La Navegación y la Separación de Conceptos

A. M. Reina, J. Torres, M. J. Escalona, J. A. Ortega
Departamento de Lenguajes y Sistemas Informáticos
Universidad de Sevilla

Avda. Reina Mercedes S/N
41012 Sevilla
Fax: 95 455 71 39. Tlf: 95 455 68 77
{ reinaqu, jtorres, escalona, ortega}@lsi.us.es

Resumen

La popularidad de la multimedia y la aparición de Internet ha ocasionado que cada vez haya más aplicaciones que se ejecutan en el contexto de la red de redes. Esta proliferación ha planteado la necesidad de revisar las metodologías de diseño tradicionales para recoger aspectos nuevos que surgen en este entorno. Uno de estos nuevos conceptos es la navegación. Por otra parte, e intentando resolver algunos problemas que surgieron en la programación orientada a objetos, se ha abierto un nuevo campo de investigación conocido como orientación a aspectos o separación de conceptos. Este artículo trata de identificar la navegación como un nuevo concepto a separar, en el sentido de la orientación a aspectos. Como una primera forma de separar estos conceptos proponemos el uso de XML y la especificación XLink.

Palabras clave

Hipermedia, navegación, separación avanzada de conceptos, programación orientada a aspectos, XML, XLink.

1. Introducción

El uso cada vez más generalizado de Internet y de las tecnologías web, ha creado la necesidad de revisar las metodologías de diseño tradicionales y adaptarlas para que puedan recoger nuevos conceptos característicos de las aplicaciones web que no se trataban bien en las metodologías tradicionales. Han surgido así nuevas propuestas metodológicas como HDM [7], RMM[9], OOHDM[14] o la metodología de diseño de N. Koch[8].

Todas estas metodologías tienen en común el hecho de que consideran la navegación como una característica clave dentro de la etapa de diseño, y por ello, le dedican una fase completa al diseño de la estructura de navegación de la aplicación web.

Atendiendo a las metodologías de diseño web se puede afirmar, por tanto, que la navegación es un concepto lo suficientemente importante en una aplicación web como para prestarle por completo la atención durante una fase entera.

Por otra parte, uno de los principios básicos de la ingeniería del software es la máxima "divide y vencerás". Dándole una nueva vuelta de tuerca a esta máxima ha surgido con fuerza en los últimos años la separación avanzada de conceptos y la programación orientada a aspectos. Esta nueva comunidad de investigadores defiende la idea de que un sistema se programa mejor si se especifican por separado cada uno de los conceptos que lo componen y se dan las relaciones entre ellos, dejando a los mecanismos proporcionados por la programación orientada a aspectos la tarea de componerlos.

En este trabajo, concretamos la propuesta realizada en [12], proponiendo para obtener una primera separación de la estática de la navegación el uso de la especificación XLink propuesta por el consorcio W3C [6] acompañada, por supuesto del lenguaje de marcado XML[3].

En la sección 2 se da una visión general de las ideas propuestas en el campo de la separación avanzada de conceptos, y algunas de las tendencias más generales. En el tercer apartado se da una definición del concepto de navegación tal y como lo entendemos en el ámbito de este artículo, además se da una idea del tratamiento que recibe la navegación por parte de las metodologías de diseño. El cuarto apartado presenta el problema utilizando un ejemplo. Luego se propone una forma de separar la navegación.

El último apartado se deja para obtener conclusiones y apuntar las futuras líneas de trabajo.

2. Separación Avanzada de Conceptos

En los últimos años, ha surgido con fuerza una campo de investigación conocido como separación avanzada de conceptos. La investigación en este campo se basa en la idea de que es más fácil programa un sistema, si se piensa sobre cada uno de sus conceptos de forma separada. Por lo tanto, se considera la modularización como concepto clave para el desarrollo de un sistema complejo, siendo ésta una forma de conseguir la deseada separación de conceptos. El objetivo fundamental que se propone es la especificación independiente de los aspectos del sistema y su posterior integración. Esta integración se dejará en manos de los mecanismos de composición que proponen las distintas tendencias del campo de la separación avanzada de conceptos. La Figura 1 muestra de forma esquemática cuál es la base del campo de la separación avanzada de conceptos.

Existen varias líneas de investigación que están intentando encontrar nuevas dimensiones para separar estos conceptos más allá de las ideas tradicionales de módulo y clase. Algunas de las más importantes son: los métodos adaptivos [11], los hiperespacios [15], los filtros de composición [1], y la programación orientada a aspectos[10].

Últimamente, a estas tendencias se las conoce con el nombre genérico de Desarrollo de Software Orientado a Aspectos o por sus siglas inglesas AOSD (*Aspect-Oriented Software Development).* Además de las extensiones a los lenguajes de programación, se están estudiando también las metodologías de análisis y de diseño tradicionales y cómo se reflejan los aspectos en dichas etapas dentro del desarrollo de un proyecto software.

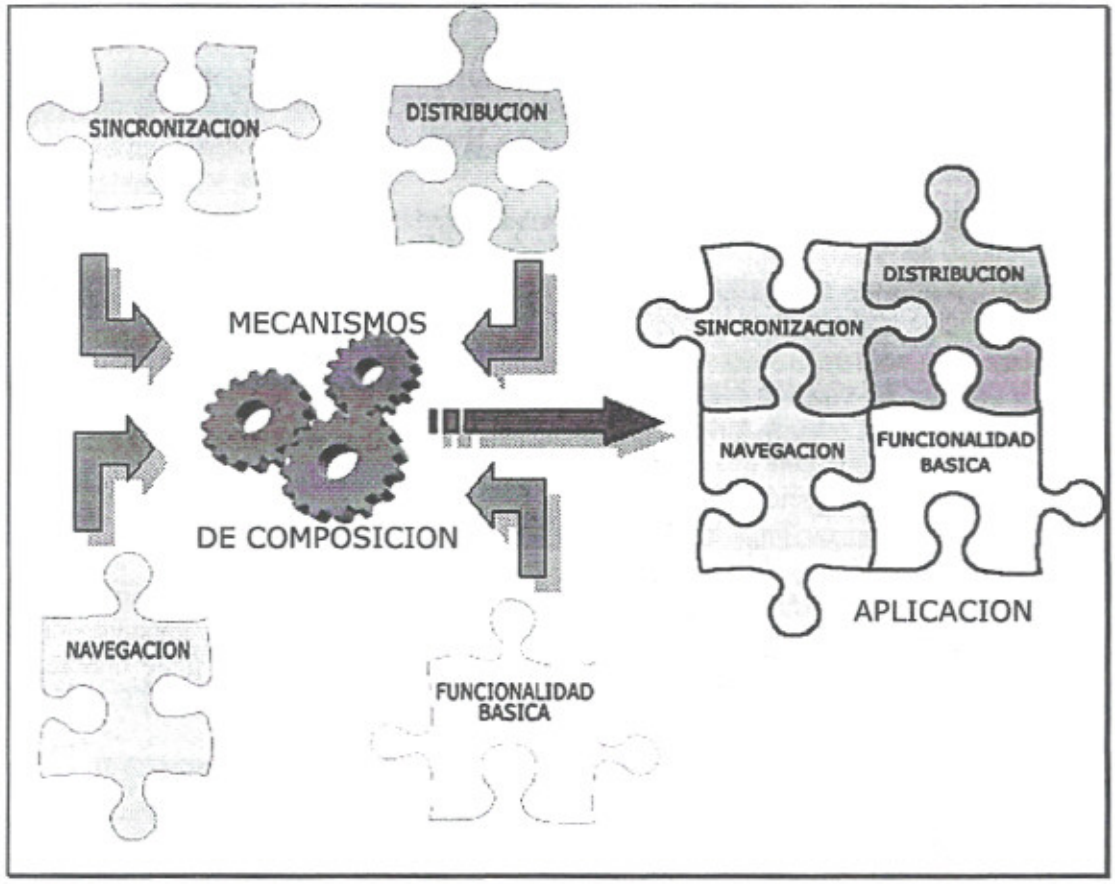

Figura 1. Objetivos de la separación avanzada de conceptos.

Las principales diferencias que hay entre los distintos planteamientos son:

- Los mecanismos de composición que utilizan.
- La importancia o no de la funcionalidad básica con respecto a los otros aspectos o conceptos.

Para hacer tratar con estos nuevos conceptos, han surgido dos palabras que hacen referencia a esta nueva modularización y a la forma de relacionar estos conceptos: aspectos o conceptos que se cortan (*crosscutting concerns*) y puntos de corte (*pointcuts*).

G. Kiczales definió un aspecto como *una unidad modular que se disemina por la estructura de otras unidades funcionales.*

De manera más informal podemos decir que los aspectos pueden verse como aquellas unidades que se diseminan por todo el código, y que son difíciles de describir locamente con respecto a otros componentes. No suelen ser unidades de descomposición funcional del sistema, sino propiedades que afectan al rendimiento del mismo o a la semántica de los componentes, tales como la sincronización de procesos concurrentes, el manejo de errores, o la navegación.

Los puntos de corte (*pointcuts)* sirven para definir cómo se componen las diferentes piezas de código fuente, y son mecanismos que le permiten al programador referirse a puntos léxicos particulares en el código y/o a eventos.

3. La navegación

Aunque la navegación es un concepto que se utiliza ampliamente en el entorno de las aplicaciones web, pero creemos que necesita ser definido, ya que la frontera entre navegación e interfaz puede resultar difusa. Por lo tanto, a continuación definiremos qué entendemos por navegación en el ámbito de este artículo.

La idea de navegación históricamente ha estado ligada a la noción proveniente del mundo del hipertexto de saltar de página en página a través de un hiperenlace. En este artículo trabajamos con una idea un poco más amplia de navegación, y adoptamos el concepto de navegación definido por Schwabe y Rossi en OOHDM, es decir, asociamos la navegación al hecho de moverse a través de un espacio de información.

Esta concepción de la navegación implica que no consideramos como parte del proceso de navegación todos los enlaces de las páginas web. Por ejemplo, cuando utilizamos un buscador como *google* o *altavista*, y realizamos una consulta, tenemos como resultado una página web, que en la mayoría de los casos, tiene una serie de enlaces justo en la parte inferior de la página (Figura 2).

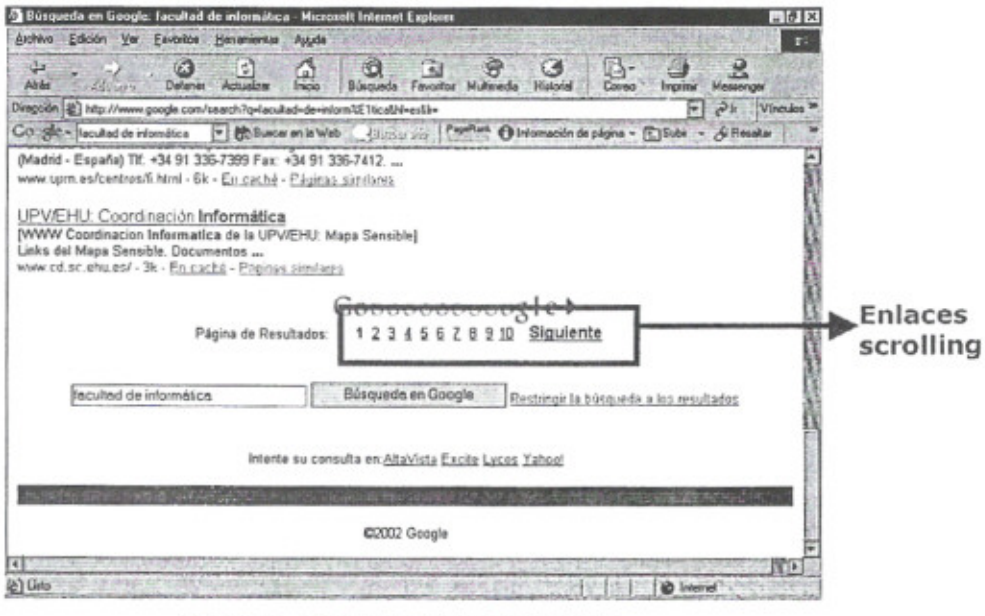

Figura 2. Enlaces del buscador *google*.

Estos enlaces no son considerados como parte del proceso de navegación, sino que son una mera forma de hacer un desplazamiento por el resultado de la consulta (*scrolling*).

3.1. La navegación en las metodologías

Una de metodologías pioneras en introducir la navegación como un aspecto a tratar dentro de la etapa de diseño de una aplicación Web fue HDM [7]. Esta metodología introdujo una serie de primitivas, algunas de las cuales, han servido como base a otras que surgieron con posterioridad.

OOHDM (Object-Oriented Hypermedia Design Model) es quizás una de las metodologías de diseño para aplicaciones en entorno Web que más se ha consolidado. En esta metodología la navegación se considera un paso crítico en el diseño de una aplicación hipermedia [14]. En la fase de diseño de navegación se construye un modelo que es una vista del modelo conceptual de clases, con lo que se permite la construcción de diferentes modelos de navegación para un mismo modelo conceptual.

La metodología enunciada por Koch [8] también parte del modelo de clases, y obtiene un modelo para la navegación como una vista del modelo de clases, pero esta propuesta, a diferencia de OOHDM, se basa en UML.

Estas metodologías tienen en común algunas ideas importantes, entre las que hay que destacar la utilización una serie de primitivas para definir la navegación: nodos (son vistas de las clases conceptuales), enlaces (son vistas de las relaciones del esquema conceptual) y estructuras de acceso (son modos alternativos para navegar, y pueden ser índices, visitas guiadas, visitas guiadas indexadas).

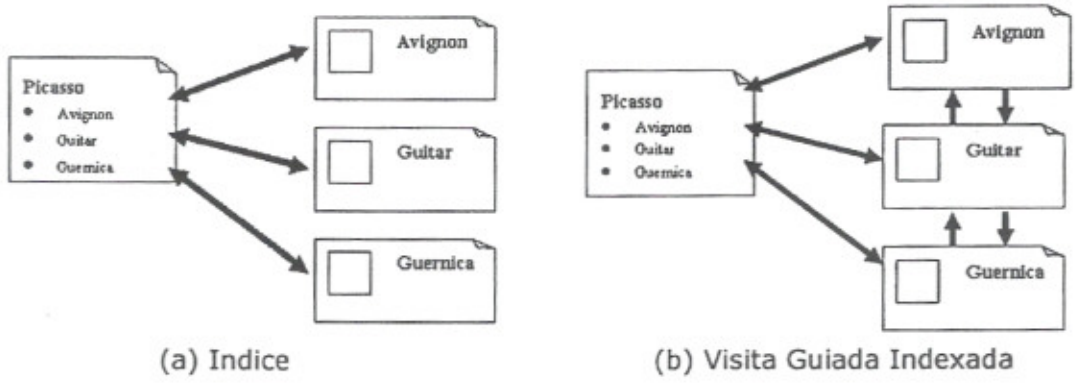

Figura 3. Ejemplos de estructuras de acceso.

En la Figura 3 se muestra un esquema de la diferencia existente entre dos estructuras de acceso. Se puede comprobar cómo el elegir una estructura de acceso u otra en el diseño de la navegación implica la creación de una serie de enlaces para definir la navegación entre los distintos nodos, entre las distintas informaciones.

4. Planteamiento del problema a través de un ejemplo

Para explicar mejor la problemática, se ha decidido tomar un caso real como ejemplo para entender mejor cuáles son los puntos conflictivos.

Supongamos que se ha diseñado una aplicación web para un museo, y que uno de los requisitos iniciales de navegación era que se tenía que poder navegar desde un pintor hasta cualquier pintura de ese pintor. Para conseguir este requisito, se decidió implementar una estructura de acceso índice, implementando una navegación como la que aparece en la Figura 3 (a). Lo que nos lleva a tener un modelo de clases de navegación y un modelo de estructura de navegación como los que aparecen en la Figura 4 .

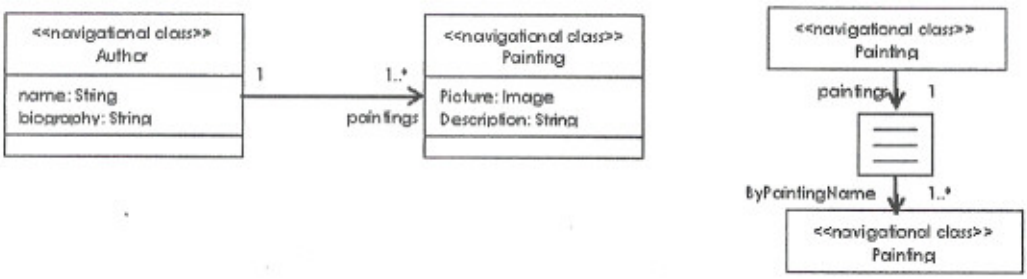

Figura 4. Modelo de clases de navegación y modelo de estructura de navegación mediante un Índice.

Este diseño nos puede dar lugar a una página web como la que aparece en la Figura 5, en la que se puede apreciar una página web bastante simple que implementa un nodo Pintura, y su código HTML.

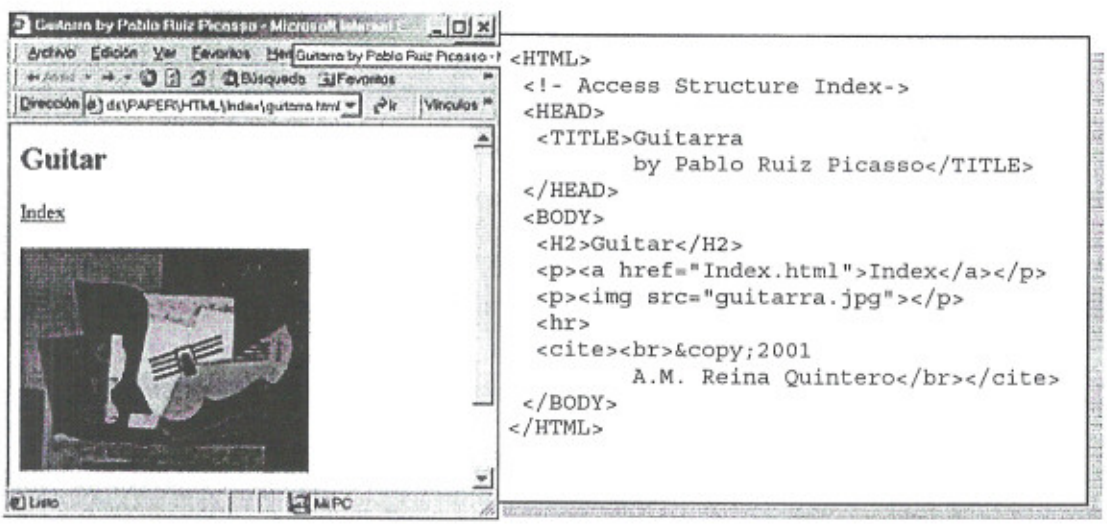

Figura 5. Página HTML implementando una estructura de acceso Índice.

Supongamos que le enseñamos esta página al cliente, y el cliente cree que con esta navegación no es suficiente, sino que se quiere mover por todas las pinturas de un mismo autor. Esto implica que hay que cambiar el diseño, y de lo que se representaba en el modelo de la Figura 4 se ha de pasar al modelo de la Figura 6.

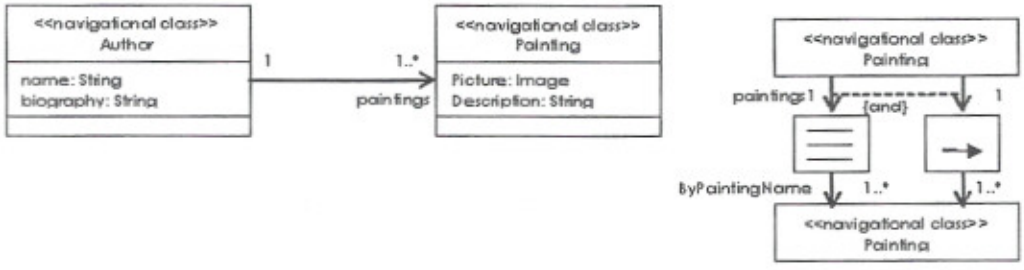

Figura 6. Modelo de clases de navegación y modelo de estructura de navegación para una Visita Guiada Indexada.

Un cambio tan conceptualmente simple como el hecho de cambiar una estructura de acceso por otra, implica que hay que recorrerse todos los nodos Pintura e irle añadiendo un enlace o dos más (uno al siguiente elemento y otro al anterior), para obtener algo como lo que aparece en la Figura 3 (b).

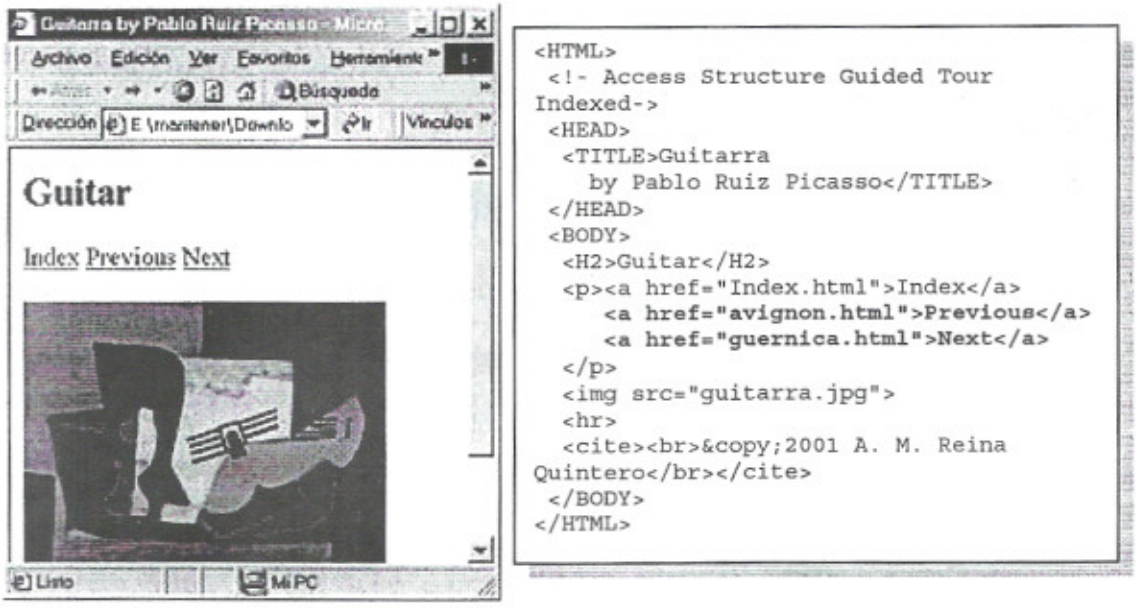

Figura 7. Página web que representa a un nodo Pintura y su código HTML.

En la Figura 7 se muestra la implementación de un nodo Pintura con la estructura de acceso Visita Guiada Indexada. Se ha resaltado en negrita las nuevas líneas de código HTML introducidas para poder tener esta estructura de acceso.

5. La navegación como un aspecto

El ejemplo del apartado anterior demuestra que, por una parte, podemos considerar la navegación como un aspecto, en el sentido de la separación avanzada de conceptos, ya que es un concepto que corta a la funcionalidad básica, y que por lo tanto, se debe separar, para poder razonar sobre él más fácilmente.

La separación a nivel de diseño parece que ya está resuelta por las metodologías, ya que se centran en la navegación, le dedican una etapa completa a su diseño, tratándola de forma separada.

La separación a nivel de implementación es lo que habría que conseguir. Esta separación a nivel de implementación tiene dos vertientes claramente diferenciadas:

- Una parte estática, es decir, a nivel de datos.
- Una parte dinámica, a nivel de proceso.

Para conseguir la separación de la parte estática, es decir, de la estructura de enlaces, se propone el uso de XML[3], XLink [6] y XPointer[5], como se indica en [13].

El uso del Lenguaje Extensible de Marcado, junto con la tecnología de hojas de estilo, nos permite tener una separación entre datos y presentación. Ahora, con la incorporación de XLink, podremos separar datos, presentación, y estructura de enlaces.

```
<paintings xmlns:link="http://www.w3.org/1999/xlink"
           xlink:type="extended">
    <painter xlink:type="locator"
             xlink:label="painter"
             xlink:href="picasso.xml"/>
    <painting xlink:type="locator"
              xlink:label="painting"
              xlink:href="guernica.xml"/>
    <painting xlink:type="locator"
              xlink:label="painting"
              xlink:href="avignon.xml"/>
    <painting xlink:type="locator"
              xlink:label="painting"
              xlink:href="guitarra.xml"/>
    <link xlink:type="arc"
          xlink:from="painter"
          xlink:to="painting">
</paintings>
```

Figura 8. Archivo XLink con una estructura de enlaces.

En la Figura 8 se muestra una estructura de enlaces para el ejemplo presentado en la sección anterior, expresada mediante el lenguaje de especificación XLink, de forma que tendríamos completamente separados, por un lado los enlaces, por otro lado los datos, y por otro la presentación, utilizando las hojas de estilo (CSS [2] y XSL[4]).

6. Conclusiones y trabajo futuro

Se ha caracterizado la navegación como un aspecto, en el sentido de la terminología utilizada en la separación avanzada de conceptos. Se ha separado el aspecto estático de la navegación, lo cual implica separar la estructura de enlaces. Para ello se ha utilizado el Lenguaje eXtensible de Marcado, y la especificación XLink.

Se ha mencionado el aspecto dinámico de la navegación, pero no se ha mencionado cómo separar esta parte dinámica. Esto se deja como trabajo futuro.

Se quiere ver también si algunas de las técnicas de separación y composición que se utilizan en el campo de la separación avanzada de conceptos serían válidas para obtener esta separación de la parte dinámica. Para ello habría que despejar claramente cuáles son los conceptos que se cruzan y los puntos de corte.

7. Bibliografía

1. Bergmans, L., Aksits, M.. *Composing Crosscutting Concerns Using Composition Filters.* Communications of the ACM. Vol. 44, n. 10, October, 2001.
2. Bos, B., Wiun Lie, H., Lilley, C., Jacobs, I. Cascading Style Sheets, level 2. CSS2 Specification. W3C Recommendation. May, 1998 [Online: *http://www.w3.org/TR/REC-CSS2*].
3. Bray, T., Paoli, J., Sperberg-McQueen, C.M., Maler, E., *Extensible Markup Language (XML) 1.0. (Second Edition)*. W3C Recommendation. October, 2000. [Online: *http://www.w3.org/TR/REC-xml*].
4. Deach, S. *Extensible Stylesheet Language (XSL) Specification.* Apr, 1999. [Online: *http://www.w3.org/TR/WD-xsl*].
5. DeRose, S., Maler, E., Daniel, R.. *XML Pointer Language (XPointer) 1.0.* W3C Recommendation. September, 2001.[Online: *http://www.w3.org/TR/xptr/*]
6. DeRose, S., Maler, E., Orchad, D., *XML Linking Language (XLink) 1.0.* W3C Recommendation. June, 2001. [Online: *http://www.w3.org/xlink/*].
7. Garzotto, F., Schwabe, D., Paolini, P. *HDM – A Model Based Approach to Hypermedia Application Design,* ACM Transactions on Information Systems, Jan., 1993.
8. Hennicker, R., Koch, N.. *Systematic Design of Web Applications with UML.*In Unified Modelling Language: System Analysis, Design and Development Issues (2001). K. Siau and T. Halpin (Eds.). Idea Group Publishing, 1-20.
9. Isakowittz, T., Stohr, E. A., Balasubramanian, P.. *RMM: A Methodology for Structured Hypermedia Design.* Communications of the ACM. Vol. 44, n. 10, October, 2001.
10. Kiczales, G., Hilsdale, E., Huguin, J., Kersten, M., Palm, J., Griswold, W.G.. *An Overview of AspectJ.* In Proceedings of the European Conference on Object-Oriented Programming. Springer-Verlag, 2001.

11. Lieberherr, J., Orleans, D., Ovlinger, J.. *Aspect-Oriented Programming with Adaptive Methods.* Communications of the ACM. Vol. 44, n. 10, October, 2001.
12. Reina, A. M., Torres, J. *Analysing the Navigational Aspect.* In *Workshop on Aspect-Oriented Software Development,* Bonn, Alemania, 2002.
13. Reina, A.M., Torres, J. *Separating the Navigational Aspect.* Acepted in *Workshop on Aspect-Oriented Programming for Distributed Computing Systems.*
14. Schwabe, D., Rossi, G.. *An Object Oriented Approach to Web-Based Applications Design.* TAPOS – Theory and Practice of Object Systems, vol. 4, 1998.
15. Tarr, P., Ossher, H., Harrison, W., Sutton, S. M., *N degrees of separation: Multi-dimensional separation of concerns.* Proceedings of the 21st International Conference on Software Engineering, May., 1999.