모바일 어플리케이션의 세분화된 전력 소모 측정을 위한 안드로이드 프레임워크의 프로세스와 서비스 분석[1]

김소정, 박재현, 이정원
아주대학교 전자공학과
e-mail : miellonia@ajou.ac.kr, jaehp2560@gmail.com, jungwony@ajou.ac.kr

Analysis of Process and Service of Android Framework for Fragmenting Power Consumption of the Mobile Applications

So-Jung Kim, Jae-Hyeon Park, Jung-Won Lee
Dept. of Electrical and Computer Engineering, Ajou University

요 약

모바일 기기의 발전과 함께 다양한 어플리케이션의 기능이 제공되면서 전력에 대해 효율적인 관리가 필요해졌다. 효율적인 관리를 위하여 모바일 디바이스에서의 전력 소모 원인을 분석해야만 한다. 그러나 기존 연구에서 사용되었던 안드로이드 프레임워크의 수정을 통한 전력 분석 방법은 개발에 대한 용이성이 낮은 단점이 존재하고, 프로파일링을 통한 전력 분석 방법은 각 디바이스 모델에 의존적인 단점이 존재한다. 따라서 본 논문에서는 기존 연구들의 문제점을 개선하기 위해서 안드로이드 프레임워크를 수정하지 않고 획득할 수 있는 프로세스와 서비스의 목록 정보를 통해 어플리케이션이 소모하는 전력에 대한 분석을 제안한다. 본 논문에서 제안하는 방법을 GPS 를 사용하는 어플리케이션을 분석하기 위해서 적용해보았다. 이를 통해 사용자는 측정된 전력과 함께 해당 전력을 사용하고 있는 프로세스 및 서비스 목록을 확인 할 수 있다.

1. 서론

모바일 기기의 보급과 함께 여러 사용자의 요구에 따라, 엔터테인먼트, 게임, 라이프사이클, 음악 등 다양한 기능의 어플리케이션이 제공되고 있다. 어플리케이션 중에는 모바일 기기의 전력을 불필요하게 사용하는 경우가 있다. 이는 배터리 용량이 제한되는 모바일 기기의 사용시간을 감소시켜 사용자의 불편으로 이어지게 된다. 따라서 모바일 기기의 사용시간을 보장하기 위해 전력의 효율적인 관리가 요구 되며, 이를 위해서는 모바일 기기에서 전력을 소모하는 원인 분석이 선행되어야 한다[1].

전력 측정에 대한 연구는 하드웨어측면에서 활발히 진행되고 있다[1]. 그러나 하드웨어에 의해 측정된 값은 모바일 기기 전체 혹은 하드웨어 컴포넌트 단위의 전력 소모량이다. 이는 사용자가 전력을 많이 사용하는 기능을 차단하거나, 모듈 사용을 중지하는 등의 관리만 가능하기 때문에 보다 넓은 범위의 전력 관리를 수행하기에는 한계를 지니고 있다. 전력의 최적화를 사용자가 인지하지 못하는 범위까지 확장하기 위해서는 소프트웨어적인 방법을 통해 어플리케이션에 대한 전력 사용 정보를 얻는 것이 필요하다. 이는 어플리케이션 전력을 분석하는 연구에 기본이 된다.

분석에 대한 연구는 소프트웨어측면에서 여러 연구가 진행되고 있다[2, 3]. 그 중에서 PowerScope 는 프로파일링 방식을 사용하였으나, 프로파일링이 되지 않은 모듈에 대하여는 알 수 없는 문제점이 있다. EnTrack 은 안드로이드 프레임워크를 수정하여 전력을 분석하며, 개발 용이성이 낮은 단점이 있다.

따라서 본 논문에서는 디바이스에 의존적이지 않으면서 안드로이드 프레임워크를 수정하지 않는 방법으로 어플리케이션의 서비스와 프로세스 분석을 통한 세분화된 전력 분석 방법을 제시한다. 본 연구에서 제시하는 방법을 이용하여, 디바이스의 전체적인 전력 소비에 대해 어플리케이션의 서비스와 프로세스 분석을 통해 전력 소비 관계를 알 수 있다.

[1] 본 연구는 미래창조과학부 및 정보통신기술진흥센터의 대학 ICT 연구센터 육성지원사업의 연구결과로 수행되었음 (IITP-2016- H8501-16-1006)

2. 관련연구

배터리 용량이 제한된 모바일 디바이스의 사용시간을 늘리기 위하여 분석 방법에 대한 연구들이 활발하게 진행되었다. 이러한 연구들은 모바일 디바이스의 전력 정보를 프로파일링 하는 방법과 안드로이드 프레임워크를 수정하여 전력 정보에 대해 모니터링하여 분석하는 방법의 연구가 있다.

첫 번째 방법의 연구의 예로 PowerScope 가 있다[3]. PowerScope 는 에너지 사용을 프로파일링 하기 위한 도구이다. 각 컴포넌트에 대한 전력을 디지털 멀티미터를 이용하여 측정한 전력 정보를 프로파일링하여 분석한다. 하지만 각 컴포넌트의 전력 상태에 대한 프로파일링 모델 생성은 디바이스 모델 정보에 의존적이다. 만약 해당 컴포넌트의 전력 모델 정보가 없는 경우 전력의 소모를 알 수 없다. 따라서 프로파일링 방식은 컴포넌트의 사용에 대한 전력 모델 정보를 알 수 없는 경우에는 전력 측정이 불가하다.

두 번째 방법의 연구의 예로 EnTrack 이 있다[2]. EnTrack 은 안드로이드 프레임워크에서 어플리케이션과 시스템 서비스 사이의 전력 소모의 흐름을 추적하는 기능을 하는 도구이다. EnTrack 은 안드로이드 프레임워크를 부분 수정해서 Binder Driver 를 모니터링 하여 어플리케이션과 시스템 서비스 사이에서 교환되는 정보를 얻는다. 안드로이드 프레임워크 수정은 모바일을 이용하는 사용자는 물론이고 일반 개발자들에게도 힘든 작업이다. 안드로이드 프레임워크에 풍부한 지식이 있는 시스템 개발자들에게 가능한 작업으로 용이성이 낮은 것을 볼 수 있다. 또한, EnTrack 은 에너지 모델을 사용하였으므로, 디바이스의 기종과 형태가 달라지면 적용할 수 없는 문제점이 생긴다.

기존연구에서는 전력 모델이 존재하지 않는 경우에 정확한 전력 측정이 불가능한 문제와 안드로이드 프레임워크를 수정하는 문제가 있다. 본 연구에서는 실측을 기반으로 OS 수정없이 획득 가능한 프로세스와 서비스 목록으로 어플리케이션의 전력 소모를 분석한다.

3. 어플리케이션 전력 분석 방법

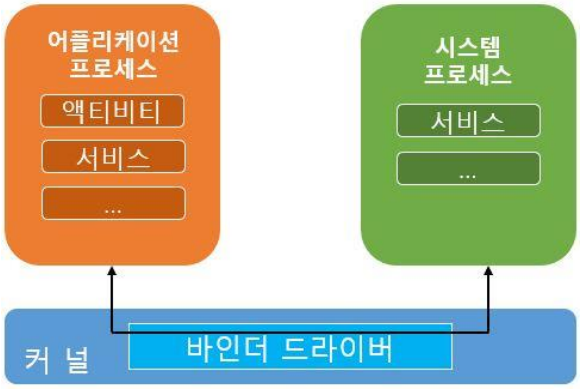
본 논문에서는 측정된 전력에 의한 어플리케이션 정보를 얻도록 한다. 전력에 대한 정보는 프로세스와 서비스 관점으로 해석한다. 전력 그래프를 주기적으로 받아오는 프로세스와 서비스 목록에 의해서 분석해 본다.

사전 연구를 통해 기존 연구의 이식성과 개발 용이성에 대한 단점을 개선하고자 이동식 전력 측정 도구를 개발하였다. 그러나 이동식 전력 측정 도구는 전력에 대한 측정 값만을 얻을 수 있으며, 분석에 관한 정보는 가지고 있지 않다. 따라서 본 논문에서는 전력 값 만을 측정하는 이전 연구의 문제점을 해결하기 위하여 측정에 의한 어플리케이션 전력 정보를 통한 분석을 제안한다.

3.1 프로세스와 서비스의 관계

안드로이드는 어플리케이션이 실행 될 때, 실행에 필요한 프로세스와 스레드들을 생성한다[4,5]. 안드로이드의 프로세스는 어플리케이션 프로세스와 시스템 프로세스로 나누어진다. 이 때, 언급하는 프로세스는 모바일 기기에서 실행중인 프로그램을 말한다. 서비스는 (그림 1)과 같이 동일한 프로세스 내 액티비티에 의해 생성이 되거나, 바인더 통신을 통해 원격 프로세스의 액티비티에 의해 생성된다.

예를 들어, GPS 를 사용하는 어플리케이션을 실행할 경우 해당 어플리케이션을 실행하기 위한 여러 시스템 서비스들이 실행된다. 서비스는 어플리케이션 서비스와 시스템 서비스를 모두 포함한 것이다. 시스템 서비스들은 GPS 모듈과 같은 하드웨어를 사용하기 때문에 해당 전력을 포함한다. 본 논문에서는 서비스가 구동되는데 필요한 전력의 합을 제외한 부분을 프로세스 전력으로 정의한다. 어플리케이션의 전력은 프로세스와 서비스 구동 전력의 합으로 구할 수 있다.



(그림 1) 안드로이드 프레임워크에서
프로세스와 서비스 관계

3.2 이동식 전력 측정 및 분석 도구

에너지 모델을 이용하는 측정방식의 경우, 측정대상인 디바이스의 모듈에 대한 전력 소비 정보를 가지고 있는 기기에만 사용 가능한 특징을 지니고 있다. 현재, 모바일 기기의 종류는 다양해지고 있기 때문에 현 시점에서는 에너지 모델을 사용하는 방법은 한계를 지닌다고 할 수 있다. 이와 달리, 우리는 사전연구에서 이동식 전력 측정도구인 PPAM (Portable Power Measurement and Analysis tools) [1]을 개발하였고, 이는 모바일 기기의 종류에 상관없이 측정이 가능하다는 장점을 가지고 있다. PPAM 은 외부에서 전력을 공급하며, 모바일 기기에서 사용한 전력은 외부에서 공급되는 전력의 변화로 측정한다. PPAM 을 통해 측정된 전력 값은 높은 정확도를 가질 뿐만 아니라, 전력 모델을 이용하는 방식의 이식성이 낮은 단점을 해결할 수 있다. (그림 2)는 연구에 사용한 이동식 전력 측정도구인 PPAM 이다. 본 연구에서는 이식성이 높은 것에 대한 장점과 개발

용이성에 대한 장점을 유지하기 위해 PPAM 을 사용하였다. 그리고 본 연구에서는 PPAM 을 통해 전력에 대한 측정 값 만을 얻을 수 있다는 단점을 보완하고자 어플리케이션의 측정값에 대한 전력 정보를 분석하기 위한 연구를 진행한다.

(그림 2) 이동식 전력측정 도구 (PPAM)

3.3 측정 방법

프로세스의 전력은 서비스를 통하지않고 소모되는 전력을 의미하며, 서비스에 대한 전력은 하드웨어의 사용으로 인한 전력도 포함되도록 한다. 다음은 본실험을 수행하기 위해 추출된 요구사항이다.

- 어플리케이션을 실행시키기 위해서 동시에 실행되는 프로세스들 또한 어플리케이션의 전력으로 포함되도록 하고 분석한다.
- 디스플레이에 의한 전력을 줄이고자 디스플레이의 밝기는 최소로 실험을 한다.
- 어플리케이션을 실행시킴으로 인한 CPU 점유율에 증가에 대한 전력을 포함한다.
- 디스플레이에 대한 전력은 상위 어플리케이션의 전력으로 본다.

실험은 Nexus4 를 이용하였고, OS 버전은 안드로이드 5.1 Lollipop 이다. PPAM 에서는 짧은 주기로 전력을 측정하여 도표의 형식으로 전력 측정 데이터를 처리한다. 1 초를 주기로 프로세스와 서비스의 목록을 기록한다. 전력 정보에 대한 기록 주기를 1 초로 정한 이유는 많은 데이터가 발생하는 것과 PPAM 에서 데이터 처리 시간에 따른 Delay 가 발생 하기 때문이다. 프로세스와 서비스의 경우 PackageName 에 의해서 맵핑시킨다. 주기적으로 목록들을 받아오는 속도를 빠르게 하여 측정되는 전력을 최대한 많이 분석할 수 있도록 한다. 이 방식은 이벤트 방식이 아니기 때문에 많은 양의 데이터를 보낼 경우, 데이터의 손실이 일어날 수 있다.

4. 구현 및 실험

4.1 실험 결과

4.1.1 Default 상태

어플리케이션의 단일 전력 측정을 위해서는 Default 상태에서의 전력을 제외시켜 주어야 한다. Default 상태는 전력 정보를 얻기 위한 어플리케이션만을 실행시킨 상태이며, 백그라운드에서 OS 를 돌리기 위한 최소의 상태를 의미한다.

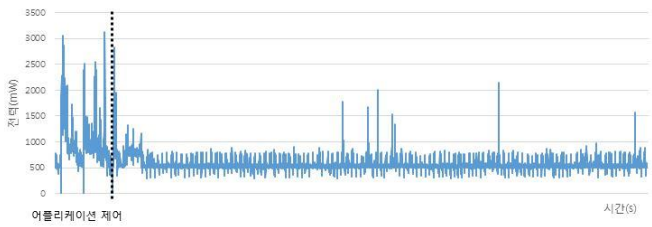
본 논문에서 수행한 실험에서는 <표 1>과 같이 Default 상태에서는 14 개의 프로세스와 20 개의 서비스가 작동한다. 점선으로 나누어진 앞부분은 어플리케이션을 실행하기 위해서 디바이스 제어에 의해서 생겨난 전력이다. 디바이스 제어 후 전력은 거의 일정한 상태로 유지된다.

<표 1> Default 상태에서 실행되는 프로세스와 서비스

No.	프로세스	서비스
1	vkvkg. servicereporter toppam	CommandSenderService
2	android.launcher	-
3	android. inputmethod.latin	LatinIME
4	android.gms	GeocodeService/ FusedProviderService FusedLocationService/ GeofenceProviderService ReportingAndroidService/ GoogleLocationService ContextManagerService / NetworkLocationService GoogleLocationManagerService BackupTransportService (총 10 개)
5	android. gms.persistent	GeocodeService / FusedProviderService FusedLocationService / GeofenceProviderService ReportingAndroidService / GoogleLocationService ContextManagerService / NetworkLocationService GoogleLocationManagerService BackupTransportService (총 10 개)
6	android.process. media	-
7	process.gapps	-
8	android.defcontainer	-
9	android.settings	-
10	android.smspush	WapPushManager
11	android.phone	TelephonyDebugService
12	android.nfc	
13	android.systemui	KeyguardService / ImageWallpaper SystemUIService
14	system	KeyguardService / ImageWallpaper SystemUIService / GeofenceHardwareService TeleBluetoothPhoneService/ LocalTransportService TeleTelecomService (총 7 개)

(그림 3)은 Default 상태의 전력 그래프 이며, 어플리케이션을 제어하는데 나타나는 전력을 제외하고 Default 상태의 전력을 계산하였으며 558.49mW 이다.

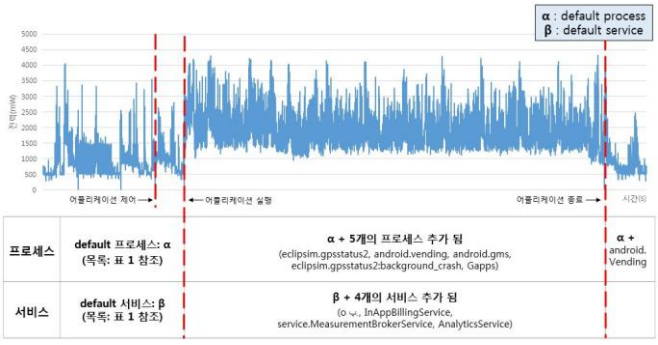
System 과 Systemui 처럼 다른 프로세스에서 같은 서비스가 실행되는 경우가 존재한다. 이것은 하나의 Process 에서 서비스를 사용하고, 다른 Process 에서 만들어진 서비스를 이용한다.



(그림 3) Default 상태 전력 그래프

4.1.2 어플리케이션 실행 상태

어플리케이션 실행에 따른 프로세스 및 서비스의 변화를 관찰하기 위해서 'GPS Status' 어플리케이션을 실행하였으며, 어플리케이션의 실행으로 인해서 eclipsim.gpsstatus2, android.vending, eclipsim.gpsstatus2: background_crash 프로세스 3 개가 추가로 실행되었다. 'GPS Status' 어플리케이션의 경우 하나 이상의 프로세스를 사용할 수 있다. 이는 프로세스 android.vending 에서 사용하는 서비스 InAppBillingService 를 사용하기 위해 실행된 것으로 추정된다. Android.gms 에서는 어플리케이션의 실행에 따라 추가적으로 2 개의 서비스가 실행되었다. (그림 4)는 GPS status 어플리케이션 실행 전력 그래프를 나타냈으며, 그 그래프에 따른 프로세스와 서비스의 목록을 맵핑시켰다. 어플리케이션을 실행 전에는 default 상태의 프로세스와 서비스 목록을 유지하며, GPS 어플리케이션의 실행으로 인해서 5 개의 프로세스와 4 개의 서비스가 추가로 실행되는 것을 확인할 수 있다. 어플리케이션의 종료 이후에는 추가되었던 1 개의 프로세스를 제외한 나머지가 종료되는 것을 확인할 수 있다.



(그림 4) GPS Status 어플리케이션 실행 전력 그래프

실제 어플리케이션에 대한 전력 정보를 얻기 위해, 전력 정보를 얻는 어플리케이션 실행 제어 부분의 전력을 제외하였다. 어플리케이션 제어는 어플리케이션들을 실행하기 위해서 모바일 기기를 조작하는 것이다. GPS Status 어플리케이션이 실행에서부터 종료되는 부분까지의 전력을 계산한 값이 1927.82mW 로 측정되었다.

<표 2>에서는 (그림 4)에서 어플리케이션 실행에 의해 default 상태의 프로세스와 서비스에서 추가된 사항을 나타낸 것이다.

<표 2> 어플리케이션 실행에 의해 추가로 실행되는 프로세스와 서비스

No.	프로세스	서비스
1	eclipsim.gpsstatus2	ㅇㅅ
2	android.vending	InAppBillingService
3	android.gms	service.MeasurementBrokerService /AnalyticsService
4	eclipsim.gpsstatus2: background_crash	-
5	gapps	-

현재 1927.82mW 는 Default 상태의 전력과 GPS Status 어플리케이션의 전력의 합을 의미한다. 실제 실행된 어플리케이션의 전력만을 얻기 위해서는 1927.82mW 에서 Default 전력의 상태를 제외시켜야 한다. GPS Status 어플리케이션만의 전력은 1369.3mW 이다.

5. 결론 및 향후 연구

본 논문에서 제안하는 방법은 사전연구를 통해 개발되었던 이동식 전력 측정 장비의 기기이식성, 기기독립성 등의 장점을 보존하면서, 동시에 전력 소비 정보를 야기하는 프로세스 및 서비스 정보를 함께 제공하여 기기 전체의 전력소비정보만을 제공하였던 단점을 개선하였다. 향후 연구에서는 어플리케이션을 프로세스, 서비스, 컴포넌트를 기준으로 나누어서 분석하는 연구, 두 개 이상의 어플리케이션의 전력을 분리하는 연구에 에 도움이 될 것이다.

참고문헌

[1] 박재현, 최기용, 이정원 "위치제공 API 를 이용한 GPS 상태변화에 따른 전력소모 측정도구 개발" 2015 KIPS 추계학술발표대회 논문집, 22(2), pp.485-488, 2015.10.

[2] Lee, Seokjun, et al. "EnTrack: a system facility for analyzing energy consumption of Android system services." Proceedings of the 2015 ACM International Joint Conference on Pervasive and Ubiquitous Computing, ACM, pp.191-202, 2015.

[3] Flinn, Jason, and Mahadev Satyanarayanan. "Powerscope: A tool for profiling the energy usage of mobile applications." Proceedings of the 1999 WMCSA'99 workshop on Mobile Computing Systems and Applications, pp.2-10, 1999.

[4] Android App Guide – Process & Thread, https://developer.android.com/guide

[5] 송무찬, "[리얼타임] 안드로이드 애플리케이션의 성능 개선을 위한 스레드 관리", 한빛미디어, 2015.05.