

위피에 적용 가능한 플랫폼 신기술 및 위피와 오픈플랫폼 공존 방향

홍종진 / 아로마소프트 R&D 4팀 부장

본고에서는 위피 플랫폼의 발전을 위해, GPOS(General Proposed Operating System) 기반의 모바일 플랫폼 신기술에 대한 전반적인 설명과 위피가 이 신기술들을 어떻게 적용할 수 있는지를 기술할 예정이다. 아직은 개인적인 사견이고 위피에 적용 여부는 단언할 수 없지만 본 글이 위피 발전에 조금이나마 도움이 되었으면 한다.

1. Apple iPhone

먼저, 최근 GPOS를 핫 이슈로 부각시킨 iPhone을 얘기하고자 한다. 현재까지 모바일 분야에서 가장 보편적으로 사용되어온 Microsoft의 Windows Mobile 플랫폼은 선전은 했지만 완전한 성공을 이루지는 못했다. 필자는 그 이유 중 Windows Mobile 플랫폼은 UI 시스템이 너무 복잡했으며 OS기본 기능도 완벽하지 않았다는 점이 큰 비중을 차지하고 있다고 생각하고 있다. 필자도 Windows Mobile 플랫폼을 다년간 사용했지만, 일반 사용자가 사용하기 위해서는 새로운 OS를 공부해야 할 정도의 열정을 필요로 했다. 즉, Windows Mobile의 사례는 GPOS의 대중화가 쉽지 않음을 우리에게 보여주고 있다.

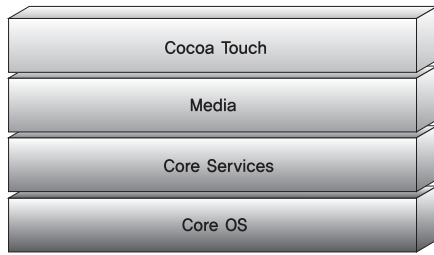
하지만 iPhone의 등장은 이러한 상식을 뒤집어 버렸다. 사용자에게 피쳐폰(Feature Phone)에서 느낄 수 있는 아주 쉬운 UI 시스템과 GPOS에서 느낄 수 있는 강력한 기능을 통해 심플하고 화려한 UI로 사용자에게 친밀하게 접근해서 빛나는 성공을 이루었으며 기술적으로도 산업계에 큰 영향을 미쳤다. 결과적으로 iPhone은 기존 Windows Mobile 시장보다 더 큰 시장 점유율을 확보했다.

그럼 iPhone의 내부를 들여다 보자.

iPhone에서 사용하는 Mac OS X는 아래와 같은 소프트웨어 스택으로 구성되며 객체지향 언어인 Objective-C 기반으로 API(Application Programming Interface)를 제공하고 있다. Objective-C는 C++의 복잡한 객체지향 언어 관련 부분을 단순화해서 자바와 유사하며 C언어를 확장한 언어이기 때문에 C++처럼 빠른 속도를 낼 수 있는 장점을 가지고 있다. 각 소프트웨어 스택은 기본적으로 Mac에서 사용하는 컴포넌트와 동일하지만 임베디드 시스템에 맞도록 최적화했다.

iPhone 출시 후 기술적인 가장 큰 변화는 대화면 터치 UI와 애니메이션 기능의 대중화이며 또 하나의 변화는 iPhone에서 사용하고 있는 Mac OS X 같은 GPOS의 적용이다.

iPhone에 대한 사용자의 첫 인상은 너무나도 자



[그림 1] Mac OS X iPhone Architecture

연스러운 스크롤과 화면 전환 등으로 표현되는 애니메이션 UI 부분이다. 사실 Apple은 디자인이라면 둘째가라면 서러워할 정도로 디자인 부분에 많이 투자하고 있으며 iPhone에서도 Core Animation이라는 기술을 사용해서 이러한 UI 시스템을 구현했다. 일반 피쳐폰에서는 iPhone 이후에 Flash를 통해서 이런 UI시스템을 제공하고 있지만 속도면이나 기능면에서 iPhone보다는 부족하다고 할 수 있다.

Core Animation은 기본적으로 Objective C API로 구성되어 있기 때문에 액션 스크립트를 사용하는 Flash보다 성능 면에서 뛰어나며 다양한 layer와 애니메이션 패턴을 지원해서 보다 강력한 애니메이션 화면을 구성할 수 있다. 단 스크립트 언어가 아니기 때문에 Flash보다는 개발속도는 느릴 수 있다.

위피의 경우 현재 애니메이션 기능이 없으며 Flash지원 기능도 표준적인 인터페이스를 가지고 있지 못하고 있다. 따라서 Flash 및 애니메이션 기능을 지원하는 API개발이 필요하다. 애니메이션은 여러 layer가 중첩 되어서 다양한 효과를 출력하기 때문에 2D 및 3D기능 및 멀티미디어 관련 기능도 개선되어야 한다.

Mac OS X는 기존 Mac에서 사용하는 GPOS로 Darwin이라는 하는 Unix 계열의 오픈 소스 커널을 사용하고 있다. FreeBSD기반의 이 커널은 인

터넷에 공개되어 있기 때문에 누구나 접근 및 수정이 가능하다. 하지만 나머지 소프트웨어는 공개하고 있지 않다.

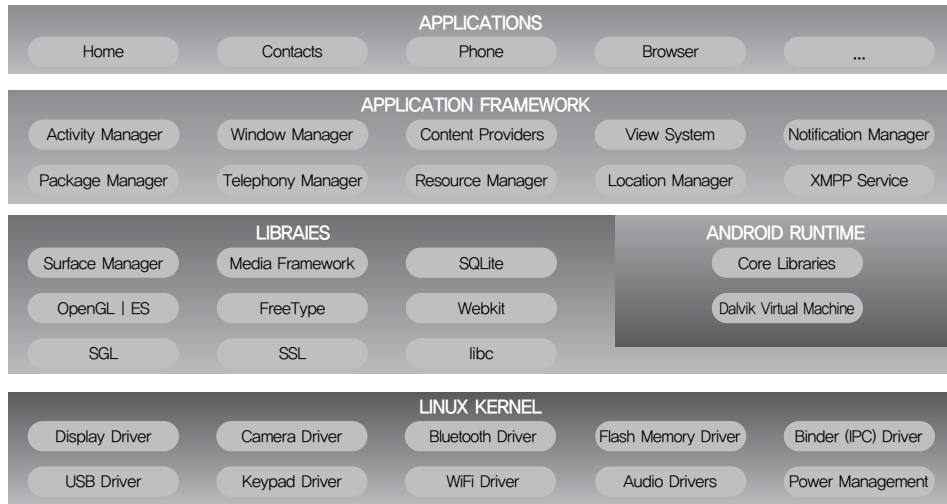
iPhone은 Unix 커널을 적용함으로써 기본 피쳐폰에서 불가능 했던 다중실행 환경 및 안정된 어플 개발 환경을 가질 수 있었다. 이 부분은 사실 구글 안드로이드 플랫폼에도 영향을 미쳐 같은 유닉스 계열은 Linux Kernel을 사용하게 되었다.

위피의 경우는 현재 단일 프로세스에 다중 위피 애플리케이션을 실행할 수 있는 환경은 준비되어 있지만, 현재의 구현에서는 대부분 위피 애플리케이션을 다중으로 실행하고 있지 않다. 따라서 위피 애플리케이션도 GPOS에서 다중으로 실행할 수 있는 구조가 필요하다. 다중 애플리케이션 모델은 다양한 이슈가 존재한다. 리소스 충돌 문제, 애플리케이션 관리 문제 등등. 하지만 이런 문제를 해결하면 많은 장점이 있다.

2. Google Android 플랫폼

Google에서 GPOS기반 모바일 플랫폼 시장의 성장을 감지하고 오픈 플랫폼인 안드로이드 플랫폼을 준비했다. 하지만 애플과는 다른 방식인 진정한 오픈 플랫폼 형태로 시장에 접근하고 있다. Mac OS X 플랫폼의 핵심부분은 모두 공개되어 있지 않고 있기 때문에 오픈 플랫폼이 아니다. Android 플랫폼은 모든 소스가 공개되어 있으면 누구나 사용가능하기 때문에 진정한 오픈 플랫폼의 형태를 가지고 있다.

Android 플랫폼은 가장 큰 장점은 Java언어 기반 애플리케이션 프레임워크(Framework)와 Full Browser

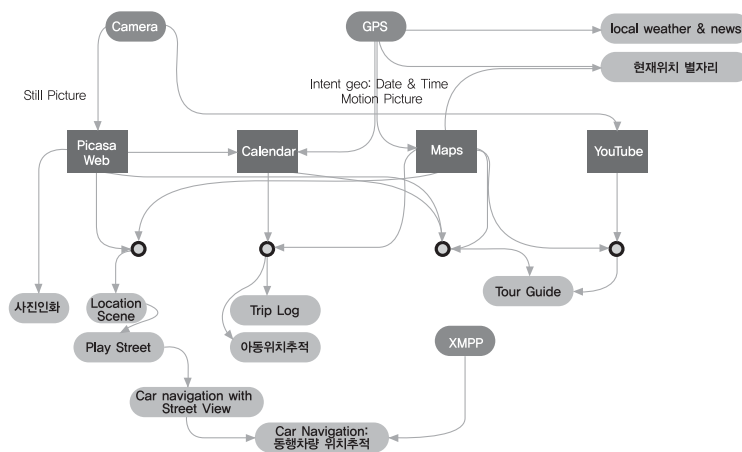


[그림 2] 안드로이드 아키텍처

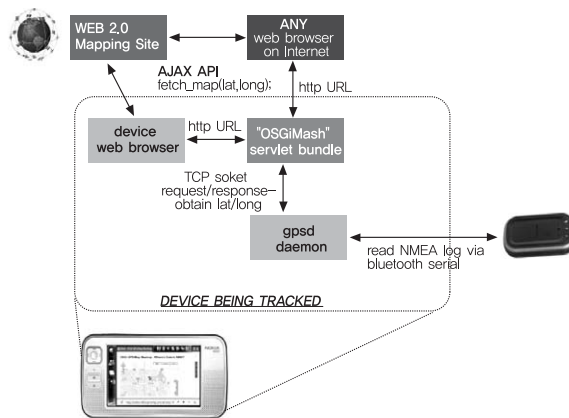
및 웹 개발환경 지원 부분이다. 구글은 Java 언어 기반의 플랫폼을 구상하면서 기존 Java ME와 Java SE와의 호환성은 배제하고 새로운 자바 플랫폼을 만들었다. 개인적으로 구글이 이런 선택을 한 이유는 기술적인 측면에서 Android 플랫폼이 Java ME 이 보다는 강력한 기능이 필요했고 Java SE보다는 슬림한 플랫폼을 원했을 거라고 추측된다. 아무튼 Android 플랫폼은 기능적으로 Java SE쪽과 유사하다. Java 언어 기반의 플랫폼의 가장 큰 장점은 쉽

게 많은 개발자를 확보할 수 있으면 생산성 높은 개발환경을 제공할 수 있다는 것이다.

구글은 Android 플랫폼에서 다양한 매쉬업 서비스가 동작하기를 원하고 있으며 [그림 3]과 같은 구글 서비스들이 이런 매쉬업의 핵심이 될 것이라 판단하고 있다. [그림 3]은 Picasa의 사진 자료와 Google Calendar, Google Map, YouTube 등을 이용해 사용자가 다양한 매쉬업 서비스를 안드로이드에 사용할 수 있다는 것을 보여 주고 있다.



[그림 3] 구글 매쉬업 서비스



[그림 4] Nokia 메쉬업 서비스

이런 메쉬업 서비스의 핵심은 Full Browser의 지원과 자바 스크립트 지원인데, 구글이 크롬 브라우저를 만든 이유도 사실 기존 자바 스크립트 엔진 성능에 한계를 인식하고 그것을 극복하기 위해서다. 안드로이드에는 Webkit이라는 브라우저 엔진을 사용하고 있는데, Webkit은 크롬 브라우저, iPhone의 사파리 브라우저 등에서 사용되고 있으며 안드로이드 플랫폼에서도 사용되고 있다. 안드로이드는 Webkit에 접근할 수 있는 API를 제공해서 어플 개발자들이 쉽게 위젯등의 웹애플리케이션을 개발할 수 있다.

[그림 4]는 Nokia의 메쉬업 서비스인데 Nokia에서도 Webkit기반의 브라우저를 제공하고 있으며 [그림 4]와 같이 맵과 디바이스 위치 정보를 이용해서 주변의 다양한 디바이스들과 연동하는 기능을 구상하고 있다.

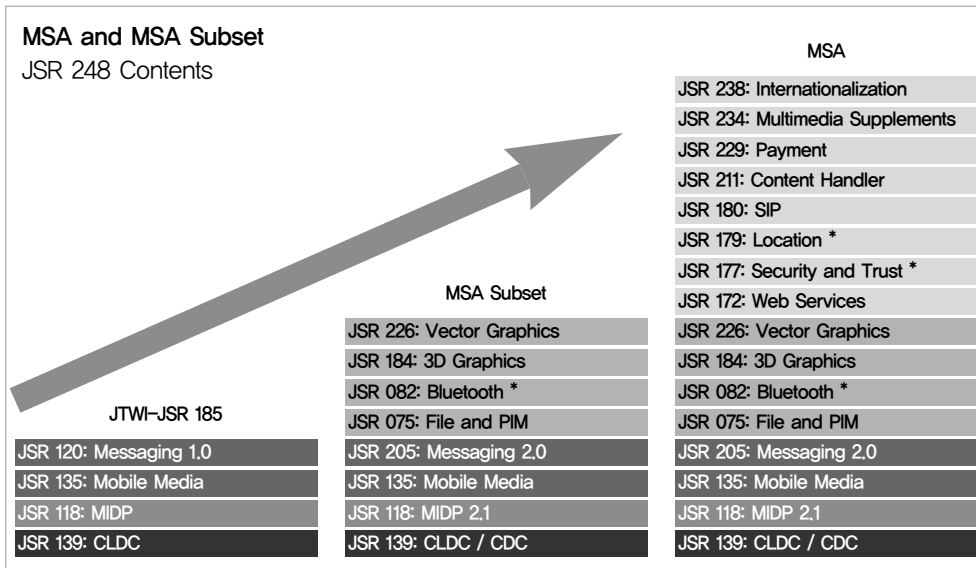
위피도 이러한 메쉬업 서비스를 지원하기 위한 API가 필요하다. 우선 Browser 엔진에 접근할 수 있는 API가 필요하며 다양한 부가 서비스를 위해서 대기화면 접근 등의 API도 부가적으로 구현되어야 한다. 이런 API를 통해서 웹기반의 메쉬업 서비스를 위피 플랫폼에도 제공할 수 있을 것이다.

3. J2ME 플랫폼

현재 위피 플랫폼은 실질적으로 이통사에 상관없이 공통적으로 개발할 수 있는 애플리케이션은 게임과 간단한 업무용 애플리케이션 정도이다. 실제로 좀더 강력한 애플리케이션을 개발하기 위해서 각 이통사는 고유의 OEM API를 사용해야 한다. 이런 OEM API를 표준화해서 API를 좀 더 풍부하게 만드는 노력 또한 위피 플랫폼의 성장에 중요하다.

위피와 유사한 Sun의 J2ME 플랫폼은 MSA라는 표준을 정해서 다양한 API를 지원하도록 유도하고 있다. [그림 5]는 MSA를 구성하고 있는 다양한 JSR을 나타내고 있다. 현재 해외에서 사용되는 대부분은 J2ME 플랫폼은 이 MSA Full Set 지원해서 호환성 및 풍부한 API를 제공하고 있다.

새로운 GPOS플랫폼이 주목 받는 가장 큰 이유는 기존 피쳐폰의 운영체계가 제조사 자체적인 플랫폼이기 때문에 기술적으로나 생산성 측면에서 한계를 나타내고 있기 때문이라고 판단 된다. 제조사의 입장에서 OS 발전에 따라 유지 관리 비용에 대한 부담이 가중되고 있는 것 또한 사실이다.



[그림 5] MSA 스택

〈표 1〉 MSA에 포함된 JSR요약

JSR	설명
JSR 139(CLDC)	VM 및 기본 API
JSR 188(MIDP)	UI 및 기타 API
JSR 75(File & PIM)	파일 액세스 및 Personal Information Management 데이터 액세스(주소록, 일정, 할일)
JSR 135(Mobile Media)	멀티미디어 API(Sound, Video, Image, Recording, etc)
JSR 184(3D Graphics)	3D 그래픽 API(Open GL ES와 유사)
JSR 205(Messaging)	Wireless Message API
JSR 225(Vector Graphics)	2D 그래픽 API
JSR 172(Web Services)	Access Web Service
JSR 177(Security & Trust)	Access Security elements.
JSR 179(Location)	위치 정보 API
JSR 180(SIP)	SIP(Session Initiation Protocol) API
JSR 211(Content Handler)	Type에 따른 Contents & 애플리케이션 연동 API
JSR 229(Payment)	전자 지불 API
JSR 234(Multimedia Supplements)	Advanced Multimedia Feature(Image Encoding/3D Audio/etc)
JSR 238(Internationalization)	국제화 지원

안드로이드, 리모와 같은 플랫폼은 아직 초기 단계라 기능적이 부분에서 많이 부족한 부분이 있다. 하지만 이들 플랫폼이 기존 제조사 플랫폼을 천천히 바꾸어 나갈 것이다. 완전히 바꾸지 못하더라도 플랫폼의 새로운 방향을 제시하며 큰 영향을 미칠 것이다.

위피도 이런 발전에 대응하기 위해서는 기술적인

부분만 아니고 생태계 구축에 관련된 부분도 많은 고민이 필요할 것 같다. 사실 기술적인 부분은 생태계 구축에 대한 명확한 공유만 있으면 투자로써 해결이 가능하지 않을까? 국산 플랫폼인 위피가 보다 발전해서 많은 사람들이 혜택을 누리기를 바란다. **TTA**