2. Estado del Arte en Procesos de Desarrollo de Hipermedia.

2.1 Presentación del Problema

No hace mucho se afirmaba que “los trabajos sobre modelado de proceso son recientes y la amplitud de la agenda de investigación aun está siendo formulada” [Curtis et al 92]. Si bien se conocen trabajos previos sobre modelos de proceso como indicamos en la introducción (y que ampliaremos en el siguiente punto), en la comunidad de hipermedia lo podemos considerar como un tema nuevo.

Para entrar en contexto utilizaremos algunas declaraciones de investigadores acerca de la necesidad de contar con modelos de proceso y lo que pueden aportar en la práctica de la creación, evolución y mantenimiento de artefactos de software en general.

“Cuando un grupo de personas trabajan cooperativamente sobre un proyecto común, necesitan de alguna manera coordinar su trabajo. En tanto las tareas son simples frecuentemente se pueden realizar de un modo informal, sin embargo cuando se incrementa el número de personas y la complejidad de las actividades, es preciso contar con mecanismos más formales...” “Se necesita un proceso de software bien definido para proveer a las organizaciones con un marco consistente para realizar y mejorar su trabajo” [Humphrey et al 89]

“Un modelo de proceso declara y establece un orden para realizar las actividades...” “Un modelo de proceso es una colección de máximas, estrategias, actividades, métodos y tareas, las cuales se organizan para alcanzar un conjunto de metas y objetivos” [Goldberg et al 95]

Queda claro que un modelo de proceso no es un conjunto de definiciones desordenadas de actividades sino que debe ayudarnos a comprender qué es importante para lograr esas metas y objetivos. Avanzando con algunos conceptos de los autores antes citados, Goldberg et al afirman que es importante definir un modelo de proceso porque ayuda a responder cinco preguntas críticas respecto de:

- planificación (qué tenemos que hacer para alcanzar nuestro objetivo?)

- autoridad (cómo podemos influenciar lo que está sucediendo para llegar a donde deseamos?)
- predicción (hacia dónde estamos yendo?)
- valoración (dónde estamos parados en el proceso de desarrollo y por qué?)
- seguimiento (cómo se alcanzó un determinado resultado?)

Además es oportuno decir que las investigaciones en modelado de procesos de software soportan un amplio rango de metas y objetivos. Curtis et al. enumeran cinco categorías fundamentales, a saber:

- *para facilitar la comprensión y comunicación entre los agentes involucrados en el proceso (agentes humanos: usuarios de proceso, desarrolladores de proceso, coordinadores, investigadores).* Se requiere estandarización de procesos, una representación comprensible y uniforme.

- *para soportar mejoramiento de procesos.* Se requiere una estrategia clara de definición, análisis y descripción de procesos.

- *para soportar la administración de procesos.* Se debe contar con un proceso de desarrollo de un proyecto, bien definido y flexible.

- *como guía automatizada en la ejecución de un proceso.* Esto es, definir guías, sugerencias en línea, material de referencia para facilitar a los agentes la ejecución de las tareas manuales o interactivas[1]. Esta guía automatizada puede estar embebida en un ambiente de desarrollo de software.

- *como soporte a la ejecución automatizada.* Es decir, de algunas porciones del proceso que se pueden controlar y ejecutar por medio de una interpretación mecánica de un modelo a partir de un servicio de descripción de procesos.

Desafortunadamente, la mayoría de las aplicaciones hipermedia se están desarrollando por medio de una estrategia ad-hoc. La falta de procesos bien definidos que cubran las distintas fases y actividades puede hacer crecer, en este nuevo campo, el fantasma de la crisis del software. Pese a que hipermedia se ha tornado un área de investigación extremadamente activa

[1] En la sección 3.1 o en el glosario se pueden encontrar las definiciones de estos conceptos

realmente hay muy poca literatura específica sobre el tema de modelos de proceso de hipermedia. Una observación interesante es analizar cuántos papers se han venido publicando en congresos de reconocimiento académico como Hypertext 97, Hypertext 96 y eventos anteriores u otros relacionados: el lector no encontrará material específico. No es el caso por ejemplo, con metodologías como OOHDM o RMM en las que se han publicado varios trabajos. En estas metodologías subyacentemente se definen actividades del proceso de desarrollo, principalmente en actividades de diseño, como luego veremos.

Sin embargo, en la comunidad científica se está comenzando a considerar la necesidad de contar con un enfoque amplio de ciclo de vida de desarrollo de hipermedia (en Hypertext 97 se dictó un tutorial sobre Ingeniería de Hipermedia, y se presentó un poster sobre un modelo de proceso flexible para la construcción de artefactos de hipermedia [Olsina 97a]). Principalmente se está viendo la necesidad de contar con un enfoque ingenieril; esto es, el empleo disciplinado, cuantificable y sistemático de principios de Ingeniería de Software para la creación, evolución y mantenimiento de artefactos de software de hipermedia.

La presente tesis pretende realizar un aporte ingenieril al proponer un modelo de proceso flexible que se adecue a la construcción de artefactos de hipermedia.

2.2 Trabajos Relacionados y Aportes

En este punto analizamos algunas características de procesos de ciclo de vida tradicionales que son públicamente conocidos en la literatura de Ingeniería de Software y comentaremos qué cosas serán de utilidad para nuestro modelo de proceso flexible. Luego, analizaremos metodologías recientemente establecidas para el proceso de hipermedia. Al menos tres de estas metodologías utilizan constructores centrados en modelos, y explícita o implícitamente definen actividades y artefactos sobre todo en las etapas de diseño y construcción.

Un buen número de modelos de proceso han sido propuestos, entre ellos el modelo de cascada [Royce 70]; el modelo en espiral [Boehm 88]; el modelo recursivo/paralelo [Berard 93]; el modelo de fontana [Henderson et al 90] y variaciones sobre los mismos. Por ejemplo una adaptación al modelo de cascada, es el realizado por el Departamento de Defensa de EU, conocido como el estándar DoD-STD-2167A [DoD 88] donde prima un estilo centrado en la documentación.

Royce fue el primero en acuñar la frase modelo de cascada (waterfall model) para representar

una secuencia bien definida de etapas, establecidas con criterios de Ingeniería de Software. Ofrece una visión de alto nivel del ciclo de vida de software. Define las siguientes etapas o tareas principales: análisis de requerimientos, especificación, diseño, implementación, testeo, operación y uso. Las mismas se representan como una cascada. La salida de una tarea es la entrada de la próxima. Como consecuencia es una estrategia simple para planificar y controlar proyectos: favorece a la administración de los mismos. La programación es una secuencia de tareas; la ejecución es lineal; con la culminación de una tarea se genera y aprueba un artefacto final (documento, código, etc.). Este modelo fue bien recibido en las esferas gubernamentales dado que se puede realizar un seguimiento de los contratos pactados y pagar en función de los distribuíbles [Goldberg et al 95].

Si bien el mayor mérito fue introducir una visión ingenieril en el desarrollo de software en contraposición a estrategias ad hoc (citadas como "code and fix", "just do it", etc.), el modelo falla por su extremada simplificación de la realidad. Con frecuencia su comportamiento secuencial es contraproducente: es reconocido el costo creciente en la corrección de errores generados en etapas tempranas del proyecto y detectados en las etapas tardías. Además el modelo no reconoce la naturaleza evolucionaria del software. Entre algunas desventajas informadas por varios autores (y que será útil que el lector tenga presente como puntos de comparación para un proceso de hipermedia), se encuentran:

- define una visión estática de los requerimientos
- toma demasiado tiempo para mostrar resultados al usuario
- demora la detección y corrección de errores hasta etapas tardías
- no promueve el desarrollo participativo ni la retroalimentación experimental entre desarrolladores y usuarios
- no promueve la estrategia de prototipación
- no establece una estrategia de reuso de artefactos

Una de las motivaciones que tuvo Boehm con su modelo en espiral fue introducir análisis de riesgo, carente en modelos como el de cascada. A diferencia de éste, las etapas se definen cíclicamente representadas por una espiral que evoluciona desde el centro de un sistema de ejes cartesianos. Las cuatro etapas están contenidas en sendos cuadrantes.

Es una estrategia en la que el foco se coloca en la identificación del problema y la clasificación de estos en distintos niveles de riesgo. Es una estrategia cuyo estilo está centrado en la detección de riesgos (risk-driven strategy), a diferencia del modelo de cascada que está

centrado en la documentación (document-driven strategy). Las etapas primitivas se pueden resumir en:

- identificar objetivos, alternativas y restricciones
- evaluar alternativas, identificar y resolver riesgos
- desarrollar y verificar los productos
- planificar la próxima fase

El modelo en espiral introduce un conjunto de ideas nuevas respecto del modelo de cascada, a saber: valoración y resolución de riesgos; una estrategia iterativa e incremental; una estrategia de prototipación para ciertas actividades; reconoce el caracter evolutivo para ciertas actividades.

Para que la estrategia recursiva/paralela sea efectiva [Berard 93], debe atacar proyectos en los que su propia naturaleza permita particionamiento. Consiste en ver a un sistema como compuesto de subsistemas lo más independientes posibles. De este modo a un problema se lo puede descomponer sistemáticamente en subproblemas. Este subproblema representa a un componente independiente, al que a su vez se le pude aplicar descomposición (la parte recursiva). Además la estrategia recursiva/paralela permite trabajar simultáneamente sobre un conjunto de componentes (la parte paralela). Así como se aplica la estrategia de descomposición la estrategia de composición es también factible.

El ciclo de vida para cada componente sigue un orden: analizar un poco, luego diseñar un poco, luego implementar otro poco y después testear. Tanto este modelo de proceso como el propuesto por Henderson et al, son más apropiados para proyectos que usan principios y tecnologías de orientación a objetos.

Es oportuno decir que para la construcción de artefactos de hipermedia no es adecuado aplicar una estrategia secuencial como la propuesta en el modelo de cascada por las desventajas antes expuestas (no favorece la estrategia participativa ni la retroalimentación experimental, no favorece la evolución de los artefactos, etc., etc.). Hablando en un sentido amplio, podemos afirmar que dada la naturaleza de las aplicaciones de hipermedia, con propiedades de interacción/navegación/interface, una mezcla planificada de estilos iterativo, incremental, concurrente y oportunístico, guiada por prototipación flexible es el más adecuado a este tipo de desarrollos, como discutiremos en el capítulo 4. (El lector debe tener presente que existen distintos tipos de proyectos [Goldberg et al 95]: “el primero de su tipo”, “una variación del

mismo”, "generación de componentes reusables", etc. para los cuales habrá que adaptar las estrategias del modelo de proceso).

Algunas características de los modelos recursivo/paralelo y en espiral nos serán de utilidad; sin embargo este último aplica el concepto de detección, análisis y resolución de riesgos por cada iteración lo que introduce demoras afectando el ritmo del proyecto e incrementando potencialmente los costos (sobre todo para proyectos de pequeña o mediana envergadura). Los modelos de proceso discutidos y sus estrategias deberán ser adaptados al desarrollo de artefactos de hipermedia ya que hay modelos, actividades, artefactos, recursos y roles que pertenecen específicamente a este nuevo campo.

Por otra parte, en esta última década han sido publicados un buen número de criterios, modelos y principios que ponen énfasis en estructurar y producir aplicaciones de hipermedia [Conklin 87, Garzotto et al 91, Grønbaek et al 94, Nanard et al 91], y recientemente se han establecido metodologías de hipermedia, sobre todo enfocadas hacia actividades de diseño, y apoyadas en modelos y principios de Ingeniería de Software. Entre ellas podemos citar a HDM [Garzotto et al 91, Garzzoto et al 93], RMM [Isakowitz et al 95], OOHDM [Rossi 96a, Schwabe et al 95a , Schwabe et al 96]. Presentaremos algunas características de las mismas y sus aportes a una estrategia integral de modelo de proceso. Finalmente, consideraremos algunos comentarios valiosos realizados por Nanard et al [Nanard et al 95] respecto de características de los entornos de desarrollo de hipermedia y su rol en el proceso de diseño.

Uno de los inconvenientes al diseñar artefactos de hipermedia es cómo representar los resultados del proceso. Modelos físicos como prototipos son necesarios pero no suficientes porque pueden resultar en ambigüedades e inadecuados como base de comparación en análisis de diseños alternativos (de alto nivel). Por lo tanto es necesario contar también con modelos lógicos, formales o semiformales, que soporten a las actividades de diseño y documentación.

HDM (Hypermedia Design Method) es el primer enfoque basado en modelos para especificar decisiones en actividades de diseño. Sin embargo HDM es un modelo de diseño y no una metodología como RMM (Relationship Managment Methodology) u OOHDM; es decir, no especifica un conjunto de actividades, el ordenamiento y el comportamiento de los pasos en las cuales se crean y evolucionan los modelos.

HDM contiene unas pocas primitivas de modelado. Un esquema en HDM está compuesto por un conjunto de entes y enlaces. Los entes están constituidos por componentes y cada

componente puede ser observado bajo diferentes perspectivas (unidades). Los entes y componentes se conectan por medio de enlaces estructurales o de aplicación. En este enfoque se emplean estructuras de acceso como índices, rutas guiadas para ordenar y facilitar el acceso a la información. Es importante resaltar ideas subyacentes en el modelo de diseño de HDM (el cual ha dado origen a ampliaciones y ha servido como cimiento de otras metodologías):

- ofrece un esquema de navegación explícito al definir distintos tipos de enlaces y entes
- sus modelos son independientes de los entornos de implementación
- ofrece primitivas de composición y estructuración jerárquica
- incluye el concepto de perspectivas de un atributo

A diferencia de HDM, en RMM como en OOHDM se reconoce un proceso de desarrollo de hipermedia. Ambas metodologías definen subyacentemente un comportamiento incremental e iterativo para actividades de diseño y desarrollo.

La metodología RMM recibió influencia de HDM y su sucesor HDM2. Es una metodología estructurada en el modelo de datos (modelo de entidad-relación, E-R). Comprende un conjunto de etapas de diseño de alto nivel, de diseño detallado y de construcción. Se puede resumir en las siguientes etapas o tareas primitivas: 1) diseño de E-R, 2) diseño (particionamiento en slices) de entidades, 3) diseño navegacional, 4) diseño de protocolos de conversión 5) diseño de pantallas de interface de usuario 6) diseño de comportamiento de tiempo de ejecución, 7) construcción y testeo.

En el diseño de E-R se analizan las entidades relevantes y las relaciones en el dominio de la aplicación. Los autores afirman la importancia de explicitar los enlaces entre los distintos objetos porque serán los caminos principales por los que los usuarios navegarán los ítems individuales de información. En la siguiente tarea, propia de aplicaciones de hipermedia, involucra el particionamiento de una entidad en trozos cohesivos de información o “slices” significativos, para después organizarlos en una red de hipertexto. RMM hereda de HDM los conceptos de enlace estructural y asociativo. “Desde el punto de vista navegacional, es importante la diferencia entre estos dos tipos de conexiones. Cuando un usuario atraviesa un enlace asociativo, el contexto de la información cambia, por ejemplo, de una facultad a un curso. No obstante, cuando se atraviesa un enlace estructural, la información del contexto queda dentro de la misma entidad” [Isakowitz et al 95]. En la tercera etapa se diseñan los caminos que posibilitarán la navegación; se analizan las relaciones asociativas descubiertas en la primera etapa. Además RMM cuenta con seis primitivas de acceso navegacional (índices,

tours guiados, etc.). En el proceso de diseño de protocolos de conversión se producen reglas de traducción del modelo previo en objetos de la aplicación (considerando el entorno de autoría: HTML, Toolbook, Macromind Director, etc.). Durante la quinta etapa se analizan esquemas de pantallas para los objetos perceptibles por el usuario. Las etapas de construcción y testeo no varían demasiado de los proyectos tradicionales; excepto que en aplicaciones de hipermedia se deben tener en cuenta los test de los principales caminos de navegación, entre otros asuntos.

A diferencia de HDM, RMM provee un proceso de diseño paso a paso y enriquece significativamente a sus primitivas. También es el caso de OOHDM el cual define un proceso de desarrollo de cuatro pasos y que a seguir describiremos. Es oportuno decir que tanto OOHDM como RMM son metodologías centradas principalmente en el diseño y construcción, no tratando actividades de etapas tempranas del ciclo de vida como por ejemplo análisis de factibilidad, planificación de proyectos, estrategias de desarrollo participativas como prototipación, y otras actividades que en el punto 2.4 introduciremos y en el capítulo 4 discutiremos con mayor detalle. Sin embargo OOHDM es la metodología de diseño más avanzada hasta la actualidad, por la potencia de sus constructores Orientados a Objetos (OO), por la clara división de preocupaciones entre las etapas conceptual, de navegación, de interfaces abstractas e implementación, y, principalmente, por la sonada estrategia de reuso por medio de patrones de diseño [Rossi 96b, Rossi et al 97].

La metodología OOHDM consta de las siguientes etapas o tareas fundamentales: 1) modelado conceptual, 2) diseño navegacional, 3) diseño de las interfaces abstractas, 4) implementación.

El objetivo fundamental en la etapa de modelado conceptual consiste en analizar y definir la semántica del dominio de la aplicación. El producto de esta tarea son diagramas de clases y objetos construidos a partir de clases, objetos, atributos (posiblemente multitipados), relaciones y subsistemas. Los autores consideran importante explicitar las relaciones entre los distintas clases porque serán los potenciales enlaces para los esquemas de navegación. No es preocupación en esta etapa el análisis de los tipos de usuario y de las tareas realizadas por ellos.

En el diseño navegacional (segunda etapa) se deben considerar principalmente aspectos cognitivos, el perfil de usuario para cada vista a construir a partir del modelo conceptual, las tareas realizadas y, en definitiva, encontrar los distintos contextos navegacionales. Las primitivas son los nodos, enlaces, clases y contextos navegacionales. Un contexto navegacional se define como un conjunto de enlaces, nodos y otros contextos. Se construyen tres esquemas fundamentales en esta etapa: el esquema de clases navegacionales y los esquemas de contextos,

y, para especificar la dinámica (es decir, el conjunto de objetos navegables accesibles a cada momento) se construyen las transformaciones de navegación.

La metodología OOHDM diferencia el diseño navegacional del diseño de interfaces abstractas. Esto permite construir diferentes interfaces para un mismo modelo navegacional. Los aspectos a tener en cuenta en este paso radican en definir qué eventos intervendrán en el lenguaje de acción y qué objetos de interface percibirá el usuario (asociados a algún estilo y metáfora); qué transformaciones de interface y de objetos navegacionales sucederán; cómo serán sincronizados los objetos multimediales de interface, principalmente los que incorporan audio y video. Básicamente se utiliza el mecanismo de ADV (Abstract Data View) para modelar cada objeto perceptible y se define un tipo de ADV para cada clase navegacional; para especificar las relaciones estáticas entre ADV, eventos externos iniciados por el usuario y objetos de interface que causan navegación se construyen diagramas de configuración; y para mostrar la dinámica de la aplicación se especifica para cada ADV su correspondiente carta-ADV.

En estas tres etapas existe una transición suave entre modelos lógicos basados esencialmente en los principios de objetos que sirven de entrada a una cuarta, de implementación, en donde los objetos de navegación y de interface se mapean a los objetos de implementación. En esta etapa se selecciona el entorno de hardware y herramientas de autoría apropiadas.

Es importante destacar los aspectos evolutivos de la metodología OOHDM con respecto a las discutidas previamente:

- *ofrece mecanismos de abstracción más ricos* (los modelos construidos están basados en el paradigma OO el cual soporta mecanismos como clases, generalización/especialización, agregación y subsistemas).
- *establece una clara división de preocupaciones entre etapas, proveyendo un conjunto bien definido de primitivas.* (Los constructores de proceso, al estar basados en modelos, permiten encarar proyectos de hipermedia de mayor complejidad, favoreciendo atributos de Ingeniería de Software como son la reusabilidad y mantenibilidad).
- *ofrece un conjunto de patrones arquitectónicos de alto nivel, útiles en el proceso de desarrollo, para documentar y reusar experiencias de diseño.* (Entre algunos patrones documentados en la bibliografía antes citada se encuentran: visiones navegacionales, contextos navegacionales, transformaciones navegacionales, observador de navegación, información bajo demanda, referencia activa).
- *permite definir un modelo de seguimiento.* (Esto favorece esencialmente mecanismos de

evolución y mantenimiento de artefactos en proyectos de hipermedia de mediana y gran envergadura).

Finalmente, en [Nanard et al 95] los autores concluyen que los métodos y técnicas formales mejoran la consistencia de las especificaciones y proveen guías bien definidas en las fases pero que buena parte de las preocupaciones en el diseño hipermedial surgen de cuestiones cognitivas y estéticas. Proponen que un ambiente de diseño hipermedial no sólo debe soportar técnicas formales sino que también debe incorporar un proceso oportunístico e incremental basado en la retroalimentación experimental.

2.3 Motivaciones

El modelado de procesos es la ciencia que sirve para representar y soportar a los procesos y a las distintas interacciones que forman parte de un problema del mundo real. Una de las dificultades en modelado de procesos es la complejidad inherente de los procesos de software de la realidad. Sólo basta con citar alguna de sus variables para comprender la complejidad subyacente: por ejemplo costos, recursos, habilidades, restricciones, artefactos creados evolucionados o reusados, duración del proyecto, transformaciones de procesos y artefactos, estructuras organizacionales, constructores de proceso, etc. conforman algunos ingredientes básicos de todo proyecto.

Los procesos de software se pueden abstraer en un modelo de proceso, es decir, podemos obtener un común denominador lo suficientemente significativo de manera que sirva con fines de comunicación, comprensión, análisis, guía y reuso en diferentes proyectos.

Un modelo de proceso debe ayudar a responder preguntas como:

- Qué hacer
- Cómo y cuándo hacerlo
- Dónde y quiénes lo harán
- Qué dependencias existirán entre tareas y otros entes
- Qué estrategia de modelado de procesos se utilizará conforme a las metas y objetivos del proyecto

Ampliando los conceptos de Goldberg et al. definimos a un modelo de proceso de desarrollo de software como a una estrategia apropiada para abstraer, organizar, ejecutar y/o controlar

(mediante el uso sistemático de heurísticas, métodos, modelos, técnicas y herramientas), a las distintas fases, actividades, recursos y artefactos de un proyecto con el fin de alcanzar las metas y objetivos establecidos.

Abstrayéndonos de las metas y objetivos particulares de cada proyecto, afirmamos que todo proceso de software debe velar continuamente por tres fines esenciales: desarrollar los artefactos de calidad, emplear los procesos óptimos y aplicar los recursos apropiados. Desarrollar los productos de calidad significa producir artefactos los cuales deban contener un conjunto de características y atributos deseados. Utilizar los procesos óptimos significa seleccionar los procesos más efectivos de manera de usarlos consistentemente para obtener la aplicación que cumplimente tales características.

Para los agentes intervinientes en el ciclo de desarrollo, los atributos representan una serie de restricciones a cumplir por los productos que se están construyendo, los procesos que se están empleando, y los recursos que se están asignando. Para que la descripción del modelo de proceso se complete es preciso definir las características y atributos que representen unidades de medida concreta para observar, retroalimentar, evaluar, analizar, mejorar y predecir.

Lo anterior da pie para definir cuáles son los atributos que contribuyen a la calidad en un proceso de desarrollo. Por ahora sólo diremos (ya que será tratado con mayor profundidad en la capítulo 5) que un proceso de calidad debe ser: repetible, correcto, medible, relevante, debe permitir ciclos de retroalimentación y aprendizaje conjunto, debe ser flexible (adaptable a varias definiciones de proceso, útil para ser instanciado en varios proyectos, escalable). Algunas de estas características y atributos para procesos (artefactos y recursos) son propias o deben ser adaptadas para el campo de Hipermedia (por ejemplo navegabilidad, relevancia de enlaces, confiabilidad, usabilidad, etc.)

De manera que si bien muchas de las características y estrategias de los modelos de proceso comentados en la sección 2.2 nos serán de utilidad (como estrategias iterativas, incrementales, prototipación, análisis de riesgo, etc.), nuestra propuesta es definir un modelo de proceso flexible que se adecue totalmente al desarrollo de artefactos de hipermedia. Como observamos anteriormente hay modelos (por ejemplo de calidad), entes (como actividades, artefactos, recursos, roles, constructores de proceso) y atributos de estos entes que pertenecen específicamente a este reciente campo de investigación y desarrollo. Dada la naturaleza de las aplicaciones de hipermedia, con características cognitivas y estéticas, de interacción, navegación e interface, una mezcla planificada de estrategias iterativa, incremental, concurrente

y oportunística, que soporte prototipación y reuso es lo mas adecuado para este tipo de desarrollos. Esta mezcla de estrategias debe ser planificada y se ajustará dependiendo del tipo de proyecto y producto.

Además un modelo de proceso debe abstraer e integrar diferentes tipos de información de los procesos de software en perspectivas. Una perspectiva es un enfoque particular, una vista, para especificar y comunicar información del modelo. Para tratar con la complejidad inherente de los procesos nosotros proponemos perspectivas a partir del modelo conceptual del dominio de modelado de procesos en general. Curtis et al. agrupan la información en cuatro perspectivas con el fin de abstraer la complejidad subyacente en los procesos de software, y son, a saber: perspectiva funcional, perspectiva informacional, perspectiva de comportamiento y perspectiva organizacional. Nuestra taxonomía amplía la propuesta por Curtis et al. incorporando una perspectiva metodológica la cual es fundamental para representar a los diferentes constructores de proceso para realizar las tareas (y además puede ser integrado en ambientes de ingeniería de software centrado en procesos). En el capítulo tres y cuatro se estudian algunos de estos aspectos.

En cuanto a OOHDM y RMM comentamos que son metodologías centradas principalmente en actividades de diseño y construcción, no tratando actividades de etapas tempranas del ciclo de vida, ni estrategias participativas de desarrollo, ni actividades de mantenimiento. No obstante desde el punto de vista de la perspectiva metodológica nos serán de gran utilidad para definir a los constructores de proceso que se emplearán para especificar, personalizar y ejecutar las descripciones de procesos correspondientes a las tareas de diseño de alto nivel y diseño detallado.

Si bien se pueden especificar distintas descripciones de proceso a las que se les puede aplicar diferentes constructores, nosotros emplearemos en los ejemplos (capítulo 4) los constructores de OOHDM, por las razones expuestas al final de la sección 2.2. (Por ejemplo, para la tarea de modelado conceptual, una descripción se podría personalizar y luego ejecutar con los constructores de RMM, basados en los diagramas de E-R o con los de OOHDM, basados en diagramas de clases, objetos y relaciones).

Para ingeniería de requerimientos aplicamos modelos y constructores de proceso centrados en casos de uso y modelado de interfaces (sección 4.3).

Ultimamente, en la comunidad científica de hipermedia, se está comenzando a considerar la

necesidad de contar con un enfoque amplio de proceso de desarrollo. Principalmente se está observando que es preciso contar con un enfoque ingenieril; esto es, el uso disciplinado, cuantificable y sistemático de principios y características de la Ingeniería de Software para la creación, evolución, evaluación y control de artefactos (y otros entes) en proyectos de hipermedia.

2.4 Panorama de las principales fases y procesos del MPF

Introduciremos las principales fases y procesos del modelo de proceso propuesto, a un alto nivel de abstracción, y sin hacer hincapié en las diferentes perspectivas (aunque lo que sigue se enfoca principalmente en aspectos funcionales).

El modelo está comprendido en tres fases generales [Olsina 97c] como se ve en la fig. 2.1. En cuanto a una elipse representa a un subsistema proceso (la cual puede estar compuesta jerárquicamente de actividades), y las flechas representan dependencias funcionales de entrada y salida (por ejemplo un proceso depende de la culminación de un artefacto de otro proceso o a un proceso se le envía un mensaje de control). En los laterales de la figura se puede apreciar las principales fases de la vista.

La primera fase se denomina *fase de exploración* en la cual se elicitan conceptos y requerimientos iniciales, si fuera necesario se realiza un estudio de factibilidad y se planifica preliminarmente.

La segunda fase, *de modelado dinámico*, es la fase esencial de re-planificación, elicitación y especificación de requerimientos detallados, coordinación y control, análisis del dominio, diseño, construcción, validación, integración, empleo de criterios estéticos y cognitivos, aseguramiento de la calidad y documentación. Se hace uso intensivo de modelos lógicos y físicos.

La tercera fase, denominada *fase operativa* o de vida útil de los artefactos, consiste esencialmente en tareas de documentación, configuración de cambios, mantenimiento y evolución de los artefactos. En cuanto a la tarea de mantenimiento y al igual que en Ingeniería de Software tradicional podemos considerar tres tipos de mantenimiento: correctivo, adaptativo y perfectivo.

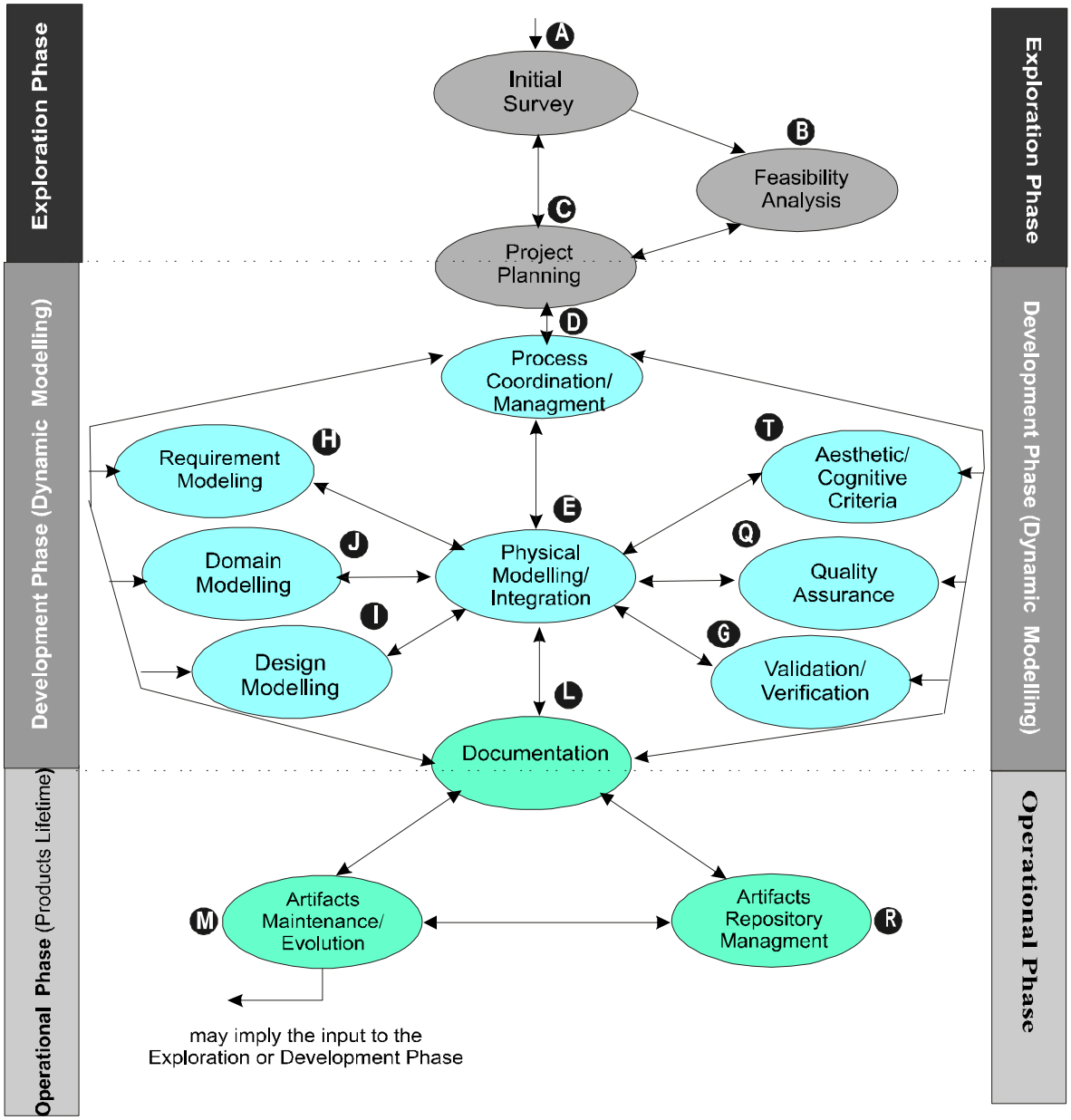

Fig. 2.1 Panorama de las fases y procesos principales del modelo de proceso flexible

Antes de comenzar el proceso de modelado detallado del dominio y de diseño arquitectónico se debe conceptualizar sobre el problema e identificar requerimientos iniciales. La entrada al proceso **A** puede ser en el mejor de los casos la extensión de un sistema en operación con

requerimientos documentados (de la fase operativa) o puede implicar la construcción de una nueva aplicación o componente en donde se necesita identificar el vocabulario del dominio, analizar a los tipos de usuarios y sus tareas y estudiar alternativas (dado el carácter de nuevo que tiene la disciplina de hipermedia es difícil encontrar componentes o aplicaciones documentadas que sirvan de entrada a los requerimientos iniciales). Aquí usamos el modelo de requerimientos para elicitar el dominio del problema. Se puede determinar inicialmente un conjunto de abstracciones claves y una secuencia de acciones que defina los modos específicos en que distintos actores interactuan con el sistema y establecer casos de uso [Jacobson et al 92]. La salida de esta tarea es una especificación inicial de requerimientos (posiblemente expresados en lenguaje natural o en algún otro tipo de descripción de procesos).

A partir de los requerimientos iniciales se puede extraer posteriormente información de cómo los usuarios realizan sus tareas y reconocer a objetos, transformaciones, nodos, contextos de navegación u otras primitivas arquitectónicas.

Es necesario confeccionar un modelo del plan inicial del proyecto, por medio de la tarea **C**. Básicamente contendrá, a partir de los requerimientos iniciales y del análisis de factibilidad (tarea **B**): una descripción del objetivo del sistema de hipermedia a construir o extender, el alcance previsible del producto final, los usuarios involucrados en el proceso y sus respectivas responsabilidades, las estrategias del modelo de proceso a utilizar, la elección del ambiente de desarrollo y de la herramientas, la posibilidad de establecer métricas en el contexto de un modelo de calidad. Como se puede ver esta actividad abarca las dos fases, por lo que el documento debe incluir la fecha y el número de revisión que sean de utilidad para el coordinador del proyecto en posteriores iteraciones.

Como se puede observar la segunda fase está conformada por siete componentes mutuamente necesarios para el proceso de desarrollo de hipermedia: el proceso **D** que corresponde al proceso coordinador y controlador de otros procesos y actividades (asignado por ejemplo a un agente humano que cumpla el rol de administrador del proyecto); el componente integrado por los procesos de alto nivel **H-J-I** que corresponden a procesos principalmente de modelado lógico y especificación independientes del entorno de implementación; el componente de tareas **E-G** guiado por la estrategia de prototipación y que corresponden a un esquema de iteración/retroalimentación experimental desarrollador-usuario para tareas de diseño, construcción y validación. Como se aprecia en la figura la estrategia de prototipación alimenta asimismo a las actividades del modelado lógico. El proceso **L** que comprende la documentación y se comunica con el repositorio de administración de artefactos (proceso **R**);

el proceso **Q** que corresponde a actividades de aseguramiento de la calidad de los procesos y, esencialmente, de los productos; el proceso **C** de replanificación del proyecto, y el proceso **T** para el empleo de criterios cognitivos (tratado en la sección 4.10) y estéticos en el desarrollo de aplicaciones de hipermedia.

El comportamiento del modelo de proceso, principalmente el ejecutado en la fase de desarrollo es una combinación planificada de estrategias *iterativa, concurrente, incremental y oportunística, guiada por prototipación flexible* (referirse a la sección 4.9).

2.4.1 Ejemplo a utilizar

Dentro de los proyectos realizados se encuentra un trabajo de autoría denominado "*Facultad de Ingeniería*" que utilizaremos para ejemplificar distintos aspectos del proceso de desarrollo.

El proyecto se construyó empleando el modelo de proceso introducido y el dominio de la aplicación representa una vista del esquema conceptual de un Sistema de Información Académico (fig 4.9). El perfil del usuario considerado es el del estudiante. El objetivo de la aplicación consiste en que potenciales ingresantes de distintas regiones conozcan facetas de la Facultad y les ayude a seleccionar los centros académicos al que concurrirán.

Desde el punto de vista de autoría la misma utiliza distintos recursos de multimedia (videos, animaciones, textos, sonidos, imágenes). En la fig. 2.2 apreciamos dos secuencias de un video de presentación, integrado en un ambiente de desarrollo Multimedia Toolbook ™.

Desde el punto de modelado lógico la aplicación está diseñada en varios contextos. Un contexto de navegación está compuesto esencialmente por clases navegacionales como nodos, enlaces, anchors y otros contextos. Representa una unidad o espacio de información semánticamente cohesiva y navegable de un modo no necesariamente secuencial. En la fig. 2.3a se aprecia el nodo principal (implementado) en donde se seleccionan los distintos contextos temáticos de la clase "*EnteFacultad*". Podemos apreciar los anchors de los contextos "*Ubicación*", "*Carreras*", "*Futuro*", etc. En la fig 2.3b mostramos la pantalla del primer nodo destino del anchor "*Carreras*"

Fig. 2.2 *Dos secuencias de la presentación inicial de la aplicación "Facultad de Ingeniería".*

a) b)

Fig. 2.3 a) *Menú principal o nodo superior en la jerarquía de la aplicación hipermedial;* **b)** *Primer nodo del contexto "Carreras"*

En la figura 2.3a podemos además apreciar dos estructuras de acceso como "*Índice*" y un "*Tour*" guiado. El índice tiene el mismo contexto temático que el mostrado en el menú pero por cada tema agrega en orden alfabético los subtemas correspondientes: así el usuario puede acceder de un modo directo a la información de su interés.

En cuanto al tour o visita guiada, está en un contexto navegacional que considera exclusivamente al estudiante dentro del perfil de usuario seleccionado. Se navega por la hiperbase teniendo en cuenta esencialmente información de "*Ubicación*", "*Carreras*" que además incluye información de requisitos de ingreso, "*Servicios*" al estudiante como becas, viviendas, etc.; y cierta información de la perspectiva futura de la Facultad.

Actualmente estamos trabajando en un proyecto de hipermedia cuyo fin es una aplicación educativa para el soporte a la enseñanza de la tecnología de orientación a objetos.

Antes de pasar a discutir detalladamente aspectos del *Modelo de Proceso Flexible* definimos y analizamos, en el capítulo siguiente, los componentes de un modelo conceptual útil para la modelización de procesos.