분산환경을 지원하는 웹 기반의 프로그래밍 언어 실습 시스템

노미라[0] 이수현
창원대학교 전자계산학과
mrroh@pl.changwon.ac.kr, suhyun@changwon.ac.kr

A Web-based Programming Language Practice System supporting Distributed Environment

Mi-Ra Roh[0] Su-Hyun Lee
Dept. of Computer Science, Changwon National University

요 약

현재 다양한 패러다임의 수 많은 프로그래밍 언어가 존재하고 있으며, 각각의 프로그래밍 언어를 실습하려면 해석기(interpreter)나 컴파일러(compiler) 같은 언어처리기를 갖추는 것이 필요하다. 언어처리기를 개별적으로 갖추는 것은 설치에 대한 시간적 부담, 시스템 자원 낭비, 업그레이드의 필요성 등으로 인해서 언어 학습과 직접적인 관련이 없는 부분에 대한 부담이 커진다.

본 논문에서는 WWW 환경에서 프로그래밍 언어를 실습할 수 있는 시스템을 구축하였다. 실습하는 프로그래밍 언어는 특정 서버에 제한적이지 않고, 네트워크를 통한 분산환경에서 확장이 용이하다. 또한 실습환경을 설정하는 구성 파일은 구조화된 문서의 작성을 지원하는 XML을 이용하여 관리자가 쉽게 구성파일을 작성할 수 있도록 했고, 자바의 정책(policy) 파일을 이용해 시스템 자원 사용 허가를 투명하게 했다.

1. 서 론

프로그래밍 언어는 주어진 문제의 해법을 정확하고 간결하게 표현할 수 있어야 하며, 프로그래머는 문제의 특성에 따라 적절한 언어를 선택할 수 있는 능력이 요구된다. 현재 다양한 패러다임의 수 많은 프로그래밍 언어가 존재하고 있으며, 이들 프로그래밍 언어들은 각각의 언어가 잘 적용될 수 있는 응용분야를 가지고 있다. 따라서 다양한 프로그래밍 패러다임에 대한 연습과 여러 프로그래밍 언어에 대한 실습이 필요하다.

다양한 프로그래밍 언어를 실습하려면 해석기나 컴파일러 같은 언어처리기를 갖추어야 한다. 그러나 언어처리기를 개인의 시스템에 설치하여 사용하는 것은 다음과 같은 문제점이 있다. 첫째, 언어 번역 프로그램을 각각의 시스템에 설치하는 것은 시스템 자원의 낭비를 가져온다. 둘째, 학습자는 비슷한 유형의 언어를 비교/분석하기 위해서 각각의 언어처리기를 설치하고 사용하여야 하지만, 많은 종류의 번역 프로그램 설치는 많은 시간을 요구하며, 각각의 설치과정이 다르므로 설치가 힘들다. 셋째, 업그레이드(upgrade)를 할 경우 각각을 모두 다시 설치해야 하므로 많은 시간이 낭비된다.

이러한 기존의 개별적으로 언어처리기를 유지하는데 발생하는 여러 문제점들에 대한 해결방안으로서, WWW의 일관된 인터페이스로 쉽고 간편하게 프로그래밍 실습을 할 수 있는 시스템을 구축한 바 있다[1]. 구축했던 시스템은 여러 언어에 대하여 언어처리기를 동시에 제공하며, 한 언어에 대하여 여러 버전의 언어처리기가 존재하는 경우 이들을 동시에 제공할 수 있고, 편리한 사용자 인터페이스를 제공한다.

본 논문에서는 이전에 구현했던 프로그래밍 언어 가상 실습 환경이 애플릿을 가져온 서버에 있는 프로그래밍 언어만을 실습할 수 있다는 제약을 개선하여, 서로 다른 서버들에 설치되어 있는 언어들에 접근함으로써 실습할 수 있는 프로그래밍 언어의 확장을 용이하게 했다. 이러한 서버의 확장으로 인해 특정한 서버에만 존재하는 해석기나 컴파일러라고 할지라도 실습 시스템의 서버만을 둠으로써 실습이 가능하고, 한쪽으로 치우치는 서버의 부하가 분산될 수 있다. 또한 시스템을 설정하는 구성파일은 구조화된 문서의 작성을 지원하는 XML[2]을 이용하여 관리자가 쉽게 구성파일을 작성할 수 있게 했고, 자바의 정책 파일을 이용해 시스템 자원 사용 허가를 투명하게 했다[3].

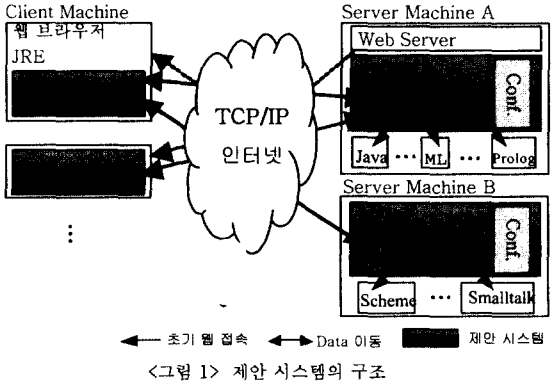
논문의 구성은 다음과 같다. 2장은 제안 시스템의 구조와 동작과정, 구성파일, 시스템 자원에 권한을 준 방법을 살펴보고, 3장은 구현 시스템과 타 시스템과의 비교 결과를 보이고, 4장에서는 결론과 향후 연구에 대하여 기술한다.

2. 제안 시스템의 구조와 동작

2.1. 제안 시스템의 구조

제안 시스템은 클라이언트/서버 모델[4]로 실제적인 실행은 서버에서 이루어지며, 전체 시스템에 대한 관리도 서버에만 집중되므로 클라이언트 쪽의 사용자의 부담이 경감된다.

제안 시스템의 구조는 <그림 1>과 같다. 웹으로 실습 홈페이지에 접근하면 클라이언트 프로그램인 자바 애플릿을 서버로부터 가져와서 실행한다. 애플릿이 실행되면 웹과는 다른 채널로 서버와 통신을 수행한다.



<그림 1> 제안 시스템의 구조

애플릿의 화면 구성과 실습 언어의 종류는 서버에 있는 구성파일(configuration file)에 의해 결정되며, 서버 관리자는 이 구성파일을 수정함으로써 실습 언어를 변경/추가 할 수 있다. 또한 학습자가 우연 또는 고의로 악성 프로그램을 작성하여 실행하는 것을 방지하기 위해 서버에 필터(filter) 기능을 포함하였다. 서버의 시스템을 조작하거나 디스크를 접근하는 등의 작업은 필터링을 통하여 거부될 수 있다.

<그림 1>에서 보듯이 실습 서버는 웹 서버와는 독립적으로 동작하며, 클라이언트가 요청한 프로그래밍 언어를 실행하여 실행한 프로세스와 클라이언트를 서로 연결해 주는 역할을 한다.

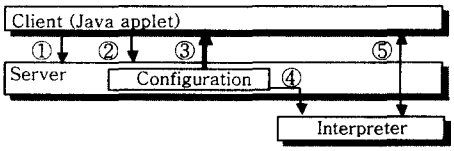
2.2. 서버 확장

기존의 시스템에서는 애플릿을 가져온 서버에 설치되어 있는 해석기나 컴파일러의 실습만이 가능했으나, 외부의 서버에 있는 프로그래밍 언어를 실습할 수 있도록 서버를 확장 시켰다. 서버의 확장으로 인해 실습할 수 있는 프로그래밍 언어의 확장이 용이해졌으며, 특정한 서버에만 존재하는 해석기나 컴파일러라고 할지라도 실습이 가능하고, 한쪽으로 집중되는 서버의 부하를 분산 시킬 수 있다. 이 과정시 외부 서버에 있는 애플릿을 다시 가져오는 것이 아니라, 외부 서버에 있는 프로그래밍 실습 시스템의 서버에 바로 접근할 수 있도록 애플릿에 네트워크 권한을 자바의 정책 파일을 이용해 설정했다. 권한을 주는 부분은 2.4절에서 자세히 보도록 한다.

2.2.1 서버의 인터프리터 실행

인터프리터는 작성한 프로그램을 바로 수행하는 번역 프로그램이다. 그리하여 학습자가 인터프리터를 요청할 경우 서버는 인터프리터를 실행하여 클라이언트와 연결되도록 해 주어야 한다. 다음은 그 동작 과정이다.

(1) " EXEC" 전송
(2) 인터프리터 이름(Section Name) 전송
(3) Section 정보 전송 (화면 구성 정보)
(4) 인터프리터 실행
(5) 클라이언트와 인터프리터가 서버를 통하여 통신

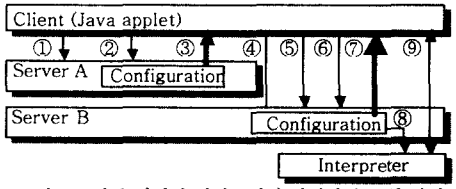


<그림 2> 인터프리터 작업환경 요청 과정

2.2.2 외부 서버의 인터프리터 실행

외부 서버의 인터프리터 실행시 애플릿을 가져온 서버의 구성파일에는 외부 서버의 주소와 포트 번호 정보만을 가지고 있으며, 이 정보를 클라이언트에 주면 클라이언트는 이 정보로 외부 서버와 소켓을 연결한다. 이 과정시 애플릿을 다시 가져올 필요는 없으며, 그 이후의 동작은 2.2.1 절의 서버의 인터프리터 실행 과정과 동일하다. 다음은 그 동작 과정이다.

(1) " REXEC" 전송
(2) 인터프리터 이름(Section Name) 전송
(3) 외부 서버의 주소와 포트 번호 전송
(4) 외부 서버와 소켓 연결
(5) " EXEC" 전송
(6) 인터프리터 이름(Section Name) 전송
(7) Section 정보 전송 (화면 구성 정보)
(8) 인터프리터 실행
(9) 클라이언트와 인터프리터가 서버를 통하여 통신

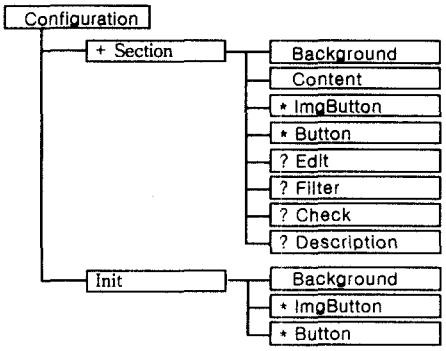


<그림 3> 외부 서버의 인터프리터 작업환경 요청 과정

2.3. 구성(Configuration)

서버와 클라이언트에서 필요로 하는 정보는 구성파일에 저장되어 있으며, 클라이언트의 화면 구성, 실습 언어의 종류 등 시스템의 전체 동작의 많은 부분이 이곳의 정보들을 중심으로 작동한다.

기존의 시스템은 필요한 정보를 하나의 텍스트 파일로 저장했으나, 관리자가 쉽게 구성파일을 작성할 수 있게끔 구조화된 문서의 작성을 지원하는 XML을 이용해서 작성했다. 구성파일은 XML 문서와 DTD 문서로 이루어져 있으며, 구성 요소의 구조도는 <그림 4>와 같다.



<그림 4> 구성 요소의 구조도

<그림 4>에서 보는 바와 같이 구성파일은 Configuration을 루트요소로 가지고 있으며, 자식노드로 하나 이상의 Section과 하나의 Init 노드를 가진다. Init은 애플릿 실행시 초기화면으로 배경화면과 실습 언어들에 접근할 수 있는 버튼들로 이루어져 있으며, Section은 실습 언어가 위치한 곳의 경로와 작업환경을 이루는 구성요소로 이루어져 있다. 내부의 구성요소는 <표 1>과 같다.

<표 1> Configuration 내부 구성 요소

Node Name	Description
Background	Client(applet)의 배경 그림
Content	실행 영역(터미널 영역)
ImgButton	이미지 버튼
Button	텍스트 버튼
Edit	편집 영역
Filter	사용 금지 명령어 지정
Check	애플릿 체크를 위한 체크박스
Description	실습 언어에 대한 정보(버전, 위치)

이미지/텍스트 버튼은 버튼 클릭시 수행하는 일을 action 속성에 작성한다. Action에 따른 수행 작업은 <표 2>와 같다.

<표 2> Action에 따른 수행 작업

Action Name	Description
INIT	초기 화면 정보 요청
EXEC	인터프리터형 언어의 실습 환경과 실행 요청
REXEC	외부 서버의 주소, 포트번호 요청
INTE	컴파일형 언어의 실습 환경 요청
DESC	실습 언어들의 정보
CMP	작성한 프로그램의 컴파일 요청
RUN	작성한 프로그램의 실행 요청
Applet	작성한 applet 실행 요청
EDITBOX	편집 윈도우 요청
LINK	새로운 웹 페이지를 요청

아래는 제안된 시스템에 사용된 구성파일의 일부분이다.

```
<?xml version="1.0" encoding="EUC-KR"?>
<!DOCTYPE Configuration SYSTEM "Config.dtd">
<Configuration>
<Section name="Prolog4.00" exec="/usr1/BinProlog4.00/bin/bp.sparc.solaris">
  <BackGround coordinates="600,0" size="100,400" img="l_prolog.jpg" />
  <Content coordinates="0,0" size="600,400" />
```

```
  <ImgButton name="초기화면" coordinates="605,5" size="90,30"
    action="INIT" img1="first1.jpg" img2="first2.jpg" />
  .....
</Section>
<Section name="ML" exec="icom.changwon.ac.kr:7085"></Section>
.....
<Init name="INIT">
  <BackGround coordinates="0,0" size="700,400" img="logo.jpg" />
  <ImgButton name="Java" action="INTE" img1="f_java1.jpg" .... />
  <ImgButton name="Prolog4.00" action="EXEC" ..... />
  .....
  <Button name="ML" coordinates="120,300" size="90,20" action="REXEC" />
</Init>
</Configuration>
```

2.4. 시스템 자원 권한 승인

제안 시스템은 애플릿이 외부 서버와의 통신이 필요하다. 자바의 보안 모델상 애플릿은 애플릿을 가져온 서버와의 통신만 가능하기 때문에 애플릿에 네트워크 권한을 주는 부분이 필요하다[5]. Microsoft 사와 Netscape사의 signed tool을 사용해서 Signed Applet 을 이용할 수 있으나[6,7], 이 방법은 여러 단점이 있다. 첫째, 각각의 벤더에서 제공하는 서로 다른 툴을 사용해야 하고 둘째, 애플릿을 실행시키는 브라우저를 구분해주어야 한다는 점 셋째, 권한의 범위를 사용자가 알 수 없어서 어떠한 악위적인 행위를 할지 모른다는 점이 있다. 이러한 단점 때문에 일부의 권한을 사용자가 직접 확인할 수 있게끔 정책 파일을 이용하여 애플릿 실행시 필요한 권한만을 주도록 구현했다.

제안 시스템에서 JRE(Java Runtime Environment)의 보안 설정 파일인 java.security 파일에 추가된 내용은 아래와 같다.

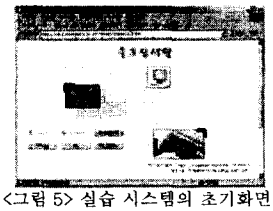
policy.url.3=http://cdcs.changwon.ac.kr/~mrroh/pulgrim2/Client/pulgrim.policy

클라이언트쪽의 사용자는 위의 주소로 접근하여 애플릿 실행시 부여되는 권한을 볼 수 있으며, 시스템 개발자는 서버의 정책 파일을 수정하여 프로그램 업그레이드시 필요한 권한을 쉽게 추가 할 수 있다. 아래는 pulgrim.policy의 내용으로 cdcs.changwon.ac.kr/~mrroh/pulgrim/Client/ 아래의 모든 코드에 icom 서버의 7085 포트로 연결할 수 있는 소켓권한을 부여하는 부분이다.

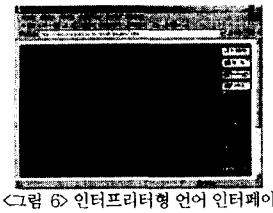
```
grant codeBase
  "http://cdcs.changwon.ac.kr/~mrroh/pulgrim/Client/*" {
      permission java.net.SocketPermission
                "icom.changwon.ac.kr:7085", "connect";
};
```

3. 구현 및 타 시스템과의 비교

제안된 시스템 구조를 바탕으로 구현한 프로그래밍 언어 실습 시스템은 Java 1.2.x를 기준으로 구현하였으며, 이 시스템을 이용하기 위해 학습자는 웹 브라우저와 JRE가 필요하고, 2.4절에서 언급했던 보안 설정파일을 수정해야 한다. <그림 5>는 실습 시스템을 수행하였을 때의 초기화면이다. 왼쪽 아래에 언어 선택을 위한 버튼이 보인다.



<그림 5> 실습 시스템의 초기화면



<그림 6> 인터프리터형 언어 인터페이스

<그림 6>은 인터프리터형 언어를 실습할 때의 작업 환경 화면이다. 인터프리터와 자료를 주고받기 위해 화면 왼쪽의 실행창을 제공하며, 이 창에서 직접 입력하면 실행된 결과가 출력된다. 컴파일 형태의 프로그래밍 언어를 작성하고 실행하기 위해서는 편집 창과 실행 창을 각각 제공하였다. 편집 창은 프로그램을 작성하고 컴파일 할 수 있으며, 실행창은 인터프리터 형태의 프로그래밍 언어의 수행과 동일하다.

<표 3>은 Web 기반의 프로그래밍 언어 실습 시스템들과 제안 시스템과의 비교를 표로 나타낸 것이다.

<표 3> Web 기반의 실습 시스템들과 제안 시스템과의 비교

	제안 시스템	SEAL System[8]	자바가상 교육센터[9]	C언어 pre-processor 환경[10]
실습 언어	Java, ML Scheme, Prolog, etc.	ML, Java, C++,C	Java	C
실습 컨텐츠	없음	있음	있음	없음
분산 처리	O	X	X	X
언어 확장성	O	△	X	X

4. 결 론

본 연구에서는 여러 시스템에 존재하는 프로그래밍 언어들을 하나의 자바 애플릿을 이용하여 WWW 환경에서 실습할 수 있는 시스템을 구축하였다.

이 시스템의 가장 큰 특징은 웹을 통해서 다양한 프로그래밍 언어를 실습해 볼 수 있으며, 프로그래밍 언어 또한 애플릿이 위치한 서버에 의존하지 않고 서비스를 하는 서버의 주소와 포트 번호만으로 실습 언어의 확장을 용이하게 했다는 점이다. 그리고 시스템을 설정하는 구성파일은 구조화된 문서의 작성을 지원하는 XML을 이용하여 관리자가 쉽게 구성파일을 작성할 수 있게 했으며, 이 시스템을 관리하는 서버 관리자는 이 구성파일을 실습할 수 있는 프로그래밍 언어에 따라 적절히 바꾸어 주면 된다. 또한 자바의 정책 파일을 이용해 애플릿을 실행시키는 로컬 시스템의 자원 사용 허가를 사용자가 직접 확인할 수 있게 했다.

향후 연구로는 서비스하는 서버의 확장에 따른 서비스 정보의 공유나 실습 교육을 위한 컨텐츠 보완, 로컬에 있는 파일을 컴파일하거나 실행할 수 있는 인터페이스 등이 필요하다고 생각되며, 프로그래밍 언어의 실습만이 아니라 유닉스, 리눅스 명령어 실습까지 고려하고 있다. 더 나아가서는 텍스트 기반의 실습이 아닌 비쥬얼(visual) 환경의 실습 시스템에 대해 연구할 계획이다.

<참고문헌>

[1] 이수현, 배성훈, 김수근, 노미라, "프로그래밍 언어 학습을 위한 가상실습환경", 프로그래밍언어연구회 1999년 동계 워크샵 논문집, pp.33-38, 1999

[2] XML SGML 모임, "표준 XML", 대청, 2001

[3] Macro Pistoia, Duane f. Reller, Deepak Gupta, Milind Nagnur, Ashok K. Ramani, Java 2 Network Security, IBM Corp., 1999

[4] Qusay H. Mahmoud, Distributed Programming with JAVA, MANNING, 2000

[5] Sun, Security, http://java.sun.com/products/jdk/1.2/docs/guide/security/index.html

[6] Microsoft, Java Security Overview, http://www.microsoft.com/Java/security/

[7] Netscape, Object Signing Resources, http://developer.netscape.com/software/signedobj/

[8] Andreas Ausserhofer, "S.E.A.L. - A New Approach in Teaching Computing", IASTED International Conference Computers and Advanced Technology in Education CATE'99 May 6-8, 1999

[9] 이승하, 한동헌, 김양우, 유갑상, "웹 기반 자바 가상교육센터의 설계 및 구현", 2001년 한국정보과학회 봄 학술발표논문집(B) 제28권 1호, pp643-645, 2001

[10] 조정우, 김진석, "Web기반 Virtual OS에서의 C언어 preprocessor 환경 설계 및 구현", 2001년 한국정보과학회 봄 학술발표논문집(A) 제28권 1호, pp31-33, 2001