



TESINA DE LICENCIATURA

Título: Patrones RIA en el traspaso a aplicaciones web 2.0. Guía de buenas prácticas y desarrollo de controles de servidor que implementan patrones

Autores: Mariano Covatti, Nicolás Gordillo, Gastón Yarza

Director: Gustavo Rossi

Codirector: -

Asesor profesional: -

Carrera: Licenciatura en Sistemas

Resumen

Debido a la constante evolución de las tecnologías informáticas y las técnicas para el desarrollo de aplicaciones en el Web, surge la necesidad de brindar a los usuarios mayores funcionalidades en un entorno donde la experiencia de usuario toma el protagonismo. Nacen las aplicaciones RIA. Las mismas ofrecen una experiencia sofisticada y atractiva que mejora la satisfacción del usuario y aumenta su productividad. Nuestro objetivo es simplificar el traspaso de las aplicaciones tradicionales a este nuevo tipo de aplicaciones, facilitando la adopción de patrones RIA en la práctica. Dado que éstas son más complejas se necesita plantear el problema desde varios puntos de vista. Para ello generamos una guía del buen uso de patrones RIA, describiendo mejores prácticas, correcto contexto, ventajas, desventajas y posibles problemas al implementarlos. Con esta base, y valiéndonos de los controles de servidor que brinda .Net, encapsulamos la funcionalidad de cada patrón, formando un toolkit que permita al programador utilizarlos de manera sencilla. El prototipo de aplicación web Pandora nos permitió explicar los patrones a través del uso de los controles que implementan los mismos. Nos dio el sustento práctico para mostrar los conceptos estudiados en una aplicación creada bajo las mismas prácticas que describimos.

Palabras Claves

RIA, patrón, Web 2.0, Aplicaciones Web, .Net, Toolkit, experiencia de usuario, User Interface

Trabajos Realizados

Estudio de aplicaciones RIA.

Comparación con aplicaciones tradicionales.

Análisis de patrones RIA existentes.

Desarrollo de toolkit de controles en .Net.

Desarrollo de prototipo de aplicación Web que muestra y utiliza patrones RIA mediante controles del toolkit.

Conclusiones

Los desarrolladores han encontrado soluciones caseras a los distintos problemas que se le presentaban, y se ha hecho muy común la abstracción de estas soluciones en patrones, que describen el problema y la solución de manera explícita. A través del uso de un control que encapsula el comportamiento de un patrón, el desarrollador tiene una poderosa herramienta a su disposición para agregar elementos RIA a las aplicaciones web de forma simple, concentrándose así en el problema de su dominio.

Trabajos Futuros

Extensión del toolkit, incluyendo patrones como drag and drop.

Utilización de los patrones en dispositivos móviles o con capacidad de presentación y procesamiento muy diferente.

Mejorar las metodologías de desarrollo existentes para este tipo de aplicaciones.



Tesina de grado

Patrones RIA en el traspaso a aplicaciones web 2.0. Guía de buenas prácticas y desarrollo de controles de servidor que implementan patrones.

Director:

Rossi, Gustavo

Alumnos:

Covatti, Mariano N° 3094/9

Gordillo, Nicolás N° 5477/1

Yarza, Gastón N° 5455/4

Clasificación:

Investigación Aplicada

Abstract

Debido a la constante evolución de las tecnologías informáticas y las técnicas para el desarrollo de aplicaciones en el web, surge la necesidad de brindar a los usuarios mayores funcionalidades en un entorno donde la experiencia de usuario toma el protagonismo. Nacen las aplicaciones RIA. Las mismas ofrecen una experiencia sofisticada y atractiva que mejora la satisfacción del usuario y aumenta su productividad.

En estos tipos de aplicaciones se observan patrones, los cuales describen soluciones óptimas a problemas comunes dentro de contextos específicos. Estos problemas comunes involucran a toda una comunidad y se resuelven, generando soluciones comunes.

Nos proponemos generar una guía del buen uso de patrones RIA, describiendo los principales patrones, sus mejores prácticas, su correcto contexto, ventajas, desventajas y posibles problemas al implementarlos.

Se propone también el desarrollo de controles de servidor en C# .Net, que implementen patrones RIA, y que conformen en su conjunto un toolkit disponible para facilitar el traspaso de aplicaciones web tradicionales a aplicaciones RIA.

Índice

Abstract	1
Índice	3
Capítulo 1. Introducción a la Web y nacimiento de RIA	6
1.1. Introducción	6
1.2. Tecnologías de desarrollo de aplicaciones ricas de internet	9
Java Applets.....	9
JavaFX	9
Microsoft SilverLight	10
AJAX.....	10
Adobe Flex.....	11
HTML 5	12
Capítulo 2. Componentes reusables	14
2.1. Ventajas aportadas por la librería de controles.....	14
Reusabilidad	14
Mantenibilidad	14
Comprensión	14
Flexibilidad	14
Estabilidad	15
Complejidad	15
Integridad	15
2.2. Controles ASP.NET.....	15
Controles de usuario (User control).....	16
Controles personalizados (Custom control).....	17
Web Parts.....	17
2.3. Controles en Java	17
Capítulo 3. Patrones RIA	19
3.1. Editado “In-Page”	20
3.1.1. Single Field In-Line Edit	20
3.1.2. Multi Field In-Line Edit	22
3.1.3. Overlay Edit	23
3.1.4. Table Edit.....	25

3.1.5. Group Edit	26
3.1.6. Rating an object.....	29
3.2. Drag and drop.....	30
3.2.1. Drag and drop de módulos.....	31
3.2.2. Drag and drop de listas	32
3.2.3. Drag and drop de objetos.....	34
3.2.4. Drag and drop de acciones.....	35
3.2.5. Drag and drop de colecciones	37
3.3. Selección directa	39
3.3.1. Selección de ítems.....	39
3.3.2. Selección de objetos.....	40
3.4. Superposición (overlay).....	42
3.4.1. Tooltip	42
3.4.2. Diálogos.....	43
3.4.3. Pop ups de entrada	45
3.4.4. Date/Time Picker.....	46
3.5. Inlays	47
3.5.1. Accordion	47
3.5.2. Tabs	49
3.6. Transicionales.....	51
3.6.1. Fade	51
3.7. Patrones de búsqueda.....	52
3.7.1. Autocomplete.....	52
3.7.2. Live Suggest.....	54
3.8. Feedback	56
3.8.1. Live Preview.....	56
3.9. Validación	58
3.9.1. Validación en línea	58
3.9.2. Contador de caracteres	59
Capítulo 4. Patrones RIA aplicados. Implementación mediante controles	61
4.1. Autocomplete.....	61
4.2. Inline validation	63
4.3. Control Fade	65
4.4. Rating an object.....	66

4.5. Character count.....	69
4.6. Control Accordion	70
4.7. Control Tooltip	72
4.8. Control DatePicker	73
Capítulo 5. Caso de Estudio: Pandora. Website del toolkit	75
5.1. Introducción	75
5.2. Estructura del Website.....	75
5.3. Explicación de los patrones aplicados en el website	76
Capítulo 6. Conclusión y trabajos futuros	82
6.1. Conclusión	82
6.2. Trabajos Futuros.....	83
6.2.1. Metodologías de aplicación de patrones RIA.....	83
6.2.2. RIA en aplicaciones móviles	89
Glosario.....	92
Referencias	93

Capítulo 1. Introducción a la Web y nacimiento de RIA

1.1. Introducción

Desde sus inicios por el año 1991, la Web (World Wide Web), ha evolucionado constante y rápidamente. Ha pasado de ser un sistema de transferencia de documentos de hipertextos para convertirse en una plataforma de desarrollo y despliegue de aplicaciones complejas; gracias al desarrollo en los servidores, protocolos, especificaciones, lenguajes, tecnologías de desarrollo, entre otros.

La aparición de Internet definió un nuevo modelo de desarrollo y despliegue, el cual se agrega al modelo existente previamente que abarcaba las aplicaciones de escritorio, y que es el de las aplicaciones Web.

El primer modelo se caracteriza por un total acceso a los recursos del sistema donde se ejecutan, interfaz de usuario rica, alta capacidad de interactividad con el usuario, gran velocidad de respuesta, programación basada en eventos. Por otro lado, es necesario instalar cada copia a utilizar en las estaciones de trabajo que correspondan, y el acceso desde diferentes puntos geográficos se vuelve más complejo.

Por el contrario, las aplicaciones Web solo necesitan un navegador para ser utilizadas desde cualquier punto con conexión a Internet, permitiendo de esta manera la ubicuidad de acceso. Como contraparte, el protocolo HTTP en el cual estas aplicaciones se sustentan es limitado por lo que la capacidad de respuesta e interactividad se ve disminuido. Sumado a esto, el cliente carece de inteligencia, ya que necesita ejecutarse a través del navegador y no se permite acceder libremente a los recursos del sistema.

Es por este motivo que, para las aplicaciones orientadas al consumidor, este modelo ha resultado confuso y propenso a errores incluso teniendo en cuenta aplicaciones sencillas.

Sin embargo, este nuevo modelo ha permitido a las organizaciones disminuir los costos relacionados al desarrollo de sistemas, pero los usuarios de los mismos reclaman cada vez con mayor énfasis que éstos contengan la misma funcionalidad e interacción que las aplicaciones de escritorio. Es por este motivo que las empresas involucradas han destinado recursos a mejorar la parte cliente de sus aplicaciones, haciendo uso de patrones de diseño y buenas prácticas que dieron lugar a una nueva clase de aplicaciones, las Aplicaciones Ricas de Internet.

El término Aplicaciones Ricas de Internet (Rich Internet Applications o RIA) fue originalmente introducido por la empresa Macromedia (en la actualidad Adobe Systems, ver ref. [1]) en el año 2002. Refiere a las aplicaciones con capacidades de interacción similares a las de escritorio, utilizando para tal fin una capa intermedia que evita la necesidad de “refrescar la página” por completo tal como hacen las aplicaciones Web tradicionales. En este nuevo modelo, los clientes contienen mayor inteligencia y la interacción con el servidor se reduce a la transferencia de datos, mostrando así una interfaz y experiencia de usuario enriquecida.

Entre los beneficios que presentan este tipo de aplicaciones se puede mencionar (ref. [8]):

- Riqueza. La interfaz de usuario puede incluir funcionalidades como “drag and drop” que no existían en las aplicaciones Web tradicionales, se pueden realizar cálculos

perfectamente en el cliente sin tener que interactuar con el servidor, realizar validaciones complejas de reglas de negocios, o mostrar animaciones de los componentes de la interfaz, efectos, etc.

- Balance Cliente/Servidor. Se realiza un mejor balance de recursos entre el cliente y el servidor, y como consecuencia el servidor puede atender más solicitudes de forma concurrente. El tráfico se reduce significativamente debido a que el cliente solicita al servidor solamente los datos necesarios para actualizar la interfaz de usuario, debido a lo cual se reduce notablemente el tamaño de los mensajes enviados entre ambos. Por esto las solicitudes individuales y las respuestas asociadas se aceleran de forma notable.
- Comunicación asincrónica. El cliente puede interactuar con el servidor de forma asincrónica, esto es, al enviar un mensaje al servidor no tiene necesidad de esperar por la respuesta del mismo para seguir realizando las acciones que correspondan. Así la aplicación se mantiene con un elevado nivel de respuesta.

El éxito de este tipo de aplicaciones se sustenta en factores como:

- Alto rendimiento y entorno de ejecución multiplataforma.
- Soporte integrado para texto, gráficos, audio y video.
- Acceso a fuentes de datos empresariales.
- Soporte para clientes desconectados.
- Seguridad y confiabilidad.
- Entorno de programación familiar (uso de lenguajes orientados a objetos y lenguajes de etiquetas como HTML y XML).
- Aprovechamiento de las arquitecturas existentes.
- Uso de protocolos estándares y API's (HTTP, XML, SOAP, J2EE, .NET)
- Uso de Patrones de Diseño.
- No hay necesidad de instalación en el cliente.
- Actualización automática a nuevas versiones.

Actualmente, se presentan varios problemas o desafíos al desarrollar aplicaciones Web ricas. Todas estas características que realzan la experiencia del usuario traen aparejadas una serie de complicaciones que se deben tener en cuenta para lograr buen suceso en el desarrollo de las mismas.

El primer desafío es el acoplamiento. Las aplicaciones tradicionales son fuertemente acopladas, ya que utilizan una conexión permanente entre cliente y servidor. Su contrapartida son las aplicaciones Web tradicionales que son débilmente acopladas y utiliza clientes livianos. En RIA, se busca lograr una alta interactividad y un fuerte acoplamiento para mejorar la experiencia del usuario, pero debe correr sobre una configuración débilmente acoplada. Esto requiere de desarrollo más complejo para la concreción del objetivo.

Otro problema es el límite existente en el ancho de banda ya que las redes actualmente no están preparadas para soportar conexiones fuertemente acopladas. Para solucionar esto, las aplicaciones deben soportar más lógica del lado del cliente, disminuyendo de esta forma la comunicación necesaria con el servidor y quitándole cierto trabajo al mismo.

También hay un desafío en el grupo humano que desarrolla la aplicación, ya que la interacción con el usuario generalmente se inicia en los equipos de diseñadores, mientras que los desarrolladores, poco acostumbrados a este tipo de tarea, tienen que lidiar con aspectos visuales mezclados con aspectos de programación de la lógica de negocio en el cliente. Por otro lado, los diseñadores no están habituados a visualizar el sistema de forma integral y suelen encontrarse con dificultades al hacer su trabajo.

Se hace indispensable una coordinación entre estas áreas, y un buen diseño del sistema en capas teniendo en cuenta estos aspectos. Aun así, el aumento de complejidad aumenta costos y dificulta la mantenibilidad de las aplicaciones. De la misma forma, el análisis de rendimiento y el diagnóstico de errores se complica ya que cierta parte del sistema se ejecuta en el cliente, administrado por su propio sistema operativo y utilizando sus recursos.

Desde el punto de vista de la ingeniería de software, se requiere un enfoque distinto al tradicional, que contemple los desafíos de este tipo de aplicaciones. Los procesos de desarrollo de aplicaciones deben ser actualizados para llevar a cabo implementaciones efectivas, desde la dinámica del grupo de desarrollo hasta el modelado de las nuevas características de la interfaz.

Un tema importante a considerar es que las aplicaciones RIA no reemplazan a los sitios Web. Se debe evaluar en qué casos es conveniente utilizar este tipo de aplicaciones, ya que no se justifica reemplazar una aplicación web convencional cuando el objetivo de esta es la intención original con la que fue creada la web.

En estos casos la Web ya se encuentra preparada y ofrece ciertas funcionalidades que se disponen sin realizar ningún esfuerzo de programación, como shortcuts, copiado y pegado, o bookmark de páginas web. Si bien es posible reemplazar todas estas funcionalidades utilizando RIA, se necesita llevar a cabo el desarrollo de las mismas.

En cambio, las aplicaciones tradicionales que utilizan texto principalmente no son las apropiadas cuando se quiere formas de navegación avanzada, o visualización y análisis de datos. Con el nuevo enfoque se mejora la experiencia del usuario, proveyendo continuidad de contexto ya que no es necesario recargar la página en su totalidad cada vez que el usuario requiere algún dato o vista que se lleve a cabo mediante alguna comunicación con el servidor.

Esto hace que el usuario se sienta más cómodo con la aplicación y se concentre más en sus tareas, ganando de esta manera en productividad. Básicamente no existen tantos baches en su atención ya que no está obligado a esperar por una nueva recarga.

Otro tipo de aplicaciones que mejora significativamente con el uso de RIA son aquellas donde se hace un uso intensivo de datos. En este caso la carga asíncrona de datos permite que la página pueda ser mostrada con algunos datos, y mientras el usuario los visualiza la aplicación siga con la carga del resto.

En una aplicación tradicional el usuario debería esperar a que la totalidad de los datos sean transferidos al cliente, para poder visualizar los mismos.

Incluso el cliente puede almacenar los datos y realizar tareas más complejas, como aplicar filtros a los mismos. De esta manera el usuario tiene la sensación de tener a su disposición los

datos como si éstos estuvieran guardados localmente en su máquina, cuando de hecho los datos reales siguen en el servidor.

Otra ventaja es la reproducción de contenido multimedia dentro de un sitio que hace la experiencia de usuario de este tipo de aplicaciones mucho más inmersiva. No hay necesidad de abrir un reproductor por separado sino que el contenido es visible directamente en la página.

Asimismo se pueden aprovechar las capacidad del cliente para realizar tareas complejas como animaciones para mejorar las transiciones entre diferentes estados de la aplicación, de manera de que la atención del usuario sea llevada gradualmente donde debe ir.

1.2. Tecnologías de desarrollo de aplicaciones ricas de internet

Este tipo de aplicación no está relacionada con una tecnología en particular, en la actualidad existen diversas soluciones y cada una de ellas aporta una capa intermedia diferente que se ejecuta en el navegador web. Entre las más populares se encuentran:

Java Applets

Un Applet es un programa escrito en lenguaje Java que puede ser incluido en una página HTML, ejecutado en el contexto de un navegador del cliente. No tiene acceso a todos los recursos del cliente, a menos que se otorguen explícitamente. Son multiplataforma y soportan la mayoría de los browsers, sin embargo a pesar de que el lenguaje se hizo popular por la inclusión de los mismos, no se utilizan ampliamente debido al tamaño del componente que es necesario instalar en el cliente (el entorno de ejecución de java, JRE) y aunque se intenta reducir el tamaño del JRE estos no han dado resultado. Además, el proceso de asignación de permisos por parte del cliente no es una tarea fácil.

Java Swing es una biblioteca gráfica para Java, es una tecnología madura y robusta, y puede ser utilizada junto con los Applets tanto como con aplicaciones de escritorio. Aporta un amplio conjunto de componentes con una apariencia visual independiente de la plataforma, sin embargo carece de una experiencia de usuario atractiva, lo cual ha limitado su aceptación en lo que refiere a las aplicaciones web.

JavaFX

JavaFX es un conjunto de productos y tecnologías desarrollados por Sun Microsystems cuyo objetivo es puntualmente la creación de Aplicaciones Ricas de Internet (RIAs) que pueden ejecutarse en una variedad de dispositivos conectados como celulares, TV, computadoras, reproductores de Blu-Ray o consolas de juegos. Se encuentra orientado a las interfaces totalmente animadas, en lugar de orientarse a la Web como lo hacen sus competidores. Se integra al JRE en los equipos de escritorios y en los dispositivos móviles al J2ME ya que es compilado a código Java.

La idea de JavaFX es simplificar el desarrollo requerido por Java a la hora de desarrollar este tipo de aplicaciones. Utiliza un lenguaje declarativo y estáticamente tipado. Actualmente es gratuito y está compuesto por un kit de desarrollo de software (SDK) que incluye el compilador y las herramientas de tiempo de ejecución. El desarrollo puede realizarse utilizando NetBeans, un entorno muy completo para la creación de aplicaciones; o también utilizando un

complemento para el entorno Eclipse que permite realizar esta tarea. Por otro lado existe un conjunto de complementos para las herramientas gráficas de Adobe que permiten exportar los gráficos creados en éstas al formato reconocido por JavaFX.

Microsoft SilverLight

Silverlight es una tecnología desarrollada por Microsoft que incluye un subconjunto de la tecnología WPF (Windows Presentation Foundation) y extiende los componentes del navegador para la creación de interfaces de usuario que incluyan gráficos 3D, audio, video y otros elementos multimedia, más allá de lo que se puede hacer solamente con HTML tradicional. Con este producto es posible cargar contenido XML dinámicamente y trabajarlo mediante la interfaz DOM. Soporta varios lenguajes de programación incluidos los pertenecientes a la plataforma .NET.

Para la creación de interfaces utiliza un lenguaje declarativo basado en XML llamado XAML. Esto generó la preocupación de grupos relacionados con los estándares de Internet que alegan que de esta manera puede haber ciertos sitios en la Web que pueden no ser accedidos sin los productos que Microsoft ofrezca. En la actualidad el complemento necesario para visualizar este contenido es gratuito, y es una versión reducida del framework de .NET. Incorpora extensiones al lenguaje Javascript para que el navegador pueda interactuar con los controles WPF, tiene soporte para la mayoría de los navegadores y en la mayoría de las plataformas (Microsoft soporta Windows y MAC OS, y en Linux se hace a través del proyecto Moonlight).

AJAX

AJAX (o Asynchronous JavaScript and XML) es una técnica que permite el desarrollo de aplicaciones Web interactivas que surge de la unión de varias tecnologías (ver ref. [5] y [12]). La idea principal es que las páginas se puedan comunicar con el servidor sin necesidad del tradicional “refresco de página”, realizando diversas tareas en background y ayudando de esta manera a mejorar la experiencia de usuario. Bajo ese término se agrupan varias bibliotecas, kits de desarrollo y conjuntos de componentes como:

- Presentación basada en estándares como XHTML y CSS
- Presentación dinámica e interactiva utilizando DOM
- Intercambio y manipulación de datos utilizando XML y XSLT
- Pedido asíncrono de datos a través de XMLHttpRequest
- Javascript como lenguaje en el cliente

AJAX introduce el concepto de la página web como aplicación, ya que el código en lenguaje JavaScript se transfiere con la página y permite cierta lógica del lado del cliente, permitiéndole detectar la ocurrencia de eventos como selección con el mouse, movimientos del mouse por encima de un objeto, uso del teclado, entre otros, y actuar en consecuencia.

De esta forma se puede responder instantáneamente a las acciones generadas por los usuarios y actualizar continuamente o a demanda los datos mostrados, mediante llamadas asincrónicas al servidor, aumentando la interactividad y capacidad de respuesta de la aplicación.

Ajax introduce un intermediario entre el usuario y el servidor. El motor de Ajax se encarga de mostrar la página, y cualquier acción del usuario que normalmente hubiese necesitado de un

requerimiento HTTP se convierte a una llamada en Javascript al motor de Ajax. Si el motor necesita alguna información del servidor, envía un mensaje asíncronico, sin que el usuario note una interrupción en la ejecución de la aplicación.

Por supuesto, las ventajas que provee esta técnica también traen aparejadas algunas complicaciones:

- La integración entre diferentes tecnologías hace más compleja la tarea de desarrollar la aplicación mientras que la depuración es más engorrosa. Las herramientas de desarrollo de JavaScript son limitadas debido a la naturaleza dinámica del lenguaje. Aunque hay herramientas que permiten programar en otros lenguajes, el producto final siempre es DHTML/JavaScript.
- El tiempo de descarga inicial de la aplicación aumenta proporcionalmente a la complejidad de la misma, ya que la cantidad de código que debe ser transferido hacia el cliente es mayor. Esto puede desconcertar al usuario.
- El código fuente de la aplicación puede ser revelado utilizando las facilidades del navegador, y además, al ser una página HTML reside en el cache del mismo; esto obliga a los desarrolladores a utilizar técnicas para proteger el código.
- AJAX no permite “server push”. La aplicación AJAX debe encuestar periódicamente al servidor para obtener datos, aun cuando no haya datos disponibles.
- Se pierde el concepto estándar de volver a la página anterior, presente en cualquier Web.
- Es posible que Ajax no funcione en todos los dispositivos, como celulares o Tablets.
- Ajax no es compatible con todos los software para discapacitados.

Todo esto hace que la selección de AJAX como estrategia para el desarrollo de este tipo de aplicaciones sea cuidadosamente planeada. De cualquier forma hoy es una técnica bien conocida, estable y madura, y es utilizada ampliamente en la Web.

Adobe Flex

Este nombre agrupa diferentes tecnologías basadas en las funcionalidades del reproductor de Flash, una máquina virtual para multimedia de alto rendimiento multiplataforma (ver ref. [38]). Este software es propietario, pero para su reproducción necesita de un plugin gratuito, que se encuentra instalado en la mayoría de las computadoras conectadas a la Web. Este reproductor se encarga de ejecutar el código en el lado de cliente, interactuar con el código JavaScript y HTML en el navegador y a su vez con las tecnologías del lado del servidor.

Hasta el surgimiento de Flex, crear contenido en Flash siempre había sido reservado a diseñadores y, en la mayoría de los casos, estaba relacionado con la multimedia. Se veía en éstos un uso extensivo de recursos de audio, video, gráficos y texto mezclados, animaciones, efectos y transiciones. Las herramientas existentes para esto no estaban dirigidas a los desarrolladores, por esto conceptos como “línea de tiempo”, “marco” y “escenario” (time line, frame y stage) eran prácticamente desconocidos para ellos.

Flex fue creado específicamente para el desarrollo de Aplicaciones Ricas de Internet, utiliza lenguajes orientados a objetos y/o declarativos basados en estándares para el desarrollo y despliegue de aplicaciones web escalables.

De esta forma se vuelve más amigable a desarrolladores, agregándole la posibilidad de utilizar este tipo de lenguajes de programación y a su vez utilizando para su ejecución el mismo reproductor que Flash. De cualquier forma, es posible que aplicaciones similares realizadas en Flex tengan un rendimiento menor que si fueran desarrolladas en Flash.

En cuanto a Ajax, la principal diferencia entre ambos es la carga de contenido parcial. En Ajax el contenido no se embebe en un control Active X, con lo cual no necesita que inicialmente se termine de cargar por completo antes de que el usuario pueda empezar a interactuar con la aplicación.

La plataforma está compuesta por:

- Lenguaje declarativo MXML, basado en XML para la creación de las interfaces de usuario.
- ActionScript, un lenguaje orientado a objetos con el cual se programa la lógica de las interfaces de usuario.
- El reproductor de Flash que se ejecuta como complemento del navegador.
- El kit de desarrollo de Flex, que incluye el framework, compiladores y depuradores.

HTML 5

La versión 5 de este lenguaje de marcas estándar en la Web pretende incluir ciertos tags especialmente enfocados en mejorar la experiencia del usuario y la facilidad de su desarrollo por parte de los desarrolladores.

El desarrollo de esta nueva versión no interrumpe de ninguna manera lo desarrollado en versiones anteriores lo cual facilita el traspaso gradual de sitios en funcionamiento a la nueva generación de aplicaciones. De hecho uno de los objetivos que se tuvo en cuenta al crear esta nueva versión fue mantener la compatibilidad y establecer claramente un camino para migrar aplicaciones convencionales a este nuevo formato.

Otros objetivos de esta nueva versión incluyen un mejor manejo de errores, menor dependencia en relación a plugins externos, reemplazo de scripts en cliente por tags en los casos en que sea posible y logro de independencia de dispositivos.

Todo esto en un ambiente de colaboración global donde cualquier persona relacionada a la Web pueda participar del proceso, y donde cada funcionalidad o característica propuesta se encuentre justificada por un uso práctico de la misma.

El resultado se traduce en mayor soporte para almacenamiento local, etiquetas destinadas a reproducción de video y audio, y una gran variedad de tags y controles creados para facilitar el trabajo del desarrollador.

Actualmente no es oficialmente un estándar, sin embargo los principales browsers del mercado ya ofrecen soporte dentro de cada uno de sus productos. Muchas funcionalidades propuestas por este lenguaje de marcado han sido agregadas a los navegadores de forma individual, pero existen hoy en día implementaciones completas.

Existe un muy buen documento que explica este lenguaje en [21], que puede ser utilizado tanto como introducción como así también para entender la justificación de la necesidad de este cambio.

Capítulo 2. Componentes reusables

Para comprender las ventajas que obtenemos con la utilización de una librería de controles, se refrescan a continuación los conceptos básicos sobre los que basaremos nuestras explicaciones. Veremos que, independientemente del paradigma utilizado, hay conceptos de los cuáles se puede sacar provecho, para hacer de la experiencia del desarrollador una tarea más fácil, que permite a su vez concentrar sus esfuerzos en otras áreas y lograr soluciones en menor tiempo (ref. [2]).

2.1. Ventajas aportadas por la librería de controles

Reusabilidad

Esta característica de los programas nos indica la posibilidad de utilizar código ya escrito en otras aplicaciones, sin necesidad de realizar grandes cambios al mismo.

La utilización de los conceptos antes mencionados, como abstracción, encapsulamiento y modularidad, nos permiten reutilizar código con facilidad, reduciendo de esta manera en gran medida el tiempo y esfuerzo del desarrollo.

En el caso de la programación orientada a objetos, ya que los mismos son independientes uno de otro, con sus estructuras y procesos incluidos dentro de sí mismos, realizar una aplicación podría limitarse únicamente a relacionar objetos ya programados de una manera particular.

Mantenibilidad

Esta cualidad indica que un programa o sistema es fácilmente modificable. Es decir que los cambios en las condiciones externas implicarán modificaciones pequeñas en el programa /sistema. El concepto de mantenibilidad implica que un programa, al igual que un ser vivo debe ser capaz de adaptarse a un medio ambiente siempre cambiante.

Relacionando este tema con nuestra librería de controles, podemos decir que al desarrollar y compilar los controles de forma independiente se logra una mayor mantenibilidad ya que los cambios en el programa no afectan a los controles.

Comprensión

Tanto los datos que componen los objetos, como los procedimientos que los manipulan, están agrupados en clases, que se corresponden con las estructuras de información que el programa trata.

Es más comprensible para el desarrollador tener por separado la implementación de los controles cuando se realiza un sitio web, ya que de esta manera puede focalizar su atención en un subconjunto del problema, y no en el problema completo.

Flexibilidad

Al tener relacionados los procedimientos que manipulan los datos con los datos a tratar, cualquier cambio que se realice sobre ellos quedará reflejado automáticamente en cualquier lugar donde estos datos aparezcan.

De esta manera, si cambiamos la implementación de algún control, todas las aplicaciones que lo utilicen se actualizarán sin que el desarrollador tenga que hacer ningún cambio importante.

Estabilidad

Dado que permite un tratamiento diferenciado de aquellos objetos que permanecen constantes en el tiempo sobre aquellos que cambian con frecuencia permite aislar las partes del programa que permanecen inalterables en el tiempo.

Complejidad

Al elevar el nivel de abstracción se disminuye el nivel de complejidad de un sistema. Como se explicó anteriormente, tener separados los controles del resto de la lógica del sitio Web nos permite disminuir la complejidad del mismo, así como también disminuye la complejidad a la que el desarrollador debe hacer frente al abordar el sistema.

De esta forma es posible modelar sistemas de mayor tamaño con menor esfuerzo.

Integridad

Constituye un mecanismo de integridad ya que la dispersión de un fallo a través de todo el sistema es menor, puesto que al presentar una división entre interfaz e implementación, los fallos internos de un objeto encuentran una barrera en el encapsulamiento antes de propagarse al resto del sistema.

Estas características han contribuido ampliamente a la buena reputación de la programación orientada a objetos, ya que es en este paradigma donde encuentran su mayor expresión.

2.2. Controles ASP.NET

En la programación actual, los conceptos de herencia y encapsulamiento son fundamentales a la hora de generar software confiable y de calidad. Estos conceptos son la base de uno de los paradigmas más utilizados hoy en día, la programación orientada a objetos.

Una forma de encapsular nuevas funcionalidades y sacar provecho de la reutilización de código en el framework orientado a objetos ASP.NET es utilizar un determinado tipo de objeto llamado control de servidor. En este framework la encapsulación es conocida como la composición o la creación de controles compuestos.

La creación de un control compuesto se realiza mediante la combinación de controles de servidor como controles hijos, pero vistos como una sola entidad. Esto promueve el ocultamiento de información y facilita el desarrollo al permitir al desarrollador del control compuesto centrarse en la funcionalidad combinada del control padre sin tener que preocuparse sobre la configuración de los controles hijos.

Los controles creados permiten simultáneamente mejorar la calidad de las aplicaciones, hacer al desarrollador más productivo y perfeccionar las interfaces de usuario.

ASP.NET proporciona tres métodos diferentes para la creación de controles compuestos: la composición a través de controles de usuario, la composición a través de controles de servidor personalizados y los webparts, éstos últimos explicados en menor medida (ver ref. [32]).

Controles de usuario (User control)

Los controles de usuario son una forma de control compuesto que puede utilizarse para encapsular funcionalidad, tal como código HTML y otros controles de servidor. En general, el enfoque en el desarrollo de un control de usuario es encapsular la lógica de negocios, y así ser usados dentro de una aplicación. Son ideales cuando la reutilización rápida y eficiente de la interface de usuario es el objetivo principal. Si la interface y la funcionalidad son lo suficientemente independiente, el control de usuario podrá ser utilizado en varias aplicaciones distintas.

De los dos principales medios de creación de controles en ASP.NET, los controles de usuario son el tipo de control más fácil de crear. Los desarrolladores de ASP.NET normalmente buscan el control de usuario como la primera opción para la creación de controles, debido a su facilidad de construcción y la simplicidad. La construcción de un control de usuario refleja de cerca las técnicas de construcción y detalles de un formulario web. Estos siguen el estilo de desarrollo de los formularios web a través del drag and drop de controles de servidor ya existentes en el framework y una superficie de diseño totalmente editable en el IDE, esto permite crear con facilidad controles con elementos de interfaz de usuario complejos. Además también soportan la separación de la lógica de presentación como son las etiquetas HTML y la interfaz de usuario, de la lógica de negocios en archivos separados.

Los controles de usuario son archivos del servidor web que pueden ser referenciados en páginas web o en otros controles de usuarios.

Algunas características de los controles de usuario son:

- Los controles de usuario son una gran forma de paquete de HTML y modularizar el desarrollo web para la lógica específica de la aplicación. También son una gran manera de sustituir la inclusión de archivos HTML.
- Los controles de usuario soportan propiedades y métodos que se pueden establecer ya sea en el HTML como atributos o en la página que posee el código de lógica de negocios.
- Los controles de usuario pueden ser almacenados en la memoria caché de ASP.NET, acelerando el rendimiento de la aplicación web.
- Algunas etiquetas no están permitidas dentro de un control de usuario, ya que los controles de usuario están alojados en los formularios web, los cuales ya poseen estas etiquetas. Específicamente de deben aparecer las etiquetas <html>, <head>, <body>, <form>. El uso de estas etiquetas pueda interferir con el correcto funcionamiento del formulario web.
- La directiva de declaración del control de usuario debe aparecer entre la apertura y el cierre del tag <form> del formulario web para garantizar el correcto funcionamiento.

Comparada a la construcción de controles de servidor personalizados, la construcción de controles de usuario es más fácil en términos de la curva de aprendizaje del equipo de desarrollo. Cualquier desarrollador capaz de construir un formulario web es capaz de crear controles de usuario. A pesar de esto, tienen grandes limitaciones. No se puede heredar de otros controles de ASP.NET, mientras que un control personalizado sí puede. La segunda

mayor limitación es la reusabilidad ya que no se pueden compartir entre procesos o sitios Web.

Controles personalizados (Custom control)

Los controles de servidor personalizados son la otra opción para el desarrollo de controles en ASP.NET, encapsulando funcionalidad mediante la herencia, la composición, o ambas cosas. Los controles de servidor personalizados no permiten el desarrollo con el estilo drag and drop como ocurre en el desarrollo de un control de usuario. Todo lo que se representa en un control personalizado es desarrollado mediante programación y es guardado dentro de una clase de código. La obligación de poner todo (diseño de la interfaz de usuario, funcionalidad, etc.) en el código provee un mecanismo de desarrollo y encapsulamiento mayor en comparación con los controles de usuario, ya que los controles personalizados se compilan en un ensamblado.

Los controles personalizados son código compilado. Cuando hemos compilado un control, podemos añadirlo al cuadro de herramientas del IDE y así aprovechar el entorno visual para su posterior uso. Debido a su naturaleza compilada, los controles personalizados se pueden instalar en una sola copia en la Cache global de ensamblados, dejando que estén disponibles para todas las aplicaciones.

Los controles de servidor personalizados ofrecen capacidades superiores en tiempo de diseño, un uso más sencillo gracias a poder integrarse en la barra de herramienta del IDE y un mayor control sobre la funcionalidad. Naturalmente, todos los beneficios de los controles personalizados no vienen gratis. En general, los controles personalizados requieren un ciclo de desarrollo más largo y un nivel de conocimiento más alto del personal de desarrollo.

Los controles de servidor personalizados tienden a implementar una funcionalidad más rica y muestran una mayor reutilización de proyecto en proyecto.

Web Parts

Los tipos anteriores pueden usarse como Web Parts limitados, ya que en realidad son una extensión de los controles personalizados. La idea principal es que los desarrolladores realicen controles para que los usuarios personalicen y modifiquen las aplicaciones de acuerdo a sus necesidades y que estos cambios sean visibles para todos los demás usuarios o a los que correspondan.

De esta manera los usuarios finales pueden cambiar la visualización y posicionamiento de los Web Parts, exportar o importar controles, crear conexiones entre ellos e incluso administrar (en caso de que el usuario posea los permisos necesarios) el acceso al sitio por parte del resto de los usuarios.

2.3. Controles en Java

En general no existe mucho soporte nativo en el lenguaje Java para esta clase de proyectos. Esta técnica está más ligada a tecnologías .NET, y aunque es posible realizarlo en Java, no existen muchas herramientas que le simplifiquen el desarrollo al programador. A continuación presentamos algunas aproximaciones que permiten el desarrollo de aplicaciones RIA.

- TAG LIB: Se crean a muy bajo nivel. Se puede crear una librería que extiende los controles HTML estándar pero el desarrollador debe administrarlas manualmente. Hay que realizar todo el camino correspondiente desde el cliente, armar el servlet y crear la vista que sea necesaria retornar.
- GWT: Librería gráfica clásica del lenguaje, programada en Java pero compilada a JavaScript. Permite ocultar varios aspectos de la tecnología Ajax, es bastante sencillo y compatible con varios navegadores. Ayuda a crear y probar aplicaciones más fácilmente por parte de los desarrolladores.
- Wickets: Controles similares a los utilizados en .NET, con algunas funcionalidades ya programadas. Es un framework para desarrollo de aplicaciones web en Java que huye del uso de XML y se basa únicamente en Java y HTML.
Básicamente es orientado a componentes y el manejo de eventos dentro de una aplicación web es muy transparente. El único problema es que hacer algo que salga un poco de los componentes que te provee wicket requiere un aprendizaje más profundo. Es una opción muy válida para desarrollar componentes webs como si los estuviéramos creando para una aplicación de escritorio, es decir, como si usáramos AWT o SWING.
- Jspresso: Framework de licencia libre que permite el desarrollo de aplicaciones RIA. Contempla todo el proceso de desarrollo y libera al programador de muchas cuestiones técnicas de la interfaz. Básicamente tiene la capacidad de tomar un modelo y realizar las acciones necesarias para lograr un sitio funcional.
- Pivot: Framework publicado por Apache. Las aplicaciones se desarrollan utilizando Java y Xml y se pueden ejecutar como applets embebidos en páginas Web o como aplicaciones por sí mismas. Se encuentra basado en Java2D, pero lo extiende con conceptos como binding de datos, efectos y transiciones e integración con servicios Web.

Capítulo 3. Patrones RIA

Los patrones de diseño ofrecen soluciones genéricas a problemas comunes. Estas soluciones son documentadas, de manera de que cualquier desarrollador pueda saber cuándo y cómo utilizar cada uno. Por tanto, esta documentación debe ser clara y completa, intentando abolir detalles de implementación, a menos que estos fueran relevantes a la solución. Son en teoría independientes de la tecnología y poseen un alto grado de abstracción.

Hay varias definiciones acerca de qué es y qué comprende un patrón (ref. [11] y [35]). Según Christopher Alexander, quién definió por primera vez el término patrones de diseño, “Cada patrón describe un problema que ocurre una y otra vez en nuestro entorno, para lo cual describe la base de una solución de manera que se pueda utilizar un millón de veces más, sin hacerlo de la misma manera dos veces”.

Los denominados patrones no son una verdad absoluta, lejos de eso, son soluciones relativas al contexto dónde se utilizan, que tal vez en determinadas circunstancias no logren resolver el problema de la manera más adecuada. Incluso pueden utilizarse como modelo para resolver un problema, sin seguir estrictamente las ayudas que éste nos proporciona, aportando de esta manera nuestra propia creatividad y el modo de utilizar el mismo. Se diferencian así de las reglas o guidelines, que suelen ser verdades estrictas independientemente de su contexto, mucho más específicas que los principios redactados en los patrones.

La idea subyacente a esto es que una solución que ya fue utilizada y probada en contextos similares tiene seguramente más probabilidades de tener éxito que una solución novedosa. Se advierte fácilmente que asegura que sea mejor sino que probablemente lo sea, ya que es una solución ampliamente utilizada. Nada nos impide como desarrolladores encontrar una solución única que se adapte perfectamente a nuestro problema y tenga el éxito esperado.

Los patrones son particularmente útiles en organizaciones grandes, ya que proveen una solución rápida, y al estar ya documentados como tal, son conocidos por las personas que trabajan en cada proyecto, otorgando de esta manera la consistencia necesaria en cualquier producto de calidad profesional.

Pero no solo son conocidos por diseñadores o desarrolladores. Otro efecto inmediato de utilizar patrones de diseño bien conocidos, es que los usuarios suelen estar acostumbrados a utilizar aplicaciones con determinadas formas, por lo que diseñar utilizando patrones permite que no sea necesario una explicación detallada al usuario para que éste pueda desenvolverse mientras utiliza el producto.

No todos los patrones que describiremos a continuación son patrones generales de diseño, sino que algunos están más ligados a una tecnología, en este caso aplican mejor con aplicaciones Web, pero muchos de ellos pueden ser utilizados en aplicaciones de otro tipo, como aplicaciones de escritorio o móviles.

En particular los patrones RIA, que serán los patrones sobre los que nos centraremos en este documento, describen la interacción, operación y presentación de widgets (bloques de construcción básicos de la interfaz gráfica de usuario) de RIA.

El desencadenante de un patrón RIA es la interacción, como un click o el paso del mouse sobre una región determinada de la pantalla, como resultado de esta acción una gran variedad de operaciones pueden ser realizadas, para finalmente actualizar la interfaz de usuario.

Recomendamos fehacientemente la lectura de los patrones descritos por Yahoo! (ver ref. [37]) y los aportes de Bill Scott (ver ref. [24] y [25]) y Wroblewski (ref. [36]), sobre los cuales basamos gran parte de este trabajo.

Para favorecer un mejor entendimiento de la función de los patrones RIA, los hemos dividido en varias categorías basadas en ref. [24]:

- Editado “In-Page”
- Drag and drop
- Selección directa
- Superposición (overlay)
- Inlay
- Transicionales
- Búsqueda
- Feedback

Cada categoría contiene una colección de patrones que resuelven de una u otra manera un problema dado, cada uno con diferentes ventajas y contras, que deberán tenerse en cuenta para la selección del patrón adecuado en cada momento.

3.1. Editado “In-Page”

Esta categoría agrupa patrones que permiten que el usuario edite contenido de una página dada sin necesidad de redirigirse a otra que cumpla tal función. Muchas veces es necesario que el usuario tenga acceso al resto de la información de la página mientras se encuentra editando un campo, o es útil ver el resto de los elementos cuando se trata de una gran cantidad de ítems.

3.1.1. Single Field In-Line Edit

Es el patrón más simple de esta categoría. Consiste en permitir al usuario editar un campo de texto, sin necesidad de redirigirse a otra página web para hacerlo.

Ventajas y desventajas

- No es necesario ir a una nueva página para editar
- Visualización y edición se encuentran en el mismo contexto
- La accesibilidad puede verse afectada

Experiencia de usuario

El usuario se siente más cómodo, ya que no necesita salir del contexto actual para editar la información.

Contexto de uso

- Cuando se intenta evitar que el usuario navegue a una nueva página para realizar la edición
- Cuando las ediciones se realizan mejor en el contexto de la página actual

- Cuando la facilidad de edición de la información es importante en relación a la facilidad de lectura, de otra manera se debería ocultar esta funcionalidad

Buenas prácticas

- Mostrar un tooltip invitando a editar el campo
- Resaltar el fondo del área editable
- Cambiar el cursor del mouse mientras se navega por encima del área editable
- Proveer una alternativa con mejor accesibilidad que permita editar en una página diferente cuando se trata de usuarios discapacitados
- Para volver editable la información solo basta con hacer click con el mouse sobre el área editable.

Posibles errores de uso

- Puede no quedar clara al usuario la funcionalidad que está habilitado a realizar
- La página puede quedar muy cargada si se permiten demasiadas ediciones
- El botón para editar se encuentra lejos de la información editable

Ejemplos

No es ampliamente utilizado en la Web este tipo de patrón, pero podemos verlo en acción en el sitio Stack Exchange. Este sitio es un foro donde se pueden hacer preguntas acerca de diferentes aspectos de la informática.

En el momento en que se crea un comentario, el usuario que lo escribió está habilitado para editarlo mediante un link al final del mismo como se puede apreciar en la imagen.

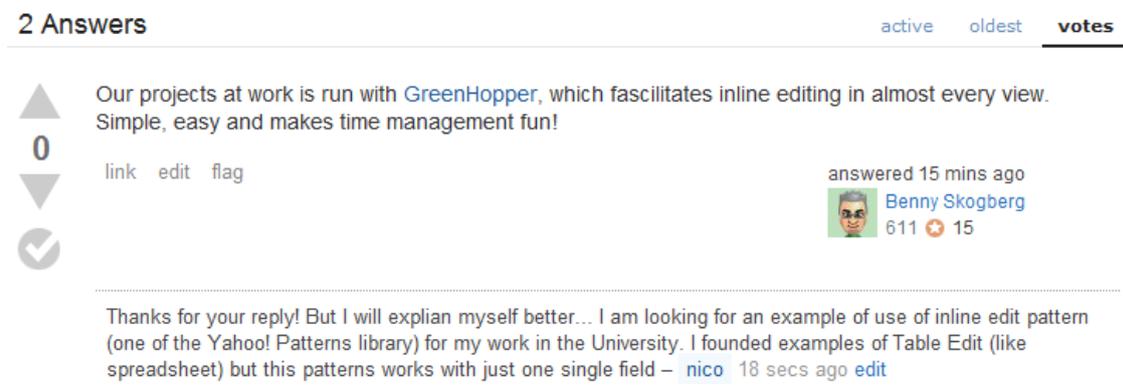


Imagen 3.1. Edición inline de comentarios (<http://ux.stackexchange.com>)

Luego de presionar el link, en lugar de redireccionar al usuario a una nueva página, el comentario que hasta el momento era de solo lectura se convierte en un campo de entrada, y se agregan botones para guardar o cancelar los cambios realizados. Una vez hecho esto el aspecto del comentario vuelve a ser tal cual era originalmente.

▲ 0 ▼ ✓

Our projects at work is run with [GreenHopper](#), which fascilitates inline editing in almost every view. Simple, easy and makes time management fun!

link edit flag

answered 15 mins ago

 [Benny Skogberg](#)
611 ⭐ 15

Thanks for your reply! But I will explian myself better... I am looking for an example of use of inline edit pattern (one of the Yahoo! Patterns library) for my work in the University. I founded examples of Table Edit (like

313 characters left

Save Edits
cancel
hide help

Imagen 3.2. Edición inline de comentarios activa (<http://ux.stackexchange.com>)

3.1.2. Multi Field In-Line Edit

Este patrón es similar al anterior, pero se extiende para proveer una solución a problemas donde hay varios campos a editar, o bien este es más complejo que un simple cuadro de texto.

Ventajas y desventajas

- No es necesario ir a una nueva página para editar
- Visualización y edición se encuentran en el mismo contexto
- El cambio entre modos de visualización y edición puede ser muy abrupto

Experiencia de usuario

El usuario puede editar una página compleja sin necesidad de redirigirse a una nueva, eliminando la necesidad de recargas completas.

Contexto de uso

- Cuando se intenta evitar que el usuario navegue a una nueva página para realizar la edición
- Cuando las ediciones se realizan mejor en el contexto de la página actual
- Hay varios campos a editar, o éstos son complejos

Buenas prácticas

- Utilizar animaciones para minimizar la diferencia entre modos al pasar de uno a otro
- Utilizar el mismo tamaño para ambos modos
- Proveer un feedback al usuario al guardar

Posibles errores de uso

- Utilizar diferentes estilos para la visualización y edición puede derivar en que el usuario no vea claramente el efecto de sus cambios

Ejemplos

En el sitio Web de Yahoo! Pulse, donde se puede guardar información personal así como otros datos relacionados al usuario se puede observar claramente el uso de este patrón. En este caso al visualizar la información personal del usuario se provee un botón Modificar para volver

editables todos los campos de la página, sin necesidad de redirigirse a una nueva para tal efecto.

Información básica		Modificar
Nombre completo	Nicolas Gordillo	Sólo visible para amigos
Nombre en pantalla	Nicolas	Visible a todos
Sexo	Hombre	Visible a todos
Fecha de nacimiento	01 marzo 1984	Sólo visible para amigos
Lugar	La Plata, Buenos Aires	Visible a todos

Imagen 3.3. Edición inline de varios campos (<http://pulse.yahoo.com>)

Podemos observar en la imagen siguiente como el look and feel de la página se mantiene, inclusive los tamaños y disposición de cada campo, al tiempo que se agregan botones para realizar las acciones de guardar y cancelar.

Al presionar el botón Guardar se muestra un feedback al usuario tal como se recomienda en “Best Practices” para dejar clara la acción llevada a cabo. Luego, se pueden visualizar los cambios realizados con el estilo que originalmente poseía.

Información básica * Información requerida		Guardar	Cancelar
Nombre completo *	<input type="text" value="Nicolas"/> <input type="text" value="Gordillo"/>	Amigos	▼
Nombre en pantalla *	<input type="text" value="Nicolas"/>	Tu nombre de pantalla sólo puede contener caracteres alfanuméricos.	
Sexo *	<input checked="" type="radio"/> Hombre <input type="radio"/> Mujer <input type="radio"/> Prefiero no decirlo		
Fecha de nacimiento *	marzo ▼ 1 ▼ 1984 ▼ <input type="checkbox"/> Ocultar mi edad	Amigos	▼
Lugar	<input type="text" value="La Plata, Buenos Aires"/>		

Imagen 3.4. Edición inline de varios campos activa (<http://pulse.yahoo.com>)

3.1.3. Overlay Edit

Este patrón se utiliza para mostrar el contenido que está siendo editado, ya no dentro del mismo contexto que el resto de los campos, sino solapando una ventana por encima de los campos no activos.

Ventajas y desventajas

- Permite que campos complejos o demasiado grandes sean mostrados para su edición sin distraer al usuario con la transición entre modos
- Resalta la importancia del campo que se encuentra siendo editado
- No es tan amigable al usuario como la edición en línea

Experiencia de usuario

Reemplaza a las ventanas del navegador en modo modal con popups más livianos, dando una experiencia más agradable al usuario.

Contexto de uso

- Cuando no se puede insertar un campo de edición complejo o demasiado grande dentro de la misma página
- Cuando lo que se desea editar es importante en sí mismo, y se quiere por tanto interrumpir el normal funcionamiento de la página (por ejemplo para evitar que un campo sea modificado sin intención)
- El contexto de la edición no es importante
- Cuando el campo no es frecuentemente editado

Buenas prácticas

- Utilizar cambios en el puntero del mouse para invitar al usuario a editar el campo
- Evitar el uso del doble click para editar
- Agregar un [editar] entre corchetes cerca del ítem a modificar para separar el link del resto del texto sin distraer al usuario
- Utilizar un estilo para la ventana superpuesta que minimice la diferencia entre ambos modos
- Usar botones para guardar o cancelar los cambios cuando sea posible
- Si es posible hacer la ventana movable para revelar información útil que haya sido ocultada
- La ventana debe aparecer cerca del link o botón que lo activó

Posibles errores de uso

- Utilizar varios overlays anidados para campos extra, se debe reacomodar la ventana para utilizar un solo overlay a la vez
- Guardar la información de forma implícita (sin utilizar botones) puede resultar confuso
- No se debe utilizar este método para reemplazar un simple editado en línea

Ejemplos

Esto se puede encontrar en el sitio de Flickr, donde al seleccionar una o varias fotos se permite modificar ciertos datos inherentes a las mismas. En el ejemplo que mostramos se muestra como mediante un cuadro de diálogo superpuesto al resto de la página (que toma un estilo más opaco) se puede editar la fecha de la foto.

Como apuntamos en las buenas prácticas de uso, se cuenta con botones para guardar o cancelar estos cambios, sin embargo no es movable ni aparece cerca del ítem que activó esta edición. Tal vez el primero de los fallos no tenga real importancia, ya que este cambio afecta a todos los elementos (fotos en este caso) por igual con lo que queda justificada su aparición centrada en pantalla.



Imagen 3.5. Edición mediante cuadro superpuesto (www.flickr.com)

3.1.4. Table Edit

Este enfoque intenta proveer al usuario funcionalidad de edición para información mostrada en forma de tablas. En este caso la presentación de la tabla es lo más importante, por lo que la invitación a su edición sólo debe mostrarse cuando está claro que es intención del usuario hacerlo.

Ventajas y desventajas

- Permite editar contenido de tablas dentro de la misma página
- Visualización y edición se encuentran en el mismo contexto
- Permite emular el comportamiento de aplicaciones de escritorio que les son familiares a la mayoría de los usuarios

Experiencia de usuario

Permite modificar contenidos de tablas en línea sin distracciones provocadas por transiciones entre modos de una manera a la que el usuario se encuentra por lo general acostumbrado debido a convenciones impuestas por aplicaciones de escritorio que han dado esta funcionalidad.

Contexto de uso

- Cuando se muestra información editable en forma tabulada

Buenas prácticas

- Evitar activar la edición con el paso del mouse por encima de la celda
- Utilizar un click para habilitar la edición. Sin embargo, en este caso el doble click para editar es tolerable ya que el funcionamiento de la aplicación web es muy similar al de aplicaciones de escritorio
- Para reflejar el modo de edición se puede ampliar la altura de la celda
- Utilizar las convenciones con las que están familiarizados los usuarios

Posibles errores de uso

- Utilizar un solo click para habilitar la edición puede ser molesto para el usuario

- Si se habilita la edición con el paso del mouse por encima de la celda, cambiar la altura de la celda puede interferir con la visualización de las celdas siguientes

Ejemplos

Un ejemplo de este tipo de patrón se haya realizado en el sitio de Accenture, donde cada empleado debe completar la cantidad de horas que ha trabajado en cada proyecto durante un período de tiempo. De esta forma se intenta simular el comportamiento que vemos en una planilla Excel en aplicaciones de escritorio, dando la posibilidad de cargar datos dentro de la misma pantalla.

The screenshot shows the 'myTimeandExpenses' web application interface. At the top, there is a navigation bar with the Accenture logo and the text 'High performance. Delivered.'. Below this, there is a user profile section for 'Mariano Oscar Covatti' and a date selector for '30/04/2011'. The status is 'Processed'. There are buttons for 'Record', 'Review', 'Represent', and 'Authorize'. Below the navigation, there are tabs for 'Time', 'Expenses', 'Adjustments', 'Summary', and 'Assignments'. The main content area is a table with columns for days of the week (sáb, dom, lun, mar, mié, jue, vie, sáb, dom, lun, mar, mié, jue, vie, sáb) and a 'Total' column. The table contains data for 'Req WBS Unassigned Time (NG403020)' and 'Holiday (970X00)'. The 'Req WBS Unassigned Time' row shows 8.0 hours for Monday, 8.0 for Tuesday, and 8.0 for each of the other days, with a total of 64.0. The 'Holiday' row shows 8.0 hours for Friday and Saturday, with a total of 16.0. The 'Total Hours' row shows 8.0 hours for each day, with a total of 80.0. The 'Work Schedule' row shows 8.0 hours for each day, with a total of 80.0. The 'Daily Overtime' row is empty. The 'Saturday Overtime, hours UNTIL 1 PM' row is empty. The 'Saturday Overtime, hours AFTER 1 PM' row is empty. At the bottom of the table, there are buttons for 'Delete', 'Time Entry Template', and 'Calculate / Save Hours'. A copyright notice is visible at the bottom: '© 2008 Accenture All Rights Reserved. Accenture Confidential. For Internal Use Only.'

Assignments	sáb 16	dom 17	lun 18	mar 19	mié 20	jue 21	vie 22	sáb 23	dom 24	lun 25	mar 26	mié 27	jue 28	vie 29	sáb 30	Total
Work Location																
Req WBS Unassigned Time (NG403020)			8,0	8,0	8,0					8,0	8,0	8,0	8,0	8,0		64,0
Holiday (970X00)						8,0	8,0									16,0
Total Hours			8,0	8,0	8,0	8,0	8,0			8,0	8,0	8,0	8,0	8,0		80,0
Work Schedule			8,0	8,0	8,0	8,0	8,0			8,0	8,0	8,0	8,0	8,0		80,0
Daily Overtime																
Saturday Overtime, hours UNTIL 1 PM																
Saturday Overtime, hours AFTER 1 PM																

Imagen 3.6. Table Edit en sitio de Accenture (myte.accenture.com)

3.1.5. Group Edit

Este patrón permite mostrar ambos modos de lectura y edición prácticamente sin variaciones.

Ventajas y desventajas

- Es prácticamente transparente la transición entre ambos modos
- Facilidad de lectura de la información
- Es difícil descubrir la funcionalidad

Experiencia de usuario

De esta manera se ofrece al usuario la forma más transparente de ofrecer edición dentro de la misma página donde se visualiza la información.

En el caso de aplicaciones Web tales como portales es muy común el uso de módulos. En lugar de utilizar páginas separadas para la configuración de cada módulo se suele utilizar el mismo contexto para editar los módulos que se encuentran juntos.

Contexto de uso

- Cuando se desea que la funcionalidad de edición afecte mínimamente a la visualización de la pantalla
- Cuando hay muchos ítems editables en la misma pantalla

Buenas prácticas

- El método para activar y desactivar el modo de edición debería ser similar
- La transición entre ambos modos debería ser instantánea
- Evitar utilizar animaciones para el cambio de modo
- Utilizar solo un botón editar para editar todo el contenido de la página que sea modificable

Posibles errores de uso

- Puede ser difícil descubrir la funcionalidad ofrecida

Ejemplos

Este patrón se puede observar en Google Docs. Allí se observa una lista con todos los archivos almacenados en el servidor, luego de seleccionar uno de ellos a su derecha aparece una descripción además de otros datos adicionales. Como se puede ver en la imagen se provee de un link Edit mediante el cual queda claro la funcionalidad ofrecida.

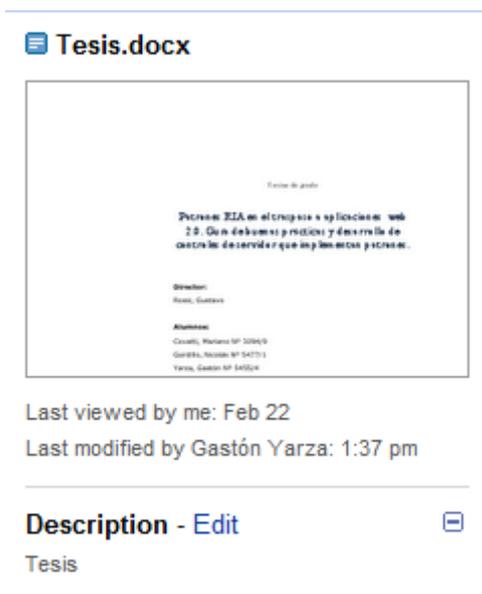


Imagen 3.7. Group Edit en Google Docs (*docs.google.com*)

Al hacer click sobre el link no se redirecciona a otra página ni se abre ninguna ventana, sino que por el contrario el texto sigue manteniendo prácticamente el mismo estilo sin variaciones importantes. De esta forma el usuario queda habilitado para realizar cambios sobre la descripción, y al finalizar presionando el mouse en cualquier otro lugar de la pantalla se vuelve inmediatamente a la presentación original.

Tesis.docx



Last viewed by me: Feb 22

Last modified by Gastón Yarza: 1:37 pm

Description - Edit

Tesis

Imagen 3.8. Group Edit en Google Docs, modo edición (docs.google.com)

Otro ejemplo se puede ver también en un producto de Google como lo es Gmail. Aquí se pueden crear etiquetas para organizar los mails como si se tratara de carpetas en el disco. Al presionar en el link correspondiente a la administración de etiquetas podremos ver un listado con algunas acciones posibles sobre cada una. En la imagen siguiente se puede ver una zona de la pantalla en cuestión.



Imagen 3.9. Group Edit en Gmail, modo lectura (www.gmail.com)

Al pasar el mouse sobre cada etiqueta se muestra un recuadro que invita al usuario a hacer click sobre la misma. Finalmente, si el usuario aprieta el mouse en una de ellas, el recuadro que se mostraba pasa a ser de un color más llamativo y el texto descriptivo se selecciona para su edición, indicando de esta manera que ya se puede modificar la información.

Al presionar en otro punto de la pantalla, los cambios quedan salvados. Se respeta en este caso la buena práctica de habilitar y deshabilitar la edición de forma similar.

3.1.6. Rating an object

Este patrón permite ingresar una valoración personal a un ítem determinado de forma directa y simple. De esta manera, toda la comunidad participa de la evaluación de los ítems existentes en el sitio orientando de esta manera a los demás usuarios acerca de la calidad de cada uno.

Este patrón permite a los usuarios evaluar un objeto respecto de una escala, por ejemplo, de una a cinco estrellas. De esta manera, se provee una herramienta para que la comunidad participe y evalúe contenido, dando la posibilidad a otros usuarios del sitio de ver la valoración del resto de ellos acerca de un objeto.

Además, el uso del patrón puede mejorar la calidad de los resultados de búsqueda, mediante la ordenación de los objetos por su valoración.

Ventajas y desventajas

- Provee una forma simple de valorar un objeto.
- El usuario no se desvía de su actividad al evaluar un objeto.
- Fomenta la participación.
- Puede mejorar notablemente la calidad de los resultados de búsqueda.
- No se dispone de un “control” sobre la valoración de los contenidos.

Experiencia de usuario

El usuario experimenta un sentimiento de participación en el sitio, involucrándose en el mismo de una manera fácil y directa. Tiene acceso directo al contenido más valorado por la comunidad.

Contexto de uso

- Cuando se intenta ingresar una valoración de un objeto de forma directa
- Cuando se intenta incrementar la participación de los usuarios en el sitio
- Cuando se intenta mostrar la valoración de un objeto o grupo de objetos.
- Poner al objeto a su valoración por la comunidad (sea pública o restringida).

Buenas prácticas

- Proveer la menor cantidad de opciones posibles
- Mostrar estrellas o iconos relacionados para cada una de las alternativas
- Usar imágenes o botones simples para votaciones con solo dos alternativas

Posibles errores de uso

- Confusión entre la valoración personal y la de la comunidad, antes y después de la votación
- Proveer demasiadas opciones

Ejemplos

En el sitio amazon.com se puede ver un ejemplo de utilización del patrón. Cada artículo publicado en el sitio puede ser evaluado por los usuarios registrados al mismo. Esta evaluación está entre los números 1 a 5, representados cada uno por una estrella.

Blood, Bones & Butter: The Inadvertent Education of a Reluctant Chef [Hardcover]

[Gabrielle Hamilton](#) (Author)

★★★★☆ (46 customer reviews) |  (63)

Imagen 3.10. Artículo con posibilidad de votar (www.amazon.com)

Cuando el usuario pasa sobre una estrella se abre un pop up (Ver patrón: *Pop up*) que muestra la cantidad de votos totales divididos en las diferentes opciones, y se permite votar mediante un click en la descripción de la opción correspondiente.



Imagen 3.11. Pop up que permite votar un artículo (www.amazon.com)

Este método puede no ser tan directo ya que se necesita esperar a la aparición del pop up para realizar la evaluación, otros sitios permiten la votación directamente pulsando la estrella que corresponde. Sin embargo, presenta muy claramente al usuario el momento y el efecto de su votación.

3.2. Drag and drop

Esta categoría de patrones incluyen aquellos donde el usuario obtiene como funcionalidad la capacidad de mover objetos de un lugar a otro solo con el movimiento del mouse.

Al utilizar un patrón de esta categoría, hay ciertos momentos que deben ser estudiados para decidir qué interacción es adecuada para que el usuario sepa en qué situación se encuentra a cada momento, dónde está habilitado a depositar el objeto y que lugares son inválidos para tal fin.

La división de patrones en esta categoría es algo autoritaria, bien podría tratarse de un único patrón, pero debido a ciertas diferencias en cuanto a buenas prácticas y posibles errores se han tratado aquí como patrones diferentes, cada uno tratando un tipo distinto de objeto draggeable.

3.2.1. Drag and drop de módulos

Este patrón se utiliza para mover módulos dentro de una página, mostrando en todo momento al usuario cuál sería el resultado de sus acciones.

Ventajas y desventajas

- Permite a los usuarios personalizar una página de forma simple
- Puede provocar jitter en la página
- Puede ser difícil descubrir la funcionalidad

Experiencia de usuario

El usuario obtiene un gran control sobre la página que está navegando, mediante la personalización del orden de los módulos puede modificar la página de acuerdo a sus gustos y necesidades.

Contexto de uso

- Cuando se quiere habilitar al usuario a mover objetos dentro de una página, visualizando el resultado de sus acciones
- Cuando se necesita personalizar el ordenamiento de los paneles en una página de acuerdo a las necesidades de cada usuario

Buenas prácticas

- Mostrar el módulo mientras se arrastra, creando huecos de manera de previsualizar el estado aproximado en el que quedarán posicionados
- Mostrar el módulo arrastrado con cierta transparencia para indicar que su posición aun no es definitiva
- Mantener la página lo más estable posible para evitar el jitter
- Usar el punto central del módulo para posicionar el mismo
- Si se trasladan objetos pequeños mantener la página estable
- La interfaz debe ayudar al usuario a detectar módulos que pueden ser trasladados, así como debe mostrar visualmente qué lugares son correctos para soltar el elemento

Posibles errores de uso

- Mostrar la página previsualizada puede ocasionar confusión en ciertas ocasiones donde el diseño es muy complejo
- Utilizar objetos opacos mientras se mueven por la página

Ejemplos

Un ejemplo de este patrón es el sitio de iGoogle, donde se permite personalizar la página para mostrar las diferentes secciones de acuerdo a los gustos del usuario. Aquí se permite arrastrar las diferentes secciones hacia el lugar de la pantalla donde se quiera mostrar.

Esta funcionalidad se presenta mediante un cambio de cursor al pasar por encima de la barra de título de cada sección, invitando al usuario a mover el módulo. Luego, con el botón del mouse apretado se presenta una imagen del módulo sobre el que se está trabajando más transparente para indicar que el campo aún no tiene lugar fijo.

Al dejar de presionar el mouse, si se dejó caer sobre una posición válida, el módulo queda anclado en esta nueva posición, de otra manera vuelve a su lugar original.

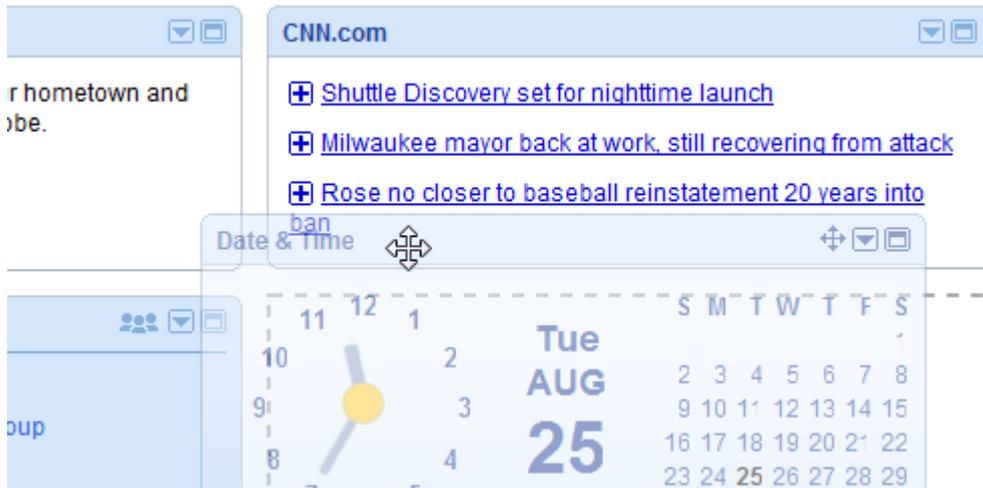


Imagen 3.12. Drag and drop de módulos en iGoogle (www.google.com/ig)

3.2.2. Drag and drop de listas

Este patrón es similar al anterior, con la restricción de que el movimiento se realiza en una sola dirección, puede ser de arriba/abajo o derecha/izquierda. De esta manera no es necesario marcar con un rectángulo el área donde se va a depositar el objeto sino que simplemente alcanza con dejar un espacio vacío en su lugar o una barra simple (lo cual hace el movimiento más suave).

Ventajas y desventajas

- Permite reordenar una lista de la manera más directa posible
- Permite previsualizar el orden final en el que quedará la lista
- En listas largas puede no ser el mejor método

Experiencia de usuario

De esta manera, el usuario puede modificar el ordenamiento de una lista de objetos de la forma más directa posible, de una manera totalmente natural.

Contexto de uso

- Cuando se permite que una lista se pueda reordenar por el usuario
- Cuando la lista no contiene gran cantidad de elementos

Buenas prácticas

- Si el drag and drop es una funcionalidad secundaria, invitar al usuario a que la siguiente vez que necesite realizar dicha acción lo haga de esta manera
- Si es posible, dejar un hueco para indicar la posición donde quedaría el ítem arrastrado
- Si se busca velocidad o los ítems son demasiado grandes utilizar la barra para indicar donde se ubicaría el ítem al soltarlo

Posibles errores de uso

- La funcionalidad puede quedar oculta al usuario
- Si la lista es demasiado larga se puede hacer dificultoso arrastrar los ítems, en estos casos se pueden ofrecer otras alternativas extra o accesos directos a diferentes partes de la lista

Ejemplos

Un buen ejemplo de este patrón lo obtenemos del cliente de mail de Google, Gmail, que entre sus funcionalidades permite abrir una pequeña ventana donde rápidamente se pueden cargar las tareas pendientes, así como reordenar las mismas.

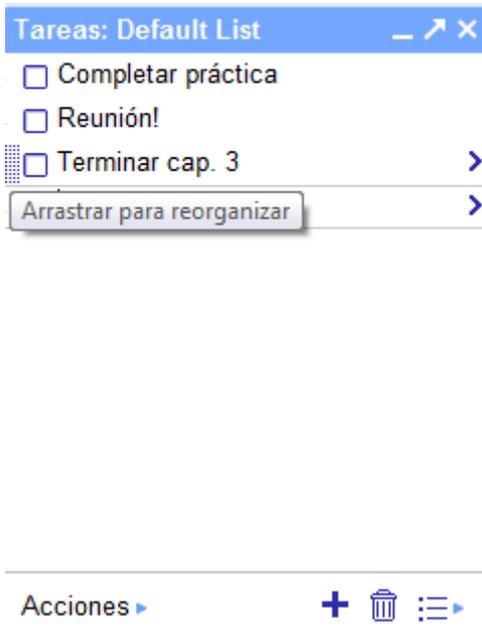


Imagen 3.13. Invitación para realizar Drag and drop de listas en Gmail (www.gmail.com)

Al colocar el puntero del mouse sobre la parte izquierda del ítem, se cambia el cursor para mostrar una mano y se muestra un tooltip invitando al usuario a arrastrar el ítem a su posición correcta.

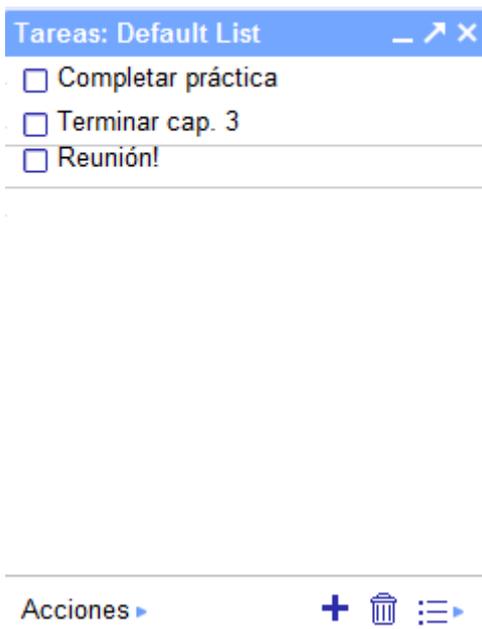


Imagen 3.14. Drag and drop de listas en Gmail (www.gmail.com)

Al mover el mouse se crea un espacio en la posición donde quedaría posicionada la tarea, así como también se quita el color de selección. Finalmente, al soltar el ítem la lista vuelve a su estado natural, con el ítem recién trasladado marcado en color.

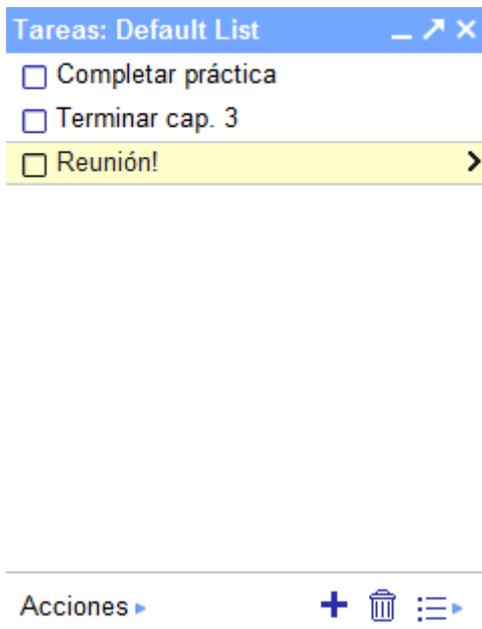


Imagen 3.15. Drag and drop de listas en Gmail, posicionamiento final (www.gmail.com)

3.2.3. Drag and drop de objetos

En este caso el drag and drop se utiliza para cambiar relaciones entre objetos, más aun cuando estas relaciones pueden ser representadas gráficamente.

Ventajas y desventajas

- Provee una manera muy simple de modificar relaciones entre objetos
- Puede ser difícil descubrir la nueva funcionalidad

Experiencia de usuario

El usuario puede administrar las relaciones entre diferentes objetos de una manera muy cómoda y natural, rápida y simple.

Contexto de uso

- Cuando se permite modificar relaciones entre objetos que pueden ser representadas gráficamente

Buenas prácticas

- Proveer feedback al usuario mientras se arrastra el objeto mostrando que lugares son o no válidos para soltar el mismo
- Proveer feedback al usuario luego de mover el objeto 3 pixeles o mantener el botón del mouse presionado por medio segundo
- Si la interfaz es demasiado compleja, intentar mantenerla estable mientras se arrastra el objeto
- Cuando se trata de relaciones padre/hijo resaltar al padre cuando se quiere agregar al objeto como descendiente para indicar que es un lugar correcto para soltarlo

- Utilizar la posición del puntero del mouse para determinar el sitio donde se suelta el objeto, sin utilizar desplazamientos adicionales
- Cambiar el aspecto del cursor para indicar que un objeto es movable

Posibles errores de uso

- Hacer que el usuario deba arrastrar demasiado el objeto para comenzar la animación del movimiento puede hacer parecer que el objeto no es movable
- Mostrar feedback del movimiento inmediatamente que el usuario selecciona y mueve el objeto puede resultar en acciones involuntarias o que la interfaz parezca demasiado sensible

Ejemplos

Un ejemplo donde se puede ver este tipo de comportamiento es en el mail de Yahoo!. Aquí el usuario puede arrastrar uno o varios mails para ubicarlos en la carpeta que corresponda. Para indicar la acción, al empezar a arrastrar el objeto (en este caso un mensaje) se presenta un ícono que representa un sobre con un número que indica la cantidad de ítems que están siendo procesados.

Para indicar los posibles lugares donde estos ítems pueden ser colocados, se muestran de otro color. En caso de soltar el objeto sobre un punto incorrecto simplemente no se realiza ninguna acción.

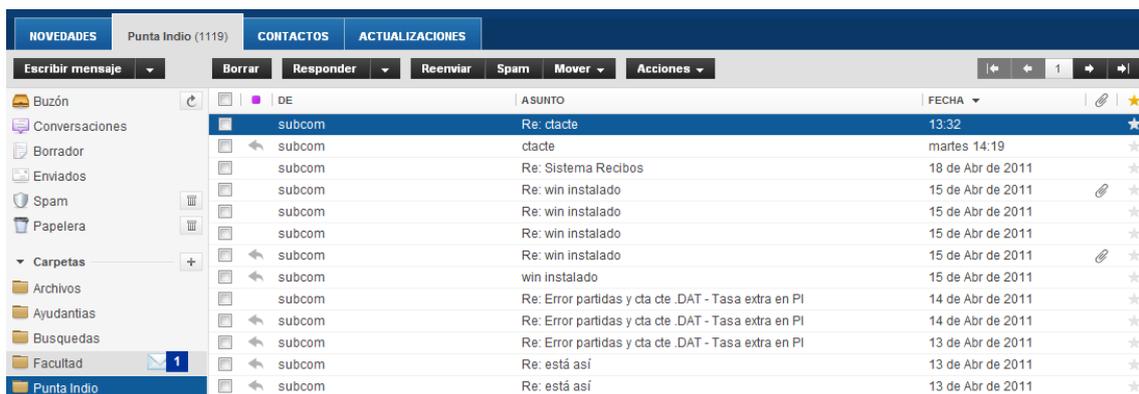


Imagen 3.16. Drag and drop de objetos en Yahoo! (ar.yahoo.com)

3.2.4. Drag and drop de acciones

En este caso, la acción de mover un objeto se utiliza para realizar una acción sobre el mismo. El ejemplo más utilizado de este patrón es mover un archivo a la papelera de reciclaje para realizar la acción de eliminación del mismo.

Ventajas y desventajas

- Se realizan acciones de manera simple y efectiva
- Puede forzarse su uso obteniendo situaciones complejas y poco claras
- La funcionalidad puede ser difícil de descubrir por parte del usuario

Experiencia de usuario

El usuario realiza acciones simplemente seleccionando objetos y moviéndolos hacia donde corresponda de manera natural, como eliminar un archivo trasladando éste a la papelera puede ser equivalente a lanzar un papel en un cesto de basura.

Contexto de uso

- Cuando las acciones se sobreentienden de forma natural mediante el movimiento correspondiente
- Cuando se desea proveer una alternativa para realizar acciones

Buenas prácticas

- Utilizarlo cuando sea natural realizar acciones de esta manera
- No utilizarlo demasiado, ya que no es fácil descubrir esta funcionalidad
- No usarlo para acciones simples, en estos casos evaluar alternativas más directas
- Proveer indicaciones claras que muestren la disponibilidad de esta funcionalidad

Posibles errores de uso

- Si se utiliza drag and drop de manera forzada se obtiene una solución poco clara y natural para el usuario, nunca debe influir en el armado del diseño de la página

Ejemplos

En Google Maps se puede observar que al buscar una ruta de un punto a otro y ubicar el cursor sobre la ruta se invita al usuario a utilizar drag and drop para cambiar el curso de la misma. Esta aplicación es completamente natural, equivalente a cómo haría una persona con un mapa impreso en papel.

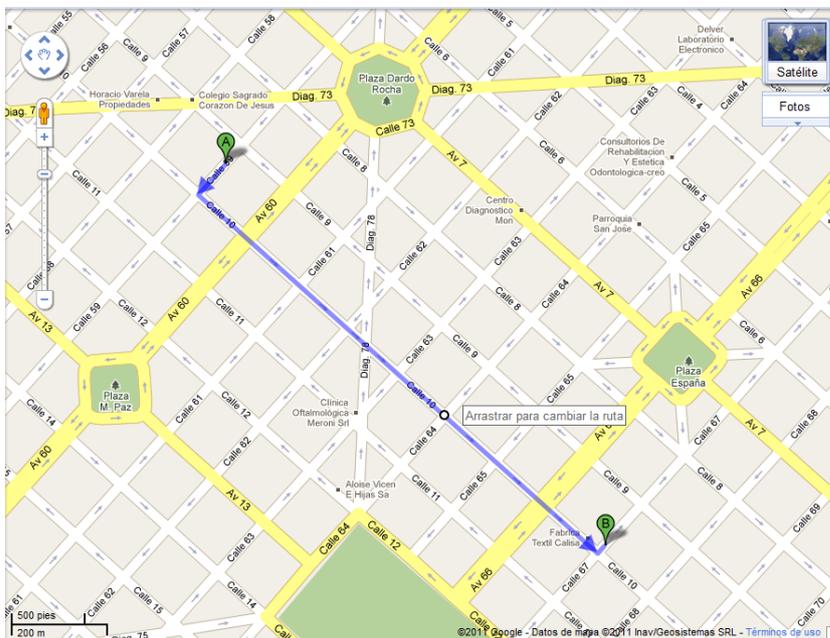


Imagen 3.17. Invitación al Drag and drop de acciones en Google Maps(maps.google.com)

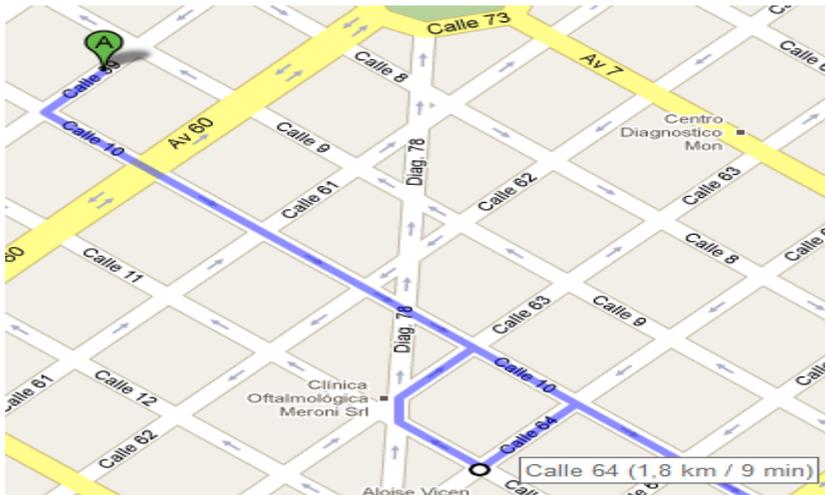


Imagen 3.18. Traslado de un punto en la ruta en Google Maps(maps.google.com)

Mientras el usuario arrastra el punto seleccionado, en tiempo real Google recalcula la ruta que debería tomar para llegar a destino, teniendo en cuenta que se debe pasar primero por el punto en cuestión. Adicionalmente, se muestra un detalle acerca del desvío que estamos realizando a la ruta original.

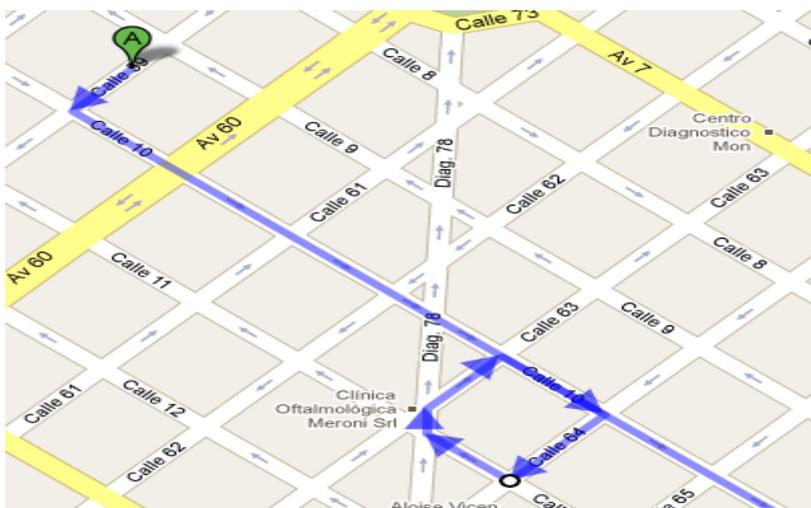


Imagen 3.19. Ruta final elegida (maps.google.com)

Finalmente, el mapa queda actualizado. Se debe notar aquí que la sección donde se forzó a un cambio de ruta queda marcada de diferente manera indicando de esta manera la modificación manual a la que fue sometida.

3.2.5. Drag and drop de colecciones

La última variación de drag and drop se utiliza para mover objetos y agregarlos a una colección, por ejemplo en un sitio de venta online se podrían arrastrar productos a una colección de ítems a comprar (habitualmente llamado carrito de compras).

Ventajas y desventajas

- Proporciona un método fácil y directo para agregar ítems a una colección
- Puede ser difícil descubrir la funcionalidad ofrecida

Experiencia de usuario

El movimiento de arrastrar un objeto, que podría ser por ejemplo un producto, a un contenedor de objetos (el carrito de compras) es la manera más fácil, directa y natural de realizar dicha acción.

Contexto de uso

- Cuando se quiere brindar una alternativa más directa para agregar objetos a colecciones

Buenas prácticas

- Usarlo como alternativa para agregar ítems a una colección
- Cuando se comienza a arrastrar un ítem resaltar el área donde debería dejarlo para agregarlo a la colección
- Intentar persuadir al usuario en cuanto sea posible para utilizar drag and drop

Posibles errores de uso

- Puede ser difícil de descubrir, por tanto no debe ser la única manera de realizar esta acción
- Si no queda claro cuál puede ser el contenedor sobre el cual se puede depositar el ítem puede llevar al usuario a la confusión

Ejemplos

Este patrón se puede observar en Flickr, donde un usuario puede subir todas sus fotos para mostrarlas a sus contactos. Luego, puede también organizarlas en álbumes agregando y quitando las fotos previamente cargadas. Este proceso puede ser realizado utilizando drag and drop. En la parte inferior de la pantalla se muestran todas las fotos cargadas, al hacer click con el mouse se cambia el cursor del mismo, y una pequeña imagen más clara aparece en el álbum invitando al usuario a soltar la foto seleccionada allí.

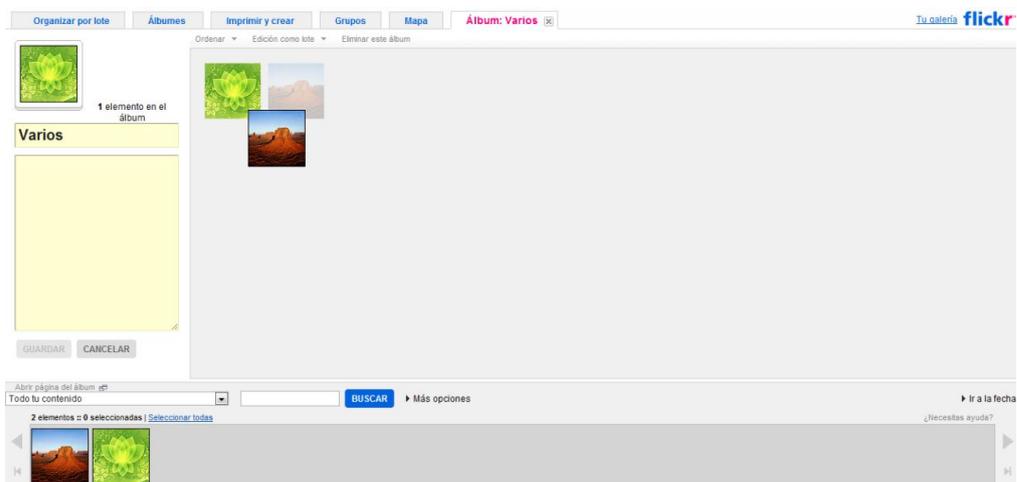


Imagen 3.20. Drag and drop para agregar a álbum de fotos en Flickr (www.flickr.com)

En la captura siguiente se puede observar la invitación a eliminar una foto, indicando claramente cuál es la zona donde es permitido soltar el objeto arrastrado. Luego de realizada la acción, una pequeña animación confirma el hecho.

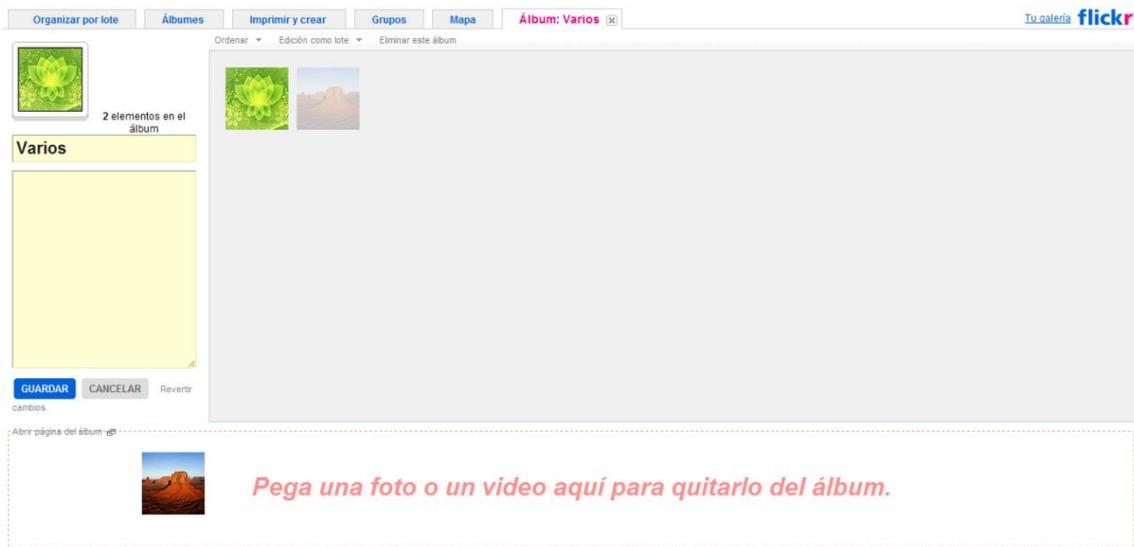


Imagen 3.21. Drag and drop para quitar fotos del álbum de fotos en Flickr (www.flickr.com)

3.3. Selección directa

El objetivo de estos patrones es brindar al usuario la posibilidad de tratar con los elementos que ve en pantalla de forma directa. El hecho de poder hacer click sobre un ícono o elemento de nuestro escritorio es una clara muestra de lo que significa interactuar directamente con un objeto.

3.3.1. Selección de ítems

Este patrón es ampliamente utilizado en la Web, mayormente para marcar elementos como seleccionados para realizar luego una determinada acción sobre los mismos. Para esto se suele utilizar casillas tipo checkbox o botones de dos estados, mientras que para llevar a cabo la acción se utiliza en general un botón.

Ventajas y desventajas

- Permite seleccionar directamente varios elementos para su tratamiento posterior
- Permite marcar/desmarcar un elemento dado

Experiencia de usuario

El usuario puede realizar selecciones múltiples discontinuas con suma facilidad.

Contexto de uso

- Cuando necesita seleccionar uno o más ítems para realizar una acción con posterioridad
- Cuando se desea poner una marca sobre uno o varios ítems

Buenas prácticas

- Utilizar checkboxes para selecciones discontinuas
- Utilizar cada checkbox en la fila que se desea seleccionar

- Resaltar el elemento elegido además de marcar el checkbox
- Hacer válida la selección solo en el contexto de la página actual
- Mostrar claramente la cantidad de elementos seleccionados
- Si es posible, deshabilitar las acciones mientras todavía no se seleccionó ningún ítem
- Utilizar este patrón para seleccionar un objeto sin abrirlo

Posibles errores de uso

- Mantener la selección de ítems a lo largo de varias páginas puede ser confuso
- Combinar con drag and drop
- Puede ser confuso saber si se debe presionar sobre una fila o sobre el checkbox

Ejemplos

Este patrón se puede encontrar en gran cantidad de sitios, especialmente en los servidores de correo. Aquí se adjuntan imágenes de Gmail, donde se pueden ver algunos elementos de la bandeja de entrada seleccionados y resaltados para su mejor visualización.

Luego de realizar la selección correspondiente podemos elegir que acción queremos realizar, ya sea eliminar los mensajes, moverlos o cualquiera de las posibilidades que el correo nos ofrece.



Imagen 3.22. Selección de mensajes en bandeja de entrada de Gmail (www.gmail.com)

En el caso de Gmail se mantiene un historial mientras cambiamos cada página, de manera que cuando volvemos a páginas anteriores podemos encontrar que los ítems que han sido marcados aun lo siguen estando. Sin embargo, para prevenir errores por acciones indeseadas sobre algunos elementos, el resultado de las acciones realizadas solo son aplicables a los ítems de la página actual.

3.3.2. Selección de objetos

Este tipo de selección se da cuando un elemento puede ser directamente seleccionado en pantalla. En el primer patrón de esta categoría, la selección se hacía indirectamente sobre un casillero o una bandera sobre el elemento a tratar, en este caso se selecciona el objeto en sí.

Ventajas y desventajas

- Es la manera más simple de interactuar con objetos
- No es fácil de implementar

Experiencia de usuario

El usuario experimenta una sensación de familiaridad ya que el comportamiento de este patrón es muy similar al funcionamiento de las aplicaciones de escritorio.

Contexto de uso

- Cuando se quiere permitir la selección múltiple de objetos
- Cuando se quieren seleccionar varias filas en una tabla scrolleable
- Cuando los elementos seleccionables puedan ser movidos con técnicas drag and drop
- Cuando se quiera simular el comportamiento de aplicaciones de escritorio

Buenas prácticas

- Acompañar su uso con las teclas Shift y Control para seleccionar varios ítems consecutivos y realizar selección discontinua respectivamente
- Si es posible, utilizar selección de ítems cuando la capacidad del navegador sea limitada
- Utilizar este patrón para seleccionar y abrir un objeto

Posibles errores de uso

- El mecanismo puede ser complejo para aprender a utilizar
- La tecla Control utilizada para objetos no continuos no funciona de la misma manera en Mac

Ejemplos

Esto se puede ver en acción en Flickr. Luego de subir las fotos personales, uno puede organizar las mismas dentro de álbumes para su mejor organización. Esto se realiza mediante selección de objetos, habilitando también las teclas Shift y Control, de manera que el usuario puede elegir una o más fotos continuas o no, y arrastrarlas al álbum correspondiente.

Notar la invitación al uso de este último patrón, y la relación intrínseca que existe entre la selección de objetos y Drag and drop.

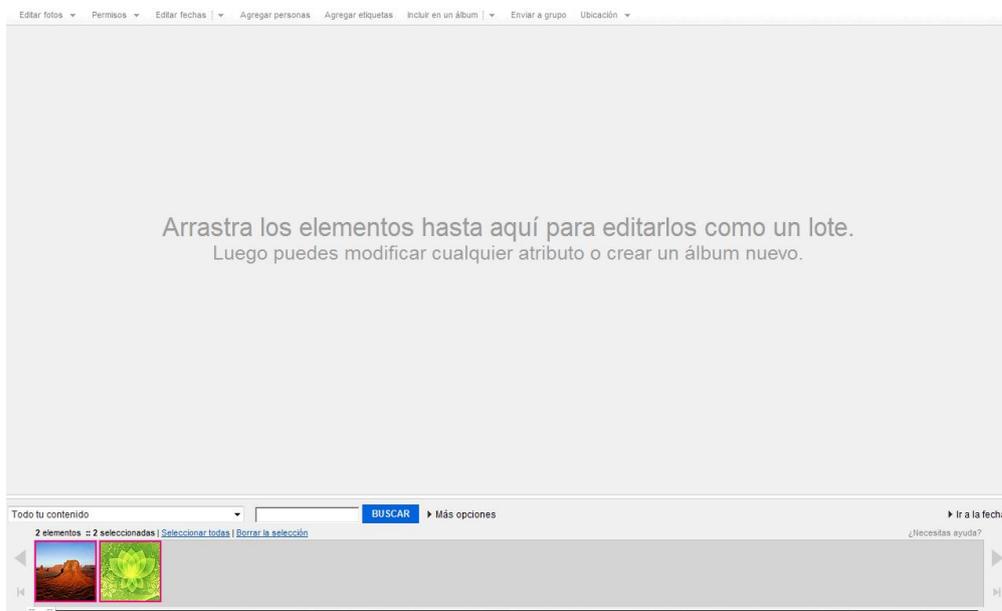


Imagen 3.23. Selección de fotos en Flickr (www.flickr.com)

3.4. Superposición (overlay)

Básicamente, se trata de pop ups livianos, esto quiere decir que son ventanas que emergen en la misma página que la actualmente visitada, sin necesidad de crear una nueva ventana del navegador. Esto usualmente es más rápido y genera una interfaz más limpia al usuario, ya que el desarrollador tiene completo control sobre el estilo y los controles de este pop up.

3.4.1. Tooltip

Permite mostrar más detalles acerca del objeto sobre el cual estamos arrastrando el mouse sin necesidad de mostrar una nueva página.

Ventajas y desventajas

- Muestra un mayor detalle sin perder el contexto actual
- Puede ser molesto para el usuario al ser utilizado en un contexto erróneo

Experiencia de usuario

El usuario puede obtener detalle de ciertos ítems de su interés sin perder el contexto en el cual se encuentra navegando. Esto le produce una sensación de satisfacción y control, ya que puede evaluar rápidamente con gran detalle antes de seleccionar un ítem en particular.

Contexto de uso

- Se requiere mostrar información adicional rápidamente
- Se requiere mostrar un detalle dentro del contexto de la página

Buenas prácticas

- Retardar la aparición del pop up en unas décimas de segundo para prevenir la aparición de pop ups no deseados
- Quitar el pop up inmediatamente cuando se mueve el mouse fuera del objeto
- La apertura y el cierre del pop up debe hacerse de manera similar
- Quitar animaciones innecesarias que retarden la aparición del pop up
- Mostrar el detalle mediante un click si se quiere hacer la acción más explícita
- Usar click para activar el tooltip si éste contiene otros links en su interior

Posibles errores de uso

- Mostrar el pop up sin retardo puede ser molesto al usuario
- Mostrar el pop up muy lentamente puede conllevar problemas de usabilidad
- El pop up debe mostrar más detalle que el ofrecido en el objeto original
- El pop up no debería cubrir otros ítems dentro de la misma página, para facilitar su rápida selección en caso de que el usuario lo requiera

Ejemplos

Un ejemplo de uso de tooltip es el sitio de películas www.cuevana.tv, donde se muestra un listado de películas con título y una imagen del poster de cada una. Al pasar por encima de esta imagen rápidamente se muestra un tooltip con una sinopsis y algunos datos extra.

Sin embargo, este ejemplo nos sirve para mostrar algunos errores del uso de este patrón. Primero, el pop up aparece sin esperar a que el usuario se detenga obteniendo de esta manera

una experiencia algo molesta. Por otro lado, el detalle de la película cubre el área donde se muestran los demás ítems del listado dificultando de esta manera nuestra búsqueda.



Imagen 3.24. Pop up con detalle de cada película (www.cuevana.tv)

Un buen ejemplo de este patrón puede encontrarse en Google Maps. Aquí a medida que el usuario se desplaza por el mapa puede detenerse en algún punto clave, indicado mediante el cambio de cursor del mouse. Un instante después aparece un pop up con una descripción de la ubicación junto a otros datos interesantes de la misma.



Imagen 3.25. Pop up con detalle del punto elegido en el mapa (maps.google.com.ar)

3.4.2. Diálogos

De la misma manera que los tooltip, los cuadros de diálogo son pop ups livianos donde el usuario puede confirmar o cancelar una determinada acción. De esta manera se logra que la apertura de cada cuadro sea más rápida y que la aplicación tenga completo control sobre la apariencia y posición de éste.

Ventajas y desventajas

- Permite abrir diálogos más rápidamente
- Permite completo control sobre la apariencia del diálogo
- Permite confirmar una acción sin perder el contexto de la misma
- Mantiene la apariencia en diferentes navegadores/sistemas operativos
- Sus ventajas puede tentar a un uso excesivo, que puede entorpecer el flujo de las acciones del usuario

Experiencia de usuario

La experiencia se ve afectada ya que los diálogos son visualmente más atractivos y más rápidos que una ventana modal del navegador. El usuario se ve beneficiado con un diseño más armónico, sin perder el contexto en el cual está trabajando.

Contexto de uso

- Cuando se necesitan realizar confirmaciones sin perder el contexto actual
- Cuando se necesita control sobre la apariencia del diálogo

Buenas prácticas

- Permitir cerrar el diálogo de manera similar a la acción de su apertura
- Para cuestiones que no se deban ignorar usar una ventana modal
- Evitar diálogos innecesarios que interrumpen el trabajo del usuario
- No usar overlays cuando se puede interactuar dentro de la misma página sin inconvenientes

Posibles errores de uso

- Su uso excesivo puede interrumpir demasiado la acción del usuario

Ejemplos

Un ejemplo de uso de este patrón se puede ver en el mail de Yahoo!. Al presionar sobre un ícono en la papelera de reciclaje se procede a eliminar todo su contenido, pero antes de realizar efectivamente dicha acción que podría resultar peligrosa para la información del usuario, se pide una confirmación mediante un cuadro de diálogo liviano, con la misma apariencia que el resto de la aplicación, donde el usuario puede aceptar la acción o cancelarla en caso de error.

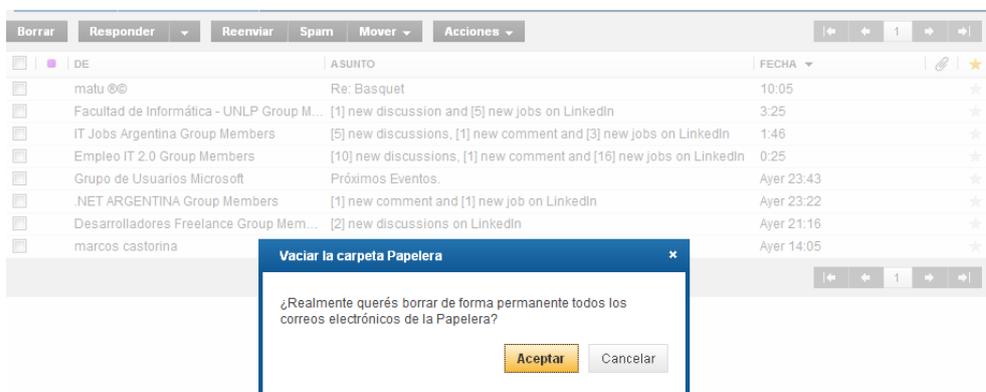


Imagen 3.26. Pop up de confirmación de eliminación de mensajes (ar.yahoo.com)

En general, cuando se trata de eliminar información (ya sean mails, archivos u otro tipo de objeto) se utiliza este patrón ya que permite resguardar al usuario de acciones involuntarias que pueden perjudicarlo gravemente.

3.4.3. Pop ups de entrada

Este patrón es el más complejo de los anteriores, se trata de permitir ingresar información dentro de los pop ups livianos.

Ventajas y desventajas

- Permite simplificar la interfaz
- Permite ingresar datos dentro del contexto actual
- Si no se usa adecuadamente, puede entorpecer el ingreso de otros datos

Experiencia de usuario

El usuario se encuentra más focalizado en los datos que debe completar en cada momento, abstrayéndose del resto del contexto para no generar distracciones innecesarias.

Contexto de uso

- Cuando se desea simplificar el contexto de la página
- Cuando se desea proporcionar cierta ayuda al usuario
- Cuando el contexto no es tan relevante
- Cuando la información editada es auxiliar

Buenas prácticas

- Utilizar bordes bien definidos para separarlo del resto de la página
- Utilizar transparencia para no ocultar totalmente los ítems por debajo
- Permitir quitar el cuadro fácilmente haciendo click en cualquier lugar
- Usar botones standard para guardar los cambios

Posibles errores de uso

- Entorpecer la vista de otros ítems debido a la superposición
- Ocultar el ítem que activó el pop up con la superposición del mismo
- Utilizar pop up cuando se podría realizar una edición dentro de la misma página

Ejemplos

Un sitio donde se usan extensivamente este tipo de cuadros modales es en Google Calendar. Aquí cada usuario registrado puede organizar su agenda gráficamente, añadiendo eventos y tareas en un calendario personal.

Al crear un nuevo evento, realizando un click sobre la hora en la que se desea cargar en la fecha deseada, se abre un cuadro modal donde aparece la fecha y hora automáticamente y se permite al usuario cargar una descripción del evento.

Esto permite agregar eventos rápidamente, sin necesidad de desorientar al usuario con detalles que tal vez no sean de su interés. En caso de que esta información le resulte insuficiente al usuario siempre puede presionar en “Editar detalles del evento”, el cuál abrirá una nueva página con todos los detalles a su disposición.

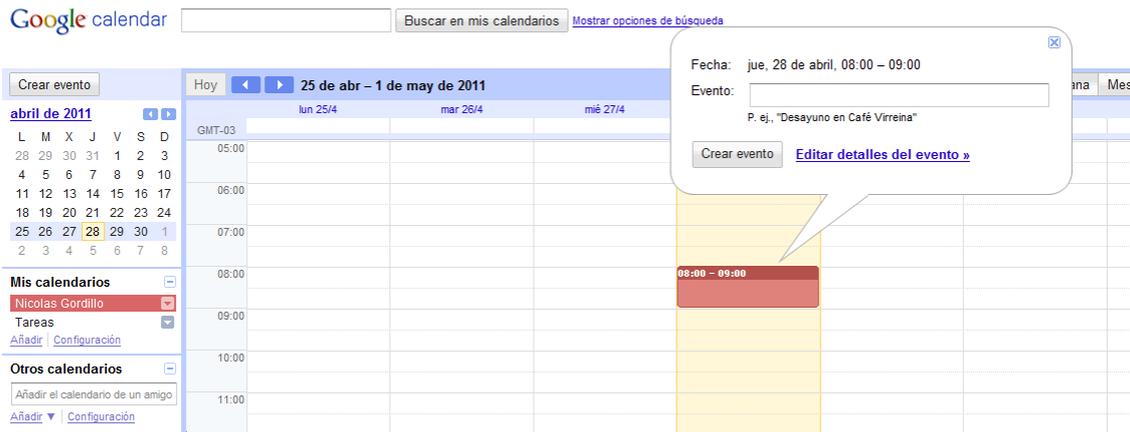


Imagen 3.27. Pop up de entrada de datos. Carga de un evento en calendario personal de Google. (www.google.com/calendar/)

3.4.4. Date/Time Picker

Este tipo de pop ups son similares a los de entrada de datos, pero se trata específicamente del ingreso de una o más fechas. Para esto se muestra un calendario de fácil navegación donde el usuario puede ingresar la fecha deseada, que se cargará luego en el campo de entrada a llenar.

Ventajas y desventajas

- Permite prevenir errores en formatos de fechas
- Permite ingresar fechas sin necesidad de escribirlas
- En ocasiones, podría ser más rápido para el usuario ingresar las fechas mediante el teclado

Experiencia de usuario

El usuario puede ingresar fechas en el sistema de manera fácil y cómoda, sin necesidad de utilizar el teclado y previniendo errores de formato de las mismas.

Contexto de uso

- Cuando se necesita realizar ingresos de fechas
- Cuando se intenta prevenir errores en el formato de las fechas

Buenas prácticas

- La navegabilidad entre meses/años debería ser directa
- Al seleccionar una fecha debería cerrar el pop up, sin necesidad de presionar sobre algún botón de acción
- Utilizar un botón de acción cerca del campo de entrada para activar el pop up
- Ubicar al usuario en la fecha actual automáticamente

Posibles errores de uso

- Utilizar Date/Time picker para fechas muy anteriores (como fechas de nacimiento) puede ser algo molesto
- No permitir navegabilidad directa entre años en casos donde la fecha podría ser bastante anterior a la fecha actual
- Si se deben ingresar muchas fechas, podría ser más lento el proceso

Ejemplos

Un ejemplo de este patrón se puede encontrar en despegar.com. En la búsqueda de vuelos, se permite elegir la fecha de partida y de regreso mediante un pop up con un calendario. Este pop up se abre con un click en un calendario pequeño que se encuentra a continuación del campo y se cierra de la misma forma haciendo click en el link "Cerrar" o en la fecha deseada.



The screenshot shows a flight search form with the following elements:

- Radio buttons for "Ida y vuelta" (selected), "Solamente Ida", and "Múltiples Destinos".
- Text input for "Origen: Buenos Aires, Argentina (BUE)".
- Text input for "Destino: Ingrese una ciudad de destino".
- Text input for "Partida: dd/mm/aaaa" with a small calendar icon.
- Calendar for "Regreso:" showing "Marzo 2011" and "Abril 2011".
- Grid for "Adultos:" with days of the week (Lu, Ma, Mi, Ju, Vi, Sá, Do) and dates.
- Checkboxes for "también" and "sólo Ej".
- Dropdown menu for "Opciones".
- Buttons for "Hotels" and "Cerrar".

Imagen 3.28. Date/Time Picker en acción (despegar.com)

3.5. Inlays

Esta categoría incluye patrones que se caracterizan por mantener al usuario dentro de la misma página para mostrar cierta información o realizar ciertas acciones. Esta categoría se contrapone directamente con la recientemente explicada acerca de patrones de superposición.

3.5.1. Accordion

Es una lista de ítems, cada uno descrito por un título, que tiene al menos dos niveles y cuyo nivel superior puede ser plegable, ahorrando de esta manera espacio con información que podría ser redundante. Al utilizarlo como menú este patrón permite que el usuario navegue por las secciones principales de una página, mientras que al mismo tiempo puede ver subítems de otras. De otra manera, se puede utilizar de forma similar al tab para mostrar nueva información sin redirigirse a una nueva página.

Ventajas y desventajas

- Permite que el usuario navegue por las secciones principales del sitio mientras observa las subsecciones de otra

Experiencia de usuario

Además del atractivo visual que este patrón provee, el usuario se ve beneficiado por la facilidad que adquiere al momento de la navegación del sitio.

Contexto de uso

- Cuando se necesita de un menú para navegar por las diferentes secciones/subsecciones del sitio pero no hay suficiente espacio para mostrar todos los enlaces
- Cuando se quiere mostrar nueva información sin necesidad de recargar la página

Buenas prácticas

- Cada ítem debería expandirse con un simple click del mouse

- Cuando se expande un ítem, el resto deberían contraerse
- La transición cuando se expande un ítem debería realizarse por medio de una animación gradual
- Mantener el panel más importante desplegado por defecto
- Resaltar el panel actualmente elegido para diferenciarlo claramente del resto
- Evitar el uso de Accordion anidados, si esto sucede pensar en otra estructura como podría ser un árbol

Posibles errores de uso

- Expandir un ítem utilizando hover puede ser molesto
- No debería usarse para mostrar información que se podría visualizar sin problemas dentro de la misma página
- Evitar que se contraigan los elementos no seleccionados puede causar que la página se estire demasiado y pierda las ventajas de utilizar el patrón

Ejemplos

Encontramos un ejemplo en el sitio Web de Technicolor, donde se presenta un menú con las diferentes secciones del sitio, cada una con un conjunto de sub secciones. Mostrar todas las sub secciones en pantalla sería engorroso para el usuario, y requeriría que realice scroll para poder ver todo el contenido.

En cambio de esta manera, el menú tiene un aspecto atractivo y presenta las secciones principales en una sola pantalla, sin scroll. Luego, si se presiona sobre un título se accede a las sub secciones de éste.

Cabe notar también que al expandir una pestaña se cierran automáticamente la que estuviera abierta hasta el momento, de esta manera se logra mantener el espacio en pantalla estable.

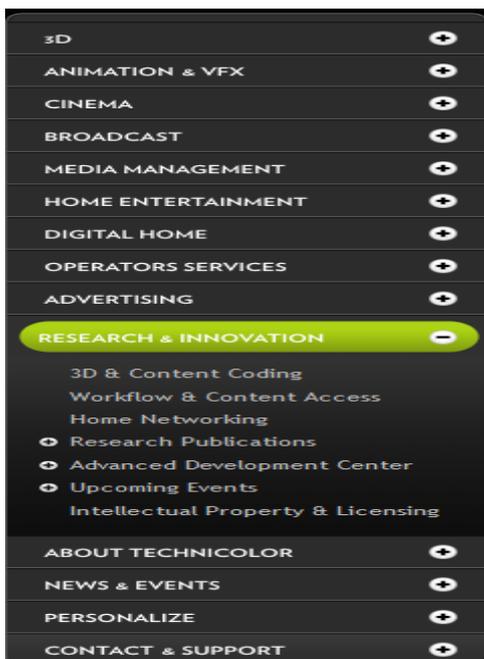


Imagen 3.29. Accordion en Technicolor (<http://www.technicolor.com/>)

3.5.2. Tabs

En lugar de navegar a través de diferentes páginas, los tabs pueden utilizarse para mostrar contenido dentro de la misma página. Está demostrado que los usuarios en general utilizan solo las primeras pestañas, no así el resto, que se visitan menos frecuentemente.

Ventajas y desventajas

- Permite visualizar distintos tipos de contenido dentro del mismo contexto
- Permite mostrar nueva información sin necesidad de recargar la página ni dirigirse a una nueva

Experiencia de usuario

El usuario puede visitar distinta información sin perder el contexto de la página actual, manteniendo de esta manera el marco de trabajo en el cual se encuentra.

Contexto de uso

- Cuando se necesita mostrar información sin necesidad de dirigirse a una nueva página
- Cuando la información a mostrar se encuentra dividida en categorías, tipos, etc.
- Cuando hay más de dos secciones para mostrar
- Cuando los nombres de las secciones son relativamente cortos

Buenas prácticas

- Usar tabs para mostrar contenidos adicionales
- Evitar el uso de varios tabs dentro de la misma página
- Si se utilizan varios tabs dentro de la misma página, mostrarlos con estilos diferentes
- Activar cada tab con un click del mouse
- Si es importante mostrar el contenido de las demás pestañas se puede utilizar el hover para su activación
- Colocar el contenido más importante en las primeras pestañas

Posibles errores de uso

- Utilizar el hover para activar cada tab puede ser molesto para el usuario
- Si el contenido de las pestañas varía en altura puede ser algo confuso

Ejemplos

Un ejemplo de este uso es el sitio de Yahoo!. Aquí se utilizan varios tabs para de esta forma poder presentar una gran cantidad de información al usuario, sin necesidad de agrandar físicamente la página ni desorientar al usuario con información superflua.

Con este objetivo se ofrece un tab donde se muestran noticias y hechos curiosos a los que Yahoo! les asigna mayor importancia, y por esto son colocados cerca del borde superior de la página.

HOY - 29 abril 2011



¿Cómo es nuestra conexión a Internet?

Según el Net Index, la velocidad promedio de descarga en la Argentina es de 3,16 megabits. [El ranking >>](#)

- Yahoo! Finanzas
- Dulces y "productivos" sueños



La "humillante" apuesta que hizo Gonzalito



Nuevo mapa de la teoría de la evolución



Rezaados en calidad de internet



Boda Real: El primer beso en público

1 - 4 de 28 < || >

Imagen 3.30. Primer Tab en Yahoo! (www.yahoo.com.ar)

Adicionalmente, se ofrece otra pestaña de noticias, pero esta vez dividida en categorías de manera que el usuario que se encuentre buscando noticias de un determinado tema pueda hacerlo fácilmente.

NOTICIAS **DEPORTES** INTERNACIONAL ↑



¡Los novios dieron el sí!
El príncipe Guillermo y Kate Middleton intercambiaron votos en la Abadía de Westminster. [Mirá el video >>](#)

 [La Familia Real llegando a Westminster Abbey](#)

- Policía británica arresta a 18 personas durante la boda...
- Príncipe Guillermo y Kate Middleton ya están casados
- Ejecutivo KEPCO ve revés para mercado nuclear tras...
- ¿Sufriste algún inconveniente para transitar por el...
- Llegan las primeras columnas para el acto de Moyano y...
- Héctor Recalde: "Nadie pidió y nadie ofreció un lugar en..."
- Cristina, entre el legado de Néstor, la ruptura y el...
- Carrió quiere alejar a su partido de la "mugre"
- Sigue la búsqueda de tío sospechoso de matar a niña de...
- La Boda Real inglesa se siguió con la máxima tecnología...

actualizado 11:06 Más: [Noticias](#) | [Entretenimiento](#)

Imagen 3.31. Segundo Tab en Yahoo! (www.yahoo.com.ar)

Cabe notar que el orden de estos tabs puede ser modificado por el usuario con solo hacer click en el ícono correspondiente, y que el aspecto de cada uno de ellos es totalmente diferente (incluso se puede notar un pequeño borde en cada uno) con lo que se pretende ayudar al usuario a notar la independencia entre ambos.

3.6. Transicionales

Esta categoría se refiere a patrones donde se utilizan animaciones o efectos especiales durante un corto período de tiempo.

3.6.1. Fade

Aplicar brillo o quitarlo de una cierta zona permite captar la atención del usuario hacia el lugar que corresponda. Esto puede hacerse tanto resaltando la parte sobre la que se quiere hacer foco, como así también oscureciendo el resto de la pantalla o ambos.

No solo el color atrae la atención del usuario, sino que hay que prestar especial atención al movimiento. Con esto se tiene por un lado el contraste que se logra al modificar el color, y por otro la duración de la transición que capta la atención de la persona en mayor medida.

De esta manera se puede comunicar la importancia de un objeto, el estado de la aplicación o cuando un objeto dado se encuentra en uso.

Ventajas y desventajas

- Provee una manera de resaltar cierta información
- Reduce la contaminación visual en la página
- Provee un indicador de estado al usuario

Experiencia de usuario

La experiencia del usuario se ve mejorada ya que se orienta su atención hacia donde corresponde de una manera amena y visualmente atractiva.

Contexto de uso

- Cuando se necesita atraer la atención a una zona de la interfaz, o quitarla de otra
- Cuando se quiere indicar que un objeto está en uso
- Cuando se quiere reducir la contaminación visual en la página, oscureciendo ítems que no se encuentran en uso
- Cuando se quiere mostrar que una página está lista o no

Buenas prácticas

- Utilizar un buen contraste entre la zona resaltada y el resto
- La duración de la transición debe tomar el tiempo justo para llamar la atención del usuario

Posibles errores de uso

- Un uso excesivo puede desorientar al usuario
- Si la duración de la transición es demasiado rápida puede no atraer lo suficiente la atención del usuario
- Si la duración de la transición es demasiado lenta puede ser molesto al usuario

- Un bajo contraste puede que no haga notar la diferencia entre el foco y el resto de la página

Ejemplos

Este comportamiento lo encontramos en diversos sitios, entre otros en el sitio de Puntovista cuyos links en el menú principal se realzan al pasar el mouse por encima de ellos para captar la atención del usuario invitándolo de esa manera a ingresar en cada sección.



Imagen 3.32. Fade en link a "Outsourcing" en el sitio de Puntovista (<http://www.puntovista.com.ar/>)

Por otro lado podemos ver el sitio "Marines", donde utilizan Fade para resaltar la foto sobre la cual se posiciona el mouse que igualmente invita al usuario a visitar la sección.



Imagen 3.33. Fade en fotos de Marines (<http://officer.marines.com/>)

3.7. Patrones de búsqueda

Esta categoría de patrones intentan asistir al usuario mientras éstos intentan buscar cierta información dentro del sitio. Para esto, se provee de cierta información en tiempo real que puede ayudar al usuario a definir su filtro de búsqueda o previsualizar el resultado de sus acciones.

3.7.1. Autocomplete

Este patrón consiste en mostrar posibles entradas al usuario a medida que éste inserta caracteres dentro de un campo de texto. De esta forma el usuario puede evaluar si alguna de

las opciones mostradas es el resultado esperado, eligiendo la opción que corresponda, o por el contrario si ninguna opción se relaciona a su búsqueda puede ingresar una nueva entrada sin haber sido interrumpido en su trabajo.

La diferencia fundamental con un Combo Box (o Select) es que en estos casos no se permite agregar nuevos elementos, mientras que en el patrón autocomplete sí se permite dicha acción.

Ventajas y desventajas

- Permite ingresar datos difíciles de recordar o ambiguos de modo asistido
- Permite prevenir errores mostrando los resultados probables en línea, de esta manera el usuario no tiene que volver a ingresar el campo luego

Experiencia de usuario

El usuario es provisto de una asistencia en línea al completar un campo de texto, sin que por esto deba ser interrumpida su labor.

Contexto de uso

- Cuando la entrada puede ser seleccionada de un conjunto definido de datos
- Cuando la entrada se puede mostrar de varias maneras
- Cuando se pueden repetir o agregar nuevos ítems a los existentes previamente en un campo de texto
- La cantidad de elementos es demasiado grande para mostrar en un Select
- La velocidad y precisión de la entrada es un objetivo principal

Buenas prácticas

- Usar el patrón para asistencia en los campos de texto
- Mostrar los resultados cuando el usuario hace una pausa en la escritura
- Permite el uso de la tecla Tab para seleccionar el ítem correcto
- Basar la entrada en múltiples campos

Posibles errores de uso

- Al presionar la tecla Tab, en lugar de seleccionar la primera coincidencia, se deja el texto ingresado por el usuario
- La asistencia tarda demasiado en mostrar los posibles resultados
- El usuario debe hacer scroll para buscar si un resultado coincide

Ejemplos

Este patrón es utilizado por Yahoo! en su cliente de mail. Al escribir un nuevo mail, o reenviar uno, cuando el usuario elige los destinatarios se muestra a medida que el usuario va ingresando la/s dirección/es de mail una lista con los posibles mails (recogidos de la libreta de contactos del usuario).

Esta búsqueda no solo tiene en cuenta las direcciones de correo, sino que además busca la entrada en el nombre o apodo de cada uno como vemos en la siguiente imagen.

Al presionar la tecla Tab se elige el primer usuario, de otra forma se permite con las flechas del teclado o simplemente haciendo click con el mouse elegir cualquier contacto de la lista.

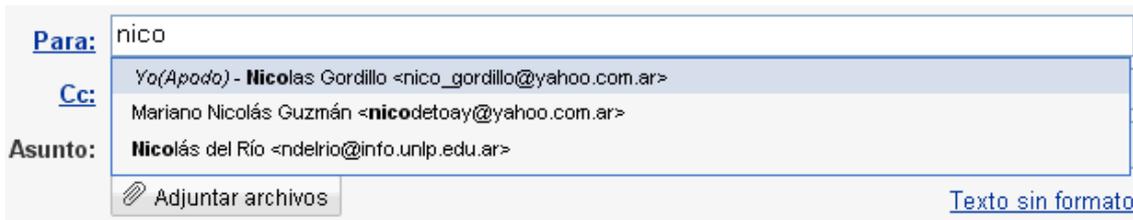


Imagen 3.34. Autocomplete con direcciones de email (<http://www.yahoo.com.ar/>)

Otro ejemplo de uso es en el sitio Ruta 0, donde un usuario puede consultar un mapa con la ruta que debe tomar para llegar de su ciudad de origen hasta la ciudad de destino. En este caso al ingresar los caracteres de cada una de las localidades se muestra un listado con las primeras coincidencias disponibles en la base de datos del sitio.

En este caso la tecla Tab no es utilizada para elegir el ítem deseado con lo cual la búsqueda se hace algo más complicada. Además, la lista suele ser más grande que la pantalla en ciertas búsquedas, lo cual dificulta su visualización ya que se debe hacer scroll para ver toda la información completa.



Imagen 3.35. Autocomplete con localidades (<http://www.ruta0.com/>)

3.7.2. Live Suggest

Este patrón es muy similar al Autocomplete, pero mientras que el anterior busca en los valores reales para una entrada, el Live Suggest provee términos de búsqueda para llevar a cabo la misma.

Ventajas y desventajas

- Permite realizar búsquedas de modo asistido
- Orienta al usuario mostrando las búsquedas más probables o incluso algunas opciones novedosas de acuerdo al contexto
- Puede dificultar la búsqueda si se sugieren entradas alejadas del objetivo del usuario

Experiencia de usuario

Se provee al usuario de una asistencia no intrusiva en su objetivo por completar una búsqueda determinada.

Contexto de uso

- Cuando se quiere brindar asistencia en sus búsquedas al usuario dentro de la misma página
- Cuando el usuario necesita realizar una búsqueda y no está seguro acerca de las palabras claves a utilizar

Buenas prácticas

- Utilizar Live Suggest para sugerir asistencia, no para muestra directa de resultados
- Proveer sugerencias resaltando el contexto en el que se realizan las mismas
- Mostrar sugerencias luego de un delay conveniente
- La interfaz debe ayudar al usuario a recorrer las sugerencias sin distraer al usuario
- Mostrar sugerencias solo si se está seguro del objetivo del usuario, de otra manera hacerlas menos frecuentemente

Posibles errores de uso

- La asistencia tarda demasiado en mostrar resultados
- Los sugerencias se muestran demasiado pronto
- Se muestran sugerencias lejanas al objetivo de la búsqueda, distrayendo al usuario

Ejemplos

Un claro ejemplo de este patrón nos lo provee el buscador más popular en la Web. En el sitio de Google (www.google.com) se provee de un cuadro de búsqueda donde a medida que el usuario ingresa caracteres se muestra una lista con los términos más buscados que tienen relación a la entrada.



Imagen 3.36. Live Suggest con términos de búsqueda sugeridos (<http://www.google.com/>)

3.8. Feedback

Estos patrones necesitan reaccionar inmediatamente para proveer una alta interacción con el usuario. Permiten al usuario saber que está pasando en la aplicación en tiempo real.

3.8.1. Live Preview

Este patrón permite que el usuario entrevea o previsualice su entrada antes de que la aplicación interprete estas entradas. Es un buen método para prevenir errores ya que se evitan antes del momento de aceptación de la entrada.

Ventajas y desventajas

- Permite mostrar los resultados de realizar una determinada acción antes de hacerla
- Permite previsualizar productos
- Permite prevenir errores
- Puede tornar la interfaz demasiado compleja

Experiencia de usuario

La experiencia del usuario se ve mejorada sustancialmente, pudiendo prevenir errores y mostrando las consecuencias de realizar una determinada selección incluso antes de realmente llevarla a cabo. Esto da una sensación de seguridad y confianza en la persona sumada al control de sus elecciones y la posibilidad de experimentar distintas opciones.

Contexto de uso

- Cuando se intenta prevenir errores
- Cuando se pueda mostrar gráficamente la selección del usuario
- Cuando se pueda mostrar la selección en tiempo real
- Utilizar este patrón para evitar transiciones de páginas

Buenas prácticas

- Mostrar feedback de inmediato
- Colocar la previsualización en el mismo contexto que la acción
- Modificar el objeto real al instante
- Si es posible, mostrar la vista previa mientras el usuario provee la entrada. De otra manera, hacerlo al realizar la acción explícitamente
- Utilizar “hover” del mouse para actualizar la vista preliminar

Posibles errores de uso

- La interfaz para mostrar edición y previsualización en la misma página puede ser algo compleja

Ejemplos

Un buen uso de este patrón se encuentra en el sitio de publicidades BlogAds, que permite previsualizar los anuncios a medida que se ingresan los datos del mismo. Al comenzar el proceso se debe elegir el tamaño del anuncio, para más tarde elegir el título, una imagen o video y la dirección Web de la empresa. A medida que se ingresan estos datos se muestra en un pequeño recuadro las modificaciones que se vayan realizando, pudiendo evaluar si es el aspecto deseado y modificándolo en línea hasta lograrlo.

3.9. Validación

3.9.1. Validación en línea

Este patrón es útil para informar al usuario de errores en la entrada antes de terminar de ingresar los mismos.

Ventajas y desventajas

- Previene errores antes de enviar toda la información
- Puede afectar la performance si hay muchas validaciones

Experiencia de usuario

El usuario es prevenido de sus errores antes de terminar de realizar las acciones en la página, con lo cual puede realizar acciones correctivas sin esperar transiciones de página.

Contexto de uso

- Cuando se quieren prevenir errores
- Cuando se puede mostrar un feedback resumido dentro de la pantalla actual

Buenas prácticas

- Si es posible proveer el feedback mientras el usuario escribe, de otra manera hacerlo con algún botón de acción
- Si se realiza feedback mientras se ingresa la entrada, colocarlo cerca de ésta (usualmente a la derecha del campo de entrada)
- Si el feedback se realiza mediante un botón mostrarlo sobre el campo de entrada

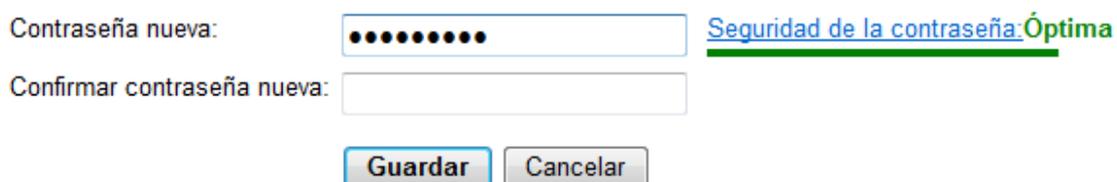
Posibles errores de uso

- Puede comprometer la performance

Ejemplos

Un ejemplo de validación en línea puede ser el cambio de contraseña en Gmail (www.gmail.com). Tal como sucede en la mayoría de los correos electrónicos, la aplicación calcula la seguridad que ofrece la contraseña ingresada en base a la cantidad de caracteres, la combinación de los mismos, o el uso de mayúsculas y minúsculas entre otros criterios.

A medida que el usuario escribe cada carácter obtiene como respuesta la seguridad que ofrecen los caracteres escritos hasta el momento. De esta forma se evita que se ingresen contraseñas demasiado cortas o inseguras, antes de enviar el formulario al servidor.



Contraseña nueva: Seguridad de la contraseña: **Óptima**

Confirmar contraseña nueva:

Imagen 3.39. Inline Validation para evaluar seguridad de clave (<http://www.gmail.com/>)

3.9.2. Contador de caracteres

Este patrón es útil cuando el tamaño de la entrada es estrictamente limitado o la aplicación no puede manejar entradas demasiado grandes correctamente. Permite al usuario conocer en todo momento la cantidad de caracteres que le restan, pudiendo adaptar la entrada a esta restricción.

Ventajas y desventajas

- Permite al usuario adaptar la entrada al tamaño permitido en tiempo real
- Muestra claramente las restricciones de tamaño en un campo determinado

Experiencia de usuario

El usuario conoce las limitaciones de tamaño en tiempo real, evitando de esta manera una recarga de la página mostrando el error o el hecho repentino de estar inhabilitado a seguir escribiendo nuevos caracteres.

Esta conciencia del usuario facilita la tarea de adaptar la entrada deseada respetando los límites que el campo impone.

Contexto de uso

- Cuando la entrada es limitada
- Cuando no se pueden manejar entradas largas de datos

Buenas prácticas

- Actualizar el contador de caracteres en tiempo real
- El texto donde se muestra la cantidad de caracteres restantes debe ser una oración completa con un sentido claro
- Utilizar un tamaño de fuente pequeño, y ubicarlo debajo de la entrada a controlar

Posibles errores de uso

- Utilizar este patrón cuando es posible manejar de alguna manera entradas largas

Ejemplos

En el sitio <http://theamazingrace.canalspace.tv/> se puede ver un claro ejemplo de este patrón. The Amazing Race es un show televisivo donde los participantes deben llevar a cabo una serie de desafíos para proclamarse ganadores del concurso.

Los aspirantes deben inscribirse en el sitio Web, y es importante para los seleccionadores poder conocer el perfil inicial de cada uno con la menor cantidad de información posible.

Es por esto que en el formulario de contacto de este sitio pueden verse varios campos limitados en su tamaño donde los participantes dejan sus experiencias relatadas brevemente.



¿Qué tipo de actividades les gusta hacer juntos? Restan 108 caracteres

Los tipos de actividades que ...

Imagen 3.40. Contador de caracteres (<http://theamazingrace.canalspace.tv/>)

Otro muy conocido ejemplo es el sitio de Twitter, que permite postear mensajes de corta longitud. En este caso la cantidad de caracteres restantes se muestran directamente, sin mostrar una oración completa como remarcamos en “Best Practices”. Quizá esto se deba a la popularidad de este sitio, el cuál basa su publicidad en esta longitud de mensaje, por lo que descarta el uso de una oración descriptiva por algo más directo ya conocido por sus usuarios.



Imagen 3.41. Contador de caracteres en Twitter (<http://www.twitter.com>)

Capítulo 4. Patrones RIA aplicados. Implementación mediante controles

Los controles personalizados y los de usuarios explicados en el capítulo 2 son dos formas diferentes y distintas de crear controles para aplicaciones .NET. Ambos tienen pros y contras, y cada uno tienen ventajas en escenarios particulares.

En el desarrollo de toolkits RIA optamos por elegir controles personalizados por ser código compilado y poder agregarse la librería en cualquier proyecto y además por poseer la capacidad de agregar los controles de la librería a la barra de herramientas del IDE y así facilitar el uso por parte de los desarrolladores que utilicen el toolkit. Sin embargo la razón más importante para la elección de estos controles es el mayor control sobre la funcionalidad a desarrollar que se posee y la posibilidad de crear controles más ricos.

Entre los controles que hemos desarrollado se pueden contar:

- Autocomplete
- Rating an object
- Datepicker
- Character count
- Fade
- Inline validation
- Accordion
- Tooltip

Cada uno de estos controles aporta una serie de beneficios al usuario, ofreciéndole cierta flexibilidad para utilizarlo en distintas aplicaciones con diferentes estilos y formatos. Consigo trajeron aparejadas una serie de desafíos y dificultades al ser desarrollados que explicaremos a continuación, en general relacionadas con la tecnología disponible. Como se ha comentado antes en el presente trabajo, desarrollar aplicaciones fuertemente acopladas en una configuración débilmente acoplada conlleva mayor complejidad con la que deben lidiar los programadores y diseñadores del sistema.

4.1. Autocomplete

Descripción

Este control nos agrega la funcionalidad de autocompletado de un campo de entrada con valores filtrados por el sistema. Para ello se asocia con un TextBox (control nativo del framework .NET), para permitir a los usuarios encontrar rápidamente un valor, aprovechando la búsqueda y el filtrado.

Al ingresar un texto en el TextBox asociado, el autocomplete empieza a buscar las entradas coincidentes y muestra el primer valor filtrado. Al entrar más caracteres, el usuario puede generar un filtro más específico.

Facilidad de implementación

Al momento de agregar la funcionalidad AutoComplete en un website, este control nos permite implementarlo de manera simple y efectiva. Solo necesitamos agregar el control al

web form y asociarlo a un campo de entrada, luego definir un método en la clase servidor del web form que dado un string de entrada con el texto a buscar, nos retorne el primer valor string que nos devuelva la búsqueda. Es de destacar la manera en que esta implementado el control: el dato que envía al servidor a través de Ajax es solo el texto de búsqueda, sin enviar el estado de la página o ViewState, con lo cual la respuesta del servidor es más rápida.

Customización

ControlToLink

Control al cual se le asociara la función de AutoComplete. El control será usado para el ingreso del texto a buscar. Se admiten solo controles del tipo TextBox.

MethodName

Indica el nombre del método "autocompleteMethod" que se ejecutará en el servidor. El método se desencadena una vez que se han tipeado como mínimo 3 caracteres y pasados 400 milisegundos de espera de actividad sobre el TextBox. Estas restricción son pre-impuestas, en el primer caso para no realizar consultas en el servidor que generen filtros poco específicos, ya que este caso generaría retrasos y esperas al usuario sin generarle una experiencia ágil; y en el segundo caso para no realizar una petición al servidor por cada tecla que se tipea, en caso del usuario conocer el filtro a aplicar, el tiempo promedio de tpeo entre caracteres no supera los 400 milisegundos, como consecuencia para filtrar por una palabra se generaría una sola petición al servidor y para el usuario resultados específicos más rápidamente.

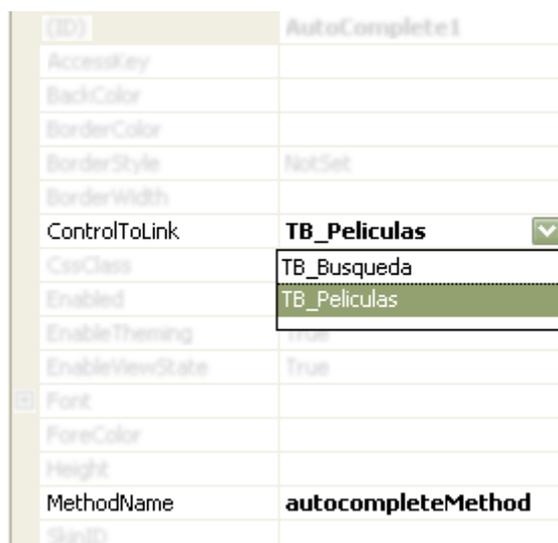


Imagen 4.1. Propiedades del control Autocomplete.

Una vez especificado el nombre el método que se ejecutara en el servidor, lo siguiente a realizar es la implementación de dicho método en el código ASP de la página contenedora. Este método contiene algunas particularidades que deben tenerse en cuenta al momento de la implementación si queremos que el control funcione correctamente. La primer cuestión a tener en cuenta es agregarle al método implementado el atributo "[System.Web.Services.WebMethod]", esto se debe a que el método debe interactuar con código cliente como si se tratase de un servicio web. La segunda cuestión es declarar el método como estático (static) en el código ASP de la página, esto se debe a que el método debe ser invocado desde el código cliente, y como allí no tenemos disponible la instancia de la

clase "System.Web.UI.Page" que corresponde a la página contenedora, por lo tanto, la única forma de invocar a un método desde el código cliente de la página es declarándolo como estático. Y la tercer y última cuestión a tener en cuenta es la interface del método autocomplete, este método recibe como parámetro un `string` (texto ingresado) y debe retornar como salida otro `string` (texto resultante de la búsqueda). Un ejemplo de cómo quedaría un método correctamente implementado en el código ASP de la página sería:

```
[System.Web.Services.WebMethod]
public static string autocompleteMethod(string text)
{
    //Código que realiza el filtrado de las opciones
    return text;
}
```

Experiencia de usuario

Con el uso de este control, el desarrollador agrega una funcionalidad muy requerida en los website actuales (como es la ayuda al usuario en las búsquedas) con solo soltar el control en la página desde la paleta de herramientas y configurar unas pocas propiedades ya mencionadas, despreocupándose de la comunicación con el servidor, así como también de la lógica de presentación de los datos recibidos.

4.2. Inline validation

Descripción

Este control nos permite validar entradas de datos en una página. Para ello se asocia con un formulario HTML, el cual actuará como desencadenador de la validación de los datos antes del envío de la página. Además posee una propiedad que nos permitirá cargar los controles a ser validados, indicando el tipo de validación y la obligatoriedad. En caso de haber errores en la carga, el control nos mostrara junto al control validado, el mensaje con la descripción del error a corregir.

Facilidad de implementación

Al momento de agregar la funcionalidad Inline Validation en un website, este control nos permite implementarlo de manera simple y rápida. Solo necesitamos agregar el control al web form y asociarlo a un formulario de entrada de datos, luego se debe agregar de manera sencilla (panel visual) cada campo de entrada de datos, eligiendo el tipo de validación a aplicar. El control disparará la validación antes del envío de datos del formulario.

Customización

ShowImageState

Esta propiedad establece si se muestran las imágenes que representan el estado de la validación. Los posibles valores son True o False.

FormToValidate

Esta propiedad establece el formulario HTML al cual se asociará la validación. En el evento submit de este formulario es donde el control desencadenara la validación. En caso de haber algún error en el ingreso de los datos, el control no hará el envío y mostrará el mensaje de error correspondiente.

ForeColor	Red
FormToValidate	aspnetForm
Height	
Items	(Colección)
ShowImageState	True
Skid	

Imagen 4.2. Propiedades del control Inline Validation.

Items

Esta propiedad representa el conjunto de controles a ser validados. Para asociarlos nos despliega de manera visual una ventana donde podemos ir agregando uno a uno los controles, indicando para cada uno de ellos las siguientes propiedades:

- **ControlToLink:** Esta propiedad representa control al cual se le asociará la validación. El control a enlazar, debe ser un control de algunos de los siguientes tipos: TextBox, DropDownList, ListBox, CheckBox, RadioButton, RadioButtonList.
- **ValidationMode:** Esta propiedad representa el modo de validación sobre el control asociado. Entre los posibles modos se encuentran las validaciones por tipo Decimal, Mail, Fecha, Url, Alfanumérico, Entero.
- **Required:** Esta propiedad representa si el campo es obligatorio a completar o no. Los posibles valores son True o False.

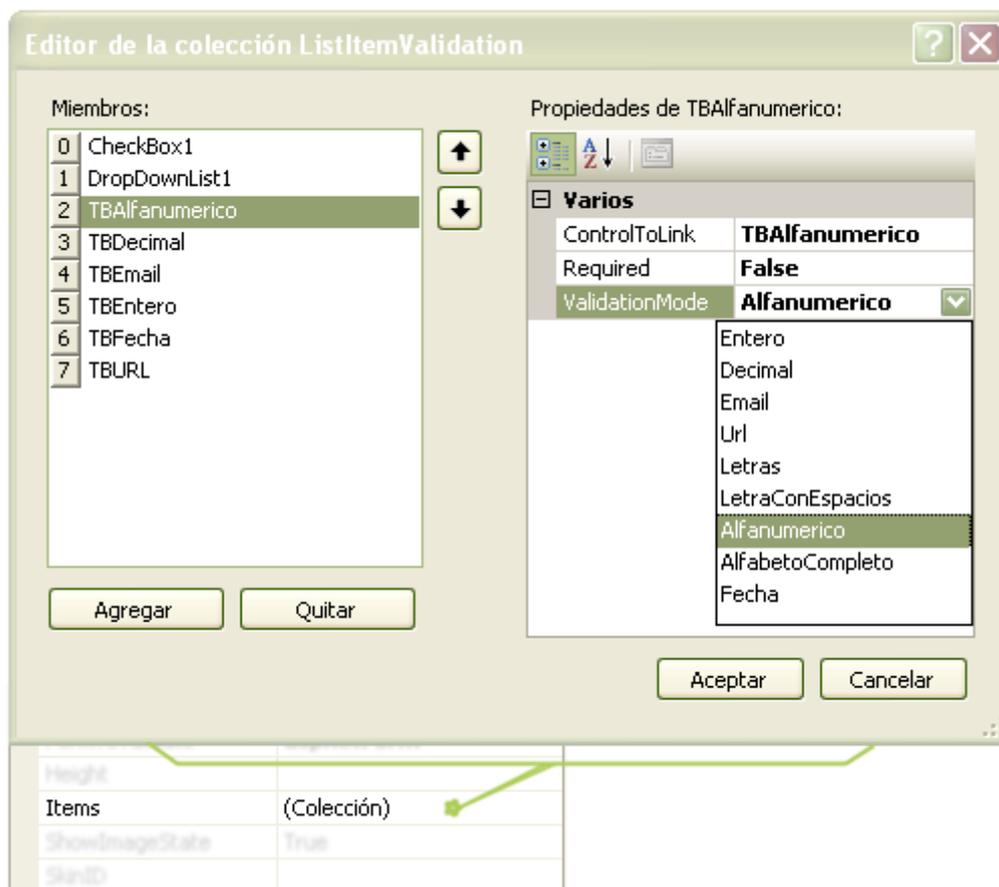


Imagen 4.3. Propiedad Item de control Inline Validation.

Experiencia de usuario

Disponible desde la paleta de herramientas, el control encapsula toda la lógica del patrón Inline Validation. El desarrollador solo debe definir el formulario de ingreso de datos. Luego, utilizando el control, lo enlaza con el formulario definido, agrega cada control que desea validar eligiendo el tipo de validación y obtiene de forma rápida esta funcionalidad, que además implementa las mejores prácticas de uso.

4.3. Control Fade

Descripción

Este control se enlaza con un control contenedor, para generar de forma sencilla un efecto de opacidad en el hover de sus elementos. Entre los posibles elementos a realizar el efecto, se encuentran las imágenes, párrafos, listas y filas de una tabla.

Facilidad de implementación

Simple de usar. Este control solo requiere que se lo enlace con un control contenedor, y se elija a que elementos se va a aplicar el efecto.

Además cuenta con otras opciones de customización, entre las que se incluyen el nivel de opacidad, y la duración del efecto.

Customización

ControlToLink:

Control al cual se le asociara el Fade. El control a enlazar, debe ser un control contenedor, el cual contenga imágenes, listas o párrafos. Se admiten solo los tipos Panel, GridView, y Datalist como contenedores.

Opacity:

Determina el nivel de opacidad a aplicar en los elementos.

Esta propiedad es de tipo enum EOpacity y los valores posibles son:

- Low: opacidad apenas perceptible.
- Medium: opacidad media sobre el elemento.
- High: opacidad alta sobre el elemento.

Duration:

Determina la duración de la transición del efecto de apertura y cerrado.

Esta propiedad es de tipo enum EDuration y los valores posibles son:

- Slow: transición lenta.
- Medium: transición media.
- Quick: transición rápida.

ApplicationType:

Determina el tipo de aplicación del control.

Esta propiedad es de tipo enum EApplicationType y los valores posibles son:

- Image: aplica fade a todas las imagenes (elemento html img) contenidas en el control contenedor ControlToLink.
- Paragraph: aplica fade a todos los párrafos (elemento html p) contenidos en el control contenedor ControlToLink.
- List: aplica fade a todos los elementos de una lista (elemento html li) contenidas en el control contenedor ControlToLink.

- Row: aplica fade a todos los elementos fila de una tabla (elemento html tr) contenidas en el control contenedor ControlToLink.

(Expressions)	
(ID)	Fade1
AccessKey	
ApplicationType	Image
BackColor	Image
BorderColor	Paragraph
BorderStyle	List
BorderWidth	Row
ControlToLink	GV_Peliculas
CssClass	
Duration	Medium
Enabled	Slow
EnableTheming	Medium
EnableViewState	Quick
Font	
ForeColor	
Height	
Opacity	Medium
SkinID	Low
TabIndex	Medium
ToolTip	High
Visible	
Width	

Imagen 4.4. Propiedades del control Fade.

Experiencia de usuario

Disponible desde la paleta de herramientas, el control encapsula toda la lógica del patrón fade. El desarrollador que desea utilizar el patrón, utiliza este control, y lo asocia con un contenedor, de una manera muy sencilla.

4.4. Rating an object

Descripción

Este control está compuesto por un conjunto de elementos interactivos (a menudo estrellas), que muestra la valoración de uno o varios objetos asociados, e invita al usuario a que exprese su valoración sobre cada uno de ellos.

El estado inicial representa sobre cada objeto, un valor que se obtiene del promedio de los puntos recibidos.

A medida que el cursor del ratón pasa sobre los iconos, van cambiando de estado indicando la valoración. Al hacer clic sobre uno de los iconos, se guarda la calificación y se actualiza la valoración promedio, y el control queda en un estado deshabilitado para volver a valorar.

Facilidad de implementación

Este control se incorpora de manera sencilla a la página web para permitirnos implementar el patrón *Rating*, solo requiere que se lo enlace con un control y se elija con que apariencia se mostrará al usuario, para así proveer de manera amigable la funcionalidad requerida.

Además cuenta con distintas combinaciones para adecuar el comportamiento a nuestra necesidad.

Customización

UrlImageOff

Url de la imagen que representa un icono no activo.

UrlImageOn

Url de la imagen que representa un icono activo.

UrlImageLoading

Url de la imagen que actúa como loading, la cual se muestra en la transición en que se guarda la calificación.

CantTotalesDeVotos

Propiedad que setea a la instancia del control, la cantidad total de votos recibidos.

CantTotalesDePuntos

Propiedad que setea a la instancia del control, la cantidad total de puntos recibidos.

CantItem

Cantidad de items o iconos en que se sub-divide el control. Por defecto es 5.

Indice

Propiedad que permite el uso del control en colecciones de datos. Al ser un control usado frecuentemente en colección dinámicas de objetos, una característica con la que nos encontraremos habitualmente es que no sabemos a priori con qué cantidad de objetos debemos enlazar. La solución sugerida es incluir el control dentro de la colección dinámica, generándose N controles *Rating* en cada ejecución (uno por cada objeto de la colección). Esta propiedad nos permite especificar el índice o campo clave que nos permita identificar a que objeto de la colección hacemos referencia en el *MethodName* invocado, el cual es compartido por todos los controles *Rating* generados dentro de la misma.

MethodName

Nombre del método que se ejecuta en el servidor cuando el control es clickeado eligiendo una valoración.

Una vez especificado el nombre el método que se ejecutara en el servidor, lo siguiente a realizar es la implementación de dicho método en el código ASP de la página contenedora. Este método contiene algunas particularidades que deben tenerse en cuenta al momento de la implementación si queremos que el control funcione correctamente. La primer cuestión a tener en cuenta es agregarle al método implementado el atributo

`"[System.Web.Services.WebMethod]"`, esto se debe a que el método debe interactuar con código cliente como si se tratase de un servicio web. La segunda cuestión es declarar el método como estático (`static`) en el código ASP de la página, esto se debe a que el método debe ser invocado desde el código cliente, y como allí no tenemos disponible la instancia de la clase `"System.Web.UI.Page"` que corresponde a la página contenedora, por lo tanto, la única forma de invocar a un método desde el código cliente de la página es declarándolo como

estático. Un ejemplo de cómo quedaría un método correctamente implementado en el código ASP de la página sería:

```
[System.Web.Services.WebMethod]
public static void Rating_Demo_Changed(int indice, int rate, int votos,
int puntos)
{
    //Código que registra la valoracion del objeto
}
```

TextRateIt

Texto inicial que se muestra en el hover de los iconos del control.

TextVotes

Texto que se muestra en el hover de los iconos del control para informar los votos recibidos.

TextPoints

Texto que se muestra en el hover de los iconos del control para informar los puntos recibidos.

BorderColor	
BorderStyle	NotSet
BorderWidth	
CantItem	5
CantTotalesDePuntos	0
CantTotalesDeVotos	0
CssClass	
Enabled	True
EnableTheming	True
EnableViewState	True
Font	
ForeColor	
Height	
Indice	-1
MethodName	Rating_Control_Changed
SkinID	
TabIndex	0
TextPoints	puntos recibidos sobre
TextRateIt	ranqueame!
TextVotes	votos.
ToolTip	
UrlImageLoading	
UrlImageOff	
UrlImageOn	
Visible	True
Width	

Imagen 4.5. Propiedades del control Rating.

Experiencia de usuario

El desarrollador agrega una funcionalidad compleja de manera sencilla, debiendo preocuparse solo en el almacenamiento y la calificación de cada ítem asociado, teniendo ya encapsulado en el control, toda la lógica de presentación del patrón, y contando con la posibilidad de

customización a través de varias propiedades públicas, y despreocupándose de la validación del campo, ya que siempre contendrá un valor de formato fecha.

4.5. Character count

Descripción

Este control nos permite limitar la cantidad de caracteres a ingresar en un campo de entrada de datos en una página. Para ello se asocia con un TextBox (control nativo del framework .NET), para permitir a los usuarios ir visualizando la cantidad de caracteres restantes posibles a ingresar.

Al ingresar caracteres en el TextBox asociado, el Character Count va decrementando la cantidad de caracteres permitidos.

Facilidad de implementación

Al momento de agregar la funcionalidad Character Count en un website, este control nos permite implementarlo fácilmente y en pocos pasos. Solo necesitamos agregar el control al web form y asociarlo a un campo de entrada, luego especificar la cantidad máxima de caracteres permitidos y el mensaje que se muestra indicando la cantidad de caracteres restantes.

Customización

ControlToLink

Esta propiedad establece el control al cual se le asociara la función de Character Count. Se admiten solo controles del tipo TextBox.

MaxLength

Esta propiedad establece la cantidad máxima de caracteres de ingreso permitidos en el control asociado.

Text

Esta propiedad establece el texto que se muestra para indicar la cantidad de caracteres restantes para el ingreso.

(Expressions)	
(ID)	CharacterCount2
AccessKey	
ControlToLink	TB_Comentario
CssClass	
Enabled	True
EnableTheming	True
EnableViewState	True
Font	
ForeColor	White
Height	
MaxLength	100
SkinID	
TabIndex	0
Text	caracteres restantes.
ToolTip	
Visible	True
Width	

Imagen 4.6. Propiedades del control CharacterCount.

Experiencia de usuario

Disponible desde la paleta de herramientas, el control encapsula toda la lógica del patrón Character Count. El desarrollador solo debe elegir el campo de ingreso de datos. Luego, utilizando el control, lo enlaza al campo agregando esta funcionalidad, que además implementa las mejores prácticas de uso.

4.6. Control Accordion

Descripción

Este control se asocia con un conjunto de elementos, para proveer la funcionalidad del patrón accordion.

Estos elementos asociados, deben tener la siguiente estructura:

- Una parte llamada “título”, que es la que va a permanecer siempre visible.
- Una parte llamada “información ampliada”, que es la que va a estar oculta.

Cada ítem plegado puede ser ampliado de forma individual, para mostrar su información.

Las propiedades principales del control son `ClassNameTitle` y `ClassNameInfo`. La primera indica cual es el “título” del ítem, y la segunda cual es la “información ampliada”, utilizando para ello la información de la clase CSS del elemento HTML.

Facilidad de implementación

Este control se incorpora de manera sencilla a la página web, utilizando para la asociación la clase css de los elementos.

Además cuenta con distintas opciones de customización, para adecuar el comportamiento y apariencia del control a un gran abanico de contextos.

Customización

ClassNameTitle:

ClassName que actúa como título del Accordion.

ClassNameInfo:

ClassName que actúa como información ampliada del Accordion.

EventType:

Evento que desencadena la apertura del Accordion.

Esta propiedad es de tipo enum `EEventType` y los valores posibles son:

- Click: la apertura se realiza al hacer clic sobre un elemento de clase `ClassNameTitle`.
- Hover: la apertura se realiza en el hover sobre un elemento de clase `ClassNameTitle`.

Duration:

Determina la duración de la transición del efecto de apertura y cerrado.

Esta propiedad es de tipo enum `EDuration` y los valores posibles son:

- Slow: transición lenta.
- Medium: transición media.
- Quick: transición rápida.

Animation:

Determina la animación en la transición del efecto.

Esta propiedad es de tipo enum EAnimation y los valores posibles son:

- Swing: la animación de apertura y de cierre transcurren a una velocidad constante y son simultáneas.
- EaseInExpo: primero se realiza la animación de apertura que transcurre a velocidad constante. Luego el cierre se produce a una velocidad mucho mayor.
- EaseInElastic: primero se realiza la animación de apertura que transcurre a velocidad constante. Luego en el cierre se agrega un efecto de elasticidad.
- EaseInBack: primero se realiza la animación de apertura que transcurre a velocidad constante. Luego se realiza el cierre de forma similar.
- EaseInBounce: primero se realiza la animación de apertura que transcurre a velocidad constante. Luego en el cierre se agrega un efecto de rebote.

Deployed:

Determina si el Accordion comienza con su primer ítem desplegado.

(Expressions)	
(ID)	Accordion1
AccessKey	
Animation	Swing
BackColor	
BorderColor	
BorderStyle	NotSet
BorderWidth	
ClassNameInfo	acordeon_desa
ClassNameTitle	acordeon_titulo
ClrClass	
Deployed	False
Duration	Medium
Enabled	True
EnableTheming	True
EnableViewState	True
EventType	Click
Font	
ForeColor	
Height	
SkinID	
TabIndex	0
ToolTip	
Visible	True
Width	

Imagen 4.7. Propiedades del control Acordion.

Experiencia de usuario

Disponible desde la paleta de herramientas, el control encapsula toda la lógica del patrón accordion. El desarrollador tiene que enfocarse en la definición de la estructura de los elementos plegables. Luego, utilizando el control, lo enlaza con la estructura, y obtiene de forma sencilla esta funcionalidad de gran utilidad cuando se desea optimizar el espacio.

4.7. Control Tooltip

Descripción

Este control provee información ampliada que se muestra cuando el ratón pasa sobre un elemento. Está diseñado para ser sencillo de aplicar, tomando el atributo *Title* de los elementos como fuente de información; y un estilo visual totalmente configurable a través de CSS.

Facilidad de implementación

Este control solo requiere que se lo enlace con un control contenedor para proveer la funcionalidad.

Se puede utilizar el atributo `CssClass` para configurar los estilos visuales desde el `css`.

Customización

ControlToLink:

Control al cual se le asociara el Tooltip. El control a enlazar, debe ser un control contenedor. Se admiten solo los tipos `Panel`, `GridView`, y `Datalist` como contenedores.

Duration:

Determina la duración de la transición del efecto.

Esta propiedad es de tipo `enum EDuration` y los valores posibles son:

- `Slow`: transición lenta.
- `Medium`: transición media.
- `Quick`: transición rápida.

CssClass:

Referencia al `className` que le da estilos al tooltip.

(Expressions)	
(ID)	Tooltip2
AccessKey	
BackColor	#454545
BorderColor	#383E3A
BorderStyle	Solid
BorderWidth	3px
ControlToLink	P_Custom
CssClass	tooltipCustom
Duration	Medium
Enabled	True
EnableTheming	True
EnableViewState	True
Font	
ForeColor	White
Height	
SkidID	
TabIndex	0
ToolTip	
Visible	True
Width	

Imagen 4.8. Propiedades del control ToolTips.

Experiencia de usuario

Disponible desde la paleta de herramientas.

El desarrollador que desea implementar el patrón, utiliza este control, y lo asocia con un contenedor, de una manera muy sencilla.

4.8. Control DatePicker

Descripción

Este control proporciona una interfaz gráfica familiar que ayuda a los usuarios a seleccionar fácilmente la fecha de un evento, cumpleaños, o un rango de fechas. Por defecto establece el día actual, y anula los campos de entrada de datos a fin de garantizar la validez sobre el campo fecha.

Facilidad de implementación

Este control se incorpora de manera sencilla a la página web, solo requiere que se lo enlace con un control y se elija con que apariencia se mostrará al usuario, para así proveer de manera amigable la funcionalidad requerida.

Además cuenta con distintas combinaciones para adecuar el comportamiento a la nuestra necesidad.

Customización

ControlToLink:

Control al cual se le asociara el datepicker. El control a enlazar, debe ser del tipo TextBox.

ControlToCompare:

Control al cual se le asocia una fecha de comparación del datepicker. Esta combinación genera un nuevo comportamiento en el control, ya que la fecha máxima elegida TextBox asociado al ControlToLink está limitada por la fecha elegida en el TextBox asociado a esta propiedad, y la fecha mínima elegida en el TextBox asociado a esta propiedad está limitada por la fecha elegida en el TextBox asociado al ControlToLink.

DataFormat:

Determina el formato de la fecha del datepicker. Esta propiedad es de tipo enum EDateFormat y los valores posibles son:

- ddmmaaaa: establece una fecha de la forma dd/mm/aaaa. Ejemplo: 17/08/2010.
- dd_mm_aaaa: establece una fecha de la forma dd-mm-aaaa. Ejemplo: 17-08-2010.
- aaaammdd: establece una fecha de la forma aaaa/mm/dd. Ejemplo: 2010/08/17.
- aaaa_mm_dd: establece una fecha de la forma aaaa-mm-dd. Ejemplo: 2010-08-17.

Theme:

Determina el theme o estilo del datepicker. Esta propiedad es de tipo enum ETheme y los valores posibles son:

- Smoothness: calendario con tonalidad gris.
- Cupertino: calendario con tonalidad celeste.

MinDate: Fecha mínima válida de elección del datepicker. Si no se elige una fecha, el datepicker no tendrá la restricción de fecha mínima.

MaxDate: Fecha máxima válida de elección del datepicker . Si no se elige una fecha, el datepicker no tendrá la restricción de fecha máxima.

(Expressions)	
(ID)	DatePicker2
AccessKey	
BackColor	
BorderColor	
BorderStyle	NotSet
BorderWidth	
ControlToCompare	
ControlToLink	TB_FechaMes
CssClass	
DataFormat	ddmmaaaa
Enabled	True
EnableTheming	True
EnableViewState	True
Font	X-Small
ForeColor	
Height	
MaxDate	
MinDate	
SkinID	
TabIndex	0
Theme	Smoothness
ToolTip	
Visible	True
Width	

Imagen 4.9. Propiedades del control DatePicker.

Experiencia de usuario

Disponible desde la paleta de herramientas.

El desarrollador que desea implementar el patrón, utiliza este control, y lo asocia con uno o más controles, pudiendo generar distintos comportamientos que abarcan una gran cantidad de necesidades. Así de una manera muy sencilla se puede implementar el siempre complejo problema de la elección de una fecha válida, sin la necesidad adicional de la validación posterior.

Capítulo 5. Caso de Estudio: Pandora. Website del toolkit

5.1. Introducción

En capítulos anteriores se explicaron los distintos patrones RIA y sus ventajas. Luego varios de estos patrones se llevaron a la práctica mediante controles y como último desafío nos propusimos mostrar el uso de estos controles que implementa los patrones RIA, de ahí nace Pandora como caso de estudio de esta tesis.

Al plantearnos el WebSite Pandora como un caso de estudio, también debíamos enfrentar el reto de mostrar el ToolKits de forma correcta y dinámica. Es decir, el WebSite debía contener los conceptos y aplicar los patrones estudiados, brindando una rica experiencia de usuario y alejándose de las webs tradicionales. Y que mejor manera de aplicar los patrones estudiados que con los controles desarrollados para el ToolKits, con lo cual el desafío más importante fue mostrar y explicar los controles desarrollados con los mismos controles.

5.2. Estructura del Website

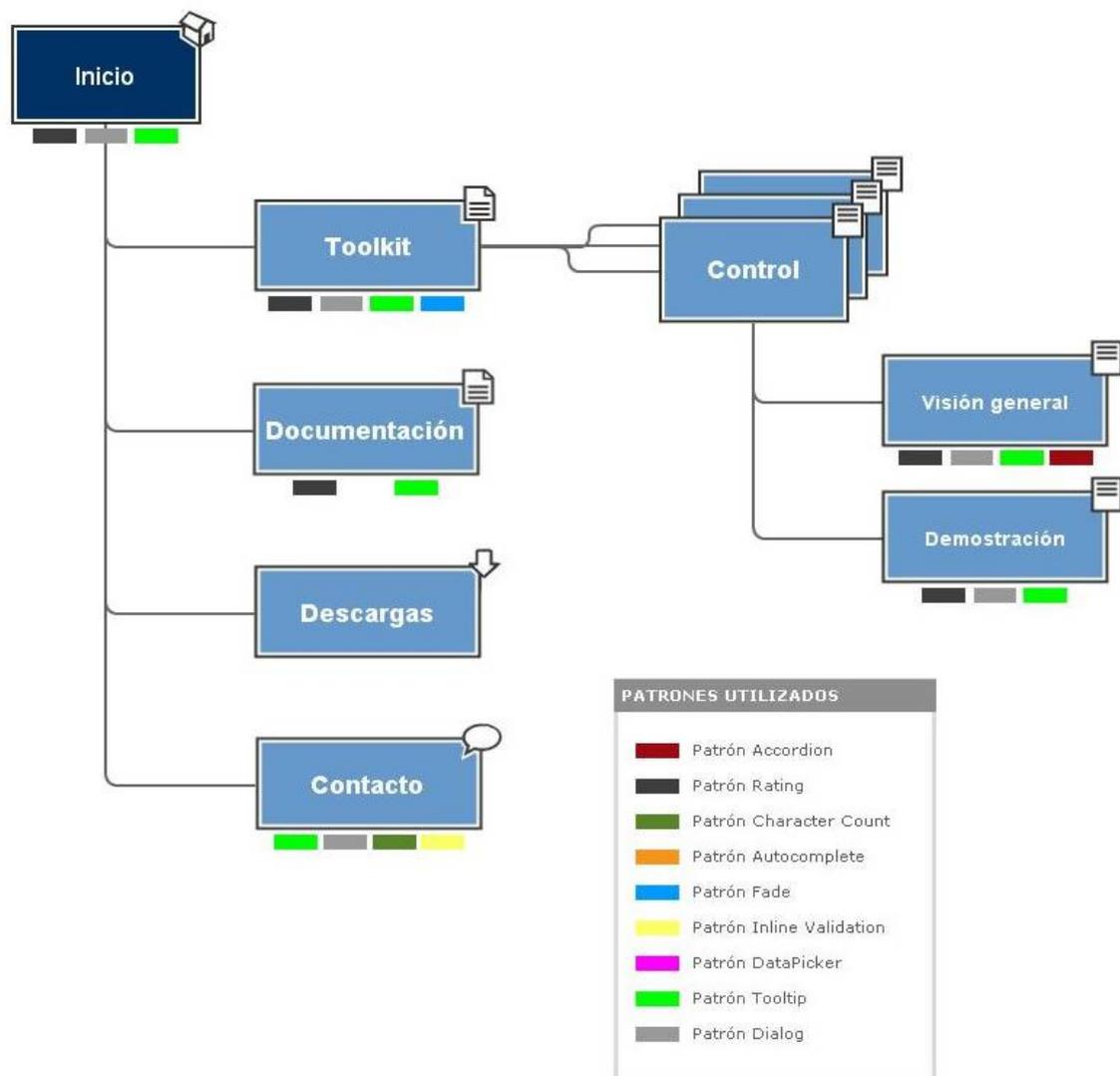


Imagen 5.1. Diagrama de navegación del Website.

5.3. Explicación de los patrones aplicados en el website

Patrón Accordion

Este patrón, está implementado por el control Accordion del Toolkit. Es utilizado para presentar dentro de la pantalla de visión general de un control, su facilidad de implementación y experiencia de usuario.

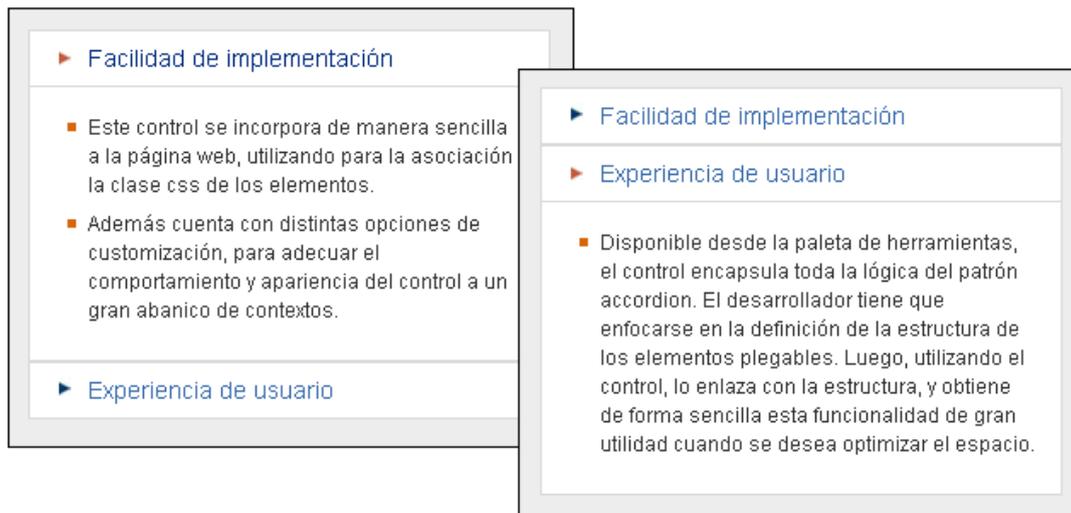


Imagen 5.2. Patrón Accordion.

La ventaja que genera la implementación de este patrón en el website, es que nos permite mostrar la información detallada de una de las características ya mencionadas, ocultando el detalle de las demás, pero dejando siempre visible el título de la característica para poder desplegar la información detallada sin necesidad de recargar la página.

Patrón Rating

Este patrón, está implementado por el control RatingObject del Toolkit. Es utilizado para presentar tanto dentro de la pantalla de visión general del WebSite, así como en las pantallas

visión general y demostración particular de cada control, la valoración actual del control, dado por los votos recibidos y permitir la valoración del mismo en caso de no haberlo hecho.



Imagen 5.3. Patrón Rating.

La ventaja que genera la implementación de este patrón en el website, es que permite al usuario de una manera simple interactuar con el website, sintiéndose parte contribuyente y aportando información valiosa para la generación del top 3 de los controles mejores valuados presentada en el home del website. De igual manera permite al usuario ver de forma rápida un promedio de valuación de cada control.

Patrón Character Count

Este patrón, está implementado por el control CharacterCount del Toolkit. Es utilizado para presentar al usuario la cantidad de caracteres restantes permitidos en el campo comentario del formulario de contacto del website.

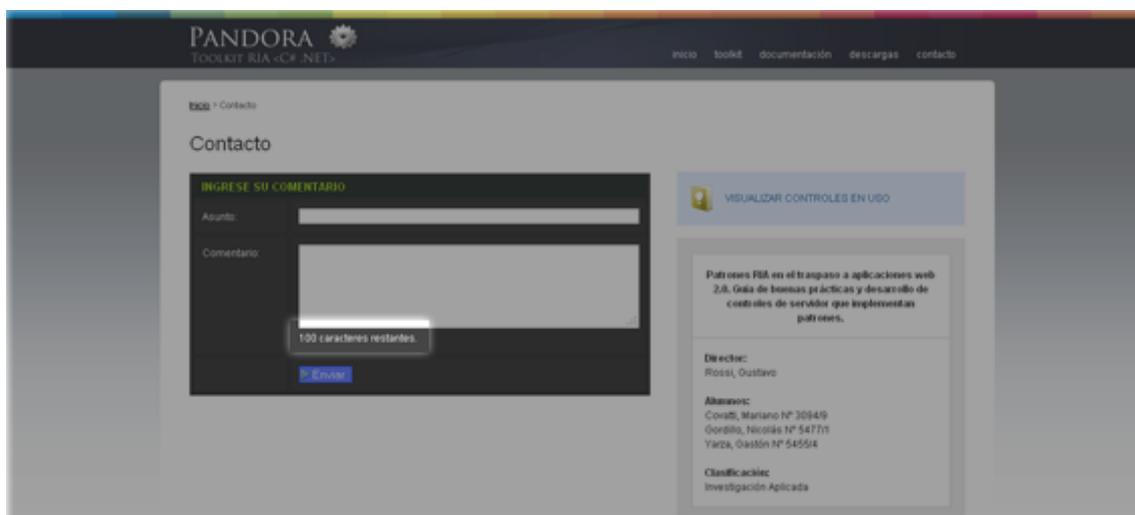


Imagen 5.4. Patrón Character Count.

La ventaja que genera la implementación de este patrón en el website, es que permite al usuario saber en tiempo real y sin recarga innecesarias, conocer el tamaño de la entrada, y así poder aprovechar al máximo la cantidad de caracteres permitidos.

Patrón Fade

Este patrón, está implementado por el control Fade del Toolkit. Es utilizado en la página de visión general del website donde son presentados todos los controles del toolkits, para resaltar el control sobre el que estamos dispuesto a acceder a su especificación, aplicándole más brillo que al resto de los controles presentados en la página.



Imagen 5.5. Patrón Fade.

La ventaja que genera la implementación de este patrón en el website, es que permite orientar al usuario dentro de la lista de controles presentados, resaltando aquel donde posamos el puntero del mouse, esto sin contaminar la visual de la página, ya que es realizado aplicándole más brillo al control resaltado.

Patrón Inline Validation

Este patrón, está implementado por el control InlineValidation del Toolkit. Es utilizado para validar como requerida la entrada del campo asunto en el formulario de contacto del website.

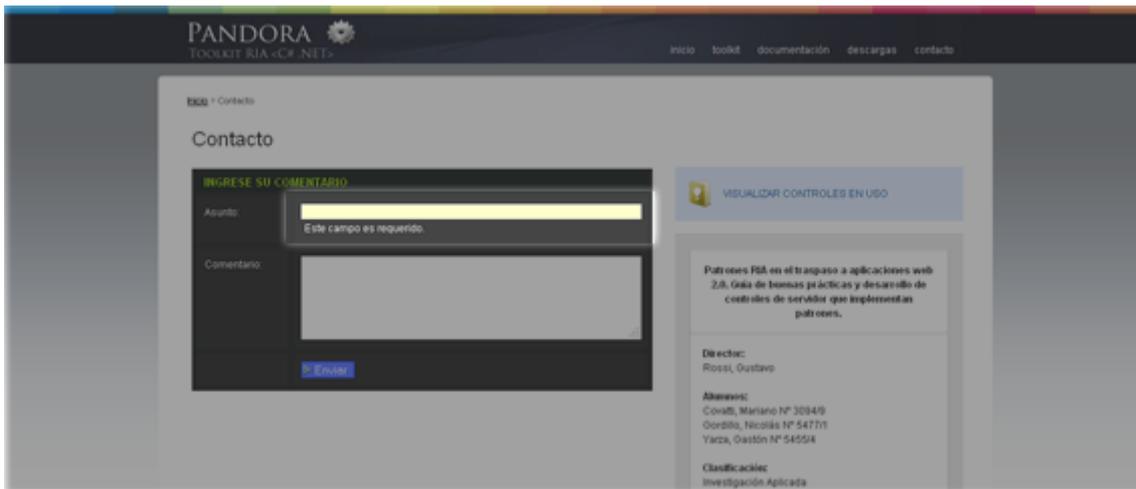


Imagen 5.6. Patrón In Line Validation.

La ventaja que genera la implementación de este patrón en el website, es que permite validar la entrada en el campo asunto en el formulario de contacto del website, no permitiendo el envío del mail sin haber llenado ese campo.

Patrón Tooltip

Este patrón, está implementado por el control Tooltip del Toolkit. Es utilizado para presentar al usuario una explicación descriptiva de las propiedades customizables de cada control, estas características son presentadas en la sección de demostración de cada control en el website.

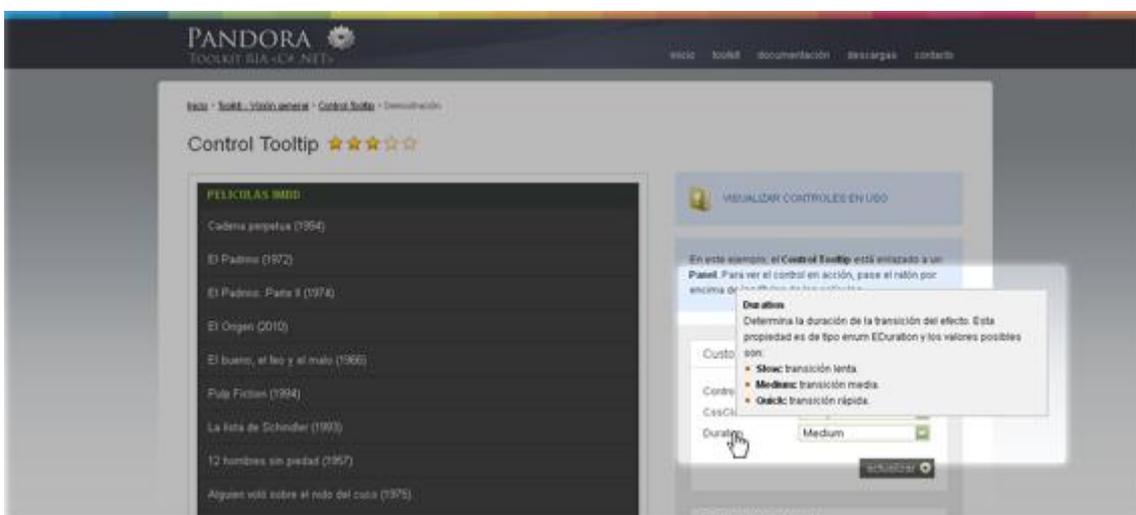


Imagen 5.7. Patrón Tooltip.

La ventaja que genera la implementación de este patrón en el website, es que permite ampliar la información de cada propiedad customizable de los controles sin moverse del contexto en el que se encuentra. Al usuario le genera una sensación de satisfacción ya que con solo posarse sobre la propiedad les brinda la información detallada.

Patrón Dialog

Este patrón, no está implementado por medio de un control del Toolkit. Es utilizado para presentar al usuario la información de los patrones implementados en cada página del website.



Imagen 5.8. Patrón Dialog.

La ventaja que genera la implementación de este patrón en el website, es que permite presentar la información de los patrones implementados de rápida y sin perder el contexto en el cual se encuentra el usuario. Además permite una mejor experiencia visual ya que se tiene un completo control sobre la apariencia del dialogo abierto, no dependiendo del sistema operativo ni del navegador que utilice el usuario.

Capítulo 6. Conclusión y trabajos futuros

6.1. Conclusión

Las aplicaciones web han cambiado desde sus orígenes, no hace tanto en el tiempo, y hasta el día de hoy han mostrado sucesivas transformaciones que reemplazaron los documentos de hipertexto con los complejos sitios Web actuales, cada vez más similares en presentación e interacción con el usuario a las aplicaciones de escritorio.

La ingeniería de software ha estado acompañando este cambio, pero no lo suficiente. Se encuentran disponibles varias aproximaciones a metodologías de desarrollo RIA, pero sin embargo no son ampliamente utilizadas. Los diseñadores y programadores han encontrado soluciones caseras a los distintos problemas que se le presentaban, y se ha hecho muy común en estos días la abstracción de estas soluciones en patrones, que describen el problema y la solución de manera explícita.

Nuestro objetivo es simplificar el traspaso de aplicaciones tradicionales a las aplicaciones RIA, dado que éstas son más complejas naturalmente se necesita plantear el problema desde varios puntos de vista. Lo que intentamos hacer en este trabajo es facilitar la adopción de patrones RIA en la práctica. Para ello nos valimos de los controles de servidor que brinda .net, y encapsulamos la funcionalidad de cada patrón en ellos de manera que el programador pueda utilizar esta solución de la manera más simple posible.

Nos pareció muy interesante cómo a través del uso de un control, que encapsula de cierta forma el comportamiento de un patrón, el desarrollador tiene una poderosa herramienta a su disposición para de una manera relativamente sencilla, agregar funcionalidad RIA a las aplicaciones web, pudiendo concentrarse de esta forma en el problema de su dominio.

Como complemento a la plataforma .net hemos elegido jQuery, un framework con gran cantidad de plugins que implementan funcionalidades RIA en el cliente. De esta forma, la tarea de realizar controles de servidor que implementen patrones se nos facilitó enormemente.

La elección de los patrones que incluimos en nuestra librería intentó cubrir las distintas categorías de patrones RIA. Sus complejidades a la hora de implementarlos fueron de menor a mayor, debiendo excluir algunos patrones cuya complejidad excedía el alcance del presente trabajo. Por otra parte, hay otros patrones que creemos no se pueden embeber en un control de forma genérica, sino que solo pueden ser representados con un conjunto de controles relacionados.

Existen otras propuestas comerciales que apuntan a la realización de librerías o bibliotecas de controles Web. Incluso hay software libre, pero en estos casos no se centran en librerías sino en controles por separado. Aun así las librerías comerciales no están centradas en los patrones RIA sino en facilitar funcionalidades específicas (RIA o no). Por tanto no proveen explicaciones sobre cuando utilizar un patrón u otro. Así, muchos controles implementan parcialmente la solución de un patrón, y otros no implementan ninguna.

De hecho, en otras tecnologías también se está siguiendo por este camino. Algunas soluciones están más maduras que otras, pero en definitiva todas persiguen el mismo objetivo. A través

de toolkits bien organizados con alto nivel de abstracción la tarea del desarrollador puede ser simplificada para que no necesite conocer todo el “detrás de escena” de la implementación, sino que solo necesita saber cuándo y cómo utilizarlo.

Por último, el prototipo de sitio web Pandora nos permitió explicar los patrones a través del uso de los mismos. Se explican las funcionalidades de cada uno, cuándo se debe utilizarlo, cuáles son las mejores prácticas, todo esto mostrado con los mismos patrones que los controles implementan.

Los trabajos futuros que planteamos, además de la ampliación de este grupo de controles con la adición de otros importantes como lo es el drag and drop, se refieren a la utilización de los mismos no solo en la Web con dispositivos tradicionales como una computadora sino también en dispositivos móviles, cuya capacidad de presentación y procesamiento es muy distinta.

Por otra parte señalamos el acompañamiento de esta transición por la ingeniería de software, sería interesante trabajar aún más en las metodologías de desarrollo de este tipo de aplicaciones.

6.2. Trabajos Futuros

A continuación se presentan dos aristas que podrían ser estudiadas como trabajos posteriores al presente. La primera trata de metodologías específicas de patrones RIA, mientras que la segunda avanza sobre la utilización de estos patrones en aplicaciones móviles.

6.2.1. Metodologías de aplicación de patrones RIA.

En esta sección se pretende mostrar una posible vía de investigación, a partir de los conceptos preparados en esta tesis, y que trata las metodologías más adecuadas para el desarrollo de aplicaciones RIA, para que atiendan las diferencias en el modo de plantear las soluciones que este tipo de aplicaciones posee. Se mostrarán también una serie de modelos conceptuales que nos ayudarán a documentar nuestras soluciones.

Esto es actualmente una de las falencias al momento de llevar a cabo una aplicación con interfaces ricas, y para solucionarlas se desarrolló la metodología propuesta por [3] y [22] donde proponen la transformación de un modelo de negocios estándar a la combinación de especificaciones WebML y los modelos conceptuales RUX.

Se tratarán como primera medida los conceptos de diseño de las aplicaciones originales, que las aplicaciones RIA extienden. Serán éstos explicados y formularemos una serie de puntos a tener en cuenta en cada uno. Luego, se presentará un modelo de negocios de una aplicación RIA, que posteriormente se transformará en modelos WebML y RUX.

Finalmente mostraremos los distintos tipos de aproximación que podemos encontrar actualmente, sin embargo no seguiremos más allá en nuestras investigaciones sobre alguna metodología específica por tratarse de un planteo básico que puede potencialmente extenderse en una continuación del presente trabajo.

Conceptos específicos de RIA

Básicamente, hay cuatro conceptos que requieren nuestro análisis. Se trata del modelado de datos, lógica de negocios, comunicaciones y presentación.

- **Modelado de datos**
En las aplicaciones RIA, cierta información volátil y/o persistente puede ser almacenada en el cliente y luego enviada al servidor una vez que se completa la operación.
- **Lógica de negocios**
En RIA se tienen estructuras de navegación diferentes a las tradicionales, y tanto el cliente como el servidor pueden llevar a cabo operaciones complejas.
- **Presentación**
Las aplicaciones RIA mejoran la presentación e interacción con el usuario. Pueden trabajar como aplicaciones Single Page, evitando así recargas innecesarias y la carga progresiva de la presentación.
- **Comunicaciones**
RIA permite comunicaciones síncronas o asíncronas, dando lugar de esta manera a una gran variedad de formas de tratar cada acción o evento realizado en la aplicación. Este es un nuevo concepto que cruza todos los anteriores, como se muestra en la figura 4.1.

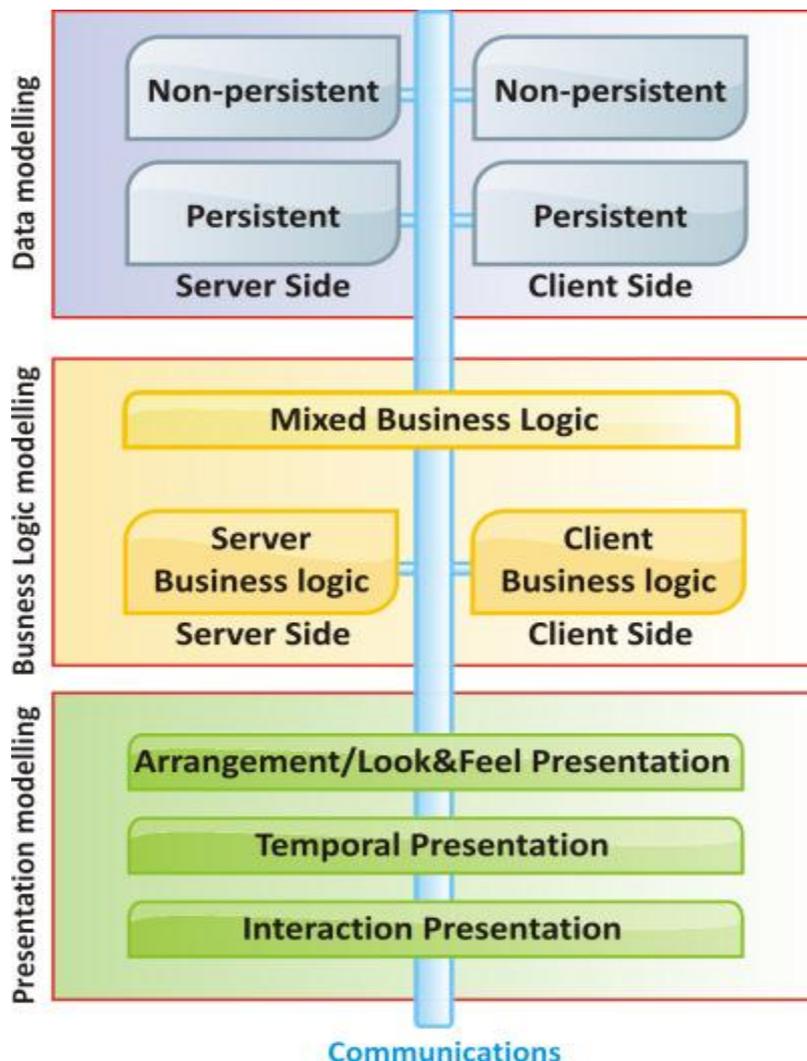


Imagen 1. Conceptos implicados en las aplicaciones RIA. Tomado de ref. [22].

Modelado de datos

Como se mencionó anteriormente, en las aplicaciones RIA se utiliza el cliente para guardar información tanto volátil como persistente. Esto marca una gran diferencia con las aplicaciones convencionales, donde toda la información se guarda en el servidor y el cliente solo se utiliza como entrada de los mismos.

Esta característica es utilizada para guardar información temporalmente, y luego de completado el proceso enviar todo al servidor para ser almacenado definitivamente. Un claro ejemplo de esto es el “carrito de compras” de cualquier e-commerce. La información referida a los productos que el cliente del comercio va adquiriendo se puede ir almacenando localmente, evitando de esta manera tráfico de red y carga al servidor, y ofreciendo al usuario una respuesta inmediata. Esta información es manipulada a medida que el cliente toma o quita productos de su carrito, para finalmente enviarse al servidor solo cuando el cliente decide realizar el pedido.

Pero este tratamiento de los datos en el cliente trae aparejado consigo algunos desafíos, como asegurar la integridad de los datos que allí se manejan, y proveer de mecanismos de seguridad a los mismos.

En aplicaciones RIA, teniendo en cuenta la distribución entre cliente y servidor, y la duración/persistencia de los datos, se encuentran cuatro divisiones:

- Datos no persistentes del lado del cliente. Los datos se almacenan temporalmente en el cliente y son desechados al salir de la aplicación.
- Datos persistentes del lado del cliente. Se mantienen incluso al finalizar la aplicación. Normalmente se utilizan cookies, pero éstas proveen de un almacenamiento muy limitado. Las aplicaciones RIA ofrecen a cambio nuevas formas de almacenamiento mucho más prácticas como archivos, bases de datos y objetos.
- Datos no persistentes del lado del servidor. Se almacenan en el servidor y se descartan cuando se concluye la sesión del usuario.
- Datos persistentes del lado del servidor. Se almacena la información en archivos o bases de datos del servidor, independientemente de la ejecución de la aplicación por parte del cliente.

A su vez, la distribución de los datos genera problemas de consistencia y replicación de éstos, para lo cual definiremos los siguientes tres niveles:

- A nivel de campo
- A nivel de tupla (contiene varios campos)
- A nivel de paquete (conjunto de tuplas)

En estos niveles, al haber un cambio en el cliente o en el servidor, se sincronizan ambos al nivel que corresponda. Cuando hablamos de consistencia a nivel de campo, cuando un campo particular cambia se produce en ese momento la sincronización entre cliente y servidor.

Lo mismo sucede a nivel de tupla, y a nivel de paquete se produce cuando se dispara el evento de sincronización luego de modificar la información.

Lógica de negocios

Tradicionalmente la lógica de negocios siempre se ubicó en el servidor, que frente a un requerimiento del cliente realizaba las tareas necesarias y retornaba un HTML para ser presentado en el cliente.

Las aplicaciones RIA, en cambio, pretenden aprovechar las capacidades que provee el cliente al máximo para maximizar de esta manera la experiencia del usuario. No es suficiente para este propósito que se aprovechen los recursos del cliente, sino que se debe cambiar también la comunicación entre éste y el servidor.

Básicamente podemos dividir la lógica de negocios de una determinada tarea de tres maneras diferentes. Se puede realizar completamente en el cliente, si es que éste tiene la capacidad necesaria para llevarla a cabo. Se puede realizar también en el servidor, como es habitual en las aplicaciones Web, o de otra manera se puede adoptar una forma mixta donde parte del trabajo del cliente y el resto en el servidor (por ejemplo, cuando los cambios en el cliente necesitan actualizar la base de datos del servidor).

Presentación

En las aplicaciones tradicionales no se dispone de soporte para multimedia (al menos en forma nativa) ni se proveen interfaces ricas. En las aplicaciones RIA se pretende mejorar este aspecto, permitiendo transmisión de audio y video, y mejorando la experiencia del usuario mediante un cambio en la interacción con el mismo permitiendo realizar drag and drop entre otras acciones sin necesitar de una gran cantidad de código JavaScript para llevarlo a cabo.

Este cambio en el aspecto y el comportamiento de la aplicación se encuentra muy ligado al dispositivo dónde se muestre. Por ejemplo, una aplicación que se muestra en una computadora de escritorio suele tener un rango determinado de pixeles en su pantalla (como podría ser 1024x768), y se dispone de un teclado y un mouse como dispositivos característicos. En cambio, una aplicación Web que se muestra en un celular o Smartphone tiene características completamente diferentes, como puede ser el tamaño de su pantalla, la ausencia de mouse o teclado completo, o la capacidad de tomar touches del usuario.

Cualquiera sea el dispositivo utilizado se deben utilizar sus ventajas al máximo posible, así como sortear las limitaciones que cada uno tiene. Una aplicación incluso pudiera adaptar su interface al dispositivo en el cuál se está mostrando.

Esto trae aparejado potenciales problemas de performance, y se pueden presentar incompatibilidades entre los distintos dispositivos y navegadores.

Se presentan básicamente tres aspectos:

- La disposición espacial de los elementos de la interface, junto con su look&feel.
- El comportamiento temporal que define las relaciones entre los elementos de la interface a través del tiempo (sin tomar en cuenta al usuario).
- El comportamiento en las interacciones con el usuario y la reacción correspondiente del sistema.

Comunicaciones

En cuanto a las comunicaciones entre cliente y servidor, en las aplicaciones tradicionales son originadas por el cliente y son sincrónicas. Las aplicaciones RIA permiten la comunicación asíncrona, originada en el cliente o en el servidor, y en una gran variedad de formas.

La manera en que una aplicación se comunica se relaciona con todos los aspectos anteriores. Se adoptarán distintos enfoques según la consistencia y granularidad de los datos. La comunicación en estos casos puede ser basada en mensajes (utilizando una cola de mensajes), replicando estados (basado en el registro de filas modificadas) o utilizando replicación de métodos (el método grabado en un lado se debe replicar también del otro). Comúnmente se serializan los datos para su comunicación, perdiendo así tiempo de procesamiento pero ganando ancho de banda en aplicaciones que transfieren datos intensivamente.

La lógica de negocios se realiza en el cliente y en el servidor por lo que para cada tarea se utilizará el tipo de comunicación que sea necesario, que además proveerá el enlace necesario entre ésta capa y la capa de presentación.

Modelado tradicional

Las aplicaciones RIA pueden ser diseñadas siguiendo modelos tradicionales, para luego refinarlos en otros que provean capacidades extra para documentar los diferentes aspectos a tratar en este tipo de soluciones.

Estos enfoques tradicionales se basan en escenarios y prototipos para representar las interfaces de manera de que toda la audiencia sea capaz de comprender las ideas que allí se detallan, ya sean usuarios, diseñadores, desarrolladores, etc. En el primer caso describen la funcionalidad de cada página, si ésta es simple no presenta problemas pero en el caso de aplicaciones complejas no son lo suficientemente completas para representar los cambios de estado en una página.

Por otro lado, los prototipos no son buenos como documentación, y presentan el mismo problema ante el modelado de cambios de estado dentro de una misma página.

Por este motivo se han desarrollado varios modelos alternativos, por lo general extensiones de modelos existentes, con nuevos elementos que se utilizan para representar los conceptos agregados por las aplicaciones RIA.

Metodología de diseño RIA

Luego de presentar los problemas y desafíos que surgen en cada capa de una aplicación, presentamos a continuación la solución propuesta por [22].

Las estrategias elegidas para manejar la información pueden ser modeladas mediante SyncML (Synchronization Mark-up Lenguaje), un lenguaje independiente de la tecnología.

Inicialmente, el modelo de datos puede ser realizado sin tener en cuenta los niveles de persistencia de los mismos. Para esto se utilizan los modelos conceptuales tradicionales de UML o diagramas de E/R. Luego en sucesivas refinaciones se pueden agregar la consistencia necesaria, el nivel de granularidad utilizado en cada caso y la seguridad que se debe aplicar a los datos sensibles (por ejemplo, almacenar datos importantes solo en el servidor).

En cuanto a la lógica de negocios se puede ver una tarea independientemente de quién sea el encargado de procesarla para luego refinar esta aproximación y determinar de qué lado y cómo se procesará. Esto permite realizar modelos con gran abstracción utilizando por ejemplo diagramas UML.

Esto está muy ligado a las decisiones que se tomen en la capa de datos, ya que si una tarea requiere de ciertos datos que se almacenan en el servidor, al menos una parte del proceso debe ejecutarse allí.

En cuanto a la presentación se puede plantear en dos pasos: primero se puede plantear el look&feel y ubicación de los elementos de la interface, y luego se pueden diseñar el comportamiento temporal de estos elementos así como su reacción frente a las acciones del usuario. Cada entidad puede tener varias reacciones, y éstas deben ser consistentes entre sí. También se pueden plantear diferentes reacciones para los mismos elementos, de manera de modelar las acciones que se realizan en los diferentes dispositivos.

La comunicación es regida por decisiones en otras áreas. Se debe tomar en cuenta como primera medida la cantidad de datos que se envía entre cliente y servidor. En general en las aplicaciones RIA se utilizan datos volátiles en el cliente y al sincronizar se utiliza mayormente la comunicación basada en mensajes, que tiene como ventaja su tolerancia a fallos, rápida respuesta a eventos asíncronos y minimización del ancho de banda utilizado.

En el caso de aplicaciones mixtas, donde el servidor realiza parte del procesamiento, se deben agrupar tareas y utilizar transacciones del lado del servidor para minimizar el ancho de banda usado y procesar los datos de manera que el sistema quede en un estado consistente.

Metodologías existentes

Actualmente existen cuatro tipos de aproximaciones diferentes en cuanto a la modelización de aplicaciones RIA.

- Extensión de algún método existente
- Combinación entre metodologías propias de aplicaciones Web y de diseño de interfaces
- Elaborar una nueva metodología
- Aproximación basada en patrones

El primer método refiere a la extensión de una metodología, agregándole a ésta nuevas características para modelar aspectos propios de aplicaciones RIA. Modelos, entidades, acciones, transformaciones adicionales son necesarias en este caso. Ejemplos de esto son OOHRIA, extensiones de OOHDM, UWE-R y extensiones de WebML.

El segundo utiliza métodos tradicionales combinados, representando la interfaz con alguna metodología específica que permita exponer las características de la aplicación. Un ejemplo de esto es WebML combinado con RUX.

En la tercera opción se analiza la creación de una nueva metodología desde sus bases, orientada específicamente a este tipo de aplicaciones. Como ejemplo encontramos ADRIA.

Por último, la aproximación basada en patrones como recomendaciones de diseño basadas en las mejores prácticas. Como ejemplos encontramos las librerías propuestas por Scott y los patrones basados en UML propuestos por UWE.

En el caso del presente trabajo no hemos optado por seguir ninguna de estas metodologías por considerar que nuestro trabajo no pretende profundizar en éste área, ya que no es este el objetivo principal ni se encuentra dentro del alcance del mismo.

Por otro lado, nuestro objetivo es realizar una librería de controles, y explicar y utilizar los mismos en un prototipo de sitio Web, esto quiere decir que no estamos trabajando en un proyecto completo en el cual sea aceptable la sobrecarga de esfuerzo debido al uso de metodologías especialmente dedicadas a este tipo de aplicaciones.

Para profundizar en este tema se pueden repasar las ref. [13] y [23], además de las mencionadas [3] y [22].

6.2.2. RIA en aplicaciones móviles

Otra línea a investigar, que claramente está ganando terreno, es el desarrollo de aplicaciones Web con interfaces ricas en una gran variedad de dispositivos, con diferentes características.

El incremento en el uso de dispositivos móviles es notable en este último tiempo, bastará con decir que existen cuatro veces más dispositivos móviles que computadoras en el mundo, y que esta cifra tiende a ir en aumento.

Dentro de este segmento de dispositivos móviles podemos reconocer un ecosistema variado, dentro de los cuales encontramos a los teléfonos móviles (celulares), PDA's (Palm o Pocket PC), Smartphones, Tablets, etc. Dentro de este conjunto de dispositivos, el que mayor proyección de crecimiento tiene en el mercado actual es el Smartphone.

La definición de lo que es un dispositivo Smartphone (o teléfono inteligente en español) es la de un teléfono móvil común (celular) pero con características superiores, similares a las de una PC personal. Dentro de estas características superiores que lo diferencian con un teléfono móvil común se destacan:

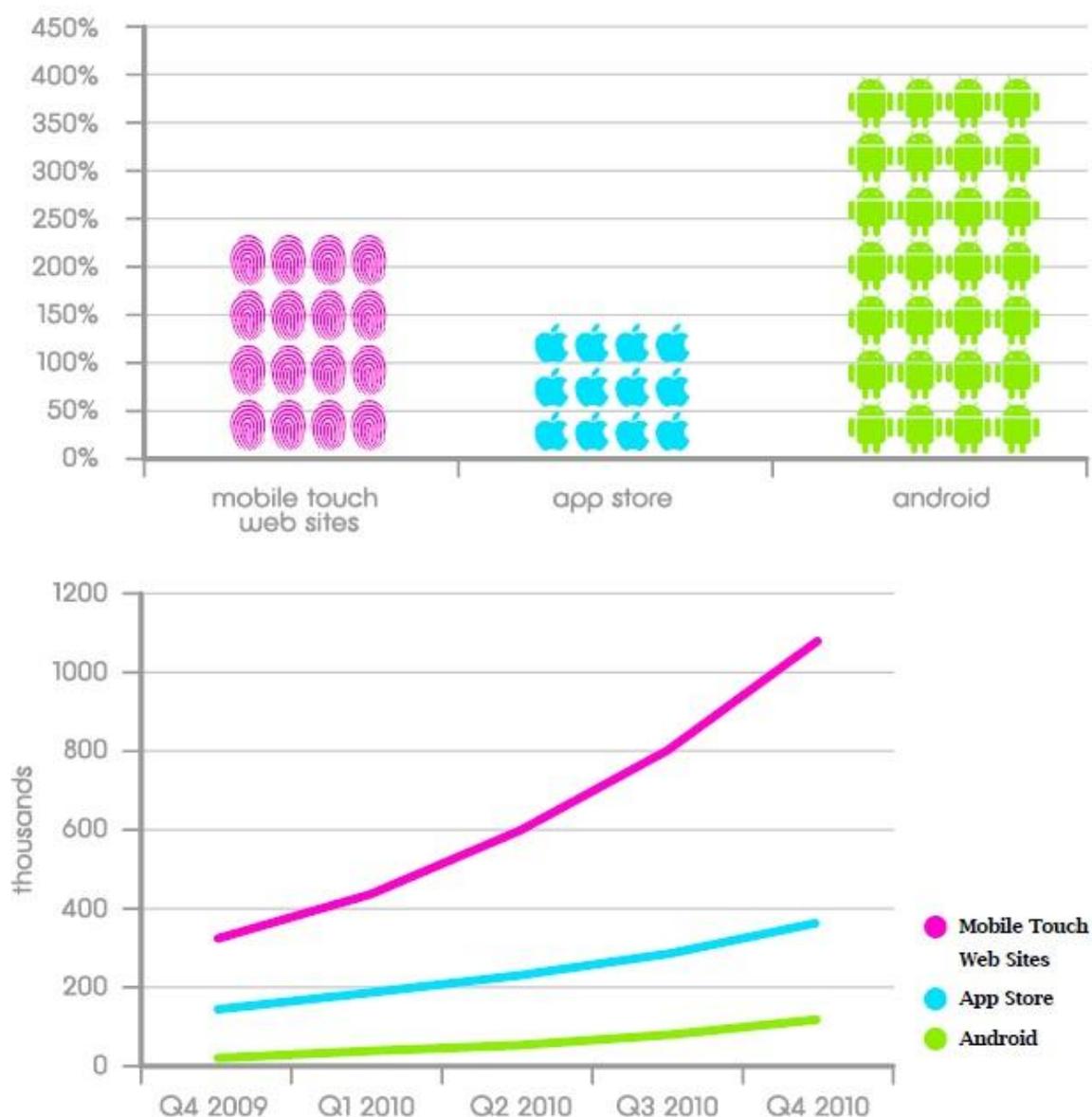
- Permite la instalación de programas de terceros.
- Utiliza cualquier interfaz para el ingreso de datos, como por ejemplo teclado QWERTY, pantalla táctil.
- Te permiten el acceso a Internet.
- Soporta correo electrónico.
- Cuenta con GPS.

Gracias a estas características superiores que poseen los Smartphone, se abre un nuevo abanico de opciones, y crece así la demanda de aplicaciones más poderosas para este tipo de segmento.

Ahora y con este panorama se genera un nuevo interrogante referido a la mayor demanda de estas aplicaciones. Es conveniente el desarrollo de una aplicación nativa para móvil, o un servicio basado en el navegador disponible en distintas plataformas?

De escoger lo primero, ¿en qué plataforma se concentrarse? Entre las opciones se encuentran iPhone, Android, RIM, Palm, Windows Mobile y Symbian.

En Abril de 2010, la empresa de búsquedas en móvil Taptu publicó un detallado informe que mostraba que el futuro de la Web móvil probablemente estará dominado por sitios Web basados en navegador disponibles para varias plataformas, más que por aplicaciones creadas específicamente para iPhone, Android o cualquier otra plataforma. La empresa estimaba que había 440.100 sitios Web para móviles táctiles en todo el mundo en aquel momento, lo cual representaba un 35% de crecimiento con meses anteriores, y un incremento anual del 232%. En comparación con las 185.000 aplicaciones para iPhone de la App Store, con un incremento anual del 144% y las 35.947 aplicaciones del Android Market, con un incremento del 403% anual. Lo que es más, Taptu espera que el mercado de Internet móvil basado en el navegador crezca mucho más rápido que el mercado de las aplicaciones y pronostica 1 millón de sitios web para móviles en el año 2011.



<http://www.taptu.com/metrics/TaptuMobileTouchWebReportApr2010.pdf>

Imagen 2. Gráfico de crecimiento de sitios Webs para diferentes tecnologías.

Este crecimiento de los sitios webs para móviles y las características destacadas de los Smartphone nos dan la posibilidad de crear aplicaciones mucho más ricas y poderosas. Esta situación es ideal para aplicar los patrones comunes a RIA, previamente descritos, en estas aplicaciones, y crear interfaces avanzadas y elaboradas para dispositivos móviles, permitiéndonos enriquecer la experiencia del usuario, creando atractivas e interactivas interfaces de usuario.

Encontramos actualmente dos proyectos que nos facilitan la utilización de un framework para el desarrollo de sitios web para dispositivos móviles con interfaces avanzadas, permitiendo una rica experiencia al usuario, en otras palabras estos framework nos permiten desarrollar aplicaciones RIA para dispositivos móviles.

Los dos framework, **JQuery Mobile Framework** y **JQTouch**, están basados en JQuery y además tienen la particularidad de ser soportados por los distintos browser de los dispositivos y las distintas plataformas sobre las que se ejecutan.

Glosario

Abstracción: Técnica de generalización mediante la cual se ocultan o ignoran detalles para capturar algo en común entre diferentes instancias.

Ajax: acrónimo de Asynchronous JavaScript and XML, es una técnica de desarrollo Web mediante la cual se evita la recarga de la página para en su lugar realizar una llamada al servidor y actualizar solo la zona necesaria.

Applet: es un componente de una aplicación Web que se ejecuta en el contexto de otro programa, por ejemplo un navegador Web. No puede ejecutarse en forma independiente. Un Java Applet es un programa que se embebe en código HTML.

Encapsulamiento: relacionado al concepto de ocultación. El objeto se aísla del exterior previniendo modificaciones sin autorización o los llamados efectos secundarios.

RIA: aplicaciones ricas de internet.

Framework: estructura conceptual y tecnológica de soporte que permite que diferentes proyectos sean organizados y desarrollados. Suele incluir artefactos, programas, bibliotecas.

jQuery: biblioteca o framework de licencia libre de JavaScript.

Look and feel: estilo general de un sitio Web.

Modularidad: característica mediante la cual un sistema se divide en partes llamadas módulos que permiten simplificar el diseño de una solución.

Swing: evolución del AWT, permite crear componentes de interfaz de usuarios.

Web Part: control de servidor de .NET que puede ser manipulado por los usuarios finales.

Widget: bloque de construcción básico de la interfaz gráfica de usuario.

Referencias

- [1] Adobe. *Rich Internet application development*, <http://www.adobe.com/devnet/ria/>
- [2] BOOCH Grady. *Análisis y diseño orientado a objetos con aplicaciones*, Wilmington (Delaware, EEUU): Addison-Wesley Iberoamericana, 2da edición, 1996, pp. 638.
- [3] BRAMBILLA Marco, PRECIADO Juan Carlos, LINAJE Marino, SANCHEZ-FIGUEROA Fernando, *Business Process-Based Conceptual Design of Rich Internet Applications*, Yorktown Heights (New York ,EEUU): ICWE '08. Octava Conferencia Internacional sobre Ingeniería Web, 2008, pp. 7.
- [4] BUSCH Marianne, KOCH Nora. *Rich Internet Applications – State-of-the-Art*, Alemania: Ludwig-Maximilians-Universität München, Diciembre de 2009, pp. 18.
- [5] CALDERON Adam, RUMERMAN Joel. *Advanced ASP.NET AJAX Server Controls For .NET Framework 3.5*, Indianapolis (Indiana, EEUU): Addison Wesley, 1ra Edición, 2008, pp. 645.
- [6] CAMERON Rob, MICHALK Dale. *Pro ASP.NET 3.5 Server Controls and AJAX Components*, Berkeley (California, EEUU): Apress, 2008, pp. 740.
- [7] CAMPANA Oscar. *Especificación de Interfaces y Patrones RIA*, La Plata (Argentina), Facultad de Informática (UNLP), Marzo de 2009, pp. 135.
- [8] CARBALLO Nono E. *Arquitectura de proyectos para Aplicaciones Ricas de Internet desarrolladas con Adobe Flex 3 y Java*, Santiago de Cuba (Cuba), Fecha, pp. 21.
- [9] CECIL Richard F. *Documenting the Design of Rich Internet Applications: A Visual Language for State*, UX Matters, <http://www.uxmatters.com/mt/archives/2007/12/documenting-the-design-of-rich-internet-applications-a-visual-language-for-state.php>, Diciembre de 2007.
- [10] DZone. *Javalobby*, <http://www.javalobby.org/>
- [11] FOWLER Martin. *Patterns*, IEEE Software, IEEE Computer Society, Mar/Apr 2003, pp. 56-57.
- [12] GARRETT Jesse James. *Ajax: A New Approach to Web Applications*, <http://www.adaptivepath.com/ideas/ajax-new-approach-web-applications>, Febrero de 2005.
- [13] GARRIGÓS Irene, MELIÁ Santiago, CASTELEYN Sven. *Adapting the Presentation Layer in Rich Internet Applications*, Alicante (España), ICWE '9 Proceedings of the 9th International Conference on Web Engineering, 2009, pp. 292-299
- [14] KALBACH James. *Designing Web Navigation*, Sebastopol (California, EEUU): O'Reilly Media, Inc., 1ra Edición, Agosto de 2007, pp.416.
- [15] KALBACH James. *Designing Web Navigation Optimizing the User Experience*, Sebastopol (California, EEUU): O'Reilly Media, Inc, 1ra Edición, 2007, pp. 412.
- [16] Keynote, *White Paper: Rich Internet Applications: Design, Measurement, and Management Challenges*, Keynote Systems Inc., 2006, pp. 15.
- [17] LOTT Joey, PATTERSON Dann. *Advanced ActionScript 3 with Design Patterns*, EEUU: Adobe Press, 2007, pp.286.
- [18] LUTTINGER Avigdor, *White Paper: Los desafíos del desarrollo de Aplicaciones Ricas para Internet (RIA) y cómo superarlos*, Magic Software, 2009, pp. 9.

- [19] PALACIOS Eliseo, *Tesis: Aplicaciones ricas en Internet (RIA). Un enfoque de refactorización*, La Plata (Argentina): Universidad Nacional de La Plata, 2008, pp. 84.
- [20] PARIHAR Mridula, AHMED Essam, CHANDLER Jim, HATFIELD Bill, LASSAN Rick, MACINTYRE Peter, WANTA Dave. *ASP.NET Bible*, New York (EEUU): Hungry Minds Inc., 2002, pp. 855.
- [21] PILGRIM Mark. *Dive into HTML 5*, <http://diveintohtml5.org/>
- [22] PRECIADO Juan Carlos, LINAJE Marino, SANCHEZ-FIGUEROA Fernando. *Designing Rich Internet Applications with Web Engineering Methodologies*, Como (Italia), ICWE '07. Septima Conferencia Internacional de Ingeniería Web, 2007, pp 8.
- [23] ROSSI Gustavo, URBIETA Matias, GINZBURG Jeronimo, DISTANTE Damiano, GARRIDO Alejandra. *Refactoring to Rich Internet Applications. A Model-Driven Approach*, Como (Italia), ICWE '07. Septima Conferencia Internacional de Ingeniería Web, 2007, pp 12.
- [24] SCOTT Bill, NEIL Theresa. *Designing Web Interfaces*, Sebastopol (California, EEUU): O'Reilly Media, Inc, 1ra Edición, 2009, pp. 336.
- [25] SCOTT Bill. RIA Patterns Best Practices for Common Patterns of Rich Interaction - Presentation Transcript, <http://www.slideshare.net/interactionpatterns.org/ria-patterns-best-practices-for-common-patterns-of-rich-interaction-presentation>
- [26] STALEY Tad. *Planning for RIA success*, Adobe Consulting, 2007.
- [27] Technology for solutions. *Archive for the 'RIA: Flash, Silverlight, HTML5' category*, <http://blog.tfsla.com/category/ria/>
- [28] TIDWELL Jenifer. *Designing Interfaces*, Sebastopol (California, EEUU): O'Reilly Media, Inc, 1ra Edición, 2005, pp. 352.
- [29] TROELSEN Andrew. *Pro C# 2008 and the .NET 3.5 Platform*, Berkeley (California, EEUU): Apress, 4ta Edición, 2007, pp. 1400.
- [30] TROELSEN Andrew. *Pro C# 2010 and the .NET 4 Platform*, Berkeley (California, EEUU): Apress, 5ta Edición, 2010, pp. 1753.
- [31] UI Patterns. *User Interface Design Patterns*, <http://ui-patterns.com/>
- [32] VOGEL Peter. *Professional Web Parts and Custom Controls with ASP.NET 2.0*, Indianapolis (Indiana, EEUU): Wiley Publishing, Inc., 2006, pp. 480.
- [33] W3 Schools. <http://www.w3schools.com>
- [34] Wikipedia. <http://es.wikipedia.org/>
- [35] WINN Tiffany, CALDER Paul. *Is this a pattern?*, IEEE Software, IEEE Computer Society, Jan/Feb 2002, pp. 59-65.
- [36] WROBLEWSKI Luke. *LukeW*, <http://www.lukew.com/>
- [37] Yahoo Developer Network. *Yahoo Patterns RIA - Best Practices for Common Patterns of Rich Interaction*, <http://developer.yahoo.com/ypatterns/about/start.html>.
- [38] YAKOV Fain, RASPUTNIS Victor, TARTAKOVSKY Anatole. *Rich Internet Applications With Adobe Flex and Java*, EEUU: SYS-CON Media, 1ra Edición, 2007, pp 697.