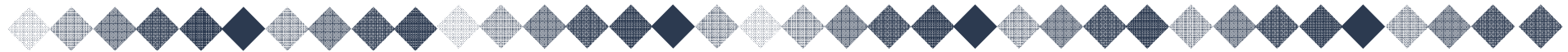


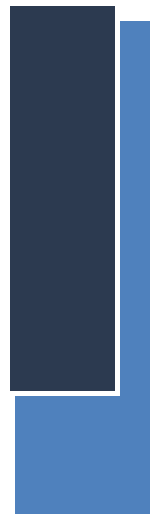


Data Communication (CE14773)



Chungnam National University
Dept. of Computer Science and Engineering
Computer Communication Laboratory

Sangdae Kim - 00반 / Cheonyong Kim- 01반





Contents

- ◆ Thread
 - ❖ Thread concept
 - ❖ Thread in chatting & filetransfer

- ◆ Chatting & File Transfer.
 - ❖ Overview



Thread

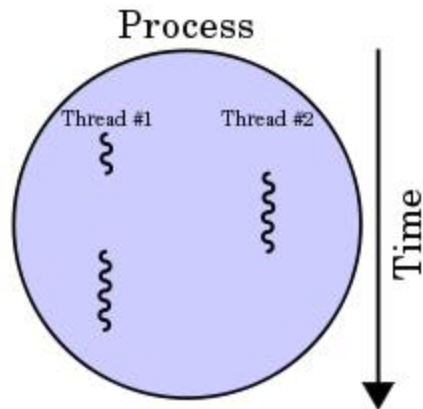




Thread Concept

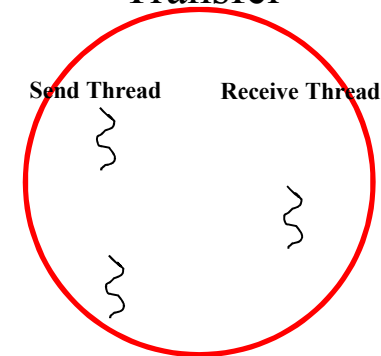
◆ Concept

- ❖ 어떠한 프로그램 내에서(특히 프로세스) 실행되는 흐름의 단위
- ❖ 프로세스 내의 메모리를 공유하여 사용
- ❖ 예를 들어, 워드프로세스(하나의 프로세스)에서 글을 작성하는 것과 맞춤법 검사를 수행하는 것은 별도의 쓰레드라고 볼 수 있음
- ❖ 이번 Chatting & File Transfer에서는 송신/수신부에 이를 활용할 수 있음



두 개의 쓰레드를 실행하고 있는 하나의 프로세스

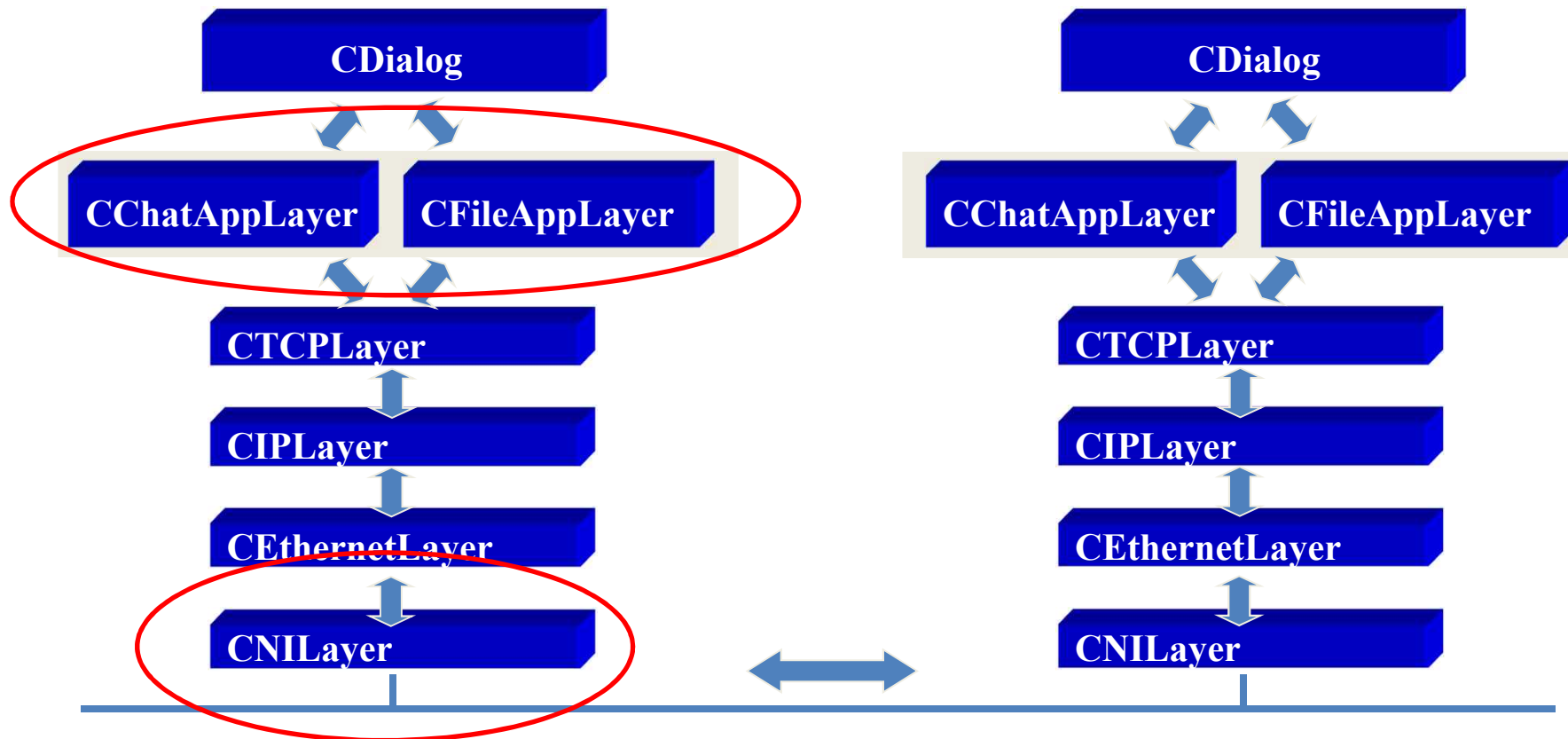
Chatting & File Transfer



Send, Receive를 실행하고 있는 Transfer 프로세스



Thread in Chatting & Filetransfer





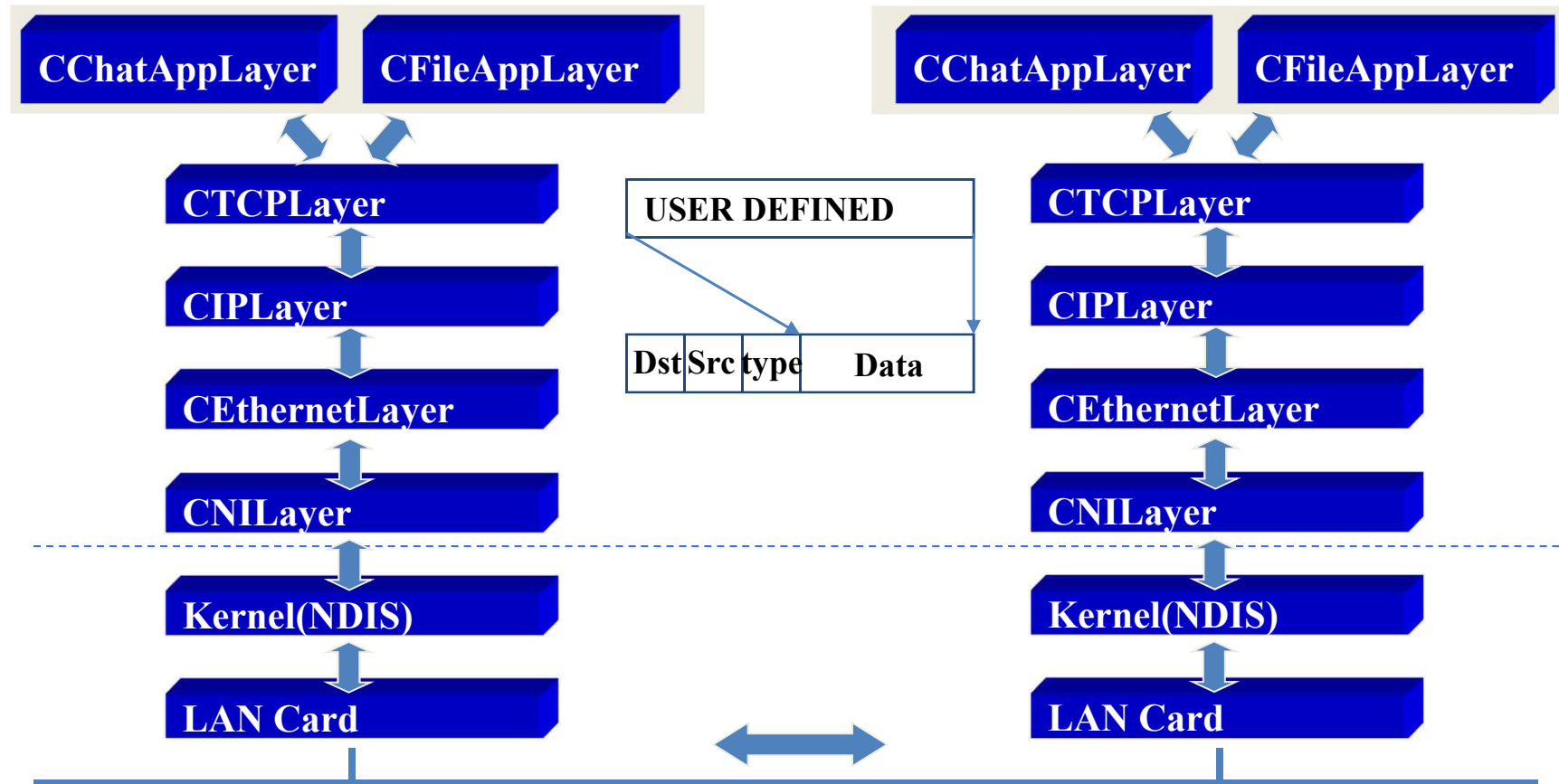
Chatting & File Transfer Overview





Introduction

◆ Layer Model

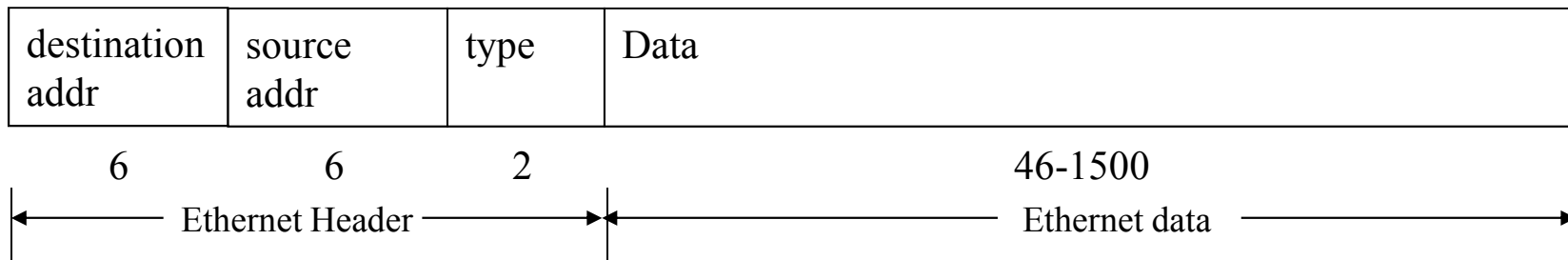




Data Structures

◆ Ethernet Protocol

```
typedef struct _ETHERNET
{
    ETHERNET_ADDR enet_desaddr ;    // 상대방 주소
    ETHERNET_ADDR enet_srcaddr ;    // 자기 주소
    unsigned short enet_type ;      // frame data type
    unsigned char  enet_data[ MAX_ETHERNET_DATA ] ;
} ETHERNET, *LPETHERNET ;
```

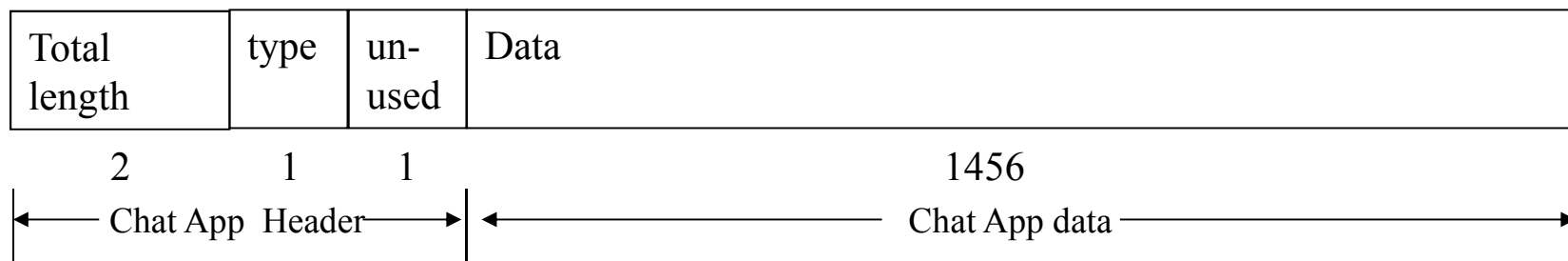




Data Structures

◆ Chat Application Protocol

```
typedef struct _CHAT_APP
{
    unsigned short    capp_totlen ;      // 메시지 총 길이
    unsigned char     capp_type ;       // 메시지 타입
    unsigned char     capp_unused ;     // 우선 사용 안함
    unsigned char     capp_data[ MAX_APP_DATA ] ;
} CHAT_APP, *LPCHAT_APP ;
```

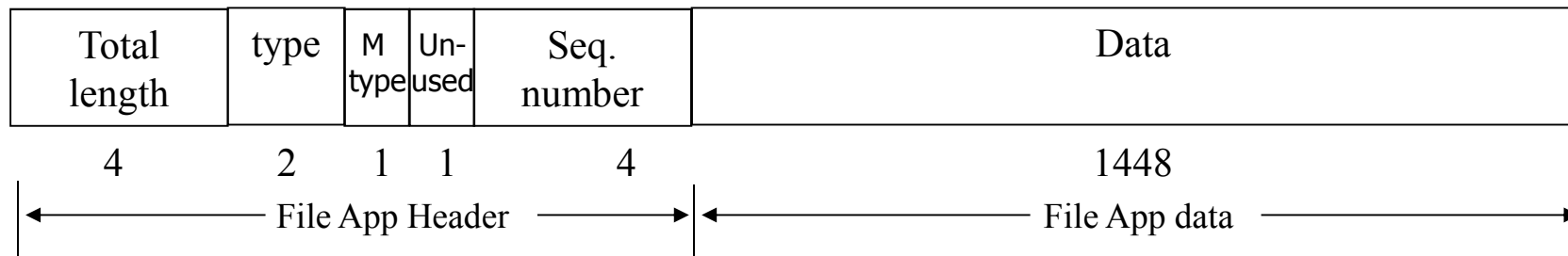




Data Structures

◆ File Transfer Application Protocol

```
typedef struct _FILE_APP
{
    unsigned long    fapp_totlen ;           // 총 길이
    unsigned short   fapp_type ;            // 데이터 타입
    unsigned char     faa_msg_type          // 메시지 종류
    unsigned char     unused ;              // 사용 안함
    unsigned long     fapp_seq_num ;        // fragmentation 순서
    unsigned char     fapp_data[ MAX_APP_DATA ] ;
} FILE_APP, *LPFILE_APP ;
```





Requirements Analysis

◆ Basic Design

- ❖ One to one 방식
 - ◆ 시스템에는 하나의 프로세스만 동작하며 통신할 상대는 다른 시스템의 프로세스으로써 1:1 통신을 한다.
- ❖ Chatting과 File 전송을 동시에 가능
 - ◆ File Transfer Thread 구현
- ❖ Chatting Message와 File Size는 제한 없이 전송 가능
 - ◆ Fragmentation 가능
- ❖ 실제 Network Protocol (Ethernet Protocol)을 이용하여 Ethernet Frame을 송수신
 - ◆ Packet driver 이용하여 정보를 얻어 winPcap을 이용하여 송수신
- ❖ Ethernet Frame을 수신
 - ◆ 수신만을 하기 위한 Thread 구현
- ❖ class CBaseLayer는 homework #2(IPC)의 소스를 그대로 사용할 것



Requirements Analysis (cont.)

◆ Optional Design

❖ Reliable Communication

- ◆ 송수신 간의 데이터 전송 시 신뢰성을 보장하기 위한 메시지 구현
- ◆ 데이터 전송 후 모든 데이터 전송이 성공되면 수신 측에서 잘 받았다는 메시지를 송신 측에게 보내고 이를 받으면 송신 측에서는 다음 전송을 대기한다.
- ◆ 만약, 송신 측에서는 보내고 수신 측에서는 수신 준비가 안 되었을 경우 송신 측에서 수신 대기 하지 않는 것을 감지하여 (Timer를 작동 후에 2초간 대기하여 응답이 없을 경우) “time-out” 메시지를 화면에 출력할지 아닐지를 결정
- ◆ Hint. : chatApp와 FileApp의 data type을 추가하여 그 자료구조를 이용
- ◆ 구현 시 가산 점수 부여



Requirements Analysis (cont.)

◆ Chatting Application Layer (class CChatAppLayer)

- ❖ Field Port Number of TCP Layer : 2080 (user defined)
- ❖ Chatting Message에 대해서 Fragmentation을 함으로써 길이에 제한 없이 전송하도록 구현
- ❖ Fragmentation
 - ◆ 전송 시
 - 데이터를 일정 크기로 잘라서 하나씩 전송한다.
 - capp_type에 메시지 데이터가 첫 부분(0x00)인지 중간 부분 (0x01) 인지, 마지막 부분 (0x02) 인지 구분하여 저장 후 전송
 - ◆ 수신 시
 - 첫 부분 일 경우 그 크기만큼 버퍼 할당하고 중간부분이면 계속 버퍼에 쌓는다.
 - 마지막 부분일 경우 받은 메시지를 모두 모아서 Dialog 객체에게 넘겨준다.



Requirements Analysis (cont.)

◆ File Transfer Application Layer (class CFileAppLayer)

- ❖ Field Port Number of TCP Layer : 2090 (user defined)
- ❖ File에 대해서 Fragmentation을 함으로써 size에 제한 없이 전송하도록 구현
- ❖ Fragmentation

◆ 전송 시

- 파일명을 얻고, 파일을 열어서 버퍼에 저장
- 첫 부분 : 파일에 대한 정보를 전송 (파일 명), fapptype = 0x00
- 중간부분 : 파일의 데이터가 전송 (Fragmentation적용), fapptype = 0x01
- 마지막 부분 : 모든 데이터가 다 전송되고 마지막이라는 메시지, fapptype = 0x02
- 파일을 닫는다

◆ 수신 시

- 첫 부분(fapptype = 0x00) 일 경우 그 파일명을 받아서 파일을 생성
- 중간부분 (fapptype = 0x01) 일 경우 파일에 데이터를 써서 덧붙인다
- 마지막 부분 (fapptype = 0x02) 일 경우 파일의 크기 및 순서를 고려 잘 전송 받았는지 확인하고 파일을 닫는다.



Requirements Analysis (cont.)

◆ TCP Layer(class CTCPLayer)

- ❖ 수신 받은 데이터로부터 Port 번호를 읽어서 Chatting Application Layer 나 File Transfer Application Layer로 구분해서 보낸다.
- ❖ 그 밖의 구현에 필요한 member method는 추가해도 무방
- ❖ Dest_Port가 2080이면 ChatappLayer에게, 2090이면 FileAppLayer에게 데이터 넘김

◆ IP Layer (class CIPLayer)

- ❖ 미리 저장한 IP주소를 저장해서 송신한다.
- ❖ 수신 받은 데이터로부터 IP주소를 읽어서 자신의 것이면 TCP Layer로 보내고, 아니면 Discard한다.



Requirements Analysis (cont.)

◆ Ethernet Layer(class CEthernetLayer)

- ❖ 수신받은 프레임 중에서 destination address가 자기 것이 아니면 discard한다.
- ❖ 그 밖의 구현에 필요한 member method는 추가해도 무방
- ❖ 헤더를 제외한 데이터 부분은 IPLayer로 전송

◆ Network Interface Layer (class CNILayer)

- ❖ winPcap을 이용하여 기본적인 packet 송수신 operation 구현한 class
- ❖ Adapter와 상위 Layer간의 데이터 송수신의 중간자 역할을 담당
- ❖ CBaseLayer를 상속
- ❖ Packet32.h를 이용하여 Mac 정보를 얻어온다.



Requirements Analysis (cont.)

◆ Thread 구현

❖ UINT ReadingThread(LPDWORD lpdwParam)

- ◆ 수신된 Ethernet Frame을 받아서 상위 레이어로 올리는 역할을 담당

- ◆ Synchronous Mode로 동작

BOOL PacketReceivePacket(LPADAPTER AdapterObject,
LPPACKET lpPacket, BOOLEAN Sync, PULONG BytesReceived)
// sync = true ;

◆ UINT FileTransferThread(LPDWORD lpdwParam)

- Chatting과 File Transfer를 동시에 가능하게 하기 위해 Thread를 구현



User Interface (UI)

◆ Chatting

- ❖ Chatting Message가 출력될 Window
- ❖ 보내질 Chat message가 입력될 Window
- ❖ Send Button

◆ File Transfer

- ❖ File Name을 불러올 Button
- ❖ File Name을 화면에 출력할 Window
- ❖ 전송 상태를 보여줄 Window
- ❖ Send Button

◆ Address

- ❖ Source Ethernet Address 보여줄 Window
- ❖ Destination Ethernet Address 입력할 Window
- ❖ 설정 Button



User Interface (cont.)

◆ Example

Chatting and File Transfer using Ethernet Protocol

Chatting

채팅 창(CListbox)

보낼 메시지창(CEdit) Send(S)

File Transfer

파일명 입력(CEdit) File(Q)

전송상태 출력(CStatic) Send(E)

Source

Ethernet Address

주소출력(CStatic)

IP ADDRESS

0 . 0 . 0 . 0

Destination

Ethernet Address

주소입력(CEdit)

IP ADDRESS

0 . 0 . 0 . 0

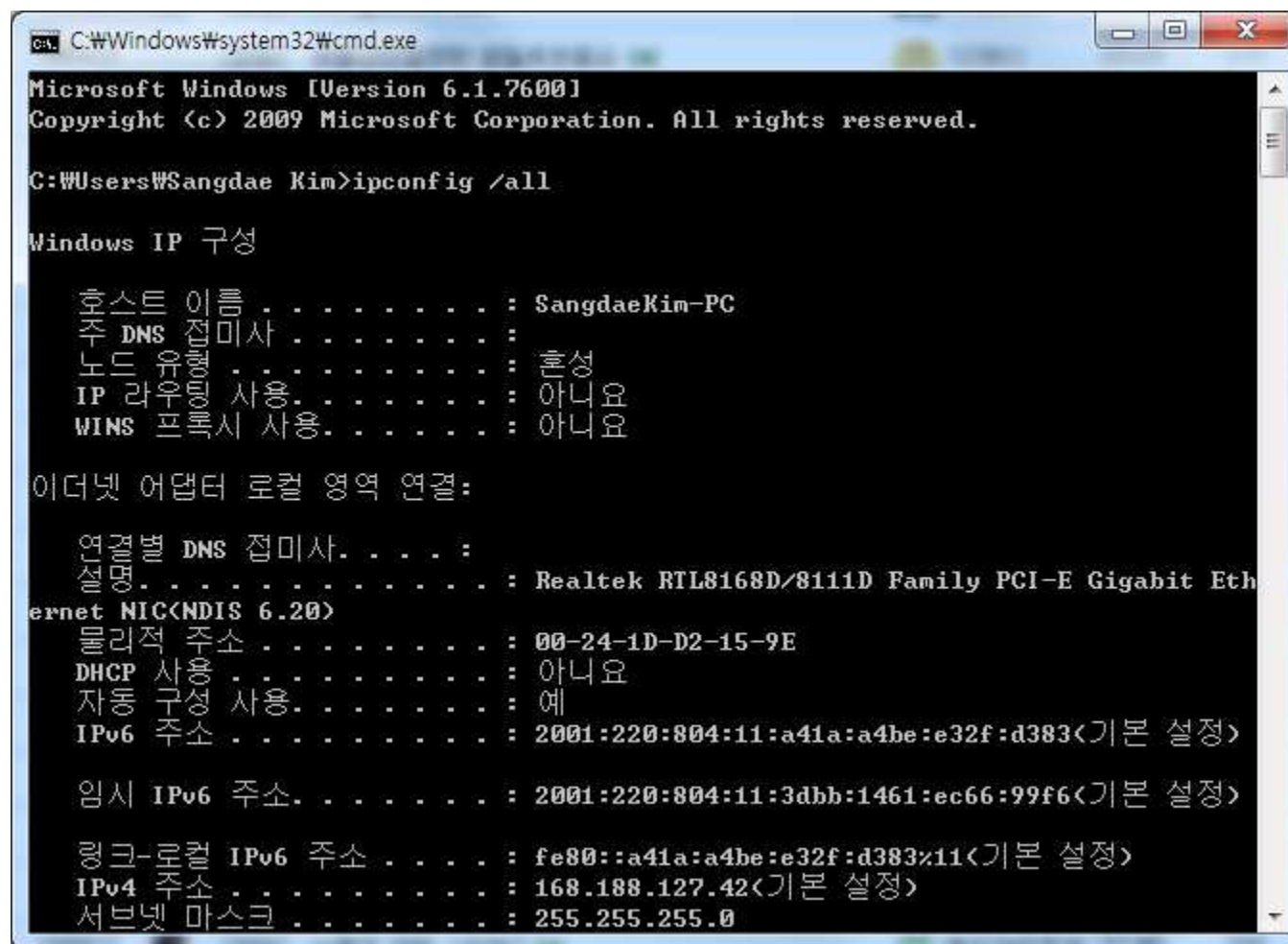
화일명 입력(CButton)

주소설정버튼(CButton)

설정(Q)

◆ ipconfig /all

- ❖ Can get the IP address and Ethernet Address of the Network Interface Card.



```

C:\Windows\system32\cmd.exe
Microsoft Windows [Version 6.1.7600]
Copyright (c) 2009 Microsoft Corporation. All rights reserved.

C:\Users\Sangdae Kim>ipconfig /all

Windows IP 구성

호스트 이름 . . . . . : SangdaeKim-PC
주 DNS 접미사 . . . . . :
노드 유형 . . . . . : 혼성
IP 라우팅 사용 . . . . . : 아니요
WINS 프록시 사용 . . . . . : 아니요

이더넷 어댑터 로컬 영역 연결:

연결별 DNS 접미사. . . . . :
설명. . . . . : Realtek RTL8168D/8111D Family PCI-E Gigabit Eth
ernet NIC(NDIS 6.20)
물리적 주소 . . . . . : 00-24-1D-D2-15-9E
DHCP 사용 . . . . . : 아니요
자동 구성 사용 . . . . . : 예
IPv6 주소 . . . . . : 2001:220:804:11:a41a:a4be:e32f:d383<기본 설정>

임시 IPv6 주소. . . . . : 2001:220:804:11:3dbb:1461:ec66:99f6<기본 설정>

링크-로컬 IPv6 주소 . . . . . : fe80::a41a:a4be:e32f:d383%11<기본 설정>
IPv4 주소 . . . . . : 168.188.127.42<기본 설정>
서브넷 마스크 . . . . . : 255.255.255.0
  
```



Tips (cont.)

- ◆ homework #2에 그대로 Layer만 삽입, 새로 구현 하지 말고 Workspace를 새로 작성하여 직접 UI도 그리면서 프로젝트 수행
- ◆ 과제 수행 전에 driver development kit와 Packet Protocol을 설치
 - ❖ VC 환경설정(Header Directory Include)
- ◆ 평소에 과제를 수행할 때는 전용선을 사용하여 네트워크에 영향을 주지 않도록 주의
 - ❖ 전용선은 수업시간에 조당 1개씩 배당되며 데모 후 반납
- ◆ 다큐먼트 작성에 소홀하지 않도록 주의
- ◆ 과제 제출일
 - ❖ 최종 제출: 4월 14/15일 24:00 시까지
 - ❖ 데모: 4월 중



Tips (cont.)

◆ 제출 양식

❖ 파일명 예시

◆ [분반]조_hw과제번호_(코드/문서)

파일명 예시

- 00반 1조 1번째 과제 제출 :

- 코드 : [00]1_hw01_code.zip
- 문서 : [00]1_hw01_doc.doc (or .docx, .hwp)
- 압축파일 : [00]1_hw01.zip (상기의 두 파일을 압축)

- 01반 3조 2번째 과제 제출 :

- 코드 : [01]3_hw02_code.zip
- 문서 : [01]3_hw02_doc.doc (or .docx, .hwp)
- 압축파일 : [01]3_hw02.zip (상기의 두 파일을 압축)

❖ 메일 제목 예시:

◆ [분반]3_hw02_조장

- 00반 1조 1번째 과제 제출 : [00]1_hw01_조장명
- 01반 3조 2번째 과제 제출 : [01]3_hw02_조장명

❖ Debug 제외한 나머지 코드 압축