에 있다면, 초급 환경에서 명령 형식 자체는 부차적 문제이다. 한글 혹은 아이콘(icon) 명령이 초급 환경의 한 요소가 될 수 있으며, 아직 외국

어에 익숙하지 않은 어린 학생들에게 한글 명령, 한글 변수(variable) 이름은 그 자체만으로 큰 친화력이 있다. 한글 명령에 의한 키 입력 횟수 증가 문제는 각 명령어에 대한 축약 명령어(약자 형식으로 원래 이름의 길이를 줄인 명령어)를 정의하거나, 명령의 첫 자만 입력하면 그 글자로 시작하는 명령이 자동으로 나타나게 함으로써 해결할 수 있다.

■ 멀티미디어 기능 강화

명령 실행 결과를 즉시 화면 상에 그림 형태로 나타내는 것은 학생들의 흥미와 호기심 유발 효과를 높이고 반영적 추상화에 의한 오류 수정을 용이하게 한다는 이점이 있다. *TWo*는 컴퓨터와 통신 기술의 발달에 따라 보편화되고 있는 멀티미디어 작업의 기본 요소로서 텍스트, 음성, 애니메이션, 동영상 등에 대한 프로그래밍 기능을 제공한다.

■ 반영적 객체(reflective object) 개념 적용

객체 지향 프로그래밍[8]의 효용성이 인정되면서 점차 이를 지원하는 프로그래밍 언어의 활용도가 높아지고 있으며, 특히 90년대 이후 개발된 객체 지향 언어 Java [6,10,11,12,13]는 최근 급속히 사용자가 늘어나고 있는 추세이다. 프로그래밍 언어에 도입되고 있는 객체 지향 개념은 실세계를 보다 효과적으로 모델링(modeling)하여 프로그래밍하게 할 뿐만 아니라, 코드의 재사용성, 코드의 안정성 등 여러 가지 측면에서 장점을 가지고 있다. 객체 개념과 더불어 *TWo*는 반영 개념을 도입하고 있다. 반영 개념은 시스템의 적응성(adaptability)을 높여 향후 새로운 기술과 새로운 요구 조건을 쉽게 수용할 수 있게 하며, 수준별 학습 환경의 구성을 용이하게 한다. 반영 구조를 기초로 *TWo*는 객체 지향 프로그래밍에 대한 개념 정립 없이도 초급 프로그래밍을 가능케 하며, 애니메이션 등의 고급 프로그래밍 기술을 효과적으로 구사하게 한다.

■ 학습 이론 적용

*TWo*는 기본적으로 교육용 프로그래밍 언

어이므로 학습 효과가 극대화되도록 지원하는 기능이 중요하다. 프로그래밍 언어의 본질을 이해하고 컴퓨터 원리를 학습하는 것 뿐만 아니라, 학생들의 논리력과 사고력을 배양하는데 도움을 줄 수 있게 설계되어야 한다.

2.2 설계 원칙

*TWo*는 언어이자 하나의 독자적인 시스템이다. 시스템이 지속적으로 활용되기 위해서는 그 설계 단계부터 시스템 사용의 편의성이나 효율성, 향후 대두될 수 있는 요구조건에 대한 적응성 등이 고려되어야 한다. *TWo* 시스템에 적용된 설계 원칙을 몇 가지로 정리하면 다음과 같다.

■ 일관성

기존 기능이나 새로 추가한 기능을 일관된 방법으로 사용할 수 있게 하며, 초급 환경에서 익힌 개념과 기능이 고급 환경까지 적용되도록 수준별 학습 환경을 구성한다.

■ 확장성

향후 새로운 데이터 형이나 화일 형식에 대한 지원이 요구될 때 혹은 명령어나 기능을 추가하거나 보완해야 할 때, 시스템 수준이나 사용자 수준에서 쉽게 수용할 수 있게 한다.

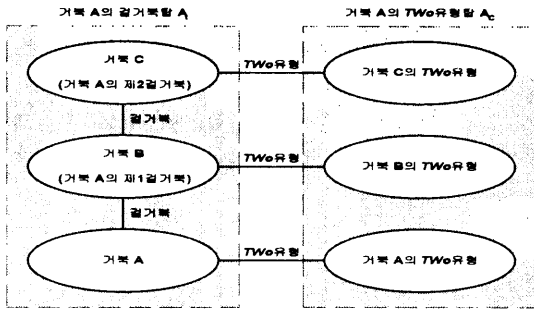
■ 호환성

로고 뿐만 아니라 피하미와 호환성을 지닌 명령 환경을 제공하여, 이미 이들 언어를 익힌 학습자들이 프로그래밍 작업을 수행하는데 문제가 없게 한다.

3. *TWo* 프로그래밍

3.1 기본 개념 및 용어

프로그래밍 언어 *TWo*는 '거북'들의 제반 활동을 표현하는 언어이며 거북들이 활동하는 공간(시스템)이다. *TWo*는 모든 활동체의 대명사인 '거북'을 통해 인간 세계에서 발생 가능한 제반 현상을



(그림 1) 걸거북탑/TWo유형탑과 거북

표현하는 언어로, 학문·예술 분야의 활동은 물론 인간과 자연에 의한 제반 현상을 표현할 수 있다. TWo 프로그래밍은 거북들의 활동을 제어하여 그림, 소리, 움직임이 가미된 멀티미디어 작품을 만드는 것이다. 거북은 속성(멀티미디어 효과를 위한 속성), 행동(멀티미디어 효과를 유발하는 활동), 모습(거북의 형태, 색깔) 등을 모델링하여 제어할 수 있는 대상이며, 거북들의 모습과 행동, 붓자취(거북이 붓으로 제작한 그림과 글씨) 등과 관련된 제반 효과를 TWo현상이라 한다.

TWo 내부의 유일한 모델링 대상은 거북이며, 거북 간의 상호작용은 명령(작업호출)을 매개로 일어난다. 명령은 특정 거북이 지정된 작업을 수행하도록 요청하는 것이고, 거북은 자신에게 요청된 명령을 직접 수행하거나 다른 거북을 통해 수행한다. 거북 T에게 작업 W를 수행하도록 요청하는 명령의 형식은 'T.W 인자리스트'로 여기서 '인자리스트'는 T가 작업 W를 수행할 때 필요로 하는 인자값들을 열거한 것이다. 거북 모델의 한 부류인 TWo 장치는 사용자에게 의해 유발되는 사건을 TWo 내부의 명령으로 전환해 주는, TWo와 사용자 간의 상호작용 매개체로, 사용자는 TWo 장치를 통해 TWo현상을 유발하고 개입한다.

TWo에서 거북은 (그림 1)에 보인 것처럼 다음 두 가지 대상에 의해 모델링되는 객체이다.

■ TWo유형

거북의 속성, 행동(움직임, 소리) 등 거북의 속모습에 대한 모델로, TWo의 모든 거북은 생성될 때부터 그 TWo유형이 설정된다.

■ 걸거북

거북 자신의 형태나 색 등 거북의 겉모습

에 대한 모델로, 불필요한 경우 설정하지 않을 수 있다.

TWo는 TWo유형/TWo모듈과 걸거북 개념을 기반으로 거북을 모델링한다. TWo유형과 TWo모듈은 객체지향 언어에서 사용되는 일반 개념인 클래스(class)에 대응되는 개념들이다. 클래스는 소프트웨어 재사용 단위로 상속(inheritance)에 기반한 클래스 계층구조(class hierarchy) 상에 정의되며, 객체 형으로서 프로그램 수행 중 인스턴스 객체(instance object)에 대한 연산을 정의한다. TWo에서 소프트웨어 재사용 단위는 TWo모듈이며, TWo유형은 객체형 역할을 담당한다. TWo모듈은 TWo 프로그래밍 단위로서 작업(명령 수행 절차)과 변수(값을 저장하는 장소)의 집합으로 정의하고, TWo유형은 TWo모듈들을 동적으로 조합하여 설정한다. TWo에서 특정 거북의 겉모습은 다른 거북을 이용해 모델링한다. 거북 A의 겉모습을 거북 B가 모델링할 때 거북 B가 발생시키는 멀티미디어 효과(소리, 붓자취 등)는 거북 A의 겉모습을 설정한다. 이 경우 TWo는 거북 B를 거북 A의 걸거북이라 부른다. 거북 A의 걸거북이 거북 B이고 B의 걸거북이 거북 C일 때, B와 C는 A에 대해 각각 제1걸거북과 제2걸거북이라 부르며, 이 때 C는 B의 제1걸거북이 된다. 특정 거북 T에 대해 제1걸거북, 제2걸거북, . . . , 제N-1걸거북이 설정되어 구성된 높이 N(> 0)의 탑(tower)을 거북 T의 걸거북탑 Tt라 하며, 거북 T를 Tt의 기저거북이라 한다. 걸거북탑을 구성하는 모든 거북에 대해 TWo유형이 설정되므로 걸거북탑 Tt에 대응되는 TWo유형탑(TWo유형들의 탑) Tc를 상정할 수 있으며, 이를 고려할 때 거북 T는 궁극적으로 TWo유형탑 Tc에 의해 모델링된다. 그림 1은 거북 A를 기저거북으로 한 높이 3의 걸거북탑 At와 TWo유형탑 Ac를 보여 주고 있다.

거북들이 TWo유형탑에 의해 모델링된다는 점에서 TWo유형은 TWo 구성의 핵심 요소이며, 그 특징은 다음과 같다.

- TWo유형은 특정 명령들에 대한 작업 집합을 정의한다.
- TWo유형은 모듈계층구조 상의 TWo모듈들의 집합으로 정의된다.

모듈계층구조는 *TWO*모듈 간의 상속 관계를 기반으로 설정되는 트리(tree) 형태의 계층 구조이다. 상속이란 한 *TWO*모듈이 다른 *TWO*모듈에 정의된 작업과 변수 집합을 물려받는 것을 의미하는데, 이 때 전자를 파생 모듈, 후자를 근원모듈이라 한다.

- *TWO*유형은 확장·축소할 수 있으며, 그 단위는 *TWO*모듈이다.

‘클래스’와 ‘*TWO*유형’, ‘객체’와 ‘거북’은 각각 서로 대응되는 유사 개념이지만, *TWO*유형과 거북은 뚜렷한 두 가지 특징을 가지고 있다. 그 첫째는 거북이 *TWO*유형답에 의해 모델링된다는 점이고, 둘째는 각 거북의 걸거북답을 동적으로 변경할 수 있게 함으로써 *TWO*유형답에 의한 거북 간 상호작용 양식을 동적으로 제어할 수 있다는 점이다.

*TWO*유형과 거북 개념에 기반한 *TWO* 구조의 특징을 정리하면 다음과 같다.

- 객체 지향 구조(object-oriented architecture)
거북은 특정 *TWO*유형답의 인스턴스이며, *TWO*유형은 상속 관계를 기반으로 정의된 *TWO*모듈들을 조합한 거북의 모델이다.
- 반영 구조(reflective architecture)
거북에 대해 어떤 *TWO*유형답을 설정하는 지에 따라 거북의 행동과 모습이 달라진다. 사용자는 *TWO*유형답을 동적으로 변경함으로써 거북들의 상호작용에 의해 빛어지는 *TWO*현상을 주관할 수 있다.
- 확장 구조(scalable architecture)
*TWO*는 *TWO*모듈 단위의 확장 가능 언어이다. *TWO*모듈을 상속/조합하여 새로운 *TWO*모듈을 정의할 수 있으며, 이를 통해 *TWO*의 확장성이 확보된다.

3.2 형과 식

*TWO*는 정수, 실수, 문자, 논리의 4 가지 기본형(primitive type)과 문자열, 배열(array)의 2 가지 참조형(reference type)을 정의하고 있으며, 사용자가 정의하는 참조형으로서 거북형(turtle type)을 지원한다. 참조형으로 정의된 대상을 객체(object)라 하고, 객체가 존재하는 위치값을 참조값

<표 1> *TWO* 연산자

연산자	형식	이름
[] ()	배열참조 [정수식] (명령)	배열원소참조 명령결과
*	산술식 * 산술식	곱하기
/	산술식 / 산술식	나누기
%	산술식 % 산술식	나머지
+	산술식 + 산술식	더하기
-	산술식 - 산술식	빼기
>	산술식 > 산술식	크다
<	산술식 < 산술식	작다
!	! 논리식	논리부정
==	식 == 식	같다
!=	식 != 식	다르다
&&	논리식 && 논리식	논리곱
	논리식 논리식	논리합

(reference)이라 한다. 기본값(기본형의 값)이나 참조값이 저장되는 장소를 변수(variable)라 하고, 기본값이 저장된 변수를 기본변수, 참조값이 저장된 변수를 참조변수라 하며, 참조변수에 저장된 참조값이 가리키는 객체를 해당 변수의 참조객체라 한다.

*TWO*는 식(expression)을 이용하여 하나의 값(기본값 혹은 참조값)을 표현하며, 다음 5 가지 기본식과 <표 1>의 연산자(operator)들이 식을 구성하는 요소이다.

- 상수(constant)
기본값을 표현하는 상수로서 십진값을 나타내는 정수상수(1354, -24 등)와 실수상수(3.24, -0.02 등), unicode값으로 표현되는 문자상수('a', '\n' 등)와 논리값을 표현하는 논리상수(참, 거짓)가 있고, 참조값을 표현하는 상수로서 참조객체가 없음을 나타내는 NULL이 있다.
- 문자열상수(string literal)
문자열("a string", "" 등 0개 이상의 문자들의 나열)이 저장된 객체에 대한 참조값을 표현한다.
- 변수명칭
기본변수인 경우 그 변수에 저장된 기본값을, 참조변수인 경우 그 변수에 저장된 참조값을 표현한다.

- (명령)

명령 수행 결과값(리턴값)의 형이 기본형 일 경우 기본값을, 참조형일 경우 참조값을 표현한다.
- (식)

“식”을 한 단위로 묶어 주며, “식”의 값과 동일한 값을 표현한다.

변수는 처음 생성될 때 작업변수로 설정되어, 그것을 생성한 작업 내에서만 이용 가능하고, 그 작업이 완료되면 자동적으로 소멸된다. 작업변수와 달리 모듈변수는 모듈 내의 모든 작업이 참조할 수 있고, 변수없애기 명령 이전까지 소멸되지 않는다. 변수의 형은 동적으로 바뀔 수 있으며, 특정 시점에서 변수의 형은 그 당시 해당 변수에 저장되어 있는 값의 형과 일치한다. *TWO*는 다음 명령을 통해 변수의 생성/소멸을 제어할 수 있게 지원한다.

- 저장하기 변수명칭 식

식의 값을 계산하여 주어진 명칭의 변수(해당 변수가 없을 경우 생성)에 저장한다. 다음은 몇 가지 관련 예들이다.

 - 저장하기 **max 3.2**

실수값 3.2를 변수 **max**에 저장한다.
 - 저장하기 **sum max + 5.5**

실수값 5.5를 변수 **max**에 더하여 그 결과값을 **sum**에 저장한다.
- 저장하기 변수명칭 (형명칭 식리스트)

지정된 형의 객체를 생성하고 그 참조값을 주어진 명칭의 변수(해당 변수가 없을 경우 생성)에 저장한다. 다음은 몇 가지 관련 예들이다.

 - 저장하기 **count (정수[5] 0 1 2 3 4)**

크기 5의 정수 배열을 설정하고 그 참조값을 변수 **count**에 저장한다. 0, 1, 2, 3, 4는 배열 **count**의 초기값으로 설정된다. 배열 **count**의 각 항목에 대한 참조값은 **count[0], count[1], . . . , count[4]** 형태의 식으로 표현된다.
 - 저장하기 **title (문자열 “Two 언어”)**

문자열 “Two 언어”로 초기화된 문자열 객체를 설정하고 그 참조값을 변

수 **title**에 저장한다.

- 저장하기 **artist (꼬마거북 3 2)**

인자 3, 2로 초기화된 꼬마거북 객체 (*TWO*모듈 “꼬마거북”이 *TWO*유형으로 설정된 객체)를 설정하고 그 참조값을 변수 **artist**에 저장한다.

- 변수남기기 변수명칭리스트

열거된 변수들을 모듈변수로 설정한다.
- 변수없애기 변수명칭리스트

열거된 변수들을 소멸시킨다.

3.3 *TWO*모듈과 *TWO*유형

*TWO*모듈(부록의 문법 참조)은 작업과 변수의 집합을 정의하며, *TWO*모듈 간의 유일한 상호작용 수단은 명령이다. *TWO*모듈 내에 정의된 작업은 동일 모듈 혹은 다른 모듈이 요청한 명령에 의해 구동되는데, 동일 모듈 내에서만 호출 가능한 작업을 내부작업, 다른 모듈도 호출할 수 있는 작업을 외부작업이라 하며, 이들을 정의하는 명령 형식은 다음과 같다.

- 외부작업 헤더 절차 명령리스트 끝내기
- 내부작업 헤더 절차 명령리스트 끝내기

거북은 생성된 이후 *TWO*유형 뿐만 아니라 걸거북을 바꿀 수 있어 특정 거북의 걸거북탑이나 *TWO*유형탑은 동적으로 변경된다. 이와 관련된 명령의 형식과 기능은 다음과 같다.

- 속거북빛기 모듈명칭 인자리스트

거북의 *TWO*유형을 다시 초기화하는 명령으로, 지정된 *TWO*모듈을 열거된 인자들로 초기화한다.
- 겹거북빛기 식

식값(참조값)이 지정하는 거북을 겹거북으로 설정한다.
- 모듈들이기 모듈명칭 인자리스트

거북의 *TWO*유형에 지정된 *TWO*모듈을 추가하는 명령으로, 열거된 인자들에 의해 추가되는 *TWO*모듈이 초기화된다.
- 모듈버리기 모듈명칭

속거북에서 지정된 *TWO*모듈을 제거한다.

*TWO*모듈은 첫작업이라는 명칭의 외부작업에 의해 초기화되는데, 일례로 “저장하기 artist (꼬마거북 3 2)” 명령은 *TWO*모듈 꼬마거북의 첫작업을 인자값 3과 2로 호출함으로써 거북 인스턴스를 초기화한다. 속거북빛기나 모듈들이기 명령의 경우에도 지정된 모듈의 첫작업에 의해 초기화된다. 이와 반대로 끝작업이라는 명칭의 외부작업은 모듈버리기 명령처럼 해당 모듈이 제거될 때 관련된 마무리 작업을 수행한다. 첫작업이나 끝작업이 정의되지 않은 *TWO*모듈의 경우 *TWO*가 정의한 작업들에 의해 초기화나 마무리 작업이 수행된다.

3.4 인자전달과 부분호출

*TWO*의 명령은 특정 작업에 대한 호출로서, 호출(명령)하는 작업(명령자: caller)과 호출 받는 작업(수행자: callee) 간에 인자 전달(arguments passing)과 동기화(synchronization)라는 두 가지 측면의 상호작용 요소가 수반된다. 인자전달 관점에서 *TWO*는 “값에 의한 호출(call by value)” 방식을 지원하는데, 이는 callee가 명령의 인자값(기본값 혹은 참조값)들을 전달받아 지시된 작업을 수행할 수 있게 한다. 동기화 관점에서 *TWO*의 핵심은 부분명령이며, 이는 2 개 이상의 명령(호출)을 조합하여 하나의 작업을 호출하는 개념이다.

명령자와 수행자 간의 일반적인 명령/수행 과정을 정리하면 다음과 같다.

- 1) 명령의 인자(argument: 식으로 명시)값들이 명령자 측에서 계산된다.
- 2) 인자값들이 매개변수(parameters)에 저장되면서 수행자 측으로 제어가 전달된다.
- 3) 작업 수행 완료 후 결과값(리턴값)과 제어가 명령자 측으로 돌아온다.

명령의 인자(arguments) 개수는 해당 작업의 매개변수(parameters) 개수와 일치해야 하며, 인자와 매개변수는 위치 순서에 의해 일대일 대응된다. 각 매개변수의 형은 대응되는 인자값의 형과 일치하게 된다.

3.3절에서 기술한 작업정의 형식에서 “헤더” 부분은 다음과 같이 두 가지 형식으로 정의될 수 있다.

■ add a b

■ com x y z (com1 y) (com2 x z) (com3)

여기서 ‘add a b’나 ‘com x y z’ 부분을 주헤더라 하고, ‘com1 y’, ‘com2 x z’, ‘com3’ 각각을 부분헤더라 한다. 작업 add가 주헤더만 명시되어 있는데 반해, 작업 com은 부분헤더들까지 명시되어 있다. 이 경우 작업 add에 대한 명령 형식은 “add 식 식”으로 한정되지만, 작업 com의 경우 “com 식 식 식” 형식의 명령 이외에 부분명령(부분헤더에 대한 호출)을 이용하여 com 작업을 수행시킬 수 있다. 작업 com의 부분헤더 각각에 대한 부분명령의 형식은 ‘com1 식’, ‘com2 식 식’, ‘com3’로, 이 3 가지 부분명령은 프로그램의 어떤 위치(동일한 *TWO*모듈 혹은 다른 *TWO*모듈)에서도 요청될 수 있으며 이들이 모두 호출되었을 때 작업 com이 수행된다. 작업 com의 수행이 완료된 후 제어흐름은 첫 번째로 정의된 부분헤더에 대한 부분명령, 즉 ‘com1 식’의 명령자 측으로 복귀된다. 이점에서 부분명령 수행과 관련하여 제어흐름의 주축이 되는 첫 번째 부분헤더를 제어헤더라 한다. *TWO*의 부분명령 개념은 여러 개의 비동기적 제어 흐름을 동기화해야 할 경우 활용된다.

3.5 제어흐름

*TWO*모듈 내의 모든 작업은 특정 순서로 수행될 일련의 명령들로 정의된다. 원칙적으로 작업 내의 명령들은 열거된 순서대로 수행되지만, 이러한 순차적 제어 흐름은 다음 명령들에 의해 바뀌어질 수 있다.

- 검사하기 식 참일때 명령리스트 끝내기
“식”의 값이 참일 때에 한해 “명령리스트”에 열거된 명령들을 수행한다.
- 검사하기 식 참일때 명령리스트 거짓일때 명령리스트 끝내기
“식”의 값이 참이면 “참일때” 다음의 “명령리스트”를 수행하고, 그렇지 않으면 “거짓일때” 다음의 “명령리스트”를 수행한다.
- 반복하기 식 명령리스트 끝내기
“식”의 값이 참인 동안 “명령리스트”에 열거된 명령들을 반복하여 수행한다.
- 돌아가기 식

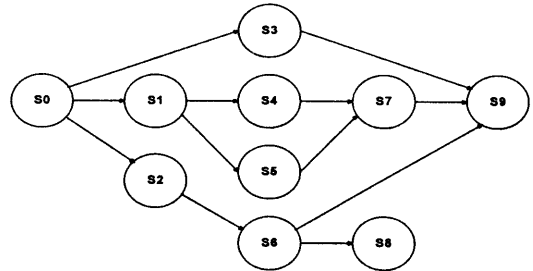
“식”의 값을 결과값(리턴값)으로 돌려주면서 명령 수행을 완료한다.

명령은 *TWo* 내부의 상호작용 수단으로 *TWo* 유형 내에 정의할 작업의 절차를 기술하는 논리적 단위이며, 거북들이 *TWo* 유형에 기술된 일련의 명령을 수행함으로써 빛어지는 제반 상황이 *TWo* 현상이다. *TWo* 현상에 대한 표현력은 거북들이 수행하는 명령들이 얼마나 세밀하게 거북의 상호작용을 제어할 수 있는 지에 의존한다. *TWo*는 명령의 수행 양식을 다양화함으로써 거북의 개별 행동 뿐만 아니라 거북들 간의 동기화까지 효과적으로 제어할 수 있게 한다. *TWo*가 제공하는 명령 수행 양식은 자체성, 조합성, 병행성 등의 3 가지 기준에 의해 다음과 같이 분류된다.

- 자체성에 의한 구분
 - 거북명령: 지정된 거북 자체에 대한 명령
 - 겹거북명령: 지정된 거북의 겹거북에 대한 명령으로, “명령이름^ 인자리스트”에서와 같이 명령이름 뒤에 ^를 붙여 거북명령과 구분한다.
- 조합성에 의한 구분
 - 단일명령: 명령 수행 결과가 명령 단위로 *TWo* 장치에 반영되는 명령
 - 조합명령: 두 개 이상의 명령 수행 결과가 조합되어 *TWo* 장치에 반영되는 명령으로, 명령리스트(조합할 명령들의 나열 형태)를 “묶어 명령리스트 끝내기” 형식으로 기술한다.
- 병행성에 의한 구분
 - 동기명령: 해당 작업이 완료된 후 다음 명령이 수행되는 명령
 - 비동기명령: 해당 작업이 시작된 후 다음 명령이 수행되는 명령으로, 다음 두 가지 형식으로 기술할 수 있다.
 - ☞ “명령이름’ 인자리스트” 형식: 명령이름 뒤에 ’ 붙여 동기명령과 구분
 - ☞ “따로 명령리스트 끝내기” 형식: 명령리스트 부분에 열거된 명령들은 순차적으로 수행되는 동기명령

*TWo*의 비동기명령과 부분명령을 효과적으로 활용할 경우 S0부터 S9까지의 10가지 작업 단위를

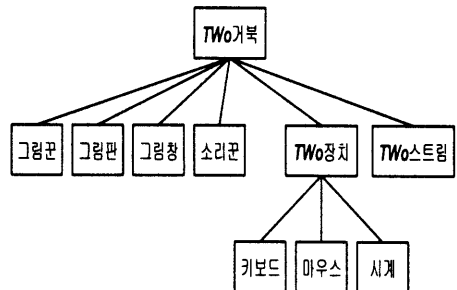
(그림 2)의 선행 관계가 만족되도록 수행시킬 수 있다. S0 완료 후 복수 개의 작업(S3, S1, S2)을 구동해야 하는 경우, S3과 S2를 각각 비동기명령으로 구동시킨 후 S1을 순차적으로 수행해 가는 방법을 활용할 수 있다. 복수 개의 작업(S4, S5)을 모두 완료한 후 S7이 구동되어야 하는 경우는 작업 S7을 두 개의 부분헤더를 설정하여 정의하고 S4와 S5의 마지막 단계에서 S7에 대한 부분명령을 요청하는 방식으로 표현할 수 있다.



(그림 2) 선행그래프의 예

3.6 모듈계층구조

TWo 모듈들은 그것을 정의한 목적에 따라 크게 시스템모듈, 환경모듈, 응용모듈로 나뉜다. 시스템모듈은 *TWo*가 언어(시스템)의 기본 구성을 위해, 환경모듈은 사용자를 위한 수준별 학습 환경을 구성하기 위해, 응용모듈은 사용자가 응용 개발을 위해 정의하는 *TWo*모듈이다. 이들은 모두 하나의 트리형 모듈계층구조로 통합되어 관리되며, 그 구간을 이루는 부분은 (그림 3)에 보인 시스템모듈들의 상속구조이다.



(그림 3) 시스템모듈 상속 구조

3.6.1 그림판과 소리꾼

*TWo*가 거북의 활동을 통해 표현하고자 하는

것은 *TWO*현상이며 이는 그림과 소리를 통해 사용자에게 전달된다. *TWO*가 그림과 소리의 제작자로서 정의하고 있는 *TWO*모듈은 각각 그림꾼과 소리꾼이며, 그림판과 그림창은 그림꾼 거북들에 대한 관찰 방법을 설정할 수 있게 지원되는 *TWO*모듈이다. 이들이 정의하고 있는 작업을 정리하면 다음과 같다.

■ 그림꾼

- 옮기기 상대좌표
그림을 지정된 방향과 거리만큼 이동시킨다.
- 돌리기 상대각도 중심점
그림을 주어진 중심점을 기준으로 상대각도만큼 회전시킨다.
- 감추기
그림을 보이지 않게 한다.
- 보이기
그림을 보이게 한다.
- 지우기
그림을 지운다
- 선긋기 색깔 굵기 첫좌표 끝좌표
지정된 색깔과 굵기로 주어진 두좌표 간의 선을 긋는다.
- 문자열쓰기 색깔 크기 모양 문자열
주어진 문자열의 문자들을 지정된 색깔, 크기, 모양(font)으로 쓴다.
- 칠하기 색깔 좌표
주어진 좌표를 포함한 폐공간을 지정된 색으로 칠한다.
- 본찍기 그림본 좌표
지정된 그림본(이미지)을 주어진 위치에 적재시킨다.

■ 그림판

- 첫작업 좌상단좌표 우하단좌표
지정된 좌표영역을 지닌 그림판을 설정한다.
- 그림겹치기 좌상단좌표 거북명칭리스트
그림꾼들의 붓자취를 겹쳐 놓는다.

■ 그림창

- 첫작업 좌상단좌표 우하단좌표
지정된 영역에 그림창을 설정한다.
- 창저장하기 화일

지정된 화일에 그림창을 저장한다.

• 창보이기

그림창을 보이게 한다.

• 창감추기

그림창이 보이지 않게 한다.

• 그림판놓기 좌상단좌표 그림판

그림창 위의 지정된 영역에 그림판을 배치한다.

■ 소리꾼

• 소리내기 소리

지정된 소리를 낸다.

• 소리끊기

소리내기 작업을 멈춘다.

3.6.2 TWO장치

*TWO*장치는 마우스, 키보드, 타이머 등에 대한 추상체(abstraction)로서 *TWO* 외부 세계의 사건을 내부화시키는 거북이다. 이들 장치에서 발생한 사건은 대응되는 *TWO*장치를 통해 다른 거북에게 전달되는데, 사건을 전달받으려 하는 거북은 그 참조값을 해당 *TWO*장치에 등록해야 하며, 이를 위해 모든 *TWO*장치가 제공하는 작업의 형식은 다음과 같다.

등록하기 거북참조값리스트

등록하기 명령을 통해 등록되는 거북은 *TWO*장치로부터 사건을 전달받기 위해 몇 가지 작업을 정의하고 있어야 하며, *TWO*장치별로 그 인터페이스를 정리하면 다음과 같다.

■ 키보드

• 키눌림반응하기 키값

• 키놓임반응하기 키값

■ 마우스

• 마우스이동반응하기 좌표

• 마우스눌림반응하기 좌표 버튼

• 마우스클릭반응하기 좌표 버튼 횟수

• 마우스끌림반응하기 좌표

• 마우스놓임반응하기 좌표 버튼

■ 시계

• 알람반응하기 시각

3.6.3 TWO스트림

*TWO*스트림은 데이터 입출력을 지원하는 *TWO* 모듈로 다음 작업을 정의하고 있다.

- 첫작업 화일경로명 참조방법
지정된 화일을 주어진 방법으로 참조할 수 있도록 준비한다.
- 읽기 형이름의문자열
해당 *TWO*스트림에서 읽어 들인 데이터로 지정된 형의 값이나 객체를 설정하여 돌려준다.
- 쓰기 식
해당 *TWO*스트림으로 주어진 식의 결과값(참조형일 경우 대상 객체의 형에 따른 데이터 레코드)을 저장한다.
- 위치정하기 식
해당 *TWO*스트림의 입출력 위치를 레코드 단위 오프셋(offset)으로 옮긴다.

4. 관련 연구

*TWO*는 1960년대 후반 MIT의 인공지능연구실에서 교육용 언어로 개발되어 지금까지 진화되어 온 언어 로고(Logo)의 “거북그래픽(turtle graphics)” 기능을 도입하여 설계되었다. 이 기능은 사용자가 명령을 입력하면 즉시 화면 상의 거북이 그 명령을 수행하는 것으로, 명령에 대한 가시적/즉시적 반영은 학습자의 학습동기를 유발하고 학습의욕을 고취시키는 데 큰 역할을 한다. 로고와 *TWO*를 비교하면 다음과 같다.

- 두 언어 모두 인터프리터 언어이다.
- 로고의 거북 모양은 하나로 고정되어 있는데 반해, *TWO*의 거북 모양은 프로그래밍이 가능하며 명령에 의해 바뀔 수 있다.
- 로고는 하나의 거북을 대상으로 명령하지만, *TWO*는 복수개의 거북들이 상호 작용하는 상황을 표현할 수 있다.
- 로고는 절차 지향 언어(procedure-oriented language)이고, *TWO*는 객체 지향 언어(object-oriented language)이다.
- 로고의 확장 단위는 절차(procedure)이고, *TWO*의 확장 단위는 모듈(module)이다.

*TWO*는 어린이로부터 성인에 이르는 포괄적인

사용자 층을 대상으로 한 프로그래밍 학습 도구로 설계되었다. 본 논문에서 기술한 개념 및 기능은 고급 수준의 내용이지만 이는 *TWO*의 핵심부에 한정된 내용이며, 실제 학습자는 사용자 수준별로 제공되는 *TWO* 사용자 인터페이스를 이용하여 *TWO* 프로그래밍을 하게 된다. 내부 구현을 몰라도 그에 대한 명령의 형식과 기능만 알면 *TWO*모듈이 제공하는 기능을 활용할 수 있기 때문에, 이를 이용한 수준별 학습 환경은 쉽게 구현될 수 있다. 일례로 로고 명령을 지원하는 몇 개의 *TWO*모듈을 추가하면, *TWO*를 외관상 로고와 동일한 언어로 만들 수 있다. 이점에서 *TWO*는 1970년에 교육용 언어로 개발된 절차지향 언어 Pascal과 다른 위치에 설 수 있다. Pascal은 처음부터 고도의 프로그래밍 개념과 기술을 학습자에게 노출시킨다. 형에 대한 명확한 개념과 프로그램 구조에 대한 이해를 전제로 프로그래밍이 가능하다. 이에 반해 초급 환경으로서의 *TWO*나 로고는 형 개념 없이도, 프로그램 전체 구조를 알지 못해도 사용할 수 있다. BASIC (Beginner's All-purpose Symbolic Instruction Code) 역시 초급 프로그래밍 교육에 적합한 대화식 언어로 개발되어 중등 및 고등 교육에 널리 쓰이고 있지만, 형 개념이나 구조적 프로그래밍 개념 등의 측면에서 취약한 단점을 지니고 있다.

교육용 언어 *TWO*의 목적과 관련된 주요 특징을 정리하면 다음과 같다.

- 프로그래밍 학습 도구
*TWO*는 형, 구조적 프로그래밍, 객체/클래스 개념, 반영(reflection) 개념 등과 관련된 기능을 제공함으로써 프로그래밍 언어에 대한 효과적 학습을 지원한다.
- 멀티미디어 작업 도구
*TWO*는 거북의 상호작용을 통해 멀티미디어 효과를 표현하는 작업도구로서, 다양한 모델로 설정된 복수 개의 거북을 통해 그래픽/애니메이션 작품을 효과적으로 제작할 수 있게 한다.
- 평생 학습 도구
*TWO*는 “거북 그래픽”과 수준별 학습 환경을 통해 다양한 층의 학습자들이 종합적 사고력과 문제 해결 능력을 높일 수 있게

한다.

5. 결론 및 향후 과제

본 논문에서 그 기본 개념과 기능을 기술하고 있는 *TWO*는 멀티미디어와 웹 기술에 기반한 새로운 교육용 프로그래밍 언어로, 본 논문의 내용은 *TWO* 핵심부에 대한 개념 설계를 바탕으로 하고 있다. 교육 현장에서 활용될 수 있는 실용적 학습 도구를 목표로 하는 *TWO*는 최근에 대두되고 있는 멀티미디어 활용 교육과 평생 학습에 대한 한 가지 해결책으로서 개발되고 있다. *TWO*가 문제 해결을 위해 취하고 있는 두 가지 주요 접근방법은 프로그래밍을 통한 멀티미디어 작업과 수준별 학습 환경이다. 첫 번째 접근방법은 멀티미디어(그림, 소리, 애니메이션 등) 효과를 촉매로 프로그래밍이 갖는 탁월한 교육적 효용성(종합적 사고력 향상, 문제 해결 능력 제고 등)을 크게 증진시키는 역할을 한다. 두 번째 접근 방법은 *TWO* 프로그래밍과 *TWO* 구조의 근간을 이루는 고급 개념(객체 개념, 반영 개념 등)과 기능(사건 처리, 동기화 등)에 대한 노출폭을 사용자 수준에 맞게 조절함으로써 *TWO*를 평생 학습의 대상으로 만든다. *TWO*가 한글 명령을 지원하는 것도 이와 맥락을 같이 한다.

로고(Logo) 프로그래밍 환경의 프로토타입을 Java로 구현하였고, 현재 이 경험을 기반으로 *TWO* 시스템을 상세 설계하고 있으며, *TWO* 프로토타입 구현 작업과 병행하여 웹 환경[9]과 다양한 학습패턴(협력학습 등)을 지원하기 위해 갖추어야 할 *TWO*의 구조적·기능적 요건을 연구해 갈 계획이다.

참 고 문 헌

- [1] 백영균(1988). 컴퓨터 프로그래밍에 대한 심리학적 접근 : Logo를 중심으로. 교육공학연구, 제4권 제1호, pp. 145-165.
- [2] 백영균(1995). 한국형 로고(LOGO). 한국교원대학교.
- [3] 송태욱, 안성훈, 김태영(1999). 객체지향개발기법을 이용한 교육용 한글 로고 언어의 설계 및 구현. 한국컴퓨터교육학회 논문지, 제2권 제4호, pp. 149-156.
- [4] 이옥화(1993). 로고 프로그래밍의 교육적 의의와 실천 방안 모색. 교육공학연구, 제8권 제1호, pp. 81-102.
- [5] Akinori Yonezawa and Takuo Watanabe(1988). An Introduction to Object-Based Reflective Concurrent Computation. Proceedings of the 1988 ACM SIGPLAN Workshop on Object-Based Concurrent Programming, pp. 50-54.
- [6] Bill Venners(1998). INSIDE the JAVA Virtual Machine. McGraw-Hill.
- [7] Cleborne D. Maddux(1985). Logo in the Schools. The Hawarth Press, Inc.
- [8] David N. Smith(1991). Concepts of Object-Oriented Programming, McGraw-Hill.
- [9] Dennis Volpano and Geoffrey Smith(1996). On the Systematic Design of Web Languages. ACM Computing Surveys, Vol. 28, No. 2, pp. 315-316.
- [10] Gosling, J. and McGilton, H(1996). The Java Language Environment: A White Paper. Sun Microsystem Inc.
- [11] James Gosling, Bill Joy, and Guy Steele(1996). The Java Language Specification. Addison-Wesley.
- [12] Ken Arnold and James Gosling(1996). The Java Programming Language. Addison-Wesley.
- [13] Kim Tolpey(1998). CORE Java Foundation Classes. Prentice Hall PTR.
- [14] Papert, S(1980). Mind Storms: Children, Computers, and Powerful Ideas. New York, Basic Books.
- [15] Pattie Maes(1987). Concepts and Experiments in Computational Reflection. Proceedings of the OOPSLA'87, pp. 147-155.

부록: 모듈 관련 문법

- 모듈 :
- 작업정의리스트
- 작업정의리스트 :
- 작업정의
- 작업정의리스트 작업정의
- 작업정의 :
- 외부작업 헤더 절차 명령리스트 끝내기
- 내부작업 헤더 절차 명령리스트 끝내기
- 헤더 :
- 작업헤더
- 헤더 (작업헤더)
- 작업헤더 :
- 명칭 명령리스트^{opt}
- 명령리스트 :
- 명령
- 명령리스트 명령
- 명령 :
- 길거북명령
- 배동기명령
- 기본명령
- 시스템명령
- 길거북명령 :
- 명령어^{opt} 시리즈^{opt}
- 배동기명령 :
- 명령어^{opt} 시리즈^{opt}
- 기본명령 :
- 명령어^{opt} 시리즈^{opt}
- 시스템명령 :
- 뛰어 명령리스트 끝내기
- 따로 명령리스트 끝내기
- 검사하기 식 참일때 명령리스트 끝내기
- 검사하기 식 참일때 명령리스트 거짓일때 명령리스트
- 끝내기
- 반복하기 식 명령리스트 끝내기
- 돌아가기 식^{opt}
- 지정하기 명칭 식
- 지정하기 명칭 (명칭 시리즈)
- 변수남기기 명령리스트
- 변수없애기 명령리스트
- 속거복빛기 명칭 시리즈
- 걸거복빛기 식
- 모듈들이기 식 시리즈
- 모듈버리기 명칭
- 명칭리스트 :
- 명칭
- 명칭리스트 명칭
- 시리즈 :
- 식
- 시리즈 식

김 성 백



1989 서울대학교 컴퓨터공학과 (학사)
 1991 서울대학교 컴퓨터공학과 (석사)
 1995 서울대학교 컴퓨터공학과 (박사)
 1996~현재 제주대학교 컴퓨터교육과 조교수
 관심분야: 컴퓨터교육, 컴퓨터구조, 컴퓨터통신
 E-mail: sbkim@educom.cheju.ac.kr

김 철 민



1988 서울대학교 컴퓨터공학과 (학사)
 1990 서울대학교 컴퓨터공학과 (석사)
 1996 서울대학교 컴퓨터공학과(박사)
 1997~현재 제주대학교 컴퓨터교육과 조교수
 관심분야: 컴퓨터교육, 운영체제, 프로그래밍
 E-mail: cmkim@educom.cheju.ac.kr

이 정 훈



1988 서울대학교 컴퓨터공학과 (학사)
 1990 서울대학교 컴퓨터공학과 (석사)
 1990.3~1992.2 대우통신 전송연구실
 1992.3~1996.2 서울대학교 컴퓨터공학과(박사)
 1996.3~1997.2 대우통신 광통신연구실 (선임)
 1997.3~현재 제주대학교 전산통계학과 조교수
 관심분야: 실시간통신, 분산시스템, 멀티미디어통신
 E-mail: jhlee@venus1.cheju.ac.kr