# Socket API 기반의 네트워크 프로그램 소스 생성기

박승철, 최진식
한양대학교 교육대학원 컴퓨터교육과
{dani70}@empal.com, jinseek@hanyang.ac.kr

# A network programming source builder based on Socket API

Seungchul Park, Jin Seek Choi
Computer Education, The Graduate School of Education Hanyang University

**요 약**

컴퓨터와 하드웨어의 발달과 더불어 데이터 통신기술은 인터넷을 필두로 하여 괄목할만한 발전을 가져왔다. 아울러 FTP, Telnet, SMTP, HTTP, DHCP등의 네트워크 응용 프로토콜도 속도와 유저 인터페이스에 있어서 수많은 변화가 있었다. 하지만 이러한 외형적인 변화에도 불구하고 TCP, UDP 방식의 Socket 프로토콜은 네트워크 프로그래밍의 가장 기초적인 원리를 제시하고 있는데, 초보 학습자들에게 있어서 Socket을 구성하는 많은 시스템 함수들을 모두 이해하고 이것을 프로그래밍에 적용하기란 쉬운 일이 아니다. 본 연구에서는 Socket의 많은 함수들을 API기반의 모듈로 캡슐화 하여 C/C++ 기반의 네트워크 프로그래밍을 보다 손쉽고 효율적으로 수행할 수 있고, 또한 네트워크 프로그래밍의 동작 원리를 효과적으로 프로그래밍 학습에 적용할 수 있는 전산 교육시스템인 네트워크 프로그램 소스 생성기를 제안 한다.

## 1. 서 론

1980년대 이후 컴퓨터 네트워크란 말이 급격하게 발전해왔다. 컴퓨터 네트워크란 용어는 원래는 하드웨어적인 개념이다. 컴퓨터가 우리의 생활에 친숙해지기 이전의 일반인들에게는 생소한 개념일수 밖에 없었다. 그러나 1990년경부터 시작된 IT인프라의 발달, 그 중에서도 인터넷이라고 하는 글로벌 네트워크의 급속한 발전으로 인해 네트워크라는 개념이 더 이상 생소한 용어가 아닌 우리에게 친숙한 개념으로 다가서게 되었다.

하드웨어의 발달은 호스트 중심의 컴퓨팅 환경을 다운사이징 컴퓨팅 모델로 변화시키는가 싶더니 이제는 바야흐로 컴퓨팅의 흐름을 공간적 개념인 유비쿼터스(Ubiquitous) 환경으로 바꾸어 가고 있으며 그 중에서도 네트워크는 이러한 급속한 변화 패러다임의 중심에 자리 잡고 있다.

인터넷의 급속한 발전은 인터넷 환경을 구성하는 하드웨어의 발달과 함께 네트워크 응용 어플리케이션의 급속한 발전이라고 할 수 있다. 외국에 있는 친구들과 실시간으로 얼굴을 마주보며 채팅을 하거나 대용량의 파일들을 순식간에 주고받을 수 있고, 메일, 웹을 기반으로 하는 온라인 커뮤니티의 발달 등이 그것이다.

네트워크 응용 프로토콜 중에서도 우리에게 친숙한 것으로써 FTP, Telnet, SMTP, HTTP등이 있는데 이러한 프로토콜의 특징은 모두 프로세스간 통신을 기반으로 하고 있다. 또한, 이러한 프로토콜을 지원하는 네트워크 응용 어플리케이션을 작성하는데 Socket API는 최소한의 프로그래밍 인터페이스를 제공한다.

그러므로, 네트워크 프로그래밍을 개발함에 앞서 Socket API(Application Programming Interface)에 대한 이해가 가장 중요하다고 할 수 있다.

하지만, 기존의 Socket 프로그래밍 관련 교재들은 내용에 있어서 프로그래밍과 네트워크에 관한 전문적인 식견이 부족한 대다수의 초보 학습자들에게 수많은 Socket API에 대한 이해와 충분한 실습을 적용할 수 있는 환경을 제공하지 못하고 있다. 즉, 교재의 부속물로 제공되는 프로그램 소스를 복사와 붙여넣기 식의 단편적인 지식 전달 위주로만 편집되어 있을 뿐이며, 네트워크 프로그래밍의 본질적인 이해와 자기주도적인 응용 프로그램을 학습할 수 있는 쉽고 효과적인 프로그래밍 학습 도구로써의 기능을 제공하지 못한다.

본 연구에서는 이러한 문제점을 해결하고자 다음과 같은 시스템을 제안하고자 한다.

첫 번째로, 초보 학습자들이 쉽게 Socket의 API를 이해하고 TCP나 UDP방식의 기초적인 네트워크 프로그램을 작성할 수 있고, 두 번째로, 캡슐화된 API를 이용하여 FTP, Telnet 등의 네트워크 응용 어플리케이션을 구현할 수 있으며, 세 번째로, 사용자 정의의 프로그램을 작성할 수 있는 환경을 제공하여 자기주도적인 프로그래밍 학습을 구현할 수 있는 교육시스템으로 네트워크 프로그램 소스 생성기를 제안하고자 한다.

## 2. 이론적 배경

시스템의 프로세스간 통신 규약으로 인터넷에서 사용되는 TCP/IP만 존재하는 것은 아니다. 제록스사가 자신의 사무기기와 컴퓨터 시스템을 통합하기 위해 제안한 망 구조인 XNS(Xerox NS) 프로토콜, LU(Logical Unit)와 PU(Physical Unit)을 기본 구조로 갖는 IBM의 SNA(Systems Network Architecture) 프로토콜 그리고 DTE와 DCE간의 인터페이스를 규정한 X.25 등은 대표적인 네트워크 프로토콜이라 할 수 있다.[3]

하지만, 이러한 프로토콜들은 굉장히 하드웨어에 의존

적이고 다른 시스템과의 통신에 있어서도 다소의 독자적인 처리과정이 필요하며 일반 사용자들이 쉽게 프로그래밍에 응용할 수 있는 API도 보편적이지 못하다.

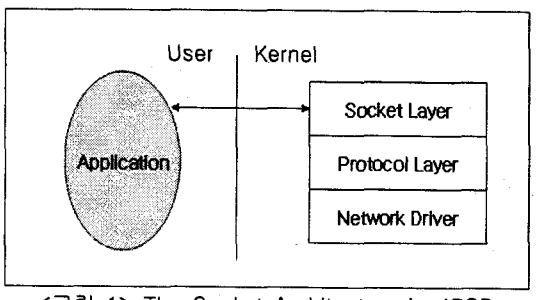
반면에 TCP/IP의 대표적인 프로그래밍 인터페이스인 Socket은 앞의 프로토콜들과는 달리 범용적인 인터페이스를 지원하고 있다. Socket은 원래 Unix BSD 시스템의 프로세스간 통신을 위하여 개발된 시스템 API이다. 원래 API란 굉장히 광의적인 개념인데 프로그램을 만드는 사용자가 해당 프로그래밍 언어와 연동하여 사용할 수 있도록 미리 개발된 함수 형태의 인터페이스 프로그램을 의미한다. 그러므로 API의 가용여부는 사용하는 운영체제와 프로그래밍 언어에 의해 좌우된다.[3]

Unix 시스템의 통신 API인 BSD Socket은 윈도우즈 프로그래밍의 데이터 통신 인터페이스인 Winsock API 개발의 모체가 되었고 실제로 대부분의 Winsock API의 시스템 함수들은 이 BSD소켓의 형태를 취하고 있다. 그러므로 네트워크 프로그래밍 학습을 함에 있어서 BSD Socket API에 대한 이해는 무엇보다도 중요하다.

### 2.1 BSD Socket

1982년 4.1c BSD Unix 시스템에서 Socket이란 개념이 처음 등장하게 되었다. 그 후 4.2 버전에서 본격적으로 사용되기 시작한 Socket API는 4.3 버전에 이르러서 오늘날의 구조와 동일한 형태를 취하게 되었으며 <그림 1>은 BSD 시스템에서의 Socket의 계층 구조를 나타낸다.[2]

<그림 1>에서 보듯이 어플리케이션은 Socket 계층에서 취급되는 시스템 API를 호출한다. Socket 계층은 시스템 호출 시에 생성되는 파일 조건자(File Descriptor)를 변환하여 호출함으로써 프로토콜 계층에 있는 루틴들을 실행시킨다. Socket의 데이터 구조는 Socket 계층과 프로토콜 계층 사이에서 서로 공유된다. 프로토콜 계층 자신은 OSI 7 계층에 상응하는 여러 응용 프로토콜을 포함하고 있으며 Socket 계층과 네트워크 드라이버와의 데이터 송수신을 담당하게 된다.[2]




<그림 1> The Socket Architecture in 4BSD

### 2.2 구성주의

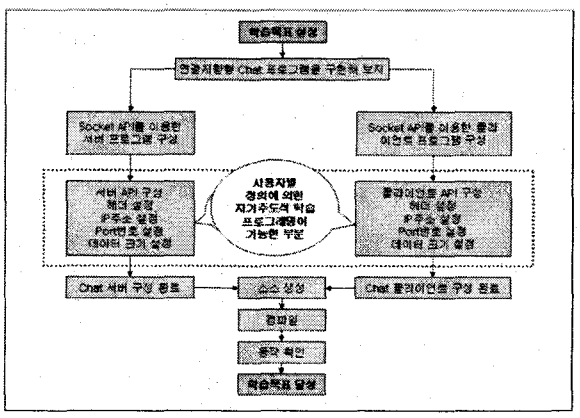
1980년대 이후 교육 현장에서 나타나는 문제점들을 해결하기 위한 대안적 교수-학습 체제로 제기된 구성주의 학습 이론은 기존의 학습관과는 차별화된 학습관을 가지고 있다. 기존의 학습관이 학습 결과의 평가에 중점을 두고 이루어진 것이라면 구성주의 학습관은 객관주의적인 지식의 존재를 부정하는 상대주의적 인식론에 근거하고 있다. 즉, 구성주의에서의 학습은 개인적 경험과 흥미에 따라 지식의 가치가 판단된다. 구성이라는 용어도 활동의 결과가 아니라 지식을 습득함에 있어서 학습자 스스로가 방법을 설계해 나가는 구성과정을 의미한다.[1]

구성주의의 기본 가정은 학자들에 따라 약간의 차이가 있으나 <표 1>과 같이 세 가지 기본 가정을 바탕으로 하고 있다.

<표 1> 구성주의 학습의 기본 전제

| | |
|---|---|
| 학습의 정의 | 개인의 주관적 경험과 사회적 상호작용을 통한 의미의 구성 |
| 학습자관 | 환경과 상호작용하여 의미를 구성하는 학습자 |
| 교사의 역할 | 촉진자, 안내자, 공동참여자 |

본 연구에서는 구성주의의 여러 가지 학습이론 중 자기주도적 문제 중심학습(Problem Based Learning)[1]을 기반으로 하여 네트워크 프로그래밍을 학습함에 있어서 기존 교재들의 문제점인 실습부족과 경험적인 문제점을 지적하고, 학습자들이 이론적 개념을 습득하는 것에 그치는 것이 아니라 프로그래밍을 학습하는 과정을 구체적인 실습을 통해 Socket의 기본원리를 습득하고, 응용 네트워크 어플리케이션의 개발에 쉽게 접근할 수 있는 유저 인터페이스 환경을 제공함으로써 네트워크 프로그래밍 지식의 완성을 이룰 수 있도록 한다. <그림 2>는 본 연구에서 추구하고자 하는 자기 주도적 네트워크 프로그래밍 학습의 절차를 예시하고 있다.




<그림 2> 시스템의 학습목표 구현 절차의 예

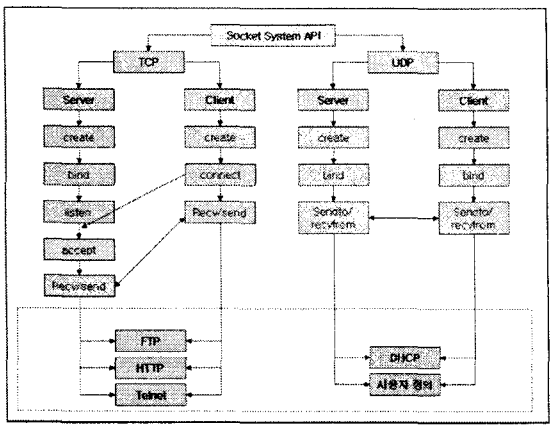
## 3. 시스템 설계 및 구성

<표 2>에 나타나 있듯이 시스템에서 제공되는 범주를 크게 두개의 큰 영역으로 구분하였고 이것은 본 시스템의 기본적인 주제가 된다.

<표 2> 시스템의 두 가지 테마

| Socket 기본 프로그램 | Socket 응용 프로그램 |
|---|---|
| TCP, UDP 방식의 Socket 기본 프로그램 | Socket API를 기반으로 하는 네트워크 응용 프로그램 |
| | 사용자 정의 프로그램 |

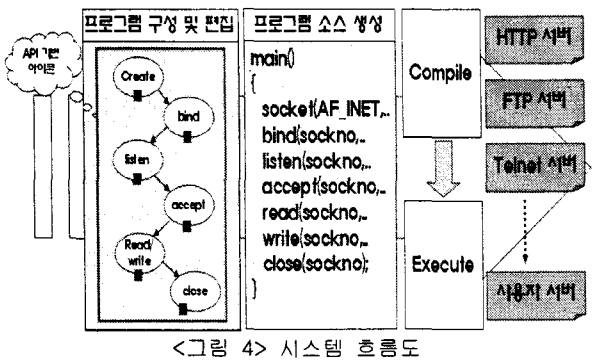
크게 연결지향방식(TCP)과 비연결지향방식(UDP)으로 나누어지는 Socket 기반 프로그램들은 일종의 서비스 제공자 역할을 수행하는 서버 프로그램과 서비스 수신자 역할을 수행하는 클라이언트 프로그램으로 구성된다. 어떤 방식의 응용프로그램을 설정하느냐에 따라 프로그램은 다른 구조를 갖게 되며 동작원리도 서로 다르게 된다.<그림 3> 은 시스템에서 제공하는 Socket API의 기본적인 동작원리와 구조를 설명하고 있다.



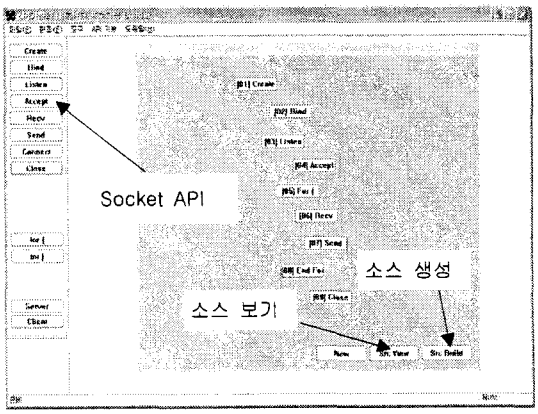

<그림 3> 시스템의 Socket API 함수와 동작(부분)

사용자는 기존과 같이 코딩을 하는 것이 아니라 시스템에서 제공하는 Socket의 API함수를 작업창에 올려놓고 프로그램을 구성한 후<그림 5 참조> 소스를 생성하고 컴파일 및 실행까지 하게 된다. 이것은 시스템의 기본적인 흐름을 나타내며 다른 응용 어플리케이션의 구성도 같은 방식으로 수행된다.

<그림 4>은 시스템의 전체적인 업무흐름도를 나타낸다.




<그림 4> 시스템 흐름도




<그림 5> TCP 서버를 API로 구성한 화면

3.1 시스템 기능

3.1.1 프로그램 작성 및 편집 기능

첫 번째로, Socket의 기본 시스템 API(create, bind, listen, accept, connect, recv, send 등)들은 그 속성이 버튼 형태로 캡슐화 되어 있고 이를 사용자가 선택할 수 있다. 예를 들어, Socket을 생성하는 시스템 함수인 create가 선택된다면, 하나의 시스템 API로 독립되어 프로그램소스를 생성하는 작업 창 위에 놓이게 되며 이것은 클릭 & 클릭 방식으로 생성된다.

이것은 MFC의 CButton을 상속받은 새로운 개체로서 이 개체가 구성이 될 때 각각의 시스템 함수 구성에 필요한 파라미터 값 등의 환경 변수들을 입력으로 받아들인다.

프로그램 생성절차를 설명하면 사용자는 먼저 서버, 클라이언트 중 어떤 방식의 프로그램을 구성할 것인지 선택한 후 앞서 언급한 클릭 & 클릭 방식으로 각각의 독립된 API들을 작업창에 생성한다. 이 API들을 프로그램의 기동순서에 따라 순서대로 서로 조합을 시키면 이 API 버튼들의 조합은 TCP 또는 UDP방식의 서버나 클라이언트 프로그램들을 이루는 API의 조합이 되고, 이것은 본 시스템의 가장 기본적인 기능이 된다.

두 번째로, Socket을 기반으로 한 여러 네트워크 응용 어플리케이션(FTP 서버, FTP 클라이언트, Chat서버, Chat클라이언트, HTTP, Telnet 등)을 생성할 수 있다.

서론에서도 언급했지만 프로세스간 통신을 하는 네트워크 어플리케이션은 대부분 Socket을 기반으로 작성되어 있다. 이것은, Socket이 표준적인 인터페이스를 제공하기 때문이다.

예를 들어, 하나의 TCP서버를 생성하였다면 이 TCP서버를 응용한 FTP서버를 만들 수도 있고, Telnet서버를 구성할 수도 있으며 각각의 특성에 맞는 독립적인 서버를 생성할 수 있다. 이는 모든 네트워크 응용 어플리케이션들이 비슷한 방식의 프로그래밍 구조를 취하고 있기 때문이다.

세 번째로, 사용자 정의 네트워크 프로그램을 생성할 수 있다. 네트워크 프로그래밍의 기본 원리와 인터페이

스를 벗어나지 않는 범위 내에서 사용자가 임의로 프로그램을 구성할 수 있다. 예를 들어, UDP방식의 네트워크 프로그램들은 보통 동기를 맞추지 않는 것이 프로토콜의 기본 원리이지만 요즘 주목받고 있는 Reliable UDP방식은 네트워크 응용 프로그램이 UDP 방식을 사용한다고 할지라도 동기를 맞추는 TCP 형태로 구현될 수도 있음을 의미한다.[13] 이것은 게임 같은 응용 어플리케이션에 많이 사용되고 있다.

이외에도 UDP 프로그램의 헤더를 TCP 헤더나 HDLC방식 등 사용자가 원하는 형태로 구성할 수도 있다. 이것은 어디까지나 학습의 차원에서 이루어지는데 사용자가 임의로 헤더나 연결 방식 등을 스스로 구성하고 적용해 보는 과정에서 네트워크 패킷의 형태라든지 헤더의 구성변수, Socket API의 원리와 지식, 네트워크 프로그램의 작동방식 등에 익숙해졌을 때 학습효과가 이루어졌다고 정의하며 이것은 본 연구가 추구하고자 하는 자기주도적 네트워크 프로그래밍 학습의 가장 핵심적인 목표가 된다.

### 3.1.2 소스 생성 기능

<그림 5>는 사용자가 TCP 서버를 구현한 화면이다. 여기서 각각의 API는 소스에 삽입될 각자의 소스 정보와 일련번호를 가지고 있다. 이것은 API생성 시에 설정된 IP주소라든지 Port번호, 송수신 패킷의 크기, 그리고 삽입될 해당 시스템 API의 함수 등을 의미하는데 소스를 생성하고 실행시키는데 필수적인 환경변수가 된다.

일단 소스를 생성하게 되면 C/C++ 형태의 소스가 생성되며 이것은 컴파일 할 수 있는 기본적인 초기 소스코드가 된다. 물론 기존의 빌더들처럼 소스를 열어 수정할 수도 있다.

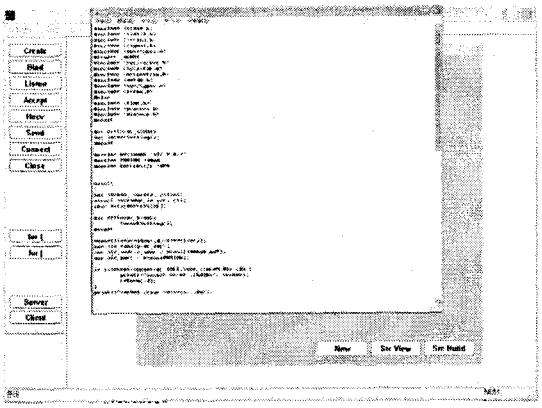
버튼의 위치를 바꾸거나 새로운 버튼을 생성하면 다시 소스를 생성한다. 화면에 나오는 버튼의 위치와 연결순서에 따라 소스가 생성된다. 물론 여기서 생성되는 소스는 최소한으로 동작할 수 있는 기본적인 프로그램의 골격을 제공한다.

좀 더 세밀한 프로그램을 작성하기 위해서 각각의 API들을 더블클릭하면 해당 API의 소스 위치로 이동하게 되며 변수를 수정하거나 if, switch, while loop등과 같은 새로운 알고리즘을 삽입하는 구체적이고 세부적인 컨트롤에 관한 부분은 사용자에게 주어진다.

### 3.1.3 소스 컴파일 및 생성 기능

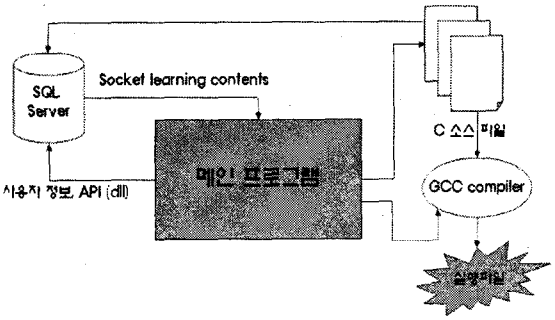
사용자가 생성한 소스는 컴퓨터에 파일로 저장되며 컴파일 하여 실행 가능한 이진파일로 생성, 실행할 수 있다. 생성하고자 하는 바이너리 파일의 이름을 수정하여 컴파일 하는 것도 가능하다.

사용자가 임의의 소스 명을 부여하지 않는다면 시스템은 내부적으로 정의된 소스 명을 부여하여 컴파일 한다. 컴파일러로는 대표적인 Windows용 컴파일러인 Cygwin을 사용한다. <그림 6>은 생성한 소스를 편집기를 사용하여 오픈한 화면을 나타낸다.

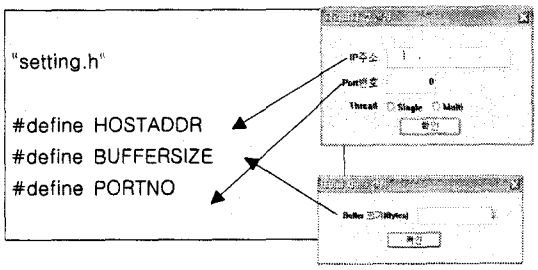


<그림 6> 소스 생성 및 편집

## 3.2 주요모듈 설명



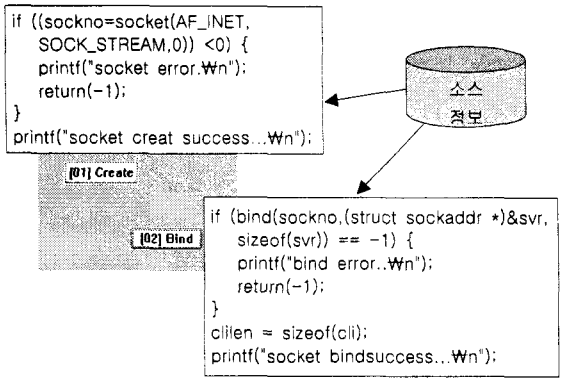

<그림 7> 시스템 구성도




<그림 8> 변수 매핑 과정

소스 생성기는 각각의 API 함수에 대한 소스와 변수를 가지고 있다. <그림 8>는 사용자가 API 생성 시에 설정되는 변수들이 프로그램 소스에서 공통적으로 사용되는 전역변수와 매핑 되는 과정을 나타낸다.

또한, 각각의 API 함수는 <그림 9>과 같이 데이터베이스에 저장되어 있는 자신의 소스코드를 소스 생성 버튼이 눌려졌을 때 메인소스로 편입시킨다. 작업창에 구현된 각각의 API를 더블클릭하면 해당 소스의 위치로 이동하며 편집도 가능하다.




<그림 9> API함수와 소스의 매핑과정

### 3.5 메뉴 구성

<표 3>은 각각의 메뉴의 기능을 간략히 설명하고 있으며, <그림 5>에서 나타나 있듯이 초기화면의 UDP Box나 TCP Box는 사용자가 임의로 지정할 수 있게 된다. 즉, UDP Box나 TCP Box 또는 둘 다 한꺼번에 불러내어 사용할 수도 있다.

<표 3> 시스템 메뉴

| | | |
|---|---|---|
| 파일 | | 현재의 프로젝트 작업 영역을 생성하고 저장한다. |
| 편집 | | 현재의 프로젝트 작업 영역을 편집한다. |
| 도구 | 컴파일 | 생성된 프로그램 소스를 컴파일 하고 결과를 나타낸다. |
| | 실행 | 컴파일된 이진 파일을 실행한다. |
| | TCP Box | TCP 관련 시스템 함수들을 가진 Tool Box |
| | UDP Box | UDP 관련 시스템 함수들을 가진 Tool Box |
| | Work Box | 각각의 API 함수들을 생성하고 소스 생성 및 편집 등을 수행하는 작업창 |

## 4. 결 론

초보 학습자의 입장에서 네트워크 프로그래밍의 지식습득과 응용에 있어서 사용자가 쉽게 접근할 수 있는 교육 시스템이 있다면 그보다 바람직한 것은 없다. 물론, 본 시스템이 여러 네트워크 프로그래밍을 자동적으로 구성할 수 있는 여건과 편리한 사용자 인터페이스를 모두 제공하지는 않는다. 좋은 프로그램을 만드는 것은 어디까지나 사용자의 능력이지만 그러한 프로그램을 만들기 위해 좀 더 쉽게 접근할 수 있는 사용자 인터페이스를 제공하는 것이 빌더와 같은 통합된 툴이라 할 수 있다.

네트워크 프로그래밍을 전문가적인 지식과 경험이 있어야만 할 수 있다는 편견을 버리게 하는 것이 이 시스템의 기본적인 목적이다.

본 시스템을 학습에 이용하게 될 때 얻어지는 기대효과는 다음과 같다.

첫째, 클릭 & 클릭 기반으로 네트워크 프로그램을 구성함으로써 코딩의 어려움 없이 소스 작성과 실행까지 할 수 있는 사용자 인터페이스를 제공한다.

둘째, 여러 다양한 네트워크 응용 어플리케이션(FTP, Telnet, HTTP, Chat 등)을 마우스 클릭만으로 기본적인 구성이 가능하다.

셋째, Socket API를 스스로 이용하여 여러 형태로 구성할 수 있는 기반을 제공함으로써 네트워크 프로그래밍 지식을 자기주도적인 학습과 실습으로 습득하게 한다.

향후 연구과제로는 보다 다양한 네트워크 응용 프로그램들을 구성하는 API의 캡슐화와 융통성 있는 콘텐츠 관리가 요구되며 현장에서 사용할 수 있는 프로그래밍 코스웨어와 연동되는 평가 시스템으로의 확대가 요구된다.

## 참고문헌

[1] 백영균 외 8명 공저, "교육방법 및 공학", 학지사, 2005
[2] Stephen A Rago, "Unix System V Network Programming", Addison-Wesley, 1995
[3] Stevens, "Unix Network Programming", Prentice-Hall, 1995
[4] 김용성, "Visual C++ 완벽 가이드", 영진닷컴, 2004
[5] 장중한, "Visual C++/MFC 따라 하기", 구민사, 2001
[6] 최호성, "MFC 정복", 가남사, 2004
[7] Komata Mitsuyuki, "TCP/IP Network Programming", 영진닷컴, 2003
[8] Yasutaka Kumei, "네트워크 프로그래밍", 정보문화사, 2005
[9] 홍릉과학 출판사 편집부, "Unix System Programming", 홍릉과학 출판사, 1997
[10] Behrouz A. Forouzan, "Data Communications and Networking", 2003
[11] 김창훈, "효과적인 소켓 프로그래밍 교육을 위한 교재 개발", 경성대학교, 2004
[12] 최대석, "네트워크 응용시스템의 효율적 개발을 위한 BSD 소켓 프로그래밍에 대한 연구", 경성대학교, 2003
[13] Volula Fotopoulos, Catherine Heaberlin, "Reliable UDP(RDP) Transport for CORBA", OMG Workshop, 2002
[14] 김선우, "윈도우즈 네트워크 프로그래밍", 한빛미디어, 2004