

라우팅프로토콜을 위한 웹기반 모델링, 시뮬레이션, 애니메이션

서현곤, 사공봉, 김기형
영남대학교 컴퓨터공학과

Abstract

웹기반 시뮬레이션은 인터넷과 웹을 통해 시뮬레이션 실험을 하기위해 개발되었다. 웹기반 시뮬레이션 언어는 자바언어를 사용하기 때문에 재사용성, 이식성, 웹에서의 실행성등의 특징을 가진다. 대부분의 웹기반 시뮬레이션 툴들은 주로 웹기반 시뮬레이션엔진 및 라이브러리의 개발에 중점을 맞추어 연구해 왔다. 따라서 이러한 툴들을 사용하여 모델을 개발하는 일은 여전히 모델개발자에게 전문성, 코딩능력등을 요구하게 된다. 본 논문에서는 웹기반 모델링 툴인 Simdraw를 소개하고 이를 이용하여 라우팅프로토콜을 시뮬레이션, 애니메이션하는 기능을 보인다. 모델 개발자는 이미 개발된 라이브러리를 사용하여 단순히 시각적 모델링 만으로 원하는 네트워크 토폴로지하에서의 라우팅 기법의 원리를 배우고 또한 성능을 평가해 볼 있다.

I 서론

기존의 시뮬레이션 환경은 시뮬레이션 언어를 기반으로 해서 시뮬레이션 환경을 구축하거나 라이브러리를 기반으로 시뮬레이션 환경을 구성하였다. 따라서 특정한 언어나 툴에 종속적이 될 수밖에 없으며, 이런 언어나 라이브러리로 개발된 시뮬레이션 모델들은 다른 시뮬레이션 다른 시뮬레이션 환경에서 서로 서로 같이 실행될 수 없으며, 다른 운영체제에서 사용하기 위해 재컴파일을 해야 하는 불편함을 가지고 있다. 또한 웹과 인터넷을 통한 실행 능력을 제공할 수가 없다.

웹 기반 시뮬레이션은 인터넷과 웹을 통해서 시뮬레이션을 수행하는 데 목적이 있다. 자바[1]와 같은 언어는 기존의 시뮬레이션 환경이 가진 단점들을 해결해 줄 수 대표적인 언어라고 할 수 있다. 즉 이기종 환경에서도 실행이 가능하고, 인터넷과 웹을 통한 실행 능력을 제공하고 있다.

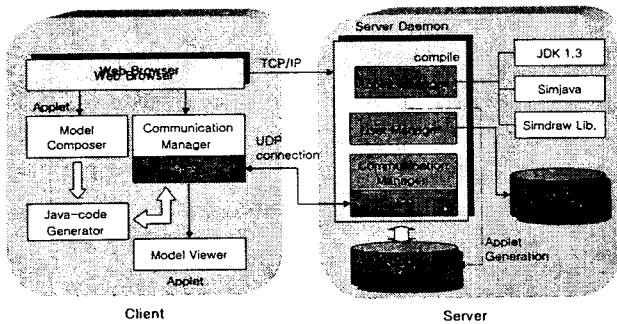
최근까지 자바로 개발된 웹기반 시뮬레이션 라이브러리를 보면 JSIM[2], Simjava[3], Silk[4], DEVS-JAVA[5], JavaSim[6] 등이 있다. 이러한 자바 라이브러리의 모델

개발 환경을 보면 다음 두 가지 형태의 개발 방법을 제공한다.

본 논문에서는 클라이언트/서버 구조의 웹 기반 시뮬레이션 제안하고 라우팅 프로토콜을 벤치마크로써 모델링, 시뮬레이션 그리고 애니메이션을 하게 된다. 클라이언트는 GUI 형태의 모델 개발환경을 제공해 주기 때문에 개발자의 코딩에 대한 부담을 덜어 주었고, 자바를 이용하여 웹 환경에서 시뮬레이션의 진행상황을 통해 효과적으로 시각화 할 수 있다. 서버는 클라이언트에게 여러 가지 다양한 모델 라이브러리를 제공하여 클라이언트가 여러 가지 모델 개발이 가능하도록 하였다.

II. SimDraw

Simdraw의 구조는 <Figure 1>과 같이 클라이언트와 서버로 구성된다. 클라이언트는 Model Composer, Java code Generator, Communication Manager, Model Viewer로 구성되며, 서버는 Applet Manager, Communication Manager, User manager와 컴파일을 위한 코드들로 구성되어 있다.



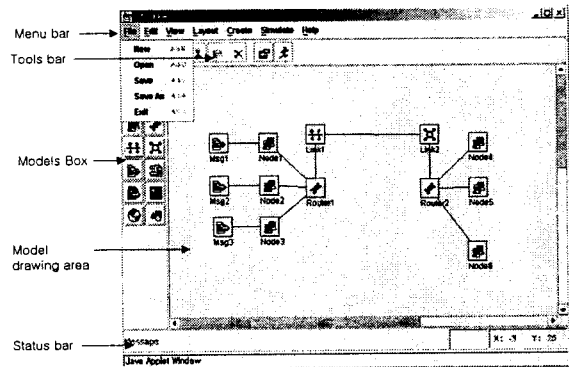
<Figure 1> SimDraw의 전체 구조

1. SimDraw 클라이언트

Simdraw의 클라이언트는 웹 브라우저를 통해 서버로부터 모델 개발에 필요한 컴포넌트와 모델 저작에 필요한 자료들을 다운 받아 쉽게 모델을 개발 할 수 있는 환경을 제공한다. 클라이언트를 구성하고 다음과 같다.

● 모델 작성기(Model Composer)

모델 작성기는 사용자가 모델 작성에 필요한 GUI와 여러 가지 기능들을 제공한다. 모델 작성기는 자바 애플릿으로 만들어져 웹 브라우저를 통해 클라이언트로 로딩된다. 모델 작성기에는 메뉴바, 툴바, 모델 박스, 모델 저작 영역, 상태 바(Status Bar)로 구성되어 있다. 메뉴바 및 툴바는 SimDraw의 환경 설정이나 모델 개발에 필요한 여러 가지 기능들이 제공하며, 모델 박스에는 SimDraw 클라이언트가 모델 저작을 위해 필요한 컴포넌트들이 등록되어 있다. 모델을 저작할 경우에 사용자는 필요한 컴포넌트를 선택한 후 모델 저작영역에 클릭하면 컴포넌트가 추가되며, 컴포넌트를 더블 클릭하면 각 컴포넌트의 속성을 수정할 수 있는 다이얼로그 박스가 화면에 나타나게 된다. 사용자는 이를 이용해 각 컴포넌트들의 속성 값을 설정할 수 있다. <Figure 2>는 SimDraw 클라이언트의 모델 작성기이다.



<Figure 2> SimDraw 클라이언트의 모델 작성기

● 자바 코드 생성자(Java code Generator)

클라이언트의 모델 작성자가 모델 작성기로 작성한 모델은 자바 코드 생성자에 의하여 시물레이션 코드로 번역된다. 시물레이션 코드는 Simjava의 Anim_applet 클래스로부터 상속받은 Runnable 인터페이스의 애플릿 코드로서 anim_init(), anim_layout() 메소드로 구성되어 있다. anim_init() 메소드는 애플릿의 GUI를 설정하는데 필요한 메소드이고 anim_layout() 메소드는 시물레이션 엔티티를 추가하고 엔티티 간의 연결 정보를 추가하는 메소드이다.

● 통신 관리자(Communication Manager)

통신관리자는 클라이언트와 서버간의 메시지 전달 및 컴포넌트 정보 전송, 그리고 모델정보와 시물레이션 코드의 전송 등을 담당한다. <표 1>은 통신 메시지와 기능을 나타낸 것이다.

메세지	기능
Req_auth	사용자 인증을 요청하는 메시지
ObjectWrite	사용자가 작성한 모델을 서버에 저장할 때 사용하는 메시지
ObjectRead	사용자가 작성한 모델을 서버로부터 읽어 올 때 사용하는 메시지
Put	시물레이션 코드를 서버에 저장하기 위한 요청 메시지
Get	시물레이션 코드를 서버로부터 읽어 오기 위한 요청 메시지
Req_to_Run	시물레이션 실행 메시지

<표 1> 클라이언트/서버간의 전달 메시지

- 모델 뷰어(Model Viewer)

모델 뷰어는 모델 작성자의 웹 브라우저를 통해 전달되는 애플릿이다. 모델 작성자는 모델 뷰어를 통해 시뮬레이션 되는 과정을 볼 수 있다.

2. SimDraw 서버

Simdraw의 서버는 컴포넌트 정보와 사용자를 관리하기 위한 기능, 클라이언트에 의해 생성된 시뮬레이션 코드 번역하는 기능 및 클라이언트와 정보를 교환하기 위한 통신 기능 등을 가진다. 서버의 구성은 다음과 같다.

- 사용자 관리자(User Manager)

사용자 관리자는 각 사용자에 대한 사용자 인증을 처리하는 기능을 수행한다. 인증된 사용자에게는 개발한 모델과 생성된 자바 코드 및 애플릿 코드를 저장할 수 있는 디렉토리 영역을 할당하게 된다.

- 애플릿 관리자(Applet Manager)

애플릿 관리자는 클라이언트에서 생성한 자바 시뮬레이션 코드를 컴파일하고 애플릿을 실행시키기 위한 HTML파일을 생성한다.

- 통신 관리자(Communication Manager)

통신 관리자는 클라이언트와 통신을 담당한다. 특히 클라이언트는 애플릿으로 구현되었기 때문에 애플릿이 실행되고 있는 로컬 시스템의 저장 매체에 접근 할 수 없기 때문에 Simdraw서버에 파일을 읽고 쓸 수 있는 기능이 필요하게 된다. 그래서 본 논문에서는 Webmin에서 개발한 JFS Ver 0.05을 사용하여 서버와 클라이언트 사이의 메시지와 데이터를 전달할 수 있도록 하였다.

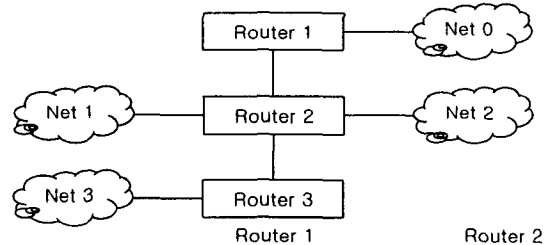
III. 라우팅 프로토콜을 위한 모델 라이브러리의 개발

이 장에서는 Simdraw를 이용하여 라우팅 프로토콜의 모델링, 시뮬레이션, 애니메이션을 하기 위한 컴포넌트를 설계하겠다. 라우팅(Routing)이란 IP 패킷을 목적지까지 전달하는 과정을 의미하며 라우팅 프로토콜은 라우팅 정보(Routing Information)를 주고받기 위한 프로토콜을 의미한다. 라우팅 정보를 전달하기 위한 라우팅 알고리즘에는 RIP(Routing Information Protocol), IGRP(Interior Gateway Routing Protocol),

EIGRP(Enhanced Interior Gateway Routing Protocol), OSPF(Open Shortest Path First)등이 있지만 본 논문에서는 가장 많이 사용되고 있는 RIP 이용한 라우팅 프로토콜의 모델 라이브러리를 설계하고 구현하였다. 본 논문에서 라우터는 RIP 알고리즘을 이용하여 라우팅 정보를 교환하여 라우팅 테이블에 최근의 라우팅 정보를 유지한다.

1 RIP 알고리즘

RIP 알고리즘은 Distance Vector Routing Protocol의 대표적인 알고리즘이다. 각 라우터의 라우팅 테이블은 데이터그램 패킷을 통하여 인접한 라우터에게 방송(advertising)된다. RIP에서는 매 30초마다 라우팅 정보를 방송해야 하면 최대 Hop을 16으로 제한하고 있다. 만약 180초 이내에 새로운 라우팅 정보가 수신되지 않으면 해당 경로를 이상상태로 간주한다. 메트릭(Metric factor)란 라우터가 경로를 결정할 때 영향을 요소로써 회선속도(Bandwidth), 전송지연(Delay), 회선신뢰도(Reliability), 회선 부하(Load), 경로 수(Hop count), 최대 전송단위(MTU : Maxmum Transmission Unit)등이 있다. 경로 수(Hop Count)란 IP 패킷이 도착지까지 경유해야하는 라우터의 수를 의미한다. RIP의 Metric factor는 단지 경로 수(Hop count)이다.



<Figure 3> Network의 예

<Figure 3>과 같이 네트워크가 구성되어 있을 때 각 라우터의 초기 라우팅 테이블은 <Figure 4>와 같다. 즉 각 라우터는 자신에게 연결된 네트워크만 알고 나머지 네트워크에 대한 정보를 알 수가 없다.

Network	Metric
Net0	0

Network	Metric
Net1	0
Net2	0

Network	Metric
Net3	0

<Figure 4> 초기 라우팅 테이블

각 라우터의 메트릭 값은 경로 수(Hop count) 값을 의미한다. 즉 라우터1에서 Net0의 메트릭 값은 0이다. 라우터가 RIP 알고리즘을 이용하여 라우팅 정보를 교환한 후의 라우팅 테이블의 상태는 <Figure 5>와 같다.

Network	Metric
Net0	0
Net1	1
Net2	1
Net3	2

Network	Metric
Net1	0
Net2	0
Net0	1
Net3	1

Network	Metric
Net3	0
Net0	2
Net1	1
Net2	1

<Figure 5> 라우팅 정보 교환후의 라우팅 테이블

2 컴포넌트

Simdraw에서 RIP 라우팅 프로토콜의 시뮬레이션 및 애니메이션을 실행하기 위해서 4개의 컴포넌트를 설계하였다. 각 컴포넌트는 하나의 라이브러리 클래스인데 이들 컴포넌트는 Simjava의 Sim_entity 클래스에서 상속받아 생성되며 멀티 쓰레드로 동작된다. 네트워크의 구성에 따라 각 컴포넌트 클래스로부터 필요한 컴포넌트 객체를 생성하여 시뮬레이션을 하게 된다.

라우터를 표현하기 위한 ROUTER 컴포넌트, 서브 네트워크를 표현하기 위한 LAN 컴포넌트, IP 패킷을 송수신하는 사이트를 표현하기 위한 NODE 컴포넌트, IP 패킷 생성 신호를 발생시키기 위한 GEN 컴포넌트가 있다.<Figure 6>은 라우터 컴포넌트의 라이브러리의 일부분이다.

```
class ROUTER extends Sim_entity{
    . . . . .
    public ROUTER(String name, int r_portcnt,
        int t_site, int t_router, int t_lan, int ploc[], int
        x, int y){
        . . . . .
    }
    public void body(){
        . . . . .
    }
}
```

<Figure 6> ROUTER 클래스

• Router컴포넌트

라우터 객체는 RIP 라우팅 프로토콜을 실행하는 객체로서 자신에게 연결된 LAN객체의 정보와 인접한 라우터 객체의 라우팅 정보를 얻기 위하여 객체 상태 정보를 요구한다. 이들에게 전송된 상태정보를 이용하여 포트 상태 테이블과 초기 라우팅 테이블을 생성한다. <Figure 7>은 라우터 객체의 실행과정 다음과 같다.

- [Step1] 라우터 객체 초기(입출력 포트 생성 및 포트 위치 결정)
- [Step2] 포트 상태 테이블 및 초기 라우팅 테이블 생성 (연결된 LAN객체에게 상태 정보를 수신)
- [Step3] 인접한 라우터 객체에게 자신의 초기 라우팅 정보를 전송
- [Step4]
 - ① 인접한 라우터에서 전송되어 오는 라우팅 정보를 수신
 - ② 수신한 IP패킷을 포워딩
 - ③ 일정한 시간 간격으로 새롭게 갱신된 라우팅 정보를 인접한 라우터에게 방송 (갱신된 라우팅 정보가 없는 경우 "alive"신호를 전송)
 - ④ Step4 반복

<Figure 8>은 라우터 객체에 의해 생성된 라우팅 테이블

블에 대한 예제이다.

Port	Metric	Network ip	interface	LAN 객체
0	0	221.13.120.0	CSMA/CD	LAN-3
1	1	150.200.30.0	Serial	LAN-2
1	1	198.200.30.0	Serial	LAN-1
1	2	210.100.50.0	Serial	LAN-4

<Figure 8> 라우팅 객체에 의해 생성된 라우팅 테이블

• LAN컴포넌트

LAN객체는 자신의 포트에 연결된 모든 노드 객체 및 라우터 객체에 대한 정보와 IP주소를 포트 상태 테이블에 저장한다. LAN객체는 CSMA/CD와 Token Ring 2가지 인터페이스를 제공한다. LAN객체의 실행 과정은 <Figure 9>와 같다.

[Step1] LAN 객체 초기(입출력 포트 생성 및 포트 위치 결정)
[Step2] 자신의 정보(IP주소, interface)를 인접한 라우터객체에게 전달
[Step3] 포트 상태 테이블을 생성(자신에게 연결된 노드 객체에 대한 정보를 유지)
[Step4] IP패킷의 포워딩(목적지 IP 클래스가 자신의 IP클래스와 동일하면 해당 노드 객체에게 IP패킷을 전송하고 아니면 라우터 객체에게 포워딩 한다)

<Figure 9> LAN 객체의 실행 과정

<Figure 10>은 LAN객체에 의해 생성된 포트 상태 테이블이다.

Port	Site name	Site id	Site IP
0	Node0	8	221.200.30.12
1	Node1	9	221.200.30.71
2	Node2	13	221.200.30.120
3	Router1	7	221.200.20.2

<Figure 10> LAN 객체의 상태 테이블

• Node컴포넌트

노드 컴포넌트는 항상 2개의 포트를 가지는데 하나는 LAN객체와 연결되어 있고 다른 하나는 GEN객체에 연결되어 있다. GEN객체에서 IP 패킷을 생성하라는 신호가 수신되면 IP 데이터그램을 생성하여 LAN객체에게 전달한다. 노드 객체는 각 노드 객체에 대하여 송수신한 정보를 유지하기 위해 패킷 송수신 테이블을 생성한다. <Figure 11>은 패킷 송수신 테이블의 예제이다.

Node id	Node ip	Send Msg	Receive Msg
0	198.12.17.3	8	3
1	198.12.17.39	9	12
2	210.100.50.126	13	4
n	221.13.120.2	7	21

<Figure 11> Node 객체의 패킷 송수신 테이블

• GEN 컴포넌트

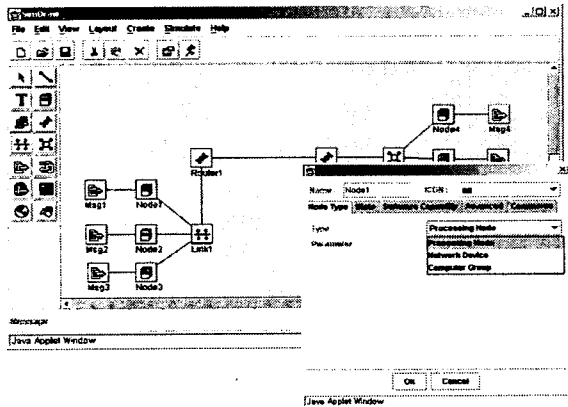
GEN객체는 노드객체에게 IP 패킷을 생성하라는 신호를 생성한다. 신호는 정규분포, 지수분포, 포아송 분포에 따라 생성되며 노드 객체와 연결된 하나의 포트만을 가지고 있다.

IV. 라우팅 프로토콜의 모델링과 시뮬레이션

본 장에서는 SimDraw를 이용하여 사용자가 모델 작성기를 이용하여 모델을 저작하고 저작한 모델에 대해 시뮬레이션 수행과정에 대해서 설명하겠다.

1. SimDraw클라이언트를 이용한 모델링

웹 브라우저를 이용해 SimDraw서버에 접속하게 되면 서버는 사용자 인증을 거치고 난 후, 자바 애플릿으로 구현된 모델 작성기가 로딩 된다. 사용자는 모델 작성기를 이용하여 만들고자 하는 모델을 저작하면 된다.

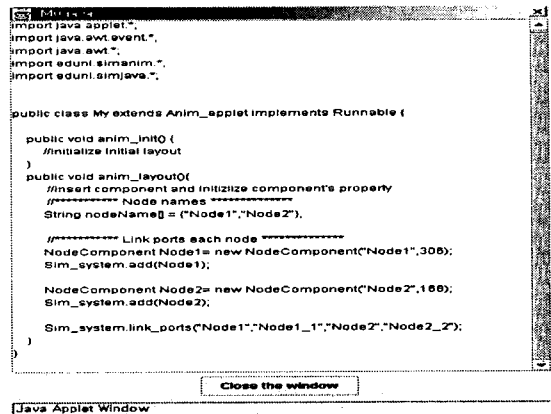


<Figure 12> SimDraw를 이용한 모델 작성의 예

<Figure 12>는 모델 작성기를 이용하여 작성한 모델의 예이다. 그림에서 오른쪽 아래에 있는 속성 다이얼로그는 각 컴포넌트들의 속성 값을 수정하는 기능을 가진다.

2. 시뮬레이션 코드 생성

모델 작성기로부터 작성된 모델 정보를 이용하여 사용자가 시뮬레이션하기 전에 가장 먼저 하는 일을 시뮬레이션 코드를 생성하는 것이다. 이때 생성되는 코드는 자바 애플릿 코드로써 사용자가 작성한 모델의 각 컴포넌트의 고유 정보를 가지고 있다. <Figure 13>은 사용자가 작성한 모델로부터 생성된 시뮬레이션 코드이다. 이렇게 생성된 코드는 서버의 사용자 디렉터리로 저장되게 된다.



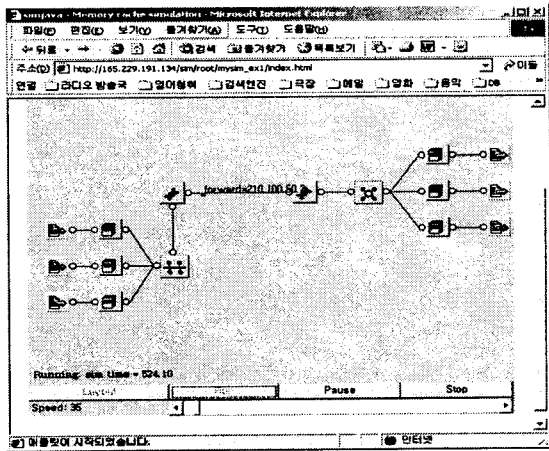
<Figure 13> 시뮬레이션 코드 생성

3. 컴파일, Html 코드 생성

서버는 클라이언트로부터 받은 시뮬레이션 코드를 SimDraw 라이브러리와 함께 컴파일 한 후 자바 바이트 코드를 생성하게 되며, 애플릿을 로딩하기 위한 Html 파일을 함께 생성하게 된다. 이렇게 생성된 코드는 모델 작성자의 웹 브라우저를 통해 자바 애플릿으로 시뮬레이션 되어 진다.

4. 모델 실행과 애니메이션

컴파일과 Html 코드 생성이 끝이 나면 서버는 사용자에게 시뮬레이션을 하고자 하는 모델에 대한 URL정보를 전달하게 되며, 모델 작성자의 웹 브라우저를 통해 작성된 모델의 시뮬레이션 결과를 확인할 수 있게 된다. 그림은 자바 애니메이션을 이용한 시뮬레이션 실행의 예이다.



<Figure 14> 시뮬레이션 실행

V. 결론

본 논문에서는 클라이언트/서버 구조의 웹 기반 시뮬레이션 툴을 제안하고 라우팅프로토콜을 벤치마크로서 모델링, 시뮬레이션 그리고 애니메이션 하였다. 제안된 웹 기반시뮬레이션 툴은 그 자체가 자바애플릿으로서 사용자는 시뮬레이션을 위해 단순히 웹브라우저만 있으면 된다. 이러한 간편함 때문에 원격교육에서의 가상 실험실 등으로 사용될 수 있다. 제안된 웹 기반 시뮬레이션 툴에서의 모델링에는 자바언어를 숙지할 필요도 없고 단순히 시각적으로 패라미터들을 셋팅함으로써 이루어지며 개발된 모델은 서버로 전송되어 컴파일 되고 바이트 코드로 변환된 뒤에 자바 애플릿으로서 사용자에게 반환된다. 사용자는 개발된 모델을 서버에 저장할 수 있으며 언제든지 다시 불러들여 변환, 추가 등을 할 수 있다.

V. 참고 문헌

- [1]Sun Microsystems, Using Swing Components, Java Tutorial, <http://web2.java.sun.com/docs/books/tutorial/>
- [2]John A. Miller, Andrew F. Seila and Xuewei Xiang, "The JSIM Web-Based Simulation Environment," 1999
- [3]Howell, F.W, The simjava home page, <http://www.dcs.ed.ac.uk/home/hase/simjava>, April, 1999

[4]Healy, K.J and R.A. Kilgore, "Introduction to Silk and Java-based Simulation", Proc. of the 1998 Winter Simulation Conference, 1998

[5]Zeigler, B.P "DEVS-JAVA modelling and simulation environment" <http://www-ais.ece.arizona.edu/SOFTWARE/software.html>

[6]University of Newcastle upon Tyne, JavaSim homepage, <http://javasim.ncl.ac.uk/>, April, 1999

[7] 남영환, 김기형, 웹기반 시뮬레이션을 위한 시각적 모델 개발 환경, 한국 시뮬레이션 학회, 8권 1호, 1999

[8] Robert Wright, "IP Routing Primer", CISCO Press, 1998.

[9] Jeff Doyle, "CCIE Professional Development : Routing TCP/IP", Vol I, CISCO Press, 1998.