

HTML5 기반의 웹 플랫폼 기술 표준화 동향

Trends on Standardizations of HTML5 based Web Platform Technology

전종홍 (J.H. Jeon) 서비스융합표준연구팀 책임연구원
이승윤 (S.Y. Lee) 서비스융합표준연구팀 팀장

* 본 연구는 방송통신위원회의 지원을 받는 방송통신표준개발지원사업의 연구 결과로 수행함.

웹은 초기에 단순히 정적인 문서 형태 정보를 공유하는 기술에서, 서로 다른 서비스를 연결하는 기술과 서로 다른 원격 데이터들을 연결하는 기술로 진화하였고, 이제는 단말의 HW를 제어하고 연결하는 단계를 넘어 모든 사물을 연결하는 기술로도 발전하고 있다. 이제 웹은 단지 개별 콘텐츠와 서비스를 제공하는 기술로서가 아니라, 다양한 응용과 서비스를 연동하고 제공하기 위한 명실상부한 플랫폼이 되어가고 있다. 기술적으로도 과거 웹 기술이 단지 브라우저와 서버 기술에만 초점을 맞추었다면, 최근에는 HTML5를 중심으로 리치 웹 애플리케이션 기술, 시스템 API 기술, 서비스 플랫폼 기술, 웹 운영체제 기술들과 같은 플랫폼 지향적인 기술개발 및 제품개발이 진행되고 있다. 본고에서는 이처럼 빠르게 발전하고 있는 웹 플랫폼 기술과 웹 응용 기술 표준화 동향에 대해 살펴보고, 향후 웹 플랫폼과 웹 응용 기술이 해결해야 할 이슈들과 나아갈 방향들에 대해 고찰해보고자 하였다.

2012
Electronics and
Telecommunications
Trends

융합환경하에서의
신성장동력 분석 특집

- I. 서론
- II. 웹 기술의 진화
- III. 웹 애플리케이션 플랫폼
기술 표준 동향
- IV. 웹 운영체제 기술 동향
- V. 결론

I. 서론

1989년 CERN의 팀 버너스 리에 의해 만들어진 월드 와이드 웹 기술은 HTML(HyperText Markup Language), URL(Unified Resource Locator), HTTP(HyperText Transfer Protocol)이라는 세 가지 기술을 기초로 간단한 문서와 자원들을 공유하기 위한 목적으로 출발하였다. 이후 1994년 기술 표준화를 위한 W3C(World Wide Web Consortium)가 창립되면서 웹 기술은 눈부신 진보와 함께 인류 생활에 있어 없어서는 안 될 중요한 기반 기술로 발전해왔다.

초기에 단순히 정적인 문서 형태 정보를 공유하는 기술(web of document)에서, 서로 다른 서비스를 연결하는 기술(web of services)과 서로 다른 원격 데이터들을 연결하는 기술(web of data)로 진화하였고, 이제는 단말의 HW를 제어하고 연결하는 단계(web of devices)를 넘어, 클라우드를 통해 정보를 공유 제공할 수 있도록 하며 모든 사물을 연결하는 단계(web of things)로 까지 진화하고 있다.

2012년 6월 현재 전 세계에는 7억 개[1]의 웹 사이트가 공식적으로 서비스를 제공하고 있는 것으로 추정되고 있고, 이중 실제 작동 중인 사이트도 2억 개에 이르며, 비공식적으로 웹 서버 기능까지 내장한 장치들을 포함한다면 실제로는 2~3배 정도 규모가 될 것으로 예상되고 있다. 인터넷 사용자의 대부분은 하루 평균 2시간 이상을 인터넷에 접속하며, 이 중 70%가 넘는 시간을 뉴스, 쇼핑, 상거래, 소셜 서비스, 엔터테인먼트 등 웹 기술로 만들어진 서비스들을 사용하며 즐기고 있다.

사실상 웹은 인터넷의 전부이며, 인터넷이 웹이라고 해도 과언이 아닌 시대에 살고 있으며, 단지 개별 콘텐츠와 서비스를 제공하는 기술로서가 아니라, 점점 더 다양한 응용과 서비스를 연동하고 제공하는 대표적인 플랫폼이 되었다고 할 수 있다.

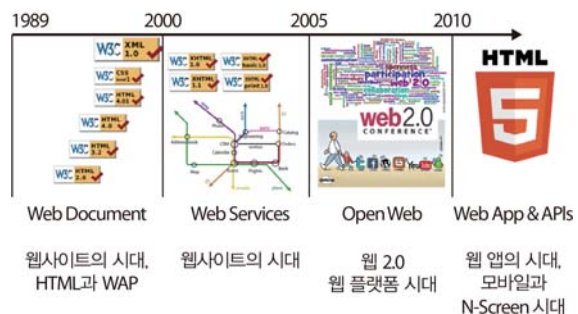
본고에서는 이처럼 빠르게 발전하고 있는 웹 플랫폼

기술과 웹 응용 기술 동향에 대해 살펴보고, 향후 웹 플랫폼과 웹 응용 기술이 해결해야 할 이슈들과 나아갈 방향들에 대해 고찰해 보고자 하였다.

II. 웹 기술의 진화

웹 기술의 진화 발전 과정을 살펴보면 (그림 1)과 같이 크게 4단계로 구분해 볼 수 있다. 첫 번째 단계는 1989년부터 1999년까지의 기간으로 HTML, URL, HTTP라는 세 가지 기술에 기초해 웹 기술이 제안되고 보다 나은 인간 중심의 정보 처리 및 지식공유 등을 목표로 하는 단계였다. 두 번째 단계는 2000년부터 2004년까지의 기간으로 XML에 기반하며 인간 중심의 정보 처리뿐 아니라 다양한 디바이스와 서비스, 멀티미디어를 연결하는 것을 목표로 하는 단계였다. 세 번째 단계는 2005년부터 2009년까지로, 구글, 아마존, 위키피디아 등의 성공과 함께 웹 산업을 제2의 전성기로 이끌며 다양한 신규 서비스가 등장할 수 있는 기반을 만들었다. 그리고 마지막 네 번째 단계는 2010년 이후부터 현재까지로, 스마트폰 및 태블릿 등 다양한 모바일 기기들을 대상으로 HTML5와 web API를 통해 한단계 진화된 웹 응용 환경을 제공하며, 위치 정보 및 소셜 정보 등을 결합하는 통합 응용 플랫폼으로서 웹이 자리 잡아 가는 단계라 할 수 있다[2].

이러한 웹 기술은 웹 마크업 기술을 기반으로 웹 애플



(그림 1) 웹 기술의 진화 발전 과정

리케이션 기술, 그리고 웹 플랫폼 기술로 진화하고 있다.

1. HTML5와 브라우저 기술

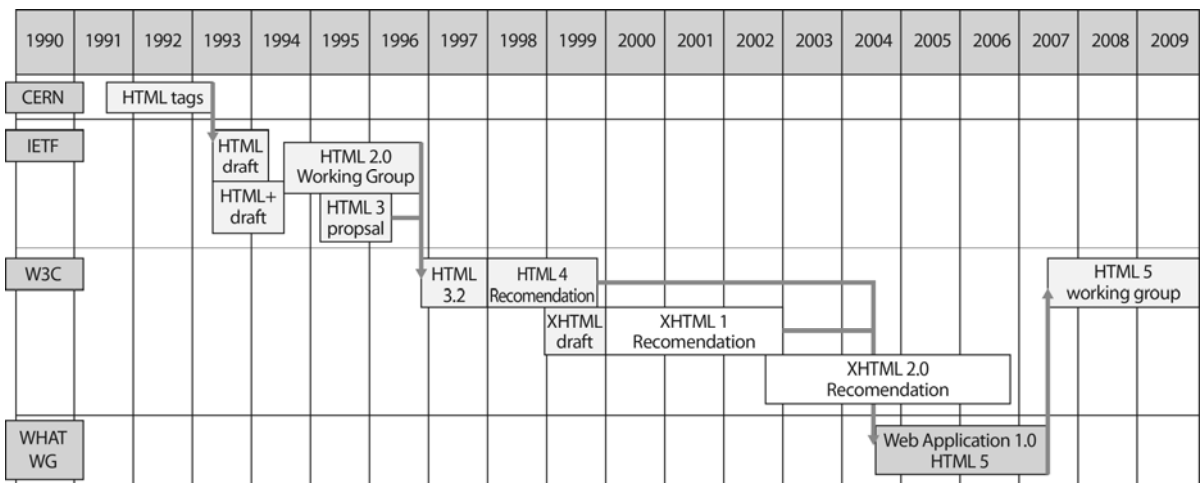
웹 기술이 확산된 가장 큰 배경은 HTML이라는 장치 독립적 표현 언어를 이용하여 정보를 표현하고, 이를 다양한 단말에서 브라우저라는 응용 프로그램으로 접속하여 활용할 수 있게 함으로써, 언제 어디서든 표준 브라우저만 있으면 웹을 사용할 수 있다는 단말 플랫폼 비종속적인 장점을 가질 수 있었기 때문이다. (그림 2)와 같이 1993년 HTML 1.0 규격이 만들어지고 난 후, 1997년 HTML 4.0과 1999년의 HTML 4.01 규격이 만들어지기까지 웹 기술은 폭발적으로 성장하였다. 그러나 HTML 자체가 갖는 확장의 어려움으로 W3C에서는 1999년부터 좀 더 다양한 확장성을 가질 수 있도록 하기 위해 XML을 기반으로 하는 새로운 XHTML(Extensible Hypertext Markup Language) 1.0 개발을 추진하였고, 2009년까지 XHTML 2.0 개발을 진행하여 왔다.

그러나 HTML은 단순함을 가졌으나 확장이 어려웠고, XHTML은 확장성은 좋았으나 지나치게 복잡하다는 단점을 가졌다. 이러한 이유로 XHTML 표준화는 계속 지연이 되었고, 이에 다양한 기술적인 진화 내역들을

흡수한 새로운 마크업 언어를 필요로 했던 업계 전문가들이 2004년 WHATWG(Web Hypertext Application Technology Working Group)를 구성하고 다양한 웹 애플리케이션에 효과적으로 사용할 수 있는 보다 단순하면서도 다양한 확장성을 갖는 HTML 5.0 규격을 만들기 시작하였다. 이에 W3C는 2008년 새로운 HTML 규격을 만들기 위한 HTML WG를 구성하였고, WHATWG의 규격을 기초로 한 새로운 HTML5 표준안을 만들기 시작하였다.









HTML5 표준의 가장 큰 특징은 기존의 HTML 표준의 한계를 극복하기 위해 별도의 추가적인 확장 기능 없이도 과거에는 불가능했던 다양한 처리들을 가능하게 하는 것에 있다. 이를 위해 HTML5에서는 다음 <표 1>과 같이 웹 애플리케이션 개발에 도움을 줄 수 있는 다양한 기능과 API를 제공하고 있다.

HTML5 표준안은 아직 초안 상태로 앞으로도 많은 수정과 보완 작업이 필요할 것으로 예상되고 있지만, 현재 스펙을 기준으로 한 구현은 이미 대부분의 브라우저에서 구현되어 동작되고 있다. 구글/애플/모질라/마이크로소프트 등을 비롯한 많은 브라우저 개발사들은 브라우저 기능 개선과 함께 자사 규격을 표준에 반영하고 서비스 개발에 반영하기 위한 확장 노력을 병행하고 있다.



(그림 2) HTML 마크업 기술의 발전 과정[3]

〈표 1〉 HTML5 주요 개선점 및 특징

HTML5 기술의 주요 특징		시사점
	<ul style="list-style-type: none"> Semantics 보다 구조화되고 다양한 기능의 HTML 태그를 제공	보다 지능화되고 다양한 형태의 풍부한 웹 문서 표현 가능
	<ul style="list-style-type: none"> Multimedia 비디오, 오디오 지원 기능의 자체 지원을 통한 강력한 멀티미디어 기능 제공	액티브엑스와 플래시 같은 별도 외부 플러그인 필요성 제거
	<ul style="list-style-type: none"> Offline & Storage 네트워크가 지원되지 않는 환경에서도 웹 이용을 가능케 하는 오프라인 처리 기능과 로컬 스토리지, DB, file 액세스 처리 기능	웹의 한계로 여겨졌던 네트워크 단절 시 처리 방법과 데이터 저장 기능 문제 해결
	<ul style="list-style-type: none"> 3D, Graphics & Effects SVG, 캔버스, WebGL 등을 통한 다양한 2차원/3차원 그래픽 기능의 제공	외부 플러그인 기능 없이 다양한 2D/3D 그래픽 처리 가능
	<ul style="list-style-type: none"> Device Access GPS, 카메라, 동작 센서 등 디바이스의 하드웨어 기능을 웹에서 직접 제어할 수 있도록 하는 기능	웹 기반 디바이스 제어 기능을 통해 본격적인 웹 애플리케이션 개발 가능
	<ul style="list-style-type: none"> Performance & Integration 비동기 통신, 다중 스레드 기능 등을 통한 웹에서의 처리 성능을 향상	웹의 가장 큰 문제 중 하나였던 성능 문제를 대폭 개선
	<ul style="list-style-type: none"> Connectivity 클라이언트와 서버 간의 효율적인 통신 기능 제공을 통한 웹 기반 커뮤니케이션 효율 대폭 강화	웹에서의 다양한 통신 기능(메시징, 응용간 통신 등) 제공을 통한 응용 개발 범위 확대
	<ul style="list-style-type: none"> CSS3 Styling Effect 기존 웹 문서의 번거로움 없이 웹 애플리케이션의 UI(스타일과 효과 등) 기능을 대폭 강화	UI 측면에서 N-스크린 서비스 제공 가능

2. 웹 애플리케이션 기술

2005년 이후로 웹 2.0의 성장과 함께 다양한 신규 응용과 기술들이 등장하기 시작하였다. 이 중에서도 가장 많은 변화를 일으킨 부분은 RSS/Atom 등의 XML 데이터 조각을 이용한 서비스 연동 기술과 AJAX와 같은 비동기식 처리 기술, 브라우저 및 JavaScript 가속화 기술, Open API와 매시업 기술 분야 등이었고, 이러한 기술들을 종합하는 웹 애플리케이션 기술 분야에서 많은 발전이 있었다[2].

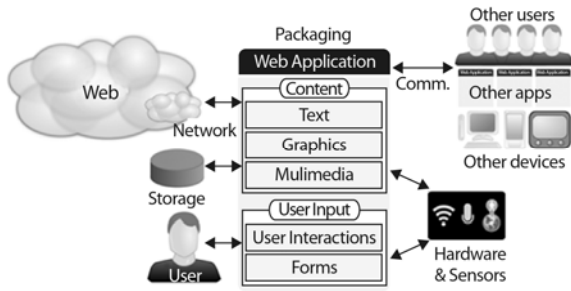
일반적으로 “웹 애플리케이션”이라는 용어는 HTTP를 통해 전달되는 웹 페이지(XHTML 또는 그 변이형과 CSS, ECMAScript로 구성되는)의 집합체들이 웹 브라우저 내에서 애플리케이션 같은 환경을 제공하는 것을 말한다. 즉, 웹 애플리케이션은 여러 페이지를 거치는 대화형 처리 절차를 가지며, 이를 위한 상태 유지와 데이터 유지를 필요로 한다는 점에서 단순한 웹 콘텐츠와

는 구분된다[4].

일반적으로 웹 애플리케이션의 경우 별도 프로그램의 설치 없이도 표준 브라우저만 있으면 계속 업그레이드된 기능을 사용할 수 있고, Open API 등을 통해 손쉽게 매시업할 수 있도록 기능을 제공하는 등 재활용을 할 수 있다는 장점을 갖는 반면, 오프라인 처리와 단말의 특성 정보를 활용할 수 없고, 브라우저의 성능에 좌우되며 대용량의 처리 등에 한계를 갖는다는 단점을 갖고 있다.

이 중에서도 단말의 하드웨어와 관련된 제어는 직접할 수 없다는 점은 가장 큰 단점으로 꼽히고 있다. 예를 들어, 간단한 애플리케이션을 통해 배터리의 잔량, 주소록의 주소 정보, 단말에 저장된 일정 정보 등을 활용하고자 해도 할 수 없다는 점은 치명적인 약점으로 꼽혀 왔다.

웹 애플리케이션을 위한 W3C의 표준들은 (그림 3)과 같이 그래픽, 멀티미디어, 장치 적응, 양식, 사용자 상호



(그림 3) 웹 애플리케이션 기술 표준 분류 및 현황[5]

작용, 데이터 스토리지, 개인정보 관리, 센서 및 하드웨어

어 통합, 네트워크, 통신 및 발견, 패키징, 성능 및 최적화 등의 12개 범주로 구분할 수 있으며 <표 2>와 같이 34종 이상의 표준들이 10개 이상의 워킹 그룹들을 통해 개발되고 있다.

III. 웹 애플리케이션 플랫폼 기술 표준 동향

일반적으로 플랫폼은 협의(狹義)의 의미로는 API를 통해 애플리케이션에 필요한 서비스를 제공하는 특수한

<표 2> W3C 웹 애플리케이션 관련 표준

기능 분류	관련 표준 문서(개발 중 또는 완료)	문서상태	개발 WG	비고
그래픽스 (7종)	SVG	IS	SVG WG	TTA 단체표준
	HTML Canvas 2D Context	WD	HTML WG	
	CSS Backgrounds and Borders	CR	CSS WG	
	CSS 2D Transforms Module Level 3	WD	CSS WG	
	CSS Animations Module Level 3	WD	CSS WG	
	Media Queries	CR	CSS WG	
	WOFF File Format 1.0	LC	Web Fonts WG	
멀티미디어 (3종)	HTML5	LC	HTML WG	
	HTML Media Capture	WD	Device APIs & Policy WG	
	HTML Canvas 2D Context	WD	HTML WG	
폼(1종)	HTML5	LC	HTML WG	
사용자 상호작용(1종)	Touch Events Specification(not published yet)	N/A	Web Events WG	
데이터 스토리지 (3종)	Web Storage	WD	Web Application WG	
	File API	WD	Web Application WG	
	Contact API	WD	Device APIs & Policy WG	
센서 및 HW 처리 (3종)	Geolocation API	CR	Geolocation WG	
	The System Information API	WD	Device APIs & Policy WG	
	The Media Capture API	WD	Device APIs & Policy WG	
네트워크 (7종)	XMLHttpRequest	CR	Web Application WG	TTA 단체표준
	XMLHttpRequest Level 2	WD	Web Application WG	
	Cross-Origin Resource Sharing	WD	Web Application WG	
	Server-Sent Events	WD	Web Application WG	
	The WebSocket API	WD	Web Application WG	
	HTML5 onLine DOM state	WD	HTML WG	
	Network Property in The System Information API	WD	Device APIs & Policy WG	
커뮤니케이션 (2종)	The Messaging API	WD	Device APIs & Policy WG	
	HTML5 Web Messaging	WD	Web Application WG	
패키징 (4종)	HTML5 Application Cache	WD	HTML WG	
	Widgets Packaging & Configuration	LC	Web Application WG	
	Digital Signatures for Widgets	CR	Web Application WG	
	Widget Access Request Policy	CR	Web Application WG	
성능 및 최적화 (3종)	Navigation Timing	LC	Web Performance WG	KS/KCLS 표준
	Web Workers	WD	Web Application WG	
	Mobile Web Application Best Practices	IS	MWBP WG	

SW 프로그램을 의미하며, 광의(廣義)의 의미로는 제품이나 서비스 개발의 기반이 되는 아키텍처로 다양한 응용과 서비스가 개발/제공되는 환경을 포괄적으로 의미한다.

초기의 웹 기술은 단순히 문서를 공유하고 정보를 표현/공유하기 위한 구조였지만, 지속적인 기술의 발전과 확장에 따라 2000년 이후부터 웹 기술은 점점 더 다양한 서비스와 데이터를 연동하고 서비스할 수 있도록 하는 응용 플랫폼의 형태로 발전을 해왔다. 이러한 웹 플랫폼 기술의 지속적인 확산 배경에는 다음과 같은 특징들이 있다.

- 1) 단순성: 웹 기술의 HTTP, URI, HTML이라는 단순한 요소들만을 활용함에 따라 기술 확장이 용이하고, 다른 응용 개발 환경보다는 손쉽게 개발자가 관련 기술을 익히고 활용하고 적용해 볼 수 있다는 장점을 갖고 있다.
- 2) 호환성: 웹 기술은 전 세계적으로 단일 표준화되어 있고 표준과 기술이 공개되어 있어 관련 기술을 손쉽게 적용하여 활용할 수 있다는 장점을 갖고 있다.
- 3) 독립성: HW 및 운영체제 의존성이 적어서 표준 브라우저와 웹 엔진이 있는 환경이라면 어떤 장치에서든 일관되게 활용될 수 있다는 장점을 갖고 있다.
- 4) 개방성: 웹 응용은 서비스 재활용이 손쉬우며, 개방형 API 등을 통해 기존에 개발된 내용을 손쉽게 융합하여 활용할 수 있다는 장점을 갖고 있다.
- 5) 이식성: 웹 기술은 소형의 단말 환경에서도 손쉽게 적용할 수 있다는 특징을 갖고 있으며, 이를 통해 다양한 서비스와 데이터들을 손쉽게 연결할 수 있다는 장점을 갖고 있다.

웹 플랫폼은 웹 서버와 웹 브라우저를 중심으로 하는 웹 응용 서비스 환경을 말하며, 다양한 웹 응용과 웹 서비스가 개발/제공되는 환경을 의미한다. 최근의 HTML5

동향에서 볼 수 있듯이 웹 기술은, 점점 더 단말/플랫폼/스크린의 경계가 없도록 하며, 모든 정보와 서비스 및 사물을 묶는 인프라와 플랫폼이자 새로운 앱과 SW 환경으로서의 형태로 진화하고 있다. 이러한 웹 플랫폼 기술들은 다음과 같이 크게 세 가지 분야로 나눌 수 있다.

1. Device API

웹 애플리케이션이 갖는 가장 큰 단점 중 하나는 네이티브 애플리케이션과 달리 단말의 하드웨어와 관련되는 제어를 할 수 없다는 점이라 할 수 있다. 예를 들어 간단한 애플리케이션을 통해 배터리의 잔량, 주소록의 주소 정보, 단말에 저장된 일정 정보 등을 활용하고자 해도 할 수 없다는 점은 치명적인 약점으로 꼽혀 왔다.

특히 이러한 웹 애플리케이션의 약점은 모바일 환경에서 더욱 치명적이라 할 수 있다. 데스크톱의 웹 애플리케이션과 달리 모바일 단말의 경우 플랫폼으로부터의 제약을 많이 갖고 있지만, 반면에 보다 다양하게 디바이스 기능들을 활용할 필요를 갖고 있어 단말 기능 접근에 대한 요구가 훨씬 크다고 할 수 있다.

이에 W3C에서는 2008년 12월 Device API와 관련되는 다양한 표준화 이슈들을 발굴하기 위해 관련 워크숍을 개최하였고, 워크숍 논의 결과를 기초로 WG 설립 작업을 진행하여 2009년 6월 DAP(Device API and Policy) WG를 발족하게 되었다.

DAP WG는 2011년 6월까지의 1, 2단계 활동을 통해 다음과 같이 10개의 API들을 도출하였고 이에 대한 표준 문서 개발을 진행하고 있다[6].

- Contacts API(Last Call)
- HTML Media Capture(Working Draft)
- Media Capture API(Working Draft)
- Messaging API(Working Draft)
- Calendar API(First Public Working Draft)
- Battery Status API(First Public Working Draft)

- Network Information API(First Public Working Draft)
- Permissions for Device API Access(First Public Working Draft)
- Gallery API(Editor's draft)
- Feature Permissions(Editor's draft)

이 밖에 신규로 다음과 같이 6개의 API 문서를 새롭게 개발하는 작업도 함께 진행하고 있다.

- Vibration API
- Menu API
- Beep API
- Application Registration API
- Generic Sensor API
- Tasks API

DAP WG는 Device API WG로 이름을 변경하고 2013년 7월 말까지를 목표로 위의 표준들의 표준화를 끝낸 후 실제 브라우저 내의 구현을 목표로 작업을 진행하고 있다. Device API 표준은 모바일 웹 애플리케이션과 관련하여 가장 많은 영향을 미칠 수 있는 표준으로, 향후 표준화가 완료되고 다양한 모바일 브라우저에서 구현된다면 강력하고도 다양한 모바일 웹 애플리케이션이 등장할 것으로 예상된다.

2. Web API

W3C는 2006년 Rich Web Client Activity를 시작하며 Web Application WG와 Web API WG를 만들어 표준화 작업을 진행하다, 2008년 Web Application WG로 통합하여 표준화 작업을 진행해오고 있다.

현재 약 20여 개 이상의 웹 애플리케이션 관련 표준안들이 Web Application WG 내에서 검토되고 협의 중에 있으며, 여기에는 XHR(XMLHttpRequest), Widget, Web IDL, Web Socket API, CORS(Cross-Origin Resource Sharing) 등이 포함되어 있다. 이 밖에도 HTML5 규격

과 관련된 Web Storage, Web Workers, Data-Cache API, DOM Level 3 Events 등도 작업 대상으로 포함되어 있다. 이러한 Web Application WG의 주요 Web API 표준화 작업은 다음과 같이 6가지 내용으로 요약할 수 있다[4],[7].

가. XHR

XHR은 AJAX와 같은 비동기식 웹 애플리케이션 개발 기법의 핵심 요소, 서버와 클라이언트 사이의 데이터 전송을 위한 기능을 정의한다. 현재 XHR 1.0을 확장하는 XHR 2.0 표준에 대한 초안 작업을 진행하고 있다.

나. Web IDL

Web IDL은 브라우저에서 구현되어 웹상에서 인터페이스를 설명하기 위한 용도로 사용될 수 있는 IDL(인터페이스 정의 언어)을 정의한다. Web IDL은 인터페이스의 정의와 더불어, 인터페이스와 ECMAScript, 그리고 자바 바인딩에 대한 명료한 적합성 요구사항을 제공하는 데 이용된다.

다. Web Socket

웹 소켓 API 규격에서는 원격 서버와의 양방향 통신을 가능하도록 하는 웹 소켓을 이용하는 웹 페이지를 가능하도록 API를 정의한다. 웹 소켓에 대한 규격은 IETF에서 표준화 작업을 진행 중에 있다.

라. Web Storage

웹 저장소 규격에서는 웹 클라이언트 내에 구조화된 키-값 쌍 데이터의 영구적 데이터 저장을 위한 API를 정의한다.

마. Web Workers

웹 워커 규격에서는 웹 애플리케이션 작성자가 메인

페이지 내에서 병렬적으로 스크립트 백그라운드 작업을 생성하여 실행할 수 있도록 하는 API를 정의한다. 이를 통해 장시간 실행되는 스크립트를 인터럽트 없이 수행 가능하도록 할 수도 있다.

바. File API

단말에서 내부 파일 접근을 위한 API로 현재 FileList와 Blog 처리, 디렉토리 시스템, Writer 등의 세 가지 API 규격을 각각 만들고 있다.

사. CORS

서로 다른 도메인 간의 웹 자원을 공유하는 웹 애플리케이션을 만들 수 있도록 하기 위한 접근 제어 및 권한 제어 표준이다.

Web Application WG의 작업 표준 현황에서 알 수 있듯이, 다수의 웹 애플리케이션 관련 규격들이 개발되고 있다. 특히 주로 HTML5와 관련하여 스토리지 처리, 백그라운드 처리, 소켓 처리, 비동기 데이터 처리 등과 같은 새로운 규격들이 개발 중에 있다는 사실에 비추어 앞으로 차세대 웹 애플리케이션의 기능과 형태에 많은 변화가 있을 것으로 예상된다.

3. Native Web App

최근 Native Web App이란 새로운 용어가 등장하기 시작하였는데, 이는 원격에 있는 웹 자원을 불러서 사용하는 것이 아니라, 로컬에 저장된 웹 자원에 기초하여 웹 응용이 동작한다는 개념에서 Native Web App이라고 불리우며, '위젯(widget)'과 같은 형태를 비롯하여 설치형 웹 응용들을 모두 포함하는 새로운 개념이다.

이 중에서도 가장 대표적인 형태가 위젯으로 위젯에 대해서는 아직도 다양한 정의와 인식의 차이들이 있기는 하지만, "사용자 기기 또는 모바일 단말에 다운로드하거나 설치할 수 있으며 간편히 쓸 수 있도록 만든 작

은 창(window) 형태의 응용" 개념으로 정의되고 있다. 위젯은 그 실행 유형과 구동 플랫폼, 구동 방식에 따라 다양한 유형으로 구분된다. 보통 웹 위젯은 웹 기술을 사용하여 구동되는 위젯 형태를 의미하며, 모바일 위젯은 모바일 단말에서 구동되는 위젯을 부른다. 물론 위젯이란 용어도 gadget, badge, module, webjitt, capsule, snippet, mini, flake 등과 같은 다양한 이름들로 불리기도 하지만, 최근에 와서는 대체적으로 위젯이란 이름으로 통일되고 있는 추세에 있으며, 그 유형도 웹 위젯으로 대표되고 있는 추세라 할 수 있다.

위젯 표준화에 대한 필요성은 2006년부터 제기되기 시작하였다. 위젯에 대한 관심이 높아지고 다양한 위젯 플랫폼들이 개발되면서, 위젯 플랫폼 간의 위젯 호환성을 높이고, 기 개발된 위젯 애플리케이션들을 공유하여 사용할 있도록 하자는 필요성이 제기되었기 때문이다. 예를 들어 A사가 개발한 위젯과 B사가 개발한 위젯 정의에 사용되는 마크업 언어가 틀리고, 구동 방법이 달랐기에 상호호환되는 동작을 할 수 없었기 때문이다.

이에 2007년부터 W3C의 웹 애플리케이션 WG를 중심으로 표준화 작업이 시작되었다. 초기에는 단순히 2개의 표준안(위젯 요구사항과 위젯 언어) 작성 계획으로 출발하였지만, 현재는 7개 문서로 나누어 현재 작업중에 있다.

- Widget 1.0: Widget Packaging & XML Configuration: 위젯 배포를 위한 패키징 및 환경 설정 규격(2011년 9월에 권고안 표준으로 제정)
- Widget 1.0: Widget Interface: 위젯 데이터 메타데이터 액세스를 위한 API 규격(LCWD, 2012년 권고안 제정 예정)
- Widget 1.0: WARP(Widget Access Request Policy): 위젯으로부터의 네트워크 액세스 제어를 위한 보안 모델 규격(CR, 2012년 권고안 제정 예정)
- Widget 1.0: Digital Signatures for Widgets:

안전한 위젯 리소스 배포를 위한 전자서명 규격(PR, 2012년 권고안 제정 예정)

- Widgets 1.0: Widget URIs: 위젯 URI 스킴 규격(WD, 2013년 제정 예정)
- Widgets 1.0: View Models Media Feature: 뷰 모델과 표현 모드에 대한 규격(PR, 2012년 권고안 제정 예정)
- Widgets 1.0: Updates: 위젯 버전 관리를 위한 규격(WD, 2013년 권고안 제정 예정)

앞으로 Widget 2.0에 대한 표준화를 추가로 시작할 것으로 예상되며, Widget 2.0에 대해서는 네이티브 웹 앱 모델로 단순한 위젯 형태가 아닌 보다 복잡한 설치형 웹 응용 형태를 기초로 하는 모델이 될 것으로 예상된다. 이러한 작업은 현재 생성을 준비 중인 Native Web Apps CG에서 진행될 것으로 예상된다.

4. 기타 신규 표준화

앞서 언급한 기술 이외에도 W3C에서는 보다 다양한 시스템 레벨 처리들이 가능할 수 있도록 하기 위해, 다음과 같은 새로운 WG를 만들어 관련 표준을 개발하고자 준비 중에 있다.

가. NFC API

비접촉식 무선 통신 기술인 NFC(Near Field Communication) 기술을 활용하여 NFC 태그 및 리더 기능, P2P 통신 등을 가능하도록 하는 API 개발을 위해 Web NFC API WG 생성을 준비 중에 있다[8].

나. System Application API

웹 운영체제를 지향하고 있는 삼성-인텔 타이젠, 모질라의 B2G(Boot To Gecko), webinos 등의 시스템 레벨 API들을 표준화하기 위해 System Application API WG 생성을 준비 중에 있다.

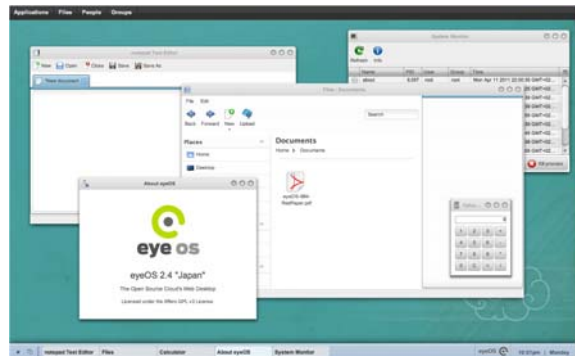
IV. 웹 운영체제 기술 동향

웹 기술이 발전함에 따라 다양한 기능과 서비스를 웹 응용으로 만들고 제공하기 위한 기술뿐 아니라, 웹 기술에 기반한 독자적인 운영체제(OS: Operating System) 환경을 만들고자 하는 여러 가지 시도들도 등장해왔다. 웹 기반 SW 플랫폼은 Native SW 플랫폼이나 자바 기반 SW 플랫폼에 비해 널리 확산되지는 않았지만, 응용 프로그램 개발자 확보 용이성, 크로스 디바이스 적용 용이성, 응용 확보의 용이성, 웹의 개방성, 글로벌 협력의 용이성, 웹 접속 디바이스의 급속한 증대 등으로 확대가 예상되고 있다[9].

웹 플랫폼 기술은 웹 서버 플랫폼 기술과 웹 클라이언트 플랫폼 기술, 그리고 웹 서비스 플랫폼으로 구성되며, 초기의 웹 운영체제는 웹 서버 플랫폼 기술을 중심으로 한 기술개발을 진행하였다면, 2단계 웹 운영체제에서는 웹 클라이언트 플랫폼과 웹 서비스 플랫폼을 발전시키는 방향으로 기술개발이 진행되고 있다.

1. 1단계 웹 운영체제(1994~2005년)

초기의 웹 운영체제의 개념은 웹 기반의 데스크톱 환경을 만들어 제공하는 것을 주요 목표로 하였다. (그림 4)와 같이 초기 web desktop 또는 webtop 환경은 브라우저 위에 웹 페이지를 통해 데스크톱 환경과 유사한 가



(그림 4) eyeOS 구동 화면

상의 데스크톱 페이지를 만들어 제공하고, 이를 통해 사용자가 웹 서비스들을 선택하고 이용할 수 있도록 하였다. 이러한 개념은 1994년 SCO를 통해 최초로 소개되었었고, 이후 eyeOS를 비롯하여 많은 서비스와 제품들이 등장하였다. 1단계 웹 데스크톱 기반의 운영체제는 1) 언제 어디서든 사용 가능하며, 2) 브라우저만 있으면 구동이 가능하며, 3) 단말 플랫폼에 무관하다는 특징을 장점으로 갖고 있었다. 그러나 반면에 1) 보안 취약성과 속도 문제, 2) 기존 데스크톱 환경과의 차별성 부족, 3) 오프라인 시 사용 불가능, 4) 응용 부족, 5) 차별성 부족 등의 여러 가지 단점과 한계를 나타냈다.

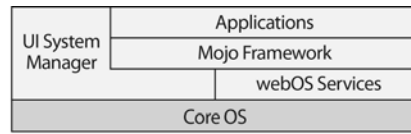
2. 2단계 웹 운영체제(2006년~현재)

2005년 웹2.0의 등장과 모바일 및 스마트폰의 급성장은 새로운 기회들을 만들었다. 이 중에서도 2009년 webOS의 등장은 웹 운영체제에 대한 개념과 생각을 바꾸는 결정적인 계기가 되었다. 과거에도 Microsoft 등은 웹 기술을 운영체제와 결합시키는 다양한 시도들을 하여 왔지만, 웹을 기반으로 하는 웹 응용 중심의 운영체제가 등장한 것은 최초였기 때문이다.

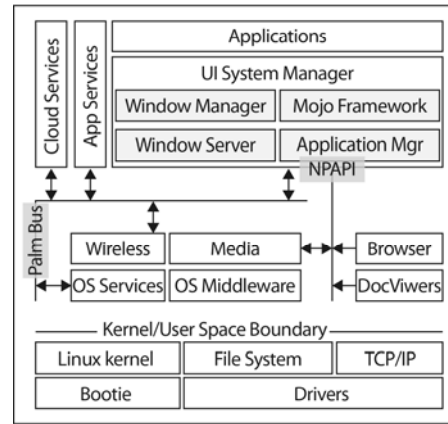
가. WebOS

(그림 5)와 같이 webOS는 리눅스 커널에서 구동되는 모바일 운영체제로 웹 2.0 기술들을 포함하면서 Mojo Framework를 기반으로 웹 언어를 기반으로 하는 모바일 응용들을 개발하고 활용할 수 있도록 하였다. 이러한 시도는 다양한 웹 운영체제에 대한 새로운 시도들을 촉발하는 계기가 되었고, 크롬 OS 등의 등장으로 이어졌다.

WebOS의 주요 특징은 모바일 단말을 대상으로 하는 플랫폼으로 웹 언어를 사용하여 프로그래밍을 한다는 혁신적인 개념과 함께 웹 응용을 핵심으로 하는 것으로 하였다. 그러나 다른 치열한 다른 모바일 플랫폼들과의 경쟁 속에서 이러한 웹 응용을 중심으로 한다는



(a) Simplified WebOS Architecture



(b) WebOS System Architecture

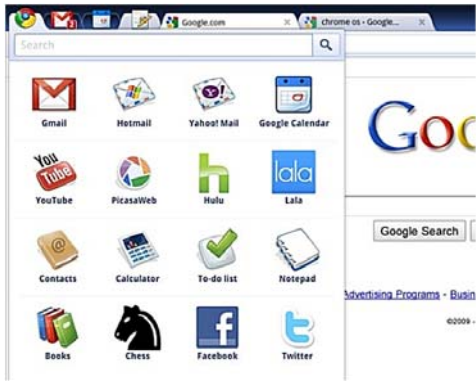
(그림 5) WebOS 아키텍처 구성도[10]

특징만으로 경쟁을 하기란 쉽지 않다는 점을 보여주었다.

나. Chrome OS

구글 크롬 OS는 2009년 오픈소스 프로젝트인 Chromium OS 프로젝트를 기반으로 만들어져 2010년 12월에 발표된 리눅스 기반의 운영체제이다. 크롬 OS는 자바 기반의 플랫폼인 구글 안드로이드와는 별개로 개발되고 있으며, 북 사양과 유사한 크롬북 단말을 대상으로 하고 있다.

구글 크롬 OS의 아키텍처는 3-tier 아키텍처 모델로 펌웨어, 브라우저 및 윈도우 매니저, 시스템 레벨 소프트웨어 및 유저랜드 서비스로 구성된다. 이처럼 크롬 OS는 리눅스를 기본 운영체제로 사용하고 있으며 크롬 브라우저를 최적화시켜 기본 인터페이스로 채택하고 있다는 점과, (그림 6)에서처럼 webOS와는 달리 브라우저를 기반으로 하는 thin client 구조를 지향하고 있으며, 구글 클라우드와의 연동을 핵심으로 하고 있다는 점을 특징으로 한다.



(그림 6) 구글 Chrome OS 구동 화면

구글 클라우드를 기반으로 한 구글 크롬 OS의 주요한 장점들로는 1) 모든 주요 자료의 클라우드 저장, 2) 저사양 단말에서도 구동 가능, 3) 빠른 부팅 속도, 4) 구글 클라우드를 기반으로 한 일관된 사용자 환경 제공 가능, 5) 지속적으로 자동 업데이트되는 운영체제를 꼽고 있다. 반면 항상 네트워크 기반의 접속을 해야 하므로 1) 오프라인 사용이 불가능하며, 2) 특정한 단말(넷북)에서만 구동 가능하다는 단점을 갖고 있다.

다. Webinos

2010년 9월부터 2013년까지 EU FP7 산하의 프로젝트로 진행되고 있는 webinos 프로젝트는 다양한 커넥티드(스마트폰, 모바일 기기, TV, 자동차 등) 단말 위에서 안전하고 견고한 웹 애플리케이션과 서비스를 이용하고 공유하는 것이 가능할 수 있도록 하는 오픈소스 플랫폼을 개발하는 것을 목표로 하고 있다. 최종적으로는 산학연이 참여하는 오픈소스 커뮤니티로 확대 발전을 목표로 하고 있다[11].

현재 Telecom Italia, Docomo 등 4개 통신사, 삼성전자와 소니에릭슨, BMW 등 3개 제조사를 비롯하여 15개의 연구기관 및 대학교 등이 참여하고 있다. 현재 게임, 소셜 서비스 등을 구현하는데 필요한 기기 지원, 자원 공유, 통신 기능, 보안 모델 등에 초점을 맞춘 연구 개발을 진행하고 있다.

라. 타이젠(Tizen)

2011년 삼성과 인텔의 주도로 만들어진 Tizen Association을 통해 개발되고 있는 운영체제로, 과거 LiMo 운영체제를 이어받은 삼성 SLP(Samsung Linux Platform)와 Moblin, Maemo 등의 경험을 이어받은 인텔의 MeeGo 운영체제 기술을 기반으로 좀 더 진화된 새로운 운영체제 개발을 목표로 하고 있다. 타이젠에서는 HTML5 기반의 웹 응용 프레임워크를 핵심 요소로 태블릿, 노트북, 스마트폰, 스마트TV 및 차량용 시스템 등에서 활용할 수 있는 플랫폼 개발을 목표로 하고 있다. 2012년 4월 30일에 타이젠 1.0 버전을 공개하였고 2012년 Q2에 실제 타이젠이 탑재된 단말을 출시할 계획으로 있다[12].

마. B2G Project

오픈소스 커뮤니티인 모질라(Mozilla)에서 추진하고 있는 오픈소스 기반의 웹 운영체제 개발 프로젝트이다. 2011년 7월 개방형 웹을 위한 완벽한 독립 운영체제 개발을 목표로 하면서, HTML5 애플리케이션을 기반으로 모바일 단말의 다양한 HW 기능들을 제어할 수 있도록 하고, 웹 기반의 애플리케이션 운영 환경을 제공하고자 하는 것을 목표로 하고 있다. UI 모듈인 Gaia, 애플리케이션 런타임인 Gecko 엔진, 하위 레벨 운영체제 시스템인 Gonk로 구성되어 있다. 2012년 하반기에 브라질 등에서 실제 B2G 플랫폼을 탑재한 상용 단말을 출시하면서 그 기능성들을 검증하고자 하고 있다[13].

V. 결론

지금까지 웹 응용 기술 및 웹 플랫폼 기술에 대한 기술 및 표준화 동향에 대해 살펴보았다. 웹 기술은 편의성, 응용성, 호환성, 보편성, 확장성을 확보하기 위해 끊임없이 진화를 해오고 있다. 과거 웹 기술이 단지 브라

우저와 서버 기술에만 초점을 맞추었다면, 최근에는 HTML5를 중심으로 리치 웹 애플리케이션 기술, 메시업 기술, 단말 제어를 위한 시스템 API 기술, 서비스 플랫폼 기술, 웹 운영체제 기술들과 같은 플랫폼을 지향하는 다양한 기술개발 및 제품개발이 진행되고 있다.

웹 플랫폼으로의 진화가 갖는 중요한 의미는 미래 플랫폼 경쟁력에 있다고 할 수 있다. 즉, 최근 급변하는 모바일 환경에서 볼 수 있듯이, 플랫폼 주도권과 경쟁력은 기업의 흥망성쇠와 미래를 좌우하는 중요 요소가 되었다. 구글과 애플에 의해 주도되고 있는 생태계, 노키아의 급속한 몰락, 전체 휴대폰 매출의 4%를 차지하는 애플이 전체 휴대폰 산업 수익의 50%를 차지하는 사건 등은 플랫폼이 곧 핵심 경쟁력이란 점을 증명하는 사건이라고 할 수 있다. 그리고 기존 사업자가 단말기, 서비스, 콘텐츠 공급 등을 주도하며 산업 전반에 영향력을 행사하던 시대에서, 플랫폼 사업자에 의해 생태계가 결정되고 주도되는 플랫폼 중심의 시대로 변화되었다는 것을 의미하기도 한다.

이러한 플랫폼 중심의 시대에는 좀 더 세밀한 플랫폼 전략(플랫폼 창조 전략, 플랫폼 활용 전략, 플랫폼 방어 전략)이 요구된다. 플랫폼 전략이란 관련 있는 다양한 그룹들을 ‘장(플랫폼)’에 모아 플랫폼을 진화시키며 새로운 사업의 생태계를 창조하는 전략을 말하는 것으로, 그룹들 간의 교류를 촉진시키고, 협력을 통한 비용을 감소시키며 네트워크 효과를 증폭시키기 위한 목적을 갖는다.

웹 플랫폼이 플랫폼 전략으로서 주목받는 이유는 네 가지이다. 첫째는, 파편화(fragment)되는 다중기기 환경에서 장치독립적이며 상호호환성을 갖는 웹 플랫폼은 큰 장점을 갖고 있기 때문이다. 둘째는, 특정 기업에 종속된 기술이 아니고 비교적 특허와 지적재산권 문제에서 자유롭기 때문에 누구든 자유롭게 사용하고 확장시키며 기회를 만들 수 있기 때문이다. 셋째는, 현재 플랫폼 체계모니를 쥐고 있는 구글, 애플, 마이크로소프트의

영향력에서 벗어날 수 있는 대안이기 때문이다. 넷째, 개방적 기술로 손쉽게 다른 개방형 기술들을 결합시켜 플랫폼화시키고 발전시킬 수 있기 때문이다.

지금까지 살펴본 웹 플랫폼의 진화와 관련하여 우리는 몇 가지 중요한 시사점을 정리해 볼 수 있다. 우선 첫째, 웹 기술의 핵심 경쟁력이 개방성과 확장성, 상호호환성에 있었던 것처럼 웹 플랫폼의 핵심 경쟁력도 개방성, 확장성, 상호호환성에 있어야 할 것이다. 즉, 표준 웹 기술만을 사용하면 단말/플랫폼/제품의 차이에 상관없이 여전히 일관되게 동작될 수 있어야 할 것이다.

둘째, 좀 더 진화된 웹 UI/UX 기술에 대한 고민이 필요하다라는 점이다. 과거 웹 데스크톱을 비롯하여 웹 운영체제의 시도에서 경험하였듯이, 진일보된 웹 UI/UX를 제공하지 않고서는 기존 플랫폼과 경쟁하기 어려움이 많다는 점이다. 그러므로 새롭게 변화되고 있는 단말과 응용 환경에 적합한 새로운 UI/UX에 대한 고민과 함께, 웹 애플리케이션에서의 효과적인 상호작용과 인터페이스 방식을 효과적으로 제공할 수 있는 웹 플랫폼 기술에 대한 새로운 많은 연구개발들이 필요하다.

셋째, 아직 웹 플랫폼 기술 안정화를 위해서는 보다 많은 원천기술 연구가 필요하다라는 점이다. 앞으로 2014년 HTML5와 웹 애플리케이션 표준화가 완료되고 다양한 웹 응용들이 등장하게 되면 웹 응용 플랫폼의 중요성과 안정화가 더욱더 중요한 이슈가 될 것으로 예상된다. 그러므로 미래 웹 플랫폼 경쟁력 확보를 위해서는 해외와 같이 핵심적인 웹 플랫폼 기술 및 표준에 대한 보다 적극적인 연구개발과 투자가 필요하다.

지난 20년 동안 웹 기술은 슬로우 스타터(slow starter)로 환경의 변화에 적응하면서 콘텐츠와 서비스를 연결하는 기술로 꾸준한 기술 발전을 진행시켜왔다. 이제 앞으로 10년간의 웹은 더욱더 많은 플랫폼으로서의 역할을 하게 될 것이며, 더욱더 많은 곳에서 사용자들과 만나게 될 것으로 전망된다. 그리고 그 선두에는 HTML5가 있다.

용어해설

HTML5 하이퍼텍스트 마크업 언어 버전 5. 웹을 표현하기 위한 가장 기본 언어로 다양한 웹 양식들과 시맨틱, 멀티미디어 요소들을 효과적으로 표현할 수 있도록 하였으며, 다양한 웹 기술들을 손쉽게 통합할 수 있도록 하고 있다.

웹 플랫폼 웹 기반 애플리케이션을 개발하고 구동할 수 있도록 하는 SW 환경으로, 단순히 플랫폼 독립적인 런타임이나 브라우저만으로 구성되는 경우와 커널까지 포함하는 운영체제 형태로 구분된다. 이밖에도 서버 기준의 환경은 서비스 플랫폼으로 별도 분류한다.

약어 정리

B2G	Boot To Gekco
CORS	Cross-Origin Resource Sharing
DAP	Device API and Policy
XHTML	Extensible Hypertext Markup Language
HTML	HyperText Markup Language
HTTP	HyperText Transfer Protocol
NFC	Near Field Communication
OS	Operating System
SLP	Samsung Linux Platform
URL	Unified Resource Locator
WARP	Widget Access Request Policy
W3C	World Wide Web Consortium
XHR	XMLHttpRequest

참고문헌

- [1] Netcraft. <http://news.netcraft.com/archives/2012/06/06/june-2012-web-server-survey.html>
- [2] 전종홍, 이승윤, “웹 2.0 기술 현황 및 전망,” 전자통신동향분석, vol. 21, no. 5, 2006. 10.
- [3] Apple Insider, “Why Apple is betting on HTML 5: a web history,” Sept. 2009. <http://is.gd/w9S9a9>
- [4] 전종홍, 이승윤, “차세대 모바일 웹 애플리케이션 표준화 동향,” 전자통신동향분석, vol. 25, no. 1, 2010. 2.
- [5] EU FP7 MobiWebApp Project report, “Standards for Web Applications on Mobile: May 2012 current state and roadmap,” May 2012. <http://www.w3.org/2012/05/mobile-web-app-state>
- [6] W3C Device APIs WG. <http://www.w3.org/2009/dap/>
- [7] W3C Web Application WG. <http://www.w3.org/2008/webapps/>
- [8] W3C NFC API WG. <http://www.w3.org/2012/05/nfc-wg-charter.html>
- [9] 이경희 외, “웹기반 SW 플랫폼 기술동향,” 전자통신동향분석, vol. 26, no. 5, 2011. 10.
- [10] HP, Overview of webOS-Plam webOS Architecture. <http://is.gd/a5Kv0w>
- [11] webinos project. <http://www.webinos.org>
- [12] Tizen. <https://www.tizen.org/>
- [13] Mozilla B2G. <https://wiki.mozilla.org/B2G>