La Web como plataforma

Pablo F. Iglesias
contacto@pabloyglesias.com
Julio del 2014 (reeditado en Noviembre 2014)

Extracto

En un entorno fuertemente competitivo como es el del desarrollo móvil, analizamos la situación actual de la web como plataforma, y sus respectivas propuestas en forma de sistemas operativos del mercado.

También haremos un recorrido por las tendencias de aquí a unos años, las nuevas prestaciones y sus ventajas competitivas frente al desarrollo nativo de sistemas operativos convencionales.

Destacado

HTML5, web, internet, webapps, móvil, sistema operativo, SO, CSS3, Tizen OS, Firefox OS, Web OS, Android, iOS, Blackberry 10, Windows Phone 8, Sailfish OS, Ubuntu.

Índice

1. Introducción

La informática, y por ende, la electrónica de consumo, ha sufrido una fuerte evolución en las últimas dos décadas con la democratización de los sistemas operativos móviles. Para que tal hazaña haya podido llevarse a cabo, ha sido necesario esperar a que la industria fuera capaz de miniaturizar los componentes clásicos de un sistema operativo, aumentando la autonomía de las baterías y su optimización con el resto del hardware existente. La llegada de una nueva generación de procesadores basados en tecnología ARM ha supuesto, al menos hasta estos últimos años, un estándar en cuanto a comunicación de bajo nivel, surgiendo plataformas basadas en este paradigma.

La figura del smartphone y el tablet se empieza a desdibujar, y en este estudio hablaremos ya no solo de los dos grandes ganadores de la nueva electrónica de consumo, sino en su especialización como dispositivos permanentemente conectados (wearables), y su implantación en objetos tradicionalmente no inteligentes (IoT).

Un sector en auge, que ha llevado a las grandes compañías tecnológicas del siglo XX a evolucionar, virando el modelo de negocio (como el caso de IBM) o directamente desapareciendo frente a nuevas figuras nacidas o adaptadas a la era de Internet (Google, Samsung, Mozilla).

Un recorrido por las entrañas de los sistemas operativos móviles actuales, por el mercado estimado, por su paulativa disgregación y por la importancia que los lenguajes web están tomando dentro y fuera de este ecosistema.

2. Arquitectura de un sistema operativo móvil

Un sistema operativo no es más que un conjunto de órdenes y programas capaces de abstraer la lógica de comunicación con el hardware, de tal manera que otros programas pueden funcionar por encima y ofrecer el acceso a la información necesaria para la interacción del usuario con la máquina.

Los sistemas operativos móviles heredan esta funcionalidad en dispositivos habitualmente menos potentes, sacrificando algunas prestaciones a cambio de una mayor autonomía o un tipo de uso más específico.

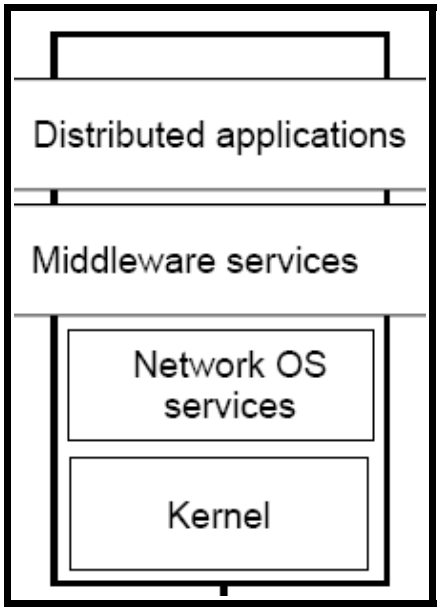
Sin embargo, una fuerte jerarquía de permisos y la dependencia habitual de una plataforma de servicios han permitido que la informática evolucione en los dispositivos móviles de forma distinta a la que lo ha hecho en el escritorio. Los sistemas operativos móviles actuales están fuertemente capados de cara a la interacción del usuario. Los mercados de aplicaciones hacen de intermediarios, abstrayendo las implicaciones técnicas respectivas a la seguridad de los datos y las posibles vulnerabilidades en éstos. El usuario, por tanto, pasa a ser un consumidor de herramientas, liberándose de la responsabilidad de su buena gestión, y perdiendo por el camino parte del control que anteriormente ostentaba.

Esto ha permitido una verdadera disrupción en el sector. Por un lado, el usuario es consumidor de aplicaciones de terceros, que a fin de cuentas, son las que de verdad ofrecen valor al sistema operativo. Y por otro, la propia comunidad de desarrolladores, que antiguamente estaba ligada única y exclusivamente a proyectos abiertos, ha favorecido el auge de un mercado acotado para cada sistema operativo, gestionado por la compañía detrás del mismo, y que en definitiva, acaba siendo el pilar monetario de todo el ecosistema.

Los sistemas operativos móviles de la actualidad dependen de la comunidad de desarrolladores en su plataforma, que con sus aportaciones, atraen a un mayor o menor número de usuarios a la misma.

El acceso a procesos propios del sistema se hace de forma escalar, mediante permisos, por lo que se mantiene esa aparente democratización de desarrollos (cualquiera puede hacer una aplicación y subirla al market), y por otro, se gestiona de forma restrictiva la interacción con el sistema, que pasa a ser una base operativa donde el usuario lanza los diferentes servicios.

Atendiendo a su arquitectura modular, encontraremos semejanzas con la mayoría de sistemas operativos de escritorio.



2.1. Núcleo

También llamado Kernel, es la capa de software encargada de la comunicación directa con el hardware. Hablamos de la capa más baja, y por tanto, su misión es comunicar las peticiones de las capas superiores a los componentes físicos, devolviendo acceso a uno u otro controlador, administración de la memoria, sistema de archivos y un largo etcétera.

Su existencia no está únicamente ligada a los sistemas operativos móviles, sino en definitiva a cualquier sistema operativo de la actualidad.

Así, encontramos que muchos de los SO móviles presentes en el mercado utilizan un kernel Linux (Android, iOS (UNIX),...) al que se han eliminado componentes superfluos para el uso que se le va a dar.

Debido a la gran acogida de Android, han surgido no pocos fork de su kernel, dando como resultado versiones distintas de Android (Fire OS de Amazon, por ejemplo) o auténticos nuevos sistemas operativos (Firefox OS de Mozilla o Tizen OS de Samsung).

Dentro del ecosistema de kernels móviles, encontramos también versiones propietarias, como la de Windows Phone y Blackberry.

2.2. Middleware

Por encima del kernel solemos tener el middleware, una capa que se encarga de comunicar los procesos de bajo nivel con los de alto. Así, esta capa gestiona los motores de comunicación y mensajería, servicios con los que cuenta el dispositivo, codecs multimedia y un largo etcétera.

También es la encargada, habitualmente, de gestionar los permisos y discriminar entre peticiones, siguiendo con la arquitectura modular restrictiva de los sistemas operativos móviles. De esta manera, cada aplicación tendrá acceso únicamente a los permisos que haya pedido de antemano, pudiendo fijarse políticas de validación y diferentes sistemas de testing preliminar (de cara a permitir su instalación desde un market oficial).

2.3. Entorno de ejecución de aplicaciones

Hablamos de una capa encargada de suministrar los componentes necesarios para que

funcionen los desarrollos de terceros (aplicaciones). Es muy dependiente, por tanto, del lenguaje nativo del sistema, y de la libertad que se otorgue a la comunidad (mediante APIs).

Cuenta además con un gestor de aplicaciones, una herramienta fundamental para la coexistencia y uso de diferentes servicios por parte del usuario.

2.4. Interfaz de usuario

Todos los sistemas operativos deben contar con una interfaz de usuario. Se trata de la capa con la que el cliente interacciona.

Se sirve para ello de gestos y elementos gráficos (botones, formularios, menús,...) que permiten al usuario utilizar los servicios desarrollados sin recurrir a una consola de comandos.

Cuenta además con un nutrido grupo de servicios y aplicaciones que vienen instaladas por defecto, ya sean propias del sistema operativo (agenda, galería de fotos, calendario,...), desarrolladas por una empresa como propuesta de valor (Google apps o iApps, por poner dos ejemplos) o instaladas en el dispositivo como acuerdos con operadoras o demás stakeholders del mercado (Dropbox en los Samsung, Vodafone Cloud en terminales comprados a la operadora).

La experiencia de usuario es distinta según el sistema operativo que se use, y va evolucionando con el tiempo. Los sistemas operativos de primera generación apenas contaban con gestos, teniendo que usar herramientas tales como teclado físico para su interacción. Con la masificación de las pantallas táctiles, hemos vivido una migración de lo físico a lo digital, hasta el punto que la mayoría de interacción en sistemas operativos de segunda generación se hace mediante botones digitales. Desde hace unos pocos años, los sistemas operativos están cada vez más adquiriendo gestos para su comunicación, lo que apunta a un futuro no muy lejano en el que buena parte de su uso esté enfocado en este paradigma y no en la digitalización del botón.

3. Mercado

Negar la evidencia es absurda. Los sistemas operativos móviles, unidos al veloz auge de los smartphones (y en menor medida de las tablets) han desplazado la importancia histórica del escritorio.

La mayoría de personas del primer mundo deciden invertir su dinero en renovar sus dispositivos móviles antes que los de escritorio. Incluso para según qué usos, la tablet ha empezado a ofrecer más valor que un portátil, siendo la elección por defecto de un sector cuyo uso de la informática pasa por el consumo de contenido multimedia y navegación por internet/redes sociales.

Así pues, vivimos una verdadera revolución tecnológica. La electrónica de consumo de nuestra era se adapta a nuestras necesidades como nunca antes lo ha hecho. Es cada vez más intuitiva, menos técnica (al menos de cara al usuario) y más práctica. Internet ha permitido liberar la información, que sea accesible por cualquiera allí donde esté, lo que sin duda ha cambiado los hábitos de la sociedad frente a la tecnología. De un uso inmersivo, programado, pasamos a un uso continuo, despreocupado. La información útil llega a nosotros mediante notificaciones, pequeñas píldoras informativas que nos avisan de que algo puede sernos interesante, por lo que la consulta sistemática queda relegada a esos momentos de necesidad, o aquellos tiempos muertos entre otras acciones.

Todo esto se ve reflejado, como cabría esperar, en el incremento sustancial del mercado móvil, que ya el año pasado superó al de escritorio, y que hoy en día sigue siendo dueño y señor del sector.

Tanto es así que gigantes como Samsung, cuyo nicho de mercado estaba bastante alejado de la informática de usuario (electrodomésticos, construcción e inversión, principalmente) han pasado en apenas un lustro a dominar el sector. Y otros, fuertemente pegados a la figura del escritorio, han sabido mover sus fichas para llegar a ser de las empresas más rentables de la historia (Apple).

Sin embargo, no todo lo el que toca el sector se vuelve de oro, y para ejemplo, la lamentable situación que desde unos años aborda Blackberry (anteriormente llamada RIM), o la pérdida de dominio de una Microsoft que llegó antes que nadie al sector, y no supo sino hasta recientemente mover ficha para posicionarse en el mismo.

En este apartado, por tanto, abordaremos algunos de los últimos estudios de mercado de la industria. Diferentes estudios con diferentes sesgos aplicados, que en unión, nos permiten observar algunas tendencias, haciendo hincapié, como cabría esperar por el título de esta investigación, en la importancia de la web en este ecosistema.

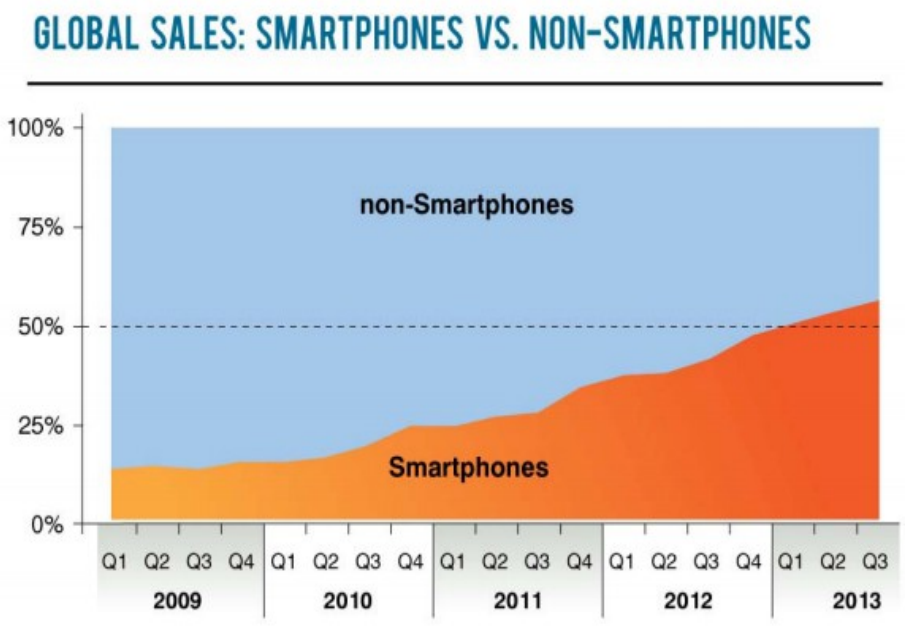
Puesto que este artículo se publica en Julio de 2014, la mayoría de estudios analizados tienen como objetivo representar una foto del mercado del Q1 de este año (la mayoría publicados entre Febrero y Mayo).

3.1. Desarrolladores

El primer estudio analizado proviene de la empresa VisionMobile, que como cada año, liberaba recientemente la sexta edición de Developers Economic, un recorrido por la situación del sector a partir del interés mostrado por los desarrolladores.

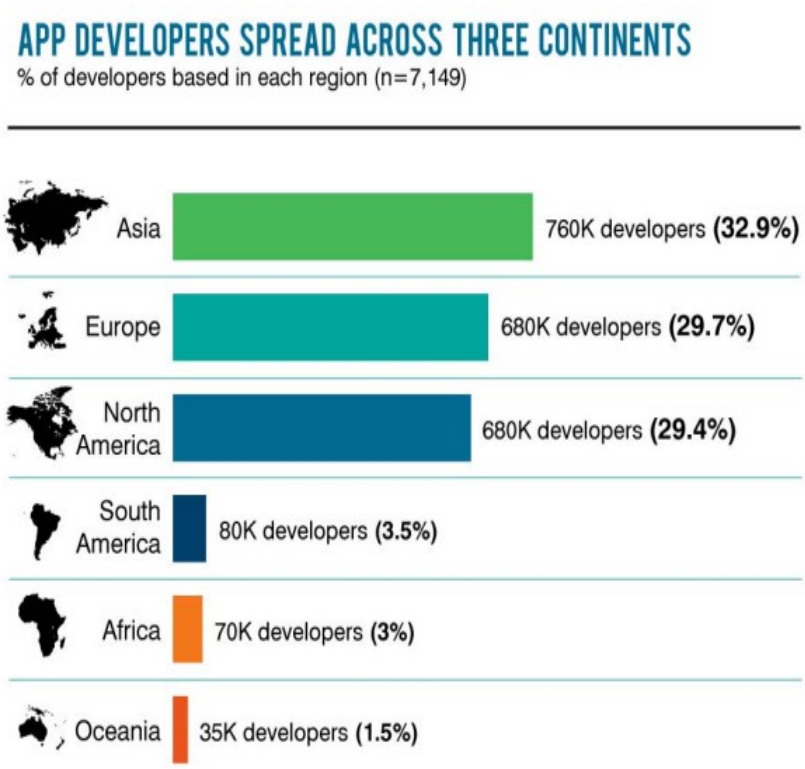
Para generar el conocimiento necesario, realiza encuestas a más de 7.000 trabajadores en más de 127 países, por lo que podemos considerarlo un baremo bastante acertado. Además, está patrocinado por compañías de la talla de Mozilla, Intel, Canonical o Blackberry (entre otras muchas), por lo que se espera un sesgo bastante poco acentuado.

La proyección económica del sector en el 2013 fue de **68 billones de dólares**, con previsión de aumentar hasta los 143 billones en 2016. Hablamos por tanto de un mercado con una fuerte tendencia a la alza, que sigue su expansión, y que aún está en pañales comparado con lo que se nos avecina.



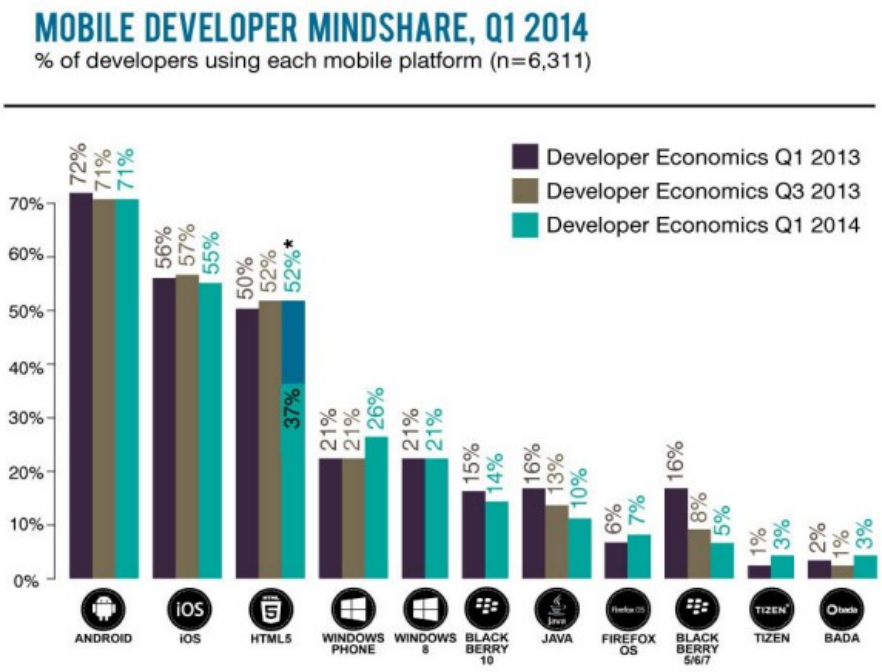
En la gráfica superior podemos ver la evolución que ha tenido el sector del smartphone respecto a los dispositivos móviles de primera generación. Ya a principios de 2013 se superó la barrera del 50%, acabando en un 55%, y llegando al 60% en el primer Q1 de este año.

Se confirma por tanto la tendencia a la desaparición de los no-smartphones, cuya principal presencia la ostenta Bada y versiones antiguas de Blackberry, principalmente en Sudamérica, África y Oceanía, y en donde sistemas operativos de bajo costo como Android (Samsung acababa el año con un 31% del sector) o Firefox OS están acelerando la transición.



Atendiendo a la evolución de su comunidad de desarrollo, Asia aglutina el 32,9% del mercado, superando a Europa y Norte América (ambas con un 29,X%) en presencia, lo que pone de manifiesto y realza la figura del gigante asiático como principal exportador y consumidor mundial de tecnología, desplazando las potencias históricas del sector, y adelantando la pérdida de poder económico ya no solo de Europa sino de EEUU frente a China.

Tanto Sudamérica como África y Oceanía tienen una presencia minoritaria, debido principalmente a que muchos de sus países están en vías de desarrollo. Sin embargo, me parece interesante señalar que si bien África y Oceanía tienen aún un largo camino por recorrer, la tendencia del mercado, y los movimientos tecnológicos de estos últimos años giran en torno a una Sudamérica cada vez más avanzada tecnológicamente, que sale ya del yugo de las primeras Blackberry, y empieza en nuestros días a dar el salto al mundo smartphone, apoyados como están en el mercado de gama baja/media.



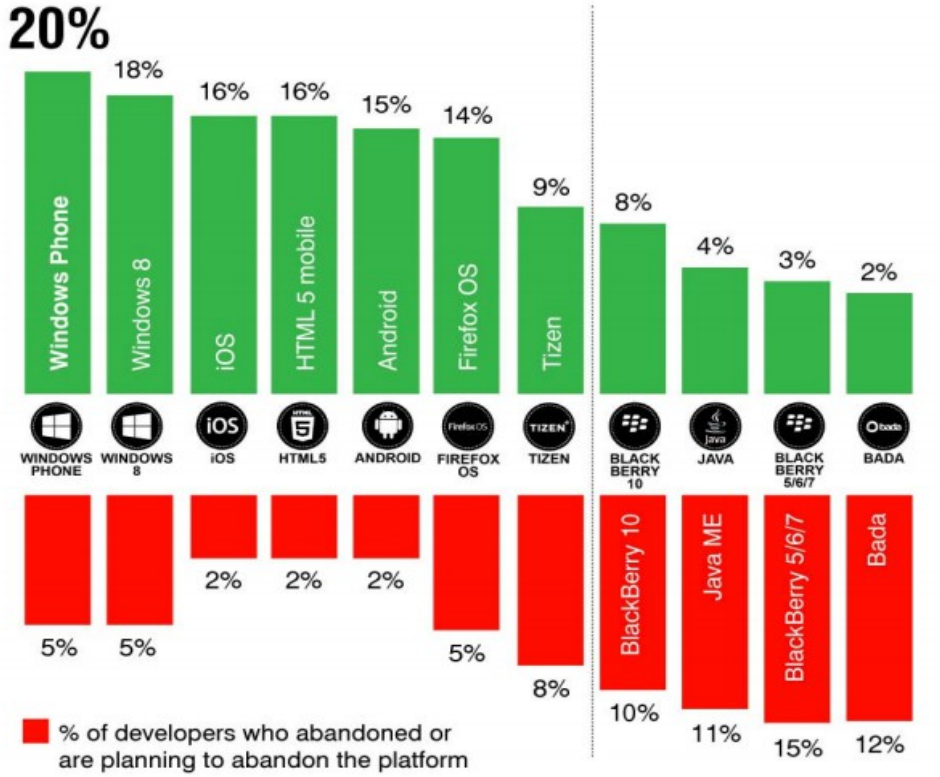
*This figure includes developers who develop hybrid apps and apps developed with HTML5 but translated to native code.

Respecto a la evolución de los sistemas operativos móviles según su interés por parte de los desarrolladores, podemos observar tres claros ganadores: Android, iOS y HTML5.

Android se mantiene como principal motor del sector, perdiendo apenas un 1% de

presencia en favor de las nuevas propuestas, situación que comparte con iOS. En el otro lado, está HTML5, que ha aumentado un 2% desde el Q1 del año pasado, teniendo en cuenta que este porcentaje se calcula no únicamente respecto a aplicaciones desarrolladas para plataformas de HTML5, sino también para aquellos sistemas que permiten aplicaciones híbridas (por tanto, engorda además el porcentaje del resto de plataformas).

Blackberry pierde un 1% de presencia con su nueva versión (BB10) y un 9% con las versiones antiguas, permitiendo a Windows Phone ascender en torno al 5% en su versión ARM, a Firefox OS un 1% (teniendo en cuenta que aún no ha despegado en la mayoría de mercados presentes en el roadmap), un 2% para Tizen (apoyado por el motor que representa Samsung en el sector) y un curioso 1% de Bada, un SO que se niega a desaparecer.



En este gráfico podemos comparar el interés mostrado por los desarrolladores de adentrarse en una plataforma (en verde) frente al porcentaje de abandono de la misma (rojo).

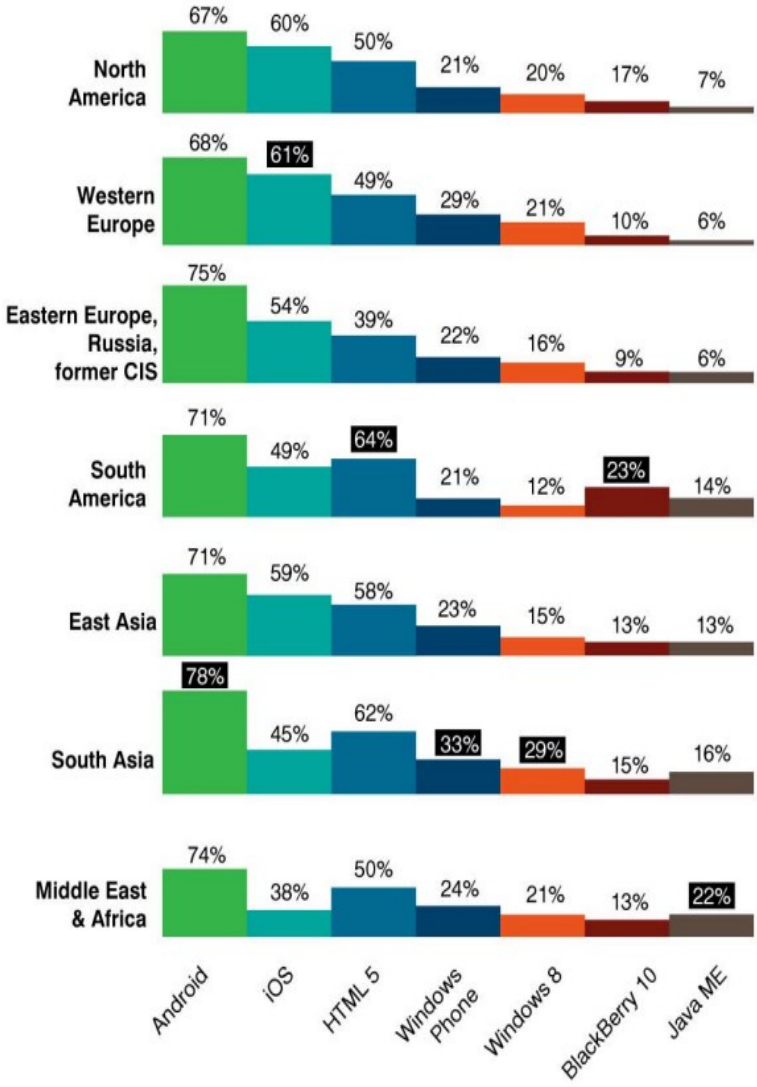
Queda por tanto claro que este año Microsoft con la nueva propuesta (cada vez más convergente) de Windows Phone y Windows 8 levanta el mayor interés en la comunidad, con un 20 y 18% respectivamente, y apenas un 5% de abandono.

Lo siguen iOS, HTML5 y Android, con un 16%, 16% y 15% y un 2% de tasa de abandono, Firefox OS (aumento hasta el 14% de interés, frente al 5% de abandono), Tizen (9% de interés frente a 8% de abandono) y los sistemas operativos en vías de desaparición (Blackberry, Bada y aplicaciones Java).

Se quedaron fuera del estudio dos plataformas (al menos) que a un servidor le hubiera gustado ver representadas: Blackberry y Jolla Sailfish OS.

El primero, después de unos 18 meses de desarrollo, aún no ha llegado a despegar, lo cual lleva a pensar que aunque con una propuesta verdaderamente disruptora (un terminal móvil que conectado a una pantalla y un teclado podría funcionar como sistema de escritorio) todavía es demasiado pronto.

El segundo, basado en Meego, propone una vuelta de tuerca a la interacción vía gestos, y aunque tuvo bastante repercusión a finales del 2013, no se ha vuelto a saber mucho más de él.



Continuamos con la presencia de los diferentes SO según la región.

A considerar, Android gana con mayor o menor fortuna en todas las regiones, siendo el sur de Asia donde más índice de penetración tiene (algo esperable teniendo en cuenta que tanto Samsung como HTC, Sony, XIAOMI y el resto de fabricantes chinos dominan el sector).

IOS encuentra su mejor aliado en el oeste de Europa, mientras HTML5 lo hace en Sudamérica, Windows Phone en el sur de Asia.

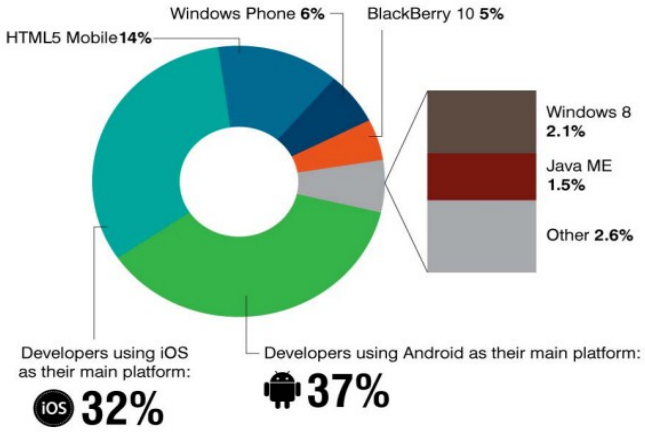
Para terminar, y como ya es habitual, Blackberry sigue siendo fuerte en Sudamérica y Java en África.

Si nos centramos en la distribución de SO por mercados de cada país, Kantar liberaba a principios de año un estudio semejante, cuya gráfica (en la siguiente página) arroja algunas particularidades que en esta primera no se pueden observar.

Así, vemos que el caso de España no sirve como ejemplo de la situación del mercado Europeo, con un Android que básicamente monopoliza el sector, asfixiando al resto de propuestas.

El brutal descenso de iOS en Italia en apenas un año (10,3% absorbido por Windows Phone) y Android, situación que también ha vivido América Latina con la bajada de los precios de dispositivos Android y el retroceso de Blackberry.

iOS sale perjudicado en todos los mercados estudiados por Kantar, manteniendo, eso sí, una fuerte presencia en EEUU (43% del mercado). También es interesante recordar que la cartera de dispositivos iOS se cierra únicamente a la cartera de una compañía (Apple), mientras que el resto de sistemas operativos cuentan con diversos fabricantes (mención especial a Android), por lo que pese a su reducción paulativa en el mercado, es digno de consideración.



Germany	3 m/e Dec 2012	3 m/e Dec 2013	% pt. Change
Android	69.0	75.4	6.4
BlackBerry	1.1	0.5	-0.6
iOS	21.7	17.3	-4.4
Windows	3.4	5.9	2.5
Other	4.8	0.9	-3.9

GB	3 m/e Dec 2012	3 m/e Dec 2013	% pt. Change
Android	54.4	54.9	0.5
BlackBerry	6.4	3.2	-3.2
iOS	32.4	29.9	-2.5
Windows	5.9	11.3	5.4
Other	0.9	0.6	-0.3

France	3 m/e Dec 2012	3 m/e Dec 2013	% pt. Change
Android	61.0	65.9	4.9
BlackBerry	5.1	1.6	-3.5
iOS	23.7	20.3	-3.4
Windows	5.0	11.4	6.4
Other	5.1	0.8	-4.3

Italy	3 m/e Dec 2012	3 m/e Dec 2013	% pt. Change
Android	54.2	66.2	12.0
BlackBerry	2.6	1.8	-0.8
iOS	23.1	12.8	-10.3
Windows	12.7	17.1	4.4
Other	7.4	2.1	-5.3

Spain	3 m/e Dec 2012	3 m/e Dec 2013	% pt. Change
Android	85.9	86.2	0.3
BlackBerry	2.4	0.2	-2.2
iOS	7.3	6.7	-0.6
Windows	1.2	5.6	4.4
Other	3.2	1.3	-1.9

USA	3 m/e Dec 2012	3 m/e Dec 2013	% pt. Change
Android	46.2	50.6	4.4
BlackBerry	0.9	0.4	-0.5
iOS	49.7	43.9	-5.8
Windows	2.4	4.3	1.9
Other	0.8	0.8	0.0

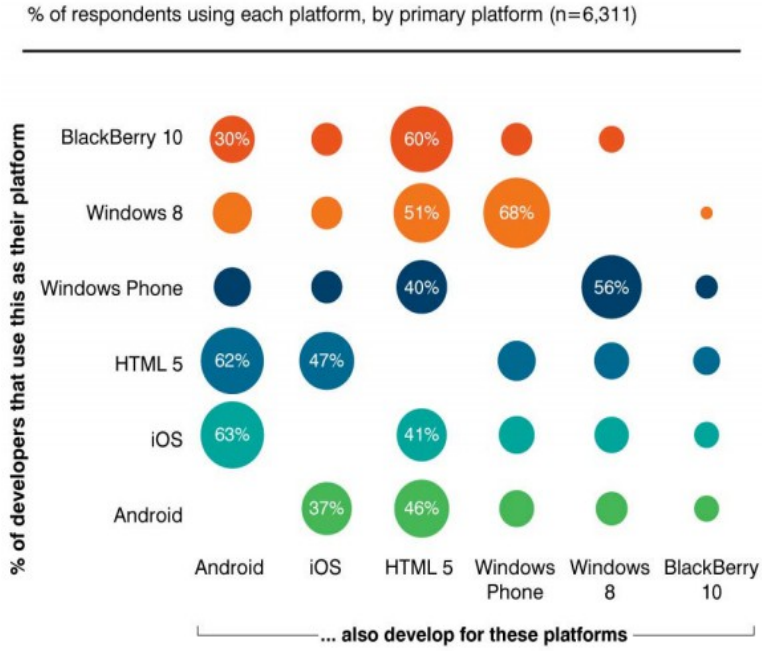
China	3 m/e Dec 2012	3 m/e Dec 2013	% pt. Change
Android	73.7	78.6	4.9
BlackBerry	0.0	0.1	0.1
iOS	21.2	19.0	-2.2
Windows	0.9	1.1	0.2
Other	4.2	1.3	-2.9

Australia	3 m/e Dec 2012	3 m/e Dec 2013	% pt. Change
Android	56.0	57.2	1.2
BlackBerry	1.0	0.8	-0.2
iOS	38.5	35.2	-3.3
Windows	3.0	5.2	2.2
Other	1.5	1.7	0.2

LatAm 3 (BR, BX, AR)	3 m/e Dec 2012	3 m/e Dec 2013	% pt. Change
Android	61.6	83.5	21.9
BlackBerry	10.3	2.8	-7.5
iOS	4.4	4.3	-0.1
Windows	6.8	4.9	-1.8
Other	17.0	4.5	-12.5

EU5	3 m/e Dec 2012	3 m/e Dec 2013	% pt. Change
Android	62.9	68.6	5.7
BlackBerry	3.7	1.5	-2.2
iOS	23.7	18.5	-5.2
Windows	5.6	10.3	4.6
Other	4.0	1.1	-3.0

Atendiendo al interés de la comunidad por una plataforma, el 37% se decide por Android, el 32% por iOS y el 14% por HTML5.



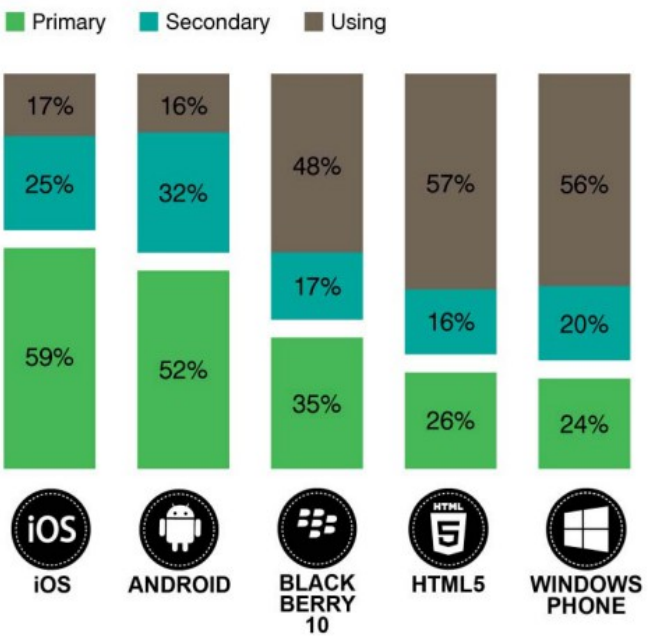
Si compulsamos estos datos con aquellos en los que se compaginan dos o más plataformas, el claro vencedor es HTML5, que pese a tener un 14% de prioridad como plataforma principal, roba casi la mitad de interés en el desarrollo multisistema.

Y es aquí donde adelantamos el primero de los puntos a considerar por la web como plataforma: su **capacidad de integración con el resto de sistemas operativos**.

Así vemos como el 60% de los desarrollos para Blackberry tienen en cuenta el desarrollo en HTML5. Un 51% para Windows 8, un 40% para Windows Phone, un 41% para iOS y un 46% para Android.

Por supuesto, es conveniente señalar que el 63% de desarrolladores iOS tienen interés en Android (37% en el lado contrario) y que una situación semejante se produce con los desarrollos de Windows 8 y Windows Phone (68% para los primeros, 56% para los segundos).

% of developers that use a platform as first, second and third choice among all developers using the platform (n=6,311)



Siguiendo con prioridad de desarrollo, iOS se alza con un interés como plataforma primaria con el 59%, Android como secundaria con un 32% y HTML5 como terciaria como un 57%.

Y es normal teniendo en cuenta que a día de hoy solo hay dos sistemas operativos móviles y uno de escritorio que han apostado por HTML5 como plataforma principal, frente a la amplia gama de sistemas operativos que cuentan con un SDK específicamente diseñado para la web.

Curioso que Windows Phone sea la tercera elección del 56% de la comunidad pese a que cuenta con la ventaja de ser cada vez más compatible con el sistema operativo de escritorio de la compañía, y con un fabricante de tanto peso en Europa como es Nokia.

Para concluir con la visión general del mercado, adjunto una gráfica resumen del interés mostrado tanto por la comunidad de desarrolladores como por el mercado de cada sistema operativo, y que saca a relucir el target objetivo.

Así, vemos como iOS está enfocado a perfiles mejor posicionados económicamente, lo que permite obtener a los desarrolladores mayor índice de ganancias por usuario.

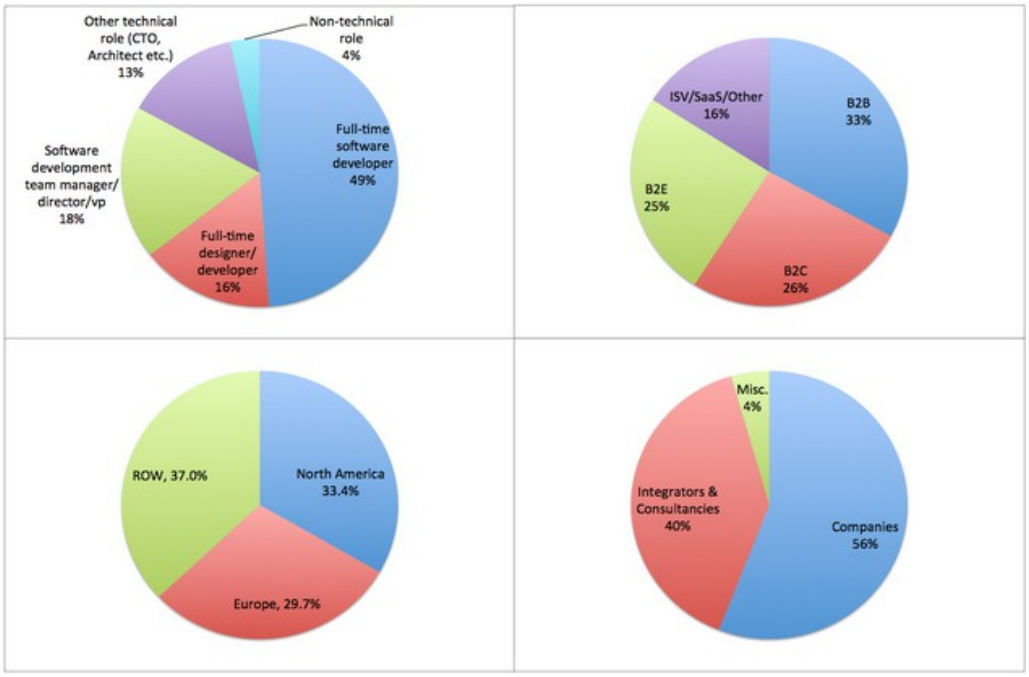
Por su parte, Android tiene un público objetivo mucho más amplio, al cubrir tanto gama baja como media y alta, y un mercado que tiende a aumentar.

HTML5 gana terreno, bien sea como plataforma de desarrollo híbrida para otros sistemas, o como propuesta de valor apoyada en el market universal.

	ANDROID	iOS	HTML5 MOBILE	WINDOWS PHONE	BLACK BERRY 10
Sales market share (smartphones, Q3 2013)	81%	13%	-	4%	2%
Mindshare	71%	55%	52%	26%	14%
Priority	37%	32%	14%	6%	5%
Loyalty	52%	59%	26%	24%	35%
Most popular in	Asia	North America	South America	Asia	South America
Median revenues	$150	$750	$150	$25	$75
Differentiating selection criterion	Open Source	Revenue potential	Ease of porting	Choice of development environment	Documentation/ Access to hardware APIs
3rd party tools index	2,8	3,1	2,5	2,5	2,3
Top revenue model	Advertising	Contract development	Contract development	Advertising	Pay per download
Segments with a strong preference to the platform	Hobbyists, Gold Seekers	Digital Media Publishers, Hunters, Guns for Hire	Product Extenders, Enterprice IT	Hobbyists, Explorers	-

3.2. La web como plataforma

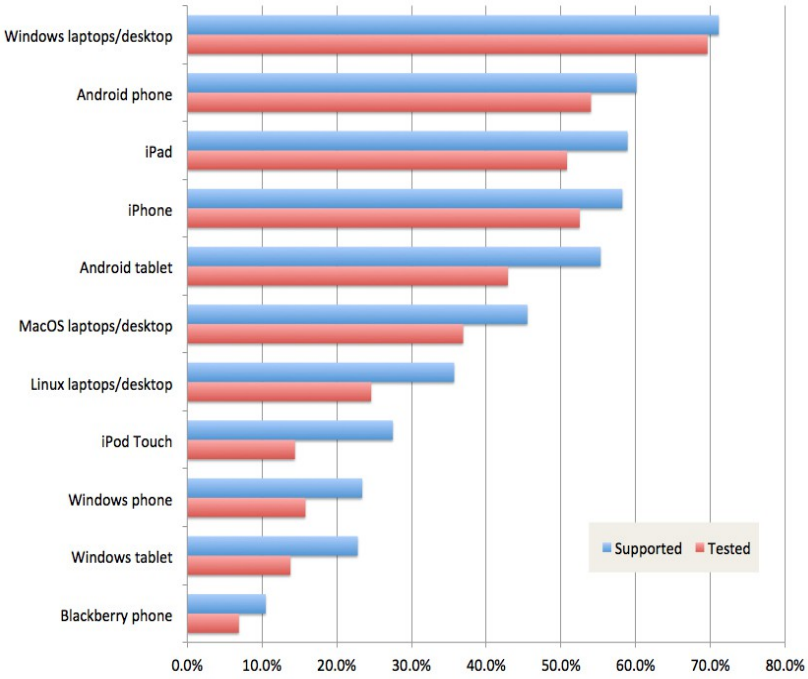
A lo largo de este estudio hemos observado como en la mayoría de situaciones, hablamos de tres grandes plataformas, siendo una de ellas HTML5.



La consultora Sencha presentaba un informe sobre la situación de HTML5 como plataforma móvil en Febrero, del que podemos sacar bastantes datos interesantes:

De los cuatro gráficos mostrados, resulta interesante apreciar el interés de Norte América por esta plataforma (33,4%) frente al de Europa (29,7%) y el resto (37%). Unos porcentajes que van acorde con el mercado de cada zona (una Europa dominada por Android/iOS, una Norte América divida entre el interés de las grandes tecnológicas, y una Sudamérica, unida a Asia y África, con un perfil de gama media, que apuesta cada vez más por HTML5).

También interesante los perfiles alrededor del desarrollo en web, con una clara predominancia por el desarrollador a tiempo completo (49%).



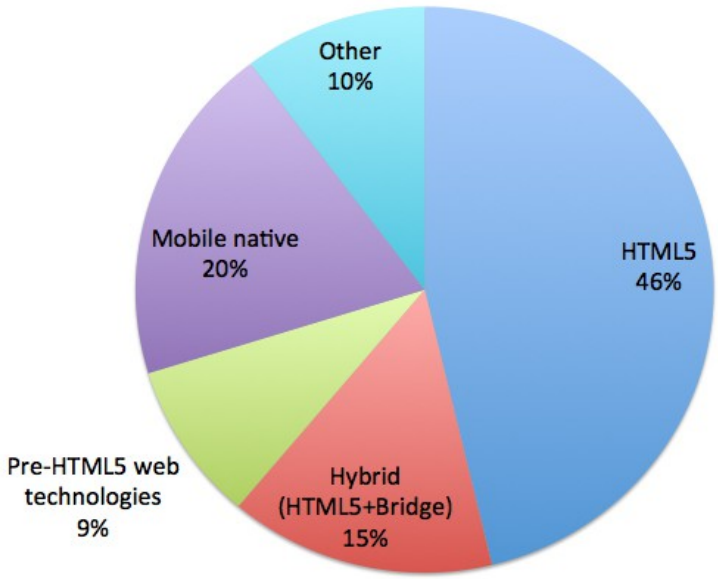
Atendiendo al carácter multisistema de HTML5, en la gráfica lateral se muestra el grado de soporte por parte de cada dispositivo, así como su nivel de testeo.

Entendiendo HTML5 como una plataforma en continua evolución, es normal que veamos porcentajes relativamente bajos, sobre todo en aquellos dispositivos cuyo entorno de desarrollo es más cerrado.

Para evitar este problema, la mayoría de sistemas operativos de nueva generación cuentan con un entorno de desarrollo

preparado para HTML5, como veremos más adelante en la propuesta de Windows 8/WP8, BB10, Tizen OS y un largo etcétera.

La presencia de HTML5 como plataforma se reafirma en países en vías de desarrollo, donde iOS pierde fuelle a favor de la gama baja/media, liderada por Android y HTML5. Y es que esta es otra de las principales ventajas de la web. Un gigante dormido que levanta el interés de la comunidad de desarrolladores, y que empieza ahora a tomar forma ya no solo como entorno híbrido de desarrollo, sino apoyado en plataformas nativas.



Que cerca del 70% de los desarrolladores utilicen o vayan a utilizar tecnologías web en sus proyectos no quiere decir que HTML5 como plataforma esté al nivel iOS o Android. De hecho, solo un 20% de estos desarrollos utilizan como plataforma móvil la web, dejando el resto para desarrollos híbridos, pre-HTML5 o para usos particulares (documentación, formulario de contacto, acceso a la página web,...).

Tiene potencial para ello, y la tendencia del mercado va por esos derroteros, pero aún faltan años para que la propuesta de una web como plataforma tenga implicaciones suficientes para destronar el duopolio de esta generación. Un duopolio que se apoya en las ventajas de la tecnología web (no queda otra) pero mantiene acertadamente un ecosistema cerrado propio (markets incomunicados y desarrollos semejantes separados).

3.2.1. Ventajas de la web como plataforma

La historia de HTML5 como plataforma es la historia de la lucha entre el interés social y el interés propio de la empresa.

HTML5 ofrece en sí la libertad de migrar desarrollos a una u otra plataforma, trasladando la información y datos de una a otra de forma transparente al usuario, algo que objetivamente no interesa cuando eres dueño de un sistema basado en desarrollos de terceros y del cual participas haciendo de intermediario. Los markets de aplicaciones se han vuelto la gallina de los huevos de oro de todo sistema operativo. Un gestor de aplicaciones en la nube que asegura un flujo constante de dinero a la empresa que está detrás, y que en casos como el de Apple, llega a quedarse con un 30% de todas las transacciones hechas en la misma.

De cara al usuario, el tener que comprar la misma aplicación dos o más veces por tener que utilizarla en diferentes sistemas operativos es un gran inconveniente. Más aún, cuando en el caso de la mayoría de juegos (por poner una topología de aplicación que requiere de un historial de uso), nuestros avances se pierden de uno a otro.

Además, el propio modelo de negocio de empresas líderes del sector como Google (agencia de publicidad web), unido a su repertorio de productos de software (Chrome OS en escritorio basado en HTML5, Android basado en Java) apuntan a la paulativa convergencia de desarrollos basados en la web, heredando particularidades de cada uno de los ecosistemas.

HTML5 es una propuesta que viene a revolucionar el paradigma de la electrónica de consumo. **El contar con una misma plataforma para cualquier tipo de dispositivo** es una característica que hace unos años no parecía importante (a fin de cuentas, o había sistemas operativos en escritorio, o para móviles), y que conforme se expande el *Internet of Things* empieza a ser una necesidad.

Es inconcebible pensar un futuro en el que cada dispositivo tenga un sistema operativo distinto. Diferentes sistemas operativos incapaces de comunicarse entre sí. Los lenguajes web, los protocolos de comunicación e Internet definen por tanto el camino a seguir.

La especialización de la plataforma, con la creación de numerosas APIs y servicios destinados a gestionar sistemas avanzados de comunicación como el dialer, el bluetooth, la SIM o el *ambient location* están ya desarrollados en HTML5, pendientes de estandarización.

Su **carácter multisistema**, unido a la facilidad de implementación bajo arquitecturas que sigan diferentes patrones (**aplicaciones híbridas**), supone en la actualidad un punto a favor de la web. Frente a afrontar un desarrollo para cada plataforma, con HTML5 una empresa puede desarrollar su proyecto y portarlo de manera sencilla al resto. En el transcurso, tiene una versión web del servicio, más otras versiones para cada sistema operativo que desee, reduciendo los costes (tanto de creación como de mantenimiento) enormemente.

Otra característica a considerar de la web como plataforma es que **es posible realizar actualizaciones de la aplicación directamente desde el servidor**. Dividiendo recursos entre el cliente y el servidor, es posible actualizar partes de la aplicación sin obligar a actualizar el cliente. Nuevas funcionalidades que todos los usuarios recibirán instantáneamente sin haber descargado una actualización, algo que lamentablemente no permiten las aplicaciones tradicionales.

Además, no hay que olvidar que **el principal objetivo de la web como plataforma es democratizar el acceso a la información**. Es decir, las aplicaciones deberían estar disponibles en cualquier lugar donde quisiéramos utilizarlas, sin tener que volver a pagar por ellas y manteniendo la misma información que teníamos en el otro dispositivo. El market de aplicaciones de Firefox OS apunta a esta convergencia: Aplicaciones disponibles tanto en la nube como en local, que pueden ser instaladas en un dispositivo con sistema operativo basado en HTML5, o gracias al navegador, en cualquier otro dispositivo, sea de escritorio, móvil, wearable o IoT.

Por supuesto, al estar basado en web, tampoco sería necesario descargar una aplicación de un market oficial, sino directamente de la página del servicio. O incluso probar la aplicación sin instalarla, desde el propio navegador.

Los requisitos necesarios para hacer correr un sistema operativo basado en la web son menores en cualquier caso a los requisitos de un lenguaje de alto nivel como java, que necesita su propia máquina virtual. En el caso de Firefox OS, el propio sistema

corre en una versión capada del navegador, dejando como veremos más adelante, la arquitectura del sistema en únicamente tres capas. Esto lo hace óptimo para entornos en los que las características técnicas están fuertemente acotadas (smartphones y tablets de gama baja, dispositivos inteligentes, wearables,...).

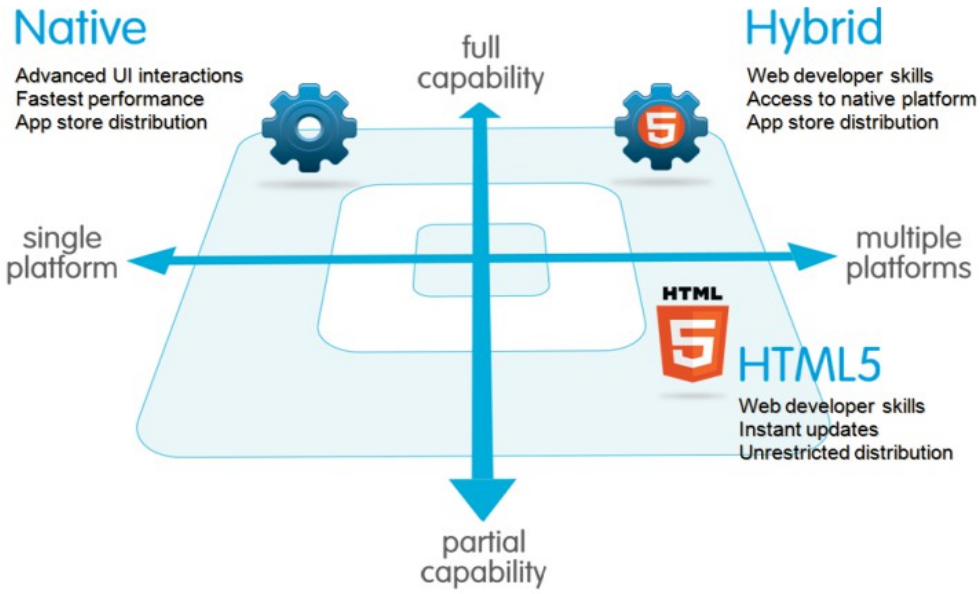
3.2.2. Inconvenientes y mitos históricos

Bajo este apartado reunimos aquellos problemas que la web como plataforma presentaba al menos en las primeras etapas.

La primera de ellas, y que sin duda ha causado un rechazo inicial a la propuesta, es que históricamente, **HTML no era un lenguaje de programación, sino de marcado**. Es decir, con HTML puro no se puede hacer nada más que la lógica de presentación de la información.

No hay que entender HTML5 como un lenguaje, sino como una plataforma formada por varios lenguajes, donde sigue estando HTML con algunas funciones nuevas, CSS3 con capacidad de gestionar diferentes condicionales (ideal para su correcto visionado dependiendo del dispositivo donde estemos) y sobre todo, una evolución de JavaScript, que en verdad es quien otorga toda la funcionalidad y dinámica a la plataforma.

El segundo conflicto lo encontramos en la idea de que **toda aplicación web debe contar con conexión para funcionar**. De nuevo otro error de no conocer la plataforma. HTML5 permite funcionar tanto online como offline. Tanto *indexDB*, como la propia caché del sistema, o el uso de base de datos local así lo atestiguan. El problema entonces podría venir del propio uso que le den los desarrolladores al servicio, algo que ocurre tanto en una aplicación web como en una para Android o iOS. Sin conexión *Facebook* solo te podrá mostrar aquello que tiene guardado en caché de la última actualización, pero en el caso de *Angry Birds*, podrías jugar sin problema.



La comunicación con el hardware es otro de los puntos oscuros de la plataforma. En verdad, el problema no vendría de la misma, sino de la compatibilidad que otorguen los componentes a HTML5. Hasta hace unos dos años, no existían APIs en HTML5 para gestionar comunicaciones de voz o notificaciones push. Situación que en la actualidad se ha solucionado. Algunas de estas librerías ya se han estandarizado, pero otras dependen todavía de empresas como Google, Microsoft o Mozilla, en espera de que la comunidad decida cuál pasa a ser la estándar. Además, plataformas como *PhoneGap* cuentan ya con intérpretes específicos para el desarrollo de apps híbridas en igualdad de privilegios respecto a las nativas de cada sistema operativo.

Rendimiento de apps híbridas frente a nativas: Aquí devolver una respuesta sin posicionarse en alguno de los dos lados es bastante complicado. Para el 99% de aplicaciones

disponibles en el market, HTML5 es tanto o más potente que cualquier otro lenguaje de programación. La duda asalta cuando estamos ante aplicaciones que requieren un nivel de renderizado realmente alto. Para estos casos, los desarrolladores pueden optar por hacer uso de apks propias según el tipo de procesador que lleve el dispositivo (en el caso de Android e Intel) o las herramientas de Apple para motores gráficos. Cuando desarrollas una aplicación híbrida (es decir, una aplicación en HTML5 destinada a correr en un sistema operativo no nativo), haces uso de herramientas que hacen de intermediarios entre la comunicación HTML5 y el lenguaje nativo, por lo que es de esperar que esa capa extra consuma más recursos que sino estuviera. Eso no quita que una aplicación bien desarrollada y optimizada funcione igual o mejor aunque sea híbrida que otra semejante y nativa, algo que quedó demostrado en la contestación que recibió hace unos años Facebook al abandonar el desarrollo de su aplicación híbrida para Android por una nativa.

En el caso de sistemas operativos basados en web, tal diferencia no existe (el propio sistema está optimizado para ese lenguaje).

4. Sistemas operativos y lenguajes web

En este apartado estudiamos la propuesta de valor que ofrece HTML5 no solo como plataforma, sino con su implementación en cada sistema operativo móvil actual. Aunque hablaremos de pasada de ellos, dejaremos de lado los sistemas operativos de escritorio, un sector que sigue presente pero se aleja del objetivo de este proyecto.

Así, obviaremos que Chrome OS es un sistema operativo basado en HTML5, y que Windows 8 o Ubuntu cuentan con un SDK para esta plataforma. Y también que cualquier aplicación desarrollada para Firefox OS puede ser abierta como aplicación (instalación del icono en el escritorio incluido) tanto en Windows, como Linux o Mac, siempre y cuando cuenten con el navegador Firefox instalado. Algo que ocurre también con las aplicaciones de Chrome.

4.1. Android

Un sistema operativo basado en Linux y desarrollado por la Open Handset Alliance, un conglomerado de fabricantes y desarrolladores liderados por Google. Android, o mejor dicho, AOSP (Android sin la parte propietaria de Google) es un proyecto open source. Y es importante señalar la diferencia, ya que el Android que viene instalado por defecto en la amplia mayoría de terminales dista mucho de seguir la filosofía open source.

La arquitectura de android sigue un patrón bastante parecido al que definimos al inicio de este documento.

Una versión reducida del **kernel** de Linux, actúa como una capa de abstracción entre el hardware y el resto de capas de software.

Las **librerías**, un conjunto de bibliotecas de C/C++ usadas por el resto de capas superiores. Se entregan al desarrollador en llamadas en el marco de trabajo de aplicaciones.

Android Runtime es un conjunto de bibliotecas escritas en Java necesarias para hacer funcionar los procesos dentro de la máquina virtual Java que usa Android, y que recibe el nombre de Dalvik (en versiones posteriores será cambiada por una nueva versión). Un dispositivo con Android puede correr diferentes máquinas virtuales a la vez de forma bastante eficiente, gestionando así diferentes procesos en diferentes máquinas, modularizando la carga de trabajo y resultando bastante sencillo terminar una para continuar con el resto.

El **Framework de aplicaciones** es el entorno de trabajo de los desarrolladores, que cuenta con todas las APIs necesarias para acceder a los diferentes elementos del dispositivo.

La capa superior es el **entorno de aplicaciones**, donde encontraremos la interfaz y toda la interacción del usuario con el dispositivo.

Android es un sistema operativo móvil diseñado principalmente para tablets y smartphones, aunque cuenta con versiones específicamente desarrolladas para entornos más concretos, como *Android Wear* para dispositivos wearables, *Glass development kit* para Google Glass y un futuro *Android TV* que parece estar aún en fase de producción.

Android soporta la mayoría de estándares HTML5 presentes, lo que permite disfrutar de una navegación web adecuada. Sin embargo, la cosa se complica cuando queremos instalar aplicaciones desarrolladas en esta plataforma.

4.1.1. Aplicaciones HTML5 portadas a lenguaje nativo

Titanium Appcelerator ofrece un framework de desarrollo basado en JavaScript que compila en lenguaje nativo (Android, iOS, Blackberry,...). Es decir, el desarrollador programa en HTML5, pero lo que al final sube al market es una aplicación nativa.

Por supuesto, tiene una desventaja, y es que el Framework llega hasta donde llega, no teniendo toda la libertad esperable para configurar tus aplicaciones de la manera que desees.

4.1.2. Aplicaciones híbridas

En este grupo, recogemos todos aquellos desarrollos basados en HTML5 que usan Frameworks para poder subir su aplicación al market adecuado. Al igual que en el anterior apartado, el más conocido es *PhoneGap*, una suite de herramientas para desarrolladores que se encarga de la comunicación con Java. Y de nuevo, encontramos la misma limitación, con el añadido de que las funciones a las que podemos acceder dependen única y exclusivamente de la compatibilidad del Framework con las implementaciones nativas del sistema. De esta manera, te obliga a tener continuamente actualizada ya no solo la aplicación, sino el intermediario.

4.1.3. WebApps

El tercer y último punto aglutina todas esas aplicaciones desarrolladas en lenguaje HTML5, y que mediante diferentes estrategias, permiten su uso dentro de un sistema operativo como Android.

Actualmente las WebApps en Android están profundamente castigadas por el sistema, lo que imposibilita el acceso a la mayoría de recursos propios de Android (descontando los básicos como acceder a la geoposición, subir una foto,...).

Sin embargo, la estrategia seguida por Google con su navegador Chrome vaticina un futuro no muy lejano en el que las WebApps tengan un tratamiento bastante más democrático. El primer movimiento lo veíamos recientemente con la posibilidad de usar extensiones de Chrome como apps de Android (a fin de cuentas, una extensión de Chrome es una WebApp). Por su parte, Firefox propone exactamente lo mismo, empaquetando una aplicación desarrollada para Firefox OS en una pestaña del navegador, y con acceso a los permisos necesarios, siempre y cuando estén contemplados en la propia app de Firefox. De esta manera, el resultado a ojos del usuario es una aplicación más “instalada” (en verdad es un acceso directo a una WebApp) junto al resto de aplicaciones.

4.2. iOS

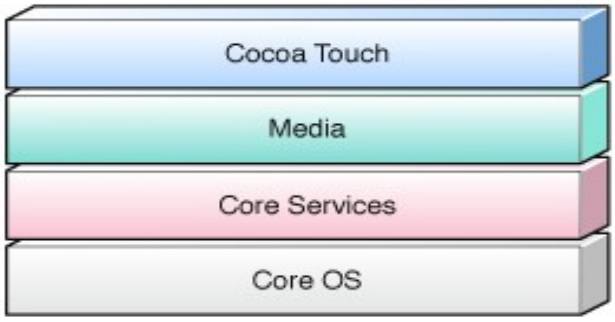
iOS (inicialmente conocido como iPhone OS) es uno de los sistemas operativos desarrollados por Apple. A día de hoy está presente únicamente en dispositivos iPhone, iPad,

iPod Touch y Apple TV, no estando disponible para su uso en dispositivos que no sean de la compañía.

La principal ventaja de esta filosofía es que el hardware de estos dispositivos está profundamente optimizado al sistema operativo que va a usar, por lo que no es raro que dispositivos con peores características técnicas de Apple funcionen igual de bien que topes de gama en Android.

Sin embargo, el contar con un software propietario y una gestión muy restrictiva de permisos imposibilita en mayor medida toda interacción con el sistema operativo que no sea nativa.

Atendiendo a su arquitectura, iOS cuenta con cuatro capas:



Core OS, el núcleo del sistema. Una evolución de OS X, el sistema operativo para escritorio de la compañía, que a su vez está basado en Darwin BSD, y por tanto, UNIX.

Core Services, contiene los servicios principales necesarios para las aplicaciones.

Media, encargada de suministrar los herramientas gráficas necesarias para la interfaz y las aplicaciones.

Cocoa Touch, que contiene el grupo de Frameworks necesarios para hacer correr aplicaciones en el sistema. Esta formado a su vez por dos elementos fundamentales: *UIKit* (interfaz de usuario) y *Foundation Framework* (clases básicas para el manejo de objetos y servicios del sistema).

Respecto a su integración con HTML5, sin duda estamos ante un entorno mucho más restrictivo, que igual que en el caso de Android, no cuenta con SDK para este tipo de tecnologías. Tanto las aplicaciones HTML5 portadas a lenguaje nativo como aplicaciones híbridas tienen cabida en los mismos límites observados en el apartado anterior (carácter multisistema a cambio de limitación en uso de APIs nativas y posibilidades que otorga el Framework que hace de intermediario). En las WebApps, y debido al carácter cerrado de la plataforma, todo servicio de navegación en iOS tiene que usar el motor de Safari, lo que hace que Firefox no esté presente en el sistema, y por tanto, no haya interoperatividad con sus aplicaciones.

Google cuenta con una versión de Chrome que sí permite generar esas aplicaciones “empaquetadas” a partir de WebApps, pero considerando que a día de hoy el acceso a la mayoría de APIs nativas del iOS están cerradas a aplicaciones web.

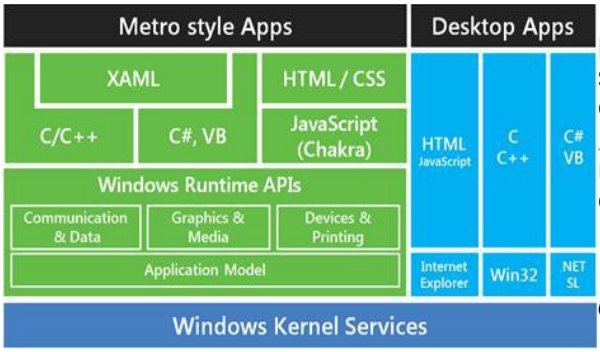
4.3. Windows Phone

Aunque en el mercado todavía hay terminales con Windows Phone 7 y 6, centraremos este apartado en Windows Phone 8 y 8.1, la última versión de la plataforma para móviles de Microsoft, con retrocompatibilidad entre aplicaciones desarrolladas para versiones antiguas (aunque la arquitectura cambie) y cada vez más soporte para convergencia de desarrollos con su hermano mayor, Windows 8/8.1.

De hecho, parte de su arquitectura (basada en Windows NT) es compartida por los dos, como es el caso del Kernel, el networking, el soporte para gráficos, sistema de archivos y multimedia.

Estamos ante un sistema operativo basado principalmente en gestos, de lo que podríamos considerar nueva generación, y cuya interacción y funcionamiento es puramente asíncrono. La decisión de adoptar como interfaz de usuario el estilo en bloques (no me atrevo a llamarlo Metro ya que cada poco le cambian el nombre...), apoyado en la elección de un buen contraste y tipografía, hace de Windows Phone un sistema diferenciador, muy rápido en carga y distinto respecto al mercado.

Eso sí, cuenta con una desventaja muy a considerar, y es que comparado con Android e iOS, el número de aplicaciones disponible es muy bajo.



En la imagen que acompaña este texto podemos ver los diferentes lenguajes que usan los sistemas de Windows. Hay que dejar claro que uno de los puntos fuertes de la suite de desarrollo de Microsoft es que esta permite trabajar al mismo nivel tanto con HTML/JavaScript, como con C/C++ o C#/VB.

Además, siempre se puede recurrir a Frameworks del tipo *Sencha*, *PhoneGap* o *Apache Cordoba*.

Las WebApps están contempladas dentro del market oficial, y son compatibles con IE10 (tanto en la versión móvil como escritorio), aunque como suele ocurrir, no disponen de todos los permisos que quizás necesiten.

Como curiosidad, existe una aplicación llamada *WebApps* que hace las veces de market de WebApps, utilizando como base la propia aplicación, y por tanto facilitando su descubrimiento e instalación. Y además, desde noviembre del 2014, Visual Studio Community 2013 es open source (pudiendo usarse para cualquier desarrollo de cualquier plataforma).

4.4. Blackberry

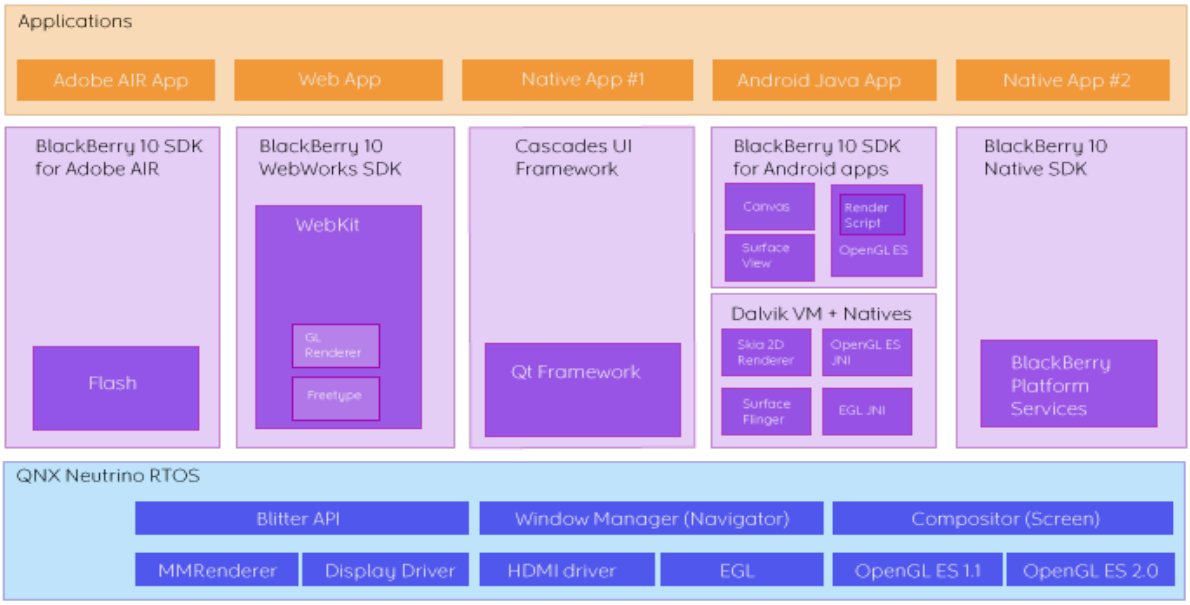
La versión analizada en este apartado es la última, BB10, por significar esta, al igual que ocurriera en Windows Phone, un cambio por completo del paradigma de versiones anteriores.

Blackberry 10 es un sistema operativo móvil propietario, desarrollado por Blackberry (anteriormente RIM), y que en un principio tenía como claro target el mundo empresarial.

Las blackberrys fueron un paso intermedio entre la figura del móvil y el smartphone. Un dispositivo con teclado físico QUERTY y un nutrido grupo de aplicaciones que añadían atractivo al sistema.

Sin embargo, una mala decisión por parte de la compañía apartó a RIM del camino, quedando relegado a lo que es hoy en día.

Blackberry 10 viene a luchar con esta situación, eliminando todos aquellos errores que cometieron en su momento con un sistema operativo cuya interacción está basada en gestos, y con la apertura de su SDK a múltiples lenguajes.



El kernel es una versión optimizada del *QNX* presente en las anteriores versiones del SO, sobre el que corre el resto del sistema.

En blackberry 10 se pueden instalar aplicaciones nativas, aplicaciones desarrolladas con tecnología Adobe AIR, WebApps e incluso aplicaciones Android, por supuesto, después de haberlas adaptado a su entorno.

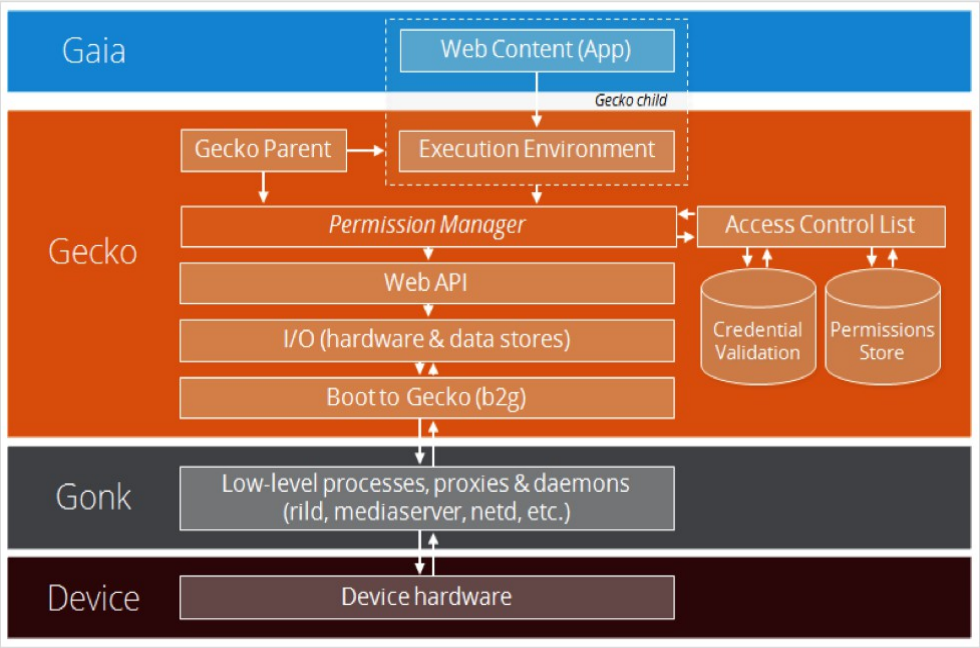
En el caso que nos compete, la documentación oficial del desarrollo de aplicaciones en HTML5 para Blackberry es de las mejores del sector.

Las aplicaciones desarrolladas son portadas mediante *Apache Córdoba*, dentro del propio SDK oficial, por lo que el nivel de acceso a permisos nativos está asegurado.

4.5. Firefox OS

La propuesta de Mozilla es de las más interesantes del mercado, ya no solo por ser la única dirigida por una fundación sin ánimo de lucro, sino que además cuenta con una trayectoria histórica de apoyo a la neutralidad de la red digna de admiración.

Firefox OS es un sistema operativo open source basado en HTML5, que se apoya en la tecnología desarrollada por Mozilla, y que por tanto, apunta a la multiplataforma.



La arquitectura de Firefox OS está dividida en tres capas:

- **Gonk:** estamos ante la capa de más bajo nivel, una suerte de kernel Linux destilado, y que adquiere cuerpo gracias a un gran número de librerías de código abierto.
- **Gecko:** Hablar de Gecko a estas alturas es hablar del motor de renderizado más conocido, y que podemos encontrar en todos los productos de Mozilla. Esto quiere decir que una aplicación de Firefox OS puede correr en un terminal con Firefox OS, pero también en cualquier lugar donde podamos instalar el navegador Firefox.
- **Gaia:** La capa visible, encargada de servir la interfaz, y basada en HTML5. La mayor ventaja que ofrece Firefox OS en este sentido es que toda la comunicación con los diferentes elementos de un dispositivo (GPS, conectividad, SMS,...) se hace mediante APIs ya estandarizadas, y otras pendientes de estandarización. Por tanto, por primera vez en la historia contamos con herramientas web para la comunicación con elementos de hardware, que están siendo ya portadas al resto de navegadores.

Sobra decir que la compatibilidad con HTML5 está más que asegurada. HTML5 es el lenguaje nativo de Firefox OS, el lenguaje nativo de la web, y por tanto, estará presente allí donde haya un navegador que lo interprete.

Como ya comentamos en otros apartados, las aplicaciones desarrolladas para Firefox OS pueden ser utilizadas como apps sin hacer ningún cambio en su código tanto en sistemas operativos de escritorio como en Android. Además, la idea detrás de la fundación Mozilla es que este sistema de democratización de aplicaciones no tenga porqué pasar únicamente por ellos, sino que se permita su interoperatividad con otros markets e incluso (teniendo en cuenta las medidas de seguridad oportunas) sea accesible desde la propia web del servicio.

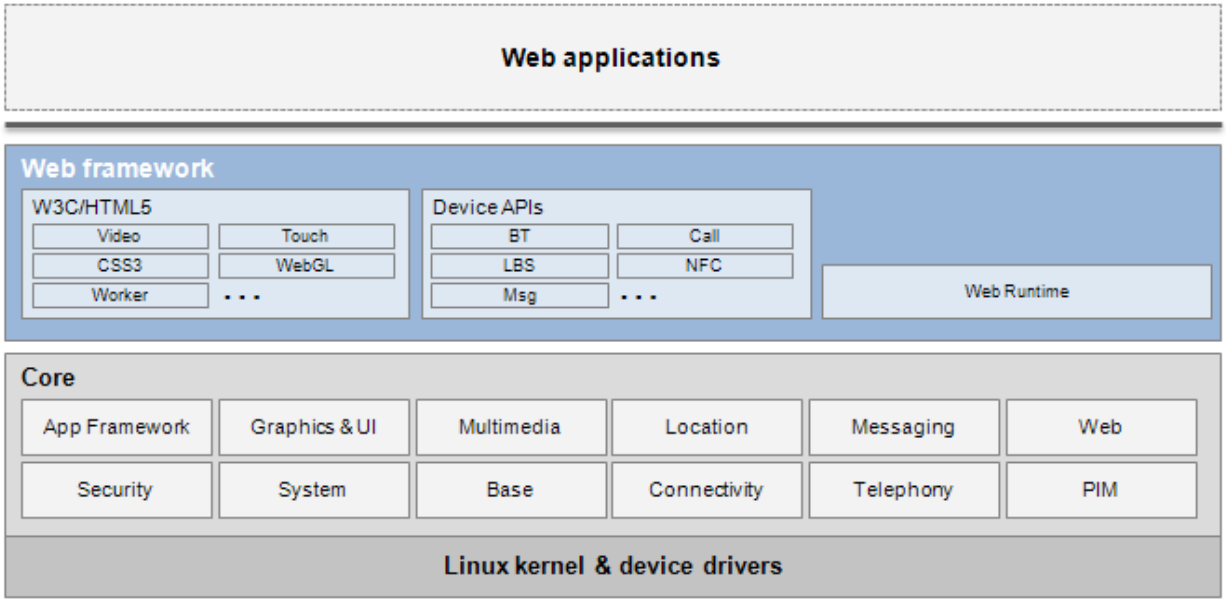
El objetivo final es que HTML5 sea la plataforma por defecto del mercado, rompiendo los jardines vallados actuales como en su día internet hizo con las redes cerradas.

Los dispositivos con este SO están actualmente disponibles en 13 países: Brasil, Colombia, Venezuela, Perú, Uruguay, México, Alemania, Polonia, Hungría, Grecia, España, Serbia y Montenegro, y se espera su expansión por el resto de América Latina y Asia en lo que queda de año.

La propuesta de Mozilla es atacar el mercado de entrada en países en vías de desarrollo, con unos dispositivos de muy bajo coste (el último presentado, 25 dólares). Aunque inicialmente el sistema ha sido diseñado para smartphones, ya se está testando en tablets e incluso en Smart Tvs y placas de DIY.

4.6. Tizen OS

Tizen es un sistema operativo móvil open source basado en Linux que nace bajo el amparo de la Asociación Tizen (anteriormente llamada Fundación LiMo), liderado por Samsung e Intel, y patrocinado por la Linux Foundation.



Aunque tenga un parentesco lejano con Meego, ese sistema operativo desarrollado por Nokia e Intel principalmente para netbooks, lo cierto es que Tizen ha seguido un camino bastante diferente, y como veremos más adelante, Meego ha acabado como proyecto externo (y amparado por Nokia) bajo el nombre de Sailfish OS.

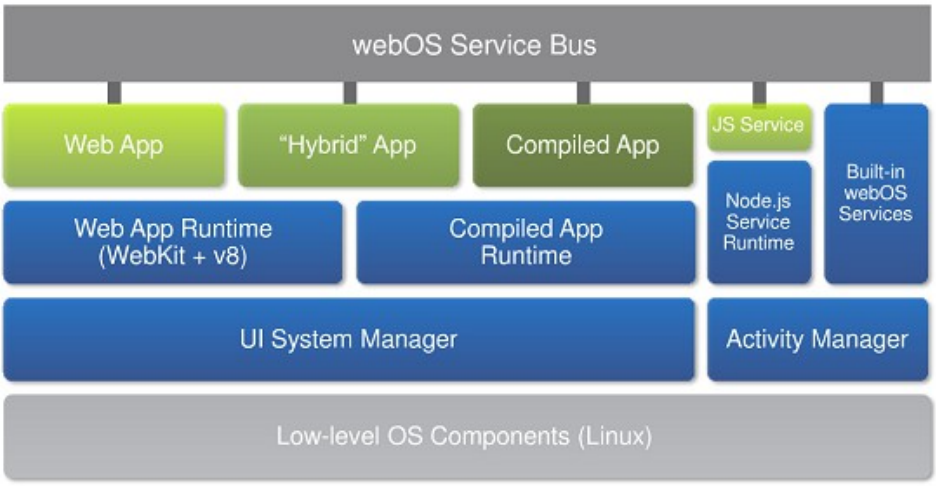
El sistema operativo está enfocado a cubrir diferentes topologías de dispositivos, estando actualmente disponible en la nueva versión del smartwatch de Samsung, así como en algunos smartphone y tablets que todavía no han salido al mercado (el primero, con nombre Samsung Z, saldrá en Rusia a lo largo de este año).

Entre sus puntos fuertes está el tener tras de sí a un gigante de la industria como Samsung, el basar su desarrollo de apps en HTML5 (el motor de renderizado es Webkit, frente al Gecko de Firefox OS) y el ser compatible (después de algunos pequeños cambios con el OpenMobile ACL) con aplicaciones Android.

Como ya ocurriera con Firefox OS, las aplicaciones HTML5 son nativas en el sistema, por lo que basta conocer el funcionamiento del SDK para desarrollarlas. No es necesario por tanto recurrir a intermediarios, aunque sea posible utilizarlos para hacer el port (si por ejemplo ya tienes un proyecto desarrollado en PhoneGap).

4.7. Web OS

La historia de Web OS es la historia del quiero y no puedo. Inicialmente desarrollado por Palm como un sistema operativo multitarea basado en Linux, fue vendido a HP, que al final liberó su código y volvió a venderlo, esta vez a LG, quien lo ha estado usando recientemente para televisores inteligentes.



Al igual Tizen OS, cuenta con un kernel de Linux y una interfaz desarrollada en HTML5 que lo hace fácilmente explotable para diferentes usos, como queda patente en la larga lista de productos en los que ha sido utilizado.

Lo cierto es que su compra por LG y su interés por utilizarlo como SO para televisores cierra bastante el mercado a terceros. No existe mucha documentación actualizada sobre el proyecto, algo por otra parte normal ya que esta nueva visión del producto se ha presentado en Enero de este mismo año.

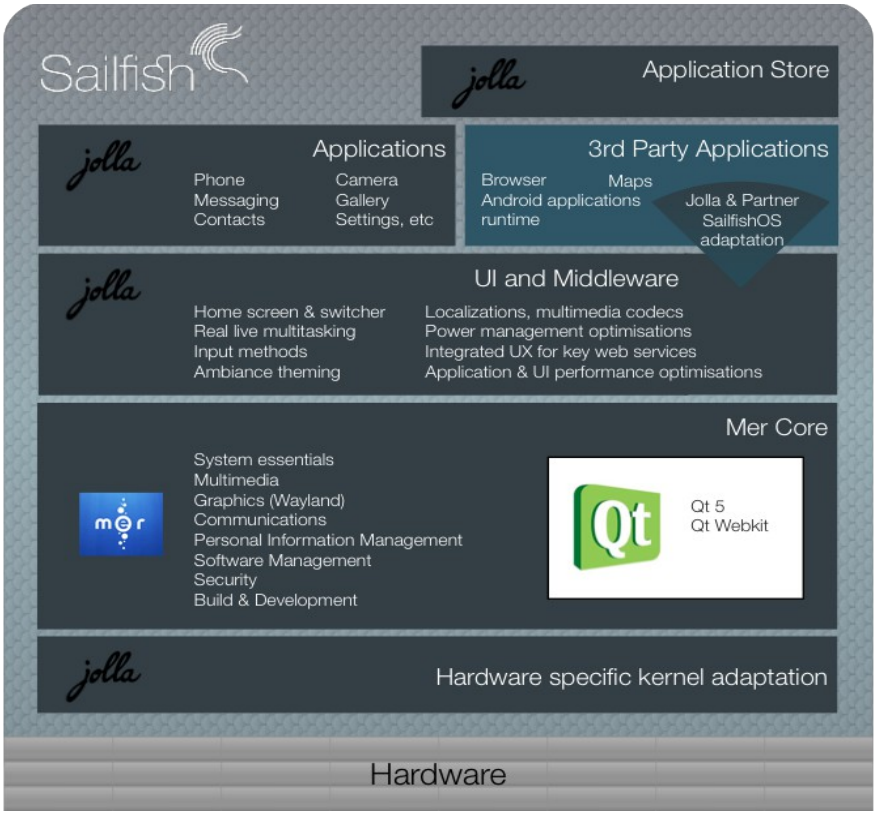
4.8. Sailfish OS

Sailfish OS es la evolución lógica de Meego, un sistema operativo desarrollado por Nokia, basado en Linux y con el objetivo de atacar el mercado de smartphones y tablets.

Cuando Meego se dejó de lado, algunos de sus trabajadores abandonaron Nokia para formar Jolla, una empresa amparada por la finlandesa, que ha renombrado el sistema como Sailfish OS.

El sistema operativo cuenta con su propio lenguaje de programación (Qt/QML), aunque acepta por mediación de sus respectivos sdks el desarrollo de apps en HTML5 (Cordoba Qt) y Java (Android apps).

La arquitectura es semejante a la que ya hemos visto en la mayoría de sistemas operativos móviles: un kernel Linux, una capa de componentes (basada en Mer), el middleware con la interfaz y las aplicaciones.



4.9. Ubuntu (Phone)

La promesa de Canonical (empresa detrás de esta distribución de Linux) era llegar a la verdadera convergencia móvil/escritorio con el nuevo Ubuntu.

De aquellas palabras hasta ahora ya ha pasado año y medio, y el proyecto sigue aún en fase alpha.

El sistema operativo, disponible actualmente para terminales Nexus, ofrece un sistema basado en gestos que hereda la apariencia de su homólogo de escritorio. La parte interesante del mismo era esa supuesta convergencia en la que un usuario podría utilizar su smartphone, llegar a casa, conectarlo a la pantalla y a un teclado, y utilizar Ubuntu como tal, algo que a día de hoy todavía no es posible.

Por las pruebas iniciales, también le falta pulir la multitarea, al resultar imposible por parte del usuario eliminar apps de la memoria, lo acaba por hacer mella en la batería del dispositivo.

Respecto a su integración con HTML5, está asegurada. Ubuntu cuenta con su propio framework de desarrollo, pero permite mediante sdk realizar lo propio en lenguaje web.

Queda por tanto tener paciencia y esperar que el proyecto no caiga en el olvido.

5. Conclusiones

Como hemos podido observar, todas las propuestas móviles del mercado (incluso las que aún están por llegar) tienen de una u otra manera compatibilidad con desarrollos HTML5.

La idea que subyace de proyectos como Firefox OS es la de liberar el sector, generando una comunidad basada en estándares, capaces de adaptar el sistema operativo a sus posibles diferentes usos. Que el usuario final pueda acceder a la información allí donde esté, indistintamente del dispositivo que tenga a mano.

Enriquecer con ello el conocimiento humano, el acceso al mismo y la comunicación.

El mercado actual tiene dos grandes vencedores: Android e iOS, basados en el mismo paradigma (control del mercado de aplicaciones). El siguiente paso sería descentralizar los markets, de manera que el usuario tenga más opciones donde elegir y más libertad para portar sus herramientas de uno a otro sistema.

Y la forma más inmediata que tenemos es mediante HTML5, una plataforma que tiene todo aquello que la sociedad necesita para gestionar la información. Seguirán existiendo desarrollos propietarios, y sistemas operativos cerrados (de hecho mientras más, más competencia y por tanto más evolución del mercado), pero en última instancia, permitir, como a día de hoy la mayoría de sistemas están permitiendo, el desarrollo de servicios web asegura que todo ese valor no se perderá cuando una u otra plataforma caiga.

El que en vez de dos grandes, haya cuatro o cinco, fuerza todavía más a que HTML5 sea el motor de comunicación entre ellos. Un conjunto de tecnologías multiplataforma, con una curva de aprendizaje medio/bajo y con las herramientas necesarias para interpretarlas al alcance de cualquier dispositivo de la actualidad.

El objetivo de este estudio no es dogmatizar en los beneficios de HTML5, sino mostrar una realidad del mercado. La misma que están llevando compañías como Canonical o Microsoft, a priori, poco interesadas en los beneficios de la web, a favorecer su auge en sus sistemas. A que fundaciones como Mozilla o Linux Foundation estén detrás de plataformas basadas en estándares HTML5. Al interés que muestra América Latina por apostar por la web para reducir costes y dar el salto a los dispositivos inteligentes en países en vías de desarrollo.

El mercado pide convergencia, y la única plataforma que está en potestad de ofrecerla a nivel universal es HTML5.

Que la comunicación del día de mañana se pueda hacer de la misma manera desde cualquier dispositivo, manteniendo para ello un nivel necesario de anonimato y privacidad. Las empresas deberán adaptar su modelo de negocio a este nuevo paradigma, quizás ofreciendo valor con una capa de servicios, con acuerdos ventajosos al cliente.

No se trata, por tanto, que todos usemos Firefox OS o Tizen OS, sino que todos tengamos acceso a las ventajas de HTML5 como plataforma. Porque la sociedad así lo está exigiendo, y el mercado entiende que es la salida más acertada de cara a esta nueva generación de tecnología.

6.- Referencias

APPIO. «Tipos de Apps: Nativas, híbridas y WebApps». Consultado el 8 de Junio de 2014.

Blackberry Developer. «Native SDK for Blackberry 10». Consultado el 15 de Junio de 2014.

Comscore. «SO Mobile market». Management & Productivity. Consultado el 8 de Junio de 2014.

Developer Economics. «Developer Economics Q1 2014». Consultado el 8 de Mayo de 2014.

F. Iglesias, Pablo (Julio de 2013). «Plataformas móviles y lenguajes web: Realidad del mercado». PabloYglesias. Consultado el 8 de Junio de 2014.

F. Iglesias, Pablo (Marzo de 2013). «Vivimos una democratización de sistemas operativos». PabloYglesias. Consultado el 8 de Junio de 2014.

F. Iglesias, Pablo (Abril de 2013). «De camino a un mundo donde el usuario no será dueño de sus dispositivos». PabloYglesias. Consultado el 15 de Junio de 2014.

F. Iglesias, Pablo (Abril de 2013). «Google en busca de un único ecosistema». PabloYglesias. Consultado el 15 de Junio de 2014.

F. Iglesias, Pablo (Diciembre del 2012). «Fastbook, demostrando que HTML5 sí le planta cara al lenguaje nativo». PabloYglesias. Consultado el 15 de Junio de 2014.

F. Iglesias, Pablo (Enero del 2013). «Desarrollando para Firefox OS: Primeros Pasos». PabloYglesias. Consultado el 15 de Junio de 2014.

Fried, Ina (Febrero del 2014). «Mozilla says new chip $25 smartphone». Recode. Consultado el 15 de Junio de 2014.

Gaia Program. «Enterprise Structure & Organization». International Business & Change Management. Consultado el 12 de Junio de 2014.

Game Developers Conference. «GDC Q1 Mobile Development». Consultado el 18 de Mayo de 2014.

Grieve, Andrew (Enero de 2014). «Run Chrome Apps on mobile using Apache Cordoba». Chromium. Consultado el 8 de Junio de 2014.

Kantar. «Mobile SO 2014». Consultado el 15 de Junio de 2014.

Osmar, Gabriel (2012). «Monografías: Sistemas operativos en dispositivos móviles». Universidad Nacional del Nordeste. Consultado el 8 de Mayo de 2014.

Pro-Chile, New York (2013). «Estudio de mercado plataformas móviles». Consultado el 19 de Mayo de 2014.

Pastor, Javier (Marzo de 2013). «Desarrollo en plataformas móviles: Así está el mercado». Xataka Móvil. Consultado el 8 de Junio de 2014.

PhoneGap. «Make easily apps with HTML, CSS and JavaScript». Consultado el 8 de Junio de 2014.

Salesfore. «Native or Hybrid app». Consultado el 15 de Junio de 2014.

Sencha. «The State of HTML5 Developer in Enterprise». Consultado el 14 de Mayo de 2014.

Shapiro, Matthias (Febrero de 2013). «Gettin started with Windows Phone 8 HTML5 Apps». MSDN. Consultado el 15 de Junio de 2014.

Titanium Appcelerator. «Appcelerator Platform». Consultado el 14 de Mayo de 2014.