論文
8-1-6

Java를 이용한 웹 기반 원격 감시제어시스템 개발 (A Development of Web-based Remote monitoring and control system using Java)

박 종 진(Jong-Jin Park)[1)]

요약

본 논문은 Java 기술을 이용하여 TCP/IP 상에서 웹 기반의 원격 감시제어시스템 개발에 대한 예를 제시하였다. Java의 Socket 클래스를 이용한 클라이언트/서버 소켓 프로그램을 구현하였고 이를 레고 블록으로 만든 온실 모델에 적용하여 웹 기반 온실 감시제어시스템을 구축하였다. 구축된 웹 기반 온실 감시제어시스템은 온실의 정보를 클라이언트 프로그램에 잘 전달하여 표시하며 웹 상에서 동작하는 Java Applet 클라이언트에서 보내는 제어 신호를 서버를 통해 온실 모델에 잘 전달하여 동작시키는 것을 볼 수 있었다.

Abstract

In this paper, a example of a development of web-based remote monitoring and control system using Java technology on TCP/IP is proposed. We implemented Client/Server socket programs using Java Socket class, and applied those to a greenhouse made from LEGO blocks and made web-based greenhouse remote monitoring and control system. Implemented System transferred states of greenhouse to client programs, and control signals from client program were transferred to Server. Greenhouse model was controlled well by the system.

논문접수 : 2007. 1. 15.
심사완료 : 2007. 2. 11.

1) 정회원 : 청운대학교 인터넷학과 부교수

* 본 논문은 2004년도 청운대학교 학술지원조성비에 의해 연구되었음.

1. 서론

최근 수십 년 동안 정보통신기술(IT : Information Technology)이 빠르게 발전하였고, 이러한 정보통신기술의 확산으로 산업사회에서 지식기반 정보화 사회로 변화되고 성숙되는 단계에 이르고 있다. 이러한 변화의 중심에는 인터넷이 자리 잡고 있으며 이제 인터넷이 없이는 거의 아무 것도 할 수 없는 사회가 되고 있다.

인터넷은 전국적 국가 기간망으로 성장하여 초고속 인터넷이 가정과 직장 등 거의 모든 곳에 연결되고 있다. 이러한 인터넷을 이용하면 인터넷이 연결된 곳에서는 어느 곳에서나 웹 브라우저나 GUI 응용 프로그램을 통해 접근 가능하므로 다양한 서비스나 시스템을 구현할 수 있다. 최근에는 이러한 인터넷을 이용한 원격 감시제어기술이 개발되어 다양한 시스템이 개발되고 있다. 환경의 다변화와 사회의 발전으로 인해 화재나 방범, 보안 등의 필요가 증가하여 인터넷 기반 무인 감시시스템이나 생산관리 및 효율성을 높이기 위한 공장자동화 시스템, 인터넷을 활용한 웹기반 원격 교육시스템, 유치원 등의 원격 보육시스템 등의 예가 있다. 인터넷에 기반한 이러한 원격시스템은 기존에 설치된 인터넷 네트워크 망을 이용하므로 비용부담이 적을 뿐만 아니라 사람의 개입을 최소화하며 효율성을 증가시킬 수 있다.

본 연구에서는 인터넷 기반 원격 감시제어시스템을 개발하기 위해 클라이언트-서버 방식의 인터넷 프로토콜을 이용하여, java applet으로 클라이언트(client)를 작성하고 응용 프로그램(application program)에 네트웍 서버를 추가하여, 둘 사이에 TCP/IP 프로토콜로 통신하는 방식을 사용하였다. 이를 위해 통신 장비(device)를 직접 제어하는 디바이스 드라이버 프로그램, TCP와 같은 트랜스포트 계층의 인터페이스를 이용하는 소켓 프로그래밍, 그리고 응용 계층을 이용하는 응용 계층 프로그램을 구현하였다. 개발된 시스템을 레고 블록을 이용한 온실의 온도제어에 적용하여 그 동작을 확인하였다.

2. Java를 이용한 원격 감시제어시스템

2.1 Java와 Java Applet을 이용한 웹 기반 감시제어시스템

Java는 선마이크로시스템즈에서 개발된 언어로 코드의 이동성과 플랫폼 독립적인 특징으로 인터넷의 발전과 함께 빠르게 전파되어 사용되는 언어이다. Java run-time 환경은 각각의 플랫폼에 구축된 가상 머신(Virtual Machine)에서 실제 프로세서에서 일어나는 것과 같은 일반적인 동작을 수행하며 네트워크와 관련된 다양한 API(Application Programming Interface)를 제공함으로써 인터넷을 통한 감시제어시스템 개발에 사용하기에 적합하다.

웹 상에서 Java로 작성된 응용 프로그램(Application)이 동작하기 위해서는 Applet 형식으로 프로그램을 작성해야 한다. Java Applet은 작고, 내장가능한 애플리케이션의 의미로 태그를 이용하여 HTML 페이지 안에 포함할 수 있으며, 포함된 Applet 바이트코드(byte-code)를 Java 가상 머신이 동작시킨다.

2.2 Java를 이용한 소켓 프로그래밍

Java를 이용한 원격제어를 위해서는 소켓 통신을 이용하는 방법과 CORBA를 이용하는 방법이 있다. 본 연구에서는 소켓 통신을 이용하여 원격감시제어에 필요한 프로그램 작성방법을 사용하였다. 네트워크에서 데이터가 전송되기 위해서는 패킷(Packet)이라는 단위로 쪼개져서 데이터가 전송되고, 상대측에서 이러한 패킷을 합쳐서 데이터를 복원한다. 이러한 동작을 해주는 것이 바로 TCP/IP 프로토콜이다. 소켓은 TCP/IP 네트워크 통로의 끝처럼 생각할 수 있다. 따라서 소켓을 이용하면 하위의 복잡한 프

로토콜과 상관없이, 유연하게 네트워크 프로그램을 작성할 수 있다. 이것은 일단 소켓을 이용하여 상대측과 접속되면, 마치 파일 시스템이나 다른 I/O처럼 사용이 가능하기 때문에, 네트워크와 관련된 복잡한 기능들에 대해서 신경 쓸 필요가 없기 때문이다. 현재 소켓은 TCP/IP 프로토콜을 사용하는 네트워크 프로그램을 작성하는 표준으로 사용되고 있다. 자바에서 소켓을 이용하여 데이터를 전송하기 위해서는 Java API의 java.net.Socket 클래스를 이용한다.

2.2.1 클라이언트/서버(Client/Server) 모델

소켓 프로그래밍을 하기 위해서는 기본적으로 작성하고자 하는 프로세스가 서버 역할을 할 것인지, 클라이언트 역할을 할 것인지를 결정해주어야 하는데 이렇게 클라이언트와 서버로 나누는 이유는 클라이언트 역할을 할 때와 서버 역할을 할 때 Socket 클래스를 사용하는 방법에 차이가 있기 때문이다. 소켓 클래스를 클라이언트에서 사용하기 위해서는 아래의 그림과 같은 3단계를 거친다.

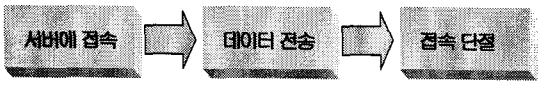

그림 1. 클라이언트에서의 소켓 사용법

1) 서버에 접속
Socket 클래스를 이용하여, 서버에 접속하기 위해서 호스트의 주소와 포트 번호를 소켓 클래스의 생성자에 넘겨줌으로써 원격호스트의 서버 프로세스에 접속하게 된다. 주의할 것은 Socket 클래스 객체는 한번 생성된 후에 접속된 원격호스트를 바꿀 수 없다는 점이다. 즉, 새로운 호스트와 연결하기 위해서는 반드시, 새로운 Socket 클래스 객체를 생성해야 한다.

2) 데이터 전송
일단 Socket 객체가 생성되면, 입출력 스트림과 연결할 수 있다. 소켓 클래스 객체의 getInputStream(), getOutputStream()메소드를 이용하면, InputStream과 OutputStream을 구할 수 있다. 일단 스트림을 구하면, BufferedReader등의 필터 스트림을 사용하면 편리하게 프로그래밍 할 수 있다. 이것은 네트워크의 특성상 데이터의 전송 속도가 일반 입출력에 비해서 느리고, 데이터가 중간에 손실되는 경우도 많기 때문이다. 일반적으로 소켓 클래스객체에 대해서는 읽기/쓰기가 모두 가능하다. 그러나 스트림의 특성상 단방향밖에는 지원하지 않기 때문에, 일반적으로 소켓 클래스 객체에 출력 스트림과 입력스트림을 하나씩 연결하게 된다. 또한 소켓에서 얻어진 OutpuStream과 InputStream은 서로에게 영향을 미치지 않고 접근이 가능하다. 즉 읽고 쓰는 것이 서로에게 독립적이다.

3) 접속단절
데이터의 전송이 완료되었으면, 원격호스트에 접속을 단절한다.

2.2.2 서버 소켓

서버는 클라이언트가 세션을 요청하면, 이에 대한 응답을 하는 프로세스로서 서버도 클라이언트와 마찬가지로 소켓 클래스를 이용해야 네트워크를 이용할 수 있다. 그러나 서버에서는 클라이언트와 달리 상대측의 접속을 대기하고 있어야 한다. 자바에서는 이러한 기능을 가진 클래스가 제공되는데, 이를 Server Socket 클래스라고 한다. 아래의 그림에서처럼 Server Socket 클래스는 특정 포트에서 대기하고 있다가, 접속 요청이 있으면 이를 새로운 소켓으로 연결하여 처리해준다.

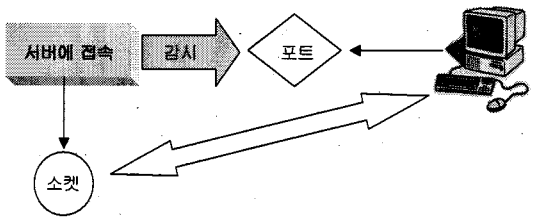

그림 2. 서버 소켓의 구조

보통 서버에는 하나 이상의 클라이언트가 접속하기 때문에, 서버는 여러 개의 접속을 처리할 수 있도록 프로그램 되어야 한다. 따라서 서버 프로그램들은 대부분 쓰레드를 이용하여 작성된다.

1) Server Socket 클래스의 생성
서버 소켓을 이용하기 위해서 Server Socket 클래스를 이용하여 객체를 생성한다. Server Socket 클래스객체는 생성 시에 감시할 포트 번호를 넘겨준다. 이 포트 번호는 서버에서 계속해서 감시하는 포트의 번호이다.

2) Server Socket 클래스의 접속 처리
서버 소켓 자체는 클라이언트에서 접속 요구가 들어왔는지 감시하는 역할만을 담당할 뿐 데이터를 읽거나 쓸 수 없다. 따라서 클라이언트에서 일단 접속 요구가 있으면, 이를 소켓 객체에 다 할당하고, 할당된 소켓 객체가 실제 접속을 담당하게 되는 구조를 갖는다. 이를 위해서, 서버 소켓의 accept() 메소드가 사용된다. accept() 메소드는 호출된 뒤, 접속 요청이 들어올 때까지 지연하고 있다가, 요청이 들어오면, 요구한 클라이언트 측과 접속을 설정한 후, 해당 접속을 유지하는 Socket 클래스 객체를 넘겨준다.

3) Server Socket을 이용한 다중접속 서버 프로그램
한 번에 하나의 클라이언트만을 지원하는 것이 아니라 동시에 여러 클라이언트의 접속 요구를 지원할 수 있도록 하는 다중 접속 서버(Concurrent Server)를 만들기 위해서 서버 소켓과 accept()메서드에 의해 생성된 소켓간의 긴밀한 협조가 필요하다. 보통 다중 접속 서버를 만들기 위해서는 서버 소켓이 accept()메서드를 얻을 때, 해당 소켓을 쓰레드로 처리해주고, 서버 소켓은 계속해서 접속 감시만을 담당하면, 다중 접속 서버를 만들 수 있다.

3. Java를 이용한 웹 기반 온실 감시제어시스템

앞에서 TCP/IP 프로토콜에 기반한 Java 소켓 프로그래밍을 통해 웹 기반 원격 감시제어시스템을 구현할 수 있음을 설명하였다. 이를 이용하여 웹 기반의 온실 감시제어시스템을 구축하기 위해 클라이언트의 접속을 대기하는 서버(Server) 모듈과 온실 시스템을 제어하기위해서 접근하는 클라이언트(client) 모듈을 구현하였다.

3.1 웹 기반 온실 감시제어시스템 개요

온실 감시제어시스템을 구현하기 위해 레고 블록을 이용하여 온실의 모델을 만들고 이를 인터페이스 보드를 이용하여 컴퓨터 시스템과 연결하였다. 실제 온실 시스템이 갖추어야할 기능은 다음과 같이 생각할 수 있다.
- 온실 내의 온도를 측정할 수 있어야 한다.
- 온실 내의 광도를 측정할 수 있어야 한다.
- 온실 내의 온도가 올라가면 식히기 위해 창을 열수 있어야 한다.
- 온실 내의 온도를 빨리 식히기 위해 팬을 동작시켜 온도를 내린다.
- 온실 내의 광도가 떨어지면 램프를 동작 시켜 광도를 유지시켜 주어야 한다.

실제 온실 시스템이라면 위와 같은 기능들이 있어야 하겠지만, 우리는 그 중에서 광도 측정과 광도 유지용 램프 부분을 제외시키고 기타 부분에 대해서 레고 시스템을 구성하였다. 다

음은 사용할 기능들이 어떻게 작동되는 지를 나타내는 시나리오이다.

- 온실 내부의 온도가 지정한 온도 이상으로 올라가면 온도센서가 이를 감지한다.
- 내부의 온도를 내리기 위해 온실의 창문을 연다. (모터로 동작)
- 창문을 여는 정도는 창문 축에 장착된 각도 센서의 카운터 수로 결정한다.
- 창문을 연 후, 온실 내부의 온도를 내리기 위해 팬을 동작시킨다.
- 온도센서가 감지한 온도가 지정온도이하로 내려가면 팬의 동작을 멈춘다.
- 다시 창문을 닫기 위한 창문 제어 모터가 동작하고 이 기간동안 온실내의 광도를 보충해 주기 위한 램프도 동시에 동작한다.
- 창문이 닫히고 램프가 꺼지면서 지정된 작업을 마무리한다.

아래의 그림은 웹 기반 온실 제어 시스템의 전체 구성도 나타내며 클라이언트에서 명령을 보내면 네트워크(Network) 망을 통하여 서버 측에 전달되고 서버는 클라이언트에서 보내준 명령을 처리하고 이에 따라 온실을 제어 한다.

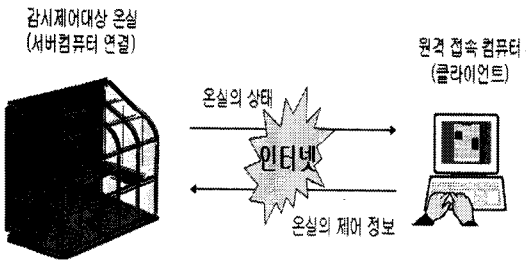

그림 3. 전체 구성도

온실 시스템을 조작하기 위해서는 온실 시스템의 기능정의에 따른 명령어 만들어야 한다. 이를 바탕으로 원격지에서 어떤 기능을 수행할 것인가를 정의해야 한다. 이것의 결과물로 신호표를 작성 한다. 신호표는 클라이언트/서버 간의 약속을 말한다. 사용될 신호표는 다음과 같다.

표 1. 신호표

신호	client -> server	설명
open	○	온실 창을 연다
close	○	온실 창을 닫는다.
manual	○	수동으로 온실 창을 관리
auto	○	자동으로 온실 창을 관리

이 신호표는 클라이언트가 보낸 메시지를 분석하고 서버는 이에 맞는 작업을 수행 한다.
다음 그림은 레고 블록으로 만든 온실의 모델을 나타낸다.

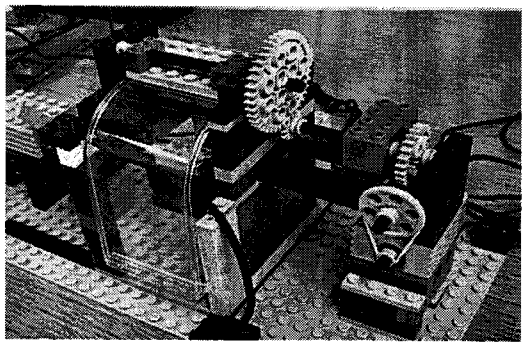

그림 18. 레고 블록 온실 모형도

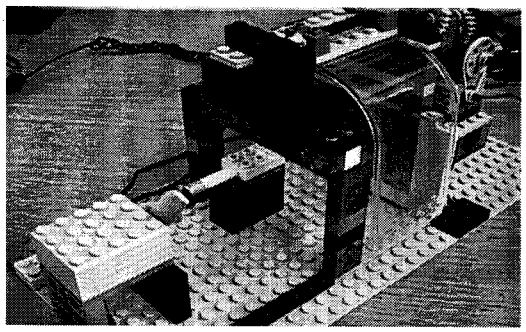

그림 19. 레고 블록 온실 모형도

3.2 클라이언트/서버 모듈 프로그램

앞에서 설명한 Java에서 제공하는 Socket API를 이용하여 클라이언트와 서버용 모듈을 작성하였다. 아래의 그림은 웹 기반 온실제어를 위한 전체적인 프로그램 도식도 이다.

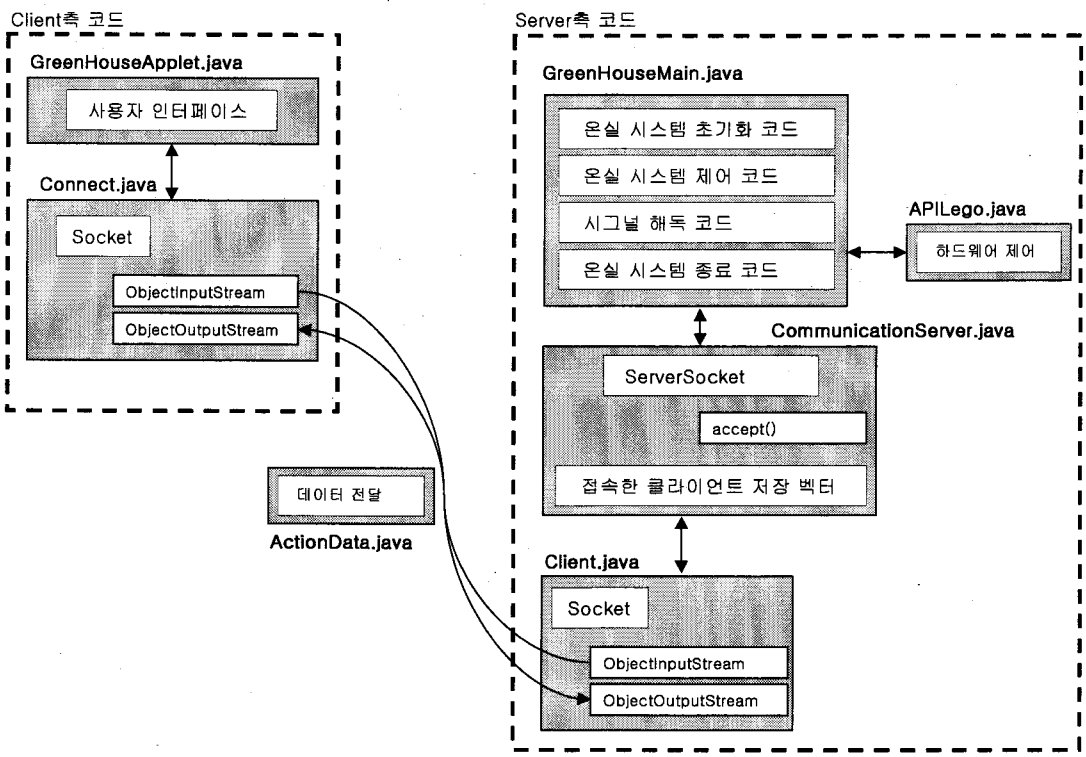

그림 6. 전체적인 프로그램 도식도

위의 그림을 보면 7개의 java파일로 구성되어 있으며 소켓통신을 통하여 객체를 주고받는 것을 볼 수 있을 것이다.

3.2.1 클라이언트 프로그램 구현

클라이언트 프로그램의 기능은 크게 둘로 나눌 수 있다. 사용자와 대화하는 사용자 인터페이스와 서버와의 통신을 담당하는 소켓프로그램으로 나눌 수 있다.

1) 사용자 인터페이스 프로그램(GreenHouseApplet.java)

이 코드는 웹 브라우저에서 실행하기 위해서 java Applet를 사용하여 만들었으며 위의 신호표를 참고하여 명령어 버튼들을 만들었다. 이것은 웹 브라우저의 플러그-인(Plug-in) 형태로 실행이 되며, 초기화시 사용자 인터페이스를 배치하고, 서버와의 통신을 위해 Connect객체를 생성 하며, 사용자와의 대화를 위하여 이벤트 처리 루틴을 만들었다.

2) 클라이언트 측 소켓 프로그램(Connect.java)
서버와의 소켓 통신을 위하여 필요하며 주요 기능은 사용자에 의해서 발생된 시그널을 서버측에 보내고, 서버측으로부터 보내어지는 온도센서, 온실창의 개폐 상태대한 값을 받아들인

다. 이때 서버와 문자 데이터로 통신하지 않고 객체로 통신 한다. 즉, 클라이언트의 제어 명령을 객체에 담아서 보낸다. 소켓 프로그램에서 객체를 사용하여 통신 할 경우 주의해야 될 점은 보낼 데이터가 있을 때마다 새로이 생성하여 보내야 한다는 것이다.

3.2.2 Server 구현

server의 기능은 크게 네 부분으로 나눌 수 있다.
① 클라이언트의 접속을 감시 및 클라이언트 접속 관리를 위하여 부분.
② 클라이언트와의 통신을 위한 부분.
③ 온실 시스템을 제어 및 전체 시스템을 관리하는 부분.
④ 하드웨어를 제어 하는 부분.
온실 시스템을 제어하는 부분과 하드웨어 제어하는 부분은 미리 작성된 프로그램을 사용하였고 원격지의 사용자와 통신을 위한 부분에 대한 설명은 다음과 같다.

1) 클라이언트 접속 감시 및 관리 프로그램(CommunicationServer.java)
클라이언트의 접속을 감시하는 코드는 서버에 특정 포트로 접속하는 클라이언트의 접속을 대기 하고 있다가 클라이언트의 접속이 이루어지고 나면 vector의 저장되어 있는 Client객체를 하나씩 읽어서 서버 측에 정보를 보낸다. Client의 객체는 서버에 접속하고 있는 사용자 수만큼 가지고 있다. 반드시 접속자 수와 vector의 크기는 일치해야 한다.

2) 클라이언트와의 통신 프로그램(client.java)
클라이언트와의 통신을 담당하는 코드로 앞에서 설명한 Connect.java 파일과 같은 구조를 가진다. 이 코드에서도 객체 통신을 위하여 ObjectInputStream, ObjectOutputStream 객체를 사용한다. 만약 서버 측에서 다른 형태의 데이터 타입으로 보낸다면 클라이언트의 서버 측에서 보내준 데이터를 받는 부분도 바뀌어야 된다. 즉 서버와 클라이언트간의 통신을 같은 데이터 포맷으로 주고받아야 된다.

3) 서버 실행하기
웹 서버로는 Apache 웹 서버를 사용하였다. 웹 서버를 실행시킨 후 서버 프로그램(GreenHouseMain.java)를 실행한다.

4) client 실행하기
클라이언트를 실행하기 위해서 앞에서 만든 자바 애플릿(GreenHouseApplet.java)을 포함하는 html파일을 만들어 실행한다. 다음 그림은 웹 브라우저를 통하여 서버에 접속한 후 온실감시 제어 프로그램을 실행한 결과 화면이다.

그림 7. 클라이언트 실행 결과 화면

4. 결론

본 논문은 Java와 Java Applet 기술을 이용하여 인터넷 프로토콜인 TCP/IP 상에서 웹 기반의 원격 감시제어시스템 개발에 대한 예를 제시하였다.
이를 위해 Java의 Socket 클래스를 이용한 클라이언트/서버 소켓 프로그램을 구현하였고 이를 이용하여 웹 기반의 온실 감시제어시스템을 구축하였다. 먼저 온실의 모델은 레고 블록을 이용하여 만들고 이를 컴퓨터와 연결하기 위해 임베디드 인터페이스 보드를 이용하였으며 클

라이언트의 접속을 대기하는 서버(Server) 모듈과 온실 시스템을 제어하기위해서 접근하는 클라이언트(client) 모듈을 구현하였다. 구축된 웹 기반 온실 감시제어시스템은 온실의 정보를 클라이언트 프로그램에 잘 전달하여 표시하며 클라이언트에서 보내는 제어 신호를 레고 블록으로 작성된 온실 모델에 잘 전달하여 작동시키는 것을 볼 수 있었다.

추후 연구 과제로는 컴퓨터 시스템에 구현된 서버 모듈을 소형화하여 하나의 마이크로프로세서에 탑재하여 동작하도록 내장화(Embedded)하는 것들이 필요하다.

참고문헌

[1] 박종진 외, "리눅스 기반 임베디드시스템", 대영사, 2004.

[2] J. Bentham, "TCP/IP LEAN Web Servers for Embedded Systems", CMPBooks, 2002.

[3] 전동환 외 역, "RTOS를 이용한 실시간 임베디드 시스템 디자인", 에이콘출판사, 2004.

[4] 최병욱 외, "임베디드 리눅스", 홍릉과학출판사, 2003.

저자 박종진은 1989년 연세대학교 전기공학과를 졸업하고, 동 대학원에서 공학석사 및 공학박사 학위를 1991년과 1997년에 각각 취득. 현재 청운대학교 인터넷학과에 교수로 재직하고 있으며 주요관심분야는 지능시스템, 내장형 시스템 및 인터넷 컴퓨팅이다.