ANDROID 기반의 USB 통신을 이용한 SMART PHONE MOUSE

PROJECT PROPOSAL DOC.

Lim, Min-woo (200910793)

Lee, Young-jun (200911412)

Park, Mi-gwan (200911388)

Jin, Hyun-wook
Professor
System Software Lab, Konkuk Univesity

목차 (CONTENTS)

내용

개요 (Introduction) _____ 1

프로젝트 환경 (Project Environment)_____ 2

요구사항 (Requirement) _____ 3

개발 방법 (Development Process) _____ 4

개발 일정 (Development Schedule) _____ 10

참고문헌 (References)_____ 11

개요 (INTRODUCTION)

개요 (Introduction)

개발 배경

노트북 터치패드 사용이 불편한 사람들은 따로 마우스를 구매하여 소지하고 다녀야 한다. 이런 불편함을 해소하기 위해 대부분의 사람들이 들고 다니는 스마트폰의 센서를 이용하여 마우스로 구현한다.

개발 목적

노트북의 터치패드 사용에 한계를 느끼는 사용자는 마우스를 별도로 구매하여야 하며, 이동시에 항상 노트북과 함께 마우스를 들고 다녀야 한다. 대부분의 사람들이 항상 소지하고 다니는 스마트폰이 마우스로 사용될 수 있다면, 이와 같은 불편함을 해소시켜 줄 것이다.
기존의 스마트폰 마우스 서비스는 노트북 터치패드 사용방식의 불편함을 해소하는 것이 아닌, 스마트폰의 무선통신을 이용한 노트북 원거리 제어에 초점이 맞춰져 있다. 또한 모두 추가적인 어플리케이션의 설치를 요구한다.
본 프로젝트는 어플리케이션 설치가 아닌 Kernel과 Framework에 새로운 메커니즘을 추가함으로서 안드로이드 스마트폰의 기본 탑재 기능과 같이 동작하도록 하며, 터치패드가 아닌 실제 마우스와 같은 사용방식을 제공한다. 그리고 기존의 USB 케이블 연결을 통한 USB 통신을 사용함으로서 데이터 통신이 되지 않는 환경에서도 작동한다.

<그림 > 별도의 마우산 App 이 필요없는 스마트폰의 기본 기능

프로젝트 환경 (PROJECT ENVIRONMENT)

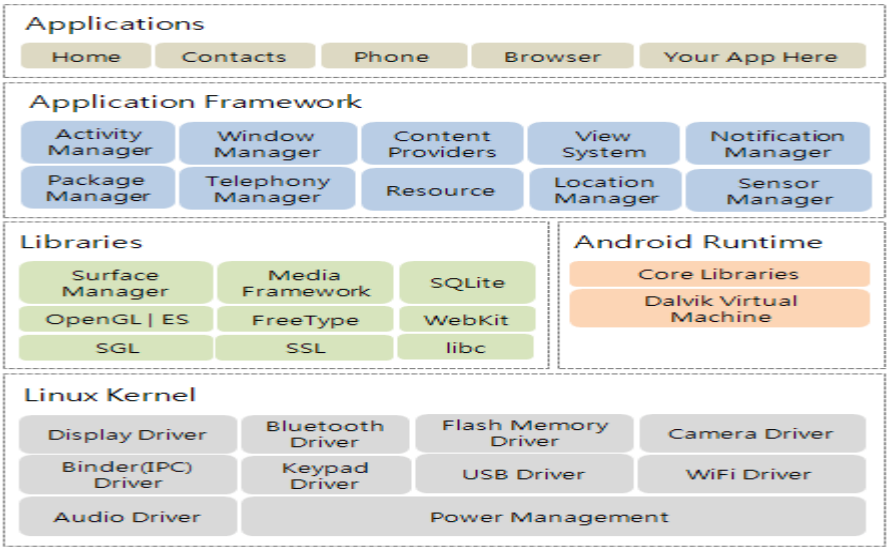
프로젝트 환경 (Project Environment)

대상 환경 (TARGET ENVIRONMENT)

Windows 7 Professional 32bit

Android 4.4.2 KitKat (Nexus 5)

Linux Kernel 3.4.0



<그림 > Android 시스템 구조

개발 환경 (HOST ENVIRONMENT)

Windows 7 Professional 32bit

Android 4.4.2 KitKat (Nexus 5)

Linux Kernel 3.4.0

Visual Studio Professional 2013

Vi(m) Editor

ADT 22.3.0

WDK 8.1

요구사항 (REQUIREMENT)

요구사항 (Requirement)

기술적 요구사항 (FUNCTIONAL REQUIREMENTS)

1. Windows Device Driver 를 개발하여 스마트폰 연결시 자동 실행 한다.
2. Windows 와 Android 간의 통신은 USB 통신을 한다.
3. Android Kernel 에 마우스 통신을 위한 별도의 파이프를 생성한다.
4. 스마트폰에서 마우스 기능을 On/Off 할 수 있다.
5. 스마트폰 화면이 꺼져도 마우스 기능이 켜져있으면 마우스가 동작한다.

비기술적 요구사항 (NON-FUNCTIONAL REQUIREMENTS)

1. 사용자는 어떠한 안드로이드 어플리케이션을 설치하지 않는다.
2. 사용자는 어떠한 윈도우 어플리케이션을 설치하지 않는다.
3. 스마트폰 사용에는 어떠한 영향도 미치지 않는다.

개발 방법 (DEVELOPMENT PROCESS)

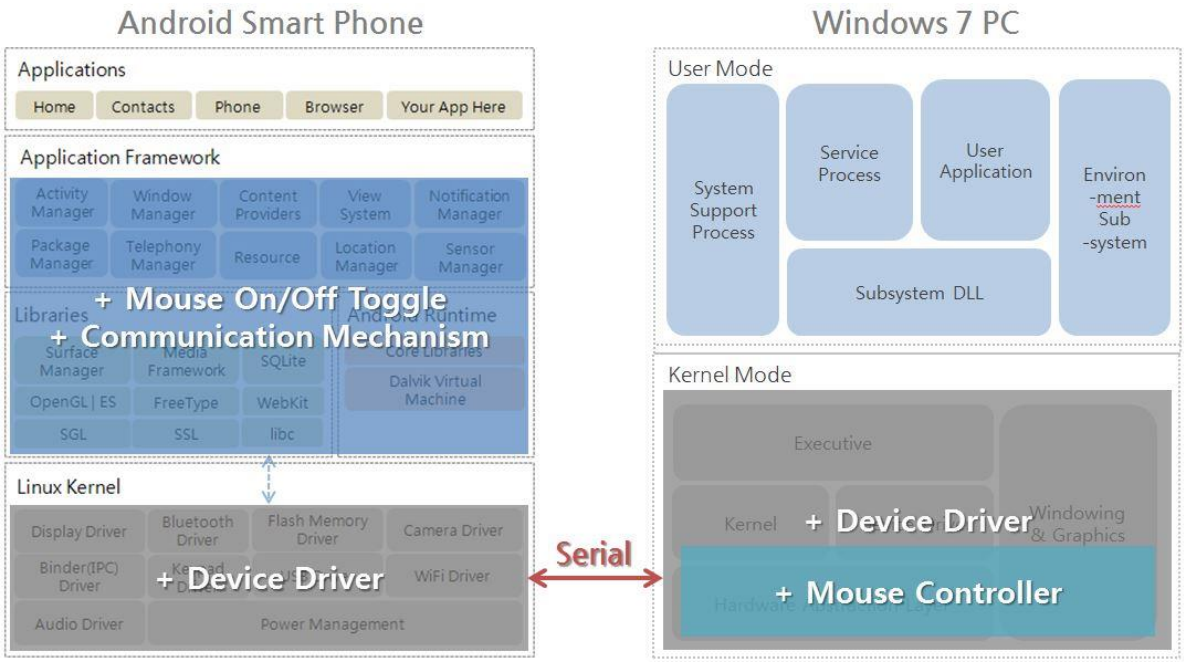
개발 방법 (Development Process)

개발은 크게 Windows 와 Android 로 나누어서 진행한다.

Windows 에서는 Device Driver 를 만들어 스마트폰 연결시 마우스 기능이 자동 실행되게 한다.

Android 는 가속도와 자이로스코프 센서를 이용하여 마우스 값을 축출 및 필터링하고, USB 파이프를 이용하여 Windows 로 마우스 값을 전송한다.

시스템 구성도



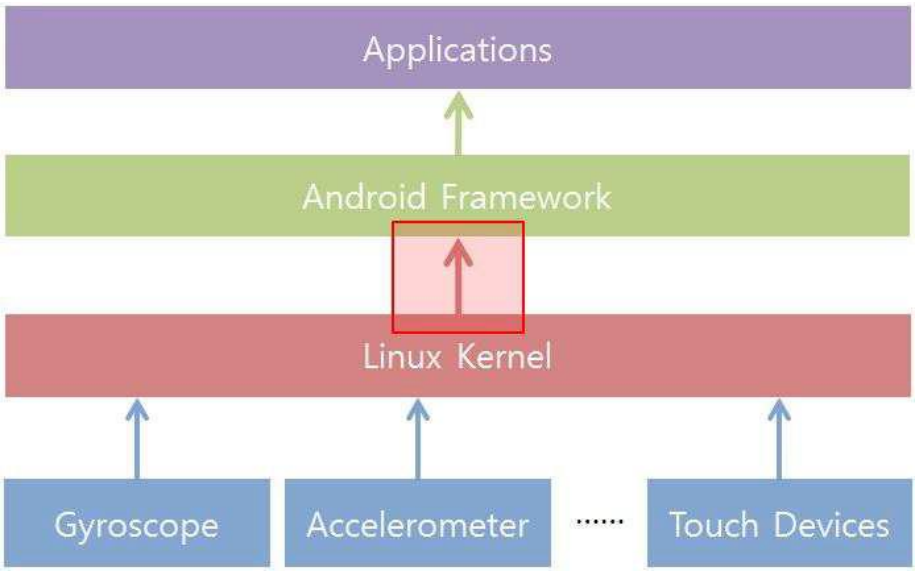
<그림 > 시스템 구성도

개발 방법 (DEVELOPMENT PROCESS)

마우스 동작 구현

- 센서 값 축출

대부분의 스마트폰에 기본적으로 탑재되어 있는 자이로스코프 센서와 가속도 센서 그리고 터치 패널을 이용해서 마우스의 동작을 구현한다. 자이로스코프 센서와 가속도 센서를 이용해서는 마우스 포인터의 움직임을 구현하며 터치 패널을 이용해서 좌우버튼과 휠의 동작을 구현한다.



<그림 > 안드로이드 스마트폰에서 발생하는 각종 센서 값의 흐름

안드로이드 스마트폰의 모든 센서 장치들은 최하단 리눅스 커널에서 관리되며, 디바이스들로부터 발생되는 값들은 최상단 사용자 어플리케이션 계층까지 한 단계씩 상승한다.

본 프로젝트에서는 사용자 입장에서의 기존 스마트폰 사용성에 어떠한 영향도 없이, 그리고 사용자의 어떠한 부가적인 어플리케이션 설치 요구도 없이 마우스 기능이 동작하는 것을 목표로 한다. 그러므로 마우스 동작에 필요한 센서값들은 최하단 계층에서 바로 추출하는 것을 기본으로 하며, 각종 보호기법으로 인하여 해당 계층에서 추출이 불가능한 상황이 발생하면 한 계층씩 올라가며 추출한다.

- 센서 값 필터링

가속도와 자이로스코프와 같은 센서를 사용할 때, 단순히 전달된 센서 값을 그대로 사용할 경우 급격한 변화에 의해 마우스 포인터의 움직임이 불안정 할 수 있다는 것이다. 이 같은 경우 변화를 완충하기 위해 데이터에 저역 통과 필터(Low-Pass Filter:LPF)를 적용한다. 저역 통과 필터를 사용할 때에는 결과 데이터를 추출하기 위해 사용되는 알파 값이 필요한데 이것은 아래와 같은 연산결과로 산출된다.

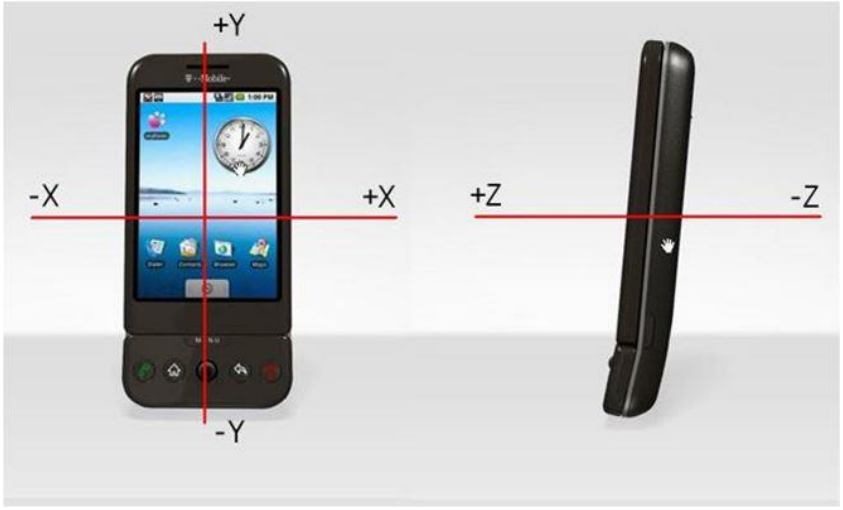
개발 방법 (DEVELOPMENT PROCESS)

Alpha = t / (t + dT)

t : 저속 통과 필터의 시정수

dT : 이벤트 전송율 혹은 이벤트 전송속도

여기서 시정수란 센서가 가속도의 63%를 인지하는데 걸리는 시간을 말한다. 위와 같은 알파 값은 0.8로 산출되며 알파에 직전 센서 값을 곱하고, (1 - Alpha)에 현재 센서 값을 곱하여 더해주면 필터링된 움직임 값을 얻을 수 있다.



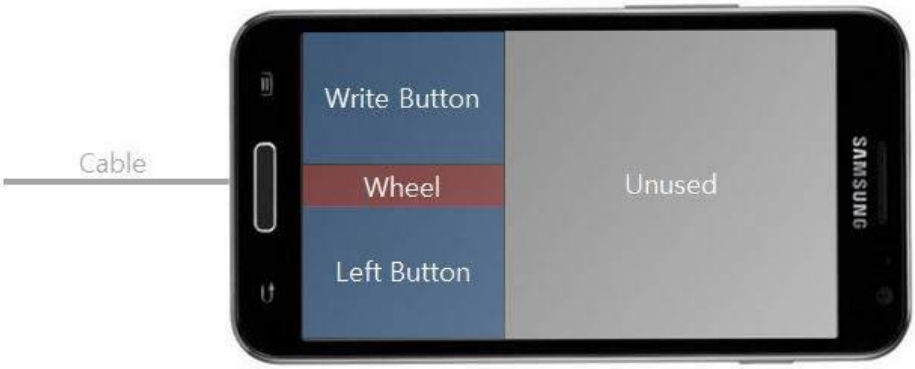
<그림> 센서의 좌표 체계

자이로스코프 센서는 x, y, z 축의 변화를 추적하여 안드로이드 스마트폰의 회전률을 정확하게 알아낼 수 있다. 그리고 가속도 센서는 특정 지점에서 특정 지점으로 이동하는 벡터형태를 인지할 수 있지만, 곡선이나 트위스트 형태의 움직임은 인지할 수 없다. 정리해보자면 가속도 센서는 직선방향의 움직임을, 자이로스코프 센서는 곡선(회전)을 인식할 수 있다. 그러므로 두 센서를 적절하게 섞어서 사용할 경우 회전을 인지하지 못하는 실제 광센서 마우스보다 더 나은 동작을 구현할 수 있다.

- 센서 값 가공

  가속도 센서는 사물의 움직임을 측정하는데 이 움직임을 이용하여 다른 성질을 가지는 데이터로 가공할 수 있기 때문에 여러 방법으로 연산을 하여 사용하곤 한다. 본 프로젝트에서와 같이 순수한 움직임만을 측정하는 경우에는 중력의 영향을 받고 있는 축의 데이터에서 9.81을 빼주어야 하고, 가속도를 측정하려면 2개 이상의 데이터를 모아서 연산을 해주어야 한다. 실제 마우스 포인터의 움직임은 가속도가 반영되어 같은 거리로 마우스를 움직이더라도 느리게 움직일 때와 빠르게 움직일 때 마우스 포인터의 이동거리는 다르다. 이와 같은 부분은 반복적으로 테스트하며 가공 알고리즘을 작성하여 실제 마우스와 동일한 움직임을 보이도록 한다. 이와 같은 특 징은 휠의 움직임에서도 동일하게 적용된다. 휠의 회전량과 회전속도를 함께 고려하여 실제

마우스와 동일한 움직임을 보이도록 한다.



<그림> 터치 패널의 분할

사용자는 터치패널을 실제 마우스를 사용할 때와 같은 느낌으로 사용한다. 여러 사용자들의 사용습관을 테스트하여 적절한 좌우버튼 및 휠 그리고 비사용 범위를 지정한다.

위와 같은 과정을 거쳐, PC-스마트폰 통신모듈에게 실제 마우스와 같이 좌우버튼의 눌림 여부, 휠의 움직임 그리고 마우스 포인터의 움직임에 대한 완전히 가공된 값을 미리 정해진 규약에 맞게 제공한다.

마우스 ON/OFF 기능

- 안드로이드 토글 버튼

  안드로이드 스마트폰 사용자는 본 프로젝트로 인해 스마트폰에 기본으로 탑재된 듯한 마우스 기능을 상단 바의 토글버튼 중 하나로 끄거나 켤 수 있다. 어떠한 어플리케이션의 설치도 요구되지 않는다.

<그림> 토글 버튼 구현 예시

개발 방법 (DEVELOPMENT PROCESS)

- 센서 Sleep 방지

  기본적으로 안드로이드 스마트폰은 일정시간 사용이 되지 안으면 Sleep 모드에 돌입한다. 이 상태에서는 화면은 물론 모든 센서들이 동작하지 않는 최대 절전 모드가 된다.

  본 프로젝트의 마우스 동작의 실제 구현은 어플리케이션 보다 하단 계층에서 모두 이루어지므로 마우스 기능이 On 상태라 하더라도 터치 이벤트만 발생하지 않으면 Sleep 모드에 돌입 가능하다. 또한 원래 Sleep 모드에 머물고 있었더라도 PC와 케이블이 연결되면 마우스 기능은 곧바로 동작해야 한다.

  이와 같이 동작하기 위해서는 스마트폰이 Sleep 모드에 돌입하더라도 본 프로젝트의 마우스 기능에 사용되는 터치 패널과 자이로스코프 및 가속도 센서는 계속해서 동작해야 한다. 이와 같은 문제는 Sensor Keep Awake 이슈라 불리며, 해결하기 위한 어플리케이션 계층의 API 함수가 존재한다. 본 프로젝트에서 어플리케이션 계층의 이용은 지양하므로 토글 버튼의 On/Off와 곧바로 연동되도록 Framework 계층에서의 해법을 찾는다.

WINDOWS DEVICE DRIVER

- KMDF (Kernel Mode Driver Framework)

  Windows Driver 개발을 위해서는 WDM(Windows Driver Model)이후 버전으로 나온 WDF(Windows Driver Foundation)를 사용하였다. WDF는 윈도우 2000 이후의 윈도우를 위한 고품질의 믿을 수 있는 드라이버를 만드는데 도움을 주는 마이크로 소프트 도구이다. WDF는 KMDF(Kernel Mode Driver Framework)와 UMDF(User Mode Driver Framework) 두 가지 종류로 나뉜다. KMDF는 표준 커널 모드 장치 드라이버 작성용이고, UMDF는 사용자 모드에서 실행할 수 있는 특정한 클래스의 드라이버 작성용이다. 우리 프로젝트에서는 마우스 제어를 하기위해 커널 모드에서 작동하는 KMDF를 채택하였다.

- INF

  INF파일 혹은 Setup Information file은 윈도우의 소프트웨어나 드라이버를 설치하기 위해서 마이크로소프트에서 제공하는 텍스트 파일이다. 드라이버 설치를 위해서 inf파일에 드라이버 경로 및 설치에 필요한 작업등을 나열해 놓는다.

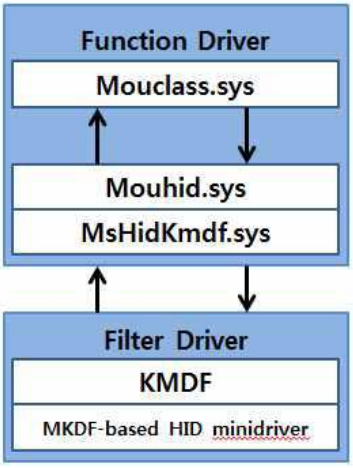
개발 방법 (DEVELOPMENT PROCESS)

```
[Version]
전반적인 기술

[SourceDisksNames.CPU]
셋업 환경 디스크명을 설정

[SourceDisksFiles.CPU]
이 INF로 셋업할 파일(드라이버 파일)을 지정

[DestinationDirs]
파일(드라이버 파일)을 어디에 복사할지 지정

[ManuFacture]
디바이스의 제조사명을 지정. 실제로는 모델 섹션명도 지정하고
이것을 기준으로 다양한 섹션으로 분기함

"제조사명 문자열" = 모델 섹션명

[모델 섹션명]
모델명을 지정하고 설치 섹션명을 식별자와 함께 지정

"모델명 문자열" = 설치 섹션명, 식별자(,옵션)

[설치 섹션명.환경CPU]
CopyFiles = 복사파일 섹션

[설치 섹션명.환경CPU.Services]
AddService = 서비스명, 0x00000002, 서비스 섹션
```

<그림> INF 파일 구조

- HID Midi Driver

  본 프로젝트에서는 스마트 폰을 마우스로 인식하여야 한다. 그리고 실제 마우스 값이 아닌 패킷을 통해 마우스를 움직여야 하므로 실제 마우스 드라이버를 만들면 데이터를 받을 수가 없어서 제대로된 동작을 하지 않게 된다. 그리하여 KMDF를 기반의 HID Mini Driver를 개발하였다. HID Mini Driver는 Filter Driver로서 HID 디바이스의 드라이버 스택에 연결된다. 그리하여 USB 통신을 사용 가능하고, HID Class Driver와 통신 가능하다.



<그림> 드라이버 스택 구조

개발 일정 (DEVELOPMENT SCHEDULE)

개발 일정 (Development Schedule)

이영준	6	7	8	9	10	11
Driver 동작을 위한 INF 파일 작성						
USB 객체 및 Pipe 정보 획득 기능 구현						
Continuous Reader 구현						
Mouse 디스크립터 작성						
HID Mouse MiniDriver 구현						
Framework - Mouse 모드 On/Off						
Framework - 휴면상태 센서 수신 유지 구현						
전체 통합 테스트 및 디버깅						

임민우	6	7	8	9	10	11
Kernel - Linux Device Driver 생성						
터치/가속도/자이로 센서 값 추출						
클릭/휠 동작 위한 센서 값 가공						
센서 값 Filtering						
Framework - Mouse 패키지 Embed						
Framework - 입력 이벤트 후킹 구현						
전체 통합 테스트 및 디버깅						

<그림 > 작업일정

참고문헌 (REFERENCES)

참고문헌 (References)

윈도우 USB 디바이스 드라이버(2011), 하마다 켄이치로, 에이콘 출판사

IT EXPERT 리눅스 디바이스 드라이버(2005), 유영창, 한빛미디어

리눅스 커널 심층 분석(2012), 로버트 로브, 에이콘 출판사

안드로이드의 모든 것 분석과 포팅(2011), 고현철, 유형목, 한빛미디어