정보학연구 제5권 제1호 2002년 3월(pp.69-77)
The Journal of Information Technology, Vol.5, No.1

블루투스 HCI 계층을 위한 클래스 구조의 설계 및 구현

김식* · 류수형**

요 약

블루투스는 차세대 근거리 무선통신 기술로 각광받고 있으며, 프로토콜 스택은 블루투스 응용 시스템을 위한 다양한 서비스를 제공한다. 블루투스 규격은 개방된 세계적인 규격으로 완전한 시스템을 정의하고 있지만 프로토콜 스택은 개발자에 따라 다른 방법으로 하드웨어와 소프트웨어의 역할을 기능성 중심으로 독자적으로 분할하여 설계 및 구현이 가능하다. 이는 프로토콜 스택의 기능적인 면에서 두 계층에서의 구현 방법이 다르다는 것을 의미한다. 본 논문에서는 프로토콜 스택 개발의 첫 단계로 소프트웨어계층의 최하위 계층인 HCI 계층의 개발에 중점을 두었다. 그 결과로서 HCI 계층의 기능적 역할 분담과 흐름제어 기능 및 패킷들 사이의 관계를 설계 및 구현하였다. 실험결과는 다른 블루투스 장치들과의 연결 과정을 보여주고 있으며, 이것은 블루투스 모듈 사이에서의 데이터 통신이 적절한 운영을 하고 있음을 증명한다.

Ⅰ. 서론

지난 수년동안 세계적으로 휴대용 장치가 급속히 보급됨에 따라 노트북PC 등과 연결한 모바일 컴퓨팅(mobile computing)이 활발하게 진행되어 왔다. 블루투스는 이러한 각종 전자장치간의 통신에 물리적인 케이블 없이 무선주파수를 이용하여, 고속으로 데이터를 주고받을 수 있는 무선통신 기술이며, 블루투스 프로토콜 스택의 표준규격을 책정하기 위한 목적으로 1998년 에릭슨, 노키아, IBM, 도시바, 인텔 5개 사가 중심 멤버가 되어 Bluetooth SIG(Special Interest Group)라는 컨소시엄을 구성했으며, 현재는 회원사가 전세계적으로 2000여 개에 이를 만큼 블루투스 기술에 대한 업계의 관심이 높아지고 있다[1,2,3,4].

이처럼 차세대 근거리 무선통신 기술로 많은 호응을 불러일으키고 있는 블루투스 기술은 그 응용 범위가 매우 넓기 때문에 블루투스 프로토콜 스택을 응용한 각종 어플리케이션 개발의 필요성은 점점 확대되고 있다. 현재 블루투스 프로토콜은 1.1버전의 규격이 나와있고 이것은 물리적인 부분으로 펌웨어를 기술한 코어(Core)사양과 상호 기기 간의 호환성을 위한 프로파일로 나뉘어져 있다. 프로파일은 블루투스를 최상위 어플리케이션에서 어떻게 사용할지를 정해 놓은 규격으로 유일한 것은 아니며, 응용목적에 따라 다양한 형태로 개발해서 사용하게 된다[1,2]. 이러한 프로파일개발에 적합한 상위계층 프로토콜을 개발하기 위해서는 하드웨어와 소프트웨어의 중간 계층으로, 블루투스 장치와 호스트 사이의 인터페이스 역할을 하는 프로토콜인 HCI 계층의 구현이 필수적이다. 본 논문은 블루투스 프

* 세명대학교 정보통신학과 교수
* 세명대학교 대학원 전산정보학과 석사과정

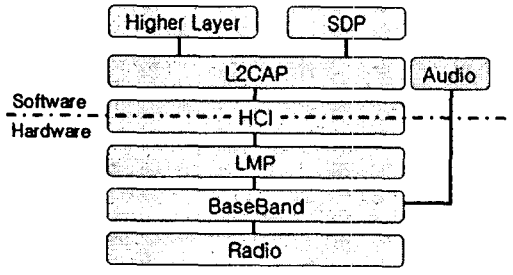
로토콜 스택을 개발하기 위한 첫 단계 연구과제로서 HCI(Host Controller Interface)를 설계 구현하고, 일련의 실험을 통해 구현된 HCI 프로토콜이 정상적으로 동작함을 확인하고 타사의 블루투스어뎁터간의 동작이 호환 가능성을 연구하고자 한다.

II. 블루투스 프로토콜 스택

이 장에서는 먼저 블루투스 프로토콜 스택에 대한 기본구조를 통해서 HCI 계층이 상·하 위 프로토콜들과 어떤 관계에 있는 지와 각 프로토콜 계층의 역할을 살펴보고, HCI의 구현에 필요한 구조와 동작에 대해서 논의하도록 한다.

2.1 블루투스 프로토콜의 기본 구조

블루투스 프로토콜 스택의 기본 구조는 Fig. 1과 같다.



〔 Fig. 1 〕 Bluetooth Protocol Stack

최하위 계층인 Radio부터 Baseband, LMP(Link Manager Protocol)계층과 HCI 계층의 일부는 블루투스 시스템의 하드웨어 및 펌웨어를 구성하는 모듈로서 장치의 특성을 정의하고, 물리적 무선 연결과 생성을 관리하며, 불필요한 신호들에 대한 필터링기능 등을 제공한다. 블루투스 프로토콜 스택에서 하드웨어 계층과 소프트웨어 계층으로 양분되어 있는 HCI 계층은 블루투스 장치와 호스트 사이의 인터페이스 역할을 하는 계층으로 전송되는 패킷들의 타입을 정의하고 있다. L2CAP (Logical Link Control Adaptation Layer)은 프로토콜 종류를 구분하기 위한 멀티플렉싱과 송·수신 데이터에 대한 패킷분할 및 재조합 그리고 QoS(Quality Of Service)정보의 전달을 목적으로 한다. SDP(Service Discovery Protocol)는 블루투스 장치들 상호간에 제공하는 서비스와 속성을 알아내는 역할을 하며, Audio는 BaseBand계층을 통해 직접 음성데이터를 송·수신한다. 그 밖의 상위 프로토콜 계층들은 블루투스 장치의 응용과 관련되는 프로토콜들이다[5,6].

2.2 HCI 계층의 구조와 동작

HCI 계층은 블루투스 하드웨어 부분과 호스트의 소프트웨어 부분으로 나뉘어 져있다. 따라서, 블루투스 하드웨어는 물리적인 버스(USB, RS232C, UART 등)를 통하여 호스트 시스템과 패킷교환을 수행하게 된다. 교환되는 패킷은 Command 패킷, 송신 데이터 패킷, 수신 데이터 패킷, Event 패킷의 네 가지 이며, 앞의 두 가지는 호스트 시스템에서 블루투스 하드웨어로 보내어지는 패킷이고, 뒤의 두 가지는 반대 방향으로 전달되는 것이다[5].

각 패킷의 구성과 HCI 계층의 설계 및 구현을 위한 고려사항은 다음과 같다.

· Command 패킷은 총 6종류 95가지로 구성되어있으며, 구현을 위해서는 모든 HCI Command

를 표준규약에 나와 있는 형태로 구성하여 버스 드라이버로 전달할 수 있는 기능이 있어야 한다.

· Event 패킷은 총 32 가지이며, 구현을 위해서는 버스 드라이버로부터 비동기적으로 전달되는 모든 HCI_Event 패킷을 받은 후, 프로토콜 스택의 상위계층으로 전달하는 기능이 필요하며, 이때 HCI_ Event 패킷은 반드시 해석(Decode)된 것이어야 한다.

· 송 · 수신 데이터 패킷에 대한 구현에는 전송할 데이터를 HCI 데이터 패킷의 표준규약 형태로 구성한 후, 버스 드라이버로 전달 할 수 있어야 하고, 버스 드라이버로부터 비동기적으로 전달되는 모든 HCI 데이터 패킷을 받은 후, 프로토콜 스택의 상위계층으로 전달하는 기능이 필요하다. 이때, 전달받은 HCI 데이터 패킷들은 상위 계층의 요구에 맞게 재구성되어야한다.

호스트에서 블루투스 장치로 전달하는 데이터 패킷은 블루투스 장치의 데이터 버퍼가 넘치지 않도록 보내야 한다. 이를 위해 블루투스 표준규약에서는 일련의 과정을 통한 흐름제어를 제공하며, 이것은 다음과 같은 메커니즘으로 동작한다.

먼저 호스트 측에서 최초에 Read_Buffer_Size_Command를 이용하여 버퍼크기를 구한 후, 버퍼크기의 범위 내에서 데이터 패킷을 전송한다. 전달된 패킷에 의해서 Number_Of_Completed_Packet_Event가 발생하며, 이 Event 정보를 통해 추가로 전달할 수 있는 패킷의 개수를 갱신한다. 이러한 일련의 과정을 흐름제어라고 한다. 흐름제어에 대한 설계시 고려 사항은 다음과 같다.

· 흐름제어는 데이터 패킷을 보내는 쓰레드와 Number_Of_Completed_Packet_Eve nt를 받는 쓰레드를 독립적으로 구현해야 하며, 이들 사이의 정보전달을 위한 대책이 마련되어야 한다.

Ⅲ. HCI 클래스 구조의 설계 및 구현

HCI의 설계는 객체지향설계가 갖는 장점을 갖추기 위해 C++언어에서 제공하는 객체지향구조를 기반으로 했다. 먼저 블루투스 표준규격에서 제공하는 기능을 기능성을 기반으로 분할하여 HCI 클래스구조를 설계하였다. 설계된 HCI의 클래스 구조는 HCI_Command 클래스, HCI_Event 클래스, Pipe 클래스, HCI_Stack 클래스로 구성된다. 다음은 설계된 각 클래스에 대한 정의와 Command전송에 필요한 동작 메커니즘 그리고 데이터 송 · 수신을 위한 흐름제어에 대해 논의한다.

3.1 HCI 계층의 기능성 분할

블루투스 프로토콜 스택에서 HCI계층이 담당해야하는 기능을 상호관계와 흐름제어를 기반으로 분할하면 Fig. 2와 같다. Fig. 2에 표현된 HCI의 기능은 다음과 같다[7].

· 상위계층에서 요구하는 서비스는 HCI계층에서 일련의 HCI명령들로 변환한 후 HCI명령패킷들을 구성하여 버스드라이버로 전달한다.

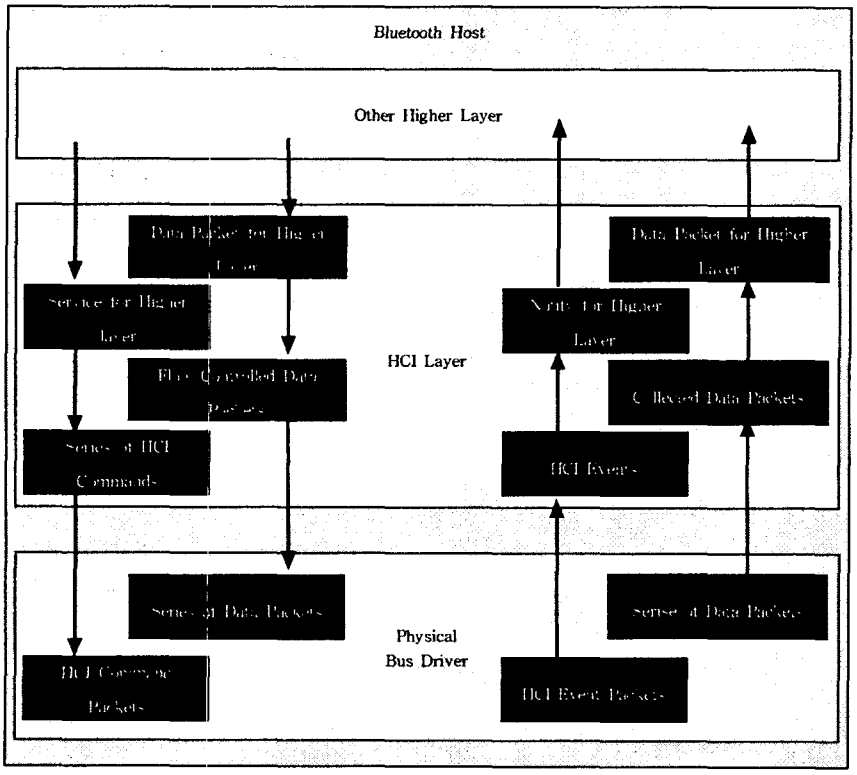
· 상위계층에서부터 전달되는 데이터패킷은 HCI계층에서, 흐름제어에 대한 처리 하에, 일련의 HCI 데이터패킷들로 구성하여 버스드라이버로 전달한다.

· 버스드라이버에서부터 전달되는 HCI이벤트 패킷은 HCI계층에서 분석하여, 상위계층에게 전달해야할 내용들만 통지한다.

· 버스드라이버에서부터 전달되는 데이터패킷은 HCI계층에서 축적하여 상위 계층에서 사용

될 수 있는 형태의 데이터패킷으로 재구성하여 전달한다.

서는 멀티쓰레드를 이용하는 프로그래밍 기법이 필요하다.



[Fig. 2] Functionality and Operational Mechanism in HCI Layer

분할된 구조를 클래스 구조로 설계할 때 블루투스 플랫폼과 개발환경의 방법론 및 객체지향 언어 특성상 고려되어야 요소는 다음과 같다.

· 멀티쓰레드: 버스드라이버로부터 전달되어 오는 HCI이벤트패킷, 수신데이터패킷은 버스드라이버로 전달하는 HCI명령패킷, 송신데이터패킷과 밀접한 관계가 있지만, 완전히 비동기적으로 전달된다. 따라서, HCI명령패킷과, 송신데이터패킷을 처리하는 프로세서와, HCI이벤트패킷, 수신데이터패킷을 처리하는 프로세서는 별도의 쓰레드로 구현되어야 한다. 이러한 처리를 위해

· 파이프: HCI계층에서 담당해야 할 흐름제어(Flow Control)기능의 구현을 위해, 데이터패킷을 전달하는 프로세스와 전송된 패킷에 대한 결과신호인 "Number Of Completed Packet Event"라는 이벤트를 받아서 적절한 처리를 하는 프로세스가 독립적인 쓰레드로 구현되어야 할뿐만 아니라, 이들 두 쓰레드간의 체계적인 통신 메커니즘이 필요하다[8].

· 다형성(Polymorphism): HCI계층의 구현에서는 상위 계층의 요구에 따라 95가지의 HCI명령에 대한 명령패킷을 구성하는 기능과, 이벤트

패킷으로부터 32가지의 이벤트중 하나로 분석(Decode)하는 기능이 필수적으로 포함된다. 이 기능에 대한 구현은 다양한 형태로 나타날 수 있으나, HCI명령은 AssembleCommand(), HCI 이벤트는 MakeEvent()와 같이 특정의 공통적인 인터페이스로 구현된다면 수많은 HCI명령과 HCI이벤트들을 체계적으로 관리와 확장성을 유지할 수 있다.

· CPU 스케줄: HCI계층의 흐름제어를 위해 구현된 두 쓰레드는 간접적으로 상대방의 트리거(Trigger)를 기다리고 있는 구조이다. 즉, 데이터패킷을 전달하는 쓰레드는 "Number Of Completed Packet Event"라는 이벤트가 처리되어 패킷을 추가로 보낼 수 있는 상태를 기다리고 있고, "Number Of Completed Packet Event"라는 이벤트는 데이터패킷이 전달된 후에 발생하는 것이다. 만약, 이벤트를 처리하는 쓰레드가 "Number Of Completed Packet Event"라는 이벤트를 전달받아 처리할 정도의 CPU의 서비스를 받지 못한다면, 쓰레드들은 교착상태에 빠지게 된다. 흐름제어를 위하여 동작할 코드는 CPU의 서비스를 고르게 받을 수 있도록 단위 쓰레드를 설계해야 한다[9].

3.2 HCI 클래스 계층구조의 설계

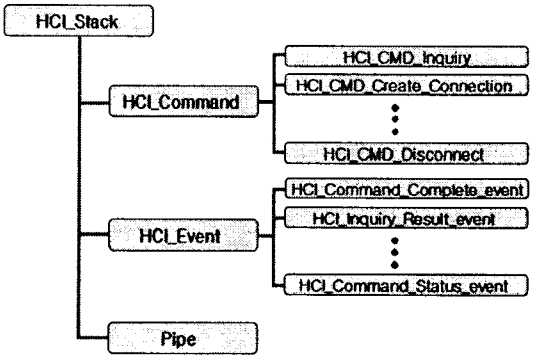
블루투스 표준규격에서 제공하는 기능을 기능성을 기반으로 분할하여 HCI 클래스구조를 설계하였다. Fig. 3과 같이 설계된 HCI의 클래스 구조는 다음과 같다.

· HCI_Command 클래스: Command의 일반적인 형식을 정의한 클래스이며, 하위 클래스로 HCI에 규정된 95개의 Command클래스 정의와 멤버함수로 AssembleCommand함수를 등록하여 각 Command패킷들의 인스턴스를 구현할 수 있도록 한다.

· HCI_Event 클래스: 블루투스 장치로부터 발생되는 Event코드의 일반적인 형식을 정의한 클래스이다. 하위 클래스로 HCI에 규정된 32개의 Event클래스 정의와 멤버함수로 MakeEvent 함수를 등록하여 각 Event패킷들의 인스턴스를 구현할 수 있도록 한다.

· Pipe 클래스: 호스트와 블루투스 장치간의 Command 전송과 데이터 송·수신을 위한 쓰레드의 유기적인 동작과 흐름제어를 위해 사용되는 IPC(Inter-Process Communication) 메커니즘 중의 하나인 파이프(Pipe)를 구현한 클래스이다.

· HCI_Stack 클래스: 위에서 정의된 클래스들을 기반으로 하여, Command실행에 따라 발생한 다양한 Event에 대해서 콜백함수를 등록하여 호스트가 Event응답을 받을 수 있게 했고, 흐름제어를 통해서 데이터 패킷을 송·수신하도록 구현할 수 있도록 한다.

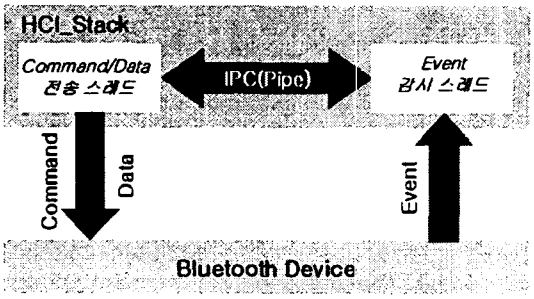


[Fig. 3] Class Architecture of HCI

3.3 Event기반의 쓰레드 동작 메커니즘

블루투스 프로토콜 표준규약에 따르면 Com-

mand의 실행에 대한 결과로 다양한 Event가 비동기적으로 발생한다는 것을 알 수 있다. 따라서, 발생되는 Event들에 대한 비동기적인 처리 메커니즘이 필수적이다. 이 논문에서는 비동기적으로 발생되는 다양한 Event들을 처리하기 위해 Fig. 4와 같은 Event기반의 멀티쓰레드 동작 메커니즘을 제안하였다[3][4].



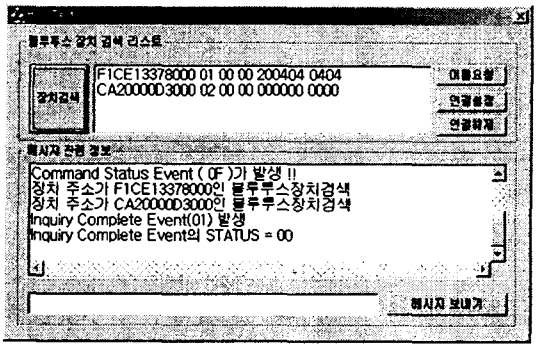
〔 Fig. 4 〕 The Operation of Multi-Thead Mechanism based on Event

이 메커니즘은 다중 쓰레드 환경에서 발생할 수 있는 쓰레드 동기화에 관련된 문제를 해결하기 위한 것으로, Event를 감시하는 쓰레드는 블루투스 장치로부터 Event가 발생했을 경우 이에 대한 정보를 파이프 객체를 이용해서 Command를 전송하는 쓰레드에게 다음 Command를 실행할 수 있도록 통지하는 역할을 한다.

호스트에서 블루투스 장치로 전달하는 데이터 패킷은 블루투스 장치의 데이터 버퍼가 넘치지 않도록 보내야 한다. 이와 같은 흐름제어를 위하여, 데이터 패킷을 블루투스 장치로 전달하는 쓰레드와 Number _Of_Completed_Packet_Event를 전달받는 쓰레드를 각각 독립적으로 동작하도록 구현하였으며, 이들 간의 정보전달을 위하여 파이프 객체를 사용하였다.

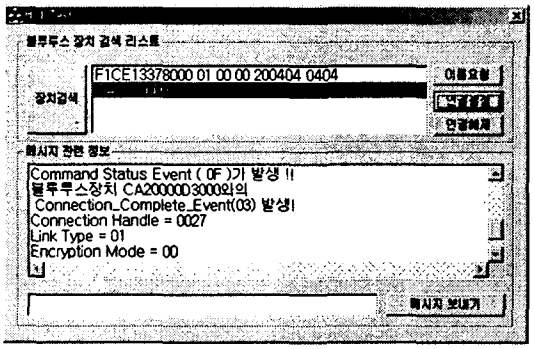
Ⅳ. 실험 및 고찰

HCI의 구현은 Windows2000을 기반으로 하였으며, 실험을 위해 사용된 장치는 MMC Technology 사의 블루투스 어뎁터(Model: MB201 UC1)와 Ericsson사의 블루투스어뎁터(Model: BMK003) 및 Widcomm사의 블루투스 어뎁터(BMW-DK)를 혼용하여 통신을 수행하였고, USB드라이버와 통신하기 위해 CSR (Cambridge Silicon Radio)에서 제공되는 BlueCore01 USB Stack 드라이버 모듈을 사용했다. 다음에 나오는 Fig. 5, Fig. 6과 Fig. 7은 HCI의 동작을 검증하기 위해 설계된 HCI 클래스 구조를 사용하여 개발된 HCI_Test 프로그램으로 실험한 일련의 결과들을 보여 주고 있다.



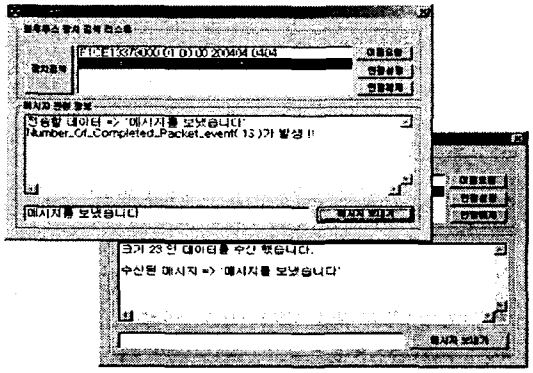
〔 Fig. 5 〕 The Experiment for Bluetooth Device Search

HCI_Test 프로그램을 실행하여 블루투스 장치를 검색한 화면은 Fig. 5와 같다. 장치검색 버튼을 누르면 상단 리스트 박스에 검색된 장치들의 주소와 세부적인 속성들이 순서대로 나오고, 중간의 메시지 박스에는 검색된 블루투스 장치들의 주소와 발생한 Event코드 정보가 성공적으로 표시되는 것을 확인할 수 있었다.



〔 Fig. 6 〕 The Experiment for Connection between Bluetooth Devices

Fig. 6은 검색된 블루투스 장치리스트 중 원하는 블루투스 장치와 연결을 설정한 화면이다. 리스트 박스에서 장치를 선택하고 연결설정 버튼을 누르면 선택된 블루투스 장치와의 연결이 이루어지고 연결된 장치의 주소와 발생한 Event 코드 정보가 메시지 박스에 성공적으로 표시되었다.



〔 Fig. 7 〕 The Experiment for Data Transfer between Bluetooth Devices

Fig. 7은 연결이 설정된 장치들 사이에서 데이터를 전송한 화면이다. 연결 설정 후 메시지를 입력하고 메시지 보내기 버튼을 누르면, 연결하고 있는 다른 블루투스 장치로 데이터가 전송된다. 수신측 호스트는 보내온 데이터 패킷의 순서와 크기에 맞게 패킷을 받게 되고, 전송 측 호스트는 전송이 확인된 데이터 패킷마다 성공적인 Event가 발생한다. 이것으로 두 장치간의 데이터통신에 흐름제어가 이루어지는 것을 확인할 수 있었다.

V. 결론

블루투스는 차세대 근거리 무선통신 기술로 각광받고 있으며, 이것은 프로토콜 스택의 개발을 통한 다양한 서비스제공을 필요로 하게 된다. 블루투스 표준규격은 무선분야부터 응용분야까지 전 시스템이 공개구조를 채택하고 있다. 그러나, 블루투스 프로토콜 스택은 개발자에 따라 다른 방법으로 하드웨어와 소프트웨어의 역할을 기능성을 중심으로 독자적으로 분할하여 설계 구현하도록 한다. 본 연구는 블루투스 프로토콜을 획득하기 위한 첫 단계 연구과제로서 상위계층 프로토콜 스택 개발을 위한 HCI를 설계구현 하였고, 표준 규약에 따르는 HCI의 동작과정을 검증하기 위한 일련의 실험들을 수행하였다. HCI를 구현하기 위하여 블루투스 표준규약을 분석하여 HCI 클래스 구조를 설계하였고, 설계된 HCI 클래스 구조를 블루투스 운영플랫폼 상에서 구현하기 위한 개발 환경의 고려 사항을 확인하였다. 구현된 HCI 클래스 구조를 사용하여 일련의 실험을 수행하였다. 그 결과 HCI 계층의 주요과업인 블루투스 장치검색, 연결설정 및 데이터 전송이 모두 성공적으로 수행되는 것을 확인하였다. 또한 기 개발된 타사의 어댑터와의 통신 실험을 통하여 구현된 HCI 클래스 구조의 범용성과 호환성에 대한 가능성도 검증

하였다. 본 연구실에서는 블루투스 플로토콜 스택의 독자적인 개발을 위하여 획득된 HCI 클래스 구조를 기반으로 L2CAP, 및 SDP의 설계 및 구현을 위해 계속 연구하고 있다. 끝으로 본 과제는 (주) 진두네트워크의 연구과제의 일환으로 수행되었으며. 본 연구 및 실험에 참가한 대원과학대학의 이상윤 교수와 현무용 교수, 그리고 본 대학의 분산연구실 대학원생 여러분에게 감사를 드린다.

참고문헌

[1] Zenocom, Zenocom Document Website, <http://www.zenocom.co.kr>.

[2] 3Com, 3Com Document Website, <http://www.3com.co.kr/mobile/wireless>.

[3] Bluetooth SIG, The Bluetooth Specification v1.1, <http://www.bluetooth.com>.

[4] CSR, BlueCore01 USBStack Document <http://www.csr.com>.

[5] Jennifer Bray and Charles F. Sturman, "Bluetooth Connect Without Cables", Prentice Hall, 2001.

[6] James Y. Wilson and Jason A. Kronz, "Inside Bluetooth Part I", Dr. Dobb's Journal, 2000.

[7] Jaap C, Haartsen, "BLUETOOTH™ : A new radio interface providing ubiquitous connectivity", IEEE VTC2000, 2000, p107-111.

[8] RON SCHNEIDERMAN, "Bluetoth's slow dawn", IEEE SPECTRUM November 2000, pp 61-65.

[9] Zhang Pei, "Bluetooth - The Fastest Developing Wireless Technology", IEEE VTC2000, 2000, pp 1657-1664.

Design and Implementation of Class Structure for Bluetooth HCI Layer

Shik Kim* · Sooh-Yung Liu**

Abstract

The Bluetooth is expected to be one of the most popular wireless telecommunication technology in the near future, and the protocol stack is essential to providing the various services with the Bluetooth-embedding systems or devices. The Bluetooth specification is an open, global specification defining the complete system, however, the protocol stack is usually implemented partly in hardware and partly as software running on its system, with different implementations partitioning the functionality between hardware and software in different ways. I investigate how to design and implement the Bluetooth protocol stack according to its specification. I focus on the HCI and the lower layer of the software protocol stack as a basic step for the development of our own protocol stack. As a result, paper provides how to partitioning the role of HCI layer, and how to implement the relationship between HCI packets, it's functionality and the flow control. Experiments show the discovering other Bluetooth devices and their connection. Furthermore experiments demonstrate the proper operation of data communication between the Bluetooth modules.

* Dept. of Information Communication, Semyung University
** Dept. of Computer Information Science, Semyung University