TEMARIO OPOSICIÓNS INFORMÁTICA

GRUPO A2 – XESTIÓN E SISTEMAS DE INFORMACIÓN

TEMA 26. HTML. APLICACIÓNS DA INTERNET ENRIQUECIDAS (RIA).

Autor: Juan Marcos Filgueira Gomis

TEMA 30. HTML. APLICACIÓNS DA INTERNET ENRIQUECIDAS (RIA).

30.1 INTRODUCIÓN E CONCEPTOS

Os documentos web e MIME poden definirse a través da linguaxe de marcado HTML, (en inglés *HyperText Markup Language*) que permite describir a estrutura e contido dun documento a través de etiquetas, ademais de permitir outros elementos complementarios como estilos ou *scripts*. Variantes desta linguaxe permiten un dominio máis estrito adaptado a ámbitos concretos como o XHTML ou MathML e máis abertos coma SGML (en inglés *Standard Generalized Markup Language)*. A última versión da linguaxe HTML5 incorpora etiquetas específicas para contidos multimedia, grandes conxuntos de datos, melloras en formularios, web semántica e outras opcións, que responden á evolución das necesidades da web hoxe en día.

En extensión, as aplicacións de Internet enriquecidas ou **RIA** (en inglés *Rich Internet Applications*), son un conxunto de tecnoloxías que buscan achegar as interfaces das aplicacións web ás das aplicacións de escritorio dotándoas de novas funcionalidades, de aí a riqueza, e axilizando aspectos como as recargas de datos. Por norma xeral precisan dun *framework*, compoñente adicional ou *plug-in* no navegador que permitan a súa interpretación. Nas aplicacións RIA a maior parte da comunicación faise de maneira asíncrona en comunicacións transparentes ao usuario que evitan gran parte das recargas de páxinas para realizar actualizacións de datos. Fronte a estas vantaxes no tocante á usabilidade en canto a mellora das funcionalidades e actualizacións de datos a principal desvantaxe será a accesibilidade da páxina para usuarios que presenten dificultades de acceso á información na web.

Moitas destas **tecnoloxías** pertencen ao mundo do software propietario atopando gran dependencia respecto das compañías que as desenvolven. As principais tecnoloxías atópanse nas plataformas Flash, Flex e AIR de Adobe, Silverlight de Microsoft, OpenLaszlo, incontables *frameworks* AJAX e Javascript, e outras tecnoloxías como as xa maduras, Applets e Java WebStart e as emerxentes como XUL, JavaFX, GWT ou Bindows. Calquera destas tecnoloxías localízase na capa de vista, usuario ou cliente como complemento á (X)HTML/CSS e intégranse coas tecnoloxías de servidores de aplicacións como .NET/JEE.

Dentro dos **casos de éxito** desta tecnoloxía, actualmente a maior parte das aplicacións e servizos web globais de maior uso dentro da Web 2.0 fan uso de tecnoloxías RIA nas súas interfaces, sendo Google Maps, Gmail, Flickr, Meebo, Orkut, e un longo etcétera.

30.2 HTML

O conxunto de estándares HTML presentan un conxunto de normas definido polo W3C, sendo a versión estándar máis habituais a HTML 4.01, a recente HTML 5 e a variante XHTML 1.0. Un documento HTML describe a estrutura e contido dunha páxina ou sitio web, permitindo acceso ao DOM e facendo uso de arquivos multimedia e *scripts* externos que á súa vez poden adoptar a forma de tecnoloxías RIA ou DHTML. Neste caso no documento HTML especifícase un obxecto incrustado na tecnoloxía correspondente e calquera navegador identifícao e reprodúceo segundo o correspondente Plugin. Grazas ao HTML pódese definir un documento cos seguintes elementos:

- Publicación de contidos en liña, como textos, encabezados, listas, táboas, parágrafos, formatos e estilos, etc...
- Asociación de documentos vía ligazóns de hipertexto.
- Deseño de formularios para permitir a entrada de información e comunicación con servidores remotos.
- Incrustación nos documentos de obxectos externos como elementos multimedia ou obxectos de aplicacións externas como follas de cálculo, documentos Pdf e outros.

Os documentos HTML especifican tamén a **codificación de caracteres** internacional do documento sendo UTF-8 ou Latin-1. Esta identificación da codificación permite que outras aplicacións, como navegadores ou procesadores de texto, poidan recoñecer e presentar adecuadamente a información do documento. Así mesmo dentro desta codificación, os caracteres especiais represéntanse a través de **entidades HTML**, que definen de maneira unívoca o carácter especial a través dun código HTML específico.

Carácter especial	Entidade HTML
Á	Á
á	á
Espazo en branco	
Ñ	Ñ
ñ	ñ
Ü	Ü
>	>
<	<

Táboa 1: Exemplos de entidades HTML.

30.2.1 ACCESIBILIDADE WEB

Outro aspecto asociado á definición de documentos web é o de **accesibilidade web**, onde o W3C elaborou as Pautas de Accesibilidade ao Contido Web 1.0 ou WCAG adaptadas no caso español coa Norma UNE 139803. Estas guías definen unha serie de regras e normas que deben cumprir os documentos web para permitir o acceso á súa información a persoas con discapacidade ou que presentan dificultades específicas.

As WCAG 1.0 son catorce pautas ou patróns de deseño web con solucións comúns para resolver problemas de acceso a contidos web. As pautas conteñen unha serie de puntos de verificación que facilitan a detección de posibles problemas de acceso. Cada un dos puntos ten asignado un nivel de prioridade ou peso segundo á importancia ou dificultade de acceso á información que leve asociado. Estes niveis son:

- **Prioridade 1.** Puntos que debe cumprir un sitio web ou doutro xeito non se permitirá que certos grupos de usuarios non poderán acceder aos contidos do sitio.

- **Prioridade 2.** Puntos que debe cumprir un sitio web ou doutro xeito a certos grupos de usuarios seralles moi difícil acceder aos contidos do sitio.
- **Prioridade 3.** Puntos que debe cumprir un sitio web ou doutro xeito algúns grupos de usuarios poderían ter dificultades á hora de acceder aos contidos do sitio.

En función ao cumprimento destes puntos de verificación establécense tres niveis de conformidade dun sitio web coas pautas de accesibilidade:

- Nivel de Conformidade **A**. Cúmprense todos os puntos de verificación de prioridade 1.
- Nivel de Conformidade **Duplo A**. Cúmprense todos os puntos de verificación de prioridade 1 e 2.
- Nivel de Conformidade **Triplo A**. Cúmprense todos os puntos de verificación de prioridade 1,2 e 3.

Prioridade	Punto de verificación	Cumprimento
1	Proporcionar texto equivalente a elementos non textuais	S/N/N.A.
1	Asegurar que a información transmitida con cor ten alternativa sen cor	S/N/N.A.
1	Identificación de cambios de idioma	S/N/N.A.
1	Organizar o documento independentemente das follas de estilo	S/N/N.A.
1	Garantir que as alternativas ao contido dinámico actualízanse cando este	S/N/N.A.
1	Non provocar destelos de pantalla	S/N/N.A.
1	Empregar unha linguaxe clara e simple no sitio	S/N/N.A.
1	Proporcionar ligazóns de texto redundantes en mapas de imaxes	S/N/N.A.
1	Nas táboas identificar as cabeceiras de fila e columna	S/N/N.A.
1	Titular cada marco ou *frame* da páxina	S/N/N.A.
1	Permitir o funcionamento sen linguaxes de scripts de cliente activados	S/N/N.A.
1	Prover alternativas auditivas sincronizadas en contidos multimedia	S/N/N.A.
1	No caso de ser imposible cumprir cos puntos anteriores cómpre crear un documento completo alternativo	S/N/N.A.
2	Garantir contraste de cor de texto e fondo lexible	S/N/N.A.
2	Empregar follas de estilo para maquetación e presentación	S/N/N.A.
2	Empregar documentos validados formalmente	S/N/N.A.
2	Empregar elementos de encabezado	S/N/N.A.
2	Empregar unidades relativas fronte a absolutas	S/N/N.A.
2	Marcar axeitadamente listas, ítems e citas	S/N/N.A.
2	Evitar navegación ou refresco de páxinas automáticos	S/N/N.A.
2	Evitar abrir novas ventás sen informar ao usuario	S/N/N.A.
2	Empregar índices ou mapas dos sitios e metadatos para aportar información semántica	S/N/N.A.

3	Proporcionar unha orde lóxica de tabulación	S/N/N.A.
3	Proporcionar atallos de teclado	S/N/N.A.
3	Incluír caracteres por defecto nos formularios	S/N/N.A.
3	Proporcionar abreviaturas para os encabezados das páxinas	S/N/N.A.
3	Proporcionar resumos para as táboas	S/N/N.A.
3	Proporcionar barras de navegación	S/N/N.A.

Táboa 2: Exemplo de lista de puntos de verificación.

Paralelamente desenvolvéronse as **Pautas de accesibilidade para ferramentas de autor** ou ATAG (en inglés *Authoring Tool Accessibility Guidelines* 1.0) co obxectivo de axudar aos desenvolvedores á hora de crear ferramentas de autor destinadas á elaboración de contidos web accesibles; as **Pautas de accesibilidade para XML** ou XAG; e as **Pautas de accesibilidade para axentes de usuario** 1.0 ou UAAG orientadas a definir un marco accesible de soporte a compoñentes software que acceden a contidos web como navegadores, reprodutores multimedia, procesadores de texto e outras tecnoloxías asistivas. Por último dentro do marco das aplicacións RIA a iniciativa **WAI-ARIA** comeza a dar soporte a requirimentos de accesibilidade para este tipo de tecnoloxías.

Unha das regras de accesibilidade web é cumprir coa validez do documento. Para avaliala unha das etiquetas, DOCTYPE, permite especificar a versión de HTML empregada para elaborar o documento. Ademais de permitir corrixir problemas de compatibilidade con navegadores permite validar que o documento estea ben formado.

Existen tres **definicións do tipo de documento** ou DTD:

- **Strict**. Estrito, respectando escrupulosamente as normas e sintaxe do documento, e evitando parte do conxunto de etiquetas posible total.
- **Transitional**. Con máis liberdade, fai uso de todo o conxunto de etiquetas e permite licenzas na sintaxe.
- **Frameset**. Reservado para documentos que fan uso de marcas para organizar a estrutura do sitio web.

A partir desta especificación existen ferramentas de **validación** en liña e para descarga, que orientadas cara a accesibilidade ou cara a corrección do código permiten avaliar a calidade do documento.

30.2.2 HTML PARA DISPOSITIVOS MOBILES

Así mesmo ao aumentar as posibilidades de conexión dos **dispositivos móbiles**, medrou a demanda de dar soporte web con esta orientación e por tanto de adaptar os documentos ás necesidades específicas deste tipo de equipos, moitas das mesmas moi ligadas ao ámbito da accesibilidade e o deseño líquido. Os protocolos máis habituais **WAP**, **WAP 2.0** e **WCSS** recollen os DTD máis empregados. O W3C proporciona un subconxunto específico para móbiles o XHTML 1.1 Basic, e a OMA (en inglés *Open Mobile Alliance*) a especificación XHTML Mobile Profile, ambos garanten o soporte dos documentos que cumpran as validacións cos dispositivos máis restrinxidos. No caso de WAP non se atopan soportados os protocolos de Internet, polo que cómpre empregar nodos pasarela. Así mesmo dispón dunha linguaxe de *script* propia, **WMLScript** baseado en ECMAScript e empregado nas páxinas **WML** (en inglés *Wireless Markup Language*) que empregan á súa vez XML. A seguinte versión WAP 2.0 si adopta os protocolos de Internet, o que fai que non sexa imprescindible a pasarela, presentando ademais melloras de rendemento para a comunicación en dispositivos móbiles dentro destes protocolos.

Outras tecnoloxías relacionadas co deseño web para móbiles serían **i-mode**, que presenta unha linguaxe moi similar a HTML para a creación de micropáxinas e situada a niveis de mercado preto do WAP en países como Xapón. Outra tecnoloxía complementaria sería o **J2ME** (en inglés Java2 MicroEdition) versión lixeira, ou subconxunto, destinado a dispositivos móbiles da plataforma Java.

30.2.3 HTML 5

O HTML busca dar resposta a moitas das necesidades actuais dos sitios web e que non foron recollidas nas versións anteriores da especificación. Nesta versión ten lugar unha maior integración co DOM, a través do IDL (en inglés *Interface Definition Language*). Dentro das novas características de HTML5 atópanse:

- **Unha nova estrutura de documento**. Con seccións como NAV, bloque de navegación do sitio web, HEADER, FOOTER, ARTICLE ou SECTION que aportan información semántica máis relevante que empregar unicamente DIVS ou táboas.
- **Novas etiquetas semánticas**. Como TIME para datas/tempo ou MARK para destacar elementos, METER, PROGRESS, así como incorporación nos documentos de arquivos RDF ou OWL con metadatos.

- **Novas etiquetas multimedia**. Como AUDIO e VIDEO que presentan ademais os códecs necesarios para reproducir os contidos incrustados.
- **Etiquetas para gráficos**. As melloras no compoñente CANVAS permiten a representación de gráficos 2D/3D, como formas, degradados, transformacións e outras opcións gráficas.
- **Novos compoñentes para formularios.** Con novos tipos de datos para correo electrónico, teléfono, números, datas, etc... que permiten a validación no cliente sen empregar Javascript.
- **Compoñentes para operar con grandes conxuntos de datos.** Compoñentes como DATAGRID, MENU ou COMMAND, proporcionan a estes elementos comúns dos sitios web características dinámicas que facilitan o control do comportamento da páxina sen librarías externas.
- **Novas APIS**. Librarías para operacións tan variadas como arrastrar obxectos, funcionamento fóra de liña, xeolocalización, almacenamento persisten en local baseadas en SQL Lite, WebSockets para comunicación entre sitios distribuídos, WebWorkers para traballo con fíos de execución paralelos e outras posibilidades que se atopan en estudo ou desenvolvemento.

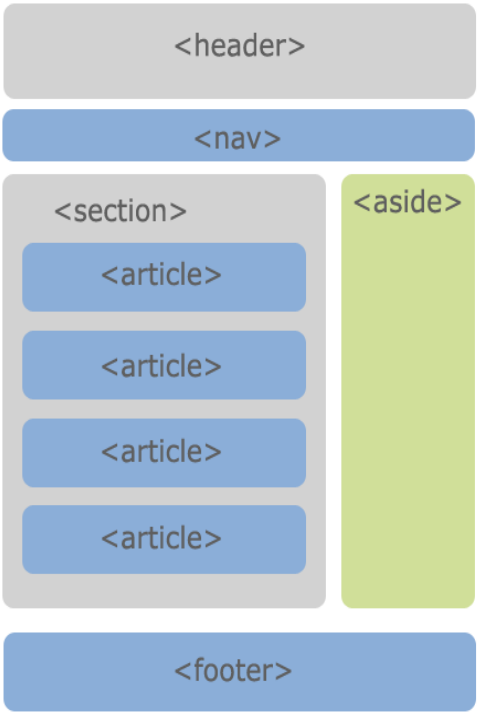


Figura 1: Exemplo de estrutura dun documento en HTML5.

30.3 AJAX

AJAX (en inglés *Asynchronous JavaScript And XML*), Javascript Asíncrono e XML, orixínase para aproveitar que a comunicación entre o usuario e a interface non é fluída permitindo realizar comunicacións asíncronas co servidor e realizar así un maior aproveitamento do ancho de banda e obter unha maior velocidade de resposta. Os datos cárganse nun segundo plano sen afectar á interface. AJAX non representa unha tecnoloxía en si mesma, senón que se trata da combinación dun grupo de **tecnoloxías** xa existentes:

- ✔ (X)HTML e CSS para o deseño das páxinas web.
- ✔ DOM (en inglés *Document Object Model*), ou Modelo de Obxectos do Documento, API que representa un conxunto de obxectos para manipular e modificar dinamicamente documentos (X)HTML e XML a través de linguaxes de Script como Javascript, JScript ou ECMAScript.
- ✔ Obxectos dos tipos Iframe ou XMLHttpRequest para intercambiar datos de maneira asíncrona co servidor.
- ✔ XML para os formatos de intercambio de datos e comunicacións a través de JSON ou EBML.

- ✔ Outras tecnoloxías para facilitar a implantación de solucións específicas como XSLT, RSS, PDF ou outras tecnoloxías RIA.
- ✔ Javascript para proporcionar o nexo común a todo o conxunto.

Como acontece coas tecnoloxías RIA en xeral, é requisito que o navegador teña **soporte** para o conxunto de tecnoloxías AJAX, doutro xeito habería que proporcionar unha alternativa HTML básica. Aínda así cada ves atópase soportado por un maior número de navegadores e non require a instalación de complementos.

O **funcionamento básico** de AJAX baséase no obxecto de comunicación asíncrona, por exemplo o XMLHttpRequest, que se instala cunha libraría, *framework* ou motor AJAX no lado do cliente. O motor AJAX proverá os métodos para permitir a comunicación asíncrona de datos ademais de definir os compoñentes reutilizables AJAX coa definición do seu comportamento, e o contido e estrutura xeral do documento. O feito de realizar todas estas funcións dende o cliente de maneira asíncrona supoñen a gran mellora de rendemento de AJAX sobre o modelo tradicional de desenvolvemento web.

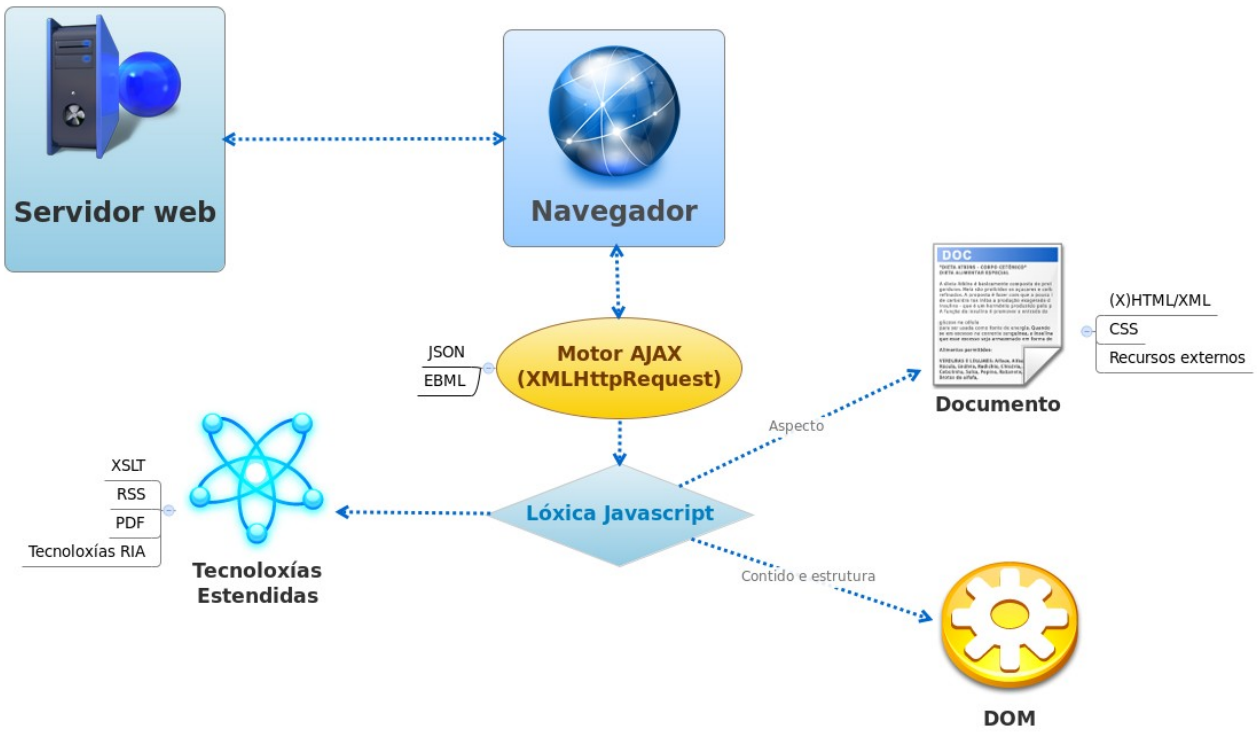


Figura 2: Funcionamento básico de AJAX.

A **refactorización** de aplicacións tradicionais ao modelo AJAX implica cambios na estrutura interna coa preserva das funcionalidades. A metodoloxía de refactorización habitual implica unha serie de cambios a pequena escala, para o que é ideal a programación orientada a obxectos. Estes cambios van dende:

- ✔ **Refactorización a nivel de método/clase.** En sistemas con pouco acoplamento, evitando por exemplo que se modifiquen os atributos das clases entre obxectos. Estes accesos múltiples poden integrarse nun nivel superior unha vez resoltas as dependencias do método/clase.
- ✔ **Creación de novas clases.** Defínense novas clases especializadas na arquitectura AJAX con responsabilidades e interfaces ben definidos.
- ✔ **Eliminación de clases intermediarias.** Elimínanse as delegacións en exceso entre clases do modelo tradicional en aplicación das consideracións sobre antipatróns de exceso de capas.
- ✔ **Compoñentes e etiquetado.** Diferentes compoñentes sobre todo da vista presentan funcionalidades comúns que se poden factorizar a través de compoñentes AJAX e etiquetados, para facilitar o mantemento e a reutilización.

Na actualidade existen infinidade de **frameworks e librarías AJAX**, incorporando compoñentes da vista, diferentes funcionalidades Javascript, elementos para comunicación asíncrona tanto no cliente coma no servidor e outros elementos para integración con outras tecnoloxías RIA. O obxectivo destes *frameworks* resúmese en facilitar o desenvolvemento de aplicacións web baseadas en AJAX, facendo fincapé en aspectos da capa de vista ou cliente. Os principais *frameworks* en canto ao seu uso máis estendido, son:

- ✔ **Prototype**. Pode ser o *framework* de uso máis estendido, tamén de código aberto dispón da extensión **Scriptaculous** para engadir animacións e efectos nos documentos, e JSON para intercambio de datos. Serve á súa vez a base de outros *frameworks* AJAX. Permite unha grande integración en aplicacións desenvolvidas con *Ruby on Rails* pero tamén pode operar de xeito independente. As súas principais características son:
  a) **DOM Estendido.** Referencia áxil a obxectos DOM, como por exemplo empregando a función $() en lugar de *document.getElementById()*.
  b) **Scriptaculous.** Aporta ao *framework* un construtor de obxectos DOM (*builder.js*), un repositorio de efectos visuais (*effects.js, slider.js*), funcionalidades de control de elementos (X)HTML (*dragdrop.js, controls.js*) e métodos para realizar test de verificación unitarios (*unittest.js*).

- ✔ **Dojo Toolkit.** Proxecto de software libre, actualmente co soporte de IBM e Sun entre outros, que contén varias APIs Javascript modulares e unha ampla coleccións de widgets para uso baixo demanda, agrupados nun sistema de paquetes ao estilo de JEE. Entre as súas principais características están a achega de:
  a) **Compoñentes empaquetados en Dijit.** Widgets para a estrutura das páxinas como menús, pestanas; específicos para calendarios, reloxos, gráficos, vectores 2D/3D, ordenación de táboas e paxinación; ademais de elementos para formularios e a súa validación, elementos HTML5 e compoñentes para mellorar a accesibilidade. Así mesmo presenta un editor de texto enriquecido e soporte drag & drop entre os seus compoñentes.
  b) **Comunicación asíncrona.** Prove dunha capa de abstracción para a comunicación transparente entre o navegador e o servidor web que fai uso de elementos Iframes ocultos para o refresco de datos.
  c) **Almacenamento no servidor.** Implementa diferentes mecanismos de almacenamento de datos vía CVS, OPML, RDF ou o servizo web Del.icio.us.

d) **Soporte para outras tecnoloxías.** Permite integración con tecnoloxías RIA como as aplicacións AIR baseadas en Javascript a través de API e soporte para móbiles.

✔ **JQuery**. Outro proxecto de software libre, que neste caso busca simplificar o acceso a documento (X)HTML, o modelo DOM, a manipulación de eventos, utilidades, animacións e efectos. Permite a instalación a través dun paquete básico moi lixeiro (*jquery.js*) que pode ser ampliado a través de plug-ins coma JExpand, para táboas, e JQueryUI para widgets con efectos visuais, entre outros moitos. Entre as súas principais características atópanse:

a) Integración na plataforma .NET e cos *frameworks* ASP .NET MVC e ASP .NET AJAX.
b) Soporte para CSS 3 e XPath.
c) Soporte para manipulación de (X)HTML E JSON.
d) Fai uso de programación non intrusiva.
e) Lixeiro e extensible.

✔ **Qooxdoo**. Colección de librarías Javascript multipropósito de código aberto, que como as vistas anteriormente permite control áxil a alto nivel de (X)HTML, CSS e DOM, ademais de proporcionar funcionalidades estendidas. A principal diferenza respecto as anteriores é que proporciona widgets de última xeración moi similares aos das aplicacións de escritorio. Entre as súas principais características atópanse:

a) **Abstracción do navegador.** Establece unha capa intermedia de abstracción do navegador coas especificacións necesarias para os principais tipos de navegadores definindo así unha interface estándar que mellora a compatibilidade sen necesidade de instalar *plug-ins* adicionais.
b) **Administración de eventos.** Prove unha interface propia con métodos para rexistrar e eliminar eventos.
c) Ferramenta de desenvolvemento de Interfaces de usuario.
d) Soporte de internacionalización i18n e localización l10n que permite o formato de tradución baseado en arquivos .po.
e) Prove *frameworks* para depuración de test unitarios e simulacións.

- ✔ **Mootools.** *Framework* de código aberto modulas e extensible que permite ao desenvolvedor seleccionar que compoñentes empregar para minimizar o peso final da libraría no cliente. Presenta un compoñente para a incorporación de efectos avanzados e transicións, estreitamente relacionados con Flash sendo un punto forte a súa integración con esta outra tecnoloxía RIA. Prove dos seguintes compoñentes:
    a) **Core**. Núcleo de funcións básicas que empregan todos os demais compoñentes do *framework*.
    b) **Class**. Libraría para instanciación e manipulación de obxectos.
    c) **Natives**. Extensións de funcións básicas Javascript.
    d) **Element e Effects/FX.** APIs para manexo de documentos HTML e aplicación de efectos sobre os seus elementos.
    e) **Remote**. Para intercambio de datos co servidor a través de peticións XmlHttpRequest, JSON ou Cookies.

- ✔ **ExtJS.** Conxunto de librarías derivadas da Yahoo! UI, actualmente emprégase como extensión de JQuery e Prototype incorporando *widgets* especializados, en especial na representación de gráficas e *grids*. Existe ademais unha adaptación específica para GWT denominada ExtGWT, con moitas optimizacións para este contorno. Incorpora unha capa propia dentro dunha arquitectura MVC o que lle permite prover de flexibilidade no tocante aos estilos, facendo uso da extensión SASS (en inglés *Syntactically Awesome Style Sheets*), unha extensión de CSS3. Así mesmo prove de librarías que facilitan a integración con AIR e Spring como backend. Dentro dos compoñentes de datos dispón de varios lectores tanto para XML como JSON. A arquitectura xeral inclúe os paquetes Base e Core coas funcionalidades comúns; os Compoñentes da interface de usuario cos widgets e gadgets; Remoting para a execución de métodos no servidor vía RPC; os Servizos de datos para lectura de vectores, XML e JSON; e o *miniframework Drag and drop* para permitir soporte de arrastre entre os compoñentes do *framework*.

- ✔ **Rico**. Baseado en Prototype e orientado cara a Web 2.0 as principais achegas deste *framework* inclúen efectos de animacións que permiten realizar transicións que poden ser interrompidas, pausadas ou reiniciadas, permitindo o solapamento de animacións. Permite a creación de efectos cinematográficos e outros efectos visuais.

Así mesmo proporciona as funcionalidades básicas para soporte AJAX e estende parte do repertorio de Prototype con melloras.

- ✔ **DWR**. (En inglés *Direct Web Remoting*). Framework de código aberto orientado á integración de AJAX con aplicacións JEE a través de mecanismos de RPC como RMI ou SOAP. Permite executar código Java nun servidor de aplicacións como se estivera no navegador do cliente, invocando os obxectos como se foran locais. Consta de dous elementos principais: un *framework* Javascript no cliente e un Servlet no servidor para procesar as peticións e xerar as respostas. O Javascript actuará como proxy das clases Java permitindo que nese código se inclúan as clases Java so servidor. Nunha chamada a un método dunha clase, DWR xera dinamicamente unha versión Javascript da clase AjaxService, invocada a través dun manexador de eventos que xestiona a interacción co servidor. Cando chega a resposta ao cliente invócase unha función *callback* para actualizar o contido do documento. Este método denomínase Reverse AJAX, soportando tres métodos básicos de envío de datos:
  1) **Polling**. O navegador pregunta ao servidor en intervalos regulares se completou a petición.
  2) **Piggyback**. O servidor espera á seguinte petición do navegador para darlle a resposta.
  3) **Comet**. O servidor responde ao navegador de xeito planificado tipo Streaming nunha resposta Http longa.

  Así mesmo, prove de dúas opcións de comunicación remota:
  1) **DWR nativo**. Empregando un superconxunto de JSON onde o motor DW (*engine.js*) manexa as peticións e prepara a execución das chamadas ao servidor.
  2) **JSON/JSONP.** API para JSON que facilita a integración con outros *frameworks* como Dojo, ExtJS ou Jquery.

  No tocante ao tema da seguridade DWR contempla protección específicas contra ataques XSS (en inglés *Cross Site Scripting*) e CSRF (en inglés *Cross Site Request Forgery*).

- ✔ **SAJAX**. (En inglés *Simple AJAX Toolkit*). Ferramenta de código aberto que de xeito análogo a DRW permite realizar chamadas a métodos do servidor en PHP, ASP, Coldfusion, Ruby, Perl, Phyton e outras linguaxes dende Javascript no navegador, sen ter que recargar a páxina. Prove dunha API para cada linguaxe de servidor, por exemplo *Sajax.php*, que se inclúe no código deste para permitir a integración co Javascript do navegador.

- ✔ **GWT**. (En inglés *Google Web Toolki*t). *Framework* de desenvolvemento AJAX dentro de aplicacións Java. Este contorno permite que ao definir unha interface Java se traduza co compilador GWT de xeito transparente a Javascript e HTML. O principal obxectivo deste *framework* é integrar nun mesmo IDE o desenvolvemento da aplicación e da parte de interfaces de usuario con AJAX pero ademais prove doutras funcionalidades como compoñentes HTML dinámicos e reutilizables, protocolos de transferencia XML e JSON, internacionalización i18n, integración con JUnit incluso nas chamadas RPC e con Javascript a través de JSNI. A arquitectura de GWT estruturase nos seguintes elementos:
    1. **Compilador Java a Javascript GWT**. Para aplicacións web xera automaticamente o código Javascript necesario para a interface Java definida.
    2. **Hosted Web Browser.** Motor de execución de aplicacións Java sen traducilas a Javascript a modo de máquina virtual Java.
    3. **Libraría de Emulación JRE**. Contén os principais paquetes Java de uso común soportadas por GWT.
    4. **Libraría de clases de Interfaces de Usuario GWT**. Prove dun conxunto de compoñentes para interfaces de usuario.

A maiores destes *frameworks* existen infinidade de alternativas e librarías para temas específicos ou versións mais ou menos simples de propósito xeral, como:

1) **AjaxAC**. *Framework* PHP que emprega AJAX no cliente e se orienta á reutilización por dispor de clases moi simples.
2) **AJAX .NET Professional**. Libraría AJAX para ASP .NET con funcionalidades básicas para controis de usuario e utilidades de uso xeral.
3) **ATLAS**. Tamén denominado ASP .NET AJAX, integra nun mesmo *framework* un conxunto de extensións para integrar AJAX en .NET, que inclúe a Microsoft Ajax Library .
4) **BAJAX**. Libraría Javascript moi lixeira (<6k), para integrar AJAX da forma máis simple posible.
5) **Taconite**. *Framework* para desenvolvemento AJAX, que automatiza tarefas para xestionar o obxecto XMLHttpRequest ou a creación de contido dinamicamente.
6) **Spry Framework for Ajax**. Libraría Javascript de Adobe para a integración de AJAX con orixes de datos XML, JSON e HTML para linguaxes de servidor como Coldfusion, PHP ou ASP .NET. Spry ofrece tres compoñentes principais: Datos, Widgets e Efectos.

7) **Tacos**. Libraría que proporciona compoñentes, efectos, validacións e funcionamento AJAX para o *framework* Tapestry.
8) **XAJAX**. *Framework* AJAX para desenvolvemento en PHP, que permite dende o navegador chamar a funcións do servidor.
9) **Zephyr**. *Framework* AJAX para desenvolvemento en PHP5 baixo o modelo MVC. Prove dun motor de modelos, soporte para datos adoDB e outras opcións.
10) **ZK**. *Framework* para desenvolvemento de aplicacións Java que busca facer transparente a tecnoloxía Javascript. Os compoñentes da interface de usuario relaciónanse con compoñentes POJO no servidor. Recoméndase a integración con Spring, Toplink ou Hibernate e aporta protección contra ataques XSS, CSRF e DoS. Verase polo miúdo máis adiante.

30.4 RIA PARA MULTIMEDIA E ANIMACIÓNS

Conforme mellorou o ancho de banda das conexións foron en aumento as tecnoloxías RIA destinadas a mellorar as funcións multimedia, gráficos vectoriais, animacións e interactividade. A pioneira destas tecnoloxías foi Flash para posteriormente aparecer Flex, AIR, JavaFX, OpenLaszlo e Silverlight como principais alternativas.

30.4.1 FLASH RIA

A plataforma Flash evolucionou de plug in no cliente (Flash Player) para a visualización de imaxes vectoriais e animacións, cara unha arquitectura RIA para dotar de novas funcionalidades ás interfaces web. Representa a primeira tecnoloxía RIA, sendo o marco que engloba diferentes tecnoloxías dentro do que se denomina Flash RIA. Dentro das Flash RIA atópanse as tres tecnoloxías principais soportadas por Adobe, e anteriormente Macromedia, as cales teñen o uso máis estendido:

1. **Flash**. Empaqueta de aplicación en arquivos SWF a modo de compoñentes.
2. **Flex**. A partir dun servidor de aplicacións JEE realiza a comunicación co SWF permitindo chamar a obxectos do servidor.
3. **AIR**. Para execución de aplicacións Flash no equipo do cliente sen necesidade de navegadores como intermediarios.

Alternativamente, existen outros provedores de tecnoloxías Flash RIA, entre os que se atopan:

1. **OpenLaszlo**. Moi similar a Flash pero cunha linguaxe de programación propia LZX.
2. **SnappMX**. Orientada cara servizos web.
3. **Zulu**. Permite desenvolver aplicacións conxuntamente co estándar XUL e Flash.
4. **XAMLON**. Permite desenvolver aplicacións Flash coa linguaxe de marcado XAML dentro da plataforma .NET.

A orientación principal de Flash a diferenza de AJAX é a de ser un complemento da interface de usuario estendendo determinadas funcionalidades, en especial no campo das animacións e da interacción co usuario. Os contornos de desenvolvemento de Flash están máis orientados cara a edición de animacións que cara o desenvolvemento web, pero foise abrindo camiño no campo da elaboración de widgets, soporte multilinguaxe, efectos 3D, control e validación de formularios. Permite integración con AJAX e Javascript pero dispón dunha linguaxe de *script* propia denominada **ActionScript**. Este *script* de programación orientada a obxectos segue o estándar ECMA-262, implementando E4X (en inglés *ECMAScript for XML*). Opera cun modelo de eventos baseado na especificación DOM, aínda que non o segue completamente. Lánzase nunha máquina virtual específica AMV2 (en inglés *ActionScript Virtual Machine*) aloxada no contorno de execución Flash Player. Por último destacar que permite conectividade con Servizos web e Bases de datos de maneira remota a través da clase *DataProvider*.

Como acontece na maioría de *frameworks* de desenvolvemento a plataforma Flash pode ampliarse con paquetes de librarías de terceiros, entre estas destacan:

a) **SPL** (en inglés *Spelling Plus Library*). Para deseño de editores de texto enriquecido con corrección ortográfica.
b) **Red5**. Servidor Flash de software libre.
c) **Papervision3D**. Motor de xeración 3D de software libre.
d) **As3corelib**. Paquete de librarías ActionScript 3 que contén clases e utilidades de uso común. Inclúe codificadores de imaxe, serialización JSON, APIs para datas, Strings e outros tipos de datos e codificación de chaves MD5 e SHA 1.
e) **SWFObject**. Javascript para incrustar contido Flash en documentos (X)HTML
f) **Tweener**. Para crear animacións e transicións directamente traballando con ActionScript.
g) **Gaia**. *Framework* orientado cara a axilización do desenvolvemento Flash.

30.4.2 FLEX

Flex é a evolución de Flash ampliando o ámbito de desenvolvemento RIA con novas tecnoloxías e formatos. Diferenciase de Flash na súa facilidade de integración con tecnoloxías de linguaxes de servidor o que facilita o uso de patróns de deseño e que emprega MXML (en inglés *Macromedia eXtensible Markup Language*) para definir o aspecto e comportamento das interfaces de usuario. Como Flash soporte a linguaxe ActionScript e a súa plataforma incorpora librarías de compoñentes para interfaces de usuario específicas. Permite integración con outras tecnoloxías do lado do servidor como Servizos Web, REST ou AMF. As aplicacións Flex poden integrarse nun documento HTML de xeito que este pode actualizar dinamicamente a vista e enviar e recibir datos asincronamente co servidor de fondo, de xeito similar a AJAX.

Nas **comunicacións cliente servidor**, Flex no cliente comunícase co servidor vía HTTP, dispoñendo de tres API RPC: HTTPService, WebService e RemoteObject. Non acceden directamente a Bases de datos remotas senón que o fan a través de capas intermedias. A través de HTTPService solicita arquivos JSP ou XML cos datos en formato de variables String, formatos de intercambio XML, E4X ou obxectos ActionScript. No caso de devolver JSON Flex dispón de librarías especializadas para serialización así como para SOAP. A través da API de RemoteObject permite realizar peticións **Flash Remoting** que devolven mensaxes binarias **AMF** (en inglés *Action Message Format*) sobre HTTP. Cando este formato ten aplicación obtense un maior rendemento que noutras tecnoloxías, como JSON ou SOAP.

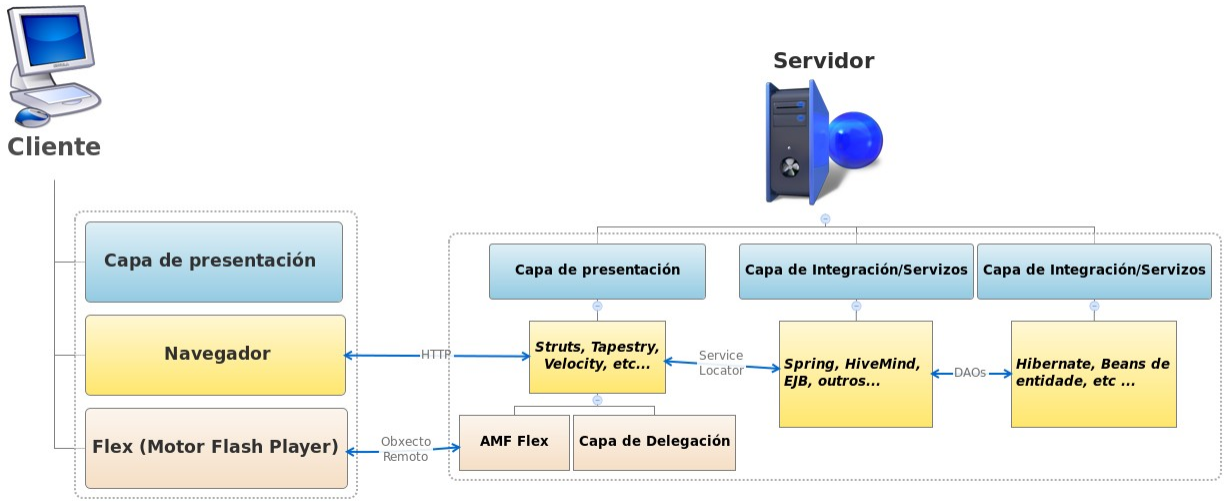


Figura 3: Integración de Flex en JEE.

Así mesmo Flex permite intercambio de datos en tempo real perante dúas vías: **XML Socket** e **Socket Binario**. Co XML Socket créase unha conexión que permanece aberta mentres dure a comunicación, ou é pechada explicitamente. No Socket Binario o funcionamento é similar pero cliente e servidor non precisan intercambiar paquetes XML especificamente senón que envían os datos como información binaria, o que permite conectar con servidores de correo como POP3, SMTP e IMAP ou servidores de novas como NNTP.

No tocante á **seguridade** á hora de integrar Flex nunha aplicación JEE será a arquitectura desta a que impoña o modelo de seguridade, variando dende un *framework* de autenticación/autorización específico a un directorio LDAP, ou arquivos de configuración XML. A información de seguridade debe engadirse aos servizos BlazeDS e LiveCycle Data de xeito que soliciten credenciais nas comunicacións co servidor.

Coma noutras tecnoloxías, en Flex están dispoñibles unha serie de ***frameworks*** multipropósito. Moitos deles tamén son válidos para AIR. Os máis empregados son:

a) **Cairngorm**. Microarquitectura que aplica un pequeno conxunto de patróns de deseño (Service Locator, Front-Controller, ...) probados en conxunto. Céntrase en tres áreas chave: manexar accións de usuario, encapsular as interaccións con servidor e a lóxica de negocio e xestionar o estado do cliente representándoo na Interface de usuario.
b) **Mate**. *Framework* orientado a eventos baseado en etiquetas, implementadas completamente en MXML. Implementa a idea de Inxección de dependencia, construíndo os obxectos para a continuación inxectar nas clases os datos. Os obxectos non solicitan a información pero esta lle é entregada polo sistema.
c) **PureMVCFramework**. Como Cairngorm representa tamén unha microarquitectura cun pequeno conxunto de patróns de deseño, con MVC e Fachada como núcleos centrais, cada unha a través dun patrón instancia única.
d) **Swiz**. *Framework* de control de inversión (Ioc), que prove metodoloxías para simplificar o manexo de eventos e as chamadas asíncronas a procedementos remotos. Emprega MVC pero a diferenza dos anteriores só no que respecta á estrutura de clases e non de directorios.
e) **Parsley**. Conxunto de librarías ActionScript para correspondencia de obxectos e entidades, rexistro de depuración, inxección de dependencia, mensaxería e outras funcionalidades estendidas.

Para a integración con sistemas de información Flex prove do Servizo de xestión de datos dentro do Servizo LiveCycle Data. Neste servizo inclúese sincronización de datos en tempo real entre cliente, servidor e outros clientes, replicación de datos, paxinación baixo demanda, e para aplicacións AIR sincronización de datos locais para conexións ocasionais das aplicacións.

30.4.3 SILVERLIGHT/MOONLIGHT

Complemento para navegadores que permite integrar na mesma extensión elementos multimedia, animacións e interactividade, de xeito similar ao WPF. Atópase baseado en XAML para a definición das interfaces de usuario, a partir das que invoca a métodos do servidor de aplicacións .NET. Permite a carga dinámica de XML co que se pode operar a través de DOM ao xeito de AJAX. Proporciona extensións Javascript e. As principais **características** do *framework* son:

a) **WPF**. Inclúe un subconxunto de WPF que estende en gran medida elementos da Interface de Usuario.
b) **XAML**. Definición da Interface de Usuario a través dunha linguaxe de marcado declarativa.
c) **Integración** con Javascript e ASP .NET AJAX.
d) Acceso a **obxectos do lado do servidor** .NET.
e) Conexión con **servizos de rede** WCF, SOAP e ASP .NET AJAX, permitindo orixes de datos JSON, XML e RSS.
f) Soporta **LINQ** para implementar o acceso a datos

A **arquitectura** de Silverlight se compón de 3 partes fundamentais:

1) **Framework de presentación básico**. Compoñentes e servizos orientados á Interface de usuario e a interacción co usuario, elementos multimedia e soporte XAML.
2) **.NET Framework para Silverlight**. Subconxunto de .NET Framework para Silverlight que contén compoñentes e librarías, recolector de lixo, WCF e CLR. Así mesmo inclúe os controis da Interface de Usuario, XLINQ, RSS/Atom, serialización XML e DLR (en inglés *Dynamic Language Runtime*)
3) **Compoñente de instalación e actualización**. Control de instalación e actualización da extensión.

Mención especial merece o apartado da **seguridade**, como acontecía con outras tecnoloxías RIA Silverlight incorpora políticas de seguridade específicas para:

a) Seguimento do Ciclo de vida de seguridade de Microsoft SDL (En inglés *Security Development) Lifecycle*)
b) Evitación de ataques XSS.
c) Illamento de código de arquivos de configuración XAP.
d) Prover acceso seguro a recursos de rede.
e) Servizos criptográficos para protección de datos de usuario.
f) Sinatura dixital das aplicacións.

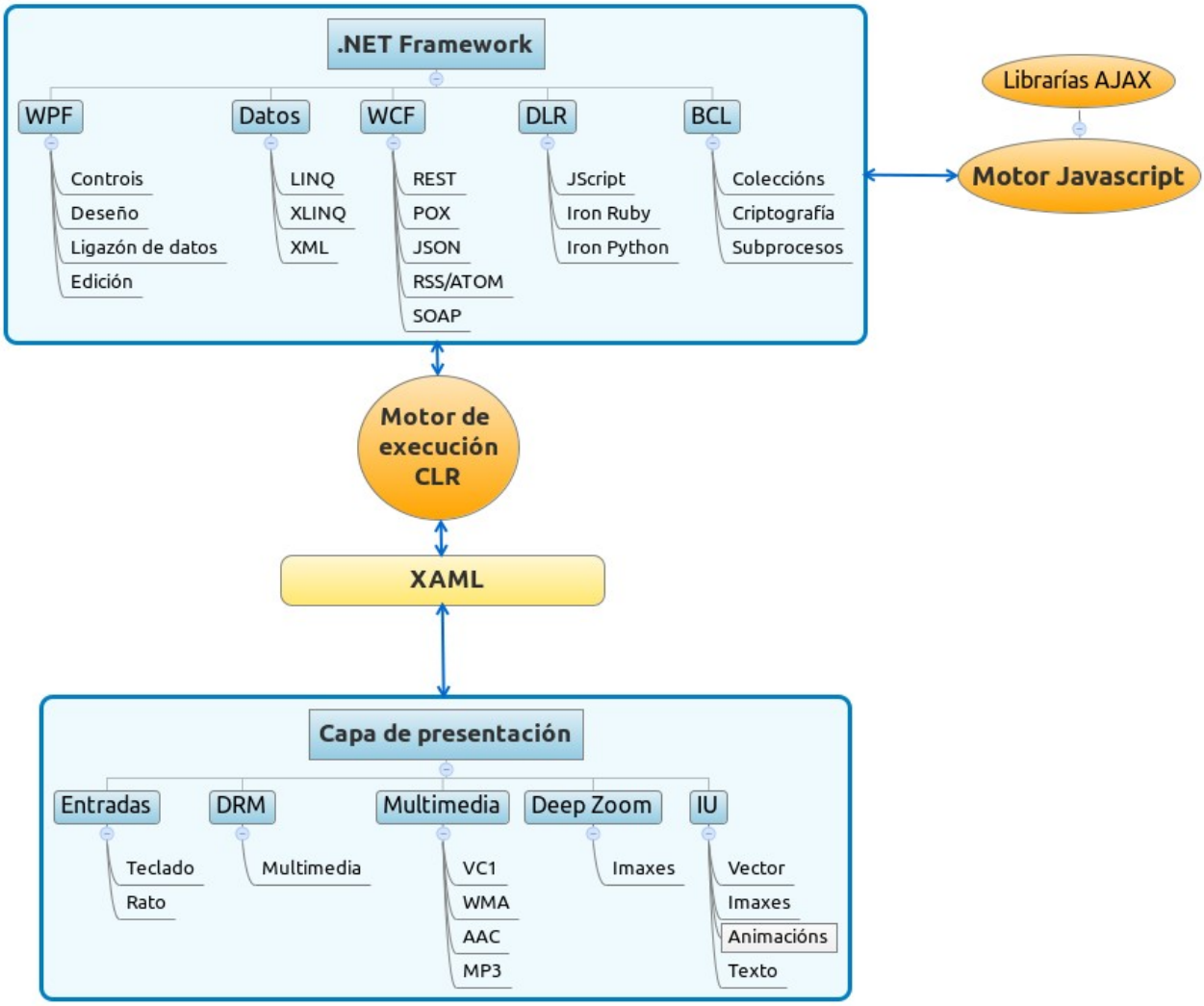


Figura 4: Arquitectura Silverlight.

30.4.4 JAVA FX

Java FX é unha plataforma que se compón de elementos web, multimedia e scripting xunto con tecnoloxías de servidor JEE para o desenvolvemento de aplicacións multiplataforma. Pode funcionar de maneira independente do navegador sempre que teña o equipo instalada unha máquina virtual Java compatible. Promove o concepto de "Perfil común" para tentar unificar todos os dispositivos que soporten JavaFX, de xeito que o mesmo modelo de desenvolvemento se adapte a calquera contorno.

Os principais **compoñentes** de Java FX son:

a) **JavaFX Script**. Linguaxe de programación declarativa, con tipos estáticos, que permite invocar métodos de calquera API de Java da plataforma.
b) **Entorno de execución JavaFX**. Especializado para o dispositivo, Escritorio/Web, Mobile, ou TV.
c) **Aplicacións JavaFX**. Independentes ou empaquetadas como arquivos JAR.
d) **Ferramentas de desenvolvemento**. Inclúe o compilador para JavaFX Script, Plug ins para IDEs como Eclipse, e librarías especializadas para gráficos, multimedia ou Servizos web, entre outros.

A **arquitectura** de JavaFX presenta nun primeiro nivel:

1. As **APIs públicas** de JavaFX. As principais funcionalidades destas APIs son permitir integración con outras linguaxes como JRuby, Groovy e Javascript, funcionalidades xenéricas e extensións para as interfaces de usuario.
2. **Grafo de Escena**. Neste grafo, definido na API *javafx.scene* represéntase nunha estrutura en árbore con nodos representando todos os elementos visuais da interface de usuario. Cada nodo pode levar os seus estilos, ademais de efectos, manexadores de eventos e control de estado.

Baixo eles atópase o motor de execución composto polos seguintes compoñentes:

a) **Prism**. Motor gráfico de alto rendemento que soporta Java2D, OpenGL e DirectX.
b) **Quantum Toolkit**. Xestiona as regras de procesos para representación gráfica fronte ao manexo de eventos.

c) **Glass Windowing Toolkit**. Sistema de control de ventás no nivel máis baixo da arquitectura gráfica de JavaFX. Prové dos servizos operativos nativos do sistema ademais de ser responsable de xestionar a cola de eventos.
d) **Motores web e multimedia**. Inclúen APIs para soporte de medios visuais e de son. Así mesmo o motor web soporta os estándares HTML5, CSS, Javascript, DOM e SVG.

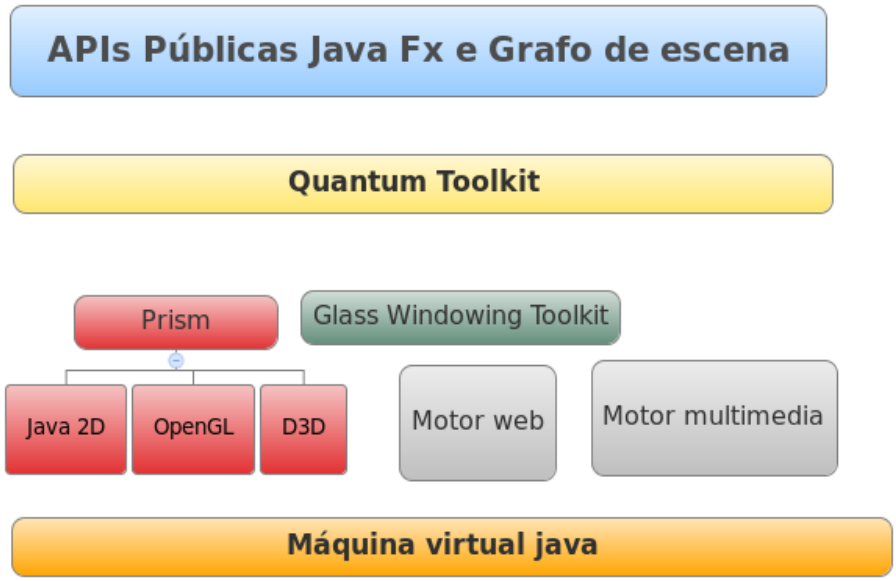


Figura 5: Arquitectura JavaFX.

30.5 OUTRAS TECNOLOXÍAS RIA

Se ben as tecnoloxías con anterioridade representan as solucións de uso máis estendido actualmente, existen outras multipropósito ou máis específicas que buscan facerse un oco no mundo das RIA. Dentro destas outras tecnoloxías, OpenLaszlo merece mención especial, se ben non representa unha tecnoloxía en si, senón un conxunto de tecnoloxías, con algunhas adaptacións específicas.

30.5.1 OPENLASZLO

OpenLaszlo é un *framework* e plataforma de desenvolvemento para aplicacións RIA con licenza GPL, precisando dun servidor propio para o aloxamento das aplicacións desenvolvidas. Unha mesma aplicación definida a través de dunha linguaxe de definición propia pode exportarse a diferentes formatos multinavegador e multiplataforma. As aplicacións OpenLaszlo poden despregarse no servidor de aplicacións da plataforma, denominado **Despregue en modo proxy**, ou ben en modo **Despregue "SOLO"** con independencia do servidor, por norma xeral empaquetada nun arquivo SWF. Outra característica peculiar de funcionamento son as **Librarías dinámicas**, que permiten unha descarga "parcial" da aplicación, para obter unha carga inicial máis rápida, sendo o resto da carga da aplicación baixo demanda. Así mesmo a funcionalidade **Krank** permite cargar as aplicación OpenLaszlo máis rapidamente, realizando un preprocesamento da vista e *scripts* de inicialización.

OpenLaszlo aporta unha linguaxe declarativa propia, o LZX deseñado para describir as interfaces do usuario, ao estilo de XUL e XForms. Esta linguaxe incorpora un *framework* de etiquetas dividido en categorías como: elementos da interface, orixes de datos, efectos multimedia e accións. Emprégase conxuntamente con Javascript/AJAX sendo este último o encargado da interacción co usuario.

A plataforma OpenLaszlo incorpora os seguintes **compoñentes**:

a) **Compilador**. Permite que unha aplicación definida perante a linguaxe declarativa LZX poida transformarse a Flash (SWF) ou DHTML-AJAX. Así mesmo presenta unha serie de módulos específicos para:
   1) **Compilación XML IU**. O compilador transforma as definicións da interface de usuario ao formato de saída especificado para a aplicación.
   2) **Compilación ECMAScript.** Clases e instancias LZX tradúcense a ECMAScript e controladores de eventos, transformándoas a Bytecode.
   3) **Compilación multimedia**. Codifícanse os arquivos en formatos de imaxe e son, así como as fontes TrueType en ficheiros OBJ para SWF ou XML.

b) **Servidor**. Fai as funcións de aloxamento da aplicación e proxy, mantendo comunicación bidireccional cos back-ends a través de JAVARPC ou outros protocolos de servizos web. Así mesmo proporciona servizos de transformación de formatos, mensaxería, *streaming*, encriptación SSL e autenticación.

c) **Contorno de execución ou LCF** (en inglés *Laszlo Foundation Class*). Inclúe compoñentes da interface de usuario, acceso a datos e servizos de rede. A LCF divídese en catro compoñentes principais:
   1) **Data Loader/Binder**. O cargador e encargado de asociar e relacionar os datos. Dirixe o tráfico de datos incluíndo o fluxo de datos cara o cliente.

   2) **Sistema de eventos.** Permite unha programación baseada en eventos ao recoller os eventos detectados no cliente.
   3) **Layout & Animation System**. Sistema de animación e escenario da aplicación que ofrece todos os elementos para a parte gráfica da aplicación así como animacións e efectos.
   4) **Servizos para as aplicacións**. Con funcionalidades extra como contadores, sons, etc...

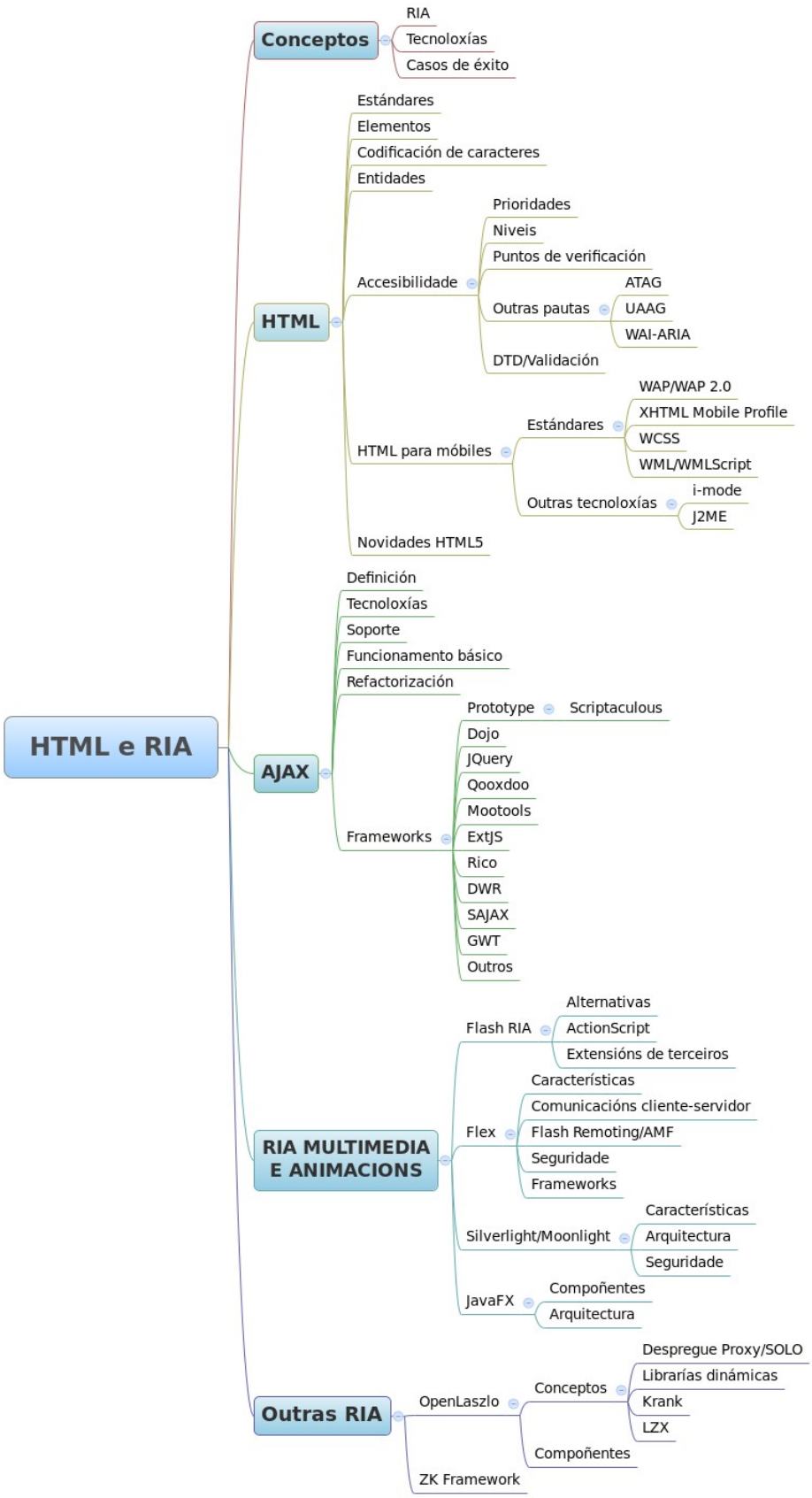
d) **Framework**. Prove dunha extensa API para animacións, estrutura de aplicación, acceso a datos, comunicacións co servidor e definición da interface de usuario. Estruturalmente segue un modelo MVC, pero ampliable ás capas que sexan precisas para cada solución.
e) **Servlet**. Tratase dun compoñente opcional para a aplicación que permite atender e dirixir peticións multimedia ou para servizos web como SOAP, JavaRPC ou XML-RPC. Fai funcións de caché e proxy para priorización e bloqueo de peticións así como de rexistro de trazas e auditoría.

30.5.2 ZK FRAMEWORK

Trátase doutro *framework* orientado a eventos que en esencia poderíase incluír cos *frameworks* AJAX aínda que con algunhas diferenzas. En primeiro lugar emprega unha linguaxe de marcado propia para as interfaces de usuario denominada ZUML que pode mesturarse con outras linguaxes de marcado como XUL e XHTML, ademais de linguaxes de *script* e expresións EL para manipulación de compoñentes e datos. Deseñouse para integración con aplicacións JEE incorporando as capacidades de AJAX en desenvolvementos áxiles e reutilizables.

Prove dun *framework* cunha implementación de ZK Spring, adaptado do *framework* Spring, e librarías con compoñentes e etiquetas JSP con soporte para AJAX. No tocante á seguridade engade protección para ataques XSS, DoS e CSRF, ademais de reforzar a autenticación e os permisos coa incorporación de *frameworks* de terceiros como Spring Security.

30.6 ESQUEMA


HTML e RIA
Conceptos
RIA
Tecnoloxías
Casos de éxito
HTML
Estándares
Elementos
Codificación de caracteres
Entidades
Accesibilidade
Prioridades
Niveis
Puntos de verificación
Outras pautas
ATAG
UAAG
WAI-ARIA
DTD/Validación
HTML para móbiles
Estándares
WAP/WAP 2.0
XHTML Mobile Profile
WCSS
WML/WMLScript
Outras tecnoloxías
i-mode
J2ME
Novidades HTML5
AJAX
Definición
Tecnoloxías
Soporte
Funcionamento básico
Refactorización
Frameworks
Prototype
Scriptaculous
Dojo
JQuery
Qooxdoo
Mootools
ExtJS
Rico
DWR
SAJAX
GWT
Outros
RIA MULTIMEDIA E ANIMACIONS
Flash RIA
Alternativas
ActionScript
Extensións de terceiros
Flex
Características
Comunicacións cliente-servidor
Flash Remoting/AMF
Seguridade
Frameworks
Silverlight/Moonlight
Características
Arquitectura
Seguridade
JavaFX
Compoñentes
Arquitectura
Outras RIA
OpenLaszlo
Conceptos
Despregue Proxy/SOLO
Librarías dinámicas
Krank
LZX
Compoñentes
ZK Framework

31.6 REFERENCIAS

Lee Babin
Beginning Ajax with PHP: from novice to professional. (2007).

Rebecca Riordan
Head First Ajax. (2008).

Michael Mahemoff.
Ajax Design Patterns. Creating Web 2.0 Sites with Programming and Usability Patterns. (2006)