http://dx.doi.org/10.7236/JIWIT.2012.12.3.163
JIWIT 2012-3-21

듀얼 스크린 안드로이드 플랫폼 프로토타입의 설계 및 구현

A Design and Implementation of Prototype of Dual Screen Platform on Android

황기태*, 조혜경**

Kitae Hwang, Hye-Kyung Cho

요 약 안드로이드 플랫폼에서는 하나의 앱이 스크린 장치를 독점하기 때문에, 사용자는 동시에 2 개의 앱을 실행하여 각 앱이 실행되는 화면을 같이 볼 수 없다. 본 논문은 두 개의 안드로이드 단말기에서 쌍으로 실행될 필요가 있는 앱을 듀얼앱으로 정의하고, 이들을 각 단말기에서 실행 시키고 통신할 수 있는 듀얼 스크린 안드로이드 플랫폼, DSAP을 설계하고 구현한 내용을 기술한다. 하나의 앱이 실행되면 상대편 단말기에 피어 앱을 원격 실행시키고 이들이 서로 통신하면서 두 개의 스크린을 가진 하나의 앱인 것처럼 작동하게 한다. 본 논문에서는 DSAP의 프로토타입을 설계 구현한 내용과 DSAP의 응용 사례를 소개한다.

Abstract Since only one application has monopoly on LCD device of the mobile device in Android platform, the user can not see two screens together displayed by two applications running simultaneously. In this paper, a dual-app has been defined as two mobile applications running on each device of a pair of two mobile devices and DSAP or Dual Screen Android Platform has been implemented. DSAP does remote-execution of the peer application of a dual-app on the peer mobile device when either application of the dual-app starts to run and supports communication of the pair app over the network. This paper describes details of design and implementation of DSAP and shows a sample case utilizing DSAP.

Key Words : 안드로이드, 듀얼 스크린, 모바일 장치, 원격 실행

Ⅰ. 서 론

안드로이드(Android)[1]는 리눅스 운영체제에 위에 독자적 자바 플랫폼을 올린 모바일 단말기의 운영체제 중의 하나로서 태블릿, 스마트 폰 등의 모바일 단말기에 탑재되고 있으나 스마트 TV 등 다양한 영역으로 확산되고 있다[2,3]

안드로이드 플랫폼은 현재 단말기 스크린을 사용하는데 몇 가지 제약 사항을 가지고 있다. 모바일 단말기에 하나 이상의 스크린을 달 수 없으며 멀티태스킹을 지원하지만 전경프로세스(foreground process)로 실행되는 앱이 스크린 전체를 독점하기 때문에 실행중인 다른 응용프로그램과 스크린을 공유하지 못한다.

이러한 제약은 사용자로 하여금 각각 서로 다른 GUI(Graphic User Interface)를 가진 두 개의 앱을 동시에 실행시켜서 두 결과 화면을 동시에 보는 작업을 근본적으로 할 수 없게 한다. 예를 들어 사용자가 동영상 재생 응용프로그램을 실행하여 동영상 강의를 시청하면서,

*종신회원, 한성대학교 컴퓨터공학과
**정회원, 한성대학교 정보통신공학과
접수일자 2012.5.8, 수정완료 2012.6.4
게재확정일자 2012.6.8

Received: 8 May, 2012, Revised: 4 June, 2012,
Accepted: 8 June, 2012
*Corresponding Author: calafk@hansung.ac.kr
Dept. of Computer Engineering, Hansung University, Seoul, Korea

동시에 e-book reader를 이용하여 전자책 교재를 보고자 하는 경우, 웹 서핑을 하는 동안 전자 사전을 동시에 보는 경우, 사진을 찍기 위해 카메라 앱을 실행하고 다른 쪽에서는 찍은 사진을 보거나 수정하는 경우, 신문을 보면서 동영상으로 제작된 뉴스를 보는 등 많은 경우가 있다. 그림 1은 이런 사례를 보여준다.

한 스크린에서는 동영상 강의를 시청하면서 다른 스크린에서는 이 강좌와 연결된 전자 교재를 본다. 전자 교재를 선택하면 이에 연결된 동영상이 자동으로 다른 스크린에 나타나며 이들은 서로 동기화되면서 작동한다.

최근 들어 듀얼 스크린을 가진 데스크톱 PC나 태블릿 PC 등 기존의 태블릿과 차별화하는 시도가 속속 등장하고 있다. 2010년 CES 2010에서 MSI는 윈도우 7를 기반으로 듀얼 스크린을 가진 넷북 PC의 프로토타입을 개발하였으며[4], 도시바는 Libretto라는 이름의 듀얼 스크린 태블릿 PC를 개발하였다[5]. 이러한 경향은 안드로이드 단말기에도 영향을 미쳐서, 2011년 1월 미국 라스베이거스에서 열린 CES 쇼에서 NEC는 LifeTouch 라는 이름의 7인치 LCD를 두 개 탑재한 듀얼 스크린 안드로이드 시스템의 프로토타입을 선보였으며[6], 2011년 2월에는 급기야 일본의 Kyocera에서 만든 듀얼 스크린 안드로이드 폰인 Echo[7]를 Sprint에서 최초로 판매하기 시작하였다.

대한민국 표준의 모바일 플랫폼인 위피(WIPI)는 하나의 모바일 단말기가 두 개의 LCD를 가지는 것을 지원한다. 모바일 단말기에 탑재된 LCD의 크기가 작기 때문에 두 개의 LCD 장치를 달아서 사용자에게 보다 편리한 사용자 인터페이스를 제공하고자 하는 개발 업체의 시도가 항상 존재하며 이러한 경향은 여전이 아직까지 하나의 LCD 스크린 밖에 지원하지 않는 안드로이드 모바일 단말기를 개선하고자 하는 방향으로 나타나고 있다.

본 논문에서는 안드로이드 커널이나 플랫폼을 수정하지 않고 두 개의 안드로이드 단말기를 연결하여 서로 연관된 두 개의 앱이 각 단말기 상에서 자동으로 작동되게 하는 플랫폼 기술인 DSAP(Dual Screen Android Platform)을 제안하고 설계 구현하였다. 본 논문에서는 서로 연관된 두 개의 앱을 듀얼앱(Dual-App)이라고 부른다. DSAP는 하나의 앱이 단말기에서 실행되면 자신의 짝이 되는 앱을 피어 단말기에서 자동으로 실행시키는 원격 실행(remote execution) 기술을 이용한다.

본 논문의 구성은 다음과 같다. 2장에서는 본 논문과 관련된 연구동향과 배경 지식을 기술하고, 3장에서는 본 논문에서 제시된 듀얼 스크린 플랫폼의 설계 및 구현 내용을, 4장에서는 듀얼 스크린 플랫폼을 탑재한 응용 사례와 시연 결과를 보이고 5장에서 결론을 맺는다.

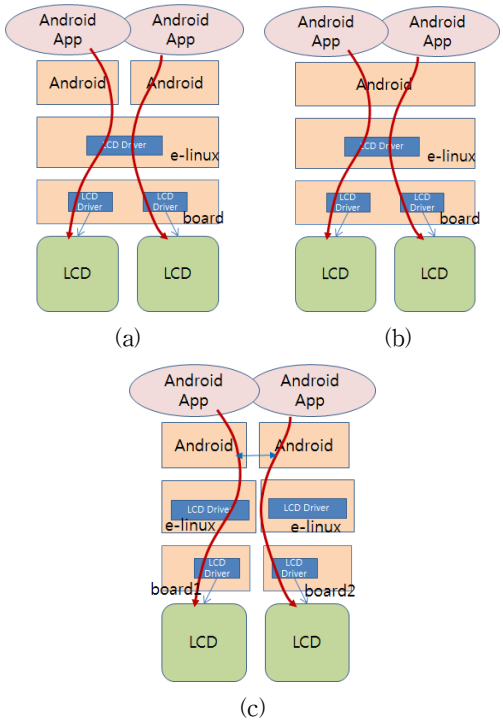
그림 1 듀얼 스크린을 가진 안드로이드 장치 사례
Fig. 1. A Sample of Android Device with Dual Screen

II. 듀얼 스크린을 가진 안드로이드

듀얼 스크린을 가지고 두 개의 앱이 동시에 자신의 스크린에 독점적으로 사용할 수 있는 듀얼 안드로이드 시스템은 여러 가지 모델로 구성해볼 수 있다. 저자의 선행연구[8]의 결과, 듀얼 스크린을 가진 안드로이드 시스템은 그림 2와 같이 (a) 서로 다른 두 개의 안드로이드 단말기에서 실행될 수도 있고, (b) 두 개의 스크린을 가지도록 개발된 안드로이드 단말기가 될 수도 있고 혹은 (c)두 개의 안드로이드 커널이 탑재되고 두 개의 스크린을 가진 하나의 모바일 단말기에 두 개의 안드로이드 운영체제가 탑재되어 두 개의 안드로이드 앱이 각각 별도의 안드로이드 운영체제에서 실행될 수도 있다.

이들은 각각 장단점이 있다. 그림 2(a) 모델은 두 개의 LCD를 달 수 있도록 안드로이드 커널을 수정하는 경우로서, 하나의 리눅스 위에 두 개의 안드로이드를 탑재하였을 때 그 성능을 감당하기 쉽지 않으며 또한 두 개의 안드로이드를 리눅스에 올리기도 쉽지 않다. 또한 두 개의 안드로이드 듀얼앱을 작동시키기 위한 기술이 필요하다. 그림 2(b) 모델의 경우 리눅스에 두 개의 LCD를 탑재하는 것은 쉽지만, 안드로이드가 두 개의 LCD를 기본적으로 다루지 않기 때문에 안드로이드 커널을 고쳐야 하는 어려운 문제가 있다. 또한 여전히 두 안드로이드 앱이

듀얼앱으로 작동시키기 위한 기술이 필요하다. 그림 2(c)의 경우는 두 개의 모바일 단말기 하드웨어를 쌍으로 가진 하나의 단말기 형태이며, 듀얼앱은 내부적으로 통신하는 프로토콜이 필요하다.



(a) 두 개의 안드로이드 커널이 탑재된 장치
(b) 하나의 안드로이드 커널이 탑재된 장치
(c) 서로 연동되어 동기화되는 두 안드로이드 단말기

그림 2. 듀얼 스크린을 가진 안드로이드 시스템 종류
Fig. 2. Android Systems with Dual Screen

이 3 가지 모델을 검토한 결과, 어떤 모델이 되는지 듀얼 스크린을 이용하는 듀얼앱 환경을 만들고자 하면, 듀얼앱이 동시에 실행되면서 통신할 수 있는 구조는 필요하다. 본 논문에서 제안하고 설계 구현한 DSAP는 바로 이런 듀얼앱의 실행 환경을 제공하는 플랫폼이다.

III. 듀얼 스크린 안드로이드 플랫폼(DSAP) 설계

1. DSAP 개요

듀얼앱은 본 논문에서 제시하는 새로운 형태의 앱(Application)이며 하나의 듀얼앱은 두 개의 안드로이드 단말기 상에서 각각 실행되면서 서로 연동하여 작동되는 새로운 개념의 두 안드로이드 앱이다. 듀얼앱을 구성하는 두 앱은 서로 다른 응용프로그램일 수도 있고 동일한 응용프로그램일 수도 있다.

DSAP은 듀얼앱이 실행될 때 피어(peer) 앱을 원격 실행시키고 두 앱이 통신하도록 돕는 소프트웨어 플랫폼과 라이버러리로 구성된다.

듀얼앱을 구성하는 두 앱은 두 개의 모바일 단말기에 각각 설치되어 있어야 하며, 그 중 한 앱이 먼저 실행을 시작하면 다른 단말기에 설치된 다른 앱이 자동으로 실행되고 이들은 서로 통신하면서 마치 하나의 앱이 두 개의 스크린을 사용하는 것처럼 작동한다.

2. DSAP의 소프트웨어 계층

본 논문에서 설계 구현한 DSAP이 설치된 시스템은 그림 3과 같이 듀얼 스크린 플랫폼(DS-Platform), 듀얼 스크린 라이브러리(DS-Library), 듀얼앱(DS-APP)의 세 개의 소프트웨어 계층으로 구성된다.

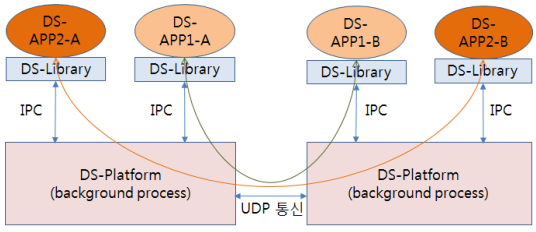


그림 3. DSAP의 3 가지 소프트웨어 계층
Fig. 3. Software Layers of DSAP

DS-APP은 DSAP에서 실행되는 듀얼앱으로서 DS-Library를 이용하여 작성한다. DS-APP은 DS-APP-A와 DS-APP-B의 두 앱으로 구성되어 이들은 항상 짝으로 동시에 작동되며 두 개의 안드로이드 단말기에 각각 설치된다. 경우에 따라 DS-APP-A와 DS-APP-B는 동일한 앱이 될 수도 있다. 듀얼앱은 DS-Lib를 이용하여 작성되고 두 개의 앱으로 작성되기 때문에 이들은 각 단말기에서 실행된다.

DS-Platform은 DSAP의 네트워크 의존적 영역(Network-Dependent Layer) 이다. 초기에 DS-APP-A와 DS-APP-B의 두 앱을 각 모바일 단말기에서 실행시키고 네트워크 세션으로 연결하는 기능과 세션 연결 후

두 앱이 데이터를 교환하며 서로 동기화하도록 돕는 계층이다. DS-Platform을 DS-APP과는 다른 백그라운드 프로세스로 실행되도록 설계하였다. 그러므로 DS-APP을 실행하는 프로세스와 DS-Platform을 실행하는 프로세스가 서로 다르다.

DS-Library는 듀얼앱에게 네트워크 독립적인 응용프로그램 인터페이스(API)를 지원하는 계층이다.

DS-Library는 DS-APP의 프로세스가 DS-Paltform과 데이터를 주고 받기 위해서는 내부적으로 안드로이드에서 제공하는 메시지 기반의 IPC(Inter-Process Communication) 방법을 이용한다.

3. DSAP의 상세 구조

본 논문에서 구현하고자 하는 DSAP의 내부 구성을 스레드를 중심으로 간략히 도식화하면 그림 4와 같다.

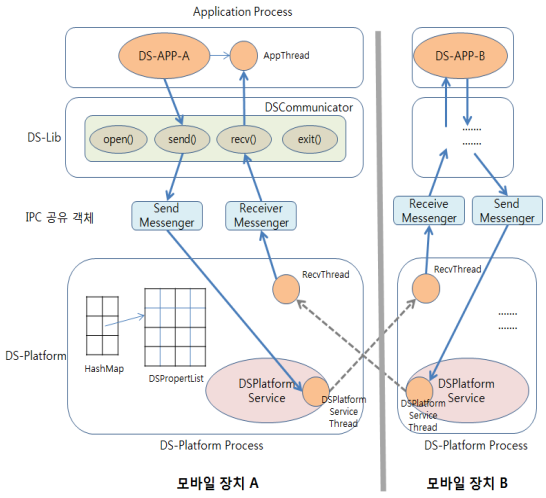


그림 4. 스레드를 중심으로 하는 DSAP의 실행 시간 구조
Fig. 4. Run-time Architecture of DSAP

가. DSPlatformService

각 모바일 장치에 설치되는 DS-Platform의 핵심은 안드로이드 서비스(Service) 컴포넌트로 구현된 DSPlatformService이다. DSPlatformService는 안드로이드 커널에 의해 스케쥴되고 관리되는 백그라운드 프로세스이다. 하나의 모바일 장치에는 하나의 DS-Platform이 설치되며 하나의 DSPlatformService 프로세스가 백그라운드에서 실행된다.

나. 스레드

DSPlatformService는 하나의 듀얼앱 세션 당 1 개의 RecvThread 스레드를 생성한다. RecvThread는 루프를 돌면서 상대편 모바일 단말기로부터 데이터가 도착하면 이를 받아서 듀얼앱으로 데이터를 전달하는 역할을 한다. RecvThread 스레드는 듀얼앱의 실행초기 세션 연결시에 DSPlatformService 프로세스에 의해 생성된다.

DSPlatformService는 독립된 프로세스이므로 메인 스레드를 가진다. 이 메인 스레드는 루프를 돌면서 듀얼앱으로부터 데이터를 받아 상대편 모바일 단말기로 데이터를 전송하는 역할을 수행한다.

다. DSCommunicator

DS-APP은 DS-Lib 계층에서 제공하는 DSCommunicator 인스턴스를 생성하고 이 객체를 활용하여 DS-Platform에 접근한다. DSCommunicator는 피어 앱과 연결하고 데이터를 주고 받기 위해 open(), send(), recv(), exit() 등의 메소드를 지원한다. open() 메소드는 세션 연결시에, send()와 recv()는 듀얼앱 끼리 데이터 송수신 시에, exit() 세션을 종료시킬 때 각각 이용된다.

라. Send Messenger와 Recv Messenger

DS-APP 프로세스와 DSPlatformService 프로세스와 사이에 데이터를 주고 받기 위해 안드로이드에서 제공하는 IPC 메카니즘을 이용한다. 안드로이드는 IPC를 위해 Messenger 객체를 제공한다. 세션 연결 시에 DSPlatformService가 Send Messenger를 생성하여 DSCommunicator로 전달한다. DSCommunicator의 send() 메소드가 Send Messenger를 이용하여 데이터를 보내면, DSPlatformService의 메인 스레드가 이 데이터를 받아 피어 단말기로 전송한다. Recv Messenger는 DSComminicator가 세션 연결시 Send Messenger를 받은 후 생성하여 DSPlatformService에게 전달한다. DSPlatformService는 Recv Messenger에 대한 레퍼런스를 Recv Thread에게 전달하여 피어 단말기로부터 데이터가 도착하면 Recv Messenger로 전송하도록 구성한다.

4. 세션 연결

듀얼앱은 각각 두 개의 모바일 단말기에 미리 설치되어 있어야 한다. 하나의 단말기에서 듀얼앱을 구성하는 한 앱이 먼저 실행을 시작하면 원격 실행을 통해 다른 단

말기에 설치된 앱을 실행시켜서 하나의 세션(Dual App Session)을 구성한다.

세션 연결 과정을 그림 5에 도식화하였으며 구체적으로 다음과 같다.

(1) 듀얼앱을 작동시킬 피어 모바일 단말기의 IP 주소를 알아낸다. 이 과정은 사용자가 직접 피어 모바일 단말기를 조작하여 알아낸다.

(2) DSCommunicator 인스턴스 생성

듀얼앱 중 DS-APP-A를 실행시킨다. DS-APP-A는 DSCommunicator 인스턴스를 생성하고 세션 연결을 위해 open() 메소드를 호출한다.

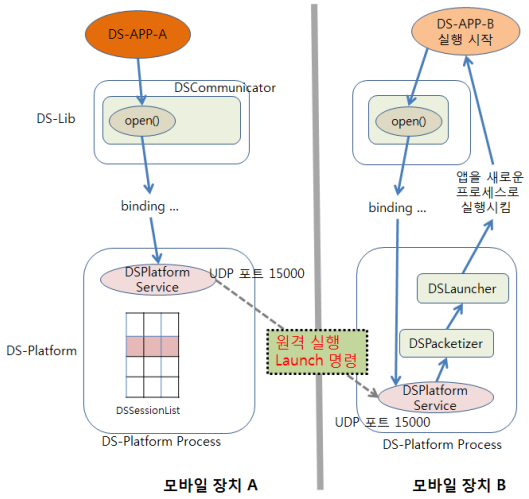


그림 5 원격 실행을 포함하는 세션 연결 과정
Fig. 5. Connection Process of a Dual App Session

(3) DS-Platform에 연결

open() 메소드는 DS-Platform을 구성하는 DSPlatformService에 연결하여 이 서비스와 IPC 통신을 위한 IBinder 객체를 얻어온다. 이때 자신이 통신하고자 하는 상대편 단말기에서 실행될 앱의 패키지명을 전달한다.

(4) 포트 할당과 세션 리스트

DSPlatformService는 현재 연결 중인 세션을 위해 UDP 포트를 하나 할당하고 표 1과 같은 세션 리스트(DSSessionList)에 한 엔트리를 작성한다.

표 1. DSSessionList의 한 엔트리 필드들
Table 1. Fields of an Entry of DSSessionList

필드	설명
PeerIP	상대편 모바일 단말기의 IP 주소
PeerPort	상대편 앱과 통신할 UDP 포트 번호. 두 앱은 UDP 포트 사용
RecvThread	RecvThread의 레퍼런스

(4) 원격 실행 지시

모바일 장치 A의 DSPlatformService는 듀얼앱의 원격 실행을 지시하는 패킷을 만들어 모바일 장치 B로 전송한다. 이 패킷에는 피어 앱의 원격 실행을 지시하는 Launch 명령과 피어 앱의 패키지명, 그리고 듀얼앱 사이에 데이터 전송시에 사용할 UDP 포트의 번호가 들어 있다.

(5) 원격 실행

모바일 장치 B의 DSPlatformService의 메인 스레드는 UDP 15000 번 포트에서 패킷을 받고 DSPacketizer를 이용하여 이 패킷을 분석하고 DSLauncher를 이용하여 DS-APP-B 앱을 실행시킨다. DS-APP-B는 실행을 시작하고 DSCommunicator 생성하고 open() 메소드를 호출하며 자신의 DSPlatformService와 데이터를 주고 받을 Messenger 객체와 스레드를 생성하여 통신할 모든 준비를 마친다.

(6) 세션 연결 완료

모바일 장치 A는 원격 실행 명령을 피어 단말기인 B에 전달한 후 자신의 DSCommunicator와 데이터를 주고 받을 Messenger 객체와 스레드를 생성하여 통신할 모든 준비를 마친다. 이제 세션 연결이 완료된다.

5. 단말기 사이의 통신

두 단말기 사이에는 이루어지는 모든 데이터 통신은 UDP를 이용한다. 통신 데이터의 크기가 비교적 작고 두 단말기 사이의 거리가 멀지 않기 때문에 굳이 실행 시간이 많이 걸리는 TCP를 이용할 필요가 없다.

두 단말기 사이의 통신은 세션 연결 시에 듀얼앱이 피어 앱을 원격 실행하기 위한 통신과 듀얼앱 사이의 데이터 통신으로 구분된다. 표 2와 같이 세션 연결 시에는 UDP 15000 번 포트를 고정적으로 사용하며 듀얼앱 사이

의 통신은 15000번 이후의 임의의 포트를 사용한다. 이 포트 번호는 세션 연결을 시작한 DSPlatformService에서 결정하여 피어에게 전달하는데 15001 번 ~ 15099 번 사이의 번호를 순서대로 순환하면서 할당한다. 세션 연결시에는 DSPlatformService의 메인 스레드끼리 통신하며, 듀얼 앱 사이의 데이터 통신은 DSPlatformService의 메인 스레드와 피어 단말기의 RecvThread 사이에서 이루어진다.

표 2. UDP 통신 포트
Table 2. UDP Ports for Communication Between Peer Apps of a Dual-App

포트 15000	듀얼앱의 세션 연결을 위해 DSPlatformService 사이의 통신 포트
포트 15001-15099	듀얼앱 사이의 데이터 통신

6. 앱의 실행

DS-APP-A를 작성하는 개발자는 따로 AppThread 스레드를 만들 필요가 있다. 이 스레드는 피어 앱으로부터 데이터를 기다리면서 수신하는 역할을 해야 한다. 아니면 이 역할을 DS-APP-A의 메인 스레드가 할 수도 있다.

DS-APP-A가 피어 앱은 DS-APP-B에게 데이터를 보내고자 한다면 DSCommunicator의 send() 메소드를 호출하면 된다. send()는 메시지(Message 객체)를 생성하여 이곳에 보낼 데이터를 담고, Send Messenger의 메소드를 호출하여 데이터를 전송하도록 하고 바로 리턴한다. 스레드의 콘트롤이 실제 네트워크로 데이터를 전송하는데 까지 미치는 것이 아니다. 그러므로 듀얼앱은 빠르게 리턴하여 통신에 따른 시간 지연을 줄일 수 있다. 데이터를 DS-APP-B로 보내는 작업은 DSPlatformService의 메인 스레드의 역할이다.

Ⅳ. 구현 및 응용

1. 패키지 구현

DSAP는 그림 6과 같이 3 개의 패키지로 구현하였다. DS-Lib 계층을 위해 DSCommunicator 클래스를 비롯한 여러 클래스로 구성되는 DSLibrary 패키지를 작성하였으며 jar 파일 형태로 배포되고 DS-APP이나 DSPlatform의 클래스들이 jar의 클래스를 활용한다.

DSPlatform 패키지는 안드로이드에서 바로 실행 가능한 apk 파일로 만들어지고 안드로이드 단말기에 설치되어 단말기가 부팅할 때 백그라운드 프로세스로 실행된다. 이 패키지는 듀얼앱이 파일을 서로 주고 받을 수 있도록 파일 전송 기능도 갖추고 있다.

DSUtility 패키지에는 원격 실행 기법을 통해 상대편 단말기에 설치된 apk 파일을 검색하는 등 사용자 편리를 위한 기능을 구현하였다.

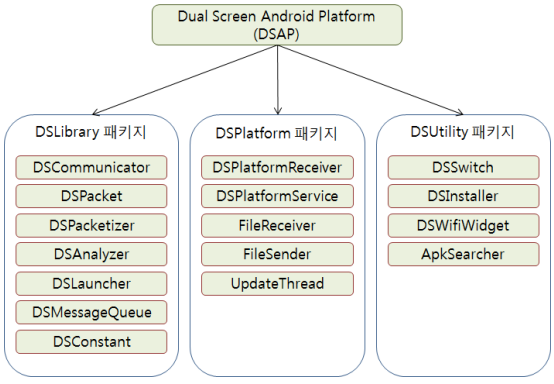


그림 6 원격 실행을 포함하는 세션 연결 과정
Fig. 6. Connection Process of a Dual App Session

2. DSAP를 이용한 학습 응용 사례

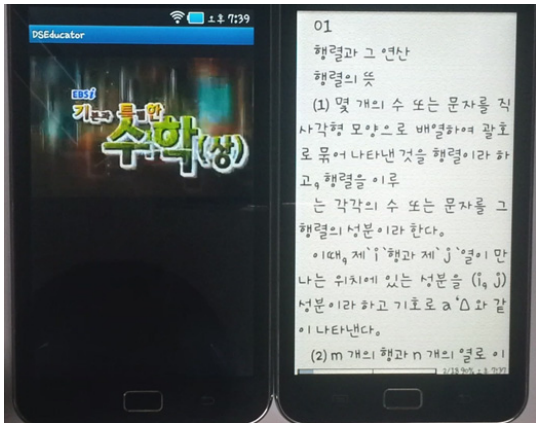


그림 7. 전자책과 동영상 강좌 시청을 동시에 할 수 있는 듀얼 앱의 응용 사례
Fig. 7. A Sample of Dual App

이 절에서는 본 논문에서 설계 구현한 DSAP 프로토타입을 응용하는 사례를 보인다. 그림 7은 전자책 뷰어(e-book reader)와 동영상 강좌 시청을 동시에 이룰 수 있는 듀얼앱, DSMoviePlayer와 DSEducator를 구현한

사례이다. 그림은 두 모바일 단말기의 화면을 캡쳐한 것으로 왼쪽은 한 단말기에서 동영상 강의를 시청하는 DSMoviePlayer가 실행되는 화면이며 오른쪽은 DSEducator이 실행되는 화면이다.

DSEducator는 FBReader라는 전자책 뷰어 오픈 소스를 수정하여 작성하였으며 DSMoviePlayer 앱은 직접 작성하였다.

V. 결론

본 논문에서는 두 개의 안드로이드 장치를 서로 연결하여 마치 하나의 앱이 실행되는 환경을 구성하는 듀얼 스크린 안드로이드 플랫폼 DSAP를 개발한 내용을 기술하였다. DSAP 플랫폼은 듀얼앱을 구성하는 한 앱이 원격 실행을 통해 피어 모바일 단말기에 설치된 피어 앱을 동시에 실행시킴으로써 두 앱이 두 개의 스크린을 사용하는 하나의 세트인 것처럼 작동하게 한다.

DSAP는 안드로이드 플랫폼에서 한 개의 앱이 스크린 전체를 독점적으로 사용함에 따라 두 개의 서로 다른 앱의 UI를 하나의 스크린에 출력할 수 없는 한계점을 극복하며 듀얼테트리스처럼 두 대의 안드로이드 단말기를 상호 연결하여 실행하는 게임이나 모바일 단말기와 스마트 TV 등을 하나의 세트로 연결하여 사용하는 등 다양한 응용에 활용될 수 있다.

참 고 문 헌

[1] “What is Android? - Android,” http://code.google.com/android/whatis-android.html

[2] Min Sik Kim, “Android Tablet PC’s Outlook (evaluation) and the Possibility of Changing Platform Strategy”, Communications of KIISS, Vol. 29 No. 6, June 2011

[3] Chi-Gyu Hwang, “Revival Scenario of Korea IT ′Contents2.0′ ”, Communications of KIISS, Vol. 29 No. 6, June 2011

[4] http://www.engadget.com/2010/01/06/msi-dualscreen-e-reader-hands-on/

[5] http://us.toshiba.com/computers/laptops/libretto

[6] http://www.echobykyocera.com/

[7] http://kr.engadget.com/2011/01/14/nec/

[8] Kitae Hwang, “Conceptual Design of Dual Screen Android System”, Journal of Engineering Research, Hansung University, Vol 9, No. 1. May 2011

※ 본 연구는 한성대학교 교내 연구 장려금의 지원을 받았음.

저자 소개

황 기 태(종신회원)

• 서울대학교 컴퓨터공학과 학사
서울대학교 컴퓨터공학과 석사
서울대학교 컴퓨터공학과 박사
• 경력
University of Callifornia, Irvine 방문 교수
University of Florida 방문 교수
현재 : 한성대학교 컴퓨터공학과 교수
<주관심분야 : 모바일 시스템>

조 혜 경(정회원)

• 서울대학교 제어계측공학과 학사
서울대학교 제어계측공학과 석사
서울대학교 제어계측공학과 박사
• 경력
Carnegie Mellon Univ. 방문연구원
현재 : 한성대학교 정보통신공학과 교수
<주관심분야: HRI/HCI for Learning>