Introducción y conceptos generales

Marcos González Sancho

PID_00192299

UOC
Universitat Oberta de Catalunya

Índice

1. Aplicaciones Rich Media

1.1. Concepto y definición

El término *rich media* hace referencia al uso conjunto de diferentes medios (texto, imagen, sonido, vídeo, animación...), y es frecuente encontrarlo en el ámbito del marketing y la creatividad, pero no tanto en el campo de las aplicaciones.

Actualmente, se están estableciendo muchos vínculos entre los medios de comunicación tradicionales e Internet, y algo similar empieza a suceder cuando hablamos de aplicaciones. Internet ha evolucionado, con el paso de los años, de una visión muy limitada de los medios a ser el referente en soporte y variedad de estos, siendo además el entorno en el que se hace un mayor uso de estos medios.

Las aplicaciones también se están adaptando en muchas áreas al concepto que existe detrás de Internet y, por consiguiente, al de multimedia, con un enfoque claro hacia el usuario. Esto ha dado lugar a una tipología de aplicaciones que sería la que englobaríamos bajo el concepto de Aplicaciones Rich Media (ARM).

> Aplicaciones Rich Media serían todas aquellas que permiten y potencian el uso de diferentes elementos y contenidos multimedia como esencia de la aplicación y también como método de interacción con el usuario, con independencia de la plataforma o entorno al que van destinadas (navegador, escritorio, dispositivos, etc.).

Cuando hablamos de Rich Media estamos englobando aspectos que afectan a la planificación, el diseño, el desarrollo y el concepto de aplicaciones, con una clara orientación hacia los métodos de interacción y comunicación con el usuario. Si bien el concepto *multimedia* no necesariamente implica interactividad, hoy en día el usuario ha dejado de jugar un rol pasivo en las aplicaciones, por lo que el diseño orientado al usuario es una máxima en este tipo de aplicaciones.

1.2. Características de una ARM

Para comprender de una forma más completa el concepto y las funciones de una ARM, es interesante analizar qué características son deseables (en parte o en su totalidad) cuando se decide crear una ARM, y cuáles se aplican de forma incorrecta.

Cuando trabajamos en la creación de una ARM, es deseable y habitualmente necesario atender a las siguientes características:

- **Diseño y desarrollo orientados al usuario** (recordemos que el contenido multimedia enriquecido está claramente enfocado al mismo).

- **Uso de diferentes medios**, ya sea como contenidos por sí mismos o como mecanismos para mostrar de forma mejorada la información al usuario.

- **Interactividad** en diseño, interfaz e interacción.

Por el contrario, existen consideraciones que no necesariamente son ciertas cuando se aplican a una ARM (en muchas ocasiones tienen sentido, pero no de forma obligatoria), tales como:

- Una ARM debe manejar el máximo tipo de medios.
- Una ARM debe ser multiplataforma.
- Una ARM debe ser multidispositivo.
- Una ARM debe estar basada en contenidos en la nube.
- Una ARM no puede ser una aplicación basada en datos.

Hoy en día, el concepto *rich media* ha sido expandido con **nuevos tipos de contenidos** "no tradicionales" que van mucho más allá de los clásicos (texto, imagen, animación, audio y vídeo), pudiendo considerarse como nuevos medios los siguientes:

- Mapas.
- Publicidad.
- Códigos QR o similares.
- Realidad aumentada.
- 3D.
- Contenido social o colaborativo.
- Contenido interactivo (educativo, lúdico...).

Además, la **interactividad**, entendida como un aspecto más que habitual y fundamental en ARM (dado que estas aplicaciones están claramente orientadas al usuario), tiene nuevos mecanismos que ofrecer:

- Entrada de vídeo/webcam.
- Entrada de audio/micrófono.

- GPS y geolocalización.
- Acelerómetro.
- Brújula.
- Sistemas de valoración.
- Compartición en redes sociales.
- Etc.

1.3. ARM y Rich Internet Applications

Dado que la definición aportada es bastante abierta, puede dar lugar a confusiones con otros términos que, desde hace un tiempo, están muy presentes en la comunidad de desarrollo, como es el caso de las **Rich Internet Applications** (**RIA**).

Podemos ver las RIA como un subconjunto de las ARM, ya que cumplen las condiciones de las ARM pero limitando su medio a un navegador.

No debemos perder de vista que, cuando se habla de ARM, no se habla de un medio o plataforma en concreto, sino que una ARM podría estar desarrollada para trabajar en un navegador (Internet), en una TV, en un dispositivo móvil, o incluso en una combinación de varias plataformas.

2. Tecnologías relacionadas

Dentro del conjunto de tecnologías que nos permiten crear una ARM, existen algunas que tienen especial relevancia por el volumen de desarrolladores que hay detrás de las mismas o por las capacidades enfocadas especialmente al trabajo con contenidos e interacciones basadas en contenidos multimedia.

2.1. JavaFX

JavaFX es un conjunto de productos y tecnologías de Sun Microsystems, que permite la creación de ARM y RIA bajo desarrollo en Java. Sus creadores lo definen como el paso siguiente en la evolución de Java hacia el mundo de los contenidos enriquecidos multimedia. Fue lanzada oficialmente en diciembre del 2008.

JavaFX está diseñado para proporcionar una plataforma ligera con capacidades gráficas de aceleración por hardware diseñada para el desarrollo de aplicaciones de negocio. Con JavaFX, los desarrolladores podrán reutilizar librerías empleadas en desarrollos previos (realizados obviamente en Java) e incluso tener acceso a capacidades nativas del sistema.

JavaFX tiene como componentes destacados JavaFX Script (lenguaje de script orientado a objetos y con una sintaxis simplificada con respecto a Java) y JavaFX Mobile (sistema de software completo para dispositivos móviles), que fueron lanzados a inicios del 2009.

Enlaces relacionados

http://javafx.com/.

http://en.wikipedia.org/wiki/JavaFX.

http://es.wikipedia.org/wiki/JavaFX.

http://www.oracle.com/technetwork/java/javafx/overview/index.html.

http://www.slideshare.net/ibantxuyn/introduccin-a-java-fx.

2.2. Silverlight

Silverlight es una herramienta de desarrollo diseñada para permitir crear atractivas experiencias interactivas que puedan funcionar en diferentes plataformas como navegador y dispositivos. Silverlight dispone de un *plugin* gratuito creado con el *framework* .NET y compatible con múltiples navegadores, dispositivos y sistemas operativos.

Silverlight fue visto desde su lanzamiento (septiembre del 2007) como la herramienta creada por Microsoft para competir con el omnipresente Adobe Flash. Para ello, la herramienta permite trabajar con funciones multimedia tales como la reproducción de vídeo, soporte para gráficos vectoriales, animaciones e interactividad.

A pesar de sus capacidades y de sus reconocidas herramientas de desarrollo, no ha alcanzado una popularidad como su competidor referente.

Enlaces relacionados

http://www.microsoft.com/silverlight/.

http://www.silverlight.net/.

http://en.wikipedia.org/wiki/Microsoft_Silverlight.

http://es.wikipedia.org/wiki/Microsoft_Silverlight.

http://msdn.microsoft.com/es-es/silverlight/bb187358.

http://msdn.microsoft.com/es-es/library/cc838158(v=vs.95).aspx.

2.3. HTML5, CSS3 y Ajax

Ajax (Asynchronous JavaScript And XML) es una técnica de desarrollo web para la creación de aplicaciones interactivas enriquecidas, que se emplea en la parte cliente (navegador) y que permite establecer una comunicación con el servidor sin realizar un cambio de página, lo que repercute de forma directa en la interacción con el usuario, ya que es posible realizar cambios en las interfaces sin realizar una recarga.

Ajax es una de las tecnologías o mecanismos que han sido base de la denominada Web 2.0, y de las RIA, y está íntimamente relacionada con otros lenguajes, formatos y estándares como XHTML, HTML, DOM, CSS, XML y JSON.

Normalmente, las peticiones del lado del cliente y la actualización de la interfaz se realizan en JavaScript, y la comunicación con el servidor puede llevarse a cabo haciendo uso del objeto XMLHttpRequest (disponible en la mayoría de navegadores actuales). La respuesta, si bien originalmente fue pensada en XML, no requiere de forma obligatoria este formato, siendo actualmente muy habitual el uso de otros formatos generalizados como JSON, HTML o texto plano.

El objeto XMLHttpRequest aún no ha sido estandarizado por la W3C (http://www.w3.org/TR/XMLHttpRequest/), de modo que a día de hoy existen algunas diferencias entre las distintas implementaciones que se han ido elaborando por parte de los fabricantes de navegadores.

Si bien Ajax como tal no es un exponente claro de desarrollo de ARM, la llegada de **HTML5** y **CSS3**, con algunas características especialmente interesantes para el desarrollo de este tipo de aplicaciones (principalmente para dispositivos en forma de *webapps*), lo ha potenciado aún más, ya que, combinando Ajax con ambos, se pueden crear aplicaciones muy interesantes que cuentan como principal y gran ventaja su compatibilidad con gran cantidad de dispositivos.

La creación de una *webapp* basada en HTML5 (y CSS3 para su estética) cuenta con las ventajas que implica el desarrollo web tales como: un *layout* flexible que permite su visualización en diferentes pantallas, compatibilidad con casi la totalidad de sistemas operativos para *smartphones* (que tienen la cualidad de ir por delante de los navegadores de escritorio en compatibilidad con HTML5) y, gracias a las nuevas características del lenguaje, capacidad de trabajar con:

- Audio (mediante la etiqueta <audio>).
- Vídeo (mediante la etiqueta <video>).
- Canvas 2D y 3D para el trabajo con gráficos avanzado.
- Geolocalización.
- Modo de trabajo fuera de línea (mediante un cacheado especialmente interesante).
- Almacenamiento persistente (incluso con base de datos).
- Mejoras en el formateo de campos de entrada (incluso para el teclado virtual).
- Mejoras en validación de formularios (por tipos: email, teléfono, etc.).
- Posibilidad de trabajo con *sockets* (WebSockets).
- Posibilidad de trabajo con ejecución multitarea (Web Workers).
- Mejoras en interacción como la API de Drag & Drop.

A pesar de todas estas características, cabe destacar que hoy no todo es tan bonito como pudiera parecer, pues siguen existiendo diferencias entre navegadores (como en el caso del vídeo), características que no tienen muy claras su futuro (como el *web database* para almacenamiento persistente en base de datos, que ha sido descontinuado y en su lugar enfocado hacia la Indexed DataBase API -http://www.w3.org/TR/IndexedDB/-, etc.).

En general, siempre que se quiera trabajar sobre HTML5, hay que afinar muy bien los dispositivos o navegadores destino, para tener claro los *fallbacks* que hay que implementar para dar soporte a todos los usuarios. Para ello, es habitual contar con herramientas que aumentan la compatibilidad de los navegadores con HTML5, como HTML5Shiv (http://code.google.com/p/html5shiv/), que permite ampliar el soporte de HTML5 a navegadores Internet Explorer antiguos, o Modernizr (http://modernizr.com/), que simplifica en gran medida la detección de todas las capacidades del navegador para HTML5 y CSS3, de manera que el desarrollador pueda ofrecer una versión compatible en caso de no soportar alguna de ellas (Modernizr incorpora, por cierto, HTML5Shiv).

Enlaces relacionados con Ajax

http://es.wikipedia.org/wiki/AJAX.

http://en.wikipedia.org/wiki/Ajax_(programming).

http://www.librosweb.es/ajax/.

http://www.w3schools.com/ajax/default.asp.

http://jquery.com/.

Como librería destacada que dispone de gestión de Ajax: http://api.jquery.com/category/ajax/.

Enlaces relacionados con HTML5

HTML5:

http://dev.w3.org/html5/spec/spec.html.

http://diveintohtml5.info/.

http://www.html5rocks.com/en/.

http://5stepstohtml5.com/.

Compatibilidad HTML5 y CSS3:

http://html5please.com/.

http://caniuse.com/.

http://www.findmebyip.com/litmus/.

http://modernizr.com/.

HTML5 Demos:

http://html5demos.com/.

HTML5 Canvas:

http://blog.nihilogic.dk/2009/02/html5-canvas-cheat-sheet.html.

http://www.html5canvastutorials.com/.

Animación con HTML5 y Canvas:

http://createjs.com/#!/CreateJS.

JavaScript Game Engines List:

https://gist.github.com/768272.

Box2D JavaScript Game Engine: http://code.google.com/p/box2dweb/.

2.3.1. Extendiendo el navegador

Finalmente, existe un tema especialmente delicado que, en el caso de aplicaciones, se convierte en vital, como es que la necesidad de las *webapps* de trabajar sobre un navegador hace que requieran de un canal extra para extender sus capacidades a las que se espera de una aplicación. Existen varias soluciones

en el mercado que actúan como puente entre el navegador y el dispositivo, de manera que desde un lenguaje como JavaScript pueden realizarse llamadas a librerías nativas que extienden las funcionalidades.

Enlaces relacionados con extensión de HTML5

http://phonegap.com/.

http://www.appcelerator.com/.

http://www.sencha.com/products/touch/.

http://www.markus-falk.com/mobile-frameworks-comparison-chart/.

Una de las más conocidas es PhoneGap, que actualmente está bajo el control de Adobe. Mediante esta herramienta, podemos crear una *webapp* que actualmente podría correr como aplicación nativa (con algunas limitaciones: http://phonegap.com/about/features) en multitud de sistemas operativos, pero unificando las llamadas a acciones nativas como, por ejemplo, consultar el GPS, obtener una imagen de la cámara del dispositivo, sistema de ficheros, etc.

Al igual que ocurre con cualquier tecnología de este tipo, una vez creada la base con la tecnología común se ha de realizar un proceso de compilación para las diferentes plataformas de manera independiente, para poder obtener como resultado instaladores compatibles con los correspondientes sistemas operativos.

Como decimos, PhoneGap es solo una de tantas, como pueden ser Titanium Appcelerator, Sencha Touch, Corona, etc.

2.4. Flash

Se conoce por **Flash** (en realidad, Flash Player) el *plugin* gratuito disponible en casi la totalidad de navegadores, que permite renderizar contenido creado en formato SWF (actualmente abierto por el Open Screen Project). Adobe Flash Professional es la herramienta o IDE de desarrollo creada originalmente por Macromedia (actualmente Adobe) para generar este tipo de contenido, que desde sus inicios ha ido solventando de forma anticipada las limitaciones o problemáticas de la web basada en estándares, ante las cuales siempre ha estado la creación avanzada de contenidos multimedia e interactividad.

Actualmente, Flash está presente en una gran cantidad de áreas de negocio en Internet como publicidad, web TV, *streaming*, desarrollo de juegos, interactivos e infografías, animación, etc., y se usa como herramienta complementaria en determinados campos en infinidad de medios.

Por el índice de penetración que ha alcanzado en Internet (alrededor del 99% según las estadísticas de la compañía), se ha convertido en un estándar de facto para contenidos multimedia en Internet y es un claro dominador hoy en día en vídeo, audio, juegos (con soporte 3D y aceleración gráfica) y publicidad.

Si bien las capacidades que van adquiriendo poco a poco los estándares como HTML5 empiezan a permitir sustituir Flash en determinados contenidos, este *plugin* aún está por delante de ellos en la mayoría de esos campos cuando se requieren características avanzadas.

Finalmente, Flash Professional (la herramienta o IDE) permite generar aplicaciones *standalone* de escritorio compatibles con sistemas Windows y Mac, reutilizando prácticamente el 100% de código que para la misma versión de Internet, aunque esta característica se ha obviado en los últimos años con la aparición de Adobe AIR.

2.5. Flex Framework

Con el nombre actualmente de **Apache Flex** (bajo el control de la Fundación Apache desde finales del año 2011), el anteriormente conocido Adobe Flex es un *framework* para el desarrollo y creación de aplicaciones enriquecidas sobre diferentes plataformas que se basa en la Plataforma Flash.

La principal característica de Flex es que aporta un set de desarrollo (SDK) que favorece y agiliza el desarrollo de aplicaciones principalmente pensadas para el consumo de grandes cantidades de datos, y que tiene una vertiente más empresarial que para la que tradicionalmente estaba diseñado Flash Professional.

Debido a que, a partir del 2008, Flex fue liberado como Open Source, es posible desarrollar con el mismo usando IDE convencionales, como por ejemplo Eclipse, o con el IDE propietario de Adobe denominado Flash Builder (que internamente está basado en Eclipse).

Flex se combina con el lenguaje MXML (similar a XML) que permite definir elementos y comportamientos para una vista y que dispone de una gran cantidad de componentes especialmente diseñados y optimizados para el desarrollo de aplicaciones empresariales basadas en datos, en combinación con sistemas y herramientas como BlazeDS, Live DataCycle Services, ColdFusion, etc.

En su versión 4.5 comienza a soportar el desarrollo multidispositivo, y en versiones *premium* aporta herramientas adicionales que son clave para la optimización de aplicaciones, tales como *profilers* de memoria y rendimiento, sistemas de testeo automatizado, soporte para Flex Unit, capacidad de compilación desde línea de comandos, etc.

Enlaces relacionados

Apache Flex:

http://incubator.apache.org/flex/.

http://en.wikipedia.org/wiki/Apache_Flex.

Adobe Flex:

http://www.adobe.com/devnet/flex.html.

http://labs.adobe.com/technologies/flex/.

http://es.wikipedia.org/wiki/Adobe_Flex.

http://flex.org/.

Flash Builder:

http://www.adobe.com/es/products/flash-builder.html.

http://en.wikipedia.org/wiki/Adobe_Flash_Builder.

http://es.wikipedia.org/wiki/Adobe_Flash_Builder.

2.6. Adobe AIR

AIR es un *runtime* (*Adobe Integrated Runtime*) multiplataforma desarrollado por Adobe con el objetivo de permitir la creación de RIA que vayan más allá de un navegador, principalmente alrededor de la Plataforma Flash. Actualmente, se puede encontrar su última versión en desarrollo en Adobe Labs (http://labs.adobe.com/technologies/flashplatformruntimes/).

Mediante Adobe AIR se mantienen las herramientas y lenguajes de desarrollo empleados habitualmente para la creación de RIA en el navegador (Flash, Flex, HTML, Ajax, ActionScript, JavaScript, etc.), y se extienden las capacidades de Flash Player a otras plataformas como el escritorio y los dispositivos (en diferentes sistemas operativos) bajo una API unificada para que el desarrollador disponga de funciones generalmente reservadas a aplicaciones nativas.

Actualmente, soporta la creación de aplicaciones con funcionalidades nativas para sistemas operativos Windows, Mac, Linux, iOS, Android y BlackBerry Tablet, con un porcentaje de reutilización de código muy alto. Las aplicaciones realizadas con Adobe AIR son empaquetadas, firmadas digitalmente e instaladas en el sistema local de ficheros, integrándose de forma similar a una aplicación nativa en el sistema de archivos, *dock*, manejo de ficheros locales, bases de datos, etc.

En la actualidad, con Adobe AIR se tiene acceso a una serie de funcionalidades comunes que, salvo que el sistema operativo las limite, son totalmente funcionales, como por ejemplo:

- Instalación nativa de la aplicación.
- Acceso al sistema de ficheros del sistema (lectura y, si lo permite el OS, escritura).
- Almacenamiento persistente básico y avanzado con base de datos local.
- Acceso a la cámara.
- Acceso al micrófono.
- Acceso al GPS para geoposicionamiento.
- Acceso a sensores.
- Compatibilidad con el uso de *sockets*.

Enlace relacionado

Estas son solamente algunas de estas funcionalidades. Para conocer la lista completa, puedes consultar la página web http://www.adobe.com/products/air/features.html.

- Decodificación de vídeo con *multi-threaded* y *streaming* avanzado.
- Capacidad de renderizado HTML en su interior (instancia del navegador).
- Soporte para 2D y 3D con aceleración gráfica.
- API para trabajo con gestos y *multitouch*.
- Depuración por USB y WiFi en equipo de sobremesa desde un terminal.

En sus últimas versiones incorpora funcionalidades realmente interesantes como comunicación con extensiones nativas (http://help.adobe.com/en_US/air/extensions/index.html) desarrolladas "a medida" para cada plataforma, soporte para arquitectura nativa en 64-bits, renderizado de gráficos con aceleración por hardware, compresión LMZA para los ficheros SWF y codificación H.264. En el campo de los dispositivos móviles integra acceso a GPS, acelerómetro, cámara y micro, etc.

Enlaces relacionados

http://www.adobe.com/es/products/air.html.

http://www.adobe.com/devnet/air.html.

http://help.adobe.com/en_US/FlashPlatform/reference/actionscript/3/index.html.

http://en.wikipedia.org/wiki/Adobe_Integrated_Runtime.

http://es.wikipedia.org/wiki/Adobe_Integrated_Runtime.

3. Plataforma Flash para la creación de ARM

Sin perder de vista las diferentes tecnologías que se han evaluado en el apartado anterior, vamos a centrarnos en las relacionadas con la denominada **Plataforma Flash** para analizar qué ventajas nos aportan a la hora de trabajar con Aplicaciones Rich Media.

3.1. Capacidades multimedia

Una de las razones más importantes por las que la Plataforma Flash (en adelante nos podremos referir a ella de forma indistinta como Flash) es un referente en la creación de ARM, es el hecho de que desde sus inicios ha sido **enfocada al soporte avanzado de diferentes medios** tales como vectores, imagen, animación, audio, vídeo, etc.

Esta realidad ha hecho que todas las herramientas y lenguajes que giran en torno a la Plataforma Flash estén claramente orientados a este hecho, lo que en la actualidad la convierte, posiblemente, en el mejor entorno en conjunto para la creación de aplicaciones multimedia.

Tanto el IDE Flash Professional como el lenguaje de programación ActionScript están enfocados a la creación e interacción con estos tipos de contenidos, lo que se traduce en una mayor agilidad y velocidad en los ciclos de desarrollo.

El IDE permite combinar, con gran efectividad, animación tradicional basada en línea de tiempo con programación avanzada en un lenguaje maduro y orientado a objetos como es ActionScript 3. La combinación trabaja de igual forma con contenido vectorial, *bitmap*, etc., y permite trabajar con los recursos de forma interna a la aplicación principal o cargarla de forma externa.

La lista de tipos de recursos que una aplicación Flash puede gestionar es realmente extensa y, hoy en día, es probablemente la plataforma con mejor relación capacidad/rendimiento en conjunto de todos estos tipos de recursos.

Estas características base del Flash Player, extendidas mediante Adobe AIR con sus funcionalidades nativas, hacen que se convierta en una opción más que interesante cuando deseamos trabajar con aplicaciones basadas en múltiples medios enriquecidos.

3.2. Capacidades multiplataforma

Además de la amplia capacidad de gestión de diferentes tipos de recursos, otra característica que hace que Flash sea una gran elección para la creación de ARM es su **compatibilidad con múltiples plataformas**.

Hoy en día, no se ponen en duda las capacidades que ofrece Flash en el navegador, estando por delante de cualquier otra tecnología en el trabajo con muchos tipos de contenidos. Conjuntamente con Adobe AIR, se extiende este planteamiento a aplicaciones de escritorio e incluso a aplicaciones para dispositivos (no solamente móviles, sino actualmente otros como TV, por ejemplo).

Existen una serie de características que potencian esta capacidad multiplataforma de Flash y que le dan una ventaja importante frente a otras tecnologías:

- El entorno de desarrollo es común para trabajar con las diferentes plataformas.

- El lenguaje de programación es el mismo para trabajar con las diferentes plataformas, siendo ampliado mediante API y funciones extendidas específicas, pero con una base común.

- El hecho de trabajar sobre un mismo *runtime* reduce considerablemente las incompatibilidades entre plataformas, ya que es el propio entorno el que las gestiona internamente.

  Por ejemplo, el uso de una misma clase para el trabajo con el sistema local de ficheros, independientemente de si es un SO Windows o Mac.

- El proceso de desarrollo y producción de la aplicación final es muy similar, en general, entre todas las plataformas soportadas.

Todas las características anteriores repercuten muy positivamente en la curva de aprendizaje necesaria para que un desarrollador (o equipo de desarrollo) reutilice sus conocimientos para comenzar a trabajar sobre una nueva plataforma. De hecho, con Flash los cambios más importantes, al trabajar para diferentes plataformas, se encuentran en la mentalidad necesaria para cada una de ellas, más que en el hecho de tener que adaptar conocimientos técnicos.

3.3. Capacidades multidispositivo

Cuando se analizan las capacidades multidispositivo de una herramienta, entran en juego muchos factores a tener en cuenta. Dependiendo de las capacidades que aporta una herramienta, se pueden tener importantes ventajas en el desarrollo.

Algunos de los aspectos que afectan notablemente en el trabajo con desarrollos que pretenden correr en diferentes dispositivos son:

- Dimensiones físicas de la pantalla.
- Densidad de píxeles de la pantalla.
- Capacidades de interacción (táctil, teclado, etc.).
- Limitaciones de memoria.
- Limitaciones de procesador.
- Acceso a capacidades específicas del dispositivo.
- Conectividad y optimización de carga de datos.
- Capacidad de instalación "nativa".

En relación con alguno de estos puntos, las características innatas de Flash aportan beneficios importantes, como por ejemplo el poder trabajar con contenido vectorial frente a diferentes tamaños de pantalla y densidad de píxeles, o gestionar la calidad del renderizado frente a limitaciones de procesador.

Además, Adobe AIR ha logrado (en sus últimas versiones) homogeneizar el acceso a las capacidades específicas del dispositivo entre sistemas operativos iOS, Android y BlackBerry Tablet, y acercar el rendimiento de las aplicaciones entre ellos.

Por ejemplo, la clase flash.sensors.Geolocation permite trabajar con una misma clase y métodos para diferentes plataformas móviles.

En el desarrollo actual bajo la Plataforma Flash, y salvando las limitaciones impuestas específicamente por algunos fabricantes (la más destacada de ellas es la prohibición de Apple de poder cargar un SWF externo e interactuar con él), se pueden unificar en gran manera desarrollos creados conjuntamente para iOS y Android, siendo en algunos casos la compatibilidad del 100%.

Finalmente, la Plataforma Flash solventa el tema de la creación de instaladores "nativos", que es un punto que, por más que sea simple, no deja de ser de vital importancia, ya que es el primer contacto que el usuario tiene con nuestra aplicación y ha de ser lo más estandarizada posible en cada plataforma.

3.4. Capacidad de gestión de recursos externos

Otro factor importante, a la hora de desarrollar ARM, es la **comunicación y carga de contenidos externos**. En los tiempos en los que Internet está marcando una filosofía de vida, la mayoría de las aplicaciones requieren de conectividad para desplegar todas sus funcionalidades, y no estamos hablando solamente de datos, sino de los propios contenidos en diferentes formatos.

Cuando se trabaja con diferentes formatos de forma externa se ha de disponer de mecanismos eficientes para solicitar la carga, gestionarla y monitorizarla. Flash realiza estos procesos de forma muy similar, independientemente del tipo de contenido que se cargue, y permite controlar la carga en la petición, durante la petición y tras la petición.

Con respecto a la carga de información entendida como datos, Flash soporta multitud de formatos tales como texto plano, parejas variable / valor, XML y JSON, entre otros. Amplía estos mecanismos básicos con el soporte para *sockets* (que permiten el desarrollo de aplicaciones multiusuario en tiempo real), *web services* e incluso datos serializados en formato binario a través de AMF (Action Message Format) que potencian notablemente la comunicación entre cliente y servidor.

Por si esto fuera poco, cuenta con mecanismos de trabajo con determinados tipos de contenido que son vitales para una mejor experiencia de usuario, como es el caso del soporte de carga de audio y vídeo en *streaming*. A día de hoy, Flash es claramente dominador en estos campos, ya que aunque HTML5 haya mejorado notablemente las capacidades de los estándares en estos frentes, su soporte y capacidades son muy limitados con respecto a la tecnología de Adobe.

4. Consideraciones

Por todos estos factores y más características que sería imposible enumerar en detalle, la Plataforma Flash es hoy en día una posibilidad muy interesante y real en el desarrollo de ARM.

Desde el punto de vista técnico y de desarrollo ofrece ventajas importantes, como son sus capacidades multimedia, multiplataforma y multidispositivo. Además, y gracias a la reutilización de herramientas y código, desde el punto de vista empresarial (tanto en equipos de desarrollo o de profesionales individuales) ofrece igualmente grandes ventajas por las reducciones de tiempos y costes que conlleva el hecho de poder aprovechar en gran medida los conocimientos existentes en un entorno a la hora de adentrarse en otros diferentes.

No obstante, siempre es fundamental tener en cuenta que **no existe una tecnología preferible de forma absoluta**. Cada proyecto, en función de sus características, objetivos y condicionantes, puede obligarnos a tomar una decisión diferente en cuanto a la tecnología y a las herramientas a emplear.

La tecnología (y sobre todo en la parte cliente) es muy cambiante y está expuesta a las tendencias que imponen fabricantes, sistemas operativos, etc.; por ello, es importante tener un conocimiento global amplio, para poder tomar decisiones acertadas a la hora de escoger nuestras herramientas de trabajo.