

Reutilización, Parametrización y Patrones de Diseño

*Todo ello bajo un enfoque práctico de Software
basado en Componentes y Tecnología Web*

José Carlos Calvo Tudela
<http://www.inteligencia.com>
ingeniería inteligente

Índice

Introducción

El objetivo que persigue este documento es el de aportar una visión práctica y real al Diseño de Software basado en componentes y bajo tecnología Web.

Así se verán las vertientes más actuales que rigen la comunicación e intercambio de información vía Web, las necesidades y requerimientos que hay que satisfacer en el ámbito profesional para conseguir aplicaciones que se puedan usar y sean usadas durante un periodo de tiempo lo más amplio posible adaptándose a los cambios que se producen en el entorno.

Como siguiente paso se comentará el mundo del diseño de aplicaciones Web que han de soportar la lógica de estas necesidades, y han de conseguir adaptarse e integrarse con los cambios que se producen en las comunicaciones Web, de forma que podamos conseguir aplicaciones reutilizable y configurables, consiguiendo así una gran flexibilidad para amoldarse a estos cambios.

Todo esto se enfocará bajo el diseño de software de calidad, viendo aspectos tales como la Reutilización, la Parametrización y el uso de Patrones de Diseño aplicables a estas aplicaciones.

Situación Actual

Actualmente estamos en una situación en la que la Información es el centro de todo lo que esté relacionado con la tecnología, por lo tanto, para hacer factible esa idea de que la Información pueda estar en todas partes sin ningún tipo de barrera ni límite, debemos hacer aplicaciones que sean capaces de comunicarse con cualquier otra aplicación, sea cual sea su origen o cometido, así como de comunicarse de la forma más afable posible al usuario final, ya que la información no sirve de nada si no hay un usuario que quiera consumirla, y para que la consuma, ésta ha de estar presentada lo mejor posible al usuario, de forma que le resulte cómodo y fácil el acceso a ella.

Así que para satisfacer las necesidades de los usuarios, la Información debe llegar a ellos de forma transparente. Entonces, nosotros, como diseñadores de software, debemos conseguir aplicaciones que sean capaces de relacionarse con cualquier fuente de información, aparentando de cara al usuario que la Información es homogénea, es decir, que el usuario no debe apreciar si estamos consiguiendo la información a través de una Base de Datos local, remota o a través de otra aplicación que nos de servicio en la otra parte del mundo. Y no es solo esto, sino que si el día de mañana aparece otra nueva fuente de información, nuestra aplicación ha de ser lo suficientemente flexible como para poder acogerla lo mejor posible.

Como vemos no vale sólo con conseguir una aplicación que recupere información de ciertos sitios, la idea es más general, por lo que debemos conseguir flexibilidad, escalabilidad y configurabilidad.

Por otra parte, no tenemos solo el problema de la recopilación de la Información, sino que también cabe la posibilidad de que nuestra aplicación pueda ser una fuente para otras aplicaciones.

Así que concluyendo estas ideas y tras mostrar brevemente el alcance al que se desea llegar en el tratamiento de la información, cabe resaltar que para conseguir esto, no hay otra forma que la estandarización, debemos diseñar aplicaciones que se comuniquen de forma estándar, para así conseguir la máxima adaptabilidad al mundo que le rodea.

Para entrar más a detalle en estas necesidades me centraré en la tecnología Web, ya que es la tecnología que más tiene que ver con el flujo y el tratamiento de la Información, dada su arquitectura y alcance mundial.

Actualmente en el mundo Web, nuestras aplicaciones son visitadas por usuarios, de forma directa, también pueden ser accedidas por otras aplicaciones, lo que se denomina un Servicio Web, que veremos más adelante. Y por otra parte, nuestras aplicaciones recopilan información tanto de Bases de Datos internas como de otras fuentes de Información, es decir, otras aplicaciones que nos dan servicio.

Un Servicio Web, es un punto de acceso que podemos ofrecer a nuestra aplicación. Este punto de acceso se utiliza para ofrecer información a otras aplicaciones automáticas que lo utilizan. Este Servicio se define en base a unos estándares de comunicación, de forma que ambas aplicaciones sepan comunicarse sin necesidad de

adaptarlas específicamente. El estándar de comunicación que siguen es el XML, las peticiones son en formato XML y las respuestas también. De esta forma un aplicación no necesita conocer el funcionamiento interno de otra aplicación de la quiere obtener información, ni siquiera necesita ponerse de acuerdo con sus diseñadores para poder conseguir esa colaboración, simplemente conociendo el estándar XML, se pueden comunicar fácilmente, y así podríamos de cambiar de fuente de información, si ningún problema, sin más que configurar las peticiones y las respuestas para adaptarnos a otra fuente, pero el sistema de comunicación es el mismo. De esta forma se evita esa barrera de comunicación que existe en caso de establecer una comunicación específica con un sistema, que una vez que está hecha, es difícil cambiar de proveedor o cambiar esa comunicación, ya que no es sólo tu aplicación, sino, también todas aquellas que dependen de ella.

Diseño de Aplicaciones Web

Entonces pasaremos a ver en detalle cómo diseñar software que sea capaz de adaptarse a estas necesidades y requerimientos.

Las claves para el éxito de nuestra aplicación serán la Reutilización, la Parametrización y el Diseño Basado en Componentes, y cómo los Patrones de Diseño nos pueden ayudar a conseguirlo de la mejor forma posible.

Diseño Basado en Componentes

El Diseño Basado en Componentes consiste en diseñar de forma separada cada parte del Sistema que tiene una lógica propia, y hacer esta parte de forma que sea independiente del resto, es decir, hay que identificar aquellas zonas que por sí solas tienen sentido, y no necesitan de nada más para su funcionamiento. Gracias a este tipo de diseño podremos reutilizar componentes en otros Sistemas que los necesiten, reemplazarlos por otros que hacen la misma función,... La idea es conseguir piezas de software que puedan llevarse a otro sitio o que puedan cambiarse y todo siga igual.

Por ejemplo, en rasgos generales, y según el esquema de necesidades visto anteriormente, una aplicación Web, podría dividirse en un componente de recuperación de información, que de forma independiente se encarga de ir recopilando información de las fuentes que hay en la Web, después tendríamos un componente de traspaso de información, que se encargará de hacer accesible la comunicación que tenemos almacenada o que conseguimos a través del componente de recuperación de información, y también podríamos tener un componente de procesamiento interno, que se encarga de transformar la información recopilada por el componente de recopilación, para hacerla accesible al componente que se encarga de transmitir esta información.

Reutilización

Este concepto se basa en diseñar componentes que abarquen el mayor número de supuestos, es decir, que sean lo suficientemente abstractos como para poder ser usados desde más de un sistema.

Por ejemplo una forma de reutilización es crear un componente que se encargue por ejemplo de interactuar con nuestra Base de Datos, y permita cualquier tipo de interacción con ella. Así cualquier Sistema que diseñemos podría usar este componente para acceder a la Base de Datos.

Parametrización

La Parametrización es otra forma de Reutilización. Se basa en hacer componentes que hacen una misión específica, por ejemplo acceder a la Base de Datos de Oracle y no sería capaz de acceder a SQL Server, pero si le cambiamos la configuración podría acceder a SQL Server y no a Oracle. Así si hacemos componentes parametrizables podemos adaptarlos dependiendo de las necesidades a un supuesto o a

otro. Así podemos hacer un componente, que instanciándolo con distintas configuraciones satisfagan un conjunto de necesidades que son parecidas, pero no iguales.

Como puede verse, el conseguir componentes que sean parametrizables, pueden conseguir diversas funcionalidades y adaptaciones de forma sencilla. Así si un componente además de reutilizable es parametrizable, llegamos al punto extremo de reutilización y capacidad de evolución.

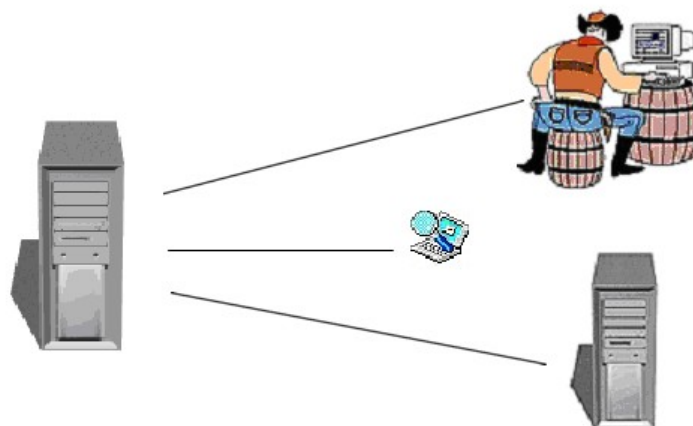
Además de esta potencialidad, nos permiten otras cosas, en el caso que nos aborda, la tecnología Web, sería deseable que diseñemos un Sistema que gracias a su parametrización podamos también crear una herramienta de configuración del sistema, y así conseguimos que los clientes de nuestros productos puedan configurar y adaptar el Sistema a las necesidades de cada momento, y mientras nosotros nos dedicamos a diseñar otros Sistemas, en lugar de estar configurando constantemente los nuevos cambios y demandas de la actualidad. También cabe destacar que esta característica es muy deseable por los clientes de Software.

Patrones de Diseño

Los Patrones de Diseño nos ayudan a conseguir unos diseños optimizados con poco esfuerzo, ya que son diseños que están pensados y testados, así nosotros sólo tenemos que adaptarlos a nuestra situación. Estos patrones nos aportan ideas que serán reutilizables y adaptables a un amplio campo de problemas.

Esquema general

En una aplicación Web es muy importante diferenciar de forma clara la lógica de negocio, de la presentación. Ya que en toda aplicación Web podemos encontrarnos con que las formas de presentación para un mismo producto son diversas y no es conveniente replicar la lógica de nuestra aplicación en cada una de las formas posibles de presentación, consiguiendo así un producto más centralizado y de más fácil mantenimiento.



Caso práctico de Aplicación

Para mostrar todo lo expuesto hasta ahora veremos un caso de aplicación en el que se diseña una aplicación Web, basada en componentes con elementos reutilizables y parametrizables, y usando patrones de diseño.

El caso sería el siguiente

Se necesita construir una aplicación Web que se encargará de vender cierto tipo información. La idea es que actualmente se puede conseguir el teléfono fijo de una persona, siempre y cuando ésta sea el titular del contrato, en cambio, nadie ofrece un servicio completo tal que dado una persona, busque el teléfono de la casa donde vive. Se tiene gran expectativa de mercado en este ámbito.

La aplicación deberá conseguir esta información de varios sitios:

- Una página Web, ofrece de forma gratuita una parte de la información que se necesita. La información que ofrecen son archivos PDF, que tienen un listado de personas y sus teléfonos, pero sólo las personas que tienen contrato, es decir, lo que hay habitualmente, estos archivos tienden a sufrir modificaciones a lo largo del tiempo debido a las bajas y altas que se producen, así que habrá que descargarlos y analizarlos constantemente.
- La otra parte de la información se conseguirá comprando unas Bases de Datos a una empresa que se dedica a vender información sobre integrantes de las familias del territorio nacional. Es decir, nos dan unas Bases de Datos con una relación de personas que viven en el mismo hogar. Esto no es gratuito. También sufren modificaciones periódicas debido a las nuevas familias que se forman, así como los jóvenes que se independizan de sus casas.
- Por último, en caso que tengamos una petición para la que no tengamos datos a ofrecer, se pedirán instantáneamente a una empresa que nos ofrece un Servicio Web con toda la información que necesitamos, aunque puede no tenerla. Este debería ser el último sitio de donde solicitar la información, ya que cada petición que se les hace nos cuesta más dinero del que nosotros pedimos por ella, debido a que la legalidad con la que esa empresa consigue los datos es dudosa, pero en caso de problema legal, ellos son los responsables, no nosotros. Así que esta fuente se usará en última instancia, para dar una imagen a nuestros clientes de fortaleza y sientan que somos capaces de conseguir la información que se busca, así nos abriremos camino en el mercado. Aunque esperamos que con el tiempo, cada vez se use menos, ya que tendremos más información en nuestras Bases de Datos y no los necesitaremos.
- Además, y para no usar demasiado la tercera opción, se intentarán buscar nuevas vías de información, pero por ahora son las que se han enumerado.

Por otra parte, la información que vendemos se podrá presentar de varias formas:

- En formatos HTML, PDF y RTF para los usuarios que accedan por nuestro portal Web.
- En caso que en un futuro decidamos vender esta información a otras empresas habrá que idear un sistema de comunicación para pasarles la información de forma On-Line.

Además se desea que el sistema devuelva la información en uno de los tres formatos descritos dependiendo del cliente que acceda, para así conseguir un servicio más personalizado y de mayor calidad, ya que conseguimos dar al cliente la información según sus gustos.

También se intentará en un futuro añadir información de móviles, fax, u otros modos de contacto posibles como correo electrónico, postal... Para poco a poco ofrecer un servicio más completo. Pero aún se está estudiando y no será a corto plazo.

Discusión

Esta sería la descripción del sistema a elaborar, como puede verse abarca un conjunto bastante amplio de requisitos de toda índole y naturaleza, y además no queda perfectamente limitado, ya que quedan bastantes puntos sin concretar debido a que son cosas que se están estudiando o posibles ampliaciones.

Entonces hay que crear un sistema que sea suficientemente genérico, consistente y que sea fácil de mantener, como para poder cubrir las necesidades y poder aumentarlo sin muchos problemas.

Utilizando un Diseño Basado en Componentes, para organizar los entes independientes, pueden detectarse los siguientes componentes:

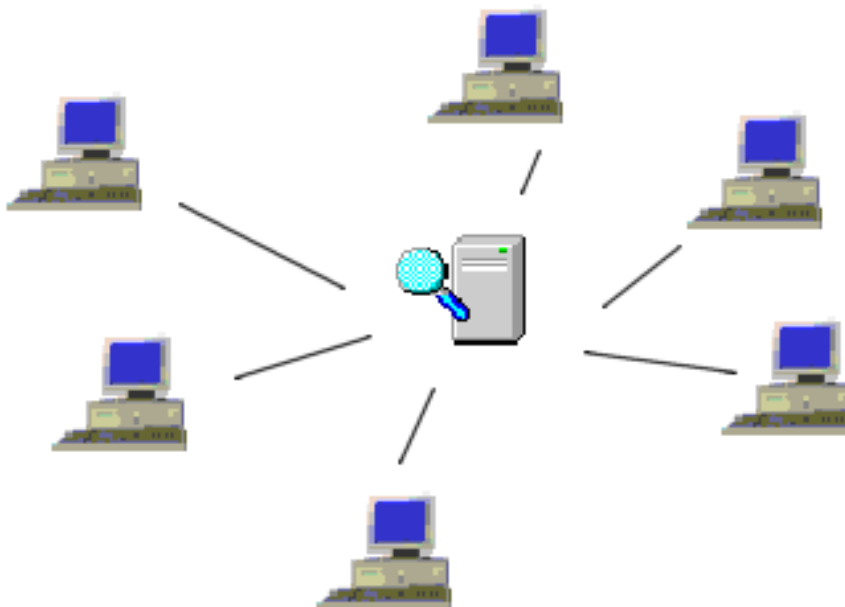
- Componente que se encargará de descargarse y analizar los ficheros PDF de la Web gratuita.
- Componente que se encargará de analizar y enlazar con la información de la Web gratuita, las Bases de Datos que compramos con las familias.
- Componente que se encargará de comunicarse con la empresa que nos dará servicio cuando no tengamos datos de una consulta.
- Componente para crear una salida en PDF.
- Componente para crear una salida en RTF.
- Componente para crear una salida en HTML.
- Componente que le dé el ciclo de vida a una petición.
- ...

En principio, serían estos, aunque poco a poco intentaremos diseñar componentes más abstractos y reutilizables que estos.

Arquitectura

Para dar soporte a estas necesidades se utilizará una Arquitectura de Repositorio, ya que es deseable que todos los datos estén en un mismo lugar tanto para los componentes de recopilación de información, como para los de extracción de la misma. Ya podemos ver como se empiezan a usar Patrones de Diseño, sin tener que pensarlo mucho, podemos observar leyendo la literatura que esta Arquitectura enlaza muy bien con nuestras necesidades. Además ganamos en flexibilidad, ya que de esta forma los componentes de salida son independientes a los componentes de entrada. De esta forma, en caso de tener una nueva fuente, como bien nos especifican, simplemente tendremos que construir un componente que interaccione con el Repositorio y los demás componentes de entrada así como los de salida, no se verán afectados.

ARQUITECTURA DE REPOSITORIO



Componentes

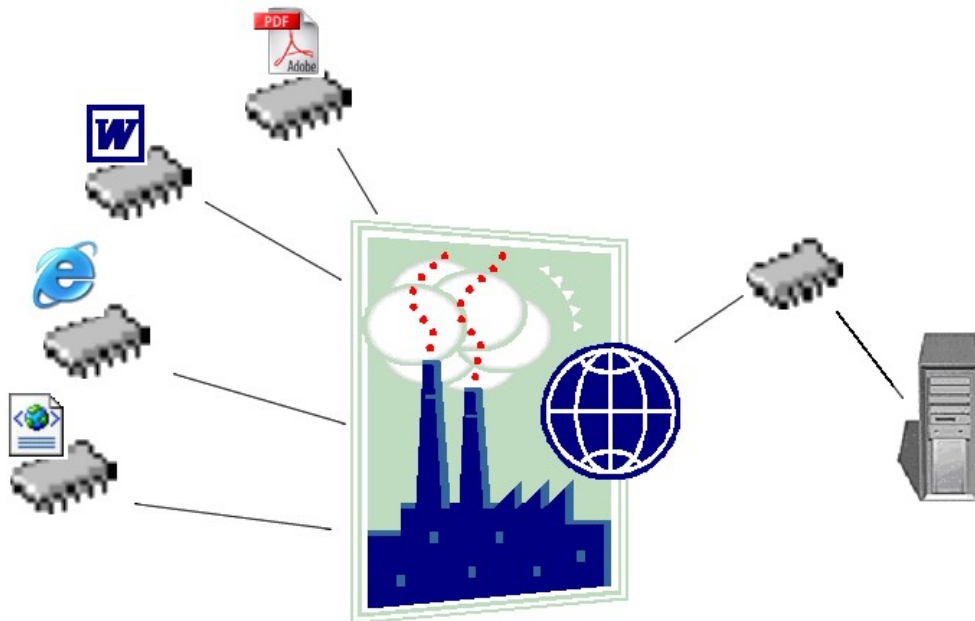
Nos centraremos primero en los componentes de recopilación de información. En general, este tipo de elementos en un sistema Web, son los menos reutilizables y parametrizables, debido a que debemos acotarnos a lo que nos ofrecen las fuentes, si todas publicasen bajo ciertas reglas o parecidos, podría intentar unificar para conseguir componentes reutilizables, pero en general no es así y prácticamente debemos construir un componente para cada fuente, y más aún cuando no se rigen por estándares como pueden ser los Servicios Web, sino que nos dan la información de diferentes formas. Así que lo único que puede hacerse para conseguir la mayor cantidad de código reutilizable es crear pequeños componentes que sean comunes a todas las entradas de información, como pueden ser, componentes de acceso a Base de Datos, componentes de representación de la información y ya que se hacen reconocedores de ficheros PDF,

pues conseguir otro componente capaz de leer PDFs tal que puedan usarlo futuras aplicaciones para el manejo de los mismos.

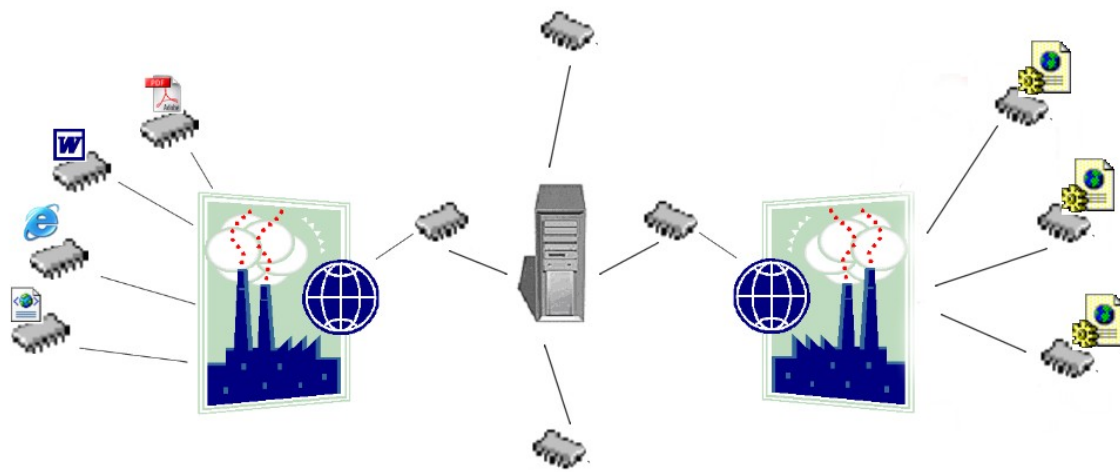
Con respecto al ciclo de vida y a los componentes de salida, sí podemos crear componentes tan reutilizables y parametrizables como queramos ya que esa es nuestra zona de diseño y no tenemos restricciones externas, siempre y cuando, tendamos a la estandarización, porque si no, no podremos transmitir la información como se desea, además la estandarización hace más receptivas a otras empresas que quieran que les demos servicio, ya que podrán construir componentes de entrada reutilizables y estarán agradecidos de ello.

Entremos en detalle. Con respecto a las salidas HTML, PDF, RTF u otras, está claro que solo intervienen a nivel de presentación y no tienen porque encargarse de obtener los datos de nuestra Base de Datos. Así que utilizaremos para ellos un Patrón de Diseño que nos dará bastante fuerza en la reutilización y parametrización de cara a las personalizaciones que se desean por usuario. Este patrón será el de Factoría Abstracta. La Factoría Abstracta se encarga de generar en base a unos parámetros de entrada, un objeto, todos los objetos que devuelve una Factoría Abstracta han de implementar una misma interfaz. De esta forma podríamos conseguir una Factoría tal que le damos los datos del cliente, y la Factoría en base a ellos nos dan el procesador de presentación que necesitamos, así estos procesadores pueden tener un método que pida los datos a mostrar y les aplique la presentación que implementa.

Así esta Factoría admitirá los datos del cliente y nos devolverá un objeto, puede ser para construir salidas en HTML, PDF, RTF u otros dependiendo del objeto que devuelva, pero una vez que lo ha devuelto, no nos interesa saber qué es lo que hace, suponemos que es el correcto, ya que de eso se encarga la Factoría, pero lo que sí sabemos es que le damos los datos a mostrar y él se encarga de darles el formato deseado, así nosotros le damos los datos y lo que nos devuelva lo mostramos al cliente.



Con respecto al componente que implementa el ciclo de vida, gracias a la Factoría le hemos quitado la lógica de presentación con lo cual se vuelve bastante reutilizable, ya que se puede usar en cualquier tipo de presentación. Pero podría mejorarse aún más. De cara a posibles nuevos servicios, es decir, otros datos a mostrar, podríamos volver a usar la Factoría Abstracta para que nos proporcione en base a unos parámetros de entrada de la petición, crear un objeto que se encarga de recuperar la información de la Base de Datos. De esta forma tendríamos un componente principal que simplemente coge los parámetros de entrada de la petición se los pasa a una Factoría “A” y obtiene un objeto que recupera la información, después con los datos del cliente se los da a la Factoría “B” y obtiene el procesador de presentación, y por último le da los datos al procesador de presentación obtiene el informe final a dar al cliente.



Con estas Arquitecturas de Diseño conseguimos que por ejemplo el añadir un nuevo tipo de presentación no implique ningún cambio en ninguna parte del software, simplemente se crea el procesador y se añade a la Factoría de procesadores.

Lo mismo ocurre con nuevos servicios que se quieran vender, simplemente se crearía el componente que genera los datos a mostrar y se añadiría a su Factoría correspondiente.

Otro factor a tener en cuenta es que cada pieza de software, por pequeña que sea, si tiene sentido por sí sola, es decir, tiene sus funciones bien delimitadas, y además puede ser usada en más de un sitio, deberá ser un componente. En nuestro Sistema tendremos que ocuparnos del control de acceso, facturación,... y no es conveniente que cada aplicación Web que hagamos para la empresa en cuestión se encargue de estas lógicas que son tan independientes al producto que se vende. Así que lo más conveniente es tener un componente de facturación y otro de control de acceso, y así con cualquier pieza que desprenda lógica común.

Tecnología de Comunicación

De cara a que posiblemente se quiera vender la información a otras empresas, y se hará por los mecanismos estándares para ellos, es decir, con XML, entonces todos los componentes que se encargan de generar la información a de nuestras Bases de Datos devolverán la información en formato XML, y los componentes que generan las presentaciones, recibirán el XML y lo transformarán. Esto no es difícil, gracias a que XML es un estándar existen herramientas para transformar XML a cualquier formato, así para transformar a HTML se usarán ficheros XSL, para PDF se usarán XSL-FO y para RTF se pueden usar XSL-FOP, por último para dar servicio a otras empresas no hará falta que se le aplique nada, ya que como se genera en XML, se le puede pasar directamente.

Parametrización

Y pasando ahora al tema de la parametrización de nuestro sistema, comentaré varios puntos que pueden parametrizarse para potenciar el sistema.

El primer punto y más importante es el de conseguir que la Factoría de procesadores de presentación sea configurable por personas no informáticas, como pueden ser las personas que trabajen en atención al cliente. Entonces debemos sacar la lógica de asociación de un cliente a un procesador fuera del código fuente. La idea sería poner estas configuraciones en Base de Datos, y añadir una pieza software que se encargará de alimentar esa configuración.

Otro punto es el de la forma de cobro, el componente de facturación no debe implementar de forma interna las tarifas de un determinado servicio, sino que deberá ser configurable, para adaptarnos fácilmente a los cambios de precio e incluso a los descuentos que se hagan a ciertos clientes. Esta característica es deseable que no la tengan que configurar personas del equipo de desarrollo cada vez que hay un cambio, sino que debe ser configurado por el equipo de facturación de la empresa.

Y por ejemplo otro punto a configurar puede ser el de la Factoría de servicios, ya que puede ser que haya unos determinados servicios para unos clientes y otros para otros clientes, con lo que sería interesante que hubiese un sistema de reglas o algo similar con lo que el equipo de marketing controle los productos que vende a cada cliente.

Donde queremos llegar es a sacar toda la lógica posible a zonas de configuración con lo que conseguir una aplicación Web lo más flexible y adaptable posible. Así conseguiremos un producto de calidad y que vivirá durante bastante tiempo.

Conclusiones

Con esto se quiere conseguir una idea, que aún está muy lejos, pero debemos tender a ella. La idea es la de “Las Pastillas de Software”, que acertadamente comenta Juan Carlos Granja en un artículo.

Esto consiste en que a nivel de nuestro sistema hemos conseguido esto de las pastillas de software, ya que tenemos piezas de software que funcionan por sí solas y podemos poner o quitar estas piezas a nuestro gusto, consiguiendo así funcionalidades diversas, y una de las piezas no influye en ninguna otra, con lo que conseguimos montar un sistema a partir de “Pastillas de Software” que son compatibles con él.

Pero esto es a nivel de nuestro sistema como ya he comentado, la utopía sería conseguir esto a nivel general, es decir, que se construya un componente software que tiene cierta funcionalidad y sin ningún problema se pudiese integrar en cualquier sistema del mundo. De esta forma podríamos construirnos un sistema a medida sin más que integrar varias pastillas, podríamos aumentar una característica añadiendo otro componente, y todo funcionaría perfectamente con un comportamiento de sistema global.

Esto lo tenemos ya en otros campos como el hardware, en el que se tiene un PC y si queremos ver la televisión en él, simplemente le pinchamos una tarjeta sintonizadora y perfectamente se integra en el sistema hardware y funciona, además cualquier fabricante puede construir cualquier pieza y en principio todas son capaces de integrarse.

Esta sería la idea a conseguir en la industria del software, pero mientras, lo único que podemos hacer es conseguir esta idea a nivel local y ganar esa potencia interna y consiguiendo software lo más reutilizable posible y basado en componentes.

Bibliografía

- Monografía: Ingeniería del Software (Juan Carlos Granja Álvarez)
<http://www.ati.es/novatica/1995/nov-dic/nv118.html>
- The Web Engineering Community Portal
<http://www.webengineering.org/>
- Component Based Software Engineering
<http://www.cs.ucl.ac.uk/staff/W.Emmerich/lectures/3C05-03-04/CBD.pdf>