PONTIFICIA UNIVERSIDAD CATÓLICA DEL ECUADOR
FACULTAD DE INGENIERÍA
ESCUELA DE SISTEMAS

DISERTACION PREVIA A LA OBTENCION DEL TÍTULO DE INGENIERO EN SISTEMAS

"ANÁLISIS COMPARATIVO DE LAS TECNOLOGÍAS ORIENTADAS AL DESARROLLO DE APLICACIONES WEB RIA (RICH INTERNET APPLICATION) Y SU IMPACTO EN EL FUTURO DEL INTERNET"

JAIME GUSTAVO PROAÑO BÁEZ
DIRECTOR: ING. OSWALDO ESPINOSA

QUITO, 31 DE MAYO DE 2013

Dedicatoria

A mis hijos, Gabriel y Agustina.

Piensen siempre que ustedes pueden lograr

todo lo que en la vida se propongan,

que nadie les diga lo contrario.

Agradecimiento

A mi papi y mi mami por su esfuerzo y aguante, a mi negrita por su apoyo y paciencia incondicionales, te amo mi negra. A mis amigos y amigas que han estado en las malas y peores, a mis hermanas, mi hermano, sobrinos y sobrinas por alegrarme la vida con sus historias de lucha diaria, superación y éxitos.

TABLA DE CONTENIDOS

CAPÍTULO I

1 Comportamiento de las Aplicaciones Enriquecidas de Internet RIA (Rich Internet Applications)

1.1 Evolución de las páginas web

La idea original al desarrollar la web fue permitir a los investigadores compartir información relacionada por medio de hipertexto[1]. En el año de 1980 Tim Berners-Lee desarrolló un primer proyecto de software denominado ENQUIRE[2]. El mismo era un programa escrito en Pascal capaz de encontrar información relacionada a partir de una base de datos, presentaba hipervínculos bidireccionales y permitía una edición directa de la información en el servidor. Es a partir de este proyecto que empieza a materializarce la realización práctica del concepto que actualmente conocemos como Web.

En Marzo de 1989 Tim Berners-Lee escribe un documento[3] que sirvió de base para presentar el proyecto posterior del World Wide Web en 1990. Este documento pretendía persuadir al CERN[4] para que la gestión de la información se realice por medio de un sistema global de hipertexto, para

---

[1] El término hipertexto aparece por primera vez en el año de 1965

[2] El término se refería a Enquire Within Upon Everything, en castellano *Preguntando de Todo Sobre Todo*.

[3] El documento original se puede encontrar en http://www.w3.org/History/1989/proposal.html.

[4] La Organización Europea para la Investigación Nuclear comunmente conocida por la sigla CERN utilizadas para Consejo Europeo para la Investigación Nuclear, es el mayor laboratorio de investigación en física de partículas a nivel mundial.

ese entonces a este modelo de distribución de información el autor lo denominó “mesh” (malla). El problema que trataba el documento era la dificultad que se presentaba al tratar de hacer el seguimiento a la información que se generaba en proyectos de tamaño considerable en el CERN, dado el volumen de información que se manejaba en los mismos, lo que a la larga ocasionaba la pérdida de información relevante; la solución derivó entonces en un sistema distribuido de hipertexto.

En mayo de 1990 Tim Berners-Lee hace circular su propuesta nuevamente pero modificada y escrita de una manera más formal junto al científico Robert Cailliau co-autor. Para el mes de septiembre del mismo año Mike Sendall, jefe de división da el visto bueno para la adquisición de un servidor NeXT[5] cube y aprueba el inicio del desarrollo del sistema global de hipertexto. El trabajo empieza en octubre y mediante la utilización del ambiente de desarrollo de NeXTStep[6] se escribe el programa de interfaz de usuario que comprendía un navegador y un editor de hipertexto, el nombre que se dio al programa fue “WorldWideWeb”.

---

[5] ***NeXT Computer***, Inc. (cuyo nombre cambió a NeXT Software, Inc.) fue una compañía informática, conocida entre el público por sus avanzados ordenadores y en el mundo de la programación por sus plataformas de desarrollo (orientadas a objetos). NeXT se fusionó con Apple Inc.

[6] ***NeXTSTEP*** es el sistema operativo orientado a objetos, multitarea que NeXT Computer, Inc. diseñó para correr en los ordenadores NeXT (informalmente conocidos como "black boxes").

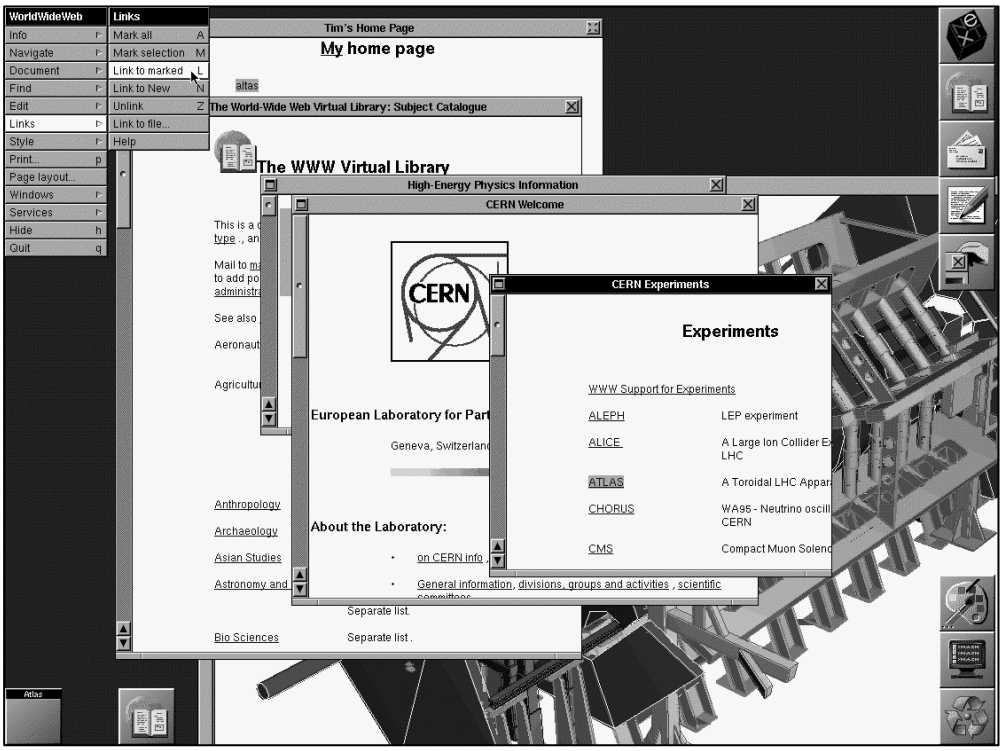

Fig. 1 – Interface del primer navegador de hipertexto WorldWideWeb.

Fuente: http://www.w3c.es

Entre octubre y diciembre de 1990 el equipo de desarrollo se incrementó y a finales de año se logran hacer las primeras pruebas con éxito. Estas pruebas involucraron el acceso a archivos de hipertexto, artículos de noticias de Internet, así como búsquedas específicas en CERNMV[7].

El 6 de agosto de 1991, el equipo de desarrollo encabezado por Berners-Lee envió un pequeño resumen del proyecto WorldWideWeb al newsgroup[8] alt.hypertext. Esta fecha es crucial para el desarrollo de la web pues señala el debut de la web como un servicio disponible públicamente en Internet.

---

[7] CernVM es un software virtual de aplicación base diseñado para los participantes de los experimentos del CERN LHC (Colisionador de Hadrones).

[8] Resumen del proyecto WorldWideWeb enviado por Tim Berners-Lee a newsletter: http://groups.google.com/group/alt.hypertext/msg/395f282a67a1916c

El concepto, subyacente y crucial, del hipertexto tiene sus orígenes en viejos proyectos de la década de los 60, como el Proyecto Xanadu[9] de Ted Nelson y el sistema de comunicación en línea NLS de Douglas Engelbart. Los dos, Nelson y Engelbart, estaban a su vez inspirados por el ya citado sistema basado en microfilm "memex", de Vannevar Bush.

El gran avance de Tim Berners-Lee fue unir hipertexto e Internet. En el proceso, desarrolló un sistema de identificadores únicos globales para los recursos web y también: el Uniform Resource Identifier[10].

El sistema WorldWideWeb tenía algunas diferencias con los otros sistemas de hipertexto que estaban disponibles (Hypercard, Storyspace, Intermedia, SuperCard) en aquel momento:

- El modelo de obtención de información requería enlaces unidireccionales en vez de los bidireccionales. Esto hacía posible que un usuario pueda enlazar a otro recurso sin necesidad de ninguna acción del propietario de ese recurso. Con ello se reducía significativamente la dificultad de implementar servidores web y navegadores, pero en cambio presentaba el problema crónico de los enlaces rotos.
- A diferencia de sus predecesores, como HyperCard11, WorldWideWeb era no-propietario, haciendo posible desarrollar

[9] El Proyecto Xanadu fue el primer proyecto de manejo de hipertexto desarrollado en 1960 por Ted Nelson.

[10] ***U****niform* ***R****esource* ***I****dentifier*, identificador uniforme de recurso, es una cadena corta de caracteres que identifica inequívocamente un recurso (servicio, página, documento, dirección de correo electrónico, enciclopedia, etc.). Normalmente estos recursos son accesibles en una red o sistema.

servidores y clientes independientemente y además añadir extensiones sin restricciones de licencia.

Entre 1991 e inicios de 1993 se realizan diferentes actividades enfocadas a la difusión del WorldWideWeb, siendo finalmente el 30 de abril de 1993 el día en que el CERN anuncia mediante un comunicado[12] que la tecnología WorldWideWeb sería de uso público.

Para junio de 1993 se identificaron alrededor de 130[13] servidores web; se había desarrollado el primer navegador con interface gráfica, el VIOLAWWW, un programa desarrollado en la Universidad de Berkley a partir del navegador de hipertexto WorldWideWeb, funcionaba bajo plataformas NeXT y sirvió a posteriori de base para el desarrollo del popular navegador MOSAIC.

---

[11] Es un ambiente de programación de hipertexto para la Macintosh introducido por Apple en 1987. El modelo del ***HyperCard*** consiste de tarjetas, y conjuntos de tarjetas, llamados pilas.

[12] Revisar el documento emitido en los ANEXOS.

[13] Entre 1993 y 1995 Matthew Gray (en la actualidad Ingeniero de Software de Google) realizó un informe que denominó “Midiendo el crecimiento de la Web”, para llegar a obtener esta información utilizó como herramienta principal el primer agente automata web o spider - World Wide Web Wanderer

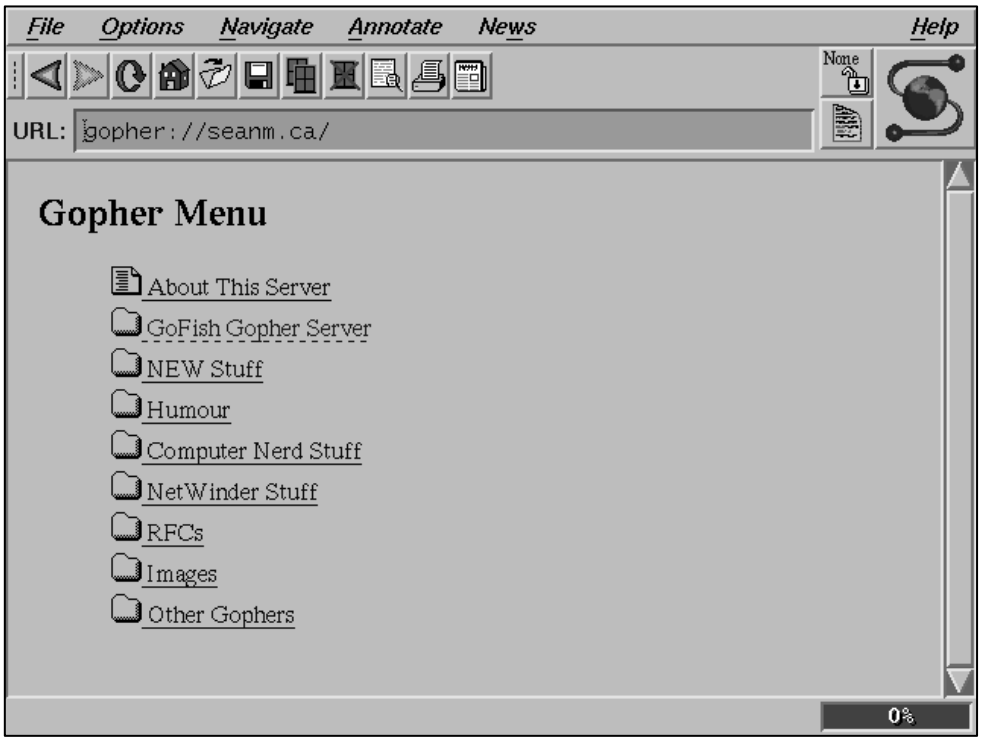

Fig. 2 – Interfaz de usuario del navegador Mosaic.

Fuente: http://www.w3c.es

Los investigadores concuerdan en que el punto de inflexión de la World Wide Web comenzó con la introducción del navegador web Mosaic. El navegador gráfico fue desarrollado por un equipo del NCSA[14] en la Universidad de Illinois en Urbana-Champaign (NCSA-UIUC), el proyecto fue dirigido por Marc Andreessen y financiado por la Iniciativa de Computación de Alto Desempeño y Comunicación un programa de fondos iniciado por el entonces gobernador Al Gore. Antes del lanzamiento de Mosaic, las páginas web no integraban un amplio entorno gráfico y su popularidad fue menor que otros protocolos anteriores ya en uso sobre Internet, como el protocolo Gopher y WAIS. La interfaz gráfica de usuario de Mosaic permitió a la web

[14] Organismo norteamericano relacionado con la investigación en el campo de la Informática y las Telecomunicaciones, con sede en la Universidad de Illinois. Jugó un papel muy importante en el desarrollo del World Wide Web dado que introdujo el navegador Mosaic.

convertirse en el protocolo de Internet más popular de una manera fulgurante.

En septiembre ya había versiones de Mosaic para PC y Macintosh. El tráfico de la WWW[15] alcanzaba el 1% de todo el tráfico de Internet y había 204 servidores. Es el comienzo del crecimiento explosivo de la web. A finales del 94 ya había más de 10.000 servidores y 10 millones de usuarios. En 1997, más de 650.000 servidores.

A partir de ese momento las páginas web evolucionaron rápidamente y de la misma manera se desarrollaron estándares, y alrededor de éstos tecnologías que han permitido no sólo mejorar la experiencia del usuario en la web sino también mejorar la forma en que la información es transmitida y almacenada en las bases de datos de los servidores. Hasta tener la capacidad de incluir dentro de los contenidos que presentan, estructuras de texto más complejas y gráficos, con el uso de programas como plugins para reproducir archivos de audio y video o para mostrar contenido multimedia en vivo.

Los desarrolladores web complementaron el funcionamiento básico de los navegadores que pasaron de renderizar código HTML a invocar código script[16] que se guarda en las computadoras de los usuarios (clientes). Estos

---

[15] Ya se había acuñado el término World Wide Web como estándar para el desarrollo de la técnología de transmisión de información en la red Internet.

[16] Un script (cuya traducción literal es 'guion') es una secuencia de instrucciones de programa usualmente simple, que por lo regular se almacena en un archivo de texto plano. Los script son casi siempre interpretados, pero no todo programa interpretado es considerado un script. El uso habitual de los scripts es realizar diversas tareas como combinar componentes, interactuar con el sistema operativo o con el usuario. Por este uso es frecuente que los shells sean a la vez intérpretes de este tipo de programas.

scripts pueden crear elementos de interface con diferentes efectos como rollovers[17], menús personalizados, y otras ayudas de navegación. También pueden ejecutar diferentes métodos de interfaz de usuario, por ejemplo, validar la entrada de datos que se ingresan en un formulario HTML.

Esta capacidad de los scripts, mientras mejora la interacción del usuario con las páginas web individuales, no cambia el modelo básico de la lógica de la aplicación que se ejecuta en el servidor entre el sitio web y el servidor cada vez que el usuario hace clic. Este comportamiento se denomina sincrónico, lo que significa que, después de cada clic el usuario espera mientras el servidor se encarga de procesar los datos de entrada y envía al navegador una página de respuesta para su descarga. Por ejemplo en el comercio electrónico, una interacción típica del usuario con un sitio web involucra una serie de páginas web que representan pasos de un proceso más amplio que componen una aplicación web, se sigue utilizando el enfoque sincrónico tradicional. Esto sucede porque la información que se solicita al servidor depende de las decisiones que el usuario tome al realizar una transacción.

1.2 De páginas web tradicionales a Aplicaciones Enriquecidas de Internet (RIA - Rich Internet Application)

Los primeros intentos de pasar a desarrollar páginas web tradicionales a Aplicaciones Enriquecidas de Internet (RIA) se dan a finales del año 2005. Los desarrolladores web buscaban la manera de realizar aplicaciones web que se parecieran más a las aplicaciones tradicionales de escritorio, para

[17] Efecto producido por el cambio de apariencia de un elemento gráfico cuando el apuntador del ratón se desplaza sobre él.

lograr esto el modelo de aplicaciones RIA transfiere la ejecución de algunos de los procesos que antes se daban en el servidor hacia el cliente web.

A medida que se desarrollan nuevos estándares se crean también nuevas tecnologías que permiten crear productos diferentes a nivel de software. La tendencia ha sido crear aplicaciones que se ejecuten en diferentes plataformas y en la actualidad también en diferentes dispositivos.

Muchos de los sitios web que en sus inicios eran utilizados para presentar únicamente información a modo documental hoy tienen diversas formas de suministrar información a los usuarios, tras elevar los niveles de interacción con un sitio web se ha mejorado no solo la experiencia de usuario sino también las posibilidades de crear aplicaciones web.

En una aplicación web clásica la mayoría de acciones del usuario en la interfaz de la misma en el navegador desencadena una petición HTTP a un servidor web. Esta solicitud usualmente inicia en el servidor un procesamiento de información: recuperando datos, realizando cálculos o comunicando a la aplicación con diversos sistemas legacy[18]. Este proceso, a pesar de ser corto, toma un tiempo en ejecutarse. El usuario debe esperar a que el proceso completo sea realizado.

Es entonces cuando el servidor genera nuevamente una página web entera basada en el nuevo estado provisto por toda la información recuperada.

[18] Un sistema heredado (o sistema legacy) es un sistema informático (equipos informáticos o aplicaciones) que ha quedado anticuado pero continúa siendo utilizado por el usuario (típicamente una organización o empresa) y no se quiere o no se puede reemplazar o actualizar de forma sencilla.

Finalmente esta página total es retornada al navegador que es el encargado de renderizar completamente el contenido nuevamente como lo hizo la primera vez. Este comportamiento de las aplicaciones clásicas genera malestar en el usuario porque:

- No está pensado para crear una experiencia de usuario favorable.
- Mientras el servidor realiza una tarea el usuario debe esperar a que concluya la misma.
- En cada paso que el usuario realiza con una actividad permitida en la aplicación, el usuario debe esperar más tiempo.

Uno de los avances más notables tiene una estrecha relación con el modelo del uso del usuarios: inicialmente el paradigma de interacción se caracterizaba por el limitado número de elementos con los que se contaba (básicamente enlaces y elementos de formularios) y donde una página de respuesta constituía la unidad mínima de información, con lo que cualquier petición al servidor causada por una interacción por parte del usuario implicaba una recarga completa de toda la aplicación.

En la actualidad existen una cantidad considerable de tecnologías, propietarias y no propietarias, que son capaces de introducir en la web el paradigma de la aplicación de escritorio consistente en crear aplicaciones web con dos características fundamentales:

1. Una alta interactividad con el usuario, a través multitud de elementos de interacción que antes sólo eran viables en entornos de escritorio (como menús, árboles, deslizadores, etc.), programables bajo cualquier evento de usuario (como clic de ratón, pulsación de tecla,

drag and drop, etc.) y con un aspecto visual totalmente personalizable.

2. Una alta velocidad de respuesta a la interacción del usuario. La unidad de información mínima es la que desee el desarrollador: una etiqueta, una tabla o quizás toda la página. De este modo, es posible conseguir una respuesta inmediata a las acciones del usuario, descargando sólo los datos absolutamente necesarios.

Como en la mayoría de los avances informáticos, varias tecnologías compiten por ser la forma estándar de facto para crear aplicaciones RIA. Los principales contendientes son el conjunto de tecnologías de Adobe (Flex), Java, Silverlight, y la colección de tecnologías Web conocida como AJAX[19].

Para empezar a tener una idea de cómo la tecnología hace posible crear aplicaciones RIA, a continuación una lista de características que AJAX como colección de tecnologías incorpora:

- Presentación basada en estándares usando XHTML y CSS.
- Visualización e interacción dinámicas usando el Document Object Model.
- Intercambio de datos y la manipulación usando XML y XSLT.
- Recuperación de datos asincrónica usando XMLHttpRequest.
- Javascript como tecnología vinculadora.

[19] AJAX es un término acuñado en 2005 por Jesse James Garrett, quien es autor del libro "The Elements of User Experience". Garret definió a AJAX como la interacción de muchas tecnologías, cada una floreciendo por su propio mérito e interactuando de poderosas nuevas formas.

Cada una de las tecnologías que permiten desarrollar aplicaciones RIA introducen un elemento intermedio entre el usuario y el servidor. En lugar de cargar una aplicación web completa representada en una página web, al inicio de la sesión de la aplicación, el navegador carga un conjunto de librerías o *scripts* colocadas por lo general en un frame o contenedor oculto.

Este conjunto de librerías es responsable de renderizar la interfaz que el usuario observa cuando la aplicación se comunica con el servidor. Además, le permitir interactuar con la aplicación de manera asíncrona a lo que sucede con el servidor produciendo los siguientes beneficios:

- Elimina la naturaleza start-stop-start-stop de la interacción en la web.
- El usuario no está en una constante espera a que la ventana del navegador, con un reloj de arena, obtenga una respuesta del servidor para que la aplicación haga algo.
- La aplicación responde de mejor manera.

Para entender la problemática de accesibilidad que solucionan las RIA, es conveniente analizar las tecnologías que se utilizan y la forma en la que se desarrollan.

En general, una aplicación RIA puede crearse siguiendo dos modelos:

1. Mediante el uso de tecnologías estandarizadas, como (X)HTML, CSS o JavaScript.
2. Mediante el uso de tecnologías incrustadas, haciendo uso de unas determinadas marcas (X)HTML que permiten ejecutar una aplicación externa en el agente de usuario. Las tecnologías más habituales que

siguen este modelo son las tecnologías propietarias Adobe Flash y Microsoft Silverlight.

1.3 Modelo de comportamiento RIA

Un modelo de referencia nos ayuda a establecer un marco común de referencia o marco conceptual bajo el cual se desarrolla una aplicación. RIA introduce un nuevo modelo de programación de aplicaciones, combina las ventajas de los dos modelos que habían predominado hasta el momento de su aparición: el de las aplicaciones cliente-servidor y el modelo multi-capa utilizado por las aplicaciones web. El principal objetivo de éste modelo es mejorar la experiencia del usuario.

Con las aplicaciones RIA, los usuarios reciben respuestas instantáneas sin esperar a las conexiones de ida y vuelta contra el servidor que requerían las aplicaciones web tradicionales. Pero además, en muchos de los casos, las RIA pueden funcionar en cualquiera de los sistemas operativos que tenga instalado el usuario en su equipo (son multiplataforma) y utilizan el protocolo de comunicación de Internet, TCP/IP.

Se espera un gran desarrollo de este tipo de aplicaciones en un futuro próximo de cara al gran público y al ámbito interno de las organizaciones. Las principales ventajas que introducen estas aplicaciones son las siguientes:

- Agilidad en la respuesta.
- Cálculos rápidos, controles prediseñados y funciones gráficas, interactivas y multimedia avanzadas.

- En muchos casos no requieren de instalación en el equipo del usuario (es suficiente con disponer de un navegador web), por lo que no es necesario pensar en distribuciones de software.
- Uso desde cualquier ordenador con acceso a Internet.

Pero también existen ciertos retos con los que las tecnologías RIA deberán lidiar en el futuro:

- Las RIA introducen cambios en los hábitos de navegación y en el uso de las aplicaciones web, y el usuario tardará un tiempo en digerirlos. Además, se dan ciertas complicaciones para el cumplimiento de los niveles de accesibilidad.
- Algunas de las tecnologías RIA que hacen uso del navegador web deberán superar algunos aspectos no resueltos aún, como la posibilidad de introducir "Favoritos" o la de utilizar el botón "Atrás" del navegador web.
- Las RIA deberán considerar la optimización de los motores de búsqueda o la capacidad de los sistemas de análisis para monitorizar sitios web construidos con esta tecnología.
- La incidencia de estas aplicaciones sobre aspectos relacionados con la seguridad deberá estudiarse en su globalidad a la hora de definir la arquitectura de sistemas y aplicaciones de la organización.

1.4 Interacción navegador/servidor

Tomemos como punto de partida a un navegador web, que en su forma más simple representa el motor cliente donde se realizan las actividades en una aplicación. Bajo este concepto básico el usuario interactúa con una

aplicación web por medio de enlaces o botones. Cuando el usuario hace click en estas zonas de interacción de una aplicación web, éstos hacen las veces de disparadores de eventos o en este caso de llamadas a uno o varios servidores.

El servidor responde a las peticiones del cliente de manera que cuando el navegador web haya recibido contenido suficiente para ser desplegado (o almacenado en su memoria caché), el navegador muestra ésta información al usuario. La experiencia del usuario al interactuar con la aplicación depende en este caso del tiempo de respuesta que la aplicación se tomó desde que el usuario hizo click en un determinado elemento de la aplicación hasta que se desplegó la información solicitada.

Incluso la descarga de una sola página web implica muchas idas y vueltas entre el cliente (navegador web) y el servidor, ya que la mayoría de las páginas web están conformadas por un conjunto de elementos que son parte de su contenido tales como: hojas de estilo CSS, archivos scripts e imágenes incrustadas; mismos que son descargadas de forma individual y casi siempre secuencial por el navegador.

En una aplicación web tradicional asíncrona este proceso sucede en repetidas ocaciones. Esto debido a que las aplicaciones web requieren intercambiar información entre el cliente y el servidor por lo que de manera casi general por lo menos una de las peticiones hacia el servidor se realiza mediante el método HTTP POST, que en comparación con el mucho más común método HTTP GET eleva el nivel de seguridad, para cargar algunos datos que un usuario ha solicitado al envíar la petición mediante un formulario.

Por ejemplo, si un usuario visita el sitio de *amazon.com* y es un usuario recurrente necesitará en primera instancia de un registro de sus datos aún si la aplicación web pudiera haber reconocido algún archivo cookie en el cliente. Al momento que el usuario se registra y es reconocida su identidad, la aplicación web ya cuenta con información suficiente para completar cualquier transacción en caso de que exista el deseo del usuario de realizar una compra.

La idea de estructurar una aplicación mediante el modelo RIA es que cuando un usuario se registra como usuario de una aplicación web, ésta cargue en primera instancia la mayor cantidad de información posible con la finalidad que se reduzcan de manera significativa las peticiones del cliente al servidor y por ende se establezca una mejora en el tiempo de respuesta de la aplicación, lo que a posteriori generará mayor satisfacción del usuario al utilizar una aplicación web.

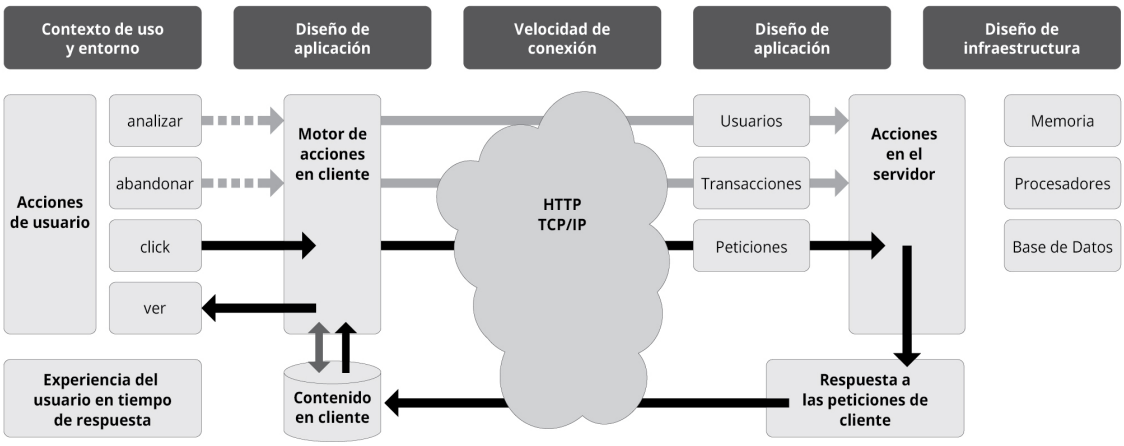

Fig. 3 – Esquema de interacción Navegador/Servidor.

Fuente: http://www.keynote.com/

El esquema de la Figura 3 representa la forma en que responde un servidor a las peticiones de un cliente. Un servidor debe responder a las peticiones

de varios usuarios al mismo tiempo. No importa cuán poderoso es el servidor, cada usuario concurrente consume una pequeña parte de los recursos del servidor: la memoria, el procesador y la base de datos.

Los servidores web pueden responder rápidamente a peticiones de información de usuarios concurrentes pues maneja cada una de las peticiones como transacciones independientes unas de otras, haciendo que la labor de consulta a un catálogo de datos sea una actividad relativamente rápida y eficiente. Sin embargo, la acción de un usuario que requiere del servidor actualizar cierto elemento o tipo de dato de la aplicación (por ej. agregar un elemento a un carrito de compras al realizar una acción) consume más recursos del servidor. Así que el número de transacciones concurrentes interacciones-servidor que actualizan la información almacenada en el cliente desempeña un papel fundamental en la determinación del rendimiento del servidor y por ende en el rendimiento de la aplicación.

En el esquema anterior de la Figura 3, las flechas de color gris y las cajas etiquetadas como Usuarios y Transacciones indican que el rendimiento del servidor está fuertemente influenciado por estos factores de simultaneidad. Los servidores web por lo general tienen un funcionamiento uniforme hasta un cierto nivel de concurrencia o usuarios simultáneos, pero una vez que sobrepasa este nivel el rendimiento de las transacciones degrada rápidamente cada uno de los recursos subyacentes generando cuellos de botella.

Como resultado de esto, cambios pequeños en el comportamiento de una aplicación o infraestructura de servicio de una aplicación pueden tener un efecto significativo sobre el tiempo de respuesta y afectar no sólo a la

experiencia del usuario sino también a la duración de transacciones que se realicen en el servidor.

La gente que diseña y realiza pruebas a los sistemas de back-end sabe que ciertas variables de comportamiento como el tiempo que un usuario se toma para pensar en ejecutar una acción o las tasas de abandono por página tienen una influencia significativa en la capacidad y sensibilidad de carga de los servidores. Ahora con las RIAs (como se indica en la Figura 3 por las líneas cortadas) se da a los diseñadores y arquitectos de aplicaciones la flexibilidad para diseñar soluciones que tomen en cuenta estas variables de comportamiento para mediante los resultados de análisis que se obtengan poder crear aplicaciones que aprovechen los recursos con los que se cuentan de mejor manera.

1.5 El motor del lado del cliente

A pesar de que la forma de implementar las aplicaciones RIA difieren o varían dependiendo de la tecnología que se utilice para su desarrollo, el modelo general introduce una capa de lógica intermedia —un motor del lado del cliente— entre el usuario y el servidor web. Dicha capa, por identificarle de alguna manera, es descargada al inicio de la sesión con la aplicación. El motor se encarga de manejar y desplegar los cambios que surjan de la comunicación con el servidor.

La implementación de esta capa permite a los desarrolladores de software crear aplicaciones web con características que el Grupo Gartner[20] ha

[20] Gartner Inc. es una empresa consultora y de investigación de las tecnologías de la información con

calificado de "soluciones entre el modelo cliente/servidor pesadas pero ricas y las ligera pero pobre modelo basado en la interfaz de usuario web."

Añadir un motor del lado del cliente no evita que se implemente el estilo de comunicación tradicional síncrono. Sin embargo, permite que la interacción del usuario con la aplicación ocurra de manera asíncrona y sea independiente de la comunicación con el servidor.

La Figura 4 (ver página siguiente) ilustra la manera en que el modelo asincrónico de comunicación de usuario y servidor implementado por RIA difiere del comportamiento síncrono de una aplicación Web tradicional. En una Aplicación Rica de Internet:

- la información se puede obtener de un servidor anticipándose al ingreso de información *(input)* del usuario;
- en respuesta a una entrada *(input)*, la pantalla puede actualizarse paulatinamente en lugar de actualizar todo su contenido a la vez;
- es posible validar múltiples entradas de usuario pueden y acumular dicha información en el cliente antes de ser enviada al servidor;
- la respuesta a algunas entradas del usuario se pueden generar sin necesidad de establecer una comunicación con el servidor;

---

sede en Stamford, Connecticut, Estados Unidos. Hasta 2001 era conocida como Gartner Group. La empresa incluye entre sus clientes a algunas de las más grandes empresas, agencias de gobierno, empresas tecnológicas y agencias de inversión como BT, CV, The Wall Street Journal, etc. se concentra en la investigación, programas ejecutivos, consultas y eventos. Fue fundada en 1979; y en 2010 contaba con 4800 empleados, incluyendo a 1250 analistas y clientes en 85 países por todo el mundo.

- existe la posibilidad de descargar en el escritorio del cliente información procesada previamente así como datos modificados en el servidor.

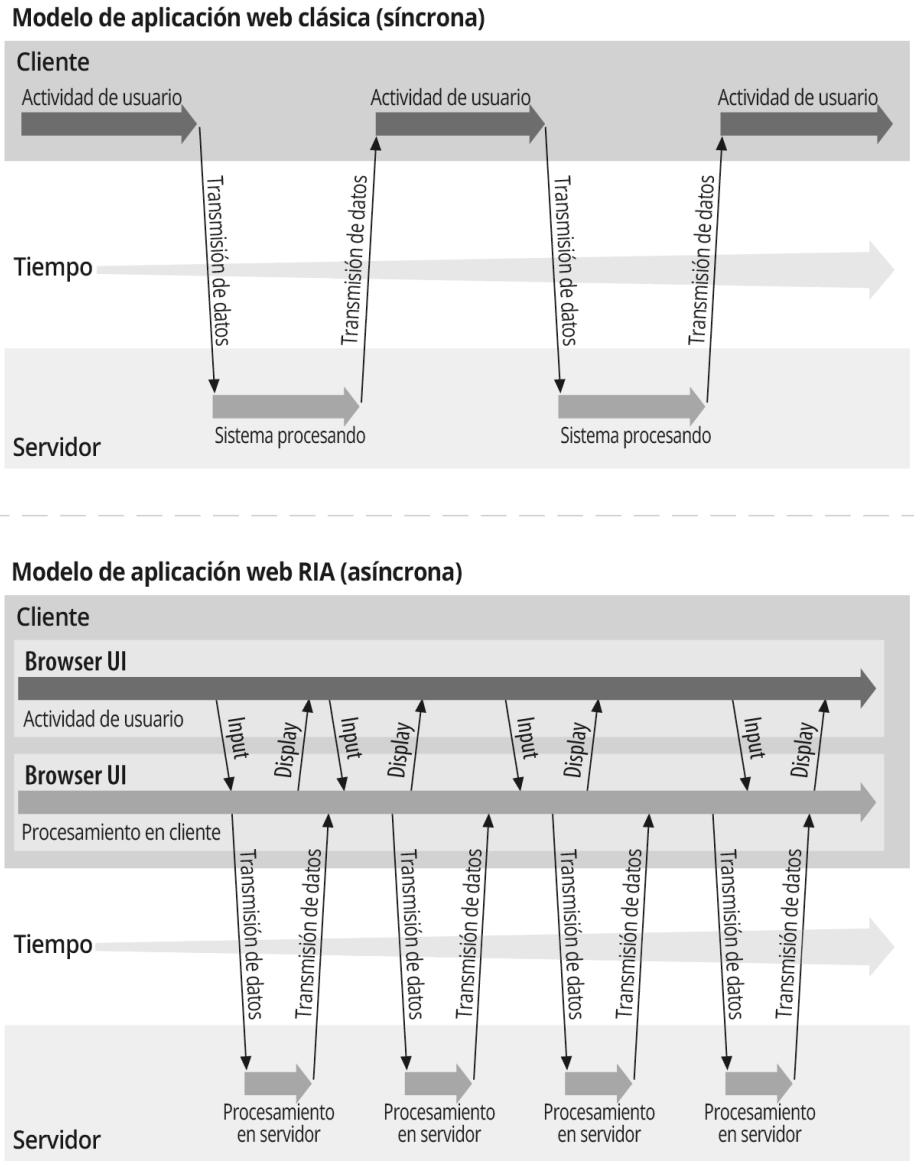

Fig. 4 – Comparación entre el funcionamiento de los modelos tradicional y RIA.

Fuente: http://www.w3.org/

Estas características hacen posible que los arquitectos, diseñadores y desarrolladores de software tengan a mano la posibilidad de crear

aplicaciones que respondan mejor a las necesidades de los usuarios, soportando estas posibilidades en las características que brinda la arquitectura RIA.

En la práctica al momento de desarrollar una aplicación bajo el modelo RIA los usuarios experimentan un tiempo de espera menor en las respuestas del servidor, sin embargo, el empleo de estas técnicas conduce inevitablemente a un diseño más complejo que una aplicación web sincrónico tradicional.

El reto que se plantea a los equipos de desarrollo es asegurar que el resultado final desemboque realmente en una mejor experiencia de usuario.

1.6 La importancia del contexto

La experiencia de muchos desarrolladores de software a pesar del gran número de afirmaciones optimistas que se han realizado en torno a cada una de las tecnologías en las que se puede aplicar este modelo ha determinado que no hay garantía alguna de que utilizando únicamente el modelo RIA se tengan aplicaciones que determinan el éxito total en la experiencia de usuario, la respuesta de las aplicaciones desarrolladas bajo RIA dependerá de varios factores.

Los tiempos de respuesta que tienen insidencia directa en la experiencia positiva o negativa de un usuario dependen de una combinación de comportamientos de los componentes en el cliente y el servidor, lo que depende en su momento inicial del diseño de la aplicación, el delineamiento de los requerimientos de infraestructura del servidor, y por supuesto de la velocidad de conexión a internet del usuario.

La aplicación RIA más efectiva será aquella en la que se tomen en cuenta estos aspectos al momento de desarrollar su ciclo de vida y además cuente con los procesos administrativos necesarios para asegurar el éxito cuando se encuentre en producción.

Además, dentro del contexto creativo del desarrollo de una aplicación RIA hay un punto importante a tomar en cuenta, y este es el diseño del look and feel[21] pues el diseño correcto adaptado a las necesidades del usuario permitirá crear los componentes correctos y aclarar de esta manera el contexto visual dentro del cual un usuario se desenvolverá con la aplicación.

Mediante un diseño de look and feel adecuado pueden establecerse divisiones dentro de la pantalla que permitirán a la aplicación comportarse de una u otra manera. Dichas divisiones se vuelven importantes pues permiten delinear los segmentos de página que los navegadores modernos pueden actualizar dinámicamente sin necesidad de recargar una página completa. Una sentencia importante a recordar es que en la web, la experiencia del usuario es fundamental.

[21] El término *Look and Feel* está ligado a la interfaz de usuario (UI) y estrechamente a la apariencia de la misma.

CAPÍTULO II

2 Arquitectura de las Aplicaciones Enriquecidas de Internet - RIA

Al desarrollar una aplicación RIA se concentra la capa de lógica de la presentación en la parte del cliente, convirtiéndolo en un "cliente pesado". Es decir, es en el cliente donde se carga parte de la aplicación que tiene que ver con la interactividad y el look and feel, y no sólo con documentos HTML como se maneja en el modelo tradicional web.

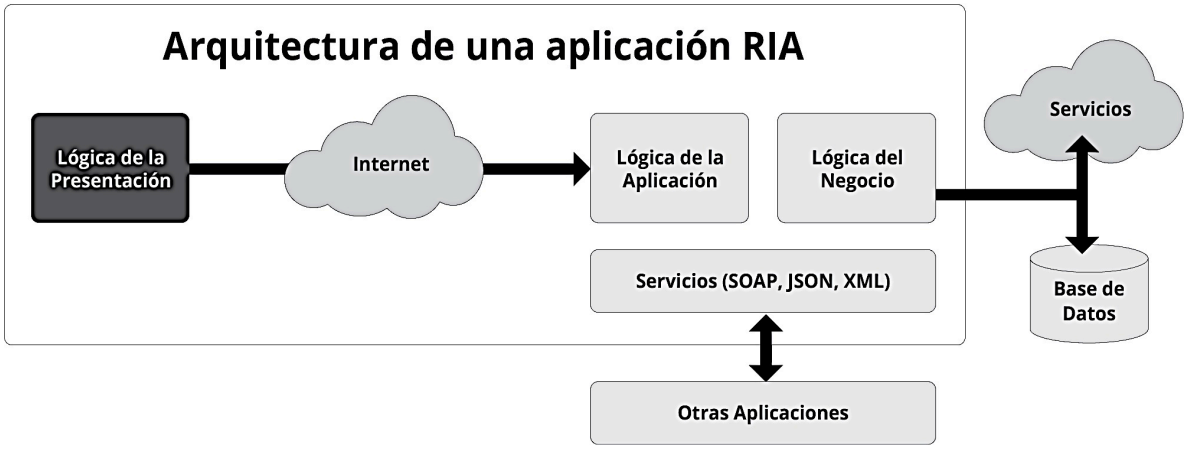

Fig. 5 – Esquema de arquitectura base de una aplicación RIA.
Fuente: http://seminariocomp.wordpress.com/

Como consecuencia de esta carga de información en el cliente, surgen algunas situaciones complejas en la comunicación entre la lógica de presentación y la lógica de negocio; cosa que no se da en las aplicaciones web tradicionales, pues la lógica de presentación y de negocio suelen estar dentro de la misma capa física en ese modelo, por lo que la comunicación entre ambas no supone ninguna complejidad.

Si entre estas dos capas tomamos en cuenta el nexo que supone el internet, la comunicación ya no se realiza de manera directa y la solución en este momento es desarrollar una capa de servicios en el servidor que el cliente pueda consumir. Esto implica que el desarrollo pueda aumentar los tiempos de ejecución y por ende se suciten posibles problemas en la creación de la capa de servicios, al exponer los métodos que necesita la aplicación, al implementar proxys, validaciones, autenticaciones; es decir, al tomar las medidas de seguridad pertinentes.

En la actualidad existen numerosas soluciones o frameworks[22] disponibles en el mercado que nos permiten desarrollar aplicaciones RIA. Cada una de las cuales establecen sus respectivas arquitecturas y plantean diferentes soluciones al desarrollo de estas aplicaciones. Algunas de las cuales son las siguientes:

- Adobe Flex
- ASP.NET (AJAX - ASPX based)
- Curl
- Open Laszlo (AJAX based)
- Nexaweb's Enterprise Web 2.0 (AJAX based)
- AltioLive (Java based)
- UltraLightClient (Java based)

[22] En el desarrollo de software, un framework o infraestructura digital, es una estructura conceptual y tecnológica de soporte definido, normalmente con artefactos o módulos de software concretos, que puede servir de base para la organización y desarrollo de software. Típicamente, puede incluir soporte de programas, bibliotecas, y un lenguaje interpretado, entre otras herramientas, para así ayudar a desarrollar y unir los diferentes componentes de un proyecto.

Representa una arquitectura de software que modela las relaciones generales de las entidades del dominio, y provee una estructura y una especial metodología de trabajo, la cual extiende o utiliza las aplicaciones del dominio.

- JavaFX - Beta (Java based)
- DOJO (AJAX based)
- JQUERY (AJAX based)
- Google web toolkit (AJAX based)
- Silverlight – beta (.NET based)

Entre las soluciones RIA mencionadas anteriormente, vamos a reducir el campo de comparación entre:

- Adobe Flex
- ASP. NET AJAX
- Curl

Dado que las empresas esperan un excelente soporte posventa del proveedor de la solución, el criterio de selección de los frameworks arriba mencionados se basa totalmente en la disponibilidad de servicios de apoyo a su alrededor. Por lo tanto, las soluciones en las versiones beta y de código abierto sin apoyo organizado han sido excluídos de este análisis.

2.1 Adobe Flex

La línea de productos de Adobe Flex es una solución desarrollada por la empresa Adobe la cual permite desarrollar soluciones de aplicaciones RIA tanto en entornos empresariales como alrededor de la web. La línea de productos Adobe Flex provee un modelo de programación denominado *Adobe Flex Framework*, integrado a herramientas de desarrollo basadas en Eclipse denominado *Adobe Flex Builder*™, y la integración de datos y servicios mediante *Adobe Flex Data Services* mismos que permiten a una

organización desarrollar soluciones completas por integrando aplicaciones existentes y sitios web.

2.1.1 *Adobe Flex Framework*

Las aplicaciones desarrolladas con Adobe Flex se ejecutan en el navegador con la ayuda de un complemento denominado Flash Player.

Este complemento[23] ofrece una multi-plataforma de tiempo de ejecución consistente que contiene una máquina virtual con soporte integrado para la visualización de texto en varios idiomas, impresión, manipulación de datos, animación y soporte multimedia en general.

Flex proporciona los componentes de servicio del lado cliente que permiten a las aplicaciones interactuar con cualquier servidor remoto a través de servicios web SOAP, servicios REST, HTTP o protocolos personalizados basados en sockets.

2.1.2 *Adobe Flex Builder*

El modelo de desarrollo Adobe Flex hace uso del estándar MXML para el diseño y maquetación de interfaces de usuario y ActionScript (una implementación de ECMAScript) para la lógica que se ejecuta

[23] Un complemento es una aplicación que se relaciona con otra para aportarle una función nueva y generalmente muy específica. Esta aplicación adicional es ejecutada por la aplicación principal e interactúan por medio de la API. También se conoce como plug-in (del inglés «[un] enchufable o inserción»), add-on («añadido»), conector o extensión.

en el lado del cliente. El ambiente de desarrollo integrado (IDE[24]) de Flex Builder provee las herramientas para la codificación, depuración y diseño visual y maquetación de la interfaz de usuario que se puede integrar con los sistemas de gestión de código fuente existentes. Además de esto, Flex proporciona soporte integrado de la herramienta para una unidad de prueba.

2.1.3 *Adobe Flex Data Services*

Adobe Flex Data Services amplía las capacidades del framework cliente de Flex, proporcionando servicios adicionales para la gestión de la transferencia de datos y la integración con las aplicaciones y la infraestructura existentes.

Adobe Flex Data Services se implementa como una aplicación Web Java y puede ser desplegado en servidores de aplicaciones Java estándar como Jboss o Tomcat. Los servicios prestados por los servicios de datos Flex se integran con los perfiles de seguridad existentes definidas en el servidor de aplicaciones Java.

Flex Data Services se puede poner en marcha utilizando herramientas de implementación estándar proporcionados con el servidor y se puede integrar a las características de clúster de las aplicaciones en el servidor. Además, las aplicaciones creadas con Flex Data Services pueden acceder a los datos de sesión existentes

[24] Un ambiente interactivo de desarrollo (Interactive Development Environment - IDE) es una aplicación de usuario que provee de facilidades a los programadores en el desarrollo de software. *Fuente: http://en.wikipedia.org/wiki/Integrated_development_environment*

en el servidor junto a la lógica de aplicaciones utilizando las API estándar de Java.

2.2 AJAX sobre ASP.NET

El conjunto de tecnologías denominado AJAX (Asynchronous JavaScript and XML) está conformado por diferentes estándares establecidos por la W3C: XHTML, CSS, DOM, JavaScript, XML y objetos XMLHttpRequest; que se utilizan en conjunto para crear aplicaciones web interactivas.

Hay varios tipos de frameworks AJAX, que van desde simples (que requieren una amplia experiencia en HTML / CSS / AJAX) hasta los que están basados en componentes que son más complejos y ofrecen una mayor capacidad resolutiva en tiempo empleado al desarrollo de aplicaciones. El framework de Microsoft ASP.NET AJAX fue seleccionado para el estudio, ya que admite la programación visual y proporciona componentes incorporados, permitiendo de esta manera el desarrollo rápido de aplicaciones.

Microsoft ASP.NET AJAX (conocido antes como el framework *ATLAS*) permite a los desarrolladores crear páginas web con elementos de interfaz de usuario (UI) con los que se encuentra familiarizado. También proporciona bibliotecas de scripts del lado del cliente que incorporan las tecnologías ECMAScript (JavaScript) y HTML dinámico (DHTML), integrando todo bajo la plataforma de desarrollo basada en el lenguaje de servidor ASP.NET.

Las librerías de scripts del cliente de ASP .NET AJAX están conformadas por archivos escritos en lenguaje Javascript (.js) que proporcionan

características de programación orientada a objetos de alto nivel en consistencia y modularidad. Este conjunto de librerías incluye las siguientes capas:

- Capa de compatibilidad de scripts ASP .NET AJAX para navegadores, misma que brinda estabilidad de la aplicación en los navegadores de uso frecuente por los usuarios (incluyendo Microsoft Internet Explorer, Mozilla Firefox, Google Chrome y Safari de Apple).
- Servicios básicos de ASP.NET AJAX, que incluyen extensiones de JavaScript, como clases, espacios de nombres, manejo de eventos, herencia, tipos de datos y serialización de objetos.
- Una librería de clases base ASP .NET AJAX, misma que incluye componentes que permiten la generación de cadenas y el manejo extendido de errores.
- Una capa de red que se encarga de la comunicación entre servicios web y aplicaciones, y que además es responsable de gestionar las llamadas remotas a métodos asincrónicos.
- Soporte para librerías Javascript que se encuentras incrustadas en un paquete o son provistas como archivos Javascript (.js) únicos. La incorporación de librerías Javascript a un paquete puede facilitar el despliegue de la aplicación y ayudar además a resolver problemas que puedan presentarse por versiones anteriores generadas o posteriores que se generen.
- Soporte para el acceso a formularios de autenticación basados en lenguaje de servidor y a la información de perfil ubicada en scripts en el cliente. Este soporte también se encuentra disponible para las aplicaciones web que no se crean con ASP.NET, siempre y cuando la aplicación tenga acceso a la biblioteca de Microsoft AJAX.

- Soporte en modo de lanzamiento y depuración, además de soporte de localización de errores tanto para código Javascript incrustado como escrito en archivos independientes.

El framework ASP.NET AJAX utiliza la interfaz de usuario del Visual Studio.NET de Microsoft mismo que permite a los desarrolladores editar, crear, depurar y probar aplicaciones.

2.3 Curl

Curl es una framework RIA, que proporciona un entorno potente y eficaz para el desarrollo de aplicaciones web de potencia industrial complejo. Al hacer uso de Curl, las organizaciones pueden crear una nueva clase de aplicaciones web basadas en proporcionar la interactividad, funcionalidad y el rendimiento de las aplicaciones cliente-servidor. Permite a las empresas además implementar soluciones SOA innovadoras y desarrollar estrategias Web 2.0. Combina la lógica del programa, gráficos y procesamiento de texto en un solo lenguaje/plataforma. El conjunto de productos Curl incluye tres elementos: el *Lenguaje Curl*, el *Ambiente de Tiempo de Ejecución Curl* y el *Entorno de Desarrollo Integrado Curl*.

2.3.1 *Lenguaje Curl*

Ha sido diseñado para ser usado en la web específicamente, soporta formato de texto enriquecido y maquetación de interfaces de usuario GUI HTML, además ofrece la rápida y fácil presentación de script de Javascript todo esto con la potencia de la programación orientada a objetos de C++, C# y Java. El lenguaje Curl soporta los tres estilos

de programación: declarativo, scripts y la orientación a objetos permitiendo de esta manera a los desarrolladores crear rápidamente prototipos mediante scripts y construir aplicaciones de escala empresarial mantenibles utilizando la programación orientada a objetos.

2.3.2 *Ambiente de Ejecución Curl (RTE)*

La plataforma de ejecución para aplicaciones Curl ofrece una extensa colección de APIs incorporados que reduces el tamaño y la complejidad de las aplicaciones Curl. La RTE Curl proporciona las siguientes características:

- Bibliotecas de gráficos 2D y 3D que permiten el acceso a sofisticadas funciones de aceleración de hardware y capacidad de renderizado; modo de presentación inmediato (opciones para un bajo nivel de renderización) y modo de operación conservado por escena.
- Soporta eventos basados en interacciones.
- Incorpora cronómetros características de animación que producen comportamientos visualización dinámicos.
- Se pueden crear y reproducir datos de audio.
- Acceso a servicios web por medio de HTTP o protocolos de socket basados en TCP/IP.

2.3.3 *Entorno de Desarrollo Integrado Curl*

El Entorno de Desarrollo Integrado Curl ofrece un completo conjunto de herramientas para escribir y depurar aplicaciones Curl. Esto permite un rápido desarrollo de aplicaciones a través de un editor de diseño Visual (EVA) que ofrece un nivel avanzado del tipo WYSIWYG que sumado a la funcionalidad de depuración de aplicaciones, pruebas y perfiles características proporciona una herramienta eficaz para el desarrollo de aplicaciones RIA.

2.4 Seguridad en las aplicaciones RIA

Tomar como estrategia referencial el modelo de tecnología RIA introduce de manera generalizada nuevos retos al momento de desarrollar aplicaciones web. A medida que las empresas toman la decisión de aplicar el modelo RIA, surgen una variedad de inquietudes concernientes a seguridad, optimización en motores de búsqueda (SEO[25]), análisis de datos en la gestión de la aplicación y accesibilidad. De estas inquietudes que surgen, la forma de solventar los problemas relacionados con la seguridad de las aplicaciones encabeza la lista.

Mientras las empresas que proporcionan soluciones de seguridad informática afirman que desarrollar aplicaciones bajo el criterio RIA introduce mayores vulnerabilidades, las empresas proveedoras de frameworks que permiten el desarrollo de RIA afirman que no es así. Sin embargo, es

[25] SEO acrónimo en inglés de Search Engine Optimization, que significa Optimización en Motores de Búsqueda.

importante que los desarrolladores tengan en claro cuáles son los peligros a los que las RIA están expuestas, pues a medida que la forma de realizar aplicaciones cambia también lo hace la forma de realizar ataques en busca de vulnerabilidades. Es así que en esta búsqueda los riesgos de ataques pasan de localizarce en la capa de red a la capa de aplicación.

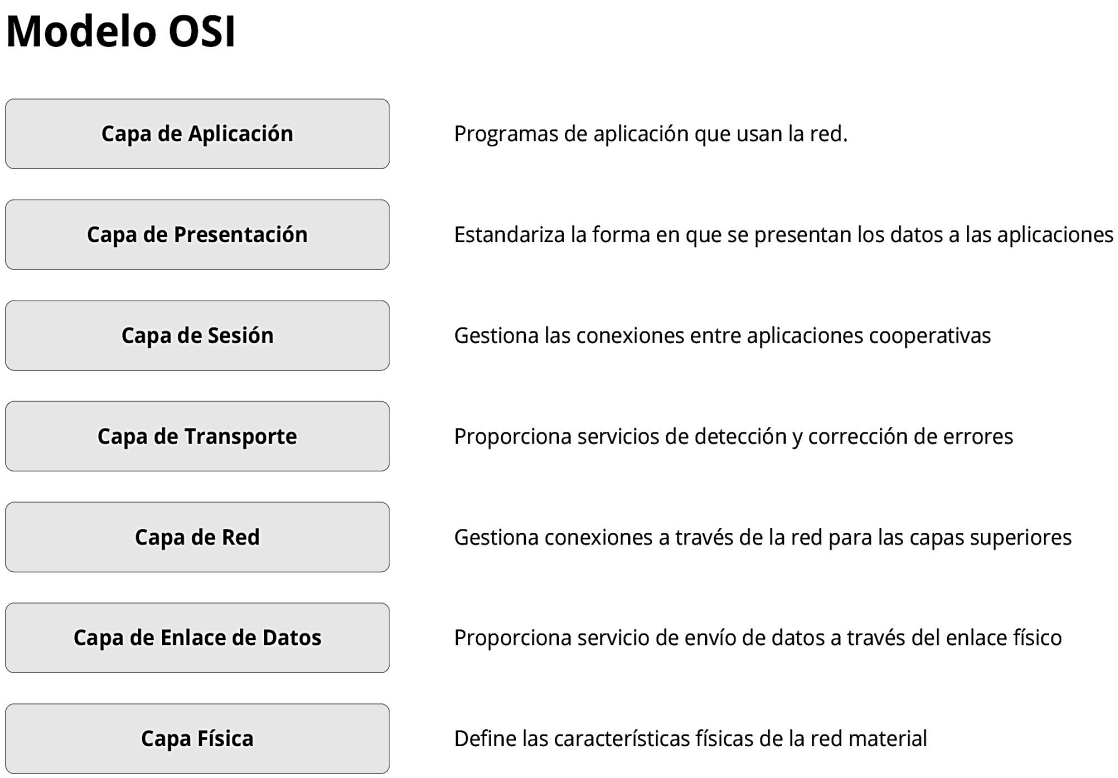

Fig. 6 – Esquema de Modelo Referencial OSI.
Fuente: ISO/IEC 10731:1994 - Information technology Open Systems Interconnection - Basic Reference Model - Conventions for the definition of OSI services.

Los desarrolladores profesionales de aplicaciones se encuentran en la mitad de todo esto; y dado que muchas veces queda muy difícil convertirse en expertos de seguridad, la demanda de los usuarios para RIAs hace que sea imposible para los desarrolladores profesionales de aplicaciones ignorar los riesgos puedan surgir al implementar RIA de manera inadecuada.

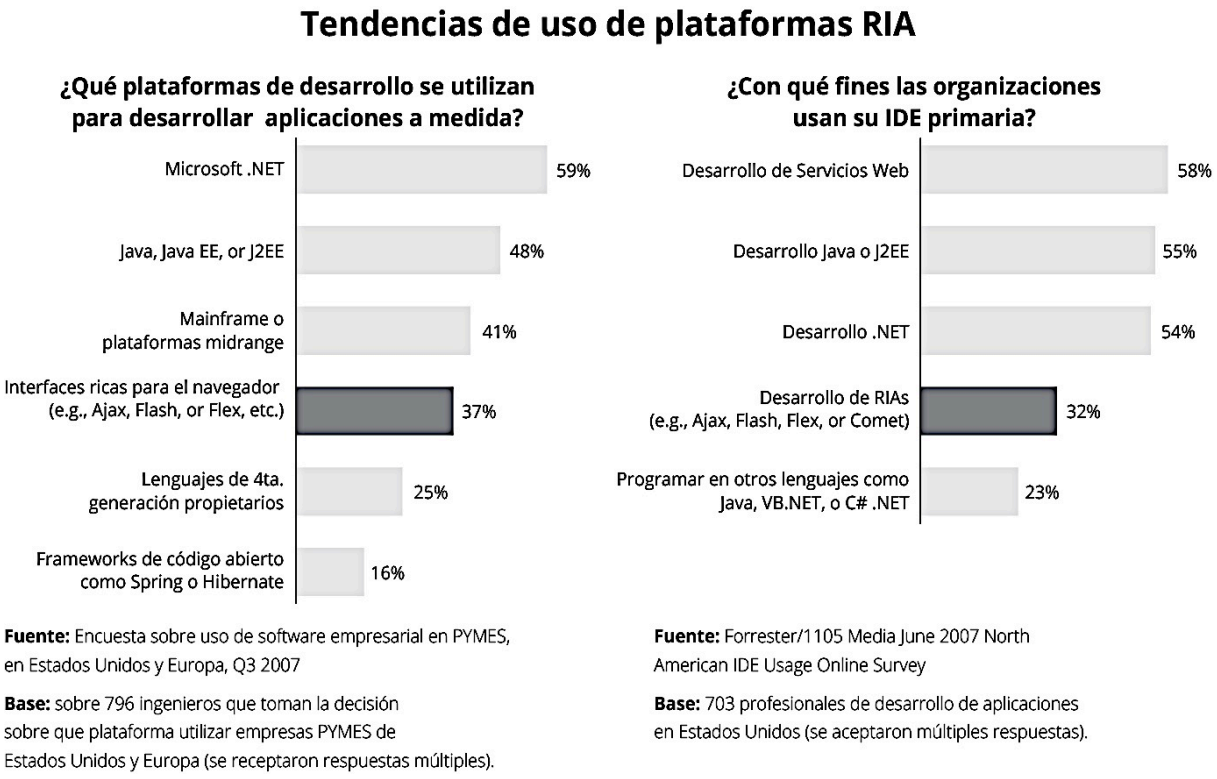

Fig. 7 – Estádisticas de tendencia de uso de RIAs.

Fuente: Forrest Research, Inc.

El primer paso hacia la reducción de los riesgos de seguridad asociados a una RIA consiste en comprender la superficie de ataque de la aplicación y determinar a continuación los pasos que se deben seguir. Definimos al área de ataque de una aplicación como la relación existente entre el código, interfaces, servicios, protocolos y procedimientos disponibles para todos los usuarios, con un fuerte enfoque en lo que está disponible a usuarios con identidades no verificadas.

Para establecer el área de ataque a una aplicación RIA vamos a tratar el problema en tres dimensiones diferentes.

2.4.1 *Ataques del lado del servidor*

Algunas de las técnicas empleadas con mayor frecuencia para realizar ataques a diferentes aplicaciones web por el lado del servidor, según un estudio realizado por Forrest Research[26], son: inyección SQL[27], Cross Site Request Forgery (CSRF[28]) y Cross-site scripting XSS[29].

Las técnicas nombradas son utilizadas por hackers para atacar aplicaciones web tradicionales por lo que las medidas para aplacar estos ataques podrían basarse en las mismas técnicas de protección. El análisis estático, el uso de cajas negras, las herramientas de pruebas de penetración y los cortafuegos de aplicación web reducen significativamente el área de ataque al código de la aplicación en el servidor detectando y resaltando cualquier posible intrusión en la ejecución de la aplicación, prueba y despliegue.

---

[26] Forrester Research es una compañía independiente de investigación de mercado y tecnología que provee asesoramiento sobre el potencial impacto de uso de la tecnología existente, a sus clientes y el público en general.

[27] Se conoce como Inyección SQL, indistintamente, al tipo de vulnerabilidad, al método de infiltración, al hecho de incrustar código SQL intruso y a la porción de código incrustado.
Fuente: http://es.wikipedia.org/wiki/Inyección_SQL

[28] Un ataque CSRF fuerza al navegador web validado de una víctima a enviar una petición a una aplicación web vulnerable, la cual entonces realiza la acción elegida a través de la víctima.
Fuente: http://es.wikipedia.org/wiki/Cross-site_scripting

[29] XSS, del inglés Cross-site scripting es un tipo de inseguridad informática o agujero de seguridad típico de las aplicaciones Web, que permite a una tercera parte inyectar en páginas web vistas por el usuario código JavaScript o en otro lenguaje script similar (ej: VBScript), evitando medidas de control como la Política del mismo origen.
Fuente: http://es.wikipedia.org/wiki/Cross_Site_Request_Forgery

Pero mientras que estas herramientas pueden ayudar mucho, no pueden automatizar la detección de todos los exploits potenciales. Los desarrolladores de RIA todavía tendrá que revisar su código de servidor para las prácticas de programación de alto riesgo, tales como el filtrado de negativos o SQL incorporado.

A pesar de que este tipo de herramientas sirven de mucho, no automatizan la detección de todas las amenazas potenciales; los desarrolladores de aplicaciones RIA deben verificar la codificación que implementen del lado del servidor en aplicaciones de alto riego con prácticas como el uso de filtros negativos[30] o SQL integrado.

2.4.2 *Ataques del lado del cliente*

Los posibles ataques se pueden realizar a una RIA del lado del cliente dependen del framework con el que se esté trabajando. Los framworks RIA usan un entorno limitado para proteger a los clientes de código malicioso, pero estos entornos limitados son creados de manera diferente. Una aplicación bajo AJAX depende del entorno en el que se ejecuta cualquier navegador, mientras que las aplicaciones basadas en la ejecución de su código por medio de plug-ins, como Adobe Flex o Microsoft Silverlight utilizan su propio entorno limitado. Los frameworks de escritorio como Adobe AIR, Curl y Nexaweb por otro lado son más permisivos brindando acceso a la máquina local. Una diferencia clave entre los entornos limitados RIA es permitir o no

[30] Los filtros negativos comprueban las condiciones que desean excluir, como comandos de JavaScript más. Los filtros positivos definen lo que es válido y rechazan todo lo demás.

el acceso entre dominios en el cliente, lo que puede minimizar el riesgo de exposición de los datos para las aplicaciones RIA de estilo mashup[31].

Los desarrolladores pueden configurar las aplicaciones en Flex y Curl para permitir el acceso a varios dominios directamente desde el cliente, pero Silverlight 1.0 por ejemplo no permite el acceso entre dominios y Nexaweb hace cumplir una política de "mismo origen" similar al modelo de entorno limitado de un navegador.

Los proveedores de frameworks RIA realizan inversiones considerables en superar las limitaciones de los entornos de prueba de una navegador. Es natural que los desarrolladores deseen tomar ventaja de la gama de nuevas posibilidades que brindan los proveedores de frameworks RIA, sin embargo, es importante que los líderes de proyecto sean conscientes de la forma en que estas decisiones afectan a las posibles áreas de ataque en el lado del cliente.

Por ejemplo, un desarrollador podría usar al almacenamiento local de datos de Flash o una cookie en el navegador para mejorar el rendimiento de la aplicación, y en el proceso, sin advertirlo expone datos importantes a una aplicación maliciosa. O el administrador de

[31] En desarrollo web, un mashup es una página web o aplicación que usa y combina datos, presentaciones y funcionalidad procedentes de una o más fuentes para crear nuevos servicios. El término implica integración fácil y rápida, usando a menudo APIs abiertos y fuentes de datos para producir resultados enriquecidos que no fueron la razón original para la que fueron producidos los datos en crudo originales.
Fuente: http://es.wikipedia.org/wiki/Mashup_(aplicación_web_híbrida)

la aplicación podría configurar el sitio para permitir el accesos cruzado de dominios con el fin de lanzar un nuevo mashup y sin querer exponer el dominio de la organización a un ataque.

El resultado final dependerá de que los líderes de proyecto aseguren el desarrollo de la aplicación tomando las consideraciones adecuadas para que los desarrolladores realicen los ajustes adecuadors en el entorno en donde se ejecuta la aplicación para asegurar el tiempo de ejecución de sus RIAs.

2.4.3 *Ataques al flujo de comunicación*

Los desarrolladores conectan a los clientes de una RIA con servidores, no solo haciendo uso de servicios asíncronos que mejoran los tiempos de respuesta de las aplicaciones, sino que además colocan parte de la aplicación para que se ejecute en el cliente donde es accesible para todos.

Esto da a los atacantes una idea de la manera más probable en la que deben realizar un ataque para que este tenga éxito. Si un atacante puede redireccionar el cliente RIA hacia su servidor, los usuarios potencialmente podrían pasar información sensible sin enterarse siquiera.

Por ejemplo, los desarrolladores AJAX deben esperar ataques de entradas remotas no autorizadas que puedan atentar a modificar Objetos de Notación Javascript (JSON) o a la carga de datos con estos nuevos comportamientos y tratar de robar información durante el traslado de esta información.

Mientras un desarrollador puede esperar ayuda de sus compañeros para que realicen previsiones y planteen soluciones a riesgos de seguridad por el lado del servidor y esperar que los proveedores de frameworks sigan invirtiendo en seguridad para proteger el ambiente de ejecución del lado del cliente; debe también estar consciente que la responsabilidad de asegurar el flujo de comunicación de datos que se da en una aplicación recae en sus manos.

El uso de XML como formato de transición de datos útiles para ser cargados en una aplicación hace posible que los desarrolladores realicen de manera correcta la transmisión de información en la aplicación, sin embargo los desarrolladores son responsables de asegurar el flujo de la información haciendo uso de encriptación SSL y separando los datos del comportamiento del servidor. Además, los desarrolladores deben tener cuidado de que cada función llamada sea analizada y se comprueben sus debilidades.

Además es recomendable evitar el diseño de esquemas de comunicación que se basan en estados de ejecución extendidos en el cliente, para evitar de esta manera que la autenticación se realice en una función llamada anteriormente y de esta manera evitar esquemas de autenticación que abarque varias sesiones del navegador en períodos largos de tiempo.

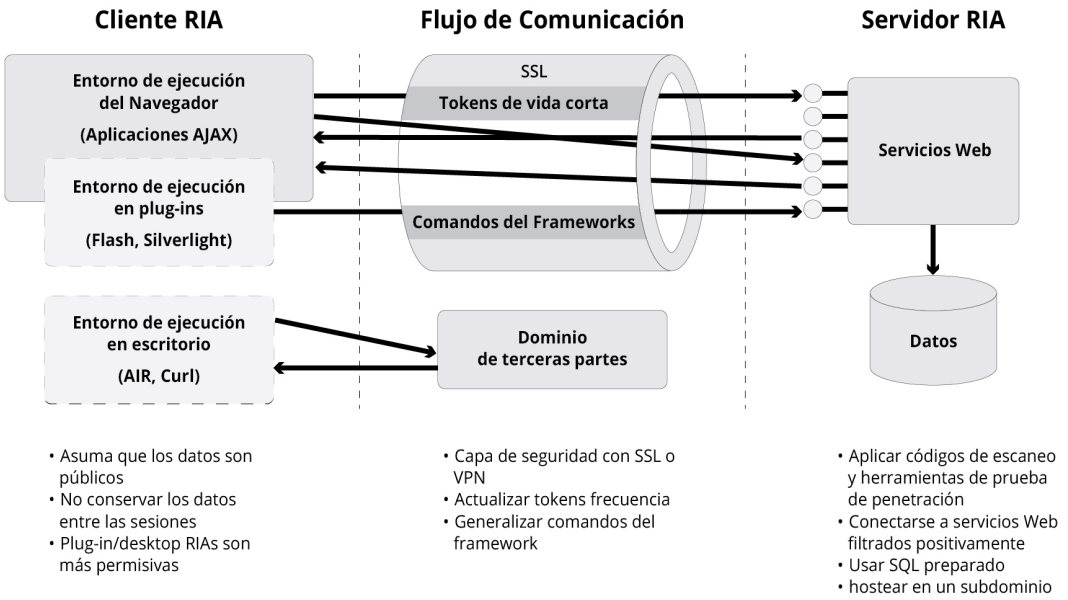

Fig. 8 – Cómo reducir los riesgos de seguridad en las tres facetas de ataque al área de desempeño de una RIA. ***Fuente:*** Forrest Research, Inc.

2.5 Ciclo corto de desarrollo

Cuando surge la necesidad por parte de una organización de implementar una RIA, como en cualquier otra implementación de software se establecen fases a través de las cuales va a desarrollarse el proyecto. El conjunto que abarca estas fases se conoce como ciclo de vida del proyecto. Muchas organizaciones identifican un conjunto de ciclos de vida específico para usarlo en todos sus proyectos.

El costo de un proyecto está ligado al tiempo y los recursos que intervienen en dicho proyecto, y para el caso específico de esta investigación se determina que el uso de una u otra tecnología influye también en el costo total de un proyecto. Hay varios modelos para perfilar el proceso de desarrollo, cada uno de las cuales cuenta con pros y contras.

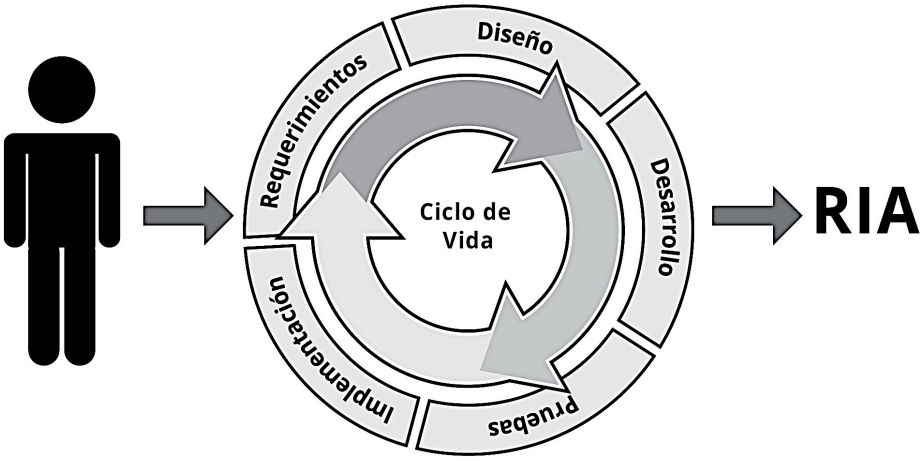

Fig. 9 – Representación del ciclo de vida estándar de una RIA. ***Fuente:*** Desarrollado por el autor

Sin embargo, lo que resulta importante al momento de definir el contexto bajo el cual se desarrollará una RIA consiste en la correcta definición de los requerimientos de los usuarios que van utilizar la aplicación. Cuando los requerimientos se definen de manera adecuada, se establecen modelos de diseño para la aplicación que ayudan a describir una solución para los problemas que se plantean. Por ejemplo, en el caso de desarrollar una RIA que funcione como un cliente de correo electrónico; si no definimos correctamente la forma en la que el usuario ingresará a la aplicación y la funcionalidad que tendrá ésta, se presentan dificultades de varios niveles cuando se esta implementando una solución.

La accesibilidad, el diseño de la interfaz, los datos que verá el usuario son puntos fuertes a tomar en cuenta al momento de desarrollar una RIA para poder determinar que parte de la aplicación se ejecutará en el cliente y que parte en el servidor. En la práctica, muchas veces sin dedicar tiempo de forma explícita para el diseño, los programadores comienzan de forma inmediata a producir código. Antes o después comienza la fase de pruebas de software (a menudo de forma tardía) y de forma casi inevitable los errores son encuentrados. Por tal razón es importante preveer los tiempos

adecuados, pero más que nada es importante tener claros los requerimientos de los usuarios. De esta manera se evitarán retrasos, ambigüedades y pérdidas por un incremento desmedido del presupuesto para la implementación de una RIA.

2.6 Aplicaciones web o aplicaciones desktop

Conforme se desarrollan nuevos dispositivos electrónicos por medio de los cuales los usuarios pueden tener acceso a datos en cualquier momento y lugar, surgen también nuevas formas de desarrollar aplicaciones. Tras el modelo cliente/servidor antiguo en donde el software se duplicaba en cada uno de los clientes pasamos al cliente ligero en donde el software se concentra en servidores de aplicaciones y hacen de esta manera más fácil el despliegue y soporte de las mismas.

Sin embargo, las aplicaciones de cliente ligero han sido muchas veces criticadas por los usuarios por la falta de interacción con herramientas de escritorio y la dependencia que tienen estas de una infraestructura de red HTML o HTTP. Entonces aparece en el medio el cliente enriquecido: mismo que da a los usuarios una mejor adaptabilidad a su forma de trabajar y menor dependencia a una conexión constante con la red. El software inteligente por denominarlo de alguna manera denota un equilibrio en el uso de recursos que comparten tanto el cliente como el servidor.

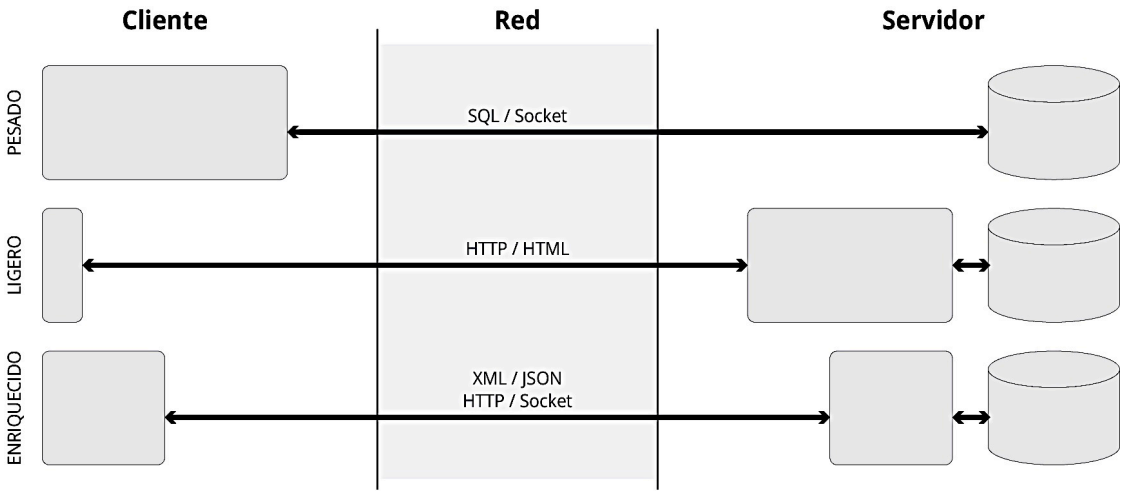

Fig. 10 – Esquema comparativo entre tipos de aplicaciones y su incidencia cliente/servidor.
Fuente: http://www.itaware.eu/

Eventualmente las RIA podrían reemplazar a las aplicaciones de escritorio comunes, sin embargo en ese punto estaríamos hablando de que se creen navegadores que tengan la capacidad de soportar ambientes locales más complejos; por lo que de alguna manera estaríamos esperando que un navegador funcione de manera autónoma lo que vendría a convertirlo en una especie sistema operativo o por lo menos en una parte importante de éste, lo que nos lleva al concepto de RSP[32] (Rich Server Platform), que propone combinar el navegador web y el escritorio en una misma aplicación.

Simplificando las cosas un poco de manera que podamos tener más claro cuál es la alternativa ideal al momento de desarrollar una solución de software es claro de suponer que las empresas pondrán sobre la mesa como principal argumento el costo que pueda representarles el desarrollo de

[32] La propuesta de RSP-UI permitiría combinar de manera gratuita las tecnologías de renderización más adecuadas para el desarrollo de aplicaciones web versátiles. Fuente: http://www.eclipse.org/proposals/rsp/

una RIA o de una aplicación de escritorio, independientemente de las necesidades que tengan siempre prevalecerá el costo.

Es así entonces que el mantenimiento de una aplicación web enriquecida RIA, al encontrarse centralizada en un servidor y a la que pueden tener acceso un número ilimitado de clientes, en el caso ideal de contar con una infraestructura que así lo permita, representará un ahorro importante de dinero; todo esto sin contar además los costos ahorrados por licencias de uso, instalación y soporte para usuarios que una aplicación de escritorio genera.

Además de la creciente demanda que existe en los usuarios de aplicaciones web empresariales por interactuar con interfaces amigables, en la búsqueda de una plataforma de desarrollo estable que cubra los requerimientos empresariales a nivel web, el uso de una RIA se hace inevitable. Si a esto le sumamos el punto de portabilidad por la flexibilidad que tiene una RIA de ejecutarse en varios ambientes y dispositivos, entonces tenemos una solución muy completa para solventar el problema que se presenta al buscar solución al desarrollo de una aplicación empresarial.

Algunas organizaciones como Oracle, Intuit, IBM, SAP, Business Objects son ejemplos de organizaciones que ven en las RIAs una alternativa valiosa para el desarrollo de sus aplicaciones empresariales. Como resultado de esto proveen ejemplos convincentes del potencial de las RIAs en las aplicaciones de negocio a negocio (B2B).

Para muchas de estas compañías, RIA representan una oportunidad para producir aplicaciones interactivas que toman las ventajas de incorporar contenidos multimedia inherentes a dicho modelo que permiten elevar los

niveles de interacción en la experiencia del usuario. Esto conlleva a pensar que las RIAs son aceptadas a nivel mundial para el desarrollo de las aplicaciones empresariales, debido a que proporcionan capacidades muy completas en el cliente a la par de un despliegue accesible.

CAPÍTULO 3

3 Evaluación entre plataformas

El objetivo principal de este trabajo es analizar la evaluación de tres frameworks para el desarrollo de aplicaciones enriquecidas de internet -RIA- con el propósito de definir, de manera general, elementos de juicio que puedan tenerse en cuenta al momento elegir un framework de desarrollo para este tipo de aplicaciones tanto a nivel empresarial como comercial.

El proceso de análisis de la evaluación se compone de varias etapas; mismas que van de la definición de *criterios de evaluación*, establecer una *metodología para comparar* los resultados obtenidos para cada uno de los criterios, analizar un *caso específico* para los tres frameworks evaluados, *comparar los resultados* y *presentar los puntajes obtenidos* como resultado de la evaluación.

3.1 Criterios de evaluación

Para poder establecer un panorama detallado que nos permita realizar una comparación entre elementos comunes de las tres plataformas en análisis consideramos los siguientes criterios de evaluación como principales: facilidad de aprendizaje, facilidad de diseño, facilidad de desarrollo, soporte en ejecución.

3.1.1 *Facilidad de aprendizaje*

La facilidad de aprendizaje se mide por la cantidad de tiempo requerido para trabajar con cierto nivel de eficiencia con una herramienta determinada alcanzando un grado de retención de los conocimientos adquiridos. Para poder medir la facilidad de aprendizaje nos basamos en los siguientes factores:

- Disponibilidad de herramientas de auto aprendizaje
- Disponibilidad de código de ejemplo
- Disponibilidad de aplicaciones de ejemplo
- Soporte del proveedor para evaluación
- Grupos / foros en línea de desarrolladores
- Reutilización de conocimientos de POO[33]

3.1.2 *Facilidad de diseño*

Una parte fundamental en el desarrollo de una RIA es utilizar una herramienta que nos permita explotar al máximo las características que aumenten de manera significativa la experiencia de un usuario al interactuar con la aplicación. Los siguiente factores son los que han sido considerados para evaluar los frameworks en cuestión:

- Disponibilidad de IDE
- Biblioteca de widgets inteligentes de interfaz de usuario

[33] Programación Orientada a Objetos

- Capacidad de diseño drag and drop
- Widgets definidos para interfaz de cliente / usuario
- Diseño de estilo declarativo para reducir codificación

3.1.3 *Facilidad de desarrollo*

En una RIA es importante la apariencia que pueda tener la aplicación, pero es mucho más importante que cada uno de los elementos que contenga una interfaz de usuario funcione a la perfección, sólo así tendremos asegurado el éxito en el intento de aumentar la experiencia de usuario. Para validar las características y facilidades que proponen los frameworks a ser analizados (Adobe Flex, ASP.NET AJAX, Curl) se evaluaron los siguientes aspectos:

- Disponibilidad de IDE para prototipado y desarrollo rápido
- IDE con soporte Intelli-sense[34]
- Retroalimentación en línea para desarrolladores
- Soporte en depuración
- Orientación a objetos
- Soporte para perfiles de la aplicación
- Esfuerzo aplicado en el desarrollo
- Líneas de código generadas en la implementación

[34] "Se refiere a un entorno de programación que trata de acelerar el proceso de codificación de aplicaciones por reducción en malentendidos, errores tipográficos, y otros errores que los programadores hacen comúnmente."
Fuente: http://en.wikipedia.org/wiki/Intelli-sense

3.1.4 *Soporte en ejecución*

Una de las muchas cosas que suceden cuando somos novatos en el desarrollo de aplicaciones web es que tenemos problemas para identificar :

- Facilidad de instalación
- Tamaño reducido / tamaño de la descarga de aplicaciones
- Tiempo inicial de arranque
- Tiempo de reinicio
- Soporte cruzado para navegadores
- Rondas de llamadas al servidor
- Uso de memoria

3.2 Metodología de comparación

El método de comparación en la investigación se usa para determinar y cuantificar las diferencias existentes entre los frameworks evaluados antes del análisis. Para esto establecemos el escenario que cuenta con variables de diferentes ámbitos pero que manejan un entorno común. Partiendo de los datos obtenidos[35] del desarrollo de una aplicación RIA con características comunes en donde se evaluaron: servicios de datos, características principales del modelo RIA en la aplicación, la selección de recursos disponibles referentes al plan de aprendizaje y una revisión del código

[35] Los datos se basan en un estudio realizado por la empresa Sonata responsable del desarrollo de las aplicaciones de prueba, *" Rich Internet Application Development Platforms Curl, Flex and AJAX a comparison".*

fuente que se genera al desarrollar la aplicación. La información se presenta utilizando métricas que permiten cuantificar las características de cada uno de los frameworks además de establecer restricciones que permiten delimitar el estudio de cada tecnología.

"La comparación incluye estudios retrospectivos que observan eventos que ya han ocurrido, y estudios prospectivos, que examinan variables hacia el futuro.

La investigación comparativa es similar a la expermentación en la medida que supone la comparación de un grupo de tratamiento a un grupo de control, pero difiere que se observa el tratamiento en lugar de conscientemente imponerlo. Esto se debe a las consideraciones éticas o, tal vez, a que no es posible hacerlo, como en los estudios retrospectivos."[36]

3.2.1 *Desarrollo de una aplicación común*

Con la finalidad de capturar la experiencia de desarrollo, este estudio se basa en el análisis a la implementación realizada de una aplicación común de "Compras en línea" haciendo uso de cada una de las tecnologías en estudio –Adobe Flex, ASP .NET AJAX y Curl– obteniendo de esta manera la información necesaria de primera mano. La aplicación se diseñó de manera que incluye características RIA y varios escenarios de procesamiento comunes presentes en

---

[36] Tomado del artículo web *"Métodos de investigación: comparación", por Anthony Carpi, Ph.D., Anne E. Egger, Ph.D.* Fuente: http://www.visionlearning.com/library/module_viewer.php?mid=152&l=s

aplicaciones empresariales. Las mismas se enumeran a continuación:

- Obtener, transformar, presentar, operar y envíar datos
- Consumir servicios de datos
- Propagación de cambios coordinados[37]
- Uso de componentes de interfaz de usuario (grid, form, tree, chart)
- Incorporar la lógica de la aplicación (resumen, agregación de datos)
- Presentación de información especializada (reporte tabular)
- Manipulación de componentes estándar (visualización de paneles maximizados y minimizados)

3.2.2 *Servicios de datos*

Las aplicaciones de cliente que tienen características RIA son habilitadas cuando éstas consumen un servicio, es decir estas aplicaciones se deben a los servicios que consumen. Los diferentes frameworks que permiten este tipo de desarrollo tienen recomendaciones y concesiones de diferentes clase concernientes a los servicios de datos.

Mientras en un escenario ideal de desarrollo es deseable trabajar sobre determinados protocolos y formatos adecuados a las

[37] Las prácticas de propagación de cambios explora la forma en que los cambios se realizan de una versión de una aplicación a otra al momento de hacer una migración de versiones.

necesidades de un negocio y de las plataformas de desarrollo seleccionadas, en la gran mayoría de casos y escenarios empresariales debemos trabajar con las arquitecturas y convenciones ya establecidas. Por lo que para poder tener un punto común para el análisis en lo que a servicios de datos respecta elegimos trabajar con REST[38] (XML sobre HTTP) .

3.2.3 *Características principales de RIA en la aplicación*

Las características de los framework que permiten el desarrollo de aplicaciones RIA que han sido tomadas en cuenta para esta evaluación consideran aspectos relacionados a diferentes niveles de desempeño: visual, tiempos de respuesta, balance de carga de trabajo en cliente/servidor y eficiencia en el flujo de trabajo en la red. Los indicadores que representan cada una de estas características son:

a. Para desempeño de componentes IU, el uso de controles inteligentes de tipo:
   - Controles tipo arbol.
   - Data grid con filtros y opciones de ordenamiento.
   - Gráficos de tipo columna.
   - Gráficos tipo pie concentricos con cuadros informativos.

[38] El modelo arquitectónico REST fue desarrollado por el Grupo Técnico de Arquitectura (TAG) del Consorcio W3C en paralelo con HTTP/1.1, basado en el diseño existente de HTTP/1.0. La World Wide Web representa la mayor implementación de un sistema conforme a la arquitectura del tipo REST.
Fuente: http://www.w3.org/2003/Talks/0521-hh-wsa/slide14-0.html

b. Para medir la respuesta de la aplicación desarrollada en un framework específico:
    - El tiempo que le toma a la aplicación solicitar y procesar datos del lado del cliente que esperan una respuesta rápida.
    - Visualización de contenido dinámico en un evento de *mouse over.*
c. Para medir el balance de trabajo en cliente / servidor:
    - Tiempo de respuesta para guardar y procesar datos desde el caché del cliente.
    - Tiempo de respuesta para acceder al servidor únicamente para consumir servicios web.
d. Para medir la eficiencia en el uso de la red:
    - Número de operaciones CRUD realizadas en el cliente con la mínima cantidad de viajes de la información a través del servidor.

La utilización de estos criterios permiten cuantificar las diferencias entre cada uno de los frameworks de manera que se hace posible identificar las virtudes de cada uno.

3.2.4 *Selección de recursos y plan de aprendizaje*

La empresa SONATA[39] a cargo del desarrollo de las aplicaciones se encargó de seleccionar perfiles de desarrolladores expertos con

[39] Sonata Software es una empresa de la India, integradora de sistemas de software. Realizó un estudio independiente de las tres tecnologías en el año 2007.

conocimientos en desarrollo web, pero que no tenían conocimiento de las tecnologías evaluadas; además, seleccionaron los recursos referentes al plan de aprendizaje desde los mismos sitios web de los proveedores de los frameworks de desarrollo. Se comparó el tiempo que le tomó a cada desarrollador en asimilar estas herramientas y tecnología.

3.2.5 *Revisión de código*

A medida que avanzó el desarrollo de la aplicación se contó con el soporte experto de los proveedores de las tecnologías quienes mediante el uso de soporte técnico proporcionaron ayuda e insumos al momento de revisar la codificación realizada.

3.2.6 *Métricas*

Los parámetros específicos mencionados en los criterios de evaluación fueron recolectados conforme avanzó el período de desarrollo, las aplicaciones fueron desplegadas y probadas en entornos de hardware similares. Dicha información será presentada más adelante *(ver sección 3.4. "Presentación de datos").*

3.2.7 *Restricciones*

Como medida de control se establecieron las siguiente restricciones:

- Tiempo - 2 meses de tiempo para desarrollar la aplicación
- Un servicio común de datos - Todas las implementaciones del lado del cliente debieron consumir el mismo servicio de datos.

3.3 Análisis de un caso de implementación

En esta sección del documento se establecen comparaciones entre ciertas características en la implementación de componentes que permiten la visualización de datos en la aplicación mediante el uso de las tecnologías Adobe Flex, AJAX sobre .NET y Curl.

Por favor, consulte el numeral 1 de la sección Anexos para obtener el código fuente generado para cada uno de los items descritos a continuación.

3.3.1 *Componente Data Grid*

Un componente data grid permite crear visualizaciones de datos extraídos de una fuente, ya sea desde una base de datos o desde un servicio web (JSON, XML). En esta etapa se evaluó el componente data grid para capturar información cargada en la aplicación que representan datos de artículos que oferta una tienda "X".

Características comparadas

Adobe Flex

- **Almacenamiento de información:** La tecnología hace uso de una colección de arreglos para almacenar los registros extraídos.
- **Asignación de datos en el componente:** Se puede asignar un arreglo de datos al Data Grid.
- **Filtros de datos:** Los filtros de información se realizan por medio de una función provista por el usuario.

AJAX sobre ASP.NET

- **Almacenamiento de información:** Esta tecnología utiliza record- sets[40] para almacenar registros.
- **Asignación de datos en el componente:** El conjunto de datos extraído puede asignarse al Data Grid pero para poder visualizar la información éste debe ser parte de un contenedor que se actualiza de manera independiente a la página.
- **Filtros de datos:** Los filtros de datos hacen uso de una solicitud de datos como sintáxis.

Curl

- **Almacenamiento de información:** Esta tecnología utiliza recordsets para almacenar registros.
- **Asignación de datos en el componente:** Los recordsets pueden ser asignados a un Record Grid[41].
- **Filtros de datos:** Los filtros de datos hacen uso de las columnas definidas en el recordset.

3.3.2 *Componente para gráficos estádisticos de barras*

Un componente para crear gráficos estadísticos usa barras horizontales o verticales que permiten establecer una relación visual con un conjunto de datos y compararlos entre sí. El objetivo de

---

[40] *Recordset* es una estructura de datos usada en programación cuya utilidad es la de almacenar información desde una tabla de una base de datos. Se usa con frecuencia para obtener conexiones con bases de datos y almacenar el resultado de las posibles consultas que se realicen.
Fuente: http://es.wikipedia.org/wiki/Recordset

[41] *RecordGrid* es una implementación específica de una clase RecordSetDisplay en el lenguaje Curl, mismo que proporciona una presentación tabular de los datos RecordSet funcionalidad integrada para la gestión de la pantalla y los datos en ella.
Fuente: http://developers.curlap.com/curl/docs/caede/en/docs/en/dguide/record-grid.html

utilizar este componente es obtener datos resumidos, por ejemplo, de la agregación de órdenes de compra en un rango de fechas, y presentarlo en un gráfico.

Características comparadas

Adobe Flex

- **Almacenamiento de información:** Para almacenar la información que permita generar el gráfico estadístico, esta tecnología hace uso de un componente XMLListCollection[42].
- **Carga dinámica de datos:** Esta tecnología permite la creación de objetos XML, a partir de esto genera los registros necesarios cuando se necesita cargar datos dinámicamente de forma específica.
- **Filtros de datos:** Esta tecnológia utiliza una clase de objetos ArrayCollection para almacenar los datos, a partir de esta información permite realizar filtros para obtener la información deseada.

AJAX sobre ASP.NET

- **Almacenamiento de información:** Hace uso de un componente denominado DataTable para almacenar la información que presenta el componente Bar chart, encargado de generar el gráfico estadístico.

[42] *XMLListCollection* es una clase de objeto que provee una colección de funcionalidades a una clase de objeto XMLList. *Fuente: http://goo.gl/slKU4*

- **Carga dinámica de datos:** La carga dinámica de datos se realiza sobre componentes OrderLines DataSet, lo que permite obtener el orden de líneas relevantes para el gráfico de barras.
- **Filtros de datos:** El resultado de agregación también se almacena en un DataTable lo que permite filtrar la información a partir de la información almacenada.

Curl

- **Almacenamiento de información:** Utiliza recordset para almacenar información antes de generar un gráfico estadístico de barras.
- **Carga dinámica de datos:** Permite la creación de registros de datos dinámicamente según requiera el usuario.
- **Filtros de datos:** Hace uso de un componente adicional llamado RecordView para obtener información filtrada de datos.

3.3.3 *Componente para obtener un reporte con datos tabulados*

Un reporte de datos tabulados permite obtener información especializada, el propósito de evaluar la generación de este tipo de reporte es comparar los componentes que utiliza cada tecnología para generarlo.

Características comparadas

Adobe Flex

- **Almacenamiento de información:** La tecnología hace uso de una colección de arreglos para almacenar los registros extraídos.

- **Visualización de información:** Hace uso de un componente de lectura DataGrid para desplegar la información en la aplicación.

AJAX sobre ASP.NET

- **Almacenamiento de información:** Para generar un reporte de este tipo, ésta tecnología optimiza la carga de datos con un componente DataRow.
- **Visualización de información:** Despliega el reporte por medio del uso de una componente asp:Table en donde se crea la estructura de la información a desplegar.

Curl

- **Almacenamiento de información:** Esta tecnología utiliza recordsets para almacenar registros.
- **Visualización de información:** Hace uso de una tabla para mostrar la información.

3.3.4 *Componente para acceder a un servicio web*

Un servicio web (web service) es un conjunto de tecnologías que tienen la capacidad de permitir que las aplicaciones intercambien datos en la web por los general en formato XML. El propósito de este tipo de componente es consultar el servicio web para obtener un conjunto completo de datos.

Características comparadas

Adobe Flex

- Recupera los datos XML y los almacena en objetos ArrayCollection.

AJAX sobre ASP.NET

- Recupera los datos XML y los almacena en objetos DataSet.

Curl

- Recupera los datos XML y los almacena en objetos Recordset.

3.3.5 *Componente para generar XML a partir de una llamada HTTP Post*

El método HTTP Post se utiliza para solicitar al servidor que acepte una entidad encapsulada de datos que permiten identificar a la petición como una solicitud del tipo URI[43] que no se muestra.

Características comparadas

Adobe Flex

En esta tecnología se debe crear cada uno de los nodos por medio de un algoritmo y almacenado como cadena de caracteres para luego asignar este valor a un objeto XML.

AJAX sobre ASP.NET

De la misma manera que con Adobe Flex, en esta tecnología se debe crear cada uno de los nodos por medio de un algoritmo pero se envía la información como un flujo de bytes[44].

---

[43] Un Uniform Resource Identifier o URI (en español «identificador uniforme de recursos») es una cadena de caracteres corta que identifica inequívocamente un recurso (servicio, página, documento, dirección de correo electrónico, enciclopedia, etc). *Fuente: http://es.wikipedia.org/wiki/Uniform_Resource_Identifier*

[44] Los flujos de bytes suponen un medio cómodo para gestionar la entrada y salida de bytes. *Fuente: http://www.scribd.com/doc/23277870/ManualJava*

Curl

La tecnología del framework Curl utiliza un objeto XDMDocument[45] para construir y almacenar los datos XML. El objeto generado bajo esta clase XDMDocument se convierte en cadena para generar los registros en formato XML.

3.4 Comparación de resultados

A medida que la implementación se avanzó se hicieron varias observaciones y se midió la optimización en desarrollo y el tiempo de ejecución. Teniendo como resultado los siguientes datos:

	ADOBE FLEX	ASP .NET AJAX	CURL
Número de líneas de código escritas en la aplicación completa	1632	2465	1170
Curva de aprendizaje - Tiempo empleado en días	15	13	19
Tamaño de descarga de la aplicación (Kb)	417	1243	216
Controles de árbol	SI	SI	SI
Data grid	SI	SI	SI
Filtrar información por columnas en el Data grid	SI	SI	SI
Ordenar información por columnas Data grid	SI	SI	SI
Gráficos de barras	SI	SI	SI
Ventana popup para Gráfico "Pie"	SI	SI	SI
Gráfico "Pie" concentrico	NO	SI	SI

[45] El modelo de documento XML de Curl XML provee funcionalidad para procesar información XML en Curl esto es leer, crear, modificar y escribir documentos XML.

	ADOBE FLEX	ASP .NET AJAX	CURL
Herramientas de información - Tool Tips	NO	SI	SI
Reportes tabulares	Posiblemente utilizando Data grid	SI	SI
Elasticidad[46]	SI	SI	SI
Ampliación / minimización de gráficos, tablas y cuadrículas de datos	SI	SI	SI
Acceso a servicios web REST	SI	SI	SI

Tabla 1 - Características evaluadas

Fuente: Realizado por el autor en base a información de *http://www.sonata-software.com*

[46] La elasticidad en este punto de la evaluación guarda relación a la forma en que una variable, módulo o estructura de datos afecta a otra en el desarrollo de la aplicación.

3.5 Presentación de puntaje y resultados

Los resultados correspondientes a los indicadores base y de acuerdo a los datos recogidos se muestran en el siguiente gráfico.

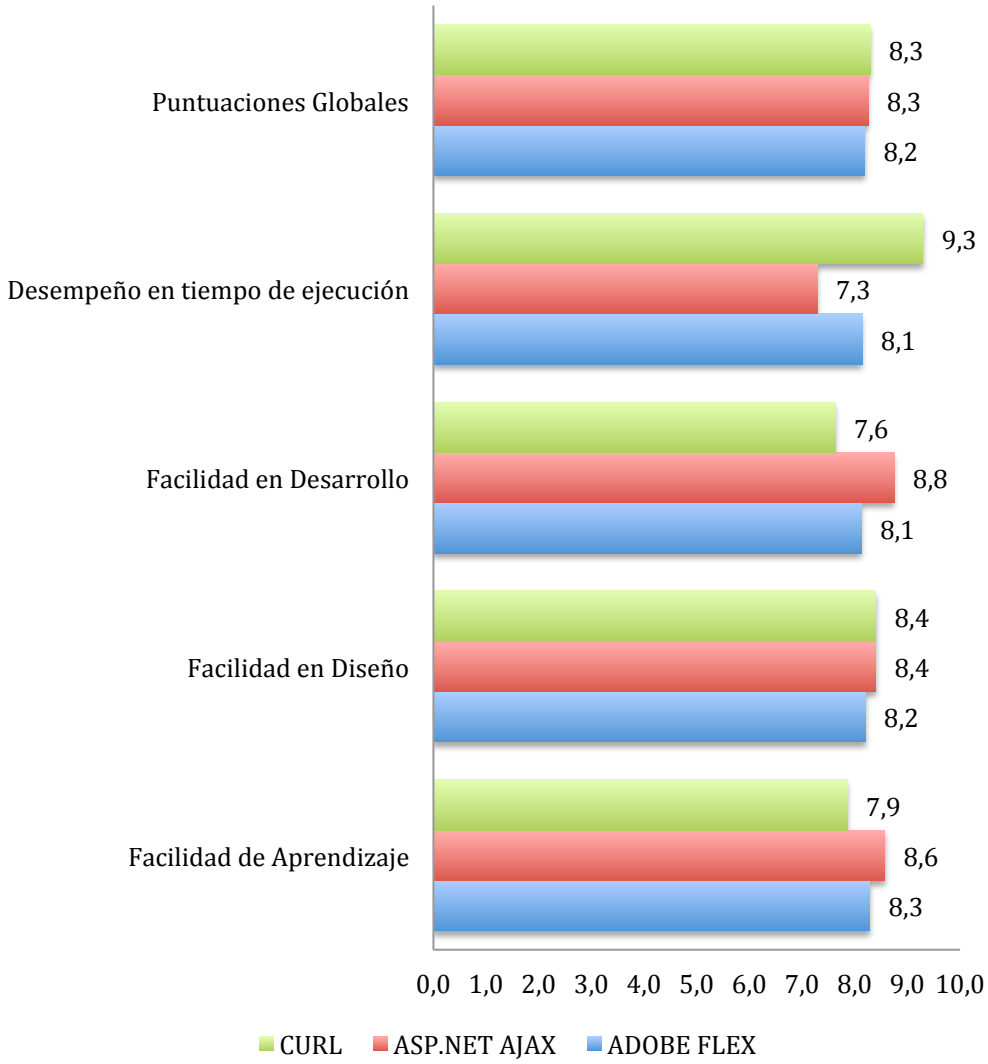

Fig. 11 – Resultados comparativos obtenidos.[47]

Fuente: Realizado por el autor en base a información de *http://www.sonata-software.com*

[47] El detalle de los resultados parciales se pueden verificar en el numeral 2 de la sección Anexos en donde se describe cada uno de los parámetros evaluados junto con los puntajes que se le asignó a cada uno.

De los resultados obtenidos se observa con claridad que cada una de las 3 tecnologías evaluadas tiene características similares, podemos apreciar también que ninguna obtiene un puntaje perfecto. Sin embargo, tienen características que se fortalecen en ámbitos específicos. Adobe Flex es una tecnología que permite trabajar de manera más cómoda en interfaces de usuario complejas, es ideal para crear diseños de componentes personalizados, ASP .NET con AJAX nos permite crear aplicaciones web enfocadas a desarrollar negocios del tipos B2B y B2C y su conjunto de herramientas es bastantes flexibles. Por otro lado, el conjunto de tecnologías de Curl resulta ideal al momento de manejar grandes cantidades de datos.

Como sucede con la tecnología el desarrollador será siempre quien explote o limite las características que posea una u otra tecnología. Estas características se verán mermadas o potenciadas por la experiencia y conocimiento que posea.

CAPÍTULO 4

4 Prospectos nuevos en la proyección de negocios basados en Aplicaciones Enriquecidas de Internet (RIA)

Las RIAs utilizan tecnologías como Adobe Flex, AJAX y Curl para evitar que los usuarios tengan que lidiar con el problema de mecanismos de respuesta lentos al interactuar con una aplicación web. En lugar de las interminables rondas de click-presentar-responder marcadas por peticiones del servidor ocasionales cuando faltan datos para enviar información, las RIAs combinan la interactividad de aplicaciones de escritorio y una rica calidad en contenido.

Por tal motivo este tipo de aplicaciones representan la siguiente transición en la evolución de las aplicaciones web, y como toda transición lógica implica que existen nuevas oportunidades para ser exploradas. Para maximizar los beneficios de la adopción de esta forma de desarrollar aplicaciones acorde a las tendencias actuales, una empresa debe evaluar diversos aspectos a tomar en cuenta; como por ejemplo, la forma en que una RIA se ajustará a una arquitectura basada en SOA[48], el rol que tendrá la aplicación en un modelo SaaS[49] y de mashups[50], además de evaluar los

---

[48] SOA - *Service Oriented Architecture,* brinda una forma bien definida de exposición e invocación de servicios (comúnmente pero no exclusivamente servicios web), lo que facilita la interacción entre diferentes sistemas propios o de terceros. *Fuente: http://es.wikipedia.org/wiki/Arquitectura_orientada_a_servicios*

[49] Software como Servicio (Software as a Service, SaaS) es un modelo de distribución de software donde el soporte lógico y los datos que se manejan se alojan en servidores a los que se accede con un navegador web desde un cliente, a través de Internet. *Fuente: http://es.wikipedia.org/wiki/Software_como_servicio*

beneficios y retos que representarán para la empresa la adopción de una RIA.

Las oportunidades de negocio en torno a este modelo de desarrollo de aplicaciones están presentes de diversas formas, ya sea por medio de la creación de RIAs que atraigan la atención de los usuarios y que ofrezcan la capacidad de interactuar no solo con la aplicación sino también con otros usuarios; o mediante el desarrollo de RIAs que permitan elevar los niveles de prestación de servicios de las empresas, ayudar a mejorar su productividad, bajar las costos mediante el buen uso de la tecnología o inclusive impulsar las relaciones inter empresariales.

Sin embargo, la tecnología sola no hace mucho para transformar una oportunidad de negocio en un negocio rentable, la diferencia radica en la manera de hacer que una solución se vuelva imprescindible. Para lograr esto, es necesario explotar al máximo el framework que utilicemos de manera que la funcionalidad de la aplicación sea capaz de lograr lo que el usuario espera que haga, incorporando a la misma capacidad visual y estética para lograr captar su atención.

La idea es que cada aplicación RIA desarrollada se vuelva una herramienta que los usuarios quieran usar, de esta manera se logra elevar el valor de un producto de software.

---

[50] En desarrollo web, un mashup es una página web o aplicación que usa y combina datos, presentaciones y funcionalidad procedentes de una o más fuentes para crear nuevos servicios. *Fuente: http://es.wikipedia.org/wiki/Mashup_(aplicación_web_híbrida)*

4.1 Siguiendo la tendencia Web 2.0

A través de las diferentes etapas evolutivas que ha tenido el Internet, tanto a nivel de desarrollo de estándares como a la forma en que los usuarios han ido utilizando la información, llegamos a la etapa en que se le ha denominado Web 2.0 y en la que nos encontramos en la actualidad.

Esta denominación nada tiene que ver con la transición que podría creerse existe de una versión a otra, pues no existen versiones de web. El término está ligado a la forma en que se han ido desarrollando y utilizando las diferentes tecnologías para permitir que los usuarios puedan crear y compartir contenidos.

De una manera simple en la actualidad las RIAs permiten por medio de su uso generar una experiencia de interacción diferente en los usuarios. Y son justamente este tipo de tecnologías las que hacen posible que la web 2.0 se desarrolle. Al realizar la transición de las aplicaciones cliente/servidor tradicionales de escritorio hacia las RIAs se obtienen un número de beneficios, tanto para los usuarios que las usan como para los diferentes negocios que las aplican. Ya sea que los usuario de las aplicaciones sean los empleados de las empresas, sus clientes o una combinación de ambos, las RIAs no solo ayudan a que la información sea accesible también ofrecen beneficios como:

- *Interfaces más intuitivas.* Las RIAs pueden lograr que la información se presente de manera eficiente. Es posible proveer una retroalimentación visual inmediata a los usuarios y proporcionar pautas que representen una ayuda de navegación para el usuario.

En este punto representan un beneficio de dos maneras, ya sea por generar un ahorro de tiempo al representar aplicaciones fáciles de usar y que mejoran la productividad de los empleados de las empresas que las aplican y usan o porque las aplicaciones permiten obtener mejores ingresos al proporcionar a los clientes herramientas que ayudan a incrementar las ventas de una compañía.

- *Mayor facilidad de uso para los clientes o usuarios finales.* Con el modelo de aplicaciones RIA los usuarios únicamente necesitan de un navegador web estándar para acceder a su aplicación. Este punto brinda una ventaja decisiva sobre los competidores de cualquier empresa cuyas aplicaciones requieran que los usuarios instalen un cliente de escritorio.
- *Aumentar el tiempo de lanzamiento en el mercado.* Dado que lanzar una RIA sigue un proceso similar al de publicar un sitio web, es posible lanzar nuevos productos y aplicaciones al mercado siguiendo un proceso mucho más rápido que los que solían seguirse para lanzar al mercado una aplicación tradicional cliente/servidor.
- *Facilidad de mantenimiento.* Una RIA reside en el servidor de aplicaciones donde no residen componentes de cliente, es posible solucionar problemas con bugs[51] una vez en el servidor y los usuarios pueden apreciar los cambios casi de manera inmediata.
- *Mejor respuesta de las aplicaciones.* Las RIAs pueden ofrecer tiempos de respuesta por debajo de un segundo a través de Internet, lo que resulta bastante útil, tanto como una aplicación tradicional.

---

[51] Un error de software, comúnmente conocido como bug (bicho), es un error o fallo en un programa de computador o sistema de software que desencadena un resultado indeseado.
Fuente: http://es.wikipedia.org/wiki/Error_de_software

- *Obtener un alcance mayor de mercado.* El uso de la web como plataforma ayuda a proporcionar a cualquier aplicación un cierto grado de disponibilidad, lo que significa que casi cualquier persona puede acceder a la aplicación desde casi cualquier lugar y en la actualidad también desde una variedad de dispositivos (laptops, tablets, smartphones).

Una vez que se sientan las bases para desarrollar soluciones Web 2.0, es mucho más accesible desplegar nuevas soluciones de forma rápida. Debido a que las RIAs residen en el servidor de aplicaciones, se puede desplegar aplicaciones de forma gradual, añadiendo funciones conforme se vayan desarrollando. Y, a diferencia de las aplicaciones compiladas, las RIAs son aplicaciones muy flexibles y se pueden configurar para satisfacer necesidades específicas.

Uno de los puntos fuertes al hacer uso de los diferentes frameworks que permiten desarrollar este tipo de aplicaciones es la capacidad que tienen las RIAs para ser objeto de personalización permitiendo inclusive adaptar la apariencia de la aplicación a la imagen corporativa de cualquier organización o empresa.

4.2 Crecimiento rentable de una RIA

Una de las mejores maneras de facilitar la adopción de una nueva tecnología es introducirla donde puede generar mayores beneficios. A primera vista se puede apreciar que las RIAs representan un medio válido para atraer la atención de los consumidores.

Esta opinión se ve reforzada por muchos[52] debido a que la primera ola de aplicaciones RIA que se han desarrollado se ha centrado en los consumidores y el consumo de medios multimedia. El enfoque es predominante en los medios y el marketing así como la aplicabilidad de las RIAs en los medios sociales. En este campo, se han impuesto los contenidos gráficos y multimedia.

Sin embargo, esto ignora la propuesta de valor que las RIAs proponen a las empresas en donde la capacidad de consumir información efectivamente; interpretar y crear datos, además de comunicar de manera flexible audiencias claves (empleados, cadena de suministros, consumidores); permite a las compañías e industrias crear ventajas diferenciadoras con sus competidores.

A pesar de la superposición de propuestas de valor que se generan, hay una clara separación entre dos tipos de RIAs; las *RIAs orientadas a consumidores* y las *RIAs orientadas a cubrir necesidades empresariales*.

4.2.1 *RIAs orientadas a consumidores*

En este tipo de aplicaciones generalmente se distinguen dos tipos de RIAs: aquellas que permiten a las empresas crear un nexo con sus clientes a través de mejorar positivamente la experiencia que le crean cuando éste consume un producto (aplicaciones dirigidas al

[52] Un estudio realizado por la empresa Forrester, "(By Any Other Name) RIAs Will Power Future Online Experiences" expone la manera en que las RIAs pueden permiten personalizar, agregar, enfatizar y socializar experiencias y darle valor agregado a los negocios en línea.

marketing y B2C[53]); y aquellas que proveen una mejor experiencia cuando los usuarios consumen contenidos como música, películas, libros (aplicaciones dedicadas al contenido - media applications). Las RIAs cumplen un desempeño bastante aceptable en estos dos ámbitos dadas las capacidades multimedias e interactivas que brindan a desarrolladores y diseñadores para atraer usuarios, permitiendo de esta manera crear aplicaciones excepcionales.

Atraer usuarios representa siempre un reto para las empresas que desarrollan aplicaciones orientadas al B2C. Es una de las causas que ha restringido el potencial de las compras en línea comparadas con visitar tiendas o negocios en el mundo real.

Por ejemplo los primeros intentos de crear tiendas de ropa en línea, eran sitios web en donde el usuario podía apreciar la ropa en un maniquí con sus medidas, muchas veces eran lentos y difíciles de usar. Las RIAs hoy en día son desarrolladas para mejorar este tipo de inconvenientes, teniendo como resultado una solución que le permite al usuario tener una experiencia más cercana a la realidad. Esto inmediatamente se traduce en una mejora en el proceso de venta de un producto, disminuyendo de manera significativa los costos asociados al mismo.

[53] B2C se refiere a la estrategia que desarrollan las empresas comerciales para llegar directamente al cliente o consumidor final. *Fuente: http://es.wikipedia.org/wiki/B2C*

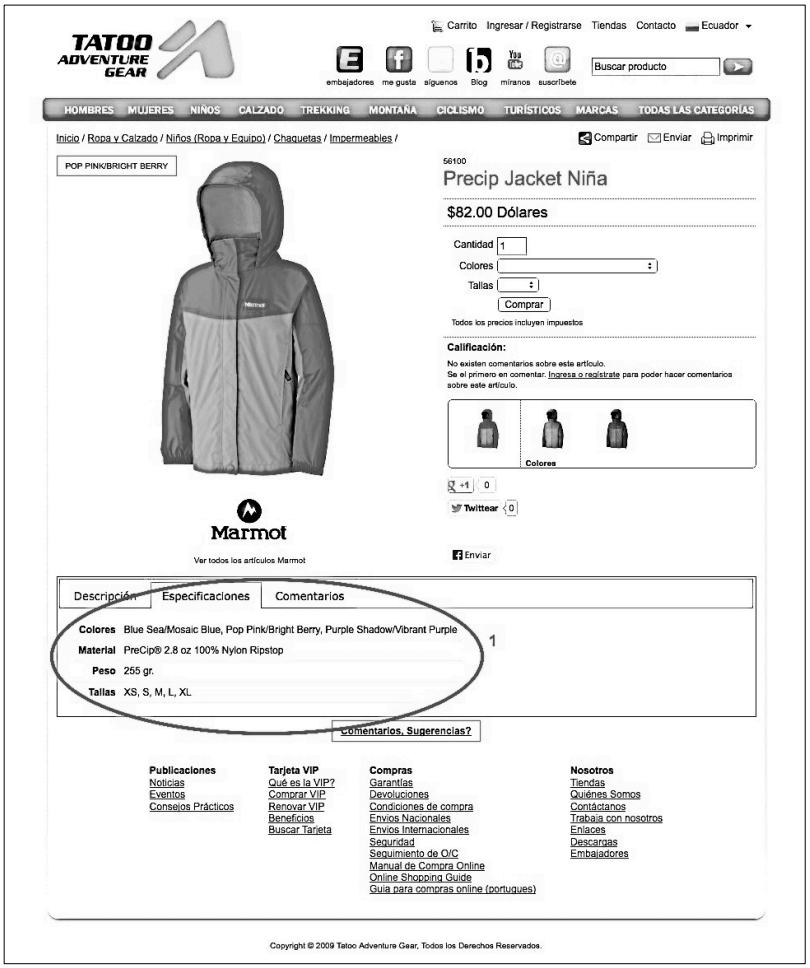

Fig. 12 – Sitio web tradicional de venta de ropa en línea.

Fuente: http://ec.tatoo.ws/

Fig. 13 – Sitio web tradicional de venta de ropa en línea.

Fuente: http://ec.tatoo.ws/

En la imagen de la figura 10 y en la de la figura 11 se puede apreciar un sitio web desarrollado bajo en esquema web tracional en donde no se toman en cuenta aspectos importantes que permitan mejorar la experiencia del usuario.

Por ejemplo, el sitio web si bien muestra una imagen grande del producto en venta no muestra información relevante para el usuario, como por ejemplo la equivalencias en la talla de la chaqueta, solo muestra información de disponibilidad en tallas S, M, L, XL lo que bajo ningún punto garantiza que la talla que seleccione el usuario se ajustará a lo que finalmente busca. Si bien es cierto existe un esquema de estandarización de tallas[54] para la producción de prendas de vestir en grandes volúmenes el usuario común por lo general no está al tanto de las medidas que describe este estándar y muchas veces los productores de ropa tampoco, pues generan los patrones de ropa de acuerdo a otras prendas. Como resultado de esta falta de información el usuario podría comprar una prenda de vestir y al recibirla llevarse un disgusto al no obtener el producto que esperaba.

Analizando este sitio tradicional mediante el cual el usuario puede realizar una compra en línea podemos apreciar también que no se proporciona al usuario información importante como fotografías a

[54] El sistema estándar de tallas de ropa *ISO/TR 10652:1991* determina los tamaños de una prenda de vestir de acuerdo a grupos de tamaños de cuerpo.

gran escala que permitan apreciar detalles de la prenda; así como tampoco se proporciona información relacionada de otros productos que podrían generar una compra en masa. En un panorama amplio, el sitio web brinda una experiencia de usuario pobre, no representa bajo ningún punto una ventaja competitiva ante la competencia de la empresa pues no se diferencia de otros sitios de su segmento comercial.

Este conjunto de características pobres son las que llevan a pensar a las empresas, o específicamente a los empresarios que no es necesario establecer prácticas orientadas a enriquecer la experiencia del usuario en la web, lo que de varias formas representa pérdidas tanto para el sector tecnológico como para el sector empresarial.

A continuación vamos a revisar un ejemplo diferente, en donde se aprovechan las funcionalidades de lo que podría acercarse bastante a una RIA, en el mismo segmento de mercado. El siguiente sitio web a ser analizado aprovecha la información y funcionalidad para lograr atraer a los usuarios y convertirlos en potenciales clientes.

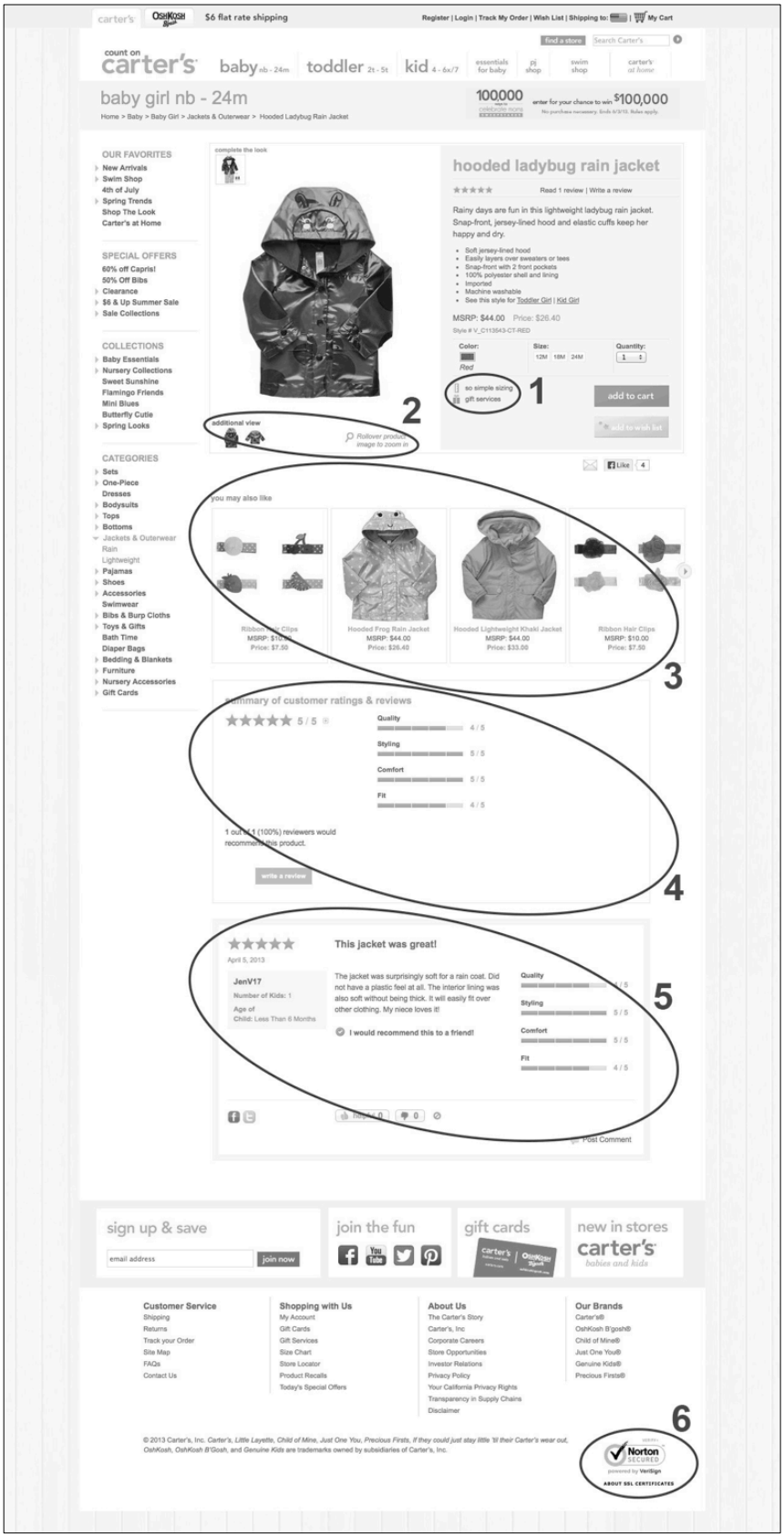

Fig. 14 – Sitio web modelo RIA de venta de ropa en línea.

Fuente: http://www.carters.com/

Como podemos apreciar distinguimos por lo menos 6 puntos de mejoras sustanciales, basados en la creación de lo que podrían considerarse componentes dentro de una RIA, concentrados en crear una experiencia de usuario positiva y atraer nuevos potenciales clientes:

1. Se establecen controles que le permiten al usuario conocer las equivalencias en las tallas de las prendas de vestir en diferentes unidades de medida, en este caso en centímetros y pulgadas además de una tabla informativa que permite establecer una relación entre tamaño y peso.

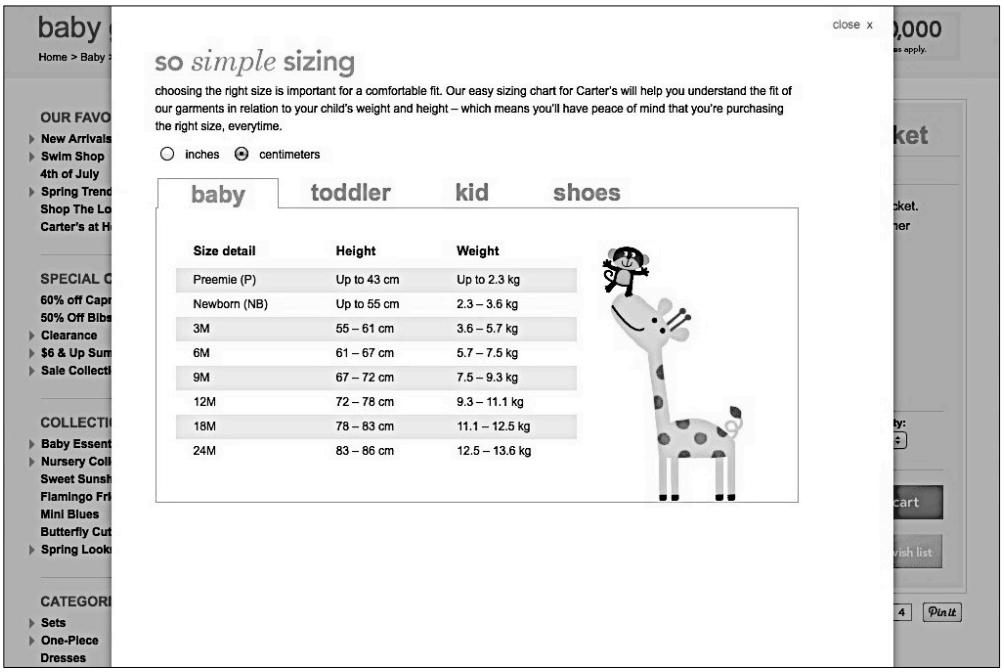

Fig. 15 – Característica de equivalencias de tallas.

Fuente: http://www.carters.com/

2. Se presenta información visual correspondiente a detalles de la prenda mediante un componente de programación que permite

realizar un acercamiento de buena calidad a cada uno de los puntos de la imagen expuesta.

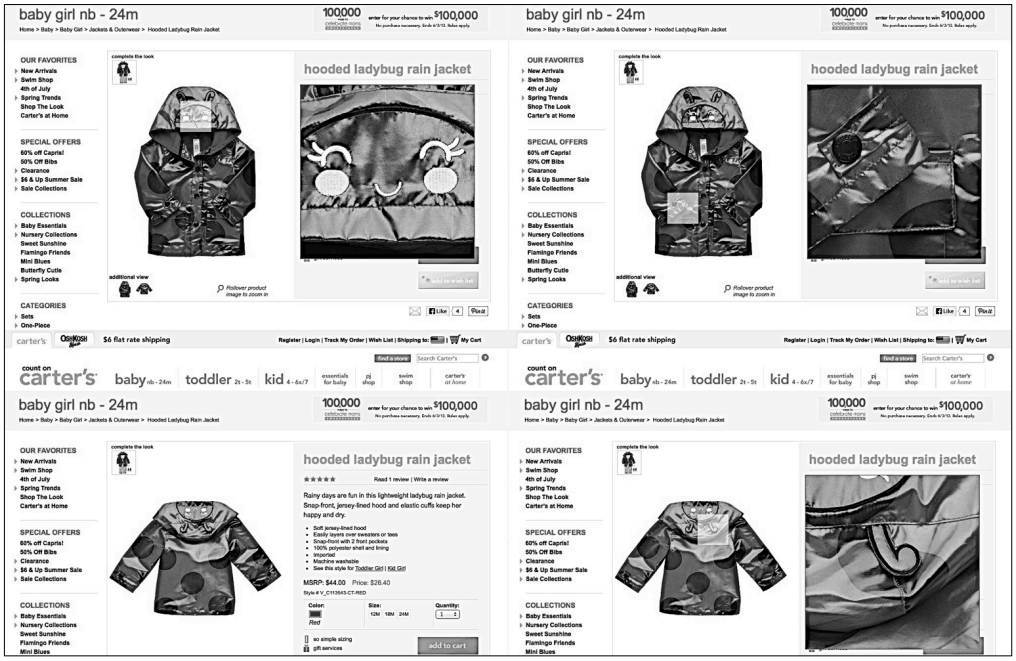

Fig. 16 – Información visual del producto.

Fuente: http://www.carters.com/

3. Se dan sugerencias de prendas con características similares a las que se muestra, lo que ayuda a generar interés por otros productos y extiende la posibilidad de aumentar el pedido de la compra final.
4. Se muestra información resumida de las revisiones y calificaciones realizadas por otros clientes respecto al mismo producto.
5. Se pueden leer comentarios de otros usuarios que ya adquirieron el producto, lo que ayuda a generar mayor confianza en los potenciales clientes al momento de tomar una decisión de compra.
6. Se utilizan información visual para representar elementos externos al sitio que proporcionan información relacionada a medidas de seguridad que están implementados en el sitio de

comercio electrónico mismos que ayudan a generar todavía mayor confianza en los potenciales clientes.

Apenas con estas contadas características la experiencia de compra del usuario cambia de manera radical, esto ayuda a establecer un canal de venta mucho más rentable y una tienda virtual a la que los usuarios siempre querrán regresar.

Entre otro ejemplo de aplicaciones reales en donde las RIAs tiene la posibilidad de cambiar la forma y hábitos de consumo en los usuarios es en aquellas en donde se dan especial importancia a la personalización de productos. Por ejemplo, un proceso de venta de autos elevado a una aplicación RIA permitiría al usuario crear un auto que se ajuste a sus gustos específicos, mirar como luciría, realizar cambios a la apariencia y al incorporar tecnologías como la realidad aumentada o simulación 3D incorporada a la RIA tener una experiencia muy cercana a conducirlo.

Sin duda que las limitaciones en este sentido las ponen las empresas y sus presupuestos.

4.2.2 *RIAs empresariales*

Durante algún tiempo las empresas han tenido que escoger entre, utilizar aplicaciones web que a menudo tienen características pobres de uso y desplegar aplicaciones de escritorio algunas veces complejas y difíciles de instalar. Tomando las mejores características de estas dos tipos de soluciones, las RIAs proponen una mejora de

desempeño, visual y de retorno de inversión lo que añade valor a un negocio en lugar de complicarlo.

Existen diferentes caminos mediante los cuales puede desarrollarse una RIA a nivel empresarial. Las RIAs funcionan bastante bien en líneas de negocio que necesitan desarrollarse de manera rápida y que requieren asimismo mantenimiento constante. Debido a su tamaño reducido pueden llegar a los usuarios a través de un servidor web o un servidor de aplicaciones en la red de comunicaciones de una empresa.

De la misma forma que sucede con las RIAs orientadas a consumidores, se pueden considerar dos áreas iniciales en las cuales concentrar el desarrollo empresarial: aplicaciones orientadas a tareas y aplicaciones que proveen soporte para la toma de decisiones. Para poder establecer el área inicial a la que se debe enfocar el desarrollo de una aplicación de este tipo es imprescindible entender los tipos de lógica que forman parte de una RIA:

- **Lógica de negocio.** Tanto la lógica de negocio empresarial, como los sistemas de relaciones con el cliente (CRM) o los sistemas de recursos de planificación empresariales (ERP), bases de datos y datos del negocio deben disponer de una interconexión congruente con la interface de una RIA. En la lógica de negocio se deben establecer los procesos relacionados a la actividad empresarial. Lo ideal es que esta interface se realice a través de un conjunto de servicios como parte de una arquitectura orientada a servicios (SOA), o a través de interfaces de programación de aplicaciones (API).

Esto significa que es importante mantener separados cada uno de los componentes de la aplicación para utilizarlos cuando sea necesario.

- **Lógica de presentación.** En el contexto de una RIA, la lógica de presentación se refiere a la lógica de control de eventos de una interfaz de usuario que interactúa directamente con los controles mostrados en la misma. Por ejemplo, cuando un usuario presiona la *tecla tab* en una interfaz de usuario, un evento es disparado y controlado por la presentación lógica o el manejador de eventos.

  La presentación lógica de una aplicación puede realizar validaciones directamente, interactuar con la lógica de la aplicación, y recuperar datos.
- **Lógica de aplicación.** La lógica de la aplicación normalmente mantiene el contexto de la aplicación o estado e interactúa con la lógica de presentación, pero no con la interfaz de usuario.

Una buena práctica para el desarrollo de una RIA es implementar el patron de desarrollo del modelo vista controlador (MVC) . En este tipo de implementación el controlador proporciona la lógica de presentación que interactúa con los controles de interfaz de la vista y también interactúa con el modelo que proporciona la lógica de la aplicación. Esta forma de plantear una arquitectura permite mantener primero una idea clara de lo que se quiere lograr y segundo al tener clara esta idea, es posible dimensionar de manera adecuada los requerimientos de una empresa para desarrollar una RIA y plantear costos cercanos a la realidad.

Para los líderes del mercado, las RIAs representan una forma de proteger los mercados y aumentar la eficiencia por medio de procesos de negocio ágiles. A medida que se desarrolla este modelo de aplicaciones la elección en torno a las RIA no estará en torno a implementarlas o no, sino que estará alrededor de que tipo de RIA implementar.

4.3 Áreas claves de impacto sobre el rendimiento de la inversión en aplicaciones RIA

Desde que el desarrollo de aplicaciones RIA ha sido adoptado por varias empresas se ha comprobado que ofrecen diferentes tipos de retornos a la inversión (ROI) que se realiza. Estos beneficios son medibles desde diferentes aspectos, mismos que incluyen mayores tasas de conversión de clientes, incremento en la productividad de las empresas, así como crecimiento de los niveles de lealtad de los clientes.

Para obtener los resultados esperados es imperativo tomar en cuenta algunos puntos claves que tienen que ver sobre todo con la manera que la aplicación se adaptará a las necesidades de una empresa o usuarios finales. Es importante analizar la manera en que se acoplará la RIA a los objetivos que se plantee la empresa y verificar la forma en que los diferentes frameworks de desarrollo se pueden acoplar al cumplimiento de estos objetivos.

Los beneficios básicos que se esperan obtener al iniciar un proyecto de este tipo son: menores costos totales de propiedad, tener una aplicación que se acople a SaaS, por supuesto añadirle valor a la experiencia de usuario y mejorar la productividad para casos de RIAs orientadas a empresas,

además de tener una aplicación que esté disponible en cualquier momento y dispositivo. A estos beneficios básicos se deben añadir el cumplimiento de normas de seguridad que garanticen la privacidad en el uso de información.

Por otro lado en un mundo globalizado como el actual en donde los clientes son cada vez más exigentes y la fidelidad hacia una marca es más difícil de obtener, establecer un compromiso más profundo con los clientes es esencial para el éxito de una organización. Las RIAs permiten crear interacciones dinámicas y útiles, en otra palabra atractivas.

Los ejecutivos de negocios reconocen con creces el valor que conlleva tener a sus clientes identificados con sus negocios. Por ejemplo, en un estudio realizado por la empresa norteamericana Economist Intelligence Unit[55], el 80% de ejecutivos entrevistados cree que la mayor atracción hacia una marca se traduce en un mayor compromiso de lealtad, mientras que el 75% dijo que ellos creen que esto representa mayores beneficios relacionados a la rentabilidad.

La fidelidad a una marca por parte de los clientes es fundamental para la transformación de los mismos en defensores activos tanto de las marcas y por ende de las empresas dueñas de las mismas. El 79% de los encuestados dijo que los clientes que están atraídos por una marca recomendarán productos y servicios a los demás.

---

[55] *Economist Intelligence Unit* comprende un área de investigación perteneciente a la empresa *The Economist* y se encarga de realizar estudios en diversos ámbitos de negocio, uno de ellos la tecnología. El estudio se lo realizó por pedido de la empresa Adobe Inc. en el año 2007.

Las RIAs representan uno de los más desafiantes procesos de desarrollo y de manera general involucran tres capas como lo había ya descrito antes, la capa sobre la cual se describe la lógica de negocio, la que describe la lógica de presentación y la que describe lógica de la aplicación. Una vez que se tienen claros los objetivos con los que tiene que cumplir una RIA es momento de empezar el ciclo de desarrollo.

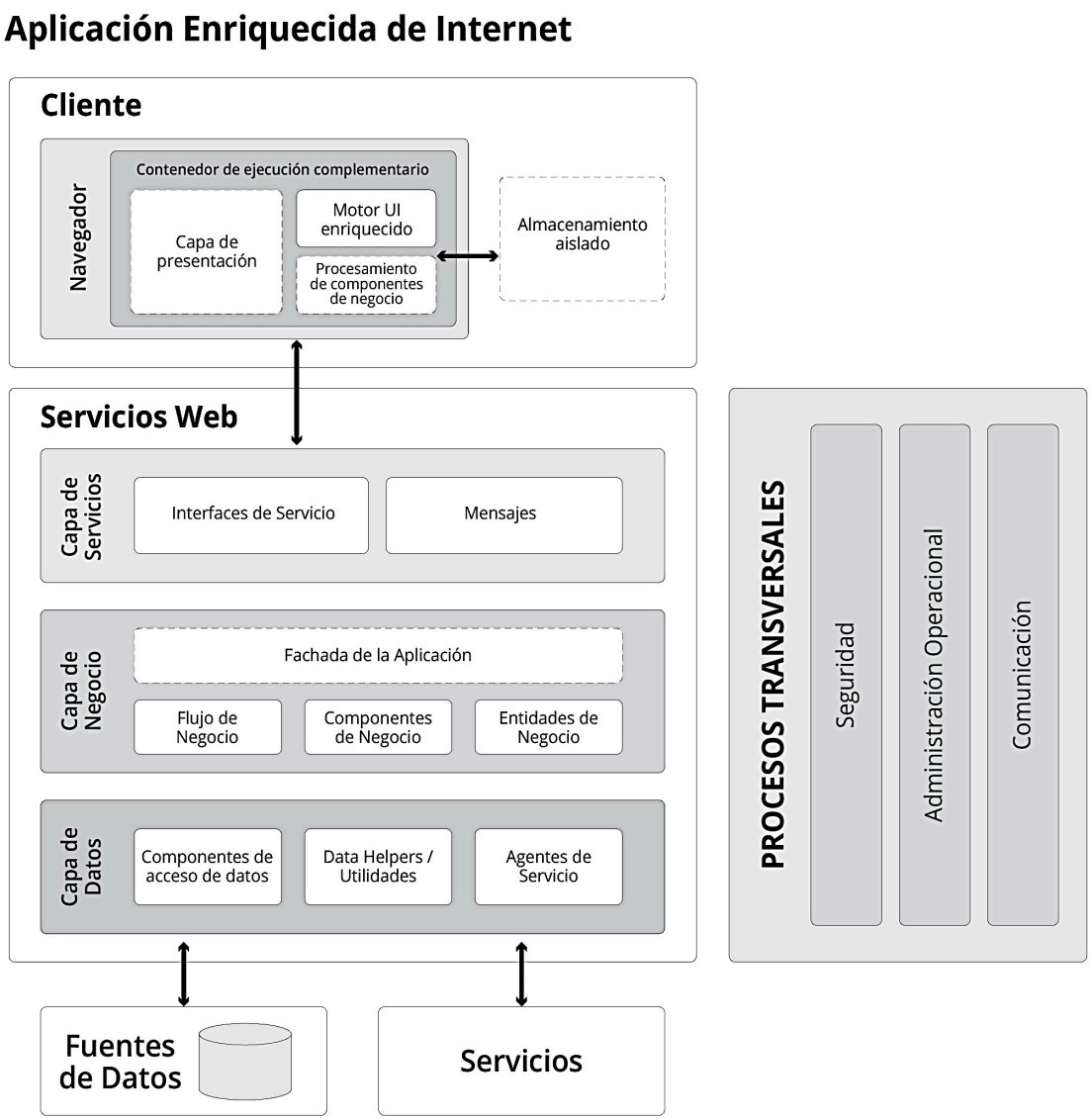

Fig. 17 – Arquitectura detallada de una RIA.

Fuente: Microsoft - Designing Rich Internet Applications

En el esquema se puede apreciar un panorama completo de los componentes o partes necesarias que intervienen en el desarrollo de una RIA. Los procesos transversales como *seguridad, administración operacional del/los servidores y las interfaces de comunicación* son puntos claves a tomar en cuenta porque intervenienen en todas las capas de la aplicación.

Las aplicaciones cliente-servidor tradicionales involucran una arquitectura relativamente sencilla, que descansa sobre una permanente conexión entre el servidor y el cliente. Con tal diseño fuertemente acoplado, casi no se requiere manejar explícitamente ni preservar distintos estados de lógica. Por el contrario, las aplicaciones web, que centralizan su procesamiento en el servidor, dejan al cliente desacoplado, o débilmente acoplado. Mientras las aplicaciones web realicen pequeños y simples procesos lógicos, y tengan una limitada riqueza de interactividad, pueden implementarse con arquitecturas web estándar y un simple manejo de sesión.

La arquitectura RIA procura por otro lado proveer el confort de la experiencia del usuario altamente interactiva y fuertemente acoplada, pero muchas veces se implementa sobre una infraestructura con una configuración débil. Se debe extender el análisis de las debilidades que tiene la configuración para solventar el sofisticado diseño de sistemas y capacidades de programación que se encuentran en equipos de desarrollo focalizados en soluciones de negocios.

Otro punto a tomar en cuenta es la interconectividad. Con el incremento sostenido del ancho de banda, más y más recursos están disponibles para las aplicaciones web, llevándonos con tecnologías como AJAX cerca del nivel del tipo de interacción de aplicaciones cliente-servidor. Esto se puede

apreciar, por ejemplo, en la función “Google Suggest” cuando carga un rango de términos de las búsquedas más populares a medida que uno ingresa las letras en el campo de búsqueda.

Esta funcionalidad puede ser apropiada para tareas como la navegación de páginas web, pero el ancho de banda actual de Internet no es suficiente para soportar aplicaciones fuertemente acopladas, con decenas de campos activos por pantalla. Para superar esta limitación, una aplicación de tipo RIA debe soportar más lógica en el lado del cliente y también equilibrar el procesamiento entre el servidor y el cliente.

Por lo tanto los desarrolladores terminan trabajando con dos procesos físicamente separados y lógicamente dependientes, que además se ejecutan en cliente y servidor. El cliente ahora precisa de un “cerebro” y “memoria” para encarar las constantes secuencias de instrucciones del servidor. Mantener una sesión con coherencia requiere en consecuencia una administración sofisticada del estado y sesión.

De lo visto hasta aquí, resulta claro que desarrollar y desplegar una aplicación de tipo RIA involucra múltiples escenarios de comunicación entre el cliente y el servidor de la aplicación. Se deberá entonces realizar los ajustes necesarios en el lado del servidor así como en los elementos que intervienen o forman parte de la comunicación; el ajuste de desempeño involucra la reducción de los escenarios de comunicación entre el servidor y el cliente al mínimo posible y la reducción del tamaño de los paquetes que se transmiten a fin de limitar el uso de la red y así alcanzar un tiempo óptimo de respuesta de la aplicación.

A mejores tiempos de respuesta logramos tener mejor receptividad de los usuarios de la RIA, obtenemos el beneficio por fidelizarlos, mejorar la productividad, en resumen, obtenemos los beneficios por fortalecer las líneas de negocio.

4.4 El aumento de la valoración de negocios con RIA

A medida que los tiempos cambian, también cambia la forma en que la tecnología llega a influir en los negocios. A inicios de los años 80 los procesos informáticos se limitaban al procesamiento de datos en servidores o mainframes y únicamente un número reducido de empresas contaban con los recursos necesarios para poseer esa tecnología, para documentar muchas veces la información en muchos casos se hacía uso de máquinas de escribir.

En la actualidad sin embargo, parece absurdo pensar en no contar por lo menos con la ayuda de computadoras para facilitar el trabajo de las empresas, las suites de oficina son las preferidas en los negocios y conforme evolucionan los negocios van incorporando nuevas tecnologías para facilitar o agilitar las labores que realizan y de esta manera producir más, y elevar su valor comercial.

La perspectiva de valoración de un negocio ha cambiado, en la actualidad no se toman en cuenta ya únicamente los activos o el patrimonio de la empresa, existe un elemento intangible mucho más valioso y ese es el nivel de servicio que brinda una empresa. La forma de hacer negocios se ha vuelto global, para una empresa el tener un sitio web corporativo extiende su presencia a Internet. Con ello, se logra visibildad a nivel mundial y las posibilidades de prestar servicios y ofrecer productos se amplían. A un nivel

básico el sólo hecho de tener un sitio web no agrega valor a un negocio, pero la forma de responder a la demanda de servicios y productos que puedan surgir por esta vía sí lo hace.

La siguiente sección brinda una vista más de cercana a las capacidades que las RIAs pueden ofrecer y la manera en que pueden ayudar a agregar valor a un negocio.

Como se describe en la sección anterior, el obtener la fidelidad de los clientes no es fácil, se necesita de estrategias complejas y obviamente de productos que brinden alta calidad para que las empresas puedan resaltar en el mercado en que se desarrollan. Las RIAs no solo que proveen de una solución a un problema momentáneo sino que plantean un escenario que pueda crecer en el tiempo. Bien planteada una RIA representa una solución escalable.

El representar una solución escalable implica que las necesidades de la empresa en relación al uso de recursos por uso de tecnología se optimizan, de manera que se bajan costos por producción de nuevas soluciones. El problema que arrastran muchas empresas en el tiempo es la poca capacidad que tienen de respuesta al seguir utilizando sistemas legacy (heredados) que utilizan conceptos de desarrollo un tanto obsoletos.

La tendencia por mejorar los negocios administrativamente hace que se estructuren los procesos de negocio de mejor manera, esto ayuda a identificar puntos en los que la empresa tiene debilidades pues no son tareas que agregan valor a la actividad empresarial. Como por ejemplo, los departamentos de archivo con los que cuentan casi todas las empresas grandes, que no son otra cosa que oficinas llenas de papeles.

Las RIAs permiten utilizar lo mejor de los recursos de TI con los que se cuenta hoy por hoy, mientras brinda una solución sin precedentes para las tecnologías que las empresas puedan adoptar en el futuro. Una RIA desarrollada de manera efectiva tiene la capacidad de extender su valor a sistemas SOA y la información, activo invaluable de una empresa, se adaptará a futuros desarrollos. Con esto es posible acceder a información desde cualquier sistema de manera rápida y fácil utilizando la web como medio de comunicación.

Las capacidades de una RIA se extienden más allá del manejo y presentación agradable de los datos, representan herramientas que pueden elevarse a niveles complejos de interacción permitiendo inclusive a los empleados o clientes de una empresa compartir conocimiento e interactuar. Al permitir esto, una RIA facilita el trabajo en equipo inclusive si éstos no se encuentran ubicados geográficamente en el mismo sitio.

Este conjunto de características hacen que sea posible para los negocios tener mayor flexibilidad ante los cambios, cosa que en el mercado sucede con frecuencia, ya que el mismo hecho de tener una tecnología que evoluciona constantemente hace que los hábitos de la gente también cambien. Cuando un negocio es flexible a los cambios puede enfrentar las crisis que puedan surgir en el mercado de manera solvente, esto de por sí ya representa un valor agregado importante.

4.5 Ventajas en el uso de Aplicaciones Enriquecidas de Internet (RIA)

En el presente documento se pueden encontrar con creces las números as virtudes de implementar una RIA. ¿Pero una Aplicación Enriquecida de Internet funciona para todos los casos? En la mayoría de veces si lo hace,

porque una RIA desarrollada correctamente y que interpreta y se basa en resolver los problemas de la lógica de negocio presenta las siguientes características:

- **Simplifican tareas complejas**. Los usuarios tienen poca capacidad de atención para tareas complejas, sobre todo cuando al estar acostumbrados al uso de la web mediante el uso de una búsqueda simple pueden obtener la información que desean encontrar.

  Eliminar la molestia causada por transacciones complejas en la web como tratar de descifrar como funciona un sistema o que información debe ser provista al mismo para obtener una respuesta, ayuda a crear un flujo de trabajo óptimo. Para RIAs orientadas a consumidores, esto se traduce en un crecimiento en las ventas; mientras en RIAs orientadas a empresas se traduce en incremento en los niveles de productividad.
- **Reducen el costo transaccional.** Cuando un cliente o usuario no puede completar una tarea en línea, toma el teléfono y se comunica con un usuario experto o un agente en un call center. Esto representa un costo de dólares por llamada frente a centavos por transacción en la web. Inclusive los formularios HTML tardan en completarse un tiempo mayor al que toma completar un formulario en una RIA esto ocurre por las rondas de ida y vuelta que tiene un formulario HTML con el servidor deja al usuario en espera de que la transacción se complete (el tiempo representa dinero).

  Por otra parte, en un ambiente empresarial las RIAs permiten la integración de diferentes fuentes de datos en una misma interfaz de usuario, reduciendo significativamente el tiempo de trabajo.
- **Aumentan las ventas en línea y otros canales de distribución.** Las tasas de abandono también llamados tasas de rebote que se dan al

utilizar un carro de compras en línea, son tan altas como las que se registran en formularios HTML de varias páginas, esto dismuye drásticamente en una RIA de una sola página. Bajas tasas de rebote se traducen en un incremento en las ventas. Inclusive si las ventas se completan a través de otros canales, por ejemplo mediante el uso de una RIA al realizar una venta por teléfono. El tiempo de respuesta de la aplicación ayuda a predisponer al cliente prospecto a comprar.

Las Aplicaciones Enriquecidas de Internet combinan los beneficios del acceso Web instantáneo a datos centralizados, con una experiencia de usuario que se plantea a través de una interfaz de usuario fácil de usar y atractiva para ver. Sin embargo, aunque estos argumentos parezcan suficientes para implementar este tipo de aplicaciones, no significa que sean siempre una buena inversión. Para obtener el mayor provecho de una RIA es primordial evaluar primero el modelo de negocio para entender los factores clave de diseño que asegurarán el éxito al implementarlas.

CAPÍTULO 5

5 Conclusiones y Recomendaciones

5.1 Conclusiones

El objeto principal de este estudio fue realizar una comparación entre tres tecnologías: Adobe Flex, ASP .NET AJAX y Curl, estos frameworks tienen importantes aspectos de innovación que representan una ayuda para los programadores al momento de plantear un proyecto de desarrollo de una RIA. Sin embargo, para considerar el empleo de uno de estos frameworks es importante en primer lugar tomar en cuenta cual va a ser la finalidad de la aplicación y las necesidades que va a cubrir.

Adobe Flex brinda un soporte ampliado para trabajar con contenidos multimedia, lo que brinda características especiales a una RIA para cubrir necesidades de personalización de interfaces y potenciar aplicaciones que necesitan darle libertad al usuario de manejar elementos de la interfaz de usuario; la ejecución de la aplicación se hace posible en el navegador pero es necesario instalar un componente adicional que sirve de interprete para cargar la aplicación. Para la ejecución de la aplicación el componente debe instalarse en el navegador del cliente, pero en realidad eso no representa un impedimento de desempeño pues en la actualidad este componente para ejecutar aplicaciones de Adobe Flex se instala en un proceso de actualización del navegador bastante sencillo.

Por otra parte ASP .NET AJAX, permite manipular y crear aplicaciones de datos con características de manejo de interfaz bastante bien, pero donde

muestra una mayor estabilidad es al momento de crear aplicaciones que necesiten el manejo de transacciones de datos. Ahora, para que sea posible desarrollar una aplicación confiable se deben tomar en cuenta que la manipulación de datos desde servicios web y el nivel de seguridad que se vaya a implementar desde el servidor se evalúe correctamente para que la aplicación no sea vulnerable a ataques informáticos. Las aplicaciones desarrolladas bajo este framework no necesitan de ningún componente adicional para funcionar, lo que es ideal en ambientes de red en donde existan restricciones de seguridad para los usuarios.

Finalmente está el framework Curl, brinda tiempos bajos de respuesta y una velocidad de carga de la aplicación bastante buena; le brinda a la aplicación un desempeño bastante aceptable cuando se trata de manipular grandes cantidades de información, esto es bastante útil para aplicaciones empresariales que manejan volúmenes grandes de datos. Para que una aplicación desarrollada bajo este framework se ejecute es necesario instalar un componente adicional en el cliente como sucede en el caso de Adobe Flex. La diferencia aquí radica en que el componente de Curl debe ser descargado e instalado en el cliente antes de ser instalada la aplicación, el componente a instalarse no es popular en Internet y es más bien de un uso exclusivo de quienes desarrollan aplicaciones bajo este framework.

Por otra parte, a medida que fuí obteniendo información de cada uno de los frameworks y entendiendo el modelo de desarrollo RIA, he identificado puntos que deben ser considerados como primarios al momento de empezar un proyecto de este tipo. El desarrollo de una RIA puede representar una solución a largo plazo cuando se tienen bien claros los requerimientos que debe llenar y el panorama que rodea el desarrollo de este tipo de aplicaciones. El describir de manera adecuada los procesos de negocio no

es lo único en lo que un equipo de desarrollo se debe enfocar al momento de emprender un proyecto de este tipo, la elección del framework debe complementar y llenar los requisitos que tienen los usuarios que van a utilizar la aplicación.

Entonces puedo concluir que el ciclo de desarrollo de una RIA requiere que se acoplen diferentes procesos en diferentes aspectos de la aplicación. Como resultado, el diseño, la planificación y la administración de un proyecto llegarán a tener cierto grado de riesgo, pues de no cumplir con los objetivos planteados, estos procesos incidirán en el costo final de desarrollo de la aplicación. Como en todo sistema de producción, cuantas más partes en movimiento, existen más oportunidades de que se produzcan fallas.

Para mitigar cualquier problema que pueda desembocar en un proyecto riesgoso es necesario guardar un balance en la administración del mismo. Si bien la fortaleza de una RIA radica en la mejora sustancial en el manejo de la aplicación por el usuario, lo que significa prestarle especial atención al desarrollo de la aplicación del lado del cliente; el desempeño de la aplicación tiene en gran medida que ver con lo que suceda con la transmisión de datos, la forma en que se consuman servicios web y por sobre todo, la manera en que se interpreten los procesos de negocio para generar los diferentes componentes que intervienen en el funcionamiento de una RIA.

A medida que pasa el tiempo se presentan nuevas formas de obtener soluciones tecnológicas, se desarrollan nuevos dispositivos y nuevas tecnologías emergen alrededor de estos dispositivos para brindar soluciones que estén al alcance de todos los usuarios. Las RIAs establecen un punto de partida para el desarrollo de la tecnología en la web y se presentan como

una solución al problema que en su momento puede surgir para las empresas con las aplicaciones heredadas. Con la ventaja de permitir crear SaaS representan además una solución que en el tiempo puede extenderse tanto como las necesidades de los usuarios y las empresas así lo requieran.

Las tecnologías estudiadas permiten desarrollar este tipo de aplicaciones, lo conveniente es estudiar muy bien la finalidad de lo que se quiere lograr para utilizar el framework adecuado que se adapte al desarrollo de cada necesidad.

5.2 Recomendaciones

Una vez que se tienen presentados los resultados, y diferentes aspectos para tomar en cuenta al momento de desarrollar una RIA han sido expuestos; me gustaría resaltar el hecho de que si bien en primera instancia la idea era realizar una comparación entre frameworks que permitan desarrollar aplicaciones RIA y resaltar la importancia de la experiencia de los usuarios en el contexto del desarrollo de este tipo de aplicaciones; a medida que se fueron involucrando más elementos podemos darnos cuenta que el desarrollo de una RIA no es tan sencillo como muchas veces plantean los proveedores de software de desarrollo.

Es importante tomar en cuenta que el universo de componentes y la forma creativa de integrar estos componentes no determina el éxito de una aplicación. Si esta aplicación no está respaldada por la infraestructura adecuada, si no se definen de manera adecuada los diferentes niveles lógicos que forman parte de la misma, si no se da importancia al contexto del servidor y a las medidas de seguridad que deben implementarse en torno a la aplicación y el desarrollo en el cliente, si no contemplamos la

encriptación de la información; entonces vamos a toparnos con más de un problema que podría llevarnos a pensar que las RIAs no son una solución adecuada.

Otro punto que considero fuerte es el de tener la idea clara de lo que se quiere lograr con la aplicación, trabajar con prototipos es una buena práctica. Siempre resultará mucho más fácil cambiar prototipos de diseño del front-end de la aplicación final que codificar nuevos cambios. El trabajar con prototipos de diseño ayudará a los usuarios a visualizar el problema de mejor manera, tendrán una perspectiva diferente de lo que desean lograr con la aplicación y surgiran nuevas ideas para mejorar la experiencia del usuario.

Cuando se logre tener claro esto, llevar a cabo el desarrollo en cualquier framework permitirá cumplir los objetivos que se planteen. No debemos olvidarnos que el desarrollo de una RIA implica dedicarle mucho tiempo a potenciar la aplicación del lado del cliente y que si al final del trabajo eso no se ha logrado se deberán replantear las cosas para aprovechar al máximo las capacidades que pueden llegar a tener este tipo de aplicaciones.

Debemos enfocar el desarrollo de nuestras soluciones en realizar aplicaciones intuitivas. Sólo de esa manera lograremos el objetivo de las RIAs, enriquecer la experiencia de los usuarios al utilizar una aplicación en la web.

ANEXOS

1 Comunicado del CERN

En el siguiente comunicado el CERN hace de uso público la tecnología del World Wide Web.

930430

ORGANISATION EUROPEENNE POUR LA RECHERCHE NUCLEAIRE
CERN EUROPEAN ORGANIZATION FOR NUCLEAR RESEARCH

STATEMENT CONCERNING CERN W3 SOFTWARE RELEASE INTO PUBLIC DOMAIN

TO WHOM IT MAY CONCERN

Introduction

The World Wide Web, hereafter referred to as W3, is a global computer networked information system.

The W3 project provides a collaborative information system independent of hardware and software platform, and physical location. The project spans technical design notes, documentation, news, discussion, educational material, personal notes, publicity, bulletin boards, live status information and numerical data as a uniform continuum, seamlessly intergated with similar information in other disciplines.

The information is presented to the user as a web of interlinked documents .

Acces to information through W3 is:

- via a hypertext model;
- network based, world wide;
- information format independent;
- highly platform/operating system independent;
- scalable from local notes to distributed data bases.

Webs can be independent, subsets or supersets of each other. They can be local, regional or worldwide. The documents available on a web may reside on any computer supported by that web.

...

Declaration

The following CERN software is hereby put into the public domain:

- W 3 basic ("line-mode") client
- W 3 basic server
- W 3 library of common code.

CERN's intention in this is to further compatibility, common practices, and standards in networking and computer supported collaboration. This does not constitute a precedent to be applied to any other CERN copyright software.

CERN relinquishes all intellectual property rights to this code, both source and binary form and permission is granted for anyone to use, duplicate, modify and redistribute it.

CERN provides absolutely NO WARRANTY OF ANY KIND with respect to this software. The entire risk as to the quality and performance of this software is with the user. IN NO EVENT WILL CERN BE LIABLE TO ANYONE FOR ANY DAMAGES ARISING OUT THE USE OF THIS SOFTWARE, INCLUDING, WITHOUT LIMITATION, DAMAGES RESULTING FROM LOST DATA OR LOST PROFITS, OR FOR ANY SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES.

Geneva, 30 April 1993

W. Hoogland
Director of Research

H. Weber
Director of Administration

ORGANISATION EUROPEENNE POUR LA RECHERCHE NUCLEAIRE
DIVISION DES FINANCES
CERN

2 Código fuente de los componentes

2.1 Componente Data Grid

2.1.1 *Código fuente en Adobe Flex - Data Grid*

```
<mx:DataGrid id="dtgOrderDetails" x="10" y="10"
height="180" variableRowHeight="true" editable="true"
width="609" >
<mx:columns>
      <mx:DataGridColumn width="70"
headerText="Sequence" dataField="Sequence"/>
      <mx:DataGridColumn width="220"
headerText="Product" dataField="ProductName"
editable="false"/>
      <mx:DataGridColumn headerText="Quantity"
dataField="Quantity" />
      <mx:DataGridColumn headerText="Price\Unit"
dataField="UnitPrice" editable="false"/>
      <mx:DataGridColumn headerText="Price"
dataField="Price" editable="false"/>
      </mx:columns>
</mx:DataGrid>
public function treeChanged(event:Event):void
{
var orderAr:Array = Order.source;
var len:Number = orderAr.length
var orderDetailsAr = orderAr.slice(0,len);
OrderDetails.source = orderDetailsAr;
OrderDetails.filterFunction=stateFilterFunc;
OrderDetails.refresh();
dtgOrderDetails.dataProvider = OrderDetails;
dtgOrderDetails.selectedIndex = 0;
}
public function stateFilterFunc(item:Object):Boolean
{
var ordId = cbxOrderData.text;
return item.OrderID == ordId ;
}
```

2.1.2 *Código fuente en ASP .NET AJAX - Data Grid*

```
<asp:UpdatePanel runat="server" ID="UpdtOrderHdr">
<asp:GridView id="GrdOrderLine" runat="server"
ForeColor="Black" Width="100%" Font-Size="X-Small" Font-
Names="Verdana"
OnSelectedIndexChanged="GrdOrderLine_SelectedIndexChange
d" Height="30%" AutoGenerateColumns="False"
AutoGenerateSelectButton="True" AllowPaging="True"
BorderStyle="Solid" BorderColor="Black"
__designer:wfdid="w107">
</asp:GridView>
</asp:UpdatePanel>
<asp:GridView id="GrdOrderLine" runat="server"
ForeColor="Black" Width="100%" Font-Size="X-Small" Font-
Names="Verdana"
OnSelectedIndexChanged="GrdOrderLine_SelectedIndexChange
d" Height="30%" AllowPaging="True" BorderColor="Black"
BorderStyle="Solid" AutoGenerateSelectButton="True"
__designer:wfdid="w222"
OnRowCreated="GrdOrderLine_RowCreated">
<EmptyDataRowStyle Wrap="False"></EmptyDataRowStyle>
<RowStyle BorderStyle="Solid"
BorderColor="Black"></RowStyle>
<SelectedRowStyle
BackColor="IndianRed"></SelectedRowStyle>
<HeaderStyle BackColor="Control" BorderColor="Brown"
Font-Bold="True"></HeaderStyle>
<AlternatingRowStyle BackColor="Tan" BorderStyle="Solid"
BorderColor="Black"></AlternatingRowStyle>
</asp:GridView>
private void LoadGrid(int OrderID)
{
DataSet dsOrderLines = new DataSet();
DataRow[] drOrderLine;
dsOrderLines = GetOrderDetails();
string qryOrdLines = "OrderID like '" + OrderID + "'";
drOrderLine =
dsOrderLines.Tables[0].Select(qryOrdLines);
DataSet dsGrid = new DataSet();
```

```
DataTable TblGrid = dsGrid.Tables.Add("OrderLines");
DataColumn dcOrderID =
dsGrid.Tables[0].Columns.Add("OrderID", typeof(int));
dsGrid.Tables[0].Columns.Add("Sequence", typeof(int));
dsGrid.Tables[0].Columns.Add("CustomerName",
typeof(string));
dsGrid.Tables[0].Columns.Add("OrderDate",
typeof(DateTime));
dsGrid.Tables[0].Columns.Add("ProductName",
typeof(string));
dsGrid.Tables[0].Columns.Add("UnitPrice",
typeof(float));
dsGrid.Tables[0].Columns.Add("Quantity", typeof(int));
dsGrid.Tables[0].Columns.Add("Price", typeof(float));
foreach (DataRow dr in drOrderLine)
{
      DataRow drGrid = dsGrid.Tables[0].NewRow();
      drGrid["OrderID"] = dr["OrderID"];
      drGrid["Sequence"] = dr["Sequence"];
      drGrid["CustomerName"] = dr["CustomerName"];
      drGrid["OrderDate"] = dr["OrderDate"];
      drGrid["ProductName"] = dr["ProductName"];
      drGrid["UnitPrice"] = dr["UnitPrice"];
      drGrid["Quantity"] = dr["Quantity"];
      drGrid["Price"] = dr["Price"];
      dsGrid.Tables[0].Rows.Add(drGrid);
}
GrdOrderLine.DataSource = dsGrid;
GrdOrderLine.DataBind();
DataRow[] drSeq;
drSeq = dsGrid.Tables[0].Select("Sequence =
max(Sequence)");
if (drSeq.Length > 0)
{
      Session["Sequence"] =
drSeq[0]["Sequence"].ToString();
}
}
```

2.1.3 *Código fuente en Curl - Data Grid*

```
{let rg:RecordGrid =
{RecordGrid
      record-source = order-details
      automatic-columns? = false,
      editable? = false,
      display-filler-column? = true,
      multiple-selection-enabled? = false,
      select-current-record? = true,
      {RecordGridColumn "Sequence"},
      {RecordGridColumn "ProductName",header-spec =
"Product"},
      {RecordGridColumn "Quantity"},
      {RecordGridColumn
            "UnitPrice",
            header-spec = "Price/Unit",
            format-spec =
            {proc {data:any, r:Record}:String
            {return "$" &{String r["UnitPrice"]}}}
      },
      {RecordGridColumn
            "Price",
            format-spec =
            {proc {data:any, r:Record}:String
            {return "$" &{String r["Price"]}}}
      },
      {on e:CurrentRecordChanged at rg:RecordGrid do
      {refresh-edit}}
}
}
{define-proc {update-rg }:void
{set order-details.filter = {RecordData OrderID =
{orderid.to-int}}}
{set order-details.sort = "Sequence" }
}
```

2.2 Componente para gráficos estádisticos de barras

2.2.1 *Código fuente en Adobe Flex - Gráfico de barras*

```
<dotnetCHARTING:Chart id="ChrtBar" runat="server" Font-
Size="X-Small" Font-Names="Verdana">
<DefaultLegendBox CornerBottomRight="Cut" Padding="4">
<DefaultEntry ShapeType="None"></DefaultEntry>
<HeaderEntry ShapeType="None"
Visible="False"></HeaderEntry>
</DefaultLegendBox>
<SmartForecast Start=""></SmartForecast>
<DefaultElement ShapeType="None">
<DefaultSubValue Name=""></DefaultSubValue>
<LegendEntry ShapeType="None"></LegendEntry>
</DefaultElement>
<ChartArea StartDateOfYear="">
<Label Font="Tahoma, 8pt"></Label>
<TitleBox Position="Left">
<Label Color="Black"></Label>
</TitleBox>
<DefaultElement ShapeType="None">
<DefaultSubValue Name="">
<Line Length="4"></Line>
</DefaultSubValue>
<LegendEntry ShapeType="None"></LegendEntry>
</DefaultElement>
<LegendBox CornerBottomRight="Cut" Position="Top"
Padding="4">
<DefaultEntry ShapeType="None"></DefaultEntry>
<HeaderEntry Value="Value" ShapeType="None"
Visible="False" Name="Name" SortOrder="-
1"></HeaderEntry>
</LegendBox>
<XAxis SmartScaleBreakLimit="2" GaugeNeedleType="One"
GaugeLabelMode="Default">
<TimeScaleLabels MaximumRangeRows="4"></TimeScaleLabels>
<ScaleBreakLine Color="Gray"></ScaleBreakLine>
<ZeroTick>
```

```
<Line Length="3"></Line>
</ZeroTick>
<MinorTimeIntervalAdvanced
Start=""></MinorTimeIntervalAdvanced>
<Label LineAlignment="Center" Alignment="Center"
Font="Arial, 9pt, style=Bold"></Label>
<TimeIntervalAdvanced Start=""></TimeIntervalAdvanced>
<DefaultTick>
<Line Length="3"></Line>
<Label Text="%Value"></Label>
</DefaultTick>
</XAxis>
<YAxis SmartScaleBreakLimit="2" GaugeNeedleType="One"
GaugeLabelMode="Default">
<TimeScaleLabels MaximumRangeRows="4"></TimeScaleLabels>
<ScaleBreakLine Color="Gray"></ScaleBreakLine>
<ZeroTick>
<Line Length="3"></Line>
</ZeroTick>
<MinorTimeIntervalAdvanced
Start=""></MinorTimeIntervalAdvanced>
<Label LineAlignment="Center" Alignment="Center"
Font="Arial, 9pt, style=Bold"></Label>
<TimeIntervalAdvanced Start=""></TimeIntervalAdvanced>
<DefaultTick>
<Line Length="3"></Line>
<Label Text="%Value"></Label>
</DefaultTick>
</YAxis>
</ChartArea>
<TitleBox Position="Left">
<Label Color="Black"></Label>
</TitleBox>
</dotnetCHARTING:Chart>
private void BuildBarChart(string sStartDt, string
sEndDt)
{
DataTable dtBarChart;
dtBarChart = BuildBarChartDataSet(sStartDt, sEndDt);
```

```
ChrtBar.Title = "CHART";
ChrtBar.ChartArea.XAxis.Label.Text = "DATE";
ChrtBar.ChartArea.YAxis.Label.Text = "PRICE in $";
ChrtBar.TempDirectory = "temp";
ChrtBar.Debug = true;
ChrtBar.Series.Name = "Product";
ChrtBar.Series.Data = dtBarChart;
ChrtBar.SeriesCollection.Add();
}
private DataTable BuildBarChartDataSet(string
StartDt,string EndDate)
{
dtBarChart = new DataTable();
DataColumn dcDate = dtBarChart.Columns.Add("Date",
typeof(string));
dtBarChart.Columns.Add("Price", typeof(string));
dtBarChart.Columns.Add("Product", typeof(string));
DataTable dtOrdDet = new DataTable("OrderDetails");
dtOrdDet.Columns.Add("OrderID");
dtOrdDet.Columns.Add("Sequence");
dtOrdDet.Columns.Add("CustomerName");
dtOrdDet.Columns.Add("OrderDate");
dtOrdDet.Columns.Add("ProductName");
dtOrdDet.Columns.Add("UnitPrice");
dtOrdDet.Columns.Add("Quantity");
dtOrdDet.Columns.Add("Price");
DateTime varDate= Convert.ToDateTime(StartDt);
DateTime varEndDate = Convert.ToDateTime(EndDate);
while (varDate <= varEndDate)
{
DataRow[] drOrdrDet;
string sVarDate = varDate.ToString("yyyy-MM-dd")+
StartDt.Substring(10);
drOrdrDet = GetAllOrdersOnDate(sVarDate);
foreach(DataRow dr in drOrdrDet)
{
dtOrdDet.ImportRow(dr);
}
Hashtable hProdName = new Hashtable();
foreach (DataRow dr in drOrdrDet)
```

```
{
String sprodName = dr["ProductName"].ToString();
String qryProd="ProductName = '" + sprodName + "'";
DataRow[] drProd;
if (hProdName.Count > 0)
{
if (hProdName.ContainsKey(sprodName) == false)
{
hProdName.Add(sprodName, dr["Sequence"].ToString());
drProd = dtOrdDet.Select(qryProd);
dPrice = 0;
foreach (DataRow drP in drProd)
{
dTemp = Convert.ToDecimal(drP["Price"]);
dPrice = dPrice + dTemp;
}
drbarChart = dtBarChart.NewRow();
drbarChart["Date"] = sVarDate;
drbarChart["Price"] =  dPrice;
drbarChart["Product"] = sprodName;
dtBarChart.Rows.Add(drbarChart);
}
}
else
{
hProdName.Add(sprodName, dr["Sequence"].ToString());
drProd=dtOrdDet.Select(qryProd);
dPrice = 0;
foreach (DataRow drP in drProd)
{
dTemp =Convert.ToDecimal(drP["Price"]);
dPrice = dPrice +dTemp ;
}
drbarChart = dtBarChart.NewRow();
drbarChart["Date"] = sVarDate;
drbarChart["Price"] = dPrice;
drbarChart["Product"] = sprodName;
dtBarChart.Rows.Add(drbarChart);
}
}
```

```
varDate = varDate.AddDays(1);
}
return dtBarChart;
}
private DataRow[] GetAllOrdersOnDate(string OrderDate)
{
dsOrderDet.Clear();
dsOrderDet = GetOrderDetails();
Session["OrderDet"] = dsOrderDet;
DataSet dstemp = new DataSet();
dstemp = GetOrderDetails();
DataRow[] drOrdrDetOnDate;
string qry = "OrderDate LIKE '" + OrderDate + "%'" ;
drOrdrDetOnDate = dsOrderDet.Tables[0].Select(qry);
return drOrdrDetOnDate;
}
```

2.2.2 *Código fuente en ASP .NET AJAX - Gráfico de barras*

```
<mx:Panel id="pnlChart" width="400" height="277"
layout="absolute" title="Chart"
doubleClickEnabled="true"
doubleClick="enlargeChart()" borderColor="#C18E7D">
<mx:ColumnChart itemClick="myHandler(event)"
showDataTips="true" x="0" y="0" id="columnChart"
width="300" height="100%" maxColumnWidth="20">
<mx:horizontalAxis>
<mx:CategoryAxis id="catAxis" categoryField="Day"
dataProvider="" />
</mx:horizontalAxis>
</mx:ColumnChart>
<mx:Canvas x="298" y="0" width="82" height="237"
id="chCanvas">
<mx:Legend dataProvider="{columnChart}" x="0" y="0"
width="82" height="118"  id="legend"/>
<mx:Canvas x="0" y="161" width="82" height="76"
id="lgCanvas">
<mx:DateField x="2" y="53" id="ToDate"
change="genXML('0')" editable="true"
formatString="YYYY-MM-DD" width="70" height="23"/>
<mx:DateField x="2" y="19" id="FrmDate"
editable="true" formatString="YYYY-MM-DD"
width="70" height="21"/>
<mx:Label x="2" y="37" text="To:"/>
<mx:Label x="0" y="1" text="From:"/>
</mx:Canvas>
</mx:Canvas>
</mx:Panel>
{
var flag:String = CtlStr;
orderSales.removeAll();
var pName:String =
Product.getItemAt(0).ProductName.toString();
var ordId:String =
dtgDataOrderDetails.selectedItem.OrderID.toString();
```

```
var check:String = "0";
if(dtgDataOrderDetails.selectedIndex > -1)
{
var index:Number = dtgDataOrderDetails.selectedIndex;
var noDays:Number;
var date:String;
var dPick:Date;
if(flag == "1")
{
date =
dtgDataOrderDetails.selectedItem.OrderDate.toString();
var dtPick:Date = stringToDate(date);
dPick = new Date(dtPick.getTime() + (2 *
millisecondsPerDay));
var strDtPick:String = (dPick.getFullYear()+"-
"+(dPick.getMonth()+1)+"-"+dPick.getDate());
ToDate.text = strDtPick;
FrmDate.text = date;
noDays = 3;
}else
{
date = FrmDate.text;
var nyeve_date = stringToDate(ToDate.text);
var now_date = stringToDate(FrmDate.text);
var nDiffDays = Math.floor((nyeve_date -
now_date)/86400000);
noDays = nDiffDays + 1;
}
var d:Date = stringToDate(date);
var proName:String;
var Total:String;
var x:XML = <day></day>
for(var j:Number=0;j<noDays;j++)
{
var x1:XML = <days></days>
var year:String = d.getFullYear().toString();
var month1:Number = d.getMonth()+1;
var month:String=month1.toString();
var day1:String = d.getDate().toString();
if(month.length == 1)
```

```
month = "0"+month;
if(day1.length == 1)
day1 = "0"+day1;
var strDate:String = (year+"-"+month+"-"+day1);
var dt:Date = stringToDate(strDate);
var dateXml:String = "<Day>"+strDate+"</Day>";
var day:XML = new XML(dateXml);
x1 = x1.appendChild(day);
for(var i:Number=0;i<SumOrder.length;i++)
{
var dDate:String =
SumOrder.getItemAt(i).OrderDate.toString();
var dgDate:Date = stringToDate(dDate);
if(dgDate.toString() == d.toString())
{
proName = SumOrder.getItemAt(i).ProductName.toString();
Total = SumOrder.getItemAt(i).Total.toString();
var myPattern:RegExp = / /g;
proName = proName.replace(myPattern, "_");
var newPattern:RegExp = /'/g;
proName = proName.replace(newPattern, "_");
var str:String = "<"+proName+">"+Total+"</"+proName+">";
var x2:XML = new XML(str);
x1 = x1.appendChild(x2);
if(pName == proName)
check = "1";
}
}
if(check == "0")
{
var pNamestr:String = "<"+pName+">0</"+pName+">";

var x3:XML = new XML(pNamestr);
x1 = x1.appendChild(x3);
}
x = x.appendChild(x1);

d = new Date(d.getTime() + (1 * millisecondsPerDay));
}
orderSales = new XMLListCollection(XMLList(x.days));
```

```
columnChart.dataProvider = orderSales;
catAxis.dataProvider = orderSales;
dtgDataOrderDetails.selectedIndex = index;
var cs:ColumnSeries= new ColumnSeries();
columnChart.series = [cs]; // associate the array
for( var n:Number=0; n<Product.length;n++)
{
cs = new ColumnSeries();
var proId:String;
proId = Product.getItemAt(n).ProductName.toString();
cs.displayName = proId;
cs.yField = proId;
cs.xField="Day";
columnChart.series[n] = cs;
}
}
}
private function sumOrderlines():void
{
var date:String;
var nextDate:String;
var proName:String;
var nextProName:String;
var total:Number;
var nextTotal:Number;
var flag:Number = 0;
var List:Object = new Object();
var k:Number=0;
SumOrder.removeAll();
for (var i:Number=0;i<Order.length;i++)
{
date = Order.getItemAt(i).OrderDate.toString();
proName = Order.getItemAt(i).ProductName.toString();
total = Order.getItemAt(i).Price.toString();
var asKey:String = proName+"_"+date;
if(List[asKey] == null)
{
List[asKey] = [[k],[total]];
SumOrder.addItem({OrderDate:date, ProductName: proName,
Total: total});
```

```
k=k+1;
} else
{
var str:String = List[asKey];
var len:Number = str.length;
var com:Number = str.indexOf(",");
var value:String =str.substring(0,com);
var valTotal:String = str.substring(com +1,len);
var index:Number = Number(value);
var numTotal:Number = Number(valTotal);
total = total + numTotal;
SumOrder.setItemAt({OrderDate:date, ProductName:
proName, Total: total},index);
var tot:String = total.toString();
List[asKey] = [[value],[tot]];
}
}
}
```

2.2.3 *Código fuente en Curl - Gráfico de barras*

```
{let bar-chart:Chart =
{LayeredChart
left-axis-parent = {ShapeGroup axis-label = "Price"},
{BarLayer rs-chart,
{splice fs},
stacking-mode = ChartStackingMode.none,
cursor = cursor-hand,
x-axis-data = {ChartDataSeries rs-chart, "OrderDate"}
}
}
}
{define-proc {create-new-rs date-val:String,
n:Time}:void
let dt:DateTime = {DateTime date-val}
let rv:RecordView =
{RecordView order-details,
filter = {RecordFilter
{proc {rec:Record}:bool
let rec-date:DateTime =
rec["OrderDate"] asa DateTime
{return
rec-date >= dt and
rec-date <= dt + n
}
}
},
sort = "OrderDate"
}
{rs-chart.delete-all}
let d:RecordData = {RecordData OrderDate = ""}
{if rv.size > 0 then
{for r in rv do
let datedt:DateTime = r["OrderDate"] asa DateTime
let date:String = datedt.info.locale-date
let price:double = r["Price"] asa double
let product:String = r["ProductName"] asa String
```

```
{if d["OrderDate"] == date then
{if {d.key-exists? product} then
set d[product] = price + d[product] asa double
else
set d[product] = price}
else
{if d["OrderDate"] != "" then
{rs-chart.append d}}
set d = {RecordData OrderDate = date}}
}
{rs-chart.append d}
}
}
```

2.3 Componente para obtener un reporte con datos tabulados

2.3.1 *Código fuente en Adobe Flex - Reporte tabulado*

```
<mx:Panel id="pnlReport" width="400" height="277"
layout="absolute" title="Report" x="399"
doubleClickEnabled="true" doubleClick="enlargeReport()"
borderColor="#C18E7D">
<mx:VBox x="0" y="1" width="100%" height="100%">
<mx:HBox width="360">
<mx:Label text="Order ID :" fontWeight="bold"/>
<mx:Label id="ChLblOrder" text="{}" width="100%"/>
<mx:Label text="Order Date :" fontWeight="bold"/>
<mx:Label id="ChLblDate" text="{}" width="100%"/>
</mx:HBox>
<mx:HBox width="361">
<mx:Label text="Customer Name :" fontWeight="bold" />
<mx:Label id="ChLblCustomer" text="{}"/>
</mx:HBox>
<mx:DataGrid id="dtgReport" height="100%" width="100%"
borderThickness="0" borderStyle="none" editable="false"
alternatingItemColors="[#ffffff, #ffffff]"
borderColor="#ffffff" themeColor="#ffffff"
enabled="false">
<mx:columns>
<mx:DataGridColumn headerText="Sequence"
dataField="Sequence"/>
<mx:DataGridColumn headerText="Product"
dataField="ProductName"/>
<mx:DataGridColumn headerText="Quantity"
dataField="Quantity"/>
<mx:DataGridColumn headerText="Price/Unit"
dataField="Price"/>
<mx:DataGridColumn headerText="Total Price"
dataField="Total"/>
</mx:columns>
</mx:DataGrid>
<mx:HBox width="361">
<mx:Label text=" "  width="100%"/>
```

```
<mx:Label text="Total :" fontWeight="bold"
width="100%"/>
<mx:Label id="DataTotal" width="51"/>
</mx:HBox>
</mx:VBox>
</mx:Panel>
public var ReportData:ArrayCollection = new
ArrayCollection();
private function report():void
{
var ordId:String =
dtgDataOrderDetails.selectedItem.OrderID.toString();
var ordDate:String =
dtgDataOrderDetails.selectedItem.OrderDate.toString();
var custName:String =
dtgDataOrderDetails.selectedItem.CustomerName.toString()
;
var totalPrice:Number = 0;
ChLblOrder.text = ordId;
ChLblDate.text = ordDate;
ChLblCustomer.text = custName;
ReportData.removeAll();
var index:Number = dtgDataOrderDetails.selectedIndex;
for (var i:Number=0;i<Order.length;i++)
{
dtgDataOrderDetails.selectedIndex = i;
if(dtgDataOrderDetails.selectedItem.OrderID.toString()==
ordId)
{
var seqNo:String =
dtgDataOrderDetails.selectedItem.Sequence.toString();
var prdName:String =
dtgDataOrderDetails.selectedItem.ProductName.toString();
var qty:String =
dtgDataOrderDetails.selectedItem.Quantity.toString();
var price:String =
dtgDataOrderDetails.selectedItem.UnitPrice.toString();
var total:String =
dtgDataOrderDetails.selectedItem.Total.toString();
```

```
ReportData.addItem({Sequence:seqNo, ProductName:
prdName, Quantity:qty,Price:price, Total: total});

totalPrice = totalPrice + Number(total);
}
}
dtgReport.dataProvider = ReportData;
DataTotal.text = totalPrice.toString();
}
```

2.3.2 *Código fuente en ASP .NET AJAX - Reporte tabulado*

```
<asp:Table id="TblReport" runat="server" Font-Size="X-
Small" Font-Names="verdana" BorderColor="Black"
BorderStyle="Solid" GridLines="Both">
<asp:TableHeaderRow runat="server" ID="tblHeader">
<asp:TableHeaderCell runat="server" Text="#" Font-
Bold="True" ForeColor="Black"></asp:TableHeaderCell>
<asp:TableHeaderCell runat="server" Text="Product" Font-
Bold="True" ForeColor="Black"></asp:TableHeaderCell>
<asp:TableHeaderCell runat="server" Text="Unit Price"
Font-Bold="True"
ForeColor="Black"></asp:TableHeaderCell>
<asp:TableHeaderCell runat="server" Width="100px"
Text="Quantity" Font-Bold="True"
ForeColor="Black"></asp:TableHeaderCell>
<asp:TableHeaderCell runat="server" Text="Price" Font-
Bold="True" ForeColor="Black"></asp:TableHeaderCell>
</asp:TableHeaderRow>
</asp:Table>
private void BuildReport(int sOrderID)
{
DataRow[] drOrderDetSel;
string sQryOrdrID = "OrderID like '" + sOrderID + "'";
drOrderDetSel =
dsOrderDet.Tables["OrderDetails"].Select(sQryOrdrID);
int iRwCnt=0;
Hashtable hDup = new Hashtable();
foreach (DataRow dr in drOrderDetSel)
{
TableRow tRow = new TableRow();
if (!(hDup.ContainsKey(dr["Sequence"])))
{
hDup.Add(dr["Sequence"], dr["Sequence"]);
for (int i =0;i<=dr.ItemArray.Length -1;i++)
{
if (i == 0)
{
TableCell tCell = new TableCell();
```

```
tCell.Text = iRwCnt.ToString();
tRow.Cells.Add(tCell);
iRwCnt = iRwCnt + 1;
LblDate.Text =
Convert.ToDateTime(dr["OrderDate"]).ToString("MM-dd-
yyyy");
}
if (i == 4 || i == 5 || i == 6 || i == 7)
{
TableCell tCell = new TableCell();
tCell.Text = dr[i].ToString();
tCell.HorizontalAlign = HorizontalAlign.Center;
tRow.Cells.Add(tCell);

if (i == 7)
{
dTotal = dTotal + Convert.ToDecimal(dr[i]);
}
}
}
TblReport.Rows.Add(tRow);
}
}
LblOrderName.Text = "Order : " +
Convert.ToString(Session["OrderID"]);
LblCustName.Text = CustName;
LblTotal.Text = "Total : " + dTotal.ToString();
}
```

2.3.3 *Código fuente en Curl - Reporte tabulado*

```
{let report-table1:Table =
{Table
border-color = "black",
border-width = 1pt,
background = "#C1C1C1",
internal-borders = "groups"
}
}
{let report-table2:Table =
{Table
border-color = "black",
border-width = 1pt,
internal-borders = "groups"
}
}
{define-proc {create-report val:String, tab1:Table,
tab2:Table}:void
{tab1.clear}
let rec:#Record =
{orders1.select-one filter = {RecordData OrderID =
{val.to-int}}}
{if-non-null rec then
let dt:DateTime = {DateTime {rec.get "OrderDate"}}
let str:String = "Order : " & {String {rec.get
"OrderID"}}
{tab1.add
{row-prototype
str,  {Fill}, dt.info.locale-date}
}
{tab1.add
{row-prototype {String {rec.get "CustomerName"}}}
}
}
{tab2.clear}
{tab2.add
{row-prototype
" " ,"#", "Product", "Quantity", "UnitPrice", "Total"}
```

```
}
let recs:{Array-of Record} =
{order-details1.select filter = {RecordData OrderID =
{val.to-int}}}
{let gt:double = 0}
{let rc:int = 0}
{if recs.size > 0 then
{for r:Record in recs do
set rc = rc + 1
let q:String = {String r["Quantity"]}
let up:String = {String r["UnitPrice"]}
let tot:double = {q.to-double} * {up.to-double}
set gt = gt + tot
let g:Graphic = {tab2.add {String r["Sequence"]},row =
rc,column = 1}
set g = {tab2.add {String r["ProductName"]},row =
rc,column = 2}
set g = {tab2.add {String r["Quantity"]},row = rc,column
= 3}
set g = {tab2.add "$" & {String r["UnitPrice"]},row =
rc,column = 4}
set g = {tab2.add "$" & tot, row = rc,column = 5}
}
let g1:Graphic = {tab2.add "Total",row = rc + 1,column =
4}
set g1 = {tab2.add "$" & gt,row = rc + 1,column = 5}
{set {tab2.get-row 0}.group = TableGroup.header}
{set {tab2.get-column 5}.group = TableGroup.header}
{set {tab2.get-row rc+1}.group = TableGroup.footer}
}
}
```

2.4 Componente para acceder a un servicio web

2.4.1 *Código fuente en Adobe Flex - Acceso a un servicio web*

```
<mx:HTTPService id="OrderDetailsData"
result="Order=ArrayCollection(OrderDetailsData.lastResul
t.NewDataSet.OrderDetails)"/>
public var Order:ArrayCollection = new
ArrayCollection();
TreeOrderDetails.url="http://203.200.7.42/RestfulWebserv
ice/Orders.aspx"
OrderDetailsData.send();
```

2.4.2 *Código fuente en ASP .NET AJAX - Acceso a un servicio web*

```
private DataSet GetOrderDetails()
{
HttpWebRequest hrst = WebRequest.Create("http://
203.200.7.42/AjaxRestFulWebService/orderdetails.aspx")
as HttpWebRequest;
hrst.Method = "GET";
System.IO.StreamReader responseReader = new
System.IO.StreamReader(hrst.GetResponse().GetResponseStr
eam());
dsOrderDet.ReadXml(responseReader);
responseReader.Close();
Session["OrderDet"] = dsOrderDet;
return dsOrderDet;
}
```

2.4.3 *Código fuente en Curl - Acceso a un servicio web*

```
{define-proc {create-orderdetails-
recordset}:TestRecordSet
let service-url:Url = {url ”http://203.200.7.42” &
"OrderDetails.aspx"}
{with-open-streams
in = {read-open {service-url}}
do
let rs:TestRecordSet = {TestRecordSet order-details-rf}
let p:SAXParser = {SAXParser}
{p.set-content-handler {RecordSetHandler rs,
"OrderDetails" }}
let source:InputSource = {InputSource character-stream =
in}
{with rs.batch-events? = true do
{p.parse source}}
{return rs}
}
}
```

2.5 Componente para generar XML a partir de una llamada HTTP Post

2.5.1 *Código fuente en Adobe Flex - XML a partir de una llamada POST HTTP*

```
<mx:HTTPService contentType="application/xml"
method="POST" id="UpdateOrderLine" />
UpdateOrderLine.url="http://198.0.102.2/FlexRestFulWebSe
rvice/orderDetails.aspx"
private function updateRow():void
{
if(UpdateOrderRow.length > 0)
{
var x:XML = <NewDataSet></NewDataSet>;
for(var i:Number=0;i<UpdateOrderRow.length;i++)
{
var ordId:String =
UpdateOrderRow.getItemAt(i).orderId.toString();
var Seq:String =
UpdateOrderRow.getItemAt(i).Sequence.toString();
var prodId:String =
UpdateOrderRow.getItemAt(i).ProductName.toString();
var UnitPrice:String =
UpdateOrderRow.getItemAt(i).UnitPrice.toString();
var Qty:String =
UpdateOrderRow.getItemAt(i).Quantity.toString();
var proId:String = ProductAr[prodId];
var cusNam:String = lblCustomerName.text;
var cusId:String = Customer[cusNam];
var x1:XML = <OrderDetails></OrderDetails>;
var idXml:String =
"<OrderID><![CDATA["+ordId+"]]></OrderID>";
var id:XML = new XML(idXml);
x1 = x1.appendChild(id);
var seqXml:String =
"<OrderLine><![CDATA["+Seq+"]]></OrderLine>";
var seq:XML = new XML(seqXml);
```

```
x1 = x1.appendChild(seq);
var proIdXml:String =
"<ProductID><![CDATA["+proId+"]]></ProductID>";
var proId1:XML = new XML(proIdXml);
x1 = x1.appendChild(proId1);
var priceXml:String =
"<UnitPrice><![CDATA["+UnitPrice+"]]></UnitPrice>";
var price:XML = new XML(priceXml);
x1 = x1.appendChild(price);
var qtyXml:String =
"<Quantity><![CDATA["+Qty+"]]></Quantity>";
var qty:XML = new XML(qtyXml);
x1 = x1.appendChild(qty);
x = x.appendChild(x1);
}
var params:Object = new Object();
params.TRAN = 'UPD';
UpdateOrderLine.headers = params;
UpdateOrderLine.request = x ;
UpdateOrderLine.send();
UpdateOrderRow.removeAll();
}
}
```

2.5.2 *Código fuente en ASP .NET AJAX - XML a partir de una llamada POST HTTP*

```
string SOrdersXMLFile;
SOrdersXMLFile = "<NewDataSet>";
SOrdersXMLFile = SOrdersXMLFile + "<Orders>";
SOrdersXMLFile = SOrdersXMLFile + "<CustomerID>" +
drCustID[0]["CustomerID"].ToString() + "</CustomerID>";
SOrdersXMLFile = SOrdersXMLFile + "<OrderStatus>" +
DropDownList1.Text + "</OrderStatus>";
SOrdersXMLFile = SOrdersXMLFile + "<OrderId>" +
Convert.ToString(Session["OrderID"]) + "</OrderId>";
SOrdersXMLFile = SOrdersXMLFile + "<OrderDate>" +
TxtDate.Text + "</OrderDate>";
SOrdersXMLFile = SOrdersXMLFile + "</Orders>";
SOrdersXMLFile = SOrdersXMLFile + "</NewDataSet>";
ASCIIEncoding encoding = new ASCIIEncoding();
byte[] data = encoding.GetBytes(SOrdersXMLFile);
HttpWebRequest myRequest =
(HttpWebRequest)WebRequest.Create("http://198.0.102.2/Re
stFulWebService/orders.aspx");
myRequest.Method = "POST";
myRequest.Headers.Add("TRAN", "UPD");
myRequest.ContentType = "application/x-www-form-
urlencoded";
myRequest.ContentLength = data.Length;
Stream newStream = myRequest.GetRequestStream();
newStream.Write(data, 0, data.Length);
newStream.Close();
```

2.5.3 *Código fuente en Curl - XML a partir de una llamada POST HTTP*

```
{define-proc public {update-orderdetails-to-db }:void
{for r in order-details do
let state:RecordState = r.state
{switch state
case
RecordState.modified
do
let doc:XDMDocument = {orderline-update-xml r,
"OrderDetails"}
let xml:String = {doc.to-XML}
let loc:StringBuf = {StringBuf web-url &
"OrderDetails.aspx?TRAN=UPD"}
{post-XML xml, loc}
}
}
}
{define-proc {orderline-update-xml
rec:Record,
table-name:XDMName,
root-name:XDMName = "NewDataSet"}:XDMDocument
let xs:XDMElement = {XDMElement root-name}
let x:XDMElement = {XDMElement table-name}
{for f in order-details.fields do
{switch f.name
case "OrderID","UnitPrice","Quantity"
do
let k:String = f.name
{x.append {XDMElement k, {f.domain.format rec[k]}}}
case "Sequence"
do
{x.append {XDMElement "OrderLine", {f.domain.format
rec["Sequence"]}}}
case "ProductName"
do
let p:String = ""
{for re in product-details do
```

```
{if re["ProductName"] == {f.domain.format
rec["ProductName"]} then
{set p = {String re["ProductID"]}}
{break}}
}
let k:String = "ProductID"
{x.append {XDMElement k, p}}
}
}
{xs.append x}
{return {XDMDocument xs}}
}
```

3 Resultados comparativos obtenidos

Ambiente de desarrollo utilizado para pruebas

Características del Servidor

CPU: 2,53 GHz Intel Core 2 Duo

RAM: 8 Gb

Sistema Operativo: Windows 2003 SP1 64 bits

Servidores Web: IIS 6.0

Características del Cliente

CPU: 2,53 GHz Intel Core 2 Duo

RAM: 8 GB

Sistema Operativo: Windows 7 Ultimate SP1 64 bits

Navegadores

Internet Explorer 10.0.9200.16576

FireFox 21.0

Chrome Versión 27.0.1453.93

Características de los Frameworks

Adobe Flex

Flash Player 9.0.28.0

Flex Builder 2

Flex Charting Services 2

ASP.NET (v.2.0) AJAX

AJAX extension : v1.0

IDE: Microsoft Visual Studio 2005

Curl

RTE: 5.0.3

IDE: 5.0.1003

Database

Microsoft SQL Server 2005

La escala de calificación va determinada por números del 1 al 10. En la escala de calificaciones se considera *1 - poco satisfactorio* y *10 - muy satisfactorio.*

3.1 Facilidad de Aprendizaje

FACTORES EVALUADOS	ADOBE FLEX	ASP.NET AJAX	CURL
Disponibilidad de herramientas de auto aprendizaje	9	8	6
Disponibilidad de códificación de ejemplo	8	8	8
Disponibilidad de aplicaciones de ejemplo	8	9	8
Soporte de proveedor para evaluación	8	9	8
Grupos / foros en línea de desarrolladores	9	10	8
Utilización de habilidades de POO	9	7	9
Tiempo para aprender el lenguaje	7	9	8
PUNTAJE GLOBAL OBTENIDO	8.3	8.6	7.9

Tabla 2 - Resultados Facilidad de Aprendizaje

Fuente: Realizado por el autor en base a información de http://www.sonata-software.com

Observaciones:

- Todas las tecnologías proporcionan documentación usable. El buscador de la utilidad de Curl, el Documentation Viewer, no admite varios términos al buscar. ASP.NET AJAX no ofrece amplia documentación sobre los extensores, pero proporciona información útil a través del Panel de Actualización.

- Todos los proveedores de las tecnologías brindan útiles ejemplos.
- ASP.NET AJAX está provisto de una aplicación de muestra a partir de donde es posible reutilizar el código.
- Tanto y Adobe Flex como Curl ponen limitaciones en el uso de la versión de evaluación. ASP.NET no es limitante.
- Todos los productos cuentan con una activa comunidad de desarrolladores. En términos de número de usuarios la comunidad de desarrolladores más grande es la de ASP.NET AJAX seguido de la comunidad de usuarios de Adobe Flex y por última la comunidad de usuarios de Curl.
- Las tres tecnologías añaden a sus características de desarrollo la programación orientada a objetos.

3.2 Facilidad en Diseño

FACTORES EVALUADOS	ADOBE FLEX	ASP.NET AJAX	CURL
Disponibilidad de IDE	9	9	9
Biblioteca inteligente de widgets de interfaz de usuario	8	7	9
Capacidad de diseño Drag'n'Drop	9	9	9
Capacidad de crear widgets de interfaz de usuario	7	9	7
Diseño de estilo declarativo para reducir codificación	8	8	8
PUNTAJE GLOBAL OBTENIDO	8.2	8.4	8.6

Tabla 3 - Resultados Facilidad de Diseño

Fuente: Realizado por el autor en base a información de http://www.sonata-software.com

Observaciones:

- Todas las plataformas ofrecen excelentes IDE.
- Para la aplicación desarrollada AJAX no proporciona el control de gráficos. Flex no tenía controles de informes especializados, como tablas, set de gráfico circular concéntrico ni una herramienta para crear comentarios auxiliares (tool tips).
- Todas las plataformas proporcionan Drag'n'Drop para el diseño de interfaz de usuario.
- Todas las tecnologías soportan la personalización de controles existentes. ASP.NET AJAX tiene una gran cantidad de soporte de para controles producidos por terceros.
- CURL soporta múltiples paradigmas de programación.

3.3 Facilidad en Desarrollo

FACTORES EVALUADOS	ADOBE FLEX	ASP.NET AJAX	CURL
Disponibilidad de IDE para desarrollo y prototipado rápido	9	9	9
IDE soporta Intelli-Sense	9	9	9
Verificación de errores en línea	8	8	8
Soporte para depuración	9	9	9
Orientación a Objetos	9	9	7
Soporte para perfiles de aplicación	5	8	5
Minimizar el esfuerzo para desarrollo	8	9	7
Generación de líneas de código en implementación	8	9	7
PUNTAJE GLOBAL OBTENIDO	8.1	7.6	8.8

Tabla 4 - Resultados Facilidad de Desarrollo

Fuente: Realizado por el autor en base a información de http://www.sonata-software.com

Observaciones:

- Todas las plataformas proporcionan soporte para el optimizar la velocidad de diseño, prototipado y desarrollo.
- Todos los IDEs proporcionan retroalimentación después de realizar la compilación.
- Todos los IDEs permiten ubicar breakpoint, evaluación de expresiones regulares, es decir brindan soporte para depuración de errores.

3.4 Desempeño en tiempo de ejecución

FACTORES EVALUADOS	ADOBE FLEX	ASP.NET AJAX	CURL
Facilidad de instalación	9	8	9
Ocupa poco espacio en descarga de la aplicación	8	6	10
Tiempo de inicio de la aplicación	8	8	9
Tiempo de reinicio de la aplicación	8	6	9
Soporte para navegadores	10	7	10
Optimización en peticiones al servidor	9	7	9
Utilización de memoria	5	9	9
PUNTAJE GLOBAL OBTENIDO	8.1	7.3	9.1

Tabla 5 - Resultados Desempeño en Tiempo de Ejecución

Fuente: Realizado por el autor en base a información de http://www.sonata-software.com

DICCIONARIO DE DATOS

- **AJAX,** *abr. de Asynchronous JavaScript and XML*, conjunto de tecnologías que permiten la comunicación asíncrona entre lado cliente y el servidor de una aplicación web.
- **API,** *abr. de Application Programming Interface*, especificación que define la forma en que los componentes de una aplicación interactúan entre sí.
- **B2B,** *abr. de Business-To-Business*, término relacionado a las transacciones comerciales que se efectúan entre dos tipos de negocios.
- **B2C,** *abr. de Business-To-Consumer*, término relacionado a las transacciones comerciales que se efectúan entre un negocio y el consumidor final de un producto.
- **CRM,** *abr. de Customer Relationship Management*, proceso implementado en las empresas para gestionar las acciones de una empresa con sus clientes o prospectos comerciales.
- **CRUD,** *abr. de Create, Read, Update and Delete*, son las operaciones básicas que se efectúan en almacenamiento persistente en una base de datos.
- **CSRF,** *abr. de Cross-Site Request Forgery,* ataque informático que busca obtener información sensible de un cliente web al explotar la confianza que tiene el usuario en un sitio web determinado.
- **CSS,** *abr. de Cascade Style Sheet,* lenguaje utilizado para describir la presentación semántica de manera flexible de los elementos que se presentan en una aplicación o sitio web.
- **DHTML,** *abr. de Dinamic HTML,* conjunto de técnicas que permiten crear aplicaciones y sitios web interactivos.
- **DOM,** *abr. Document Object Model,* proporciona un conjunto estándar de objetos para representar documentos HTML y XML.

- **ECMASCRIPT,** define un lenguaje de tipos dinámicos ligeramente inspirado en Java y otros lenguajes del estilo de C. Soporta algunas características de la programación orientada a objetos mediante objetos basados en prototipos y pseudoclases.
- **ERP,** *abr. de Enterprise Resource Planning,* son sistemas de gestión de información que automatizan muchas de las prácticas de negocio asociadas con los aspectos operativos o productivos de una empresa.
- **EVA,** *abr. de Editor Visual Avanzado,* característica de una aplicación que permite la edición visual de una aplicación.
- **GUI,** *abr. de Graphical User Interface,* representación visual de los elementos que forman parte de una aplicación y que permiten la interacción del usuario con ésta.
- **HIPERTEXTO,** es un concepto mediante el cual es posible relacionar información de diversos tipos: texto, gráfico, video y sonido.
- **HTML,** *abr. de HyperText Markup Language,* hace referencia al lenguaje de marcado por etiquetas que hace posible la creación de documentos publicables en la web e interpretados por un navegador.
- **HTTP,** *abr. de Hypertext Transfer Protocol,* protocolo de transferencia de información en Internet.
- **IDE,** *abr. de Integrated Development Environment*, provee un ambiente de trabajo para que los desarrolladores puedan trabajar con un lenguaje de programación.
- **INTELLI-SENSE,** es una característica que poseen algunas IDEs que facilitan el uso y aprendizaje de un lenguaje de programación al presentar listas de comandos, instrucciones, variables, etc. según el usuario escribe código.
- **ISO,** *abr. de International Organization for Standardization,* organismo encargado de promover el desarrollo de normas internacionales de fabricación de productos y servicio, comercio y comunicación para todas las ramas industriales a excepción de la eléctrica y la electrónica.

- **JSON,** *abr. de JavaScript Object Notation,* formato para intercambio de datos entre aplicaciones web, es una alternativa a XML.
- **LOOK AND FEEL,** término relacionado al aspecto y comportamiento de una GUI.
- **MASHUP,** puede ser una página web o aplicación que usa y combina datos, presentaciones y funcionalidad procedentes de una o más fuentes para crear nuevos servicios.
- **MULTIMEDIA,** término relacionado para referir cualquier objeto o sistema que utiliza múltiples medios de expresión físicos o digitales para presentar o comunicar información.
- **MVC,** *abr. de Model Vista Controlador***,** es un patrón de arquitectura de software que separa los datos y la lógica de negocio de una aplicación de la interfaz de usuario y el módulo encargado de gestionar los eventos y las comunicaciones.
- **NCSA,** *abr. de National Center for Supercomputing Applications,* Es un organismo estadounidense relacionado con la investigación en el campo de la Informática y las Telecomunicaciones.
- **OSI,** *abr. de Open System Interconnection*, hace referencia al modelo de interconexión de sistemas abiertos (ISO/IEC 7498-1).
- **POO,** *abr. de Programación Orientada a Objetos,* es un modelo de programación que usa objetos en sus interacciones, para diseñar aplicaciones y programas informáticos.
- **REST,** *abr. de Representational State Transfer,* es una técnica de arquitectura software para sistemas hipermedia distribuidos como la World Wide Web.
- **RIA,** *abr. de Rich Internet Application,* son aplicaciones web que tienen la mayoría de las características de las aplicaciones de escritorio tradicionales.
- **ROI,** *abr. de Return On investment,* es una razón financiera que compara el beneficio o la utilidad obtenida en relación a una inversión realizada.

- **RSP,** *abr. de Rich Server Platform,* es una propuesta de la fundación Eclipse para el desarrollo de plataforma que permita ejecutar aplicaciones RIA.
- **RTE,** *abr. de Runtime Engine,* es un entorno de software que provee los servicios necesarios para que una aplicación se ejecute.
- **SAAS,** *abr. de Software as a Service,* es un modelo de distribución de software donde el soporte lógico y los datos que maneja se alojan en servidores dedicados a soportar servicios de tecnologías de la información.
- **SCRIPT,** es un modelo de programación que permite la ejecución de comandos interpretados para desarrollar componentes aislados.
- **SEO,** *abr. de Search Engine Optimization*, es el proceso que se realiza para mejorar el posicionamiento de un sitio web ante un motor de búsqueda.
- **SERVICIO WEB,** es una tecnología que utiliza un conjunto de protocolos y estándares que sirven para intercambiar datos entre aplicaciones.
- **SISTEMAS LEGACY,** son sistemas informático que con el transcurso del tiempo y avances tecnológicos se vuelven anticuados pero que continúan siendo utilizados por los usuarios de una organización.
- **SOAP,** *abr. de Simple Object Access Protocol,* es un protocolo estándar que define la comunicación de diferentes procesos por medio de intercambio de datos XML.
- **SOCKET,** el término se usa para designar un concepto abstracto por el cual dos aplicaciones pueden intercambiar cualquier flujo de datos, generalmente de manera fiable y ordenada.
- **SQL,** *abr. de Structured Query Language,* es un lenguaje declarativo de acceso a bases de datos relacionales que permite especificar diversos tipos de operaciones en ellas.
- **SSL,** *abr. de Secure Sockets Layer,* son protocolos criptográficos que proporcionan comunicaciones seguras por una red, comúnmente Internet.

- **TCP/IP,** *abr. de Transmission Control Protocol / Internet Protocol,* describe un conjunto de guías generales de diseño e implementación de protocolos de red específicos para permitir que un equipo pueda comunicarse en una red.
- **URI,** *abr. de Uniform Resource Identifier,* es una cadena de caracteres corta que identifica inequívocamente un recurso: servicio, página, documento, dirección de correo electrónico, enciclopedia, etc.
- **WYSIWYG,** *abr. de What You See Is What You Get,* se aplica a editores de contenidos que permiten editar un documento visualizando el resultado final conforme se realizan cambios.
- **XHTML,** *abr. de eXtensible HyperText Markup Language*, es básicamente HTML expresado como XML válido.
- **XML,** *abr. de eXtensible Markup Language,* es un lenguaje que permite describir elementos de datos por medio de etiquetas, fue desarrollado por el World Wide Web Consortium (W3C).
- **XSS,** *abr. de Cross-site Scripting***,** es un tipo de vulnerabilidad que por medio de comandos en sitios cruzados busca vulnerabilidades en aplicaciones web.

BIBLIOGRAFÍA

- Eco, Humberto, *Cómo se hace una tesis,* Barcelona, Gedisa, 1995.
- Berners-Lee, Tim, *Tejiendo la Red*, SIGLO XXI EDITORES, mayo 2000.
- Booth, D., *Web Services Architecture*, W3C Working, Group Note 11 February 2004, www.w3.org/TR/ws-arch/
- Inkriti, *Java Based Rich Internet Applications: A white paper*, Framingham, Estados Unidos, www.inkriti.com
- Fain, Yakov; Dr. Rasputnis, Victor; Tartakovsky, Anatole; Stucki, Ben; Eckel , Bruce; Chotin, Matt, *Rich Internet Applications with Adobe Flex & Java Secrets of the Masters,* New Jersey, SYS-CON Books, 2007.
- Halstead, Robert H.; Hostetter, Mat; Kranz, David A., *Curl: A Content Language for the Web*, Cambridge, 2005.
- Damle, N., *Curl Programming Bible*, Wiley, 2002.
- Malyshev , Stanislav, Rich Internet Applications with PHP, Zend Technologies, 10th March 2008.
- Byous, J, *Java technology: the early years*, Sun Developer Network, 1998.
- Hammond, Jeffrey S., *Securing Rich Internet Applications*, Forrester Research, 2008.
- Zimbra, *Open AJAX Alliance,* www.zimbra.com/community/open_ajax.html.
- Hammond, Jeffrey S. with Mike Gilpin, Chenxi Wang, Ph.D., and David D'Silva, *Securing Rich Internet Applications,* Estados Unidos.
- Financial Advisor Focus, *ROI for RIAs: The bottom-line impact of digital document management technology for independent Registered Investment Advisors,* Estados Unidos.
- Laserfiche*, Make your strategic move*, www.laserfiche.com/docs/pdf/ria.pdf

- White, Kevin, *The future of application virtualization,* 30th March 2006, www.cbronline.com/article_cbr.asp?guid=5191C403-22CD-47F4-98BBFBF22219C4E0
- *http://www.ajax.org*
- *http://www.adobe.com*
- *http://www.curl.com*
- *http://en.wikipedia.org/wiki/Bar_chart*
- *http://www.hcltech.com/blogs/engineering-and-rd-services/rich-internet-application-ria-%E2%80%93-game-changer-enterprise-applicatio*
- *http://www.openlaszlo.org/lps4.9/docs/reference/*
- *http://docs.oracle.com/javase/tutorial/deployment/doingMoreWithRIA/security.html*
- *http://www.derkeiler.com/Mailing-Lists/securityfocus/secprog/2001-07/0001.html*
- *http://www.upf.edu/hipertextnet/numero-9/ria-accesibilidad-web.html*
- *http://en.wikipedia.org/wiki/Data_grid*
- *http://flex.apache.org/dev-sourcecode.html*
- *http://www.asp.net/mvc/tutorials*
- *http://www.curl.com/download/tool/index.html#snt*
- *http://sourceforge.net/apps/trac/curl-sonntag*
- *http://developers.curlap.com/curl/docs/caede/en/docs/en/dguide/charts.html*
- *http://developers.curlap.com/curl/docs/caede/en/*
- *http://www.w3c.es/Divulgacion/GuiasBreves/*
- *http://www.w3.org/Protocols/rfc2616/rfc2616-sec9.html*
- *http://es.wikipedia.org/wiki/Uniform_Resource_Identifier*
- *http://www.adobe.com/resources/business/rich_internet_apps/getting_started/*
- *http://www.adobe.com/resources/business/rich_internet_apps/workflow/*
- *http://www.adobe.com/resources/business/rich_internet_apps/benefits/*

- *http://www.adobe.com/resources/business/rich_internet_apps/*
- *http://www.scpe.org/index.php/scpe/article/view/553*
- *http://catalystresources.com/ria*
- *http://www.manufacturing.net/articles/2009/09/5-benefits-of-rich-internet-applications-for-manufacturing-roi*
- *http://msdn.microsoft.com/en-us/library/ee658083.aspx*