5.1 INTRODUCCIÓN

La potenciación de la experiencia del usuario en aplicaciones Web, permite a la próxima generación de aplicaciones proveer interacciones enriquecidas y efectivas, demostrando que la experiencia importa y por tal motivo Adobe, esta por la convicción “de que grandes experiencias construyen grandes negocios” e impulsa a Flex un producto inicialmente de Macromedia y actual propietario Adobe, a entregar mejores experiencias de usuario a una fracción del esfuerzo y tiempo requerido por otras tecnologías de desarrollo.

Flex permite entregar aplicaciones que proporcionan una respuesta inmediata y un flujo de trabajo constante a los usuarios a través de componentes de interfaz de usuario basados en XML y un lenguaje de script orientado a objetos que maneja la lógica del cliente y el control de los procedimientos.
Analicemos 3 casos para determinar el repunte tecnológico:

1. Visitando www.apress.com este sitio funciona con HTML tradicional para buscar libros de diferentes autores como se muestra a continuación:

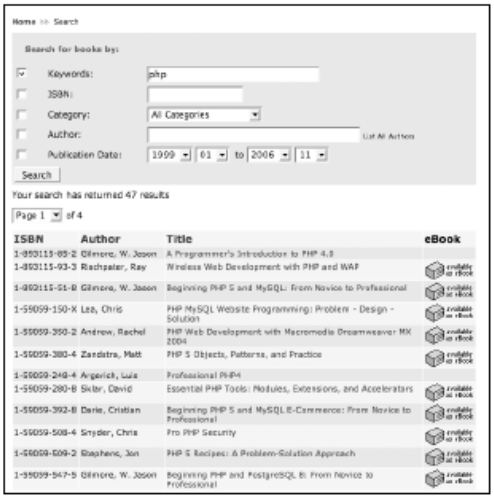

Figura 5.1 Resultado de la búsqueda en el sitio Apress

2. Ahora en Watergate Hotel y realice una reservación en https://reservations.ihotelier.com/onescreen.cfm?hotelid=2560&languageid=1 El sistema de reservación es integro con regiones que se actualizan en una

misma página, tomando la forma de una aplicación de escritorio, como se muestra a continuación:

Figura 5.2 Sistema de reservación Hotel Watergate

Finalmente, en http://flexapps.macromedia.com/flex15/flexstore/flexstore usted encuentra una Tienda virtual desarrollada en Flex y que permite arrastrar y soltar artículos en el carrito y se va procesando la compra en tiempo real, como se muestra a continuación:

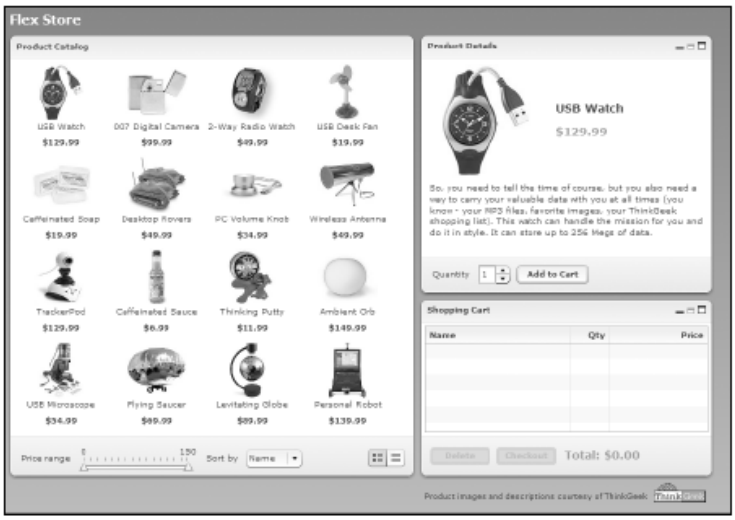

Figura 5.3 Tienda virtual Flex

Se eligió tres sitios por una razón. El primero, el sitio de Apress, era sitio web tradicional. El segundo usa Flash 8. El último se construyó en Flex y será motivo de nuestro estudio.

5.1.1 RIA ENFOQUE DE ADOBE

RIA son aplicaciones que ofrecen las experiencias de usuario más intuitivas, sensibles, y eficaces en Internet, estas combinan la experiencia de las aplicaciones de escritorio con la flexibilidad del despliegue de Internet, proporcionando un ambiente runtime (durante la ejecución) que traslada el procesamiento de la interfaz hacia el lado del cliente compilado las aplicaciones en el servidor.

La adopción creciente de tecnología RIA es un paso evolutivo que no reemplazará al HTML, más bien, extiende navegadores y dispositivos móviles continuando un papel crítico en la entrega de interfaces y navegación. Porque la tecnología RIA se despliega de forma consistente por diferentes plataformas Web y porque a través de sus elementos de la interfaz se puede apoyar gráficos en movimiento, video y audio bidireccional, las comunicaciones, y los formularios complejos, proporciona significativamente un ambiente más robusto en interfaces de aplicaciones Web. **[WWW045]**

Son obvias las diferencias de usar una aplicación en una URL y una aplicación local como Microsoft Word, la primera necesita reconstruir completamente las páginas entrando en un ciclo de petición y respuesta entre el cliente y el servidor, pero basta de este antiguo método, y Flex nos permite desarrollar aplicaciones que se actualizan en determinadas áreas con los datos pedidos en ese momento, teniendo respuestas más rápidas, peticiones disminuidas en los servidores, y tamaños de archivos más pequeños. **[LIB027]**

En un ambiente de RIA, los usuarios tienen la misma experiencia interactiva que en un ambiente de escritorio, recordemos el streaming de medios a través de **Flash Remoting**, este servidor encaminó a los ambientes de RIA con integración Java y .NET. Esto significó que Flash podría trabajar como una herramienta de presentación encima de una variedad de lenguajes de programación Web.

5.1.2 DISEÑO WEB Y FLEX

El software empresarial requiere mínimo tres capas, o estructuras que son:

- Presentación: Conjunto de elementos que el usuario observa y manipula.
- Lógica comercial: El código programado para determinar lo que debe o no realizarse en la aplicación, así como las conexiones a servidores y bases de datos.
- Datos: Consiste en la base de datos.

Flex está principalmente enfocado a la capa de Presentación. Su función primaria es crear interacción fluida en la interfaz del usuario, es por esto que Flex es también conocido como servidor de presentaciones.

5.2 ARQUITECTURA PARA RIA

Como se aprecia en la figura 5.4, Flex es una aplicación Java nativa (forma de archivo war) por esta razón el despliegue de la aplicación es la plataforma J2EE y puede interactuar con recursos de servidores como: IBM WebSphere, BEA WebLogic, Macromedia JRun, Oracle 10g Application Server, Fujitsu Interstage 6 (Japonés) y contendor de servlet Apache Tomcat. Adicionalmente tiene integridad con la plataforma .NET puesto que tiene un CLR (Common Language Runtime). **[WWW046]**

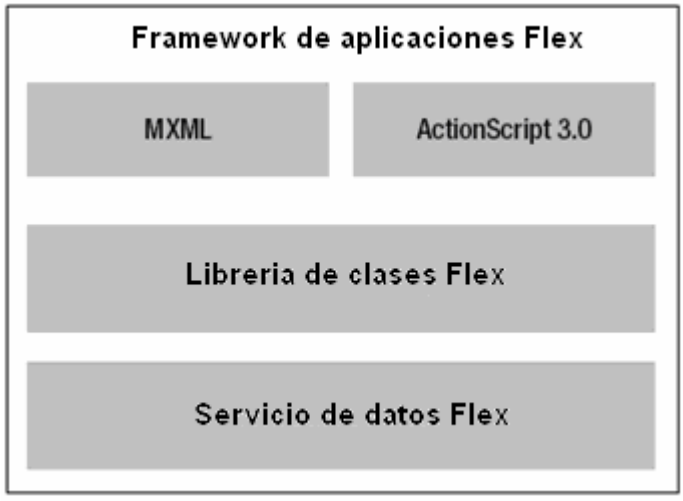

Figura 5.4 Arquitectura del framework Flex

Explicando las capas, en la cima tenemos MXML y ActionScript 3.0 que representarían al XHTML y al lenguaje script orientado a objetos respectivamente en una aplicación Web normal. Adicionalmente tenemos a Flash Player 9 que permitirá el acceso y presentación de la aplicación.
También Flex tiene una colección de clases y librerías para construir RIAs exitosas (incluyendo Flex Data Services-Servicios de datos, para la conexión de datos).

5.2.1 ESTRUCTURA DE LA APLICACIÓN

a. BIBLIOTECA DE CLASES

La biblioteca de clases contiene componentes predefinidos como: contenedores, controles, enlaces de datos, comportamientos y otros. La biblioteca usa el modelo de experiencia “Halo” para crear interfaces de usuario impactantes. Halo es un modelo de experiencia independiente de la plataforma RIA que ofrece un conjunto uniforme de señales visuales, patrones de interacción y convenciones de navegación de la aplicación, asegurando tener usabilidad en la interfaz. **[WWW047]**

Los desarrolladores pueden utilizar y extender los componentes a través de propiedades y métodos, también se puede definir su aspecto, las interacciones del usuario, la fuente y el tamaño de los componentes.

Los componentes admiten las siguientes características:

- **Eventos**: Acciones de la aplicación o del usuario que exigen la respuesta del componente.
- **Comportamientos**: Cambios visibles o audibles en un componente invocados por una acción. Los comportamientos permiten añadir con facilidad movimiento y sonido a las aplicaciones para darles a los usuarios un contexto para sus acciones. Por ejemplo, provocar que un cuadro de diálogo resalte un poco al recibir enfoque. Un comportamiento es la combinación de una acción junto con su efecto(cambio visible en el componente que ocurre en el transcurso del tiempo por ejemplo: atenuar, mover o pausar)

- **Skins**: Revestimientos que definen el aspecto del componente.
- **Estilos**: Conjunto de características, como el tamaño de fuente, alineación de texto.

La biblioteca proporciona dos tipos de componentes: los contenedores y los controles.

Un control es un componente de la interfaz de usuario que manejan las interacciones y muestran datos a ser manipulados a través de ese control, por ejemplo: DataGrid y TreeControl.

Todos los controles tienen las siguientes características:

- La API de MXML para declarar el control y los valores de sus propiedades y eventos.
- La API de ActionScript para llamar a los métodos del control y establecer sus propiedades y eventos durante la ejecución.
- Aspecto y carácter personalizables usando estilos, skins y fuentes.

Por otro lado tenemos los contenedores que definen una región de dibujo que controla la disposición para todo en el contenedor, incluyendo los otros contenedores y controles, por ejemplo: Form (formulario) para introducir datos, un Box (caja) y una Grid (cuadrícula).

b. FLEX DATA SERVICES

Flex permite tener integridad con diferentes bases de datos y **middleware** (lógica del negocio) a través de los Servicios de Datos que permiten conectar al código del servidor como Java, .NET, ColdFusion, PHP, ASP, entre otros.

Siempre tener presente no conectar directamente la aplicación con un servidor de base de datos ya que la lógica del negocio, hace la conexión con la base de datos, como se ilustra en la figura 5.5. Los componentes del Data Services de Flex se

conectan con el código y este determina lo que es, o no es, permisible (lógica comercial).

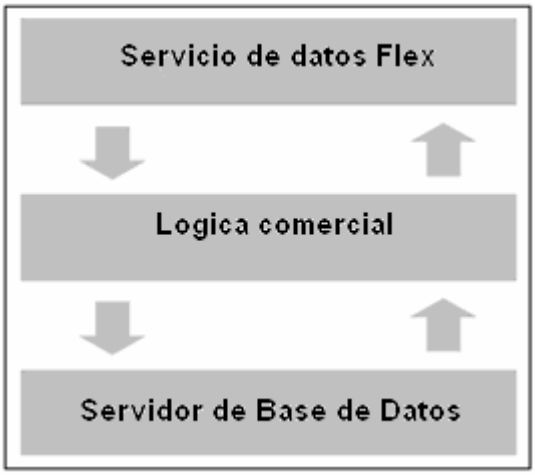

Figura 5.5 Flex y los Servicios de Datos

Además Flex tiene el Servicio de Administración de Datos que permite crear aplicaciones que trabajan con datos distribuidos. Este rasgo permite construir las aplicaciones con sincronización de datos y replicación de los mismos, pudiendo manejar colecciones grandes de datos con relaciones como uno-a-uno y uno-a-muchos.

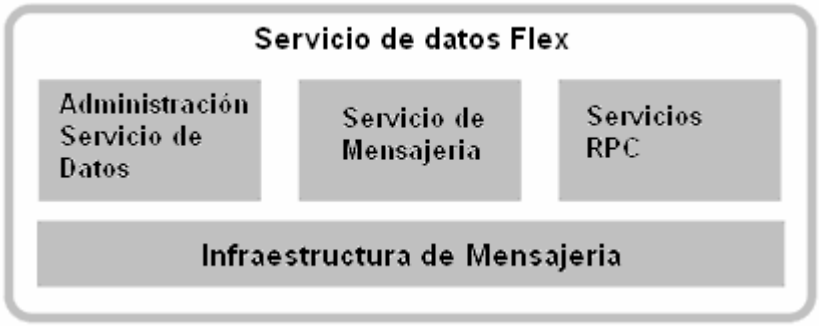

Figura 5.6 Flex y el Servicio de Administración de Datos

Como se muestra en la figura 5.6 la infraestructura de la mensajería de Flex nos permite definir el destino y origen de los mensajes de los componentes de la lógica del negocio con la aplicación de mensajería y esta integrado de:

1. El servicio de administración de datos.
2. El servicio de mensajería.
3. Servicios RPC.

c. FLEX CHARTING

Es una biblioteca de gráficos interactivos como: áreas, barra, circular, columnas, líneas, pastel, entre otros. Permite personalizar: colores, fuentes, y etiquetas mediante CSS, también nos permite realizar determinados efectos como arrastrar y soltar. Estos gráficos incorporan datos desde la base de datos a través del Servicio de Datos. Todos estos gráficos soportan propiedades que son dinámicas a los datos como dataProvider, DataTips o consejos. **[LIB027]**

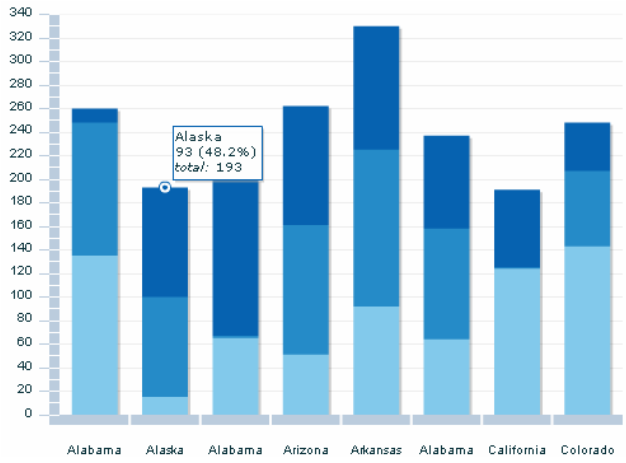

Figura 5.7 Ejemplo de Flex Charting

5.3 SERVIDOR DE PRESENTACIÓN

5.3.1 FLEX

Flex es una herramienta orientada al desarrollo de RIA, se compone de un servidor de presentación de aplicaciones Web, con código que se ejecuta en el cliente trasladando el procesamiento de la interfaz, entregando aplicaciones donde el HTML es inadecuado, por ejemplo, paneles de control de datos visuales enriquecidos, selección de productos en línea y procesos de pasos múltiples que involucran técnicas de visualización, respuesta inmediata al usuario y procesamiento local.

Flex aplica una metodología programática y un flujo de desarrollo declarativo basado en normas, junto con servicios runtime encaminados a desarrollar y desplegar los niveles de presentación dinámicos. Flex permite definir interfaces de usuario

dinámicas usando un lenguaje que se basa en XML, y que el servidor Flex transforma en aplicaciones cliente que se ejecutan en la máquina virtual Flash Player. **[WWW047]**

a. CARACTERISTICAS

1. El modelo de desarrollo resulta conocido para desarrolladores que usan JSP, ASP.NET, u otros lenguajes de script. El patrón fundamental es: crear un archivo de texto con el código fuente de la aplicación, este se despliega en el servidor y se compilará el código en una aplicación al recibir la primera solicitud HTTP, y las solicitudes subsiguientes servidas desde la memoria caché. En vez de emitir la aplicación en una serie de páginas HTML que incluyan los datos y la interfaz.
2. Flex se encamina a complementar el desarrollo Web de tecnologías existentes en los siguientes aspectos:
   - Aplicaciones empresariales para plataformas J2EE y .NET.
   - Uso de lenguajes existentes como Java, XML.
   - Servicios Web SOAP.
   - Enfoque estructurado a n-capas para la arquitectura e integración de una aplicación Web. Las aplicaciones Flex aumentan la capa de presentación, pero no cambia las capas del negocio y de integración. Esta capa es ejecutada dentro del servidor de aplicaciones, y proporciona la integración fácil mediante la influencia de código existente como acceso de objetos Java o XML. También integra la presentación con las tecnologías y frameworks existentes, como JSP y Struts.
   - Uso de patrones de diseño, como MVC.
3. Flex representa una nueva arquitectura de aplicación mezcla la flexibilidad de acceso a datos orientados a servicio **SAO** con el alcance superior y la efectividad de un cliente multiplataforma enriquecido.

El resultado son aplicaciones que son fáciles de construir y mantener, utilizando un menor ancho de banda una vez compilada la aplicación e integridad sobre la mayoría de sistemas operativos.

4. Los requerimientos en la interfaz de usuario se presentan en la siguiente tabla:

Requerimiento	**Elementos Web y experiencias**	**Beneficios del usuario final**
Procesos con varios pasos	Contexto de la Aplicación y flujo Transiciones Efectos Comportamientos	Enfoque mejorado Para proporcionarles a los usuarios pistas visuales para sus acciones y guiarlos a través del proceso.
Validación en el lado del cliente	Habilidad de realizar funciones sin requerir el servidor: Validación Formato Filtrado Ordenando	Regeneración mejorada Los clientes realizan acciones en la interfaz, sin peticiones al servidor.
Manipulación directa	Arrastrar y soltar Controles que no necesitan volver al servidor para adicional información: DataGrid Listbox Chechbox DateChooser	Interacción inmediata Permitiendo a los usuarios finales manipule la aplicación inmediatamente, por ejemplo, la capacidad de arrastrar y soltar.
La visualización de datos	Integración de datos Gráficas interactivas	Conexión a fuentes de datos múltiples vía una arquitectura orientada al servicio, combinada con los componentes de la interfaz de usuario.

Tabla 5.1 Requerimientos en la interfaz de usuario Flex

5.3.2 FLEX BUILDER

Adobe Flex Builder 2 es un entorno de desarrollo integrado basado en Eclipse para el desarrollo de aplicaciones de Internet sofisticadas (RIA) que combina la riqueza de aplicaciones de escritorio con el ámbito multiplataforma. Flex Builder permite a los desarrolladores construir rápidamente una lógica sofisticada para el cliente que se integra con XML, servicios Web o con Data Services gracias a las herramientas de diseño y de maquetación. **[WWW048]**

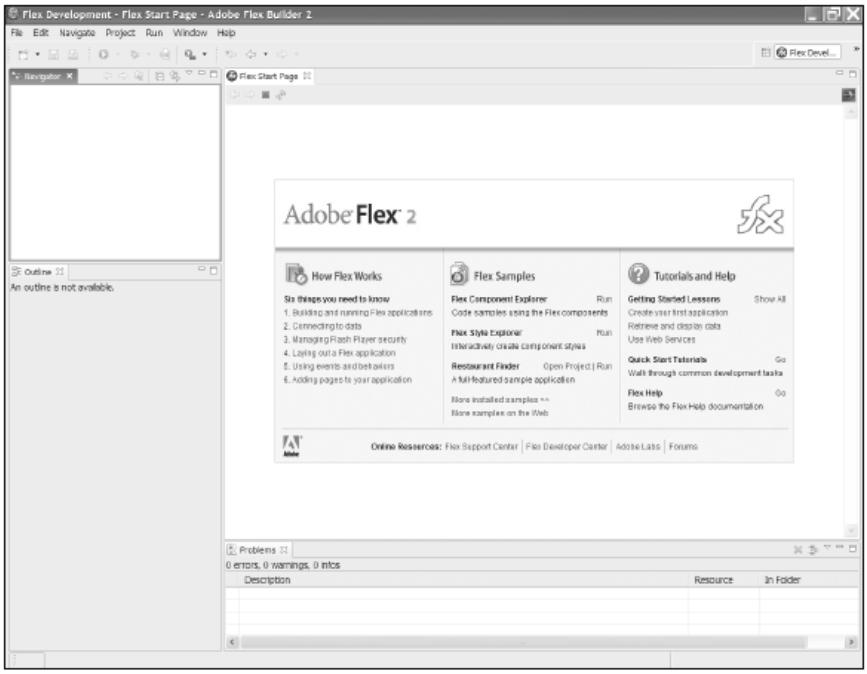

Figura 5.8 Flex Builder 2.0

Flex Builder es el IDE oficial para el desarrollo de aplicaciones Flex. Los diseñadores pueden crear con agilidad prototipos para las interfaces de aplicaciones, de una manera rápida y fácil y conectándolas a fuentes de datos. Los desarrolladores pueden entonces codificar y depurarlas de una forma productiva usando el código sugerido y las herramientas de depuración.

Flex pueden trabajar con varios tipos de herramientas. El primer tipo es cualquier editor del texto genérico por ejemplo VI, NotePad. Los archivos MXML son texto normal, pudiendo usar cualquier editor de texto esta modalidad hace que las aplicaciones sean fáciles de gestionar con los sistemas de control de código fuente. El segundo tipo que se puede usar es un IDE, por ejemplo: Borland JBuilder, JetBrains IntelliJ IDEA, Eclipse o Altova Xmlspy. **[WWW049]**

a. CARACTERISTICAS

- Actualización automática código-diseño y viceversa de los componentes MXML.
- Cuando se escribe una etiqueta aparecen todas las opciones asociadas a esta.
- Integración MXML con ActionScript.

- Ofrece sugerencias de código, coloreado y conclusión de etiquetas automáticas.
- Posee un previsualizador que permite ejecutar las aplicaciones directamente desde el IDE.
- Dispone de un depurador donde aparecen los posibles fallos de la aplicación.
- Importación de CSS para cambiar el aspecto de los componentes.
- Integración con diferentes lenguajes de programación Web: MXML, ActionScript, ColdFusion, ASP, PHP, JSP, XHTML, HTML y ASP.NET.
- Permite la importación de componentes Flash MX 2004 (.swc).
- Soporte de accesibilidad.
- También posee un validador de datos que son especialmente útiles en formularios para comprobar direcciones de correo, códigos postales, etc.
- Varios efectos a los componentes.

5.3.3 SERVICIOS DURANTE LA EJECUCIÓN

Flex incluye un conjunto de servicios durante la ejecución para la compilación, la memoria caché, la integración de los recursos y las necesidades de despliegue durante la ejecución. Todos los servicios reducen al mínimo la redundancia y explotan al máximo los recursos empresariales existentes. Por ejemplo, la lógica de la aplicación del lado del servidor, la autenticación, y la gestión de sesiones se proporcionan a través de una integración con Java subyacente. **[WWW050]**

La aplicación Flex se compila la primera vez que se acude a ella a través de una petición HTTP del cliente, y luego se mantiene en la memoria caché para las invocaciones subsiguientes, las aplicaciones Flex se recompilan automáticamente si se actualiza cualquier archivo relacionado. Tenemos los siguientes servicios relacionados a la integración de Flex:

- Proxy de servicios Web del lado del servidor, que extiende el modelo de seguridad nativo de Flash y da soporte al acceso de datos de una forma segura fuera del dominio de origen de la aplicación.

- Acceso de objetos del lado del servidor que está disponible para aplicaciones, datos e integración de directorios.
- Soporte de sesiones compartidas que permite que las sesiones de la aplicación Flex sean compartidas entre los contextos de HTML y la aplicación Flex.
- Autenticación que da soporte a los servicios de entrada al sistema.
- Servicios de detección y actualización de Flash Player que detectan, y actualizan, las instalaciones de Flash Player.

5.3.4 LIBRERIAS COMPARTIDAS DURANTE LA EJECUCIÓN

Son librerías para crear componentes propios o clases ActionScript, y que pueden ser utilizadas por diferentes aplicaciones Flex a la vez. Con este sistema de librerías se reduce el peso de las aplicaciones Web ya que no hace falta repetir el mismo código en cada una de ellas. Por ese mismo motivo, las aplicaciones Flex son cargadas más rápidamente. **[WWW050]**

5.3.5 SEGURIDAD

Actualmente, por el incremento de contenidos enriquecidos como valor agregado en un portal con servicios Web, como se muestra en la figura 5.9, se hace necesaria la implementación de estrategias para la transferencia o manipulación de información crítica entre el cliente y la aplicación. **[WWW051]**

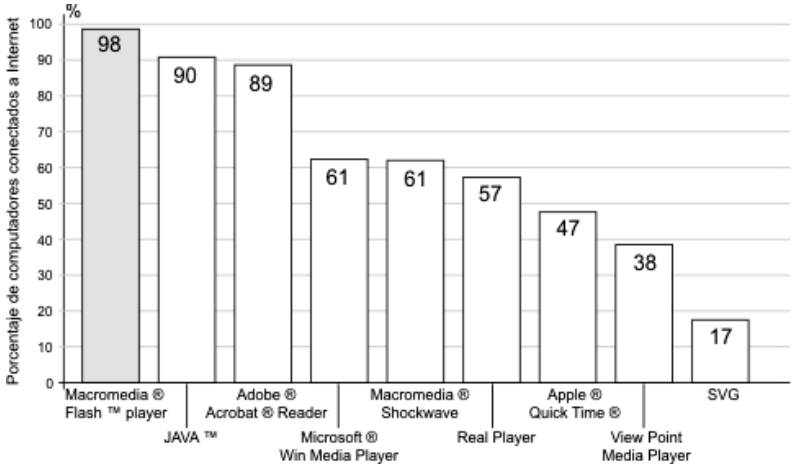

Figura 5.9 Participación de las tecnologías multimedia sobre Internet

Los peligros potenciales relacionados con estas aplicaciones son los siguientes:

Ataque en el cliente

Flash pueden publicar archivos portables con contenidos multimedia que a su vez representan un peligro mayor que los objetos embebidos ejecutados dentro de un navegador. El peligro se incrementa debido a que los archivos ejecutables publicados, dada su propia naturaleza, podrían ser portadores de código malicioso.

Los ataques pueden presentarse cuando un archivo .swf escribe o lee información en o desde el sistema de la máquina cliente. Para este fin, Adobe limitó el poder de los Local Shared Objects (cuyo comportamiento es análogo a las Cookies convencionales), haciendo que los datos sólo puedan ser escritos en un directorio específico. Así mismo, el usuario final, en la máquina cliente siempre tiene el control acerca de cuanta información puede ser alojada, y a qué dominios se les permite escribir y leer datos con esta técnica. **[WWW051]**

Ataque en la transferencia

Puede ocurrir en tiempo de ejecución, mientras intercambia información entre el cliente y el servidor, pues el objeto embebido podría ser coaccionado por una tercera entidad, aprovechando el estado de espera para enviar información no válida, o bien aprovecha el estado de envío de información desde la máquina que solicitó el servicio, para simular ser el receptor de los datos.

Para solucionar este inconveniente, Adobe utiliza todos los recursos de seguridad asociados al navegador. Por ejemplo, los servicios más críticos pueden ser alojados en servidores que operen **HTTPS**, por ejemplo la información intercambiada puede ser protegida por SSL. **[WWW051]**

Ingeniería inversa sobre el archivo .swf

Dada la condición de ActionScript como lenguaje interpretado por Flash Player, es susceptible de ser reconstruido y manipulado por terceros. Una solución es la encriptación en código nativo para los datos críticos. A menudo se utiliza el **Md5**, o

se usan técnicas de ofuscación mixtas (Md5 más **TEA**, para lograr la reorganización del código) que dificultan notablemente este procedimiento.

De cualquier forma, los archivos .swf, por obedecer a una especificación abierta, es teóricamente imposible proporcionar una solución 100% segura para proteger el código ActionScript. **[WWW051]**

a. NORMAS DE SEGURIDAD EN FLEX

Es recomendable no escribir código ActionScript que contenga sentencias con datos críticos, como las cadenas de comparación para la validación o solicitudes SQL explicitas. Si esto es inevitable, debe emplearse la encriptación con clases creadas específicamente para este propósito.

Si se necesita acceso a información crítica, es recomendable, cargarla desde un servidor confiable en tiempo de ejecución, usando un medio seguro para el transporte (SSL). Como regla general, los datos nunca deberían ser parte del código compilado dentro de un archivo .swf.

Es recomendable usar la identificación del usuario y la validación de datos fuera de los archivos, mediante el uso recurrente de scripts. Si es inevitable la inclusión de los algoritmos más privados, entonces se recomienda ofuscar el código, con herramientas como **Tevas**, **ActionScript Obfuscator**, **Viewer Screwer** o **FLASM**.

Si para la correcta prestación del servicio se hace necesaria la utilización de servicios Web prestados por terceros, se recomienda la verificación de la confiabilidad y la calidad de sus servicios. **[WWW051]**

Por otro lado el modelo de seguridad de Flex protege tanto al cliente como al servidor, posee varios mecanismos de seguridad que permiten el acceso controlado a servicios Web, servicios HTTP y a clases Java **EJB**. Además, las aplicaciones Flex al ser ejecutadas mediante Flash Player, también poseen las características de

seguridad que éste ofrece. Las características más importantes de seguridad son las siguientes: **[WWW052]**

- Permite la utilización de SSL para encriptar la transferencia de datos entre la aplicación y el servidor.
- Flash player evita que el contenido Web pueda acceder al sistema de ficheros local de un usuario, exceptuando el acceso a ficheros SharedObjects.
- La aplicación Web tampoco puede almacenar datos en la máquina del cliente, a excepción de los SharedObjects.
- El acceso a los servicios Web está restringido para evitar ataques de DoS (Denial of Service o Calidad de Servicio).
- Para acceder a clases Java, se puede utilizar mecanismos de autenticación por usuario.

5.4 FLEX Y LOS ESTÁNDARES

Flex se apoya en los siguientes estándares de la industria:

- **XML**: La construcción de los elementos de la interfaz utiliza MXML un lenguaje basado en XML y con normas relacionadas con **XForms** del W3C.
- **XML namespaces**: El atributo del xmlns es una etiqueta de MXML que especifica un espacio de nombres (namespace) de XML. Los namespaces de XML permiten referirse a más de una etiqueta XML en el mismo documento XML.
- **DOM, modelo de evento:** El modelo de evento es un subconjunto de DOM nivel 3 Eventos Nivelados que define un sistema de eventos que permite el registro del idioma neutral para manejadores de evento, descripción del flujo de evento a través de una estructura del árbol, y facilita la información contextual para cada evento.
- **SOAP**: Las aplicaciones Flex soportan servicios Web a través de SOAP donde los mensajes son transportados sobre HTTP.

- **ECMAScript/JavaScript**: Adobe se basa firmemente en el lenguaje ActionScript (lenguaje de programación orientado a objetos, es similar al lenguaje JavaScript) que esta contemplado en el estandar ECMAScript-262, cuarta edición, la misma que JavaScript.
- **CSS**: Los estilos de MXML están basados en el estándar CSS.
- **Objetos Java**. Las etiquetas de MXML actúan recíprocamente con el servidor y objetos de Java como JavaBeans.
- **SVG** (Scalable Vector Graphics) las aplicaciones Flex emplean gráficos de vector SVG a través de FlexCharting. **[LIB025]**

5.4.1 ECMASCRIPT

ECMAScript es una especificación de lenguaje de programación publicada por ECMA (European Computer Manufacturer's Association) International. El desarrollo empezó en 1996 y estuvo basado en el popular lenguaje JavaScript propuesto como estándar por Netscape. Actualmente está aceptado como el estándar ISO 16262. Define un lenguaje de tipos dinámicos ligeramente basado 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. **[LIB025]**

La mayoría de navegadores de Internet incluyen una implementación ECMAScript, al igual que un acceso al Document Object Model para manipular páginas Web. JavaScript está implementado en la mayoría de navegadores, y el Internet Explorer de Microsoft usa JScript. El navegador Opera tiene su propio intérprete de ECMAScript con extensiones para soportar algunas características de JavaScript y JScript. Cada navegador tiene extensiones propias al estándar ECMAScript, pero cualquier código que se adecue al estándar debería funcionar en todos ellos.
ActionScript, para Macromedia Flash, también está basado en el estándar ECMAScript, con mejoras que permiten a los objetos ser movidos, creados y analizados dinámicamente, mientras la película está en ejecución. **[WWW053]**

5.4.2 MXML

MXML es un lenguaje de marcado que describe los elementos de la interfaz de usuario exponen contenido y funcionalidad. MXML proporciona abstracciones declarativas para la lógica de presentación y enlaces entre la interfaz y los datos del servidor, aprovecha la reutilización de la aplicación porque separa nítidamente la capa de presentación, de la lógica del negocio.

El desarrollo con MXML se basa en el mismo proceso iterativo que se utiliza para otros tipos de archivos de aplicaciones Web, como HTML, JSP, ASP y ColdFusion (CFML lenguaje de marcado). **[WWW054]**

Asimismo, los archivos MXML son archivos XML ordinarios, de modo que eso le permite escoger de entre un amplio abanico de entornos de desarrollo. Se puede desarrollar con un editor de texto sencillo, con un editor de XML exclusivo, o con un IDE que sea compatible con la edición de texto.

Y como MXML se ajusta a la definición de esquema XML estipulada por el W3C, también puede usar la edición estructurada, el coloreado de código y las sugerencias de código (según lo que permita su editor). También MXML se usa para definir aspectos no visuales de una aplicación, como el acceso al servidor de fuentes de datos.

Por ejemplo, <mx:Button> esta etiqueta crear un control botón que usa la declaración de MXML, la propiedad id da un único nombre que se puede usar para referirse a él. La propiedad label pone el texto de la etiqueta del botón:

```
<mx:Button id="mvButton" label="I'm a button!"/>
```

El ejemplo siguiente muestra el código completo para crear el despliegue del botón:

```
<?xml version="1.0" encoding="utf-8"?>

<mx:Application
   xmlns:mx="http://www.adobe.com/2006/mxml"
   viewSourceURL="src/Saludo/index.html"
   horizontalAlign="center" verticalAlign="middle"
   width="300" height="160">
   <mx:Panel
      paddingTop="10" paddingBottom="10" paddingLeft="10" paddingRight="10"
      title="Mi primera aplicación">
        <mx:Label text="hola mundo!" fontWeight="bold" fontSize="24"/>
        <mx:Button id="myButton" label="Soy un boton!"/>
  </mx:Panel>
</mx:Application>
```

Describiendo el código en la primera línea usamos la declaración de XML. Esta línea tiene que ser la primera en cada archivo MXML.
Luego <mx:Application> define el contenedor de la aplicación y es la etiqueta raíz de la misma.
<mx:Panel> define el contenedor del panel que incluye una barra del título, un título, un mensaje de estado, un límite, y una área para sus elementos.
<mx:Label> la etiqueta representa un componente de interfaz de usuario muy simple que despliega el texto, con propiedades como fontWeight, fontSize que permiten cambiar el estilo de la etiqueta en fuente y tamaño respectivamente. **[WWW054]**
Para ampliar el conocimiento sobre la sintaxis de MXML se lo puede estudiar en el Anexo 5.1.

5.4.3 ACTIONSCRIPT 3.0

MXML, al ser un lenguaje de marcado tiene algunas limitaciones como uso en la toma de una decisión o ejecutar bloques de código en un cierto número de tiempo.
En los días tempranos de Internet, estas limitaciones se manejaron con el desarrollo del JavaScript que aumentó la habilidad del HTML. En el ambiente Flex, ActionScript es análogo a JavaScript: se lo usa para aumentar el poder de MXML.

Con ActionScript 3.0, se puede agregar interacción dinámica entre sus componentes. Por ejemplo, que una etiqueta muestre información dinámicamente basada en una

caja de texto de acuerdo a la información del usuario, o como se muestra a continuación la importación de un paquete el cual contiene una clase Saludo y una función constructora que indica una alerta de Bienvenida:

```
package {
        import flash.display.Sprite;
        public class Saludo extends Sprite{
                public function Saludo(){
                        trace("Bienvenidos a Flex 2 y ActionScript 3.0");
                }
        }
}
```

En Flex todos los componentes, librerías y paquetes usados están desarrollados con ActionScript 3.0, de hecho, si se quiere construir propios componentes, se necesita conocer ActionScript. Lo que es más, cuando se compila la aplicación Flex el código MXML se transforma en un archivo .swf, con código ActionScript 3.0. **[LIB027]**

ActionScript ha evolucionado su lenguaje de programación para algunas rutinas de animación usadas en Flash, utilizando: archivos de clase que son programas autónomos que contiene todas las variables (propiedades) y los métodos necesarios para realizar cualquiera tarea. Esto también sirve como la base, o plantilla de los objetos. Un objeto es una copia del archivo de la clase en la memoria. Desde que los archivos de la clase son autónomos y especializados, se puede usarlos en cualquier proyecto. Esencialmente, ActionScript 3.0, es un ambiente conformado por una colección grande de archivos de clase. Un archivo de clase es una manera poderosa de modularizar las aplicaciones en bloques pequeños, reusables mediante la herencia. Se identifica la herencia con la palabra extend.

ActionScript 3.0 toma el concepto de paquetes para agrupar y organizar archivos de clase relacionados. Los ambientes OOP, como Java, sólo permiten empaquetar los archivos de la clase, pero ActionScript 3.0 permite al paquete tener métodos individuales y propiedades. Esto significa que se puede construir bibliotecas de las propiedades y métodos sin que ellos estén asociados con una clase particular. Lo que es más, se puede tener el acceso a la clase cuando se necesita. Este rasgo poderoso puede ayudarnos mucho en la programación potencial.

5.5 FLEX Y OTRAS TECNOLOGIAS

5.5.1 FLEX Y FLASH

Es importante analizar que Flex nos permite trabajar con productos Adobe complementarios como Flash, ya que es una herramienta de edición para crear presentaciones y aplicaciones con una amplia variedad de contenido multimedia que incluye imágenes, sonido, vídeo y efectos especiales. El lenguaje que se usa es ActionScript que permite añadir interactividad a los elementos multimedia del documento. También se puede utilizar para añadir lógica a las aplicaciones. **[LIB011]**

Flash se usa para hacer prototipos que estén integrados en la aplicación Flex desplegando medios interactivos altamente visuales y expresivos. Es decir Flash es la herramienta para el desarrollo de contenidos interactivos visuales. Flex es un marco de aplicación diseñado para desarrolladores que generalmente programan aplicaciones del lado del servidor y no suelen utilizar herramientas de diseño y desarrollo visual. Flex muestras las ventajas de la experiencia dinámica hecha posible gracias a Flash Player. **[LIB026]**

5.5.2 FLEX Y COLDFUSION

ColdFusion fue desarrollado por Allaire Corporation para ser una alternativa al usar Perl y otras tecnologías CGI. Es un servidor de aplicaciones Web que corre en forma concurrente con la mayoría de los servidores Web de Windows, Linux y Solaris, trabaja mediante HTTP para procesar peticiones de páginas Web, cada vez que se solicita una, el servidor de aplicaciones ejecuta el script o programa contenido en la página, usa un lenguaje basado en tags, llamado CFML (ColdFusion Markup Language) con el cual se puede crear y modificar variables, igual, posee controles de flujo que en otros lenguajes de programación. **[WWW052]**

ColdFusion interactúa de manera simple con bases de datos (Sybase, Oracle, MySQL, SQL, entre otros). Usando SQL estándar, las páginas y aplicaciones Web

pueden fácilmente recuperar, guardar, formatear y presentar información dinámicamente. Además es escalable, a veces, el problema más grande es que un sitio se vuelve popular. ColdFusion está diseñado para correr en máquinas multi-procesador, y permite construir sitios que pueden correr en clusters de servidores y trabajar con múltiples arquitecturas a través de la integración de COM, CORBA, XML y EJB.

También puede ser fácilmente extendido con nuevos componentes creados con servlets o clases Java. **[WWW052] [WWW055]**

Existe una excelente integridad entre Flex y ColdFusion para desarrollar RIAs de forma rápida. En ColdFusion creamos los componentes de datos para acceder rápidamente a la lógica del negocio a través del servidor y en Flex creamos la interfaz de usuario enriquecida.

El servidor de presentación Flex puede instalarse por separado en el mismo servidor físico donde esta ColdFusion ya que posee un servlet que permite la integración de componentes datos e interfaz. **[REV002]**

Para mayor referencia mirar el Anexo 5.2 en el cual se especifica la sintaxis más importante de ColdFusion MX.

5.5.3 FLEX Y XML

La fuente de los datos

La tendencia actual, es trabajar con una mezcla de bases de datos y XML. Ya que la mayoría de los bases de datos tienen un alto rendimiento con la incorporación de XML.

Flex no tiene la capacidad de acceder una base de datos directamente. En su lugar usaremos eventos para tener acceso a archivos de clase que permiten manejar archivos XML.

Eventos

Flex permite crear una variedad de componentes, algunos, de esos interactúan con los objetos mediante eventos. Los componentes incorporados realmente se derivan de

ActionScript. Por ejemplo un Botón, automáticamente llama a su archivo de clase con todas sus propiedades, funciones, y eventos. Esta llamada (incluso el comportamiento de pulsar el botón) se hereda de clases superiores. Por esa razón, nada pasa sin un evento.

Evento del Objeto

Cuando un evento pasa en ActionScript, se genera un objeto llamado evento del objeto, que contiene dos propiedades: quién generó el evento o target y quien lo llama o type. Estas dos propiedades tienen gran importancia en el desarrollo Flex. Por ejemplo un botón que al presionarlo genere una acción:

```
<mx:Button label="Boton 1" id="miBoton" click="fillLabel(event)" />
```

Note que el evento clic llama a fillLabel(), y el manejador de eventos pasa un parámetro event que describe el objeto a ser usado, ejemplo:

```
private function fillLabel(evt:Event):void
{
        myLabel.text = evt.target.id + " esta presionado";
}
```