

Introducción

A continuación, en este primer capítulo, presentamos a Ajax y sus conceptos fundamentales: arquitectura del modelo, tecnologías componentes y características de funcionamiento.

| | |
|--|-----------|
| Conceptos básicos | 14 |
| ¿Qué es Ajax? | 15 |
| Diferencias entre aplicaciones web tradicionales e interfaces con Ajax | 17 |
| Dónde y cuándo utilizamos Ajax | 19 |
| El proceso Cliente-Servidor en Ajax | 21 |
| Usabilidad y limitaciones del modelo Ajax | 23 |
| Alternativas | 26 |
| Resumen | 27 |
| Actividades | 28 |

CONCEPTOS BÁSICOS

Ajax es una nueva manera de utilizar tecnologías ya existentes, todas ellas muy conocidas y accesibles, entre las que podemos citar a:

- (X)HTML
- CSS
- JavaScript
- DHTML
- DOM
- XML
- XSLT

(X)HTML (*eXtensible HyperText Markup Language*) y **CSS** (*Cascading Style Sheets*) se utilizan para enmarcar y definir la estructura y la presentación de un documento, dentro del cual será posible incluir elementos cuyo contenido pueda ser modificado de forma dinámica mediante **DHTML** (*Dynamic HTML*). Para localizar y acceder a tales elementos (por ejemplo DIV y SPAN) existe **DOM** (*Document Object Model*), disponible desde **JavaScript**. Nada nuevo.

Lo interesante consiste en que ese contenido puede recuperarse desde el servidor, de forma asincrónica, sin tener que actualizar la página completa: JavaScript permite, a través de un objeto llamado **XMLHttpRequest**, enviar y recibir datos entre un navegador y un servidor web.

Además, los datos recuperados desde el servidor pueden estar en formato XML (aunque no necesariamente) y ser tratados con XSLT en el lado cliente.

En resumen, las tareas clásicas en una aplicación basada en Ajax son:

- cargar la interfaz de usuario;
- realizar peticiones al servidor;
- actualizar la interfaz con los datos recuperados.

En la interfaz se incluyen tanto los aspectos relacionados a la presentación como las funciones JavaScript para actualizar los datos contenidos en ella.

Las respuestas recuperadas desde el servidor tienen la característica de ser pequeñas en tamaño, por lo que la actualización resulta muy rápida y dará la sensación de ser instantánea a la vista del usuario.

Las tecnologías sobre las cuales se basa Ajax han sido probadas y pueden considerarse maduras y no transitorias. Esto deriva en que también Ajax lo sea.

Encontraremos una breve referencia a cada una de estas herramientas y al objeto **XMLHttpRequest** en el próximo capítulo.

Hasta ahora, hemos nombrado tecnologías que se ejecutan en el lado cliente, pero una vez que el objeto **XMLHttpRequest** logra llevar una petición al servidor, nada impide allí recuperar información desde bases de datos, servicios web, o aplicar procesamientos por medio de lenguajes que se ejecuten del lado del servidor, como PHP, JSP o ASP .NET, por citar sólo algunos. Luego, el resultado es retornado a la aplicación cliente desde la cual se generó la petición.

¿Qué es Ajax?

El término fue concebido por **Jesse James Garrett** (autor de diversos artículos y libros, y fundador del sitio www.adaptivepath.com). Es un acrónimo que refiere a un conjunto de tecnologías muy populares, abiertas y accesibles: *Asynchronous JavaScript And XML*.

The screenshot shows a Microsoft Internet Explorer browser window. The title bar reads "adaptive path » ajax: a new approach to web applications - Microsoft Internet Explorer". The address bar contains "http://adaptivepath.com/publications/essays/archives/000385.php". The website header features the "adaptive path" logo and a navigation menu with "publications" highlighted. Below the header, a sub-menu shows "essay archives" selected. The main content area displays the article "Ajax: A New Approach to Web Applications" by Jesse James Garrett, dated February 18, 2005. The article text begins with "If anything about current interaction design can be called 'glamorous,' it's creating Web applications." A sidebar on the right titled "Recent Essays" lists several other articles, including "A Conversation with Michael Bierut" and "Sphere: Balancing Power and Simplicity".

Figura 1. El termino Ajax se usó por primera vez en un artículo publicado en el sitio web de **Adaptive Path**.

Ajax define conceptos acerca de la interacción de un usuario con una aplicación web, y esos conceptos están por encima de las herramientas que se utilicen: el acrónimo derivado en nombre, sólo sirve a modo de guía, ya que, como veremos más adelante, intervienen otros protagonistas en el modelo. Algunas de estas herramientas serán abordadas en próximos capítulos.

Se dice que Ajax no es una tecnología, sino una serie de tecnologías que trabajan en conjunto:

- presentación mediante (X)HTML y CSS;
- contenido dinámico utilizando DOM y DHTML;
- intercambio y manipulación de datos usando XML y XSLT;
- comunicación asincrónica mediante **XMLHttpRequest**;
- JavaScript cliente para concatenar todo lo anterior.

Incluso puede haber otras, que, utilizando el mismo concepto, abran el abanico de posibilidades a la hora de desarrollar aplicaciones con Ajax. Si bien el modelo toma, con frecuencia, ciertas herramientas para ejemplificar su funcionamiento, nada impide reemplazarlas.

En el artículo en que se introduce el término, Ajax se presenta como una arquitectura en relación con las partes que intervienen en una aplicación web, pero también como un conjunto de herramientas específicas para implementar dicha arquitectura: ambas acepciones son válidas.

Ajax forma parte del movimiento conocido como **Web 2.0** (término incorporado por *O'Reilly Media* y *MediaLive International* en una serie de conferencias dadas en 2004, que luego fue aceptado casi de forma unánime por la comunidad de desarrolladores). Éste mantiene como principio el fomento de herramientas y tecnologías que permitan participar de manera activa al usuario, interactuando con el sitio web, evitando las largas demoras (transiciones entre cliente y servidor) que dificultan la experiencia de uso. Por este mismo motivo, Web 2.0 es también conocido como *Participatory Web*, **Web Participativa**.



MICROSOFT

Microsoft fue la primera empresa en introducir el objeto XMLHttpRequest (XMLHttpRequest). Este elemento es la piedra angular en el modelo de aplicación de Ajax y fue implementado mediante un objeto ActiveX. Lo veremos en detalle en el próximo capítulo

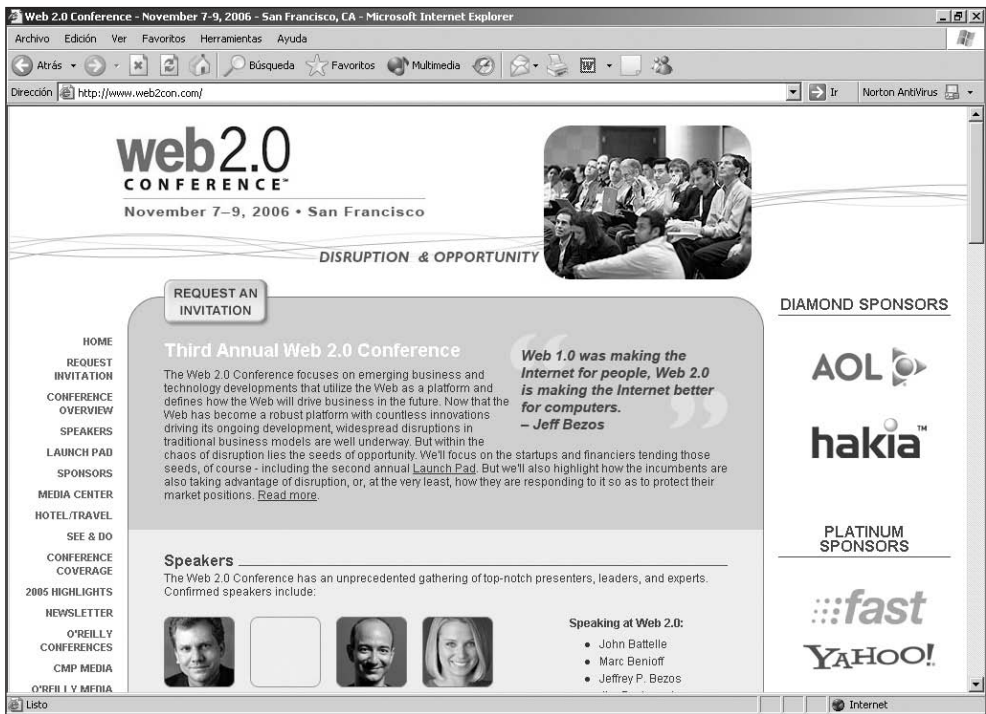


Figura 2. Ajax es uno de los componentes del movimiento Web 2.0.

Diferencias entre aplicaciones web tradicionales e interfaces con Ajax

En una aplicación web tradicional, el cliente envía una petición al servidor que, luego del procesamiento correspondiente, devuelve el resultado. Algunas características de este modelo:

- entre la petición y la respuesta se salta de una página a otra (a veces, puede ser la misma, pero se debe modificar el documento para ver reflejados los cambios);
- normalmente, no es necesario modificar todas las partes de la página, sino sólo algunas, y, sin embargo, se vuelve a cargar información que no ha sido modificada, lo que conlleva una carga innecesaria para el servidor.



JAVASCRIPT

El lenguaje de programación JavaScript puede emplearse tanto del lado servidor como del cliente, sin embargo, en este libro, tomaremos su utilización en todos los casos del lado cliente. Cabe destacar que éste es un lenguaje interpretado, y no es posible compilarlo, por lo cual el código fuente será visible para quien ejecute la página web.

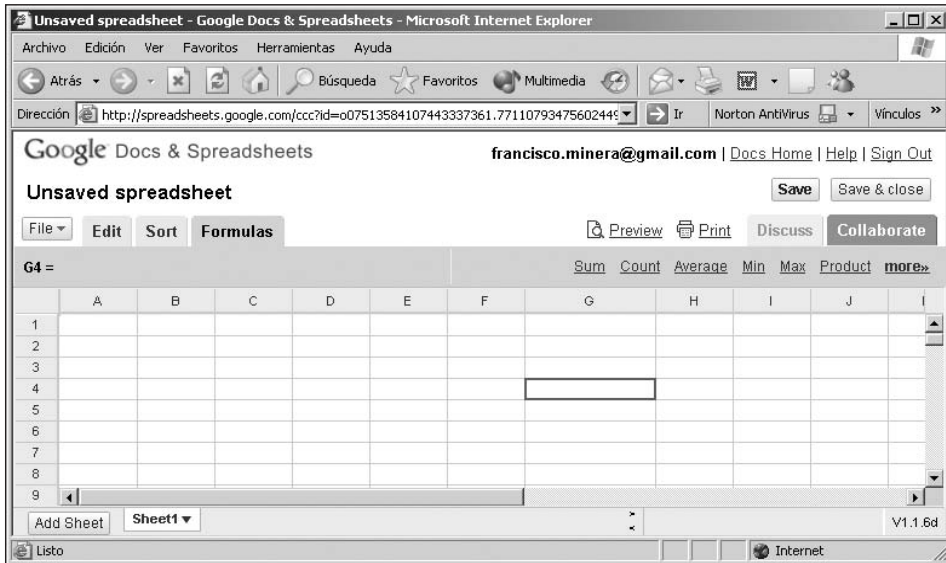


Figura 3. Aplicaciones de escritorio típicas como las hojas de cálculo comienzan a tener versiones web.

- en referencia a los puntos anteriores, el usuario deberá esperar cierto tiempo (mayor o menor según aspectos como la velocidad de conexión, el tiempo de procesamiento, el tráfico de red, etcétera) para visualizar una salida similar a la entrada.

Uno de los objetivos de Ajax es acercar la funcionalidad de las aplicaciones web (web applications) a la interactividad ofrecida por las aplicaciones de escritorio (desktop applications), o sea, poder recuperar datos desde el servidor sin tener que actualizar la página completa.

La sensación que un usuario experimenta al interactuar con una aplicación de escritorio puede representarse mediante palabras como rapidez, seguridad, simplicidad, lógica, instantaneidad, e interoperabilidad.

En su artículo, Jesse James Garrett pone de manifiesto la diferencia entre el auge que el hipertexto (acceso a documentos mediante enlaces) ha tenido en los sitios web desde los comienzos y la funcionalidad que demanda una aplicación web actual en relación con lo que el usuario experimenta frente a ellas. Finalmente, concluye en que Ajax intenta brindar la sensación ofrecida por una aplicación de escritorio sin perder las posibilidades alcanzadas hoy en día por las aplicaciones web.

Cuando se produce una comunicación con el servidor, se recuperan datos, y se actualiza la interfaz de usuario. Al navegar, se notará una mayor rapidez que la usual en las aplicaciones web tradicionales y podrá verse cómo el contenido cambia según las peticiones, sin advertir lo que sucede internamente.

Dónde y cuándo utilizamos Ajax

Una aplicación Ajax se ejecuta en entornos web, en navegadores que brinden un soporte completo para XML (en caso de utilizarse este formato para el intercambio de datos) y para el objeto **XMLHttpRequest** de JavaScript.

Actualmente, sólo dos navegadores cumplen con estos requisitos: **Internet Explorer (IE)** y **Mozilla Firefox**. Otros, como **Safari** y **Opera**, dan soporte parcial para XML (lo que no significa que las aplicaciones Ajax no funcionen en ellos).

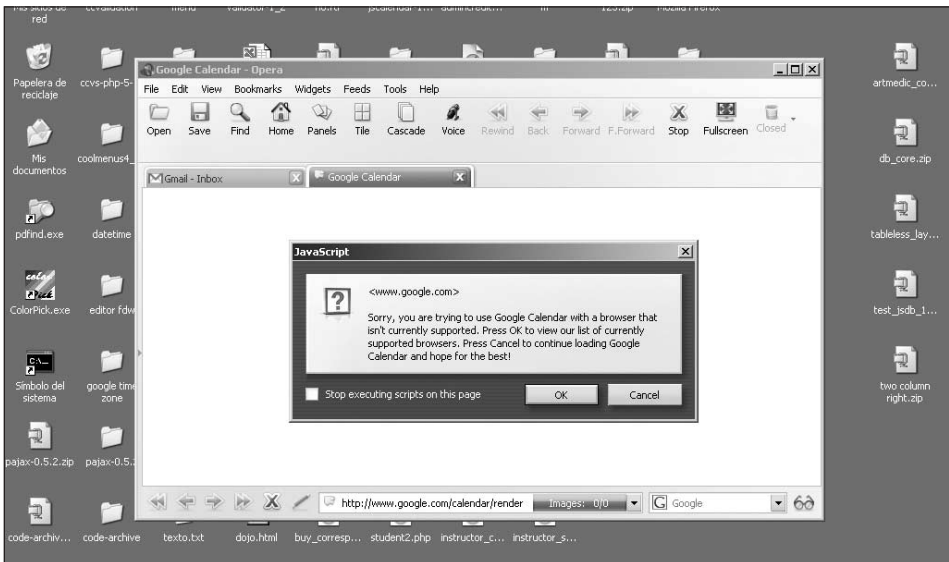


Figura 4. El soporte hacia Ajax por parte de los navegadores es un desafío permanente tanto para las empresas que los desarrollan como para los programadores de aplicaciones web.

Aplicaciones como **Gmail**, **Google Maps**, **Google Suggest**, o **Flickr** han sido desarrolladas mediante Ajax. Esto se nota a simple vista observando cómo, en determinadas situaciones, la interfaz actualiza sus datos sin necesidad de recargar la página completa: el corrector ortográfico de Gmail, la navegación de mapas en Google Maps, las sugerencias ofrecidas por Google Suggest o la edición de los títulos y descripciones de las fotografías en Flickr.

El hecho de que una empresa como Google utilice Ajax cada vez con mayor frecuencia en sus desarrollos (también podemos citar a **Orkut** y **Google Groups**, otros productos de la compañía) no hace más que afianzar y popularizar este modelo. Si sumamos que se usa cada vez más en situaciones reales, es decir, que pueden verse aplicaciones en funcionamiento más allá de las de Google en la red, y que entornos como **.net** o diversos lenguajes de programación acompañan de

buena manera este movimiento, concluimos que Ajax ha llegado para quedarse y seguir evolucionando.

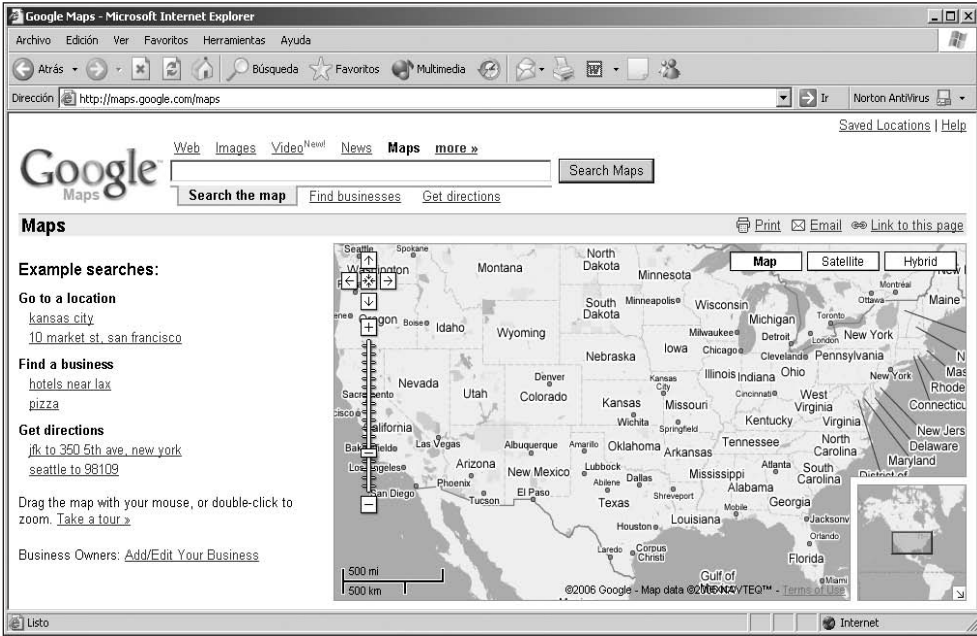


Figura 5. Google es una de las empresas pioneras en la utilización de Ajax.

Gmail: <http://mail.google.com>

Google Maps: <http://maps.google.com>

Google Suggest: www.google.com/webhp?complete=1&hl=en

Flickr: www.flickr.com

Notemos que las aplicaciones antes citadas van desde las simples hasta las complejas. Ajax es una manera de plasmar desarrollos, su vinculación con las aplicaciones resulta de lo más variada: no es una técnica sofisticada que sólo tiene utilidad en sistemas complejos, sino que puede amoldarse a situaciones reales y solucionar inconvenientes cotidianos y comunes.



TIEMPOS

Todas las herramientas componentes de Ajax están disponibles desde hace tiempo, y la aparición reciente del nuevo modelo tiene que ver con la experiencia lograda por parte de una gran cantidad de desarrolladores y de su creatividad para unir las piezas.

EL PROCESO CLIENTE-SERVIDOR EN AJAX

El punto de partida, la pregunta inicial que motiva el modelo propuesto por Ajax es cómo se conecta un cliente (un navegador) con un servidor.

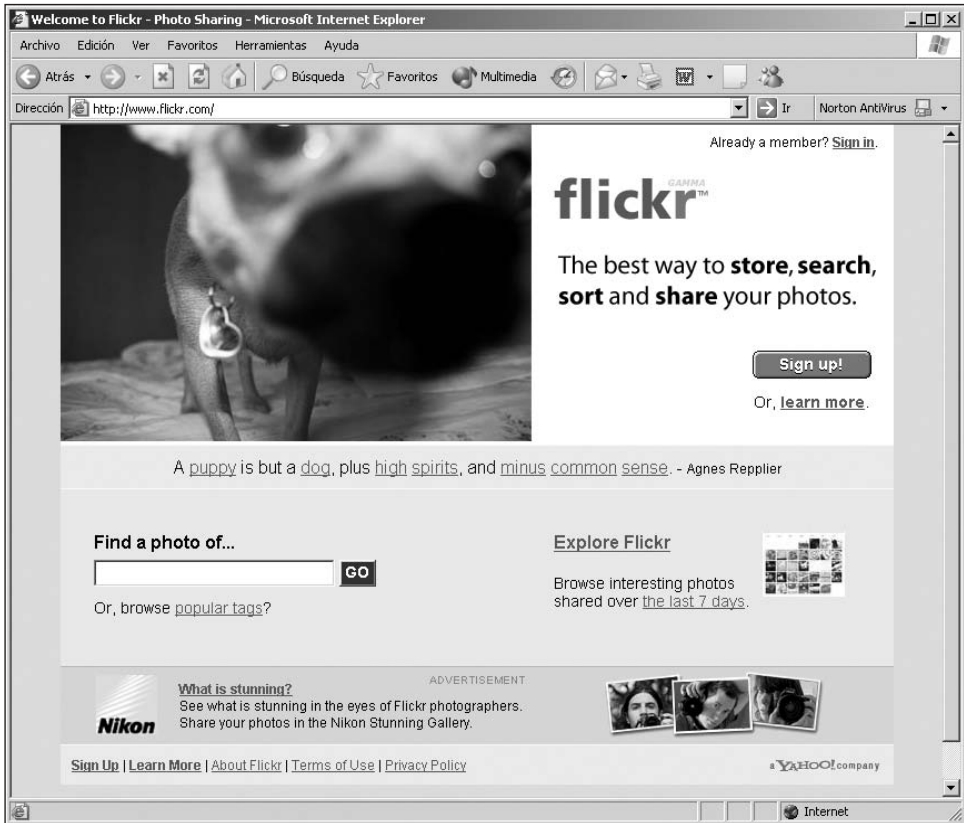


Figura 6. Flickr utiliza técnicas Ajax para administrar y catalogar imágenes (www.flickr.com).

Al introducir las diferencias entre las aplicaciones web tradicionales y las basadas en Ajax, uno de los temas tratados fue el de la recarga de páginas completas aun

★ XML

El metalenguaje XML aparece formando parte del acrónimo Ajax como formato preferido, por diversos factores (simplicidad y capacidad de extensión) para intercambiar datos con el servidor; sin embargo, veremos que no es la única alternativa para este fin.

cuando gran parte de la información contenida en ellas no sufriera modificaciones, y se habló también de interfaces Ajax.

El modelo Ajax introduce un intermediario entre el cliente (navegador) y el servidor o, si se quiere, divide al cliente en dos partes: la presentación (interfaz de usuario) y el motor Ajax (aplicación escrita en JavaScript).

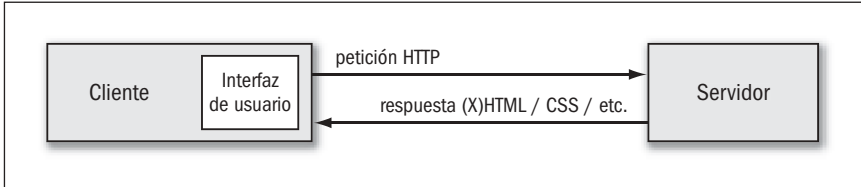


Figura 7. El modelo web tradicional difiere del utilizado por Ajax.

La interfaz de usuario se comunica con el motor Ajax a través de JavaScript, y éste envía una petición (en background, segundo plano) al servidor mediante el objeto **XMLHttpRequest**. Una vez que el servidor completó el procesamiento, devuelve la respuesta (en formato XML, por ejemplo) al motor Ajax, que a su vez actualiza datos en la interfaz (que se mantiene durante todo este proceso a la vista del usuario) mediante DHTML y DOM.

De alguna manera, las peticiones HTTP terminan siendo llamadas al servidor desde el motor Ajax y mediante instrucciones JavaScript.

Otra de las características es el balanceo de la carga de trabajo hacia el lado cliente, en vez de dejar todo en manos del servidor. Ésta es una discusión que resurge de tanto en tanto. Lo que Ajax propone, en relación con esto, es que el cliente deba recurrir al servidor sólo en aquellos casos en los que resulte absolutamente necesario hacerlo. Al adoptar este criterio, no se malgasta el ancho de banda de la red ni el tiempo del usuario a la espera de datos innecesarios que no han sido modificados en el servidor.

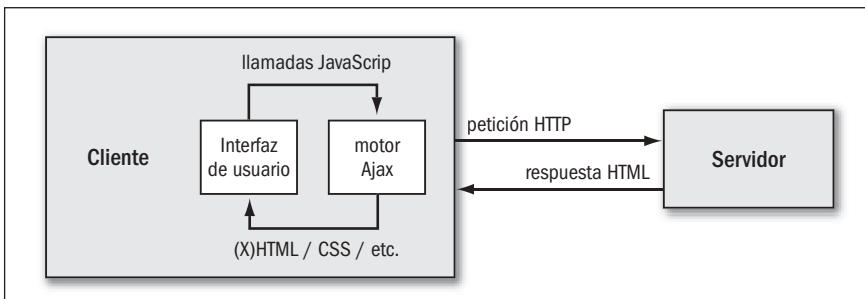


Figura 8. El modelo Ajax incorpora un intermediario entre la interfaz de usuario y el servidor.

Ajax propone que la primera carga de la página inicialice la interfaz de usuario y las funciones JavaScript para tratar los datos e intercambiarlos (actualizarlos) comunicándose con el servidor.

Si bien XML es el formato casi siempre utilizado para recuperar datos desde el servidor, cualquier otro es válido (HTML, texto plano, y JSON —ver Apéndice C— son apenas algunos ejemplos).

En el modelo, el lado cliente cumple una función importante: no sólo recae sobre él la tarea de mostrar los datos preprocesados obtenidos del servidor, sino también la responsabilidad de petitionar información, recuperarla, tratarla, adecuarla al contexto e, incluso, actualizarla, todo de forma dinámica.

La *A* de Ajax corresponde a *Asynchronous*: la **asincronía** es la capacidad que una aplicación posee de manejar procesos independientes de otros. En el mismo sentido, **sincrónico** expresa la dependencia entre procesos.

En el caso de las aplicaciones web tradicionales, cuando un navegador realiza una petición, la actividad del usuario se interrumpe hasta que se devuelve la respuesta. Los procesos en el lado cliente y servidor son sincrónicos, un proceso depende del otro, y no puede continuar hasta que finaliza.

En las aplicaciones Ajax, la actividad del usuario no se interrumpe totalmente, puesto que una de las características del modelo consiste en mantener la interfaz de usuario y bloquear de manera transitoria sólo una parte de ella. Ésta se actualizará al momento de recuperar la respuesta del servidor.

USABILIDAD Y LIMITACIONES DEL MODELO AJAX

Ajax propone cambiar de lugar ciertas fichas del tablero para proporcionar al usuario una experiencia más cómoda y elegante al acceder a una aplicación web.

El movimiento en la arquitectura de las aplicaciones basado en tecnologías probadas deja en manos de diseñadores y desarrolladores la responsabilidad y la libertad para construir soluciones innovadoras.

Estas soluciones toman como centro al usuario: no es el caso típico en el que se depende de una empresa u organización para tener a disposición más herramientas y así lograr desarrollos competitivos; aquí lo que prima es la originalidad y la búsqueda de nuevas opciones y facilidades.

Por todo esto, el tipo de aplicaciones a las que Ajax se dirige es todavía un punto que no está claro, porque surgen de manera constante nuevos sistemas disímiles entre sí tanto en funcionalidad ofrecida como en complejidad y perfil de los usuarios a los que están encaminados.

Entre las limitaciones y aspectos por tener en cuenta, podemos citar:

- **Habilitación de JavaScript:** uno de los basamentos de Ajax se ubica en este lenguaje. Si bien la mayoría de los navegadores actuales lo soporta, no se puede dar por cierto que todos los usuarios utilicen navegadores actuales. También es posible deshabilitar el soporte completo —o determinadas opciones— de JavaScript, lo que atentaría contra el correcto funcionamiento de las aplicaciones basadas en Ajax. En el caso de **Internet Explorer 6**, será necesario tener activado el soporte para objetos **ActiveX** ya que la implementación de **XMLHttpRequest** así lo requiere.

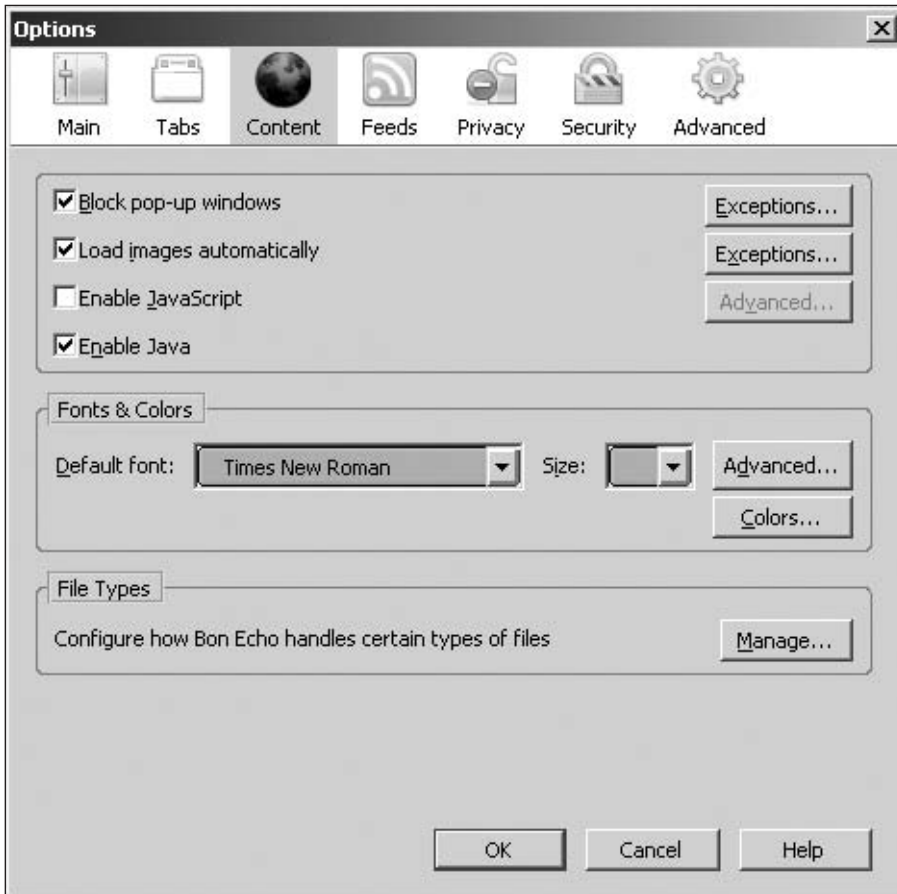


Figura 9. Desde los navegadores web, es posible desactivar el soporte para JavaScript.

- **Compatibilidad:** al estar compuesto por tecnologías que operan del lado cliente, surge el problema de la compatibilidad entre navegadores; algunos interpretan de forma diferente ciertas opciones, otros ni siquiera las interpretan. Éste es sin dudas un punto débil para cualquier modelo que deje una parte importante del funcionamiento de una aplicación web en el lado cliente, o bien, se base en gran parte en su configuración. Con la implementación correcta de estándares por parte de los distintos navegadores, se podría evitar este tipo de conflictos.
- **Seguridad:** acceder a datos almacenados en un servidor desde una aplicación cliente siempre supone una validación de datos y una autenticación de usuarios estrictas. Ésta no es, por cierto, una limitación, pero sí un aspecto de particular importancia.
- **Interacción entre la aplicación, el navegador y el usuario:** el nuevo modelo supone el análisis de ciertas situaciones que, tradicionalmente, no generaban mayores inconvenientes:
 - **Guardar documentos:** al generar contenido de forma dinámica en el lado cliente, si se almacena una página puede que su contenido (al visualizarse offline) no refleje lo que el usuario estaba viendo al momento de guardarla en disco. En el mismo sentido, surge el problema de agregar una página particular a la lista de favoritos.
 - **Historial:** en una aplicación web clásica, el usuario puede, mediante el navegador, ir hacia la página anterior o a la siguiente —si es que las hay— del historial. En Ajax, eventualmente, no se cambiará de página (sólo se modificará parte de su contenido, de acuerdo con cada proceso) por lo que las teclas **atrás** y **adelante** no funcionarán y habrá que buscar formas alternativas para moverse (modificando el historial con JavaScript o utilizando iframes invisibles para generarlo) dentro de la aplicación web.
 - **Demora:** normalmente, el tiempo muerto mientras se espera que el servidor retorne una respuesta es notorio por aspectos como la pantalla en blanco o señales dadas por el navegador (**descargando desde**). En una aplicación Ajax, las peticiones se hacen en segundo plano, por lo que habrá que buscar maneras alternativas para informar al usuario que se está realizando esta tarea, mediante barras de carga propias o elementos animados.



PORTABILIDAD

Una de las características de las aplicaciones Ajax es que no necesitan plug-ins o controladores extras para interactuar con ellas, incluso, entre plataformas diferentes. Esto es una enorme ventaja en sistemas híbridos, que posean más de un lenguaje del lado del servidor como capa de unificación en la interfaz de usuario.

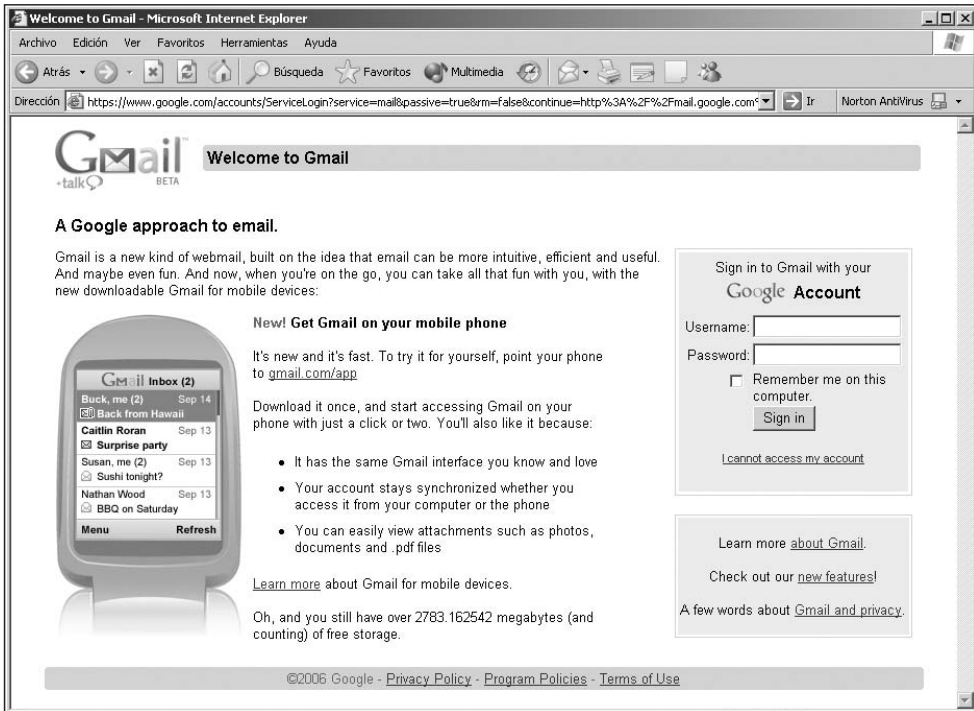


Figura 10. Gmail es una de las aplicaciones Ajax más populares (www.google.com/mail).

ALTERNATIVAS

Ajax no propone reemplazar la arquitectura que las aplicaciones web vienen utilizando de forma masiva desde hace tiempo, sino que se postula como alternativa en ciertas ocasiones e, incluso como complemento, en otras.

Si bien el término Ajax es de reciente aparición (2005), las conexiones asincrónicas desde el lado cliente sin la necesidad de recargar la página completa tienen antecedentes que, en la actualidad, pueden constituirse en alternativas:

- Los elementos **iframe** —disponibles en Internet Explorer a partir de 1996— y **layer** —Netscape Navigator, 1997— ofrecen la posibilidad de cargar documentos externos y, eventualmente, modificar desde esos archivos las páginas contenedoras de los elementos. En la actualidad, todavía se utiliza el elemento **iframe** en lugar de **XMLHttpRequest** para algunos desarrollos Ajax.
- **Microsoft's Remote Scripting** (1998) utiliza un applet de Java que puede comunicarse con el cliente mediante JavaScript. A pesar de sus años, esta herramienta funciona tanto en Internet Explorer como en Netscape Navigator, pero sólo en

plataformas Windows y, preferentemente, con servidores IIS (*Internet Information Server*) y el lenguaje ASP (*Active Server Pages*), además de requerir JVM (*Java Virtual Machine*) instalado y habilitado. Tiempo después de su desarrollo este applet fue reemplazado por **XMLHttpRequest**.

- **JavaScript Remote Scripting** (JSRS, 2000): librería escrita en JavaScript que utiliza DHTML, pero sólo trabaja de forma asincrónica.
- **Internet Explorer: download Behavior**: trabaja de forma similar a JSRS, pero exclusivamente con Internet Explorer. Webservice Behaviour permite el trabajo con el protocolo SOAP.
- **XML-RPC**, antecesor de SOAP: está disponible en variados lenguajes y utiliza XML y HTTP con llamadas a procesos remotos (RPC).
- **JavaScript on Demand** (2002): esta alternativa al JavaScript tradicional, funciona de manera parecida, reduce el tiempo de descarga en librerías extensas.
- **ARSCIF** y **Callbacks** (*frameworks*), y **SVGT**: este último es un protocolo que emplea conexiones persistentes.

RESUMEN

Acabamos de listar las principales características de Ajax: conceptos fundamentales, herramientas componentes, casos de uso, relación con los usuarios, arquitectura y limitaciones del modelo. Además de comentar las relaciones entre las aplicaciones de escritorio y las que utilizan Ajax, remarcamos las diferencias entre estas últimas con referencia a las aplicaciones web tradicionales.



TEST DE AUTOEVALUACIÓN

- 1** ¿Ajax es una arquitectura, un conjunto de herramientas, o ambas cosas?

- 2** ¿Cuáles son las herramientas componentes de Ajax?

- 3** ¿Qué función cumple el llamado motor Ajax?

- 4** ¿Qué características tiene una aplicación Ajax?

- 5** ¿Qué las diferencia de las aplicaciones web tradicionales?

- 6** ¿Qué características Ajax tiene Google Maps?

- 7** ¿Qué características Ajax tiene Gmail?

- 8** Buscar y enumerar tres aplicaciones que trabajen con Ajax.

- 9** ¿Cuáles son las limitaciones del modelo?

- 10** ¿Cuáles son las coincidencias entre una aplicación de escritorio y una aplicación Ajax?
