Departamento de Lenguajes y Sistemas Informáticos
Facultad de Informática y Estadística
Universidad de Sevilla

Avenida de la Reina Mercedes S/n. 41012 SEVILLA
Fax/Tlf: 954557139

Propuesta de metodología para el desarrollo de sistemas para el tratamiento de bibliotecas digitales

Versión 1

María José Escalona Cuaresma

Manuel Mejías Risoto

Jesús Torres Valderrama

Departamento de Lenguajes y Sistemas Informáticos

Universidad de Sevilla

Sevilla, Julio de 2000

HOJA DE CONTROL DE MODIFICACIONES

Número	Fecha	Descripción	Autores
0	6-2000	Desarrollo Inicial	M.J.Escalona M.Mejías J.Torres

ÍNDICE

1 Objetivos del documento....1

2 Introducción....2

3 OOHDM como referencia de la propuesta....5

3.1 Resumen de OOHDM....5

3.2 Aplicación de OOHDM a las bibliotecas digitales....10

4 El Proceso Unificado como referencia de la propuesta....11

4.1 Resumen del Proceso Unificado....11

4.2 Aplicación de UML a las bibliotecas digitales....14

5 Aproximación metodológica de desarrollo para los sistemas de tratamiento de bibliotecas digitales....15

5.1 Ciclo de Vida....15

5.2 Especificación de requisitos....16

5.3 Análisis....26

5.4 Diseño....28

5.5 Implementación....41

5.6 Pruebas....41

6 Conclusiones....42

7 Puntos abiertos....43

8 Bibliografía....44

1 Objetivos del documento

El desarrollo de un sistema destinado al tratamiento de bibliotecas digitales es una tarea complicada en la que suelen mezclarse complejos requisitos de almacenamiento debido a la diversidad de formatos de información (texto, imagen, vídeo, audio, etc.) que ha de gestionar, con diversas necesidades funcionales, que van desde las propias necesidades de funcionamiento de la aplicación hasta la implementación de los requisitos de navegación y de interfaz de usuario.

El desarrollo de una aplicación que engloba tal cantidad de requisitos debe plantearse como un proceso de ingeniería. Este trabajo trata de hacer patente las necesidades que se han de satisfacer a la hora de desarrollar sistemas software destinados al tratamiento de este tipo de bibliotecas. También se analizan las aportaciones que en este caso pueden hacer las propuestas metodológicas de desarrollo software según la orientación a objetos y las propuestas existentes para el desarrollo de sistemas hipermedia. Por último, se concretan las fases del ciclo de vida de desarrollo de estos sistemas y las actividades a realizar en cada una de ellas.

2 Introducción

Últimamente Internet está poniendo a disposición de los usuarios una gran cantidad de información almacenada en diferentes medios (imagen, sonido, texto, etc.). Esto hace necesario desarrollar técnicas para la recuperación eficiente de esta información, así como desarrollar arquitecturas que sean capaces de soportarla. En este marco aparecen las bibliotecas digitales. Las bibliotecas digitales se entienden como grandes almacenes de información de distinto tipo (imagen, texto, animación, etc.) que necesitan ser gestionadas y recuperadas de una forma eficiente. Una biblioteca digital es una biblioteca que ha sido extendida y mejorada mediante la aplicación de la tecnología digital. Es la unión de ordenadores, sistemas de almacenamiento y redes de comunicaciones con el contenido y el software necesario para reproducir, emular y extender los servicios proporcionados por las bibliotecas convencionales. Una biblioteca digital debe cumplir todas las tareas de una biblioteca convencional y explotar las ventajas de la tecnología digital en el almacenamiento, la búsqueda y las comunicaciones, además de la integración de los nuevos tipos de medios.

La biblioteca digital proporciona a una comunidad de usuarios un acceso coherente a repositorios de información grandes y organizados. Las bibliotecas digitales son construidas, recogiendo y organizando la información por una comunidad de usuarios y sus funcionalidades son acordes a las necesidades de información de dicha comunidad. Las posibilidades de los usuarios para acceder, reorganizar y utilizar este repositorio están enriquecidas con las capacidades de la tecnología digital.

Entre las ventajas de las bibliotecas digitales sobre las tradicionales se puede citar la posibilidad de contener documentos cambiantes y transitorios, además de los documentos fijos y permanentes propios de las bibliotecas tradicionales. También es una ventaja el hecho de permitir el uso de la información de forma cooperativa y no aislada.

El campo de las bibliotecas digitales es la unión de varios subcampos de otros dominios y abarca muchos temas de interés. Por citar algunos,

- Técnicas de digitalización (OCR, OMR).
- Integración de múltiples tipos de medios.
- Colaboración y comunicación entre usuarios.
- Búsqueda de información basada en el contenido (imágenes, vídeos).
- Interacción con el usuario.
- Técnicas de acceso.
- Protección de los derechos intelectuales y seguridad.
- Manejo de información multilingüe.
- Técnicas de búsqueda, recuperación y descubrimiento.

- Técnicas de representación.
- Arquitecturas.
- Comercio y economía.
- Almacenamiento y acceso a datos multimedia.
- Distribución de la información.
- Gestión de bases de datos distribuidas.
- Interoperabilidad (enlace de varios sistemas. CORBA, DCOM).
- Visualización de resultados (VRML, disposición espacial cercana para elementos similares en contenido).
- Técnicas de diseño (Object-Oriented Hypermedia Design Model, OOHDM).

Existen publicaciones específicas sobre bibliotecas digitales (D-Lib magazine, Int. Journal on Digital Libraries de Springer), así como informes especiales en publicaciones más genéricas (Proc. of the ACM, ACM Multimedia, IEEE Multimedia). También existen congresos especializados en el tema, como la ACM Conference on Digital Libraries y el IEEE Advances in Digital Libraries (ADL).

Entre las áreas de aplicación más importantes se encuentran la Educación, la Investigación, la Cultura, el Turismo y el Ocio. Como proyectos concretos se pueden citar las Bibliotecas nacionales, los Museos, los Archivos Históricos (como el Archivo General de Indias de Sevilla, uno de los pioneros), las redes comunitarias de servicios (poblados electrónicos), los repositorios de Technical Reports (sobre todo en Computer Science), datos geográficos (Medio Ambiente) o de Literatura y las publicaciones electrónicas (como la ACM Digital Library).

Como vemos, las aplicaciones basadas en bibliotecas digitales son sistemas complejos que necesitan importantes recursos de almacenamiento y de funcionalidad. En este sentido, los sistemas para el tratamiento de bibliotecas digitales podrían compararse con los tradicionales sistemas de gestión, puesto que requieren una cierta funcionalidad para el mantenimiento y gestión de la información que almacenan. Pero por otro lado, los sistemas para el tratamiento de bibliotecas digitales se asemejan bastante a las aplicaciones multimedia y de la web. En ellos, al distribuirse por internet, los conceptos de navegación e interfaz del usuario deben contemplarse como parte esencial del desarrollo. Esta idea se manifiesta de forma gráfica en la figura 1.

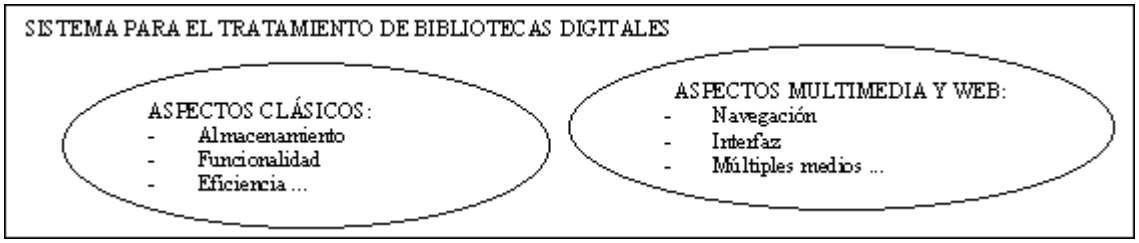

Figura 1: Relación entre las bibliotecas digitales, las aplicaciones clásicas y las aplicaciones hipermedia.

Teniendo en cuenta estas dos premisas de partida y asumiendo que actualmente no existe ningún tipo de metodología de desarrollo específica para elaborar sistemas de tratamiento de bibliotecas digitales, resulta interesante.

En los últimos años existe una tendencia a considerar el desarrollo de sistemas multimedia e hipermedia como fruto de un proceso de ingeniería del software, por lo que han surgido propuestas metodológicas como HDM (Hypertext Design Model) [Garzoto 1993] y RMM (Relationship Management Methodology) [Izakowitz 1995] o como las orientadas a objeto OOHDM [Rossi 1996] (Object Oriented Hypermedia Design Model) o EORM [Lange 1995] (Enhanced Object Relationship Model) que contemplan la necesidad de realizar un diseño previo a la construcción de sistemas multimedia, y ofrecen una serie de técnicas para obtener los modelos del sistema a desarrollar. Partiendo de que admitimos el paradigma de orientación a objetos como el más idóneo para desarrollar este tipo de sistemas, se puede considerar que las referencias metodológicas más válidas a la hora de intentar definir un marco procedimental para el desarrollo de los sistemas de tratamiento de bibliotecas digitales se pueden localizar en las propuestas metodológicas de desarrollo de sistemas software orientados a objetos con carácter general y en aquellas otras específicas del dominio de los sistemas hipermedia. De esta forma, la metodología de desarrollo de aplicaciones basadas en bibliotecas digitales tomaría ideas de las metodologías clásicas de la orientación a objetos e ideas de las nuevas metodologías de desarrollo de aplicaciones multimedia. Esta idea se plasma en la figura 2. Si observamos el diagrama de la figura 2 es similar a la de la figura 1 en la misma medida en que la propuesta de metodología que se va a realizar se ajusta al doble carácter (el carácter de aplicación de gestión clásica y el carácter de aplicación multimedia).

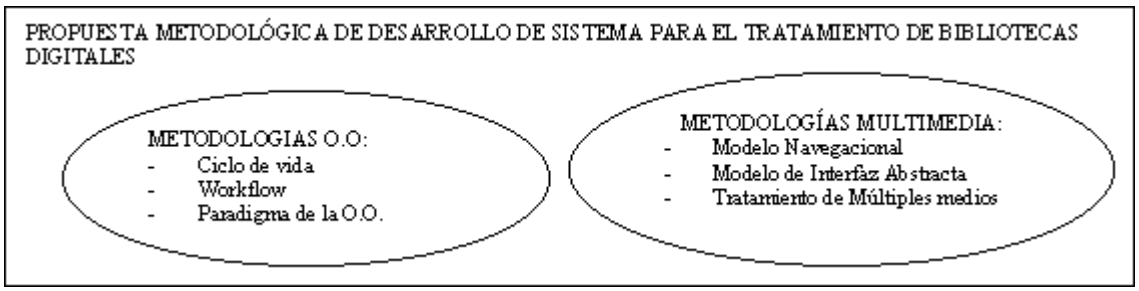

Figura 2: Relación entre las metodologías orientadas a objeto, las metodologías de aplicaciones hipermedia y la propuesta de metodología para el desarrollo de sistemas de tratamiento de bibliotecas digitales

3 OOHDM como referencia de la propuesta

De las cuatro metodologías señaladas en el apartado anterior que han tenido más auge dentro del diseño de las aplicaciones multimedia: RMM, HDM, EORM y OOHDM; solamente las dos últimas asumen el paradigma de la orientación a objetos como paradigma de diseño. Tomar el enfoque orientado a objetos como paradigma de diseño es muy útil debido al gran nivel de abstracción que ofrece y a sus mecanismos de composición que facilitan el modelado de la estructura hipermedial. [Lange 1995]. Por ello, parecen las más interesantes de estudiar como base para el desarrollo de sistemas basados en bibliotecas digitales.

Si comparamos ahora estas dos metodologías, para seleccionar una como marco de referencia, veremos que, a pesar de que en el diseño de los conceptos de navegación de la aplicación son bastantes similares, OOHDM hace más hincapié en el diseño de la interfaz de usuario, dedicando una fase de la metodología al diseño de la interfaz abstracta. Por esta razón, se elige en concreto a OOHDM y a su utilización en el desarrollo de aplicaciones basadas en bibliotecas digitales.

3.1 Resumen de OOHDM

A continuación vamos a ver un pequeño resumen de OOHDM, de sus fases, de sus objetivos y de los productos que resultan al aplicarla.

En OOHDM se proponen 4 fases de desarrollo:

- Diseño Conceptual
- Diseño Navegacional
- Diseño de Interfaz Abstracto
- Implementación

OOHDM es una mezcla de estilos de desarrollo basado en prototipos, en desarrollo interactivo y de desarrollo incremental. En cada fase se elabora un modelo orientado a objetos conceptual que recoge las características a resaltar en la misma incrementando los resaltados de la fase o fases anteriores.

Fase I: Diseño Conceptual

Se construye un modelo orientado a objetos que represente el dominio de la aplicación usando las técnicas propias de la O.O. La finalidad principal durante esta fase es capturar el dominio semántico de la aplicación en la medida de lo posible, teniendo en cuenta el papel de los usuarios y las tareas. El resultado de esta fase es un modelo de clases relacionadas que se divide en subsistemas. En la tabla 1 se esquematiza esta fase.

Fase	Diseño conceptual
Productos	Diagrama de Clases, División en subsistemas y relaciones
Herramientas	Técnicas de modelado O.O, patrones de diseño
Mecanismos	Clasificación, agregación, generalización y especialización
Objetivo de diseño	Modelo semántico de la aplicación

Tabla 1: Fase de Diseño Conceptual de OOHDM

Fase II: Diseño Navegacional

En OOHDM una aplicación se ve a través de un sistema de navegación. En la fase de diseño navegacional se debe diseñar la aplicación teniendo en cuenta las tareas que el usuario va a realizar sobre la aplicación. Para ello, hay que partir del esquema conceptual desarrollado en la fase anterior. Hay que tener en cuenta que sobre un mismo esquema conceptual se pueden desarrollar diferentes modelos navegacionales (cada uno de los cuales dará origen a una aplicación diferente).

La estructura de navegación de una aplicación hipermedia está definida por un esquema de clases de navegación específica, que refleja una posible vista elegida. En OOHDM hay una serie de clases especiales predefinidas, que se conocen como clases navegacionales: *Nodos, Enlaces y Estructuras de acceso*, que se organizan dentro de un *Contexto Navegacional*. Mientras que la semántica de los nodos y los enlaces son comunes a todas las aplicaciones hipermedia, las estructuras de acceso representan diferentes modos de acceso a esos nodos y enlaces de forma específica en cada aplicación.

1- *Nodos*: Los nodos son contenedores básicos de información de las aplicaciones hipermedia. Se definen como vistas orientadas a objeto de las clases definidas durante el diseño conceptual usando un lenguaje basado en query, permitiendo así que un nodo sea definido mediante la combinación de atributos de clases diferentes relacionadas en el modelo de diseño conceptual. Los nodos contendrán tanto atributos de tipos básicos (donde se pueden encontrar tipos como imágenes o sonidos) y enlaces.

2- *Enlaces*: Los enlaces reflejan la relación de navegación que puede explorar el usuario. Ya sabemos que para un mismo esquema conceptual puede haber diferentes esquemas navegacionales y los enlaces van a ser imprescindibles para poder crear esas vistas diferentes. Las clases enlaces sirven para especificar los atributos de enlaces y estos a su vez para representar enlaces entre clases nodos o incluso entre otros enlaces. En cualquier caso, el enlace puede actuar como un objeto intermedio en un proceso de navegación o como un puente de conexión entre dos nodos.

3- *Estructuras de Acceso*: Las estructuras de acceso actúan como índices o diccionarios que permiten al usuario encontrar de forma rápida y eficiente la información deseada. Los menús, los índices o las guías de ruta son ejemplos de estas estructuras. Las estructuras de acceso también se modelan como clases, compuestas por un conjunto de referencias a objetos que son accesibles desde ella y una serie de criterios de clasificación de las mismas.

4- *Contexto Navegacional:* Para diseñar bien una aplicación hipermedia, hay que prever los caminos que el usuario puede seguir, así es como únicamente podremos evitar información redundante o que el usuario se pierda en la navegación. En OOHDM un contexto navegacional está compuesto por un conjunto de nodos, de enlaces de clases de contexto y de otros contextos navegacionales. Estos son introducidos desde clases de navegación (enlaces, nodos o estructuras de acceso), pudiendo ser definidas por extensión o de forma implícita.

5- *Clase de Contexto*: Es otra clase especial que sirve para complementar la definición de una clase de navegación. Por ejemplo, sirve para indicar qué información está accesible desde un enlace y desde dónde se puede llegar a él.

TRANSFORMACIÓN NAVEGACIONAL: Cuando la aplicación se ejecuta en una sola vista, el contexto navegacional tiene bastante poder semántico como para representar la aplicación. Sin embargo, cuando pueden aparecer diferentes contextos de navegación a la vez, se requiere el uso de transformaciones navegacionales. A través de ellos podemos expresar concurrencias o navegaciones que se ejecutan a la par. En la tabla 2 vemos un resumen de esta fase.

Fase	Diseño navegacional
Productos	Nodos, enlaces, estructuras de accesos, contextos navegacionales y transformaciones navegacionales
Herramientas	Técnicas de modelado O.O, patrones de diseño, diagramas de estados, escenarios
Mecanismos	Clasificación, agregación, generalización y especialización
Objetivo de diseño	Establecer los recorridos que el usuario puede seguir por la aplicación

Tabla 2: Fase de Diseño Navegacional de OOHDM

Fase III: Diseño de Interfaz Abstracto

Una vez definida la estructura navegacional, hay que prepararla para que sea perceptible por el usuario y esto es lo que se intenta en esta fase. Esto consiste en definir qué objetos de interfaz va a percibir el usuario, y en particular el camino en el cuál aparecerán los diferentes objetos de navegación, qué objeto de interfaz actuarán en la navegación, la forma de sincronización de los objetos multimedia y el interfaz de transformaciones. Al haber una clara separación entre la fase anterior y esta fase hace que para un mismo modelo de navegación se puedan definir diferentes modelos de interfaces, permitiendo, así, que el interfaz se ajuste mejor a las necesidades del usuario.

VISTAS ABSTRACTAS DE DATOS (ADVs): Las ADVs no son más que modelos formales de objetos de interfaz que se usan para mostrar:

a) La forma en que se estructura la interfaz, para ello se usan las vistas abstractas de datos. Estos son objetos que tienen un estado y una interfaz. Son objetos abstractos en el sentido de que solo representan el objeto y su estado y no la implementación, sin entrar para nada en aspectos concretos como el color de la pantalla o la ubicación en ésta de la información. Así tendremos un conjunto de objetos abstractos de datos, que gestionan las estructuras de datos y de control, y un conjunto de aspectos de interfaz, como las entradas del usuario y las salidas que se le ofrecen.

b) La forma en que la interfaz se relaciona con los objetos de navegación. Para ello se usan diagramas de configuración. Los diagramas de configuración van a ser grafos dirigidos que permitirán indicar de qué objetos de navegación toman la información los ADV.

c) La forma en que la aplicación reacciona a eventos externos, para ello se usan los ADVs-Charts. Los ADVs-Charts van a ser diagramas bastante similares alas máquinas de estados. A través de ellas se puede indicar los eventos que afectan a una ADV y cómo ésta reacciona a ese elemento.

En la tabla 3 vemos esquematizada esta fase.

Fase	Diseño de interfaz abstracta
Productos	Objetos de interfaz abstracta, respuestas a eventos externos y transformaciones de interfaz
Herramientas	ADVs, Diagramas de configuración, ADV-Charts y patrones de diseño
Mecanismos	Mapeado entre los objetos de navegación y los objetos visibles
Objetivo de diseño	Modelado de los objetos perceptibles por el usuario y de cómo le afecta a la aplicación los eventos externos

Tabla 3: Fase de Diseño de Interfaz Abstracto de OOHDM

Fase IV: Implementación

Una vez obtenido el modelo conceptual, el modelo de navegación y el modelo de navegación abstracta, sólo queda llevar los objetos a un lenguaje concreto de programación, para obtener así la implementación ejecutable de la aplicación. En la tabla 4 vemos un resumen de esta fase.

Fase	Implementación
Productos	Aplicación ejecutable
Herramientas	El entorno del lenguaje de programación
Mecanismos	Los ofrecidos por el lenguaje
Objetivo de diseño	Obtener la aplicación ejecutable

Tabla 4: Fase de Implementación de OOHDM

Para terminar, podríamos decir que los puntos claves de OOHDM se encuentran en:

- OOHDM contempla los objetos que representan la navegación como vistas de los objetos detallados en el modelo conceptual.
- OOHDM abstrae los conceptos básicos de la navegación: nodos, enlaces e índices y los organiza mediante el uso de los contextos de navegación, permitiendo así una organización adecuada de los mismos.
- OOHDM separa las características de interfaz de las características de navegación, con las ventajas que esto supone.

3.2 Aplicación de OOHDM a las bibliotecas digitales

En un principio, la propuesta de OOHDM podría ser bastante apropiada para el tratamiento de los aspectos de navegación e interfaz que aparecen en los sistemas de tratamiento de bibliotecas digitales, puesto que los trata de una forma muy concreta y detallada y propone herramientas concretas para su estudio. Sin embargo, en OOHDM no se cubren aspectos tan importantes como la especificación de requisitos, el análisis o el diseño de la arquitectura del sistema. Por ello, no podemos tomarla como una propuesta que cubra, como metodología de desarrollo, todas las necesidades que surgen en el ámbito de las bibliotecas digitales.

4 El Proceso Unificado como referencia de la propuesta

Como ya hemos venido diciendo, los sistemas de tratamiento de bibliotecas digitales heredan características de los problemas clásicos de ingeniería del software, como podrían ser el hecho de que requieren una potente estructura de mantenimiento y recuperación de la información y que deben cubrir un importante y complejo número de requisitos funcionales. Pero además tienen características propias de las aplicaciones multimedia, como podrían ser las estructuras navegacionales que en ella aparecen y la multiplicidad de medios que deben tratar.

En este sentido no podemos ceñirnos a usar únicamente OOHDM como metodología de desarrollo para las aplicaciones basadas en bibliotecas digitales, puesto que vamos a detectar aspectos que esta metodología no va a ser capaz de cubrir. Para captar estos aspectos que quedan fuera de OOHDM, vamos a usar la propuesta realizada por el Proceso Unificado de UML.

4.1 Resumen del Proceso Unificado

El ciclo de vida del Proceso Unificado es el ciclo de vida que los autores creadores de UML proponen para el desarrollo. El ciclo de vida del Proceso Unificado es prácticamente un ciclo de vida en espiral. Sus características esenciales es que es incremental e iterativo. Iterativo porque se producen varios ciclos en su desarrollo en cada uno de los cuales se concreta más el producto resultado del ciclo anterior. E iterativo porque en cada ciclo, el producto resultante se va a adecuar más a las necesidades de los clientes.

El ciclo de vida del Proceso Unificado asume que en la vida del proyecto existen una serie de ciclos, cada uno de los cuales tiene cuatro fases. Cada una de las fases puede tener varias iteraciones y en éstas a su vez se distinguen cinco flujos de trabajo. Vemos un esquema de ello en la figura 3. En esta figura se usa la propia representación de la orientación a objetos para estudiar la estructura de composición que se da en el ciclo de vida del Proceso unificado..

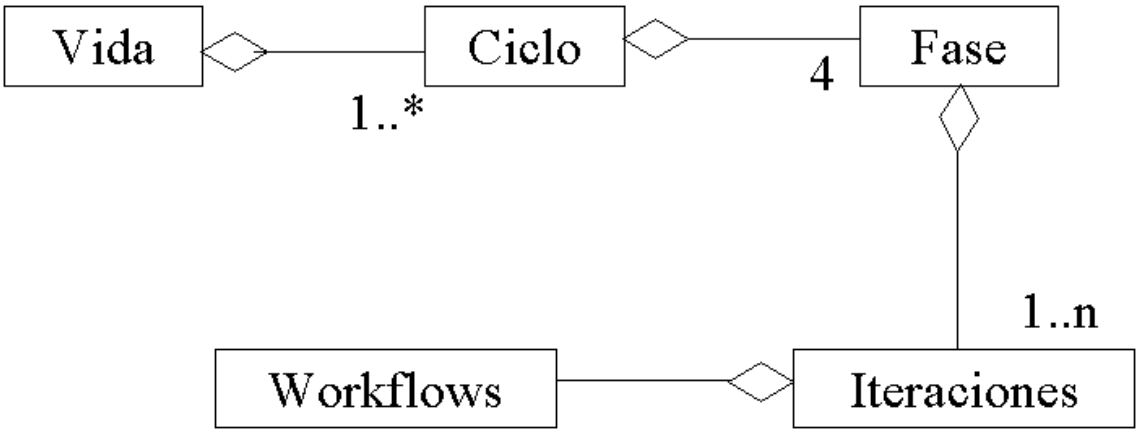

Figura 3: Esquema del ciclo de vida del Proceso Unificado

Tras cada ciclo, obtendremos un resultado final que será una versión del producto.

Como ya hemos dicho, en cada ciclo existen 4 fases que se detallan en la figura 4.

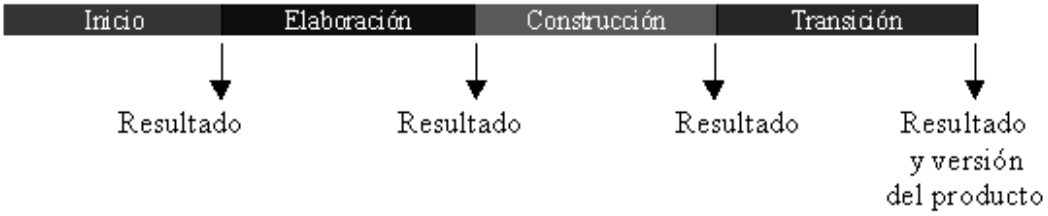

Figura 4: Fases del Proceso Unificado

Como vemos, tras cada una de las fases obtenemos un resultado de fase. El resultado de la última fase va a coincidir con la versión del producto. A continuación se presenta un pequeño resumen de lo que hay que realizar en cada una de ellas.

Fase de Inicio

En ella las tareas que fundamentalmente se deben llevar a cabo son:

- Estudiar la viabilidad del ciclo
- Identificar riesgos
- Estimar costes y realizar la planificación
- Perfilar la arquitectura del sistema

Fase de Elaboración

En esta fase, debemos:

- Revisar la identificación de riesgos de la etapa anterior
- Especificar los requisitos de usuario
- Definir la arquitectura del sistema
- Preparar el plan de Proyecto
- Revisar la planificación y estimación de costes

- Definir parámetros generales como calidad, fiabilidad, riegos, tiempos de respuesta, etc.

Fase de Construcción

En la fase de construcción, se debe:

- Completar la arquitectura
- Obtener el código ejecutable
- Obtener el producto beta

Fase de Transición

La fase de transición es la última fase del ciclo y tras ella obtendremos una versión del producto. Sus tareas fundamentales son:

- Realizar pruebas beta y subsanar deficiencias
- Formación del personal, mantenimiento y garantía

Por su parte, hemos dicho que en cada fase hay N iteraciones. Tras cada iteración se obtiene los que se conoce como producto interno. El producto interno es algo que, a diferencia de los resultados de fase, no suele darse al usuario. El esquema de las iteraciones lo vemos en la figura 5.

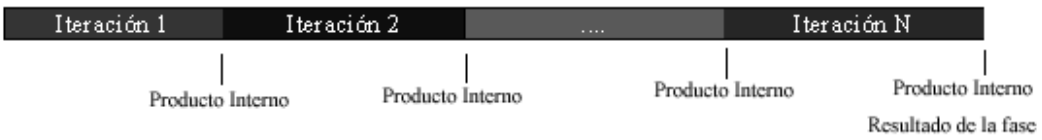

Figura 5: Iteraciones del Proceso Unificado

Por su parte, en cada iteración hay cinco flujos de trabajo: requisitos, análisis, diseño, implementación y pruebas. Realmente, es como si dentro de cada iteración elaborásemos un ciclo de vida secuencial puesto que estos flujos de trabajo son similares a los propuestos por el ciclio de vida clásico.

4.2 Aplicación de UML a las bibliotecas digitales

Analizando el modelo de desarrollo del Proceso Unificado, si lo queremos utilizar para sistemas basados en bibliotecas digitales, encontraríamos una serie de aspectos no contemplados, debido fundamentalmente a las características que hacen diferentes a estos sistemas de los sistemas clásicos de gestión. Así, por ejemplo, los aspectos de navegación que en las bibliotecas digitales se convierten en un aspecto crítico, no se tienen en cuenta en el proceso unificado. Sin embargo, y por la similitud existente entre los sistemas clásicos y los basados en bibliotecas digitales, con respecto a la recolección de los requisitos de almacenamiento, funcionales, de concurrencia o de seguridad, este Proceso Unificado sí sería aplicable en muchos aspectos.

5 Aproximación metodológica de desarrollo para los sistemas de tratamiento de bibliotecas digitales

Según lo indicado en los apartados 2.2 y 3.2, a partir del Proceso Unificado y de OOHDM podemos obtener un marco de referencia para desarrollar sistemas para el tratamiento de bibliotecas digitales [Escalona 2000].

5.1 Ciclo de Vida

Vamos a plantearnos como ciclo de vida para el desarrollo de aplicaciones basadas en bibliotecas digitales, el propuesto por el Proceso Unificado [Jacobson 1999]. Por consiguiente, el ciclo de vida propuesto será un ciclo de vida en espiral.

No obstante, la propuesta que aquí se hace difiere en las tareas a realizar en los flujos y en los productos a obtener. En la figura 6 vemos un esquema general de la propuesta.

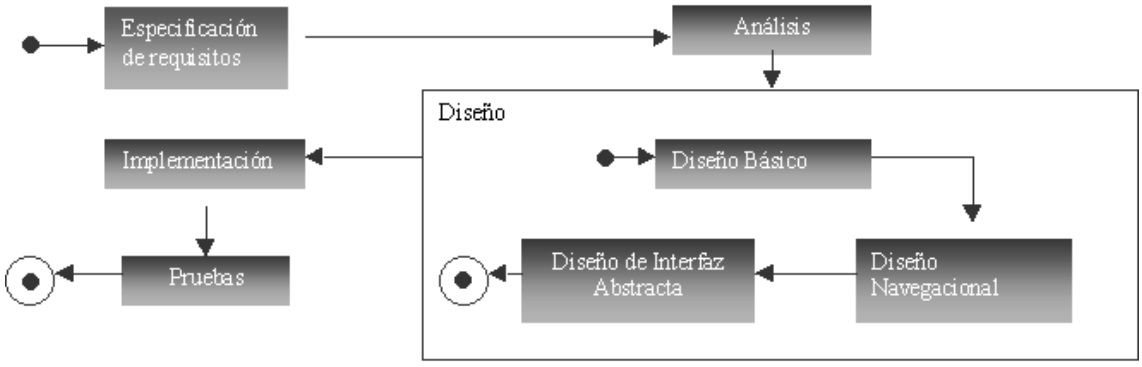

Figura 6: Esquema de la secuencia de los flujos de trabajo para la metodología de desarrollo de sistemas para el tratamiento de bibliotecas digitales

A modo de resumen, en la tabla 5 se indica cuáles son los productos que hay que obtener en cada uno de los flujos de trabajo propuestos:

Flujo de trabajo		**Productos resultantes**
Especificación de requisitos		Catálogo de requisitos del sistema
Análisis		Modelo de clases de análisis
Diseño	Diseño Básico	- La arquitectura abstracta del sistema - La división del sistema en subsistemas - El diseño de los casos de uso - El modelo de clases de diseño
	Diseño Navegacional	- Modelo de clases navegacionales - Contextos navegacionales
	Diseño de Interfaz Abstracta	- Modelo de clases navegacionales refinado con los aspectos de interfaz abstracta - Vistas abstractas de datos (ADV) - Diagramas de configuración - Modelo dinámico de los ADVs
Implementación		Programa ejecutable
Pruebas		Plan de pruebas

Tabla 5: Resumen de la propuesta metodológica y de los productos resultantes

A continuación vamos a ir analizando cada uno de los flujos de trabajo de forma más concreta. La fase de mantenimiento vamos a dejarla al margen en este momento. Sería conveniente anotar que, ya que OOHDM y el Proceso Unificado no proponen ni dicen nada sobre esta fase, habría que recurrir a otras propuestas realizadas para proyectos de desarrollo software, con el fin de encontrar una referencia en la que basarse.

Para ejemplificar las ideas propuestas, vamos a plantearnos el desarrollo de una aplicación que trate con una biblioteca digital que muestre los datos referentes a los bienes muebles del patrimonio histórico de una comunidad autónoma. Este ejemplo solo contendrá una parte de las necesidades que serían necesarias para la gestión de esta biblioteca. La idea no es desarrollar esta biblioteca, sino tomar las necesidades que surgen en su elaboración como ejemplo. Este ejemplo es parte de un ejemplo real que se está desarrollando actualmente [Torres 2000].

5.2 Especificación de requisitos

Para estudiar los tipos de requisitos que habría que recolectar en el desarrollo de aplicaciones basadas en bibliotecas digitales, lo primero que habría que tener en cuenta son las características que ya hemos destacado de este tipo de aplicaciones. Por un lado, es necesario estudiar los requisitos clásicos que se estudian en las aplicaciones clásicas de gestión [Durán 1999], pero además hay que recabar información referente a esos aspectos que acercan las bibliotecas digitales a las aplicaciones hipermedia, como podrían ser los aspectos de navegación y de interfaz abstracta de usuario.

De esta forma, de la fase de especificación de requisitos se debe conseguir el catálogo de requisitos del sistema que englobe: los requisitos de almacenamiento de información; los requisitos funcionales, descritos a través de los casos de uso; los requisitos de navegación y los requisitos de interfaz de usuario. Además se podrían

incorporar otra serie de requisitos como son los requisitos de comunicaciones del sistema, requisitos de seguridad, requisitos de portabilidad, etc. si se estimase necesario.

Iremos viendo la propuesta más concretamente a medida que la aplicamos al sistema de tratamiento de la biblioteca digital de los bienes muebles que hemos planteado en el apartado anterior .

REQUISITOS DE ALMACENAMIENTO DE INFORMACIÓN

Vamos a asumir la plantilla de requisitos de almacenamiento de información propuesta en [Durán 1999]. Sin embargo, con respecto a este grupo de requisitos, habría que tener especial cuidado en los tipos de datos que se van a almacenar. Así, en los sistemas basados en bibliotecas digitales nos encontramos con información almacenada en distintos medios: imágenes, sonidos, animaciones, gráficos, textos, documentos, etc. El conocimiento de los tipos de datos que vamos a manejar, sobretodo cuando son tan variables como en los sistemas para el tratamiento de bibliotecas digitales, es de vital importancia desde el comienzo del desarrollo de la aplicación. Aspectos como la eficiencia y la eficacia final de la misma dependerá bastante del manejo adecuado que haga de los medios con los que va a trabajar y de lo apropiado que sea el sistema de almacenamiento para el tipo de datos que se está almacenando. En las primeras fases no será necesario entrar a detallar los formatos concretos con los que vamos a trabajar, pero es aconsejable conocer los medios que se nos pueden presentar. En la tabla 6 vemos el catálogo de requisitos de almacenamiento de información del sistemas de tratamiento de bienes muebles [Durán 1999].

RI-01	Datos del bien
Descripción	Recoge los datos básicos que catalogan un bien mueble y que sirven para diferenciarlo uno de otro.
Datos concretos	DATOS BÁSICOS • Código: Número compuesto por dos dígitos que codifican la ciudad donde se encuentra el bien, tres que indican el municipio de Andalucía, cuatro que identifican el monumento en el que se encuentra y cuatro para indicar el número de bien mueble dentro del monumento • Denominación: Nombre del bien • Dirección • Provincia • Municipio • Inmueble: Nombre del monumento al que pertenece la pieza • Otra denominación: Otro nombre que pueda recibir el bien • Visitable: {Si, No}, indica si el monumento es visitable o no DATOS DE DESCRIPCIÓN • Tipologías: Conjunto de palabras que indican las tipologías a la que pertenece el bien • Periodos históricos: Conjunto de palabras que indican los periodos históricos dentro de los cuales se cataloga el bien • Iconografías: Conjunto de palabras que indican las iconografías a la que pertenece el bien • Actividades: Conjunto de palabras que indican las actividades de uso para las que está pensado el bien • Cronología: Fecha en la que se supone que se elaboró el bien • Certeza: Puede tomar los valores aproximada, exacta o poco fiable, e indica la certeza del valor del campo de cronología • Autores: Conjunto de palabras que indican los autores del bien y el papel que realizaron en la obra: Pintores, Escultores,... • Imágenes: Conjunto de imágenes que muestra fotos del bien o de partes del bien DATOS MUSEOGRÁFICOS • Nombre del museo • Fecha de ingreso en museo • Expuesta: {Si, No}, indica si está o no expuesta en el museo

Tabla 6: Requisitos de almacenamiento de información para el sistema de tratamiento de bibliotecas digitales de los bienes muebles

REQUISITOS FUNCIONALES

No vamos a pararnos a detallar cada uno de los requisitos funcionales del sistema puesto que sería demasiado complejo. Vamos a suponer tres:

- RF-01: La aplicación debe permitir dar de alta nuevos bienes, eligiendo la provincia, el municipio y el inmueble, se asignará automáticamente el número de mueble de forma secuencial.

- RF-02: La aplicación debe permitir seleccionar un bien mueble y modificar sus datos

- RF-03: La aplicación debe permitir seleccionar un bien mueble y modificar sólo sus datos museográficos

- RF-04: La aplicación debe permitir realizar consultas sobre los datos mediante los campos de identificación y descripción.

Para cada uno de estos requisitos habría que elaborar su caso de uso, de esta forma, tendríamos normalizada su representación. En la tabla 7, se muestra la plantilla que recogería la descripción del requisito RF-01. [Durán 1999]

RF-01	Proceso de dar de alta un bien	
Descripción	El sistema debe comportarse como se describe en el siguiente caso de uso concreto cuando el usuario quiere añadir un nuevo bien	
Precondición	Ninguna	
Secuencia normal	Paso	Acción
	1	El usuario solicita la opción de dar de alta un bien
	2	El sistema solicita que seleccione la provincia donde se encuentra ubicado el bien
	3	El usuario selecciona la provincia
	4	El sistema solicita que se seleccione el municipio donde se encuentra ubicado el bien
	5	El usuario selecciona el municipio
	6	El sistema solicita que se seleccione el edificio donde se encuentra ubicado el bien
	7	El usuario selecciona el inmueble
	8	El sistema muestra una lista con los muebles ubicados en ese inmueble y pide al usuario que indique la nueva denominación si no se encuentra en la lista
	9	Si el mueble no está, el usuario introduce la denominación y confirma la operación
	10	El sistema da de alta el nuevo bien con el código siguiente al último que había dentro del inmueble
Postcondiciones	Ninguna	
Excepciones	Paso	Acción
	9	Si el mueble está, el usuario se lo indica al sistema
Rendimiento	Paso	Acción
	-	-
Frecuencia esperada	5 veces / semana	

Tabla 7: Requisito funcional para el alta de los muebles para el sistema de tratamiento de bibliotecas digitales de los bienes muebles

REQUISITOS DE NAVEGACIÓN

Al entrar en conceptos como los accesos entre los objetos de la interfaz y las vistas de la información, reconocemos otra necesidad que hay que cubrir en la especificación de requisitos para los sistemas para el tratamiento de bibliotecas digitales y es que es necesario recoger los requisitos que marcan los aspectos de navegación.

En los sistemas basados en bibliotecas digitales, la información se muestra agrupada en diferentes grupos siguiendo unos determinados criterios que pueden ser muy variables. Partiendo de esta idea, la navegación nace como la posibilidad de viajar,

es decir de pasar de un grupo de información a otro. Los requisitos de navegación [Rossi 1996] van a permitir capturar cómo desea el cliente que se agrupe la información y qué posibilidades existen de ir de uno de estos grupos a otro.

Los requisitos de navegación van a estar basados sobre los *nodos* y los *enlaces* [Lange 1996]. Un nodo (o nodo navegacional) no es más que un concepto que nos va a permitir describir cómo el usuario desea que se agrupe la información que se ha descrito en los requisitos de almacenamiento de información. Cada requisito de almacenamiento de información se debe describir de manera genérica, describiendo a qué tipo información se refiere. Pero también debe describirse de manera concreta, indicando todos los datos relevantes que debe almacenar. Un nodo puede referenciar datos concretos de diferentes requisitos de información, pudiendo un dato concreto estar a su vez referenciado por varios nodos.

Por su parte, un enlace es un concepto abstracto que nos va permitir indicar que desde un nodo, en un momento dado de la ejecución se podrá navegar a otro nodo. El hecho de que exista un enlace entre dos nodos, no indica que forzosamente en la aplicación se vaya a navegar por ese enlace. Por otra parte, un enlace es un camino unidireccional entre dos nodos. Así si definimos que existe un enlace del nodo A al nodo B, en principio, no podríamos ir desde el nodo B al A a no ser que definiésemos un nuevo enlace que así lo indicara.

Para definir cada uno de los nodos que van a formar parte del sistema debemos usar la plantilla que se muestra en la tabla 8.

N-<id>	<Nombre descriptivo>
Descripción	<Descripción breve del significado del nodo>
Datos	<Listado de los datos que almacena el nodo, como serán datos específicos de algún requisito de almacenamiento se indicará como una lista de la forma:> • RI-<id>.<dato_específico> • ...
Enlaces fuente	<Listado de nodos alcanzables desde este nodo. Será una lista de la forma:> • N-<id> • ...
Enlaces destino	<Listado de nodos desde los que se puede alcanzar este nodo. Tendrá la forma:> • N-<id> • ...
Comentarios	<Otros comentarios sobre el nodo>

Tabla 8: Plantilla para la captura de requisitos navegacionales

A través del apartado “Datos” podemos indicar cualquier dato específico de cualquiera de los requisitos de almacenamiento que hayamos detectado. Además, existe la posibilidad de especificar también datos calculados que puedan aparecer.

Para nuestro ejemplo vamos a definir tres nodos. El primero de ellos, cuya plantilla vemos en la tabla 9 nos muestra un nodo en el que se muestran los datos de identificación del bien. Como vemos, esto es un recorte del requisito de almacenamiento que recoge los datos básicos que catalogan un bien mueble y que sirven para diferenciarlo uno de otro. Así, este nodo mostrará los datos código, denominación, provincia, etc. pero no mostrará datos como la tipología o la iconografía. Como vemos

también en la plantilla, desde este nodo se puede navegar al nodo N-02 y al N-03 que vemos definidos en las tablas 10 y 11 respectivamente.

N-01	Datos de identificación del bien mueble
Descripción	Recoge los datos que se catalogan como datos de identificación del bien
Datos	• RI-01.Código • RI-01.Denominación • RI-01.Dirección • RI-01.Provincia • RI-01.Municipio • RI-01.Inmueble • RI-01.Otra denominación • RI-01.Visitable
Enlaces fuente	• N-02 • N-03
Enlaces destino	• N-02 • N-03

Tabla 9: Plantilla para el nodo de identificación

El siguiente nodo que vamos a definir será el que muestre los datos de descripción del bien. Vemos que, al igual que en el anterior, este nodo referencia a un subconjunto de los datos concretos del requisito RF-01 (Tabla 6). Vemos además que este nodo sólo puede alcanzarse desde el nodo de identificación y que es sólo a él al que se puede navegar.

N-02	Datos de descripción del bien mueble
Descripción	Recoge los datos que se catalogan como datos de descripción del bien
Datos	• RI-01.Tipologías • RI-01.Períodos históricos • RI-01.Iconografías • RI-01.Activiades • RI-01.Cronología • RI-01.Certeza • RI-01.Autores • RI-01.Imágenes
Enlaces fuente	• N-01
Enlaces destino	• N-01

Tabla 10: Plantilla para el nodo de descripción

El último nodo que vamos a definir será el que muestre los datos museográficos del sistema. Este nodo nos va a mostrar los datos del requisito RI-01 que referencia a los datos de los museos y sólo va a ser accesible desde el nodo N-01.

N-03	Datos museográficos del bien mueble
Descripción	Recoge los datos que recogen los museos que albergan el bien
Datos	RI-01.Nombre del museo RI-01.Fecha de ingreso en museo RI-01.Expuesta
Enlaces fuente	N-01
Enlaces destino	N-01

Tabla 11: Plantilla para el nodo de datos museográficos

Analizando estas plantillas vemos que a través de los nodos y los enlaces, estamos definiendo un grafo direccional, que denominaremos *grafo de navegación*, que representa las posibilidades de navegación para el sistema. En este grafo, las aristas son los enlaces y los nodos son los nodos navegacionales definidos en nuestro catálogo de requisitos. Así, para nuestro ejemplo, el grafo navegacional sería el mostrado en la figura 7.

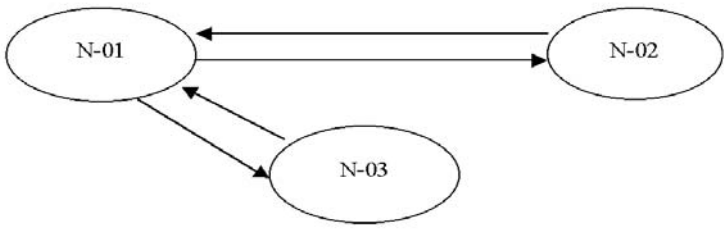

Figura 7: Nodo navegacional para el ejemplo de los bienes muebles

REQUISITOS DE INTERFAZ DE USUARIO

Con respecto a los requisitos de interfaz de usuario, en las aplicaciones de gestión clásicas se aconseja ser cuidadoso, ya que aún no se conocen bien las dificultades que pueden surgir a la hora de diseñar e implementar la interfaces [Durán 1999], por lo que se propone no entrar en detalles demasiado específico, esto es lo que hemos hecho en el ejemplo. Este razonamiento es bastante acertado cuando estamos trabajando con sistemas en los que en la interfaz no es más que la forma en la que se muestran al usuario los datos. Sin embargo, cuando estamos con aplicaciones, como las basadas en bibliotecas digitales, en las que la interfaz constituye un elemento fundamental, ya que a través de ella los usuarios interactúan con el sistema para realizar sus tareas, sí que es necesario entrar en más detalle en la fase de especificación de requisitos. El entrar en más detalles no se refiere a especificar aspectos como los colores de la pantallas o los tipos de letra, pero está claro que la presencia de información multimedia que se da en el ámbito de las bibliotecas digitales, afecta por un lado a la interfaz mediante la cual el usuario realiza sus operaciones y por otro lado a la interfaz que le presenta los resultados de su operación. Una interfaz bien diseñada ayuda a los usuarios a realizar

sus tareas de una manera cómoda. Por tanto, la interfaz debe ser diseñada pensando en los usuarios, y si es posible contando incluso con su participación. Por ello, lo que es necesario recoger son aspectos como las vistas que cada usuario va a tener de la información y los accesos que hay entre los objetos de las interfaces.

Además, cuando a las aplicaciones se le añade la posibilidad de navegación, dependiendo del grupo de usuarios con el que estemos trabajando, los nodos definidos en el modelo navegacional pueden mostrar toda o sólo parte de su información. Se definirán como requisitos de interfaz abstracta [Cowan 1992] a aquellos requisitos que nos indican qué visión y qué posibilidad de uso de la información tiene cada usuario del sistema. Para capturar estos requisitos es necesario agrupar a los posibles usuarios de la aplicación en diferentes grupos y describirlos mediante la plantilla mostrada en la tabla 12.

US-<id>	<Código de identificación>
Descripción	<Descripción del rol del usuario>
Requisitos funcionales	<Listado de los requisitos funcionales en lo que puede participara como actor.. Será una lista de la forma:> • RF-<id> • ...
Nodos	<Listado de nodos a que se visualizarán para el cliente. Si no se indican campos, se supone que el usuario podrá ver todos los datos especificados en el nodo. Si no, se indicarán los campos que verá. Esta lista tendrá el siguiente formato:> • N-<id> · RI-<id>.<dato_específico> · ... • ...
Comentarios	<Otros comentarios sobre el nodo>

Tabla 12: Plantilla para la captura de requisitos de interfaz abstracta

Cuando se capturan los requisitos funcionales, cada uno de ellos deben identificarse mediante un código de la forma RF-<id>, como ya hemos visto. En el apartado de "Requisitos Funcionales" se podrán indicar, haciendo referencia a los códigos de éstos, cuales son los requisitos funcionales en las que el usuario en concreto puede tomar el papel de actor externo. Con el apartado de "Nodos", se define la visión que para los nodos descritos tiene el usuario. Si sólo definimos el nodo, el usuario que definimos podrá ver todos los datos del nodo. Si sólo queremos que vea parte de esta información, enumeraremos qué datos del nodo puede ver. Así, para nuestro ejemplo podríamos decir que existen tres tipos de usuario:

- Usuarios que pueden dar de alta bienes y modificar cualquier dato

- Usuarios que sólo pueden modificar los datos museográficos y consultas, que serán aquellas personas que trabajan en los museos que almacenan los bienes

- Usuarios que sólo pueden hacer consultas sin modificar ningún tipo de información

Veamos como se aplicaría la plantilla de interfaz abstracta para cada uno de ellos. El primer tipo de usuario, el que puede realizar todas las operaciones, podrá realizar

consultas, modificaciones y altas y además puede ver el contenido de cualquiera de los nodos que se han definido en los requisitos navegacionales.

US-01	Usuario que pueden dar de alta bienes y modificar cualquier dato
Descripción	Será el usuario del sistema que tendrá todos los permisos para trabajar con la aplicación
Requisitos funcionales	• RF-01 • RF-02 • RF-04
Nodos	• N-01 • N-02 • N-03

Tabla 13 : Tabla para la descripción de los usuarios con todos los permisos

El segundo tipo de usuario, sólo va a poder realizar consultas y modificaciones sobre los datos museográficos. Además, no podrán visualizar el nodo de datos de descripción ni podrán ver en su totalidad los datos de identificación del bien. Vemos en su descripción, tabla 14, qué datos concretos del nodo de identificación podrá ver. Como en el nodo N-03 no se especifica nada, en principio pueden ver cualquier dato de este nodo.

US-02	Usuario de museos
Descripción	Son los usuarios de los museos que se van a encargar de gestionar los datos de museos de los bienes
Requisitos funcionales	• RF-03 • RF-04
Nodos	• N-01 - RI-01.Código - RI-01.Denominación - RI-01.Provincia - RI-01.Municipio - RI-01.Inmueble - RI-01.Otra denominación • N-03

Tabla 14: Tabla para la descripción de los usuarios de los museos

Por último, tenemos los usuarios que sólo pueden realizar consultas en el sistema. Como vemos en la tabla 15, sólo pueden ver algunos datos de identificación y las imágenes del nodo de descripción. Con respecto a los requisitos funcionales vemos que el único en el que puede tomar parte como actor es el que le permite consulta (RF-04).

US-03	Usuario de consulta
Descripción	Son aquellos usuarios que sólo pueden consultar los datos del bien
Requisitos funcionales	RF-04
Nodos	• N-01 - RI-01.Código - RI-01.Denominación - RI-01.Provincia - RI-01.Municipio - RI-01.Inmueble • N-02 - RI-01.Imágenes

Tabla 15: Tabla para la descripción de los usuarios consultores

Los requisitos de interfaz abstracta van a definir cada uno de ellos un grafo dirigido, que denominaremos *grafo de interfaz abstracta*, en el que los nodos van a coincidir con los nodos del grafo que representa las posibilidades de navegación o van a ser una parte de éstos. Las aristas de este grafo, van a ser un subconjunto de las aristas del grafo de navegación. Cada unsuario definido tendrá un grafo de interfaz abstracta. Por ejemplo, para nuestro usuario Us-03 el grafo sería el que se muestra en la figura 8.

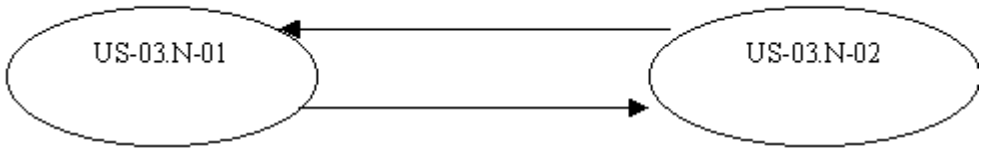

Figura 8: Grafo de interfaz abstracta para el usuario US-03

En este grafo, habrá que tener en cuenta que la visión de los nodos será la indicada en la plantilla de interfaz abstracta. Así la nomenclatura US-03.N-01 significa que es la visión que del nodo N-01 tiene el usuario US-03 y que se indica en la plantilla de US-01 (Tabla 15).

OTROS REQUISITOS

En el grupo de otros requisitos se van a englobar aquellos otras necesidades que debe cubrir el sistema y que no hemos contemplado hasta ahora. Normalmente estos requisitos son de carácter técnico o legal y no se suelen tener en cuenta hasta la fase de diseño. Para definir estos requisitos para nuestro ejemplo, vamos a usar la plnatilla propuesta en [Durán 1999]

RNF-01	Comunicaciones
Descripción	La aplicación será mostrada a través de internet

Tabla 16: Plantilla para recoger los requisitos de comunicaciones

RNF-01	Fiabilidad
Descripción	La aplicación será visible desde cualquier punto de la red, por lo que debe estar preparada para un número ilimitado de usuarios

Tabla 17: Plantilla para recoger los requisitos de comunicaciones

Como resumen final al flujo de especificación de requisitos decir que, aunque las técnicas y actividades propuestas por las metodologías de desarrollo de aplicaciones clásicas, se pueden aplicar a estas bibliotecas, es necesario ampliar el catálogo de requisitos de manera que se recojan nuevos requisitos que es necesario capturar cuando se plantea el desarrollo de un sistema para el tratamiento de bibliotecas digitales. De esta forma, el catálogo de requisitos estaría compuesto por:

- Requisitos de almacenamiento de información
- Requisitos funcionales
- Requisitos de navegación
- Requisitos de interfaz abstracta
- Otros requisitos

5.3 Análisis

El flujo de trabajo de análisis de una aplicación basada en bibliotecas digitales puede asemejarse mucho al flujo de análisis de una aplicación clásica. Esencialmente, las diferencias más notables entre una aplicación clásica y una aplicación basada en bibliotecas digitales son tres, como ya hemos enunciado: los diferentes medios de almacenamiento de información, la importancia de la interfaz y los aspectos navegacionales. Estos tres aspectos deben contemplarse en la etapa de diseño, quedando para la etapa de análisis los conceptos propios de las aplicaciones clásicas, por lo que el análisis de ambos tipos de aplicaciones puede realizarse de forma similar.

De esta forma, en análisis debemos conseguir un modelo de clases que represente al sistema. Este modelo irá acompañado por un modelo dinámico cuando resulte necesario, así como por una estructuración en paquetes cuando su complejidad sea alta. Además, en análisis se hará un refinamiento de los casos de uso para concretizarlos y asignar responsabilidades y participaciones de las clases de análisis.

De forma esquematizada, el flujo de trabajo de análisis debería asumir las siguientes tareas:

1- Realizar una descripción inicial del problema.

2- Construir un modelo de análisis inicial.

3- Agrupar las clases del modelo en paquetes y establecer relaciones entre ellos.

4- Realizar el refinamiento de los casos de uso para concretizarlos y asignar responsabilidades y participación en ellos de las clases de análisis.

Así, y basándonos en estas ideas, podríamos obtener un modelo estático para nuestro ejemplo de los bienes muebles patrimoniales, tal y como se muestra en la figura 9.

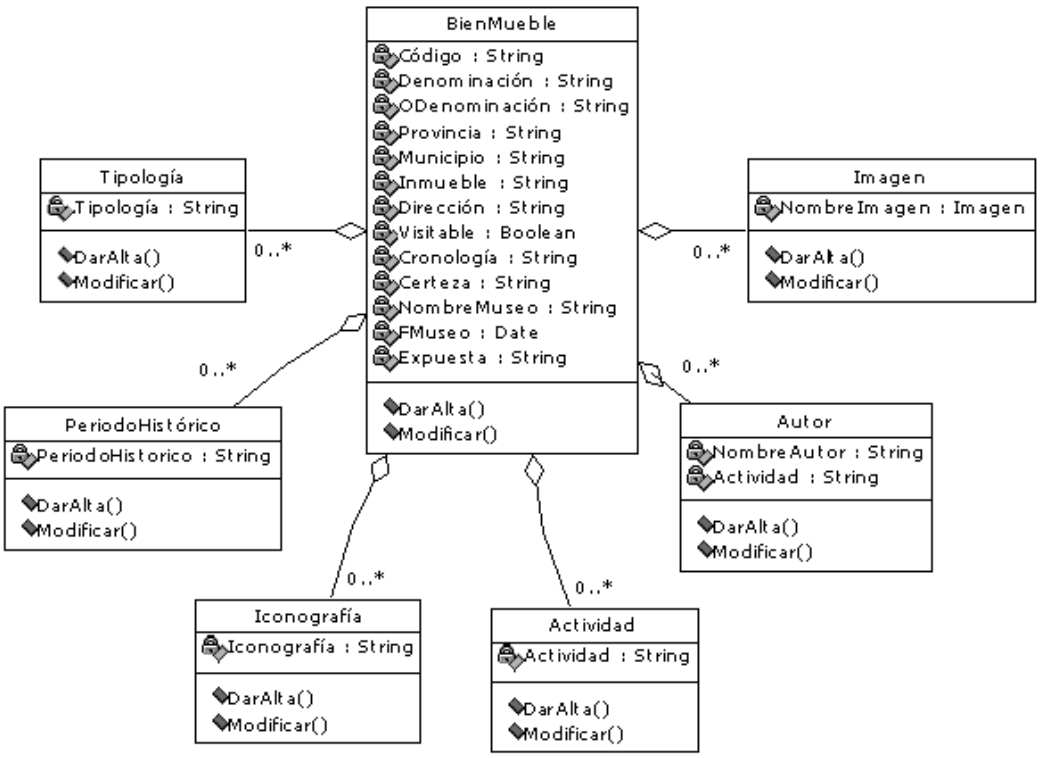

Figura 9: Modelo estático de análisis de la biblioteca digital de bienes muebles

Evidentemente habría que acompañar a este modelo con un diccionario de datos. En este caso no lo vemos necesario por la simplicidad del sistema y por el detalle que ya hemos dado de cada uno de estos atributos. También habría que agrupar las clases en paquetes y establecer las relaciones entre ellos, pero por la sencillez del problema, esto no es necesario para nuestro ejemplo.

Con respecto al modelo dinámico, en principio no aparece ninguna clase que tenga asociada un máquina de estados lo suficientemente relevante como para recogerla.

De la misma forma, y ya que en el flujo de especificación de requisitos no hemos entrado a detallar cada caso de uso, vamos a saltarnos este paso en análisis.

Con esto, ya tendríamos enmarcada lo que sería un posible flujo de análisis para aplicaciones basadas en bibliotecas digitales. Sería necesario, sin embargo, establecer una norma que sirviera como marco de referencia y que detallara cada una de las tareas a seguir dentro del análisis de un sistema para el tratamiento de bibliotecas digitales.

5.4 Diseño

En el flujo de trabajo de diseño se parte del modelo de clases de análisis conseguido en el flujo anterior, así como del análisis hecho a los casos de uso y de la agrupación en paquetes de análisis. A partir de estos productos se realiza el diseño de la arquitectura del sistema, hacer un diseño de los casos de uso, conseguir un modelo de clases de diseño y realizar una división del sistema en subsistemas. Pero además, hay que recoger y diseñar los aspectos de navegación y de interfaz de usuario.

En principio, vamos a subdividir la fase de diseño en tres subflujos:

- Diseño básico
- Diseño navegacional
- Diseño de interfaz abstracta

Diseño básico: Este subflujo se asemeja bastante a lo que sería el flujo de diseño del Proceso Unificado y a la fase de diseño conceptual de OOHDM. Por un lado, en la fase de diseño básico debemos realizar el diseño de la arquitectura del sistema. Para ello, hay que tener en cuenta que los soportes de los sistemas para el tratamiento de bibliotecas digitales se caracterizan por estar dispersos en la red, por estar implantados en sistemas heterogéneos y por la ausencia de estándares para el acceso a los mismos. Por ello, las propuestas de arquitecturas que se den en el marco de las bibliotecas digitales deben contemplar los siguientes aspectos:

(1) Almacenamiento

(2) Clasificación

(3) Interfaces para la presentación de resultados

(4) Distribución del contenido de la biblioteca al usuario final

(5) Administración y control de acceso.

En principio, el diseño de la arquitectura no debe verse afectado por aspectos de navegación o interfaz, que serán objetivos de los subflujos siguientes.

En el diseño básico también se afrontará el diseño de los casos de uso. Durante la especificación de requisitos y en el análisis se han establecido los casos de uso que representan los requisitos funcionales de la aplicación. A la hora de diseñar estos casos

de uso es conveniente tener en cuenta los métodos de diseño típicos de la Ingeniería del Software: patrones de diseño, técnicas para recoger los aspectos de concurrencia y persistencia, etc. En el diseño de estos casos de uso, hay que hacer un hincapié especial en algunos aspectos muy importantes para el entorno de las bibliotecas digitales. Así cuando diseñamos estos casos de uso debemos identificar la concurrencia inherente en el sistema. Esta tarea en el entorno de las bibliotecas digitales es esencial, puesto que los sistemas que gestionan a las bibliotecas digitales deben controlar la concurrencia. En nuestro sistema, podemos tener a la vez personas realizando modificaciones y consultas al mismo tiempo, sin que tengamos la posibilidad de saber el número de usuario puesto que al distribuirse por Internet, esto es imposible.

A la hora de estudiar la concurrencia, hay también que identificar los recursos globales y determinar los mecanismos para controlar el acceso a los mismo. Partiendo del hecho de que las aplicaciones basadas en bibliotecas suelen distribuirse por internet y que, en principio, a ellas puede acceder cualquiera, es necesario recortar y/o controlar a los usuarios que pueden usarlas.

Para nuestro ejemplo, y partiendo de los tres tipos de usuarios que definimos en la especificación de requisitos de interfaz abstracta, una posible propuesta de solución sería que los usuarios que pueden dar de alta o modificar cualquier dato (US-01) y aquellos que pueden modificar sólo los datos museográficos (US-02), accedieran al sistema a través de claves y los usuarios que accedan al sistema sin clave sólo podrán usar la función de consulta (US-03).

Cuando diseñamos los casos de uso y conocemos las necesidades de almacenamiento que hay que tener, es hora de seleccionar la estrategia básica para implementar los almacenes de datos en términos de estructuras de datos, archivos y bases de datos. Esta tarea adquiere una importante relevancia en las aplicaciones basadas en bibliotecas digitales. Ya hemos hablado de los diferentes tipos de datos que las bibliotecas digitales manejan: imágenes, sonidos, documentos, animaciones, etc. Es necesario pues, buscar sistemas gestores de bases de datos y soportes que nos permitan un almacenamiento y un tratamiento eficiente y seguro de todos los tipos de datos con los que vamos a trabajar. Debido a la multiplicidad de medios que se usan, no siempre es posible encontrar un sistema capaz de dar soporte a todos ellos. Las bases de datos de ultima generación que permiten almacenar y recuperar imágenes o sonido de la misma forma que se recuperan los tipos básicos como los texto o booleanos, están sólo naciendo. Por ello, aquí podemos proponer múltiples sistemas de almacenamiento que den soporte a nuestras necesidades. Por ejemplo, para la biblioteca de bienes muebles, vamos a proponer para todos los datos el uso de una base de datos relacional clásica y las imágenes las vamos a mantener almacenadas en una zona de memoria del servidor, de manera que en la base de datos tengamos una referencia a la ruta donde podemos encontrar las imágenes asociadas a un bien. Este diseño, que podría parecer un poco arcaico es, sin duda el que más se asume cuando hay que diseñar bases de datos con datos gráficos, debido a que es el método más barato y sencillo de diseñar.

En relación a la multiplicidad de medios, no solo hay que estudiar cómo se van a almacenar, hay que estudiar también las necesidades de consulta de la información. Esta tarea está muy relacionada con la anterior en el sentido de que las aplicaciones basadas en bibliotecas digitales cada vez requieren sistemas de consulta más elaborados y complejos. La complejidad de estos sistemas de búsqueda no viene dada exclusivamente por el hecho de que los volúmenes de la información sean muy grandes.

La complejidad también aparece en la idea de que las búsquedas sobre información no textual cada vez son más necesarias. Así, en el entorno de nuestra aplicación podríamos pensar en buscar todas las imágenes de los bienes que tengan un crucifijo en alguno de sus adornos. Pero en el entorno de los bienes muebles, así como en el de las bibliotecas digitales en general, aparecen otras consultas también complejas que pudieran permitir al usuario hacer preguntas imprecisas. Por ejemplo, el hecho de detectar los bienes muebles desarrollados en un entorno del siglo XVIII y que se encuentren en yacimientos próximos a Sevilla.

Por último, también en el diseño de los casos de uso, habrá que estudiar las condiciones de contorno. En el entorno de internet, que es el que nos ocupa, es esencial el contemplar las condiciones de contorno: qué ocurre al iniciarse la aplicación y en su caso cómo se realiza la conexión inicial al servidor; qué ocurre cuando finaliza la aplicación y la desconexión del servidor; y sobre todo que ocurre cuando se producen errores. En las aplicaciones de internet es necesario contemplar esos errores que se producen cuando hay caídas de sistemas de comunicaciones. Pero cuando estamos tratando con aplicaciones que se ejecutan en equipos diferentes a través de internet es necesario hacer un estudio exhaustivo de otro tipo de errores, que, hoy por hoy aún son un problema. Al distribuirse la aplicación por internet, no podemos controlar aspectos como el navegador que va a usar el usuario y las utilidades que éste ofrece; las capacidades del equipo que va a conectarse o la velocidad de la conexión. Por esta razón, es necesario controlar estos tipos de errores en la fase de diseño, sobretodo cuando trabajamos con animaciones o datos en los que las características temporales y de sincronización tengan una vital importancia. Así, para nuestra aplicación habría que controlar, por ejemplo, que las imágenes asociadas a los bienes se visualicen correctamente o en su caso, que si el sistema cliente no puede visualizar las imágenes, que se muestre un mensaje indicando que en ese lugar habría una imagen.

En el diseño básico también se planteará organizar el sistema en subsistemas. Debido a la complejidad funcional que suelen tener los sistemas basados en bibliotecas digitales, es bastante aconsejable el dividir el problema inicial en subproblemas más sencillos, usando así una técnica de divide y vencerás. Para nuestro sistema vamos a plantearnos dos subsistemas principales: el que permite realizar consultas y el que permite realizar altas y modificaciones. Cada uno de ellos tendrá a su vez dos tareas fundamentales. Una que controle los permisos del usuario y otro que controle lo que son las tareas de alta, consulta y modificación propiamente dichas. En modo de diagrama

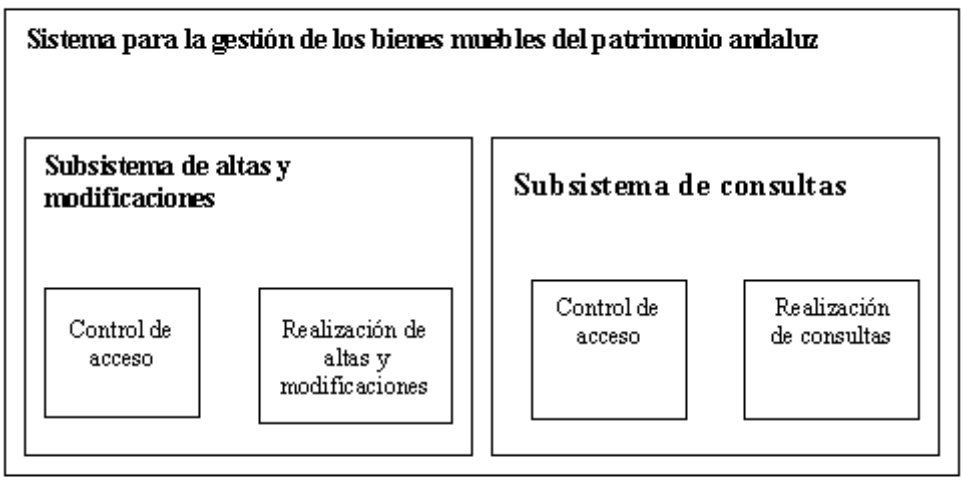

Figura 10: División en subsistemas de la biblioteca digital de los bienes muebles

quedaría como se muestra en la figura 10.

Una vez definidos los subsistemas, habrá que asignar cada uno de ellos a procesadores y a tareas. Otra de las característica de las aplicaciones basadas en bibliotecas digitales es que suelen ser aplicaciones distribuidas, por lo que será necesario establecer dónde se va a realizar cada tarea. En nuestro caso, podríamos pensar en un almacén de datos, localizado en el servidor central y una aplicación que se ejecuta en equipos remotos y que se encargan de controlar las peticiones de los usuarios. Una vez filtradas y comprobadas las consultas y modificaciones, los equipos remotos pedirán al servidor central que realice la consulta, la modificación o el alta.

Por último, hay que plantearse el conseguir el modelo de objetos básico de diseño. De la misma forma que hemos conseguido un modelo que represente al sistema en la fase de análisis, vamos a partir de ese modelo de análisis y vamos a enriquecerlo para obtener el modelo de objetos de diseño. En este modelo, no será necesario aún introducir aspectos de navegación o de interfaz de usuario, puesto que esos quedaran para los subflujos posteriores.

Para conseguir este modelo, será necesario estudiar ciertos aspectos del sistema para:

- Obtener las operaciones para el modelo de objetos a partir de los demás modelos. Esta tarea va ser el primer paso para enriquecer el modelo de objetos obtenido en análisis. Estas operaciones se van a obtener estudiando el modelo dinámico del análisis y los casos de uso. En nuestro ejemplo, en principio, no habría que añadir nuevas operaciones

- Empaquetar las clases y las asociaciones en paquetes. De la misma forma que en análisis agrupábamos las clases en paquetes que nos permitieran estructurar el sistema, en diseño es aconsejable agrupar las clases en paquetes y establecer relaciones entre ellos. Esta tarea va a facilitar la compresión del modelo y va a hacer más sencillo el trabajo del implementador.

- Retocar el modelo añadiendo los objetos que se estimen necesario para representar aquellos atributos del modelo de análisis que se van a tratar como clases.

Para nuestro ejemplo, el modelo no quedaría modificado al aplicar esta fase, quedando como se mostró en la figura 7. Esto es lógico de pensar si analizamos el hecho de que no tenemos máquinas de estado asociadas que puedan añadir operaciones u atributos que haya que representar como objetos.

Diseño Navegacional: En OOHDM la navegación se considera como un paso crítico en el diseño de las aplicaciones hipermedia. Debido a la similitud que existe entre estas aplicaciones y las basadas en bibliotecas digitales, la navegación es un punto importante y complejo que hay que contemplar de forma cuidadosa en su desarrollo. En OOHDM la navegación se construye como una vista del modelo conceptual (modelo básico en la

propuesta de bibliotecas digitales). De esta forma, un modelo conceptual puede tener varios modelos de navegación. Cada modelo de navegación no es más que una vista subjetiva del modelo conceptual [Schwabe 1996]. Vamos a asumir pues, casi en su totalidad las recomendaciones que OOHDM hace para el desarrollo de aplicaciones multimedia en el diseño de los aspectos de navegación.

A la hora de realizar un modelo navegacional, hay que tener en cuenta:

1- Qué objetos del modelo básico van a ser navegables. Esto va a estar íntimamente relacionado con los nodos definidos en la especificación de requisitos navegacionales.

2- Qué tipo de relaciones y estructuras de composición hay entre estos objetos navegables.

3- De los objetos navegables, habrá que estudiar en qué forma se mostrarán dependiendo del contexto en el que nos movamos.

4- Las conexiones entre los distintos objetos y las posibilidades de navegación de un objeto a otro. Esto va a estar bastante relacionado con los enlaces entre nodos que se detallan en la especificación de los requisitos navegacionales.

Proponemos dos modelos para recoger los aspectos de navegación: El esquema de Clases Navegacionales y el diagrama de Contexto Navegacional.

El esquema de Clases Navegacionales se obtiene a partir del modelo de clases de diseño obtenido en el Diseño básico. El modelo básico se va a modificar de manera que se van a obtener nuevas clases que pueden ser *nodos*, *enlaces* o *clases índice*. De esta forma, una clase del modelo básico puede originar una o varias clases del modelo navegacional. El modelo de Clases Navegacionales, va a estar compuesto por este nuevo modelo de clases que contiene índices, nodos y enlaces.

Antes de continuar es necesario hacer una definición más concreta de lo que son clases nodo, clases enlace y clases índice.

Una clase nodo es un tipo especial de clase que muestra una agrupación de información bajo un determinado criterio. Nos va a permitir recoger el concepto de nodo definido en el flujo de trabajo de especificación de requisitos. Los atributos y operaciones de una clase nodo van a ser atributos y operaciones de clases del modelo básico. Una clase nodo puede contener atributos y operaciones de diferentes clases del modelo básico de la misma forma que una propiedad (atributo u operación) de una clase del modelo básico puede estar en diferentes nodos. No hay que confundir el concepto de clase nodo en diseño con el concepto de nodo navegacional en la especificación de requisitos. La clase nodo no es más que un tipo especial de clase que se define en base a las clases del modelo básico, mientas que el concepto de nodo en especificación nos sirve para agrupar la información recogida en los requisitos de almacenamiento de información bajo determinados criterios. Sin embargo, existe una relación entre estos conceptos. Cada uno de los nodos definidos en el catálogo de requisitos van a generar un nodo en el modelo navegacional.

Por su parte, los enlaces son otras clases especiales que van a definir un tipo. Mientras que los nodos son propios de cada aplicación, la definición de la clase tipo enlace es genérica y podemos verla en la figura 11.

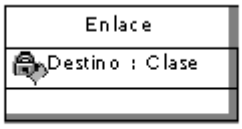

Figura 11: Definición de la clase enlace

El único atributo que tiene es el que indica la clase a la que se llega al ejecutar el enlace. Esta es una definición muy básica de la clase enlace pero pueden añadirse otros atributos u operaciones.

Definida esta clase, podemos declarar en los nodos enlaces a otros nodos recogiendo así las posibilidades de navegación que existen el sistema.

Por último, las clases índice son unas clases que sirven para representar menús, diccionarios, etc. en la navegación. Las clases índice suelen estar compuestas por bastantes atributos de tipo enlace que llevan de un nodo a otro.

Una vez definidos estos conceptos, vamos a aplicar esta idea a nuestro ejemplo. Para ello, recordemos que en la fase de especificación de requisitos hemos definido tres nodos: Datos de Identificación, Datos de Descripción y Datos Museográficos. De esta forma, obtendríamos el modelo de clases navegacionales que se muestra en la figura 12.

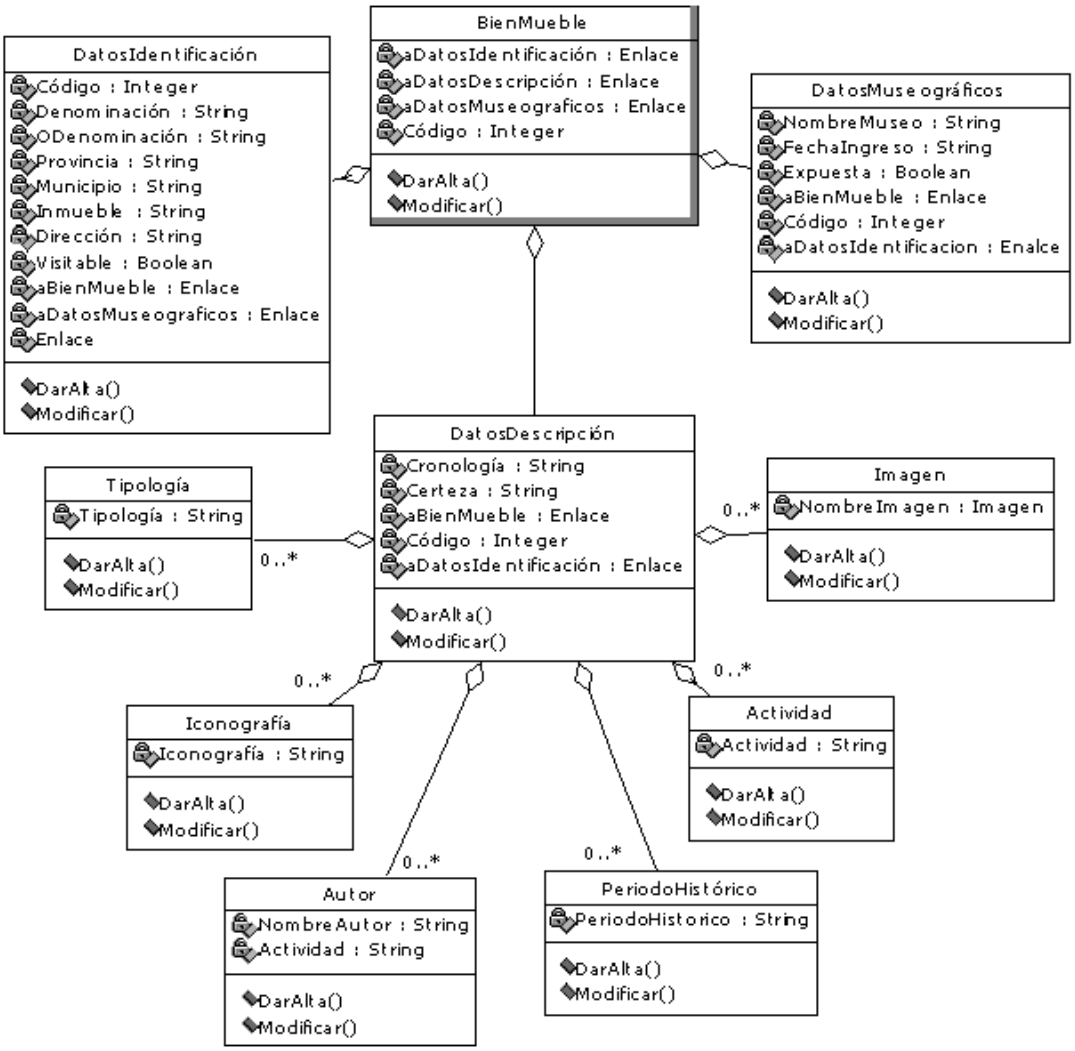

Figura 12: Modelo de clases navegacionales de la biblioteca digital de bienes muebles

Para que el gráfico quede lo suficientemente claro, hay que definir cada clase nodo y cada clase índice que surge en este modelo en base a las clases del modelo básico de diseño. Para ello, vamos a basarnos en la propuesta de [Rossi 1996]. En la figura 13 vemos la definición de la clase índice BienMueble del modelo anterior. Mediante la cláusula FROM se indica de que clase o clases del modelo básico se va a tomar información. BM indicará un objeto de la clase BienMueble del modelo básico. A continuación se indica que el atributo código de la clase índice viene del atributo código del objeto BM declarado. Por su parte, encontramos luego tres enlaces: uno que va al nodo DatosIdentificación, otro al nodo de DatosDescripción y otro que va al nodo de DatosMuseográficos.

```
INDICE BienMueble [FROM BienMueble:BM]

Código: BM.Código
aDatosIdentificación: Enlace (DatosIdentificación)
aDatosDescripción: Enlace (DatosDescripción)
aDatosMuseográfico: Enlace (DatosMuseograficos)
```

Figura 13: Definición de la clase índice BienMueble del modelo navegacional de la biblioteca digital de bienes muebles

En la figura 14, vemos la definición de uno de los nodos. Igualmente la cláusula FROM nos permite definir de qué clase del modelo básico vamos a tomar información, la clase BienMueble, y declaramos una instancia de ésta, BM. Vemos cómo se hace referencia a los atributos de esta clase que se van a visualizar desde este nodo. Además existen posibilidades de navegación sólo al nodo índice principal.

```
NODO DatosIdentificación [FROM BienMueble:BM]

Código: BM.Código
Denominación :BM.Denominación
ODenominación :BM.ODenominación
Dirección: BM.Dirección
Provincia: BM.Provincia
Municipio: BM.Municipio
Inmueble: BM.Inmueble
ABienMueble: Enlace (DatosIdentificación.BienMueble)
aDatosDescripción: Enlace (DatosDescripción)
aDatosMuseográfico: Enlace (DatosMuseograficos)
```

Figura 14: Definición de la clase nodo DatosIdentificación del modelo navegacional de la biblioteca digital de bienes muebles

Como vemos, en el modelo navegacional aparecen tres nodos que coinciden con los nodos que se detectaron en la especificación de requisitos. Además las posibilidades de navegación coinciden con las definidas en el primer flujo de trabajo.

Con respecto al otro modelo que debemos desarrollar para completar la fase que propone OOHDM es el modelo de Contexto Navegacional. Un contexto navegacional no es más que un conjunto de nodos, enlaces, clases índice y otros contextos navegacionales, que nos permiten especificar las posibilidades de navegación que tenemos. Así, podríamos tener una visión de los bienes ordenados por provincias, ordenados por inmuebles, ordenados por autores, los bienes de un autor concreto, los que pertenecen a una tipología, etc. Realmente el modelo de Contexto Navegacional nos muestra las distintas posibilidades de acceso a los datos que tenemos. Esta idea, cuando

se trata de una aplicación hipermedia en la que las formas de acceso y las consultas posibles están restringidas, es sencilla y útil. Sin embargo, en el ámbito de las bibliotecas digitales, los contextos navegacionales pueden ser muy complejos, por lo que habría que buscar nuevas alternativas de representación.

A pesar de qué con esto el modelo navegacional quedaría definido, éste será enriquecido con nuevos atributos, operaciones e incluso clases y asociaciones cuando se defina la interfaz abstracta en el subflujo siguiente. Como veremos a continuación.

Diseño de Interfaz Abstracta: una vez que se tienen diseñado los aspectos de navegación, pasamos a la especificación de los aspectos de interfaz. El hecho de separar los aspectos de navegación de los aspectos de interfaz permite diseñar distintos interfaces para un mismo modelo navegacional.

Para el Diseño de la Interfaz Abstracta se propone el uso de las vistas abstractas de datos [Cowan 1995] (ADVs). Las ADVs son objetos que tienen un estado y un interfaz. Son objetos abstractos que sólo representan el estado y la interfaz, sin hacer referencia a la implementación. Mediante el uso de los ADVs representaremos la estructura de la interfaz, cada ADV representa la interfaz que se muestra al usuario de un objeto de navegación, de una parte de un objeto del modelo navegacional o de un conjunto de ellos.

Sin embargo, entre los ADVs y las clases del modelo navegacional que tengamos en nuestro sistema del subflujo anterior, se establecen relaciones que hay que recoger. Para expresar estas relaciones se usará el diagrama de configuración.

Si a todo esto añadimos que debemos recoger el dinamismo que hace cambiar las interfaces, es decir, como cada ADV responde a los eventos externos, aparece la necesidad de usar una nueva técnica que nos permita recoger el dinamismo del sistema de interfaces. Esta técnica la constituyen los diagramas de estado.

Hasta aquí la propuesta es bastante similar a la realizada por OOHDM. Sin embargo se va añadir una tarea más. En este subflujo, se hace uso del patrón observador de manera que los ADVs hacen el papel de observador y las clases navegacionales del modelo navegacional, juegan el rol de observado. En el entorno de los ADVs las clases observadas se conocen como ADO (Abstract Data Object). Por otro lado, y debido al dinamismo que ofrece la interfaz, hemos visto que se propone un modelo que capte el dinamismo de los ADVs. Sin embargo, no se dice nada del dinamismo que pueden tener los ADO (o lo que es lo mismo, las clases navegacionales). El dinamismo de los ADVs suele venir dado por los eventos externos provocados, normalmente, por la acción del usuario. Pero es lógico pensar que estos eventos pueden dar origen a eventos que cambien el estado de las clases observadas y que deberían ser recogidos en el diseño. Por ello, vamos a proponer que, la fase de Diseño Navegacional sea ampliada y se acompañe a las clases, que se estimen oportunas, del modelo de clases navegacionales de una máquina de estados que refleje su dinamismo.

De esta forma, en la fase de diseño navegacional se debe:

- Diseñar los ADVs
- Diseñar los diagramas de configuración

- Diseñar los diagramas de estado para los ADVs
- Enriquecer el modelo de clases navegacionales

Estas cuatro tareas habrá que realizarlas para cada uno de los usuarios que se han definido en la especificación de requisitos de interfaz abstracta, puesto que cada uno de ellos va a tener una visión diferente.

Un punto abierto al estudio en el diseño navegacional es la necesidad de normalizar la nomenclatura usada para representar a los ADVs. De esta forma, podríamos abstraernos y pensar en los elementos que podemos encontrarnos en un ADV. En resumen podríamos asumir tres:

- Datos de las clases navegacionales
- Enlaces a otros ADVs
- Botones de función (Cuando hablamos de "botones" nos estamos refiriendo a la funcionalidad que podemos obtener desde el ADV en concreto que estamos usando)

Evidentemente, en una pantalla final nos encontraremos más elementos. Sin embargo, a este nivel son básicamente éstos los que nos interesa. Usando las propuestas de [Cowan 1995] , lo que serían los enlaces y los botones se muestran de la misma forma (a través de un cuadrado) y se diferencian uno de otro por su nombre. De ahí la necesidad de normalizar representaciones distintas y/o proponer la necesidad de elaborar un diccionario de datos que describa brevemente cada uno de estos elementos.

Vamos a ver un ejemplo de esto para los bienes muebles. Nos centraremos en el usuario que definimos comos US-01 por ser el que más visibilidad tenía de los nodos. Definiremos para el tres ADVs: uno que muestre los datos de identificación, otro que muestre los datos de descripción y, por último, el que mostrará los datos museográficos. Estos ADVs se traducen al modelo como clases observadoras de las clases BienesMuebles, DatosIdentificación, DatosDescripción y DatosMuseografícos del modelo Navegacional.

El primer ADV, se muestra en la figura 15 y en el usuario podrá ver todos los datos de identificación del bien.

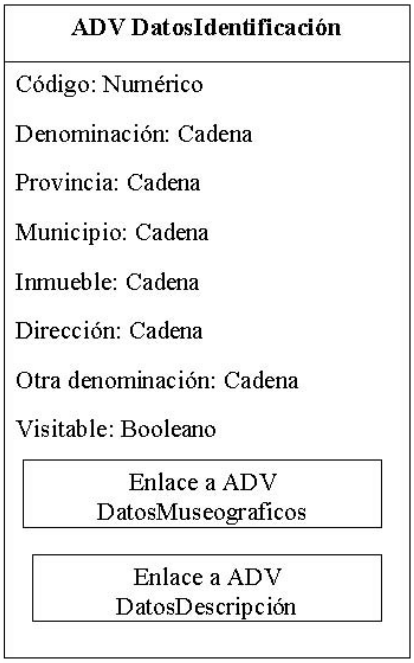

Figura 15: ADV para los datos de identificación

En la figura 16, vemos la visión que de los datos de descripción del bien tendría el usuario, así como los enlaces que éste va a tener.

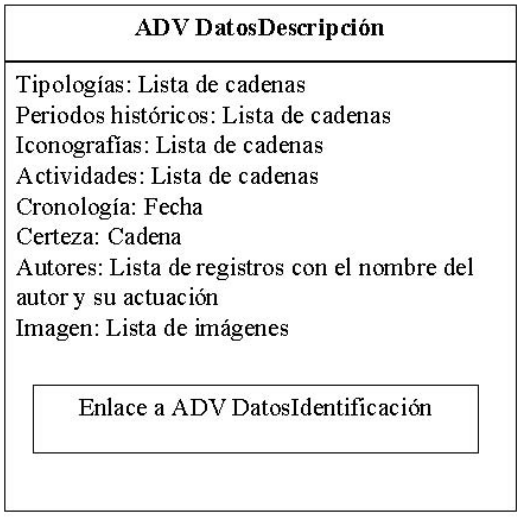

Figura 16: ADV para los datos de descripción

Por último, en la figura 17 vemos el ADV que permite definir la visión que el usuario tendrá de los datos del museo.

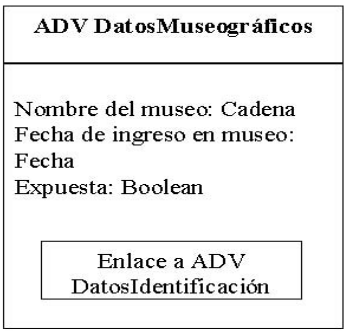

Figura 17: ADV para los datos museográficos

No parece muy necesario el acompañar a estos ADVs de los diccionarios correspondientes por la sencillez y la claridad del ejemplo. Sin embargo, cuando la complejidad aumenta esto puede clarificar bastante el diseño.

Con respecto a las relaciones que se establecen entre ellas y las clases navegacionales son bastante sencillas, las mostramos en la figura 18. Esta figura es lo que hemos llamado diagrama de configuración y permite determinar como se relacionan los ADV con las clases navegacionales.

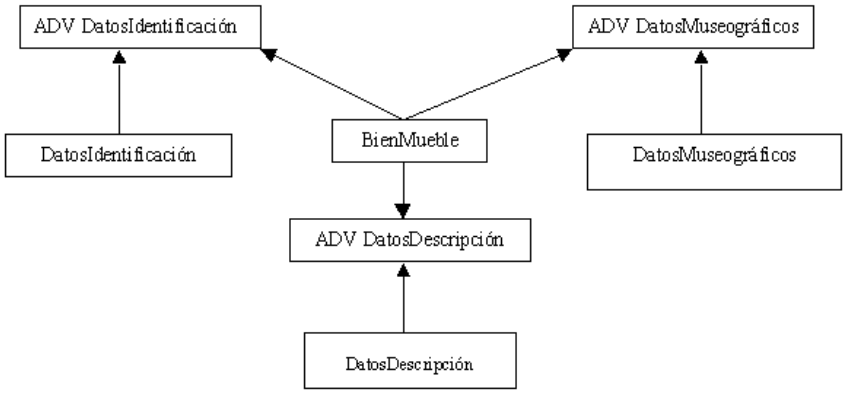

Figura 18: Diagrama de configuración para la biblioteca digital de los bienes muebles

Esto, acompañado de las máquinas de estados que representaran cómo cada uno de estos ADVs reaccionan ante los eventos, serían los resultados a obtener en el diseño de interfaz abstracta.

Tras esto, habría que revisar el modelo navegacional obtenido en el diseño navegacional y su modelo dinámico, para ver cómo se vería afectado por las decisiones tomadas.

En resumen, tras realizar los tres subflujos de diseño debe obtenerse como resultado final de diseño:

- El modelo de clases navegacionales y su modelo dinámico obtenido en el diseño navegacional y enriquecido con los eventos, propiedades, clases y asociaciones que se hayan determinado necesarias al realizar el diseño de interfaz abstracta.

- Los ADVs de cada tipo de usuario, los diagramas de configuración y la máquina de estados para cada uno de estos ADVs.

5.5 Implementación

En este flujo de trabajo, la tarea fundamental es conseguir un programa ejecutable que implemente a los modelos conseguidos en diseño. En la fase de implementación se deben implementar la arquitectura, las clases y los subsistemas, integrándolos para conseguir el sistema final.

5.6 Pruebas

Uno de los aspectos no contemplados en un elevado número de propuestas metodológicas es el que tiene relación con la definición de las pruebas que se han de realizar a un sistema para garantizar que el mismo responde a la definición de requisitos que para él se hizo. Aunque la definición concreta de las pruebas que se deben aplicar a un sistema deberá concretarse en el momento de concretar su diseño, toda propuesta metodológica debe definir las posibles estrategias a seguir, las técnicas a utilizar para realizarlas y el momento de hacer uso de cada una de ellas. En el marco de la propuesta para el desarrollo de aplicaciones basadas en bibliotecas digitales, la fase de pruebas debe elaborar, diseñar e implementar el plan de pruebas. Este plan de pruebas no solo debe recoger las pruebas a realizar, además debe indicar el orden de realización. Una vez realizado, hay que ejecutar el plan de pruebas y elaborar una memoria de resultados del mismo.

6 Conclusiones

En este documento se ha realizado una propuesta de aproximación metodológica al desarrollo de sistemas para el tratamiento de bibliotecas digitales. Para ello, se han enunciado las propuestas realizadas por el Proceso Unificado y la de OOHDM, con la idea de analizar en qué medida se pueden adoptar y en qué puntos resultan ineficientes para el desarrollo de estos sistemas. Partiendo de esto se ha hecho una propuesta de ciclo de vida y se han analizados los flujos de trabajo a realizar, así como los productos a obtener en cada uno de ellos y las subfases que los componen.

Como trabajo futuro, sería interesante concretizar más en las actividades que hay que realizar en cada uno de estos flujos y las tareas que compondrían a cada una de estas. Este trabajo se vería completado con la aplicación de las propuestas que se realicen en sistemas concretos reales.

7 Puntos abiertos

De forma concreta, en este documento se han analizado las siguientes necesidades:

- Ampliar las normas de recolección de requisitos propuestas para la orientación a objetos de manera que se puedan recoger aspectos de navegación e interfaz.

- Estudiar una arquitectura génerica que pudiera cubrir las necesidades que se plantean en las bibliotecas digitales.

- Con respecto al Diseño Navegacional, sería conveniente realizar el estudio de alternativas de diseño que sean más sencilla que las propuestas por OOHDM para los contextos navegacionales.

- Dentro del Diseño Abstracto de Datos habría que cubrir varios aspectos. Por ejemplo, sería conveniente normalizar la nomenclatura usada para representar los ADVs, como ya enunciamos. También sería conveniente estudiar técnicas que permiten añadir los aspectos de interfaz al modelo obtenido en el Diseño Navegacional de manera metódica, por ejemplo, usando patrones de diseño.

- Por último, en la implementación, resulta interesante el estudio de técnicas que permiten pasar de los modelos obtenidos en diseño al código de una manera ágil y metódica

8 Bibliografía

[Booch 1999] G. Booch, J. Rumbaugh, I. Jacobson. "Unified Modeling Language User Guide". Ed. Addison-Wesley, 1999.

[Caneiro 1994] L.M.F. Caneiro, M.H.Coffin, D.D. Coewan, C.J.P.L. Lucena, "ADVcHARTS A Visual Formalism for Higgly Interactive Systems", in M.D. Harrison, C.Johnson, eds, Software Engineering in Human-Computer Interaction, Cambridfe University Press, 1994.

[Coleman 1992] D.Coleman, F.Hayes, S.Bear, "Introducing Objectcharts or How to use Statecarts in Object-Oriented Design", IEEE Transaction on software Engineering, 18(1), January, 1992.

[Cowan 1995] D.D.Cowan, D.J.P.Lucena, "Abstract Data Views. An Interface Specification Concept to Enyhance Desingn for Reuse", IEEE Transactions on Software Engineering. 18(1), 9-18, January 1995.

[Durán 1999] A. Durán, B. Bernáldez. "Metodología para la Elicitación de Requisitos de Sistemas Software". Departamento de Lenguajes y Sistemas Informáticos. Universidad de Sevilla, 1999.

[Escalona, 2000] M.J. Escalona, M.Mejías, J.Torres. "Aproximación metodológica al desarrollo de sistemas para el tratamiento de bibliotecas digitales". V Jornadas de Ingeniería del Software. Noviembre 2000.

[Garzoto, 1993] Garzoto, D.Schwae and P.Paolini "HDM-A Model Based Approach to Hypermedia Aplication Design" ACM Trnasactions on Information System, 11 (1), Jan 1993, pp 1-26.

[Greco 1998] D. Greco, M. Eskelinen, C. Funckhouser, M.Luesebrink, J.Rosenberg "Actual & Potential Hypertext & Hypermedia", ACM Digital Library. June 20-24, 1998.

[Harrison 1993] W.Harrison, H.Ossher, "Subjects-Oriented Programming (a critique of pure objects)", In Proceeding of the Conference on Object-Oriented Programming Systems, Languajes and Applications (OOPSLA'93), pp. 411-428, Washington, September 1993.

[Hoch 1999] M.Hoch, D. Schwabe. "Group`interaction in a surround screen environment". Inst. for Visual Media ZKM, Karlsruhe, Germany. IEEE Computer Animation, 1999

[Izakowirz 1995] Izakowitz, E.Sthorz, P. Balasubramaniam: "RMM:A metodology for structured hypermedia design". Comm. Of ACM, October 1995, pp.34-35

[Jacobson 1999] I. Jacobson, G. Booch, J. Rumbaugh, "The Unifed Software Development Process", Ed. Addison-Wesley, 1999.

[Lange 1995] D.B. Lange "An Object-Oriented Design Approach for Devloping Hipermedia Information Systems", Research Report RT00112, IBM Research, Tokyo Research Laboratory, Japón, 1995.

[Matínez 1997] J.M. Martínez, J.R.Hilera, J. Martínez, J.A. Gutiérrrez, "Orientación a Objetos en la Documentación Hipermedia", Universidad de Alcalá de Henares, Departamento de Ciencias de la Computación.

[Rossi 1996] G. Rossi. "An Object Oriented Method for Designing Hipermedia Applications". PHD Thesis, Departamento de Informática, PUC-Rio, Brazil, 1996.

[Rossi 1997] G. Rossi, D. Schwabe, A. Garrido: "Design Reuse in Hipermedia Applications Development". Proceedings of ACM International Conference on Hypertext (Hypertext`97), Southampton, April 7-11, 1997. ACM Press.

[Rumbaugh 1999] J. Rumbaugh, I. Jacobson, G. Booch. "The Unified Modeling Language Reference Manual". Ed. Addison-Wesley, 1999.

[Rumbaugh 1991] J.Rumbaugh, "Modelado y Diseño Orientado a Objetos", Ed. Prentice Hall, 1991.

[Schwabe 1995] D.Schwabe, G.Rossi, "The Object-Oriented Hipermedia Design Model", Communications of the ACM, 38(8), August, 1995, pp.45-46

[Torres, 2000] M.J. Escalona, J.Torres, M.Mejías. "Aplicación de los sistemas de tratamiento de bibliotecas digitales al Sistema de Información del Patrimonio Histórico Andaluz". Boletín Trimestral del Instituto Andaluz de Patrimonio Histórico, Julio 2000.