UNIVERSIDAD DE CASTILLA-LA MANCHA
DEPARTAMENTO DE INFORMÁTICA

XIII Escuela de Verano de Informática

TENDENCIAS ACTUALES EN LA INTERACCIÓN PERSONA-ORDENADOR: ACCESIBILIDAD, ADAPTABILIDAD Y NUEVOS PARADIGMAS

Editores:

ISIDRO RAMOS SALAVERT
ANTONIO FERNÁNDEZ CABALLERO
MARÍA DOLORES LOZANO PÉREZ

XIII Escuela de Verano de Informática

TENDENCIAS ACTUALES EN LA INTERACCIÓN PERSONA-ORDENADOR: ACCESIBILIDAD, ADAPTABILIDAD Y NUEVOS PARADIGMAS

Editores:

ISIDRO RAMOS SALAVERT
ANTONIO FERNÁNDEZ CABALLERO
MARÍA DOLORES LOZANO PÉREZ

XIII Escuela de Verano de Informática

Universidad de Castilla-La Mancha

Organiza

Departamento de Informática

Escuela Politécnica Superior de Albacete

Director

Dr. D. Isidro Ramos Salavert

Secretarios

Dr. D. Antonio Fernández Caballero

Dra. Dª. María Dolores Lozano Pérez

Comité Organizador

Dr. D. Pascual González López

Dr. D. Antonio Fernández Caballero

Dra. Dª. María Dolores Lozano Pérez

Dr. D. José Antonio Gallud Lázaro

D. José Pascual Molina Massó

D. Francisco Montero Simarro

D. Víctor López Jaquero

Presentación

Como ya viene siendo habitual, la sección del campus de Albacete del Departamento de Informática de la Universidad de Castilla-La Mancha mantiene su participación en el programa de Cursos de Verano de la UCLM, y con ello su compromiso de mostrar a la sociedad las tendencias actuales en algunos de los temas de gran actualidad e importancia dentro del campo de la Informática.

El objetivo de esta edición es mostrar las *"Tendencias Actuales en la Interacción Persona-Ordenador: Accesibilidad, Adaptabilidad y Nuevos Paradigmas",* tema, sin duda, de gran interés en el ámbito de desarrollo de sistemas interactivos. Para ello, el Comité Organizador ha logrado reunir en nuestra ciudad a prestigiosos investigadores nacionales que nos ilustrarán sobre diferentes aspectos relacionados con este tema. Pero tal vez lo más relevante de este foro, es la idea básica que mueve a los organizadores y autores de los diferentes trabajos, que no es otra, que ofrecer una serie de conceptos y propuestas que permitan a los asistentes, pieza fundamental de todas las Escuelas de Verano organizadas por este Departamento, obtener una amplia visión del estado actual y perspectiva futura de la interacción entre el usuario y los sistemas informáticos.

Así mismo, desde estas líneas, los miembros del comité organizador queremos dar nuestro sincero agradecimiento a D. Isidro Ramos Salavert por presidir y colaborar tan estrechamente con nosotros en la organización de esta Escuela de Verano. A su vez agradecer a todos los ponentes la colaboración prestada y el esfuerzo realizado, uno de cuyos frutos es el libro que aquí presentamos y que resulta muy interesante como reflejo de lo que hoy es la Interacción Persona-Ordenador.

Por último, y no por ello menos importante, agradecer la continua colaboración que las instituciones locales y provinciales han venido prestando a la realización de esta tipo de encuentros de formación, que, a su vez, permiten poner en contacto a los alumnos de nuestro Campus con prestigiosos investigadores de las principales Universidades y Centros de Investigación de nuestro país. En especial, agradecemos la colaboración prestada por la Diputación Provincial y el Ayuntamiento de Albacete, la Caja de Castilla-La Mancha, el Parque Científico y Tecnológico de Albacete y el Instituto de Investigación en Informática de Albacete.

El Comité Organizador

Prólogo

La rápida consolidación de un grupo de investigación sobre el tema Interacción Persona-Ordenador en la UCLM unida a la tendencia a la ubicuidad de los ordenadores actuales han invitado a tratar este tema en la Escuela de Verano de Informática de la UCLM (EVI 03). La situación en el tema se asemeja a la presencia de tecnología eléctrica (motores por ejemplo) en una casa actual. A la pregunta: ¿Cuántos motores eléctricos hay en su casa hoy? La respuesta sería por encima de la docena… con seguridad. Lo mismo está ocurriendo con los ordenadores. En poco tiempo formarán parte de nuestro entorno vital en una cantidad elevada. Unos estarán empotrados y aumentarán las capacidades de objetos cotidianos o nuevos, otros controlarán nuestro *hábitat* y muchos realizarán el ámbito de interacción humano con una gran parte del resto del mundo: *e-mail*, *e-commerce*, *e-services*… de forma que su ausencia será vivida como un *handicap* vital (como la ceguera, la sordera,…). Ello hace que la interacción del hombre con los servicios que soporta un ordenador sea un problema de crucial importancia. De su ergonomía, versatilidad… dependerá que ese nuevo "sentido", en mucho superior a la vista, oído,… sea útil para la realización del ser humano como ser social en la Sociedad de la Información.

Estamos desarrollando la tecnología de la interacción. Este 6º sentido, que nos comunica con el exterior no está basado en la luz (como la vista), en el sonido (como el oído)… sino con la información en todas sus formas y por lo tanto en todos ellos: luz, sonido, tacto…

El diseño por lo tanto de esa interfaz hombre-máquina adquiere una importancia y vigencia enorme. La diferencia esencial del hombre con otros seres vivos consiste en su capacidad de interacción abstracta: el lenguaje. Ha sido mediante su uso desde los primeros balbuceos como sinérgicamente se ha realizado la inteligencia, la cultura, el hecho diferencial humano. Históricamente el cambio de interlocutor propiciado por la Informática: de hombre a ordenador, ha obligado a formalizar el lenguaje (Teoría de lenguajes: sintaxis, semántica, pragmática, compilación…) y en ese proceso de reflexión nos hemos replanteado el propio proceso comunicativo. De esa reflexión han surgido teorías nuevas, y nuevas metáforas interactivas que han propiciado un cambio de orden en el proceso de interacción hombre-máquina.

Conscientes de ello la EVI 03 aborda en profundidad el tema. Prestigiosos especialistas de fuera y dentro de nuestra universidad establecerán el estado actual del tema de forma precisa y actual.

Quiero agradecer a todos los ponentes su valiosa participación e invitar al público a asistir a esta nueva edición de la Escuela de Verano de Informática de la UCLM 2003.

Isidro Ramos
Stony Brook, Nueva York. Junio 2003

Índice General

Generación Automática de Aplicaciones WEB a partir de Esquemas Conceptuales Orientados a Objetos*

Oscar Pastor, Joan Fons y Vicente Pelechano

Departamento de Sistemas Informáticos y Computación
Universidad Politécnica de Valencia
Camino de Vera s/n
46022 Valencia, Spain
{opastor, jjfons, pele}@dsic.upv.es

Resumen. Este artículo presenta una aproximación para el modelado orientado a objetos de soluciones web que proporciona mecanismos para la especificación de sistemas de información dinámicos hipermediales y de comercio electrónico. Se propone una extensión de un método OO para la generación automática de código usando modelos conceptuales (OO-Method) que permite capturar y representar la semántica navegacional y de presentación de información usando primitivas de abstracción de alto nivel. Se define un proceso guiado de construcción de una solución web completa y se aplica la aproximación planteada a un caso de estudio sobre una aplicación para el comercio electrónico.

1 Introducción

Hoy en día, con la rápida expansión de Internet y los avances en el área de las tecnologías web han aparecido un nuevo tipo de aplicaciones en estos entornos, y son cada vez más complejas y dinámicas. Además, debido al acelerado crecimiento y la alta competitividad de las actividades comerciales en la Red, estos sistemas son construidos en periodos temporales muy cortos, sin el apoyo de herramientas de trabajo adecuadas y utilizando soluciones ad-hoc, lo que está llevando a construir sistemas software de baja calidad y de difícil mantenimiento y evolución.

En los últimos años han surgido gran cantidad de aproximaciones metodológicas que intentan ayudar en la sistematización de la construcción de soluciones en ambientes web, proporcionando mecanismos de abstracción que faciliten el desarrollo de estos sistemas. Además, se están intentando definir marcos de trabajo integrados que proporcionen herramientas adecuadas para dar soporte a la construcción de estos sistemas en todas sus fases. Pero actualmente no existe ningún método totalmente establecido.

Existen dos tendencias claras. Unas aproximaciones se basan en extender iniciativas orientadas al diseño hipermedial (navegacional), introduciendo

* Esta investigación está soportada por el Programa CYTED, en el proyecto VII.18, WEST y el Proyecto CICYT del programa FEDER, con ref. TIC 1FD97-1102

expresividad para dotar de dinamismo a los sistemas. Estas aproximaciones aparecieron hacia el principio o mitad de la década de los 90, con el objetivo de construir aplicaciones hipermediales donde se unía el concepto de navegación con la multimedia, en sistemas claramente estáticos (sin funcionalidad). Es por esto que la mayoría de estas aproximaciones están basadas en el Modelo Relacional clásico, o bien en extensiones de éste. Algunos ejemplos destacables de estas iniciativas son OOHDM [22] (actualización OO de HDM [9]), WebML [6], ADM [14], AutoWeb [16] y RMM [11]. El otro grupo de aproximaciones se basan en la idea de extender los métodos de desarrollo orientados a aplicaciones dinámicas (con funcionalidad), que podríamos llamar "convencionales", tratando de introducir la semántica de la hipermedia como característica inherente a este nuevo tipo de sistemas software. Este tipo aproximaciones (por lo general, más recientes) tratan de introducir características navegacionales al modelo OO. En este grupo podemos encontrar los métodos UWE [4], WSDM [7], EORM [13], OOH [10] y OO-Method [17].

Sin embargo, en estas propuestas, las características hipermediales y las propiedades funcionales son tratadas habitualmente por separado, dificultando el problema de desarrollar una aplicación web en un marco de trabajo unificado. En la práctica, estos métodos proporcionan una solución parcial, bien porque se centran en captar las características navegacionales (en detrimento de la especificación funcional del sistema), o bien en captar las características más convencionales (definición de clases y operaciones para expresar la funcionalidad), sin tener en cuenta la semántica navegacional de los sistemas.

Detrás de estas soluciones parciales está empezando a ser ampliamente aceptado que los sitios web están evolucionando de simples repositorios de información hipermedia hacia complejas aplicaciones hipermediales distribuídas, normalmente conocidas como "aplicaciones web" [3]. Además, la complejidad de los sistemas aumenta debido al gran tamaño y la cantidad de potenciales tipos diferentes de usuarios con los que pueden interactuar. Por tanto, se deben proporcionar mecanismos que faciliten y guíen la construcción de soluciones y favorezcan el reuso de soluciones ya probadas.

Nuestra propuesta proporciona una contribución en este contexto: empezando por afirmar que el modelado conceptual es necesario para desarrollar aplicaciones web correctas y de calidad, introducimos una aproximación diferente que se centra en crear un entrono de trabajo integrado, donde se aborda tanto la perspectiva funcional del sistema, como la perspectiva navegacional y la de presentación de la información. Todas estas perspectivas unidas pueden ser usadas como entrada para ambientes de generación automática de código. Para poner en práctica estas ideas, se ha diseñado e implementado un proceso de producción de software, donde se definen unas guías para aplicar técnicas de modelado conceptual para el desarrollo de aplicaciones web: integrar las tres actividades que tradicionalmente se llevan a cabo por separado. Una primera donde se modelan las operaciones, una segunda donde se expresa el concepto de navegación y una tercera donde se plasman los requisitos de presentación de información usando primitivas básicas de presentación a nivel de modelado conceptual. Así, en nuestra aproximación, la fase de modelado conceptual se considera como una única fase. Usando lo que podríamos llamar una aproximación de modelado conceptual OO, se introduce la expresividad necesaria en el modelo para

especificar correctamente los requisitos navegacionales y las características de presentación. Toda esta información es utilizada para proporcionar una guía metodológica precisa para ir del espacio del problema (modelado conceptual) al espacio de la solución (representado por el producto software final), soportado por un proceso de generación automática de código.

Este artículo está organizado en 4 secciones. La sección 2 describe el método de producción de soluciones web *OOWS*, describiendo el proceso de desarrollo, el modelo de navegación y el modelo de presentación. En la sección 3 se aplica el método a un caso de estudio, y en la sección 4 se presentan las conclusiones y los trabajos futuros.

2 OOWS: Un método de desarrollo de aplicaciones web

Nuestra intención es definir un método de desarrollo que permita especificar sistemas software para ambientes web, extendiendo un método OO existente. Este tipo de aplicaciones tienen una base común con las aplicaciones software tradicionales: la funcionalidad del sistema y la interacción con los usuarios. Sin embargo, introducen nuevas características *navegacionales* que deben ser capturadas para representar de una manera más precisa y aproximada el sistema.

El método tomado como la base para definir esta aproximación es OO-Method [19]. Este método capta las propiedades funcionales del sistema que se consideran relevantes para construir una especificación textual OO y formal de manera automática. Esta especificación formal OO constituye un repositorio de información de alto nivel del sistema que será utilizado como entrada a un *compilador (automático) de modelos conceptuales (model compiler)*. La definición de un modelo de ejecución junto con una estrategia basada en patrones de traducción (de la especificación a la implementación), hacen posible la construcción de una implementación operacional, generando un prototipo del sistema completo (incluyendo las características estáticas y dinámicas) en la plataforma destino del sistema. En el contexto del proyecto OO-Method se han dirigido muchos esfuerzos hacia el desarrollo de nuevos modelos para enriquecer el Método de Producción de Software Orientado a Objetos con la expresividad necesaria para especificar las características navegacionales. Esta extensión del método OO-Method con capacidades navegacionales y de presentación es lo que llamamos OOWS [18][20] (Método de Producción de Soluciones Web Orientadas a Objeto).

2.1 Proceso de desarrollo de una aplicación web

En este apartado se define el proceso o guía metodológica a seguir para definir y posteriormente obtener un sistema software para un entorno web. Siguiendo la aproximación OO-Method, la especificación de los requisitos de usuario debe realizarse en la fase de modelado conceptual. Para modelar la navegación asociada al sistema deseado se propone un proceso de desarrollo de soluciones-web con dos pasos principales: Especificación del Problema y Desarrollo de la Solución.

En la fase de **Especificación del Problema** se deben capturar las peculiaridades y el comportamiento que debe ofrecer el sistema para satisfacer los requisitos de usuario identificados. En este paso se incluye el conjunto de requisitos usando una aproximación de Casos de Uso [12] y posteriormente las actividades de modelado conceptual del sistema. En el modelado conceptual, las abstracciones que se derivan del problema son especificadas en términos de clases y de su estructura, comportamiento y funcionalidad, construyendo los siguientes modelos: de Objetos, Dinámico, Funcional, Navegacional y de Presentación. Éstos describen la sociedad de objetos desde cinco puntos de vista diferentes usando un marco de trabajo OO bien definido:

- El Modelo de Objetos define la estructura y las relaciones estáticas entre clases identificadas en el dominio del problema.
- En el Modelo Dinámico se describen las posibles secuencias de servicios y los aspectos relacionados con la interacción entre objetos.
- El Modelo Funcional captura la semántica asociada a los cambios de estado entre los objetos motivados por la ocurrencia de eventos o servicios.
- El Modelo de Navegación define la semántica navegacional asociada a las clases de los objetos del modelo. Es en este modelo donde se explicita la navegación permitida en la aplicación para cada agente del sistema.
- El Modelo de Presentación captura los requisitos básicos de presentación de información, orientado hacia ambientes web. Está fuertemente basado en el modelo de navegación y permite definir, de una manera abstracta la estructura lógica de presentación de los objetos navegacionales en la interfaz de usuario.

En esta fase se realiza un estudio de los tipos de usuarios que pueden interactuar con el sistema, indicando qué visibilidad sobre el sistema tendrán (qué atributos y qué operaciones podrán ver y/o activar), cómo se podrán conectar (requerirán o no identificación), y se organizarán en jerarquías de especialización [8] para potenciar el reuso en la especificación del sistema, facilitando así la tarea de modelado.

En la fase de **Desarrollo de la Solución** se propone una estrategia de generación de código basada en componentes para integrar la solución propuesta en ambientes web. En esta etapa se obtendrá una aplicación web, con una funcionalidad equivalente a la especificación inicial según una visión operativa. Esta fase se desarrolla de manera totalmente automática por un compilador de modelos conceptuales (*model compiler*) que, en función de unos patrones arquitectónicos y dependiendo de la plataforma destino, construye un sistema software que recoge los requisitos de la aplicación modelada, siguiendo uno de los principales objetivos de la metodología propuesta por OO-Method. Esta estrategia, además, facilita las tareas de mantenimiento y evolución, ya que la generación automática basada en patrones se realiza utilizando soluciones previamente probadas y validadas. Un cambio en el sistema (evolución), implicará una modificación en el modelo conceptual y una regeneración automática del sistema. Además, esta filosofía nos permite obtener de una manera más rápida aplicaciones finales de calidad, evitando entre otras, la fase de pruebas (*testing*) del sistema.

A continuación se presentan las extensiones navegacionales y de presentación de información que se introducen en la propuesta OO-Method, a lo que hemos llamado la aproximación OOWS.

2.2 El Modelo de Navegación de OOWS

En esta sección se presentan las primitivas de modelado conceptual para extender un método OO con la intención de capturar la semántica navegacional de una aplicación web. En el método OOWS, se incorpora un nuevo modelo en la fase de modelado conceptual que recoge las características navegacionales: el *Modelo de Navegación*. Su objetivo es definir cómo se le proporcionará a cada usuario del sistema el acceso a la información y la funcionalidad que le es relevante para llevar a cabo su tarea dentro del sistema y qué secuencias de caminos deberán seguir para conseguirlo.

Actualmente, y debido al gran auge de las aplicaciones de carácter comercial en Internet (*e-commerce*), es de vital importancia que el acceso al sistema facilite la interacción con el sistema e incluso se le personalice. Un ejemplo ilustrativo es la tienda virtual Amazon.com [2], en donde, si se conecta un usuario registrado, el sistema mantiene información sobre sus preferencias, sus compras anteriores, sus productos más deseados, etc., proponiéndole nuevas compras que puedan ser de su interés. Esto nos lleva a tener en cuenta también aspectos referentes a personalización a nivel de usuario [21][1][5].

Pero existe actualmente una gran controversia en lo que se refiere al concepto de navegación: ¿qué es la navegación? ¿Cómo la debemos captar y expresar? ¿Se debe definir a nivel de modelado conceptual o a nivel de implementación? Existen muchos autores que giran en torno a la misma idea, pero sin proporcionar una definición que sea aceptada definitivamente. Algunos llegan incluso a confundir conceptos, como por ejemplo tratar parte de la funcionalidad del sistema como requisito de navegación (debido a que puede haber una fuerte interrelación). Nosotros proponemos una definición de navegación lo suficientemente genérica que pueda ser aceptada por cualquier método o propuesta. Para ello revisemos las características básicas y novedosas que incorpora una aplicación web. Cuando un usuario se conecta a una aplicación web, ésta le proporciona una visión del sistema software formada por un conjunto de *nodos enlazados* entre sí definiendo los posibles caminos que el usuario puede tomar. Esta estructura se puede organizar en un grafo dirigido donde los **nodos** representan los puntos de interacción con el usuario, proporcionándole un conjunto de datos y/o servicios (cohesivos) con los que el usuario puede interactuar; los **arcos** representan una alcanzabilidad entre nodos, indicando los posibles "siguientes" nodos (o caminos) que se puede alcanzar o seguir. Según esta visión, la navegación será todo "cambio de nodo provocado al activar un enlace de navegación". De esta manera podemos definir la navegación a nivel de modelado conceptual y se elimina la problemática de definirla en términos del espacio de la solución (número de peticiones al servidor, paginación (*scrolling*) de datos, ocultación parcial de información en la misma página, etc.).

En la aproximación OOWS, los requisitos navegacionales de una aplicación web se obtienen añadiendo una "vista navegacional" (*mapa navegacional*) sobre el Modelo de Objetos de OO-Method, indicando el conjunto posible de caminos navegacionales

que se le proporcionarán al usuario. Al definir este modelo dependiente del modelo de objetos, se crea una relación fuerte de dependencia entre ambos modelos, integrándolos de una manera clara y posibilitando la comprobación automática de propiedades semánticas del sistema. La semántica navegacional de las aplicaciones hipermediales se captura en función de cada agente del sistema identificado en el modelo de objetos, adaptando o personalizando el acceso en función de las necesidades de cada tipo de usuario. Estos mapas son descritos usando una notación intuitiva basada en UML [15].

El modelo de navegación está compuesto por un conjunto de **mapas de navegación** (uno por cada agente) que representan y estructuran la visión global del sistema para cada tipo de usuario, definiendo su navegación permitida. Éste se representa directamente usando un grafo dirigido en el cual los nodos son los *contextos navegacionales* y los arcos son los *enlaces (o vínculos) de navegación*. En este nivel de abstracción, sólo nos interesa especificar qué contextos conformarán nuestro mapa de navegación y desde dónde serán alcanzables. Existen dos posibilidades: que los nodos (contextos) navegacionales sean alcanzables desde cualquier ubicación en el sistema (llamados *contextos de exploración, E*) o que los nodos sólo sean alcanzables siguiendo un camino predeterminado de pasos de navegación (llamados *contextos de secuencia, S*). Los contextos de exploración definen unos enlaces de navegación implícitos (enlaces de exploración, representados por una flecha con líneas discontinuas) que surgen del agente y terminan en el contexto, indicando que ese agente siempre puede activar ese enlace y alcanzar el contexto de exploración. Además, se puede marcar uno de estos enlaces de exploración como *default o home* (aparece una "*H*" asociada al enlace), indicando que cuando el usuario se conecte, navegará automáticamente a este contexto. En la Figura 1 se puede apreciar un mapa de navegación genérico para el agente *Agente* donde aparecen dos contextos de exploración (marcados con la etiqueta *E*) y uno de secuencia (marcado con la etiqueta *S*).

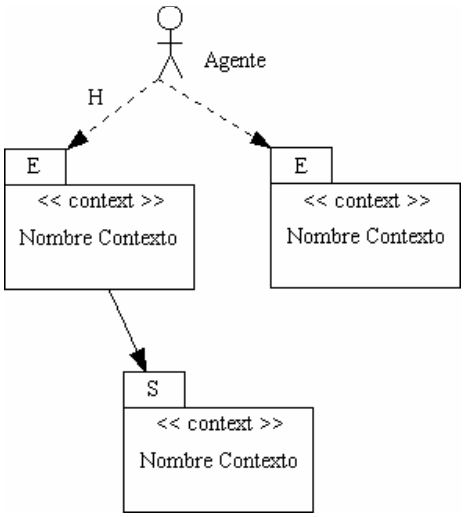

Fig. 1. Mapa de Navegación

Un **contexto navegacional** es una *Unidad de Interacción Abstracta* que representa una vista sobre un conjunto de datos y/o servicios (del esquema conceptual) accesible para un usuario en un determinado momento. Es una *Unidad* porque constituye el elemento lógico básico de creación de la navegación permitida en los mapas navegacionales. De *Interacción* porque representa una interacción con el usuario

(espera una acción/respuesta por parte del usuario, bien de navegación, bien de activación de un servicio), y *Abstracta* porque sólo se especifica qué datos y/o servicios se visualizarán en el contexto, pero no cómo se presentarán. Gráficamente se representa como un paquete UML estereotipado con la palabra «*context*». Está compuesto por un conjunto de **clases navegacionales**, estereotipadas con la palabra reservada «*view*» (Figura 2), que hacen referencia a clases identificadas en el modelo de objetos. Con ellas se puede definir la visibilidad (vista) ofertada al agente en este nodo, tanto de los atributos de la clase como de los servicios que puede activar. Este conjunto de atributos y servicios debe ser un subconjunto válido de atributos y servicios con los que el agente tenga definida una relación de agente en el modelo de objetos (relación de visibilidad de atributos y/o servicios en OO-Method).

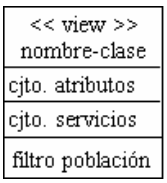

Fig. 2. Clase Navegacional

Asociado a los servicios pueden aparecer **enlaces de servicio**. Estos enlaces indican el contexto destino que se alcanzará después de la ejecución del servicio. Además, se puede especificar un **filtro de población**, que mediante una fórmula bien formada en función de atributos de la clase navegacional sobre la que se define, indican un filtrado de objetos que se llevará a cabo.

En un contexto navegacional debe aparecer una clase navegacional principal, llamada **clase directora** y opcionalmente otras que contribuyen en complementar la información de esta clase, llamadas **clases complementarias**. Las clases navegacionales están unidas entre sí por relaciones binarias unidireccionales que pueden ser definidas sobre una relación de agregación o de herencia existente entre las dos clases en el modelo de objetos[1]. En el caso de que exista más de una relación definida en el modelo de objetos entre estas dos clases, se deberá especificar una propiedad adicional de la relación que indique que rol se está utilizando, para eliminar la ambigüedad. Se pueden definir dos tipos de relaciones entre clases navegacionales: una **relación de dependencia de contexto** (se representa gráficamente mediante flechas discontinuas) que indica una recuperación de información relacionada de las instancias de la clase complementaria a través de la relación de agregación o herencia sobre la que está definida la relación, y una **relación de contexto** (gráficamente, flechas continuas), que es una relación de dependencia de contexto que además define una navegación a un nodo navegacional destino, causando la aparición de un vínculo de navegación en el mapa navegacional asociado. Las relaciones de contexto tienen las siguientes propiedades:

- *atributo de contexto*, que indica el contexto destino de la navegación
- *atributo de enlace*, que especifica qué atributo (normalmente de la clase navegacional final de la relación) se utilizará como *ancla* para activar la navegación al contexto destino

[1] No puede existir ninguna clase navegacional en un contexto que esté inconexa, es decir, que no esté relacionada con ninguna otra clase navegacional.

- *atributo de rol*, que indica el rol de la relación de agregación o herencia que estamos utilizando. Se utiliza para eliminar la ambigüedad en caso de existencia de más de una relación entre las dos clases. Es opcional si entre las dos clases existe una única relación definida en el modelo de objetos

Usando las primitivas descritas anteriormente es posible especificar completamente la semántica asociada a los requisitos de navegación para aplicaciones web. Sin embargo, y debido al carácter de Internet, nos hará falta especificar algunas propiedades adicionales de diseño para facilitar el acceso a la información deseada y que pueden llegar a ser cruciales para el sistema. Estas propiedades tienen que ver con la incorporación de *estructuras de acceso* y la definición de *mecanismos de búsqueda* de la población dentro de un contexto. Estas características se abordarán en el siguiente apartado.

2.3 Diseño Navegacional Avanzado

Una vez que se han construido los mapas navegacionales, la semántica navegacional del sistema ya ha sido capturada. Sin embargo, se pueden definir mecanismos adicionales que estructuren el acceso y permitan realizar búsquedas de información dentro de un nodo navegacional. Ambos mecanismos permitirán explorar y facilitar el acceso a la misma información, sin implicar navegación. Estas características son recogidas en la zona de características navegacionales avanzadas de un contexto de navegación (ver Figura 3).

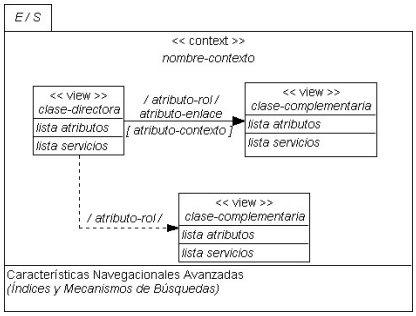

Fig. 3. Contexto de Navegación

Un **índice** proporciona un acceso *indexado* (por alguna propiedad propia o de un objeto relacionado) a los objetos principales del contexto (objetos de la clase directora). Por ejemplo, en un Museo Virtual podría existir un contexto que presentara las pinturas expuestas. En este contexto se podría definir un índice de acceso a las pinturas según corrientes pictóricas. Al seleccionar un elemento de este índice, se recuperarían todas las pinturas de esta corriente, dentro del mismo contexto.

Existen dos tipos de índices que pueden ser especificados (ver Figura 4):

– *índices de atributos (ATTRIBUTES-INDEX)*, que se definen sobre uno o varios atributos de la clase directora. Al menos uno de estos atributos actuará como *atributo de enlace* (atributo/s que servirá/n para activar el índice). Se creará un

índice donde aparecerán sólo los valores de los atributos especificados de la población de objetos del contexto. Se puede indicar que no aparezcan duplicados (*DISTINCT VALUES*), para que sólo se recuperen los valores distintos.

– *índices de relación (RELATION-INDEX)*, que se definen sobre uno o varios atributos de una clase relacionada en el modelo de objetos con la clase directora del contexto. Al menos uno de estos atributos servirá como *atributo de enlace* (atributo/s que servirá/n para activar el índice). Si existe más de una relación entre la clase directora y la clase del índice en el modelo de objetos, será necesario especificar un *atributo de rol* para eliminar la ambigüedad. Se creará un índice con todos los valores de atributos de los objetos de la clase relacionada. Se puede indicar que no aparezcan duplicados (*DISTINCT VALUES*), para que sólo se recuperen los valores distintos.

ATTRIBUTES INDEX *nombre*
  ATTRIBUTES *lista-atributos*
  LINK-ATTRIBUTES *lista-atributos*
  [DISTINCT VALUES]

RELATION INDEX *nombre*
  ON RELATION *rol-relación*
  ATTRIBUTES *lista-atributos*
  LINK-ATTRIBUTES *lista-atributos*
  [DISTINCT VALUES]

Fig. 4. Plantillas de definición de *índices* asociados a contextos

Adicionalmente, sobre un contexto se pueden especificar **mecanismos de búsqueda**, expresados como filtros de información. Estos filtros permiten restringir el espacio de objetos de la clase directora recuperados en función de una expresión-condición basada sobre alguna propiedad (atributo) de la clase directora u obtenida por una relación entre clases. Esta expresión se puede especificar en tiempo de modelado o la puede introducir el usuario en tiempo de ejecución. La Figura 5 muestra la plantilla de definición de un filtro. Existen tres tipos:

– *exacto*, que toma un único valor y devuelve el conjunto de instancias de la clase directora cuyo valor de atributo coincida exactamente con el valor indicado por el usuario
– *aproximado*, que toma un único valor y devuelve el conjunto de instancias de la clase directora cuyo valor de atributo sea semejante al valor indicado por el usuario
– *rango*, que toma dos valores, un máximo y un mínimo y devuelve conjunto de instancias de la clase directora cuyo valor de atributo esté entre estos valores[2]. Si sólo se especifica uno de estos valores, se acota únicamente por un extremo

FILTER *nombre*
  ATTRIBUTE *atributo*
  TYPE { *exact* | *like* | *range* }
  { EXPRESSION *condición* }

Fig. 5. Plantilla de definición de un *filtro* de contexto

[2] Este atributo solo tiene sentido en atributos de tipo numérico o de texto.

La Figura 6 muestra un ejemplo de un filtro completamente especificado en tiempo de modelado que recupera los libros *best-sellers*. Se puede considerar que un libro es un best-seller cuando se han vendido más de 1.000.000 de copias. Al seleccionar este filtro, la población de libros se filtraría por aquellos libros que cumplen esta condición.

```
FILTER libros_best-sellers
   ATTRIBUTE vendidos
   TYPE range
   EXPRESSION vendidos > 1.000.000
```

Fig. 6. Filtro especificado completamente en tiempo de modelado

Otras primitivas que se pueden especificar en el modelo de navegación OOWS:

- *Sesión*. Primitiva de modelado que nos permite realizar cierto control cuando un usuario se conecta al sistema. Existen básicamente dos momentos donde se pueden lanzar operaciones: ejecutar un conjunto de servicios cuando el usuario inicia o cuando termina la sesión de interacción con el sistema. El agente de estos servicios es siempre el propio usuario. La sintaxis es:

#Clase.Servicio1#	##Clase.Servicio1##
#Clase.Servicio2#	##Clase.Servicio2##
...	...
Eventos de inicio de sesión	*Eventos de fin de sesión*

- *Agente conectado*. Esta primitiva proporciona capacidad expresiva para acceder al usuario conectado a la aplicación en tiempo de ejecución. Esta expresividad es la base para personalizar el acceso y la recuperación de información según el usuario concreto. Se puede utilizar en fórmulas de filtro, en condiciones de navegación, etc. Se representa mediante el término *#Clase_Agente#.*
- *Condición de navegación*. La navegación capturada mediante las relaciones de contexto se caracteriza por ser estática, es decir, enlaza unívocamente dos contextos navegacionales. Sin embargo, dada la naturaleza de las aplicaciones web se hacen necesarios mecanismos que expresen dinámicamente condiciones de navegación que sean evaluadas en tiempo de ejecución. Éstas condiciones permiten especificar restricciones que deben satisfacerse para que se pueda producir una navegación. Se expresan en las relaciones de contexto, mediante el atributo de contexto, explicitando la condición o condiciones que deben cumplirse para alcanzar el contexto destino.

Mediante el uso de estas características avanzadas, se pueden complementar algunos requisitos navegacionales adicionales del sistema. A continuación vamos a proponer un modelo que capture los requisitos de presentación de información del sistema.

2.4 Modelo de Presentación

Una vez definido el modelo de navegación que captura la semántica navegacional del sistema, debemos asociar características de presentación al sistema. Para ello se introduce un nuevo modelo, el Modelo de Presentación, que complementa la información capturada en el modelo de navegación para la creación de interfaces con información de presentación. En este modelo se utilizan los nodos o contextos navegacionales como entidades básicas donde se definen estas propiedades de presentación adicionales.

La manera de especificar los requisitos de presentación se basará en el uso de unos patrones de presentación simples que se podrán asociar a los distintos elementos que forman un nodo de navegación. Existirán propiedades de presentación que podrán ser aplicadas a nivel de un contexto de navegación, a nivel de estructuras de acceso y/o mecanismos de búsquedas (filtros), y a relaciones (navegacionales) entre clases.

Los patrones de presentación de información que se pueden especificar son:

- *Paginación de información*. Este patrón permite capturar la semántica *scrolling* de información. Cuando se especifica paginación, el conjunto de instancias que deban ser presentadas serán "troceadas" en "bloques lógicos", de manera que en una misma "pantalla" sólo aparezcan unas cuantas instancias del conjunto de todas las posibles. Se proporcionarán mecanismos para avanzar y retroceder entre las distintas páginas lógicas que se obtienen, pudiendo especificar el tipo de paginación como *secuencial* (se proporciona acceso al siguiente, al anterior, al primero y al último bloque), o *aleatorio* (cuando además de los anteriores, el usuario puede acceder directamente a un bloque intermedio). Se podrá indicar además una *cardinalidad* en la paginación, que indicará el número de instancias que se recuperarán. Puede ser de dos tipos: *estática*, cuando se define en tiempo de modelado y *dinámica*, cuando se permite modificar en tiempo de ejecución por el usuario. En este último caso se indicará un número de instancias a mostrar por defecto (que podrá ser modificable) o una selección de valores posibles entre los que el usuario deberá elegir. Otra propiedad de la paginación es la *circularidad*. Si se pagina con esta propiedad activada, la siguiente instancia de la última instancia será la primera instancia, y la instancia anterior a la primera instancia será la última instancia. Cuando se define que un contexto está paginado, las instancias sobre las que se pagina son las de la clase directora. Si se aplica a una estructura de acceso índice o un filtro de búsqueda, se pagina el conjunto de resultados obtenidos. Por último, se puede indicar paginación sobre una relación navegacional (relación de dependencia de contexto y relación de contexto) entre dos clases cuando la cardinalidad en el modelo de objetos de la clase complementaria destino sea "muchos". En este caso se crea una paginación que se aplicará únicamente sobre el conjunto de objetos relacionados que se presentarán.
- *Ordenación*. Este patrón permite definir una ordenación de la población de una clase según el valor de uno o más atributos sobre los que se aplica. La ordenación puede ser: *ascendente (asc)* o *descendente (desc)*. En el caso de especificar varios atributos, cada atributo tendrá un carácter de ordenación ascendente o descendente, y ésta se aplicará jerárquicamente, empezando por el primer atributo, y siguiendo por los demás sucesivamente. Este patrón se puede aplicar a clases navegacionales,

indicando cómo se van a ordenar las instancias de la clase (los atributos sobre los que se ordene deben aparecer especificados en el contexto); análogamente al caso anterior, se pueden aplicar a estructuras de índices y a filtros de búsqueda, ordenando los resultados obtenidos por alguno/s de los atributo/s especificado/s por estos mecanismos.

- *Patrón de presentación*. Existen cuatro modos: *registro, tabular, maestro-detalle* (pudiendo indicar en éste último caso como presentar el elemento detalle, recursivamente) y *árbol*. Se pueden aplicar a:
  - *relaciones de navegación* (relación de dependencia de contexto y relación de contexto). El patrón de presentación definirá el modo en que la información de las instancias relacionadas será presentada. Los patrones *registro* y *tabular* son indicados para relaciones "1 a 1", mientras que los dos últimos son adecuados para relaciones "1 a muchos" ó "muchos a muchos". El patrón *árbol* es también muy adecuado para representar relaciones reflexivas.
  - la *clase directora*, indicando el modo en que se verán las instancias de esta clase. El patrón *maestro-detalle* no se puede aplicar directamente a esta clase.

Con estos patrones sencillos de presentación de información, combinados con la información de navegación definida en el modelo de navegación, podemos capturar los requisitos básicos para la construcción de interfaces del sistema, a nivel de modelado conceptual. Este repositorio de información será utilizado por el generador (compilador) para generar las distintas interfaces para cada usuario, dentro de la arquitectura de aplicación web que propone el método OOWS.

En el siguiente apartado se va a presentar un caso de estudio donde se ha aplicado el método propuesto, siguiendo el proceso definido de construcción de aplicaciones web. Al final se presenta un ejemplo de interfaz generada teniendo en cuenta las primitivas especificadas en los modelos para este caso de estudio.

3 Un caso de estudio: La tienda virtual de venta de discos *DiscoWeb*

En esta sección se aplica la aproximación OOWS en el desarrollo de una aplicación web para la compra discográfica por Internet. Los requisitos del sistema se presentan a continuación:

"El sistema está orientado a la venta *on-line* de álbumes de música, organizados por categorías. Existen dos tipos de usuarios de esta aplicación: los Administradores y los Usuarios Navegantes. Los primeros son los encargados de la gestión básica de los álbumes que se ponen en venta. Los segundos son los usuarios comunes (compradores) de esta aplicación. La funcionalidad del sistema es:

- (*Usuarios Navegantes*). Las compras que se realicen se deberán ir incluyendo, simbólicamente, en una cesta de la compra; el usuario podrá consultar en cualquier momento el contenido de su cesta y realizar modificaciones sobre su contenido. Esta cesta de la compra se creará en el momento en el que se reciba la petición de entrada en el sistema y pertenecerá al usuario que está navegando en ese momento;

todas las operaciones que el usuario realice sobre el sistema se harán de forma anónima, de modo que el usuario no deberá identificarse (registrarse) hasta que no vaya a confirmar su compra; para comprar un álbum se deberá llegar a él a través del autor o de la categoría a la que pertenece; desde la página de inicio podremos acceder a un listado de autores o a un listado de categorías y desde ahí al listado de los álbumes del autor o de la categoría que hayamos seleccionado; cuando seleccionemos un álbum de la lista, se mostrarán todos los datos de ese álbum y se podrá comprar. Esto hará que el álbum sea incluido en la cesta de la compra de ese usuario y que se muestre su contenido actual; mientras veamos el contenido de la cesta, podremos cambiar el número de unidades que se desea adquirir de cada álbum de los comprados hasta el momento o eliminar alguna de las compras de la cesta; cuando se decida confirmar la compra se realizarán dos acciones: la primera consistirá en crear una factura (para lo que el comprador debe haberse identificado) y la segunda será reducir el stock de los álbumes comprados; cuando se haya confirmado una compra, ya no se podrá modificar el contenido de la cesta.

– (*Administradores*). Gestionar y mantener el catálogo de productos del sistema, así como mantener la lista de autores que poseen discos en la tienda".

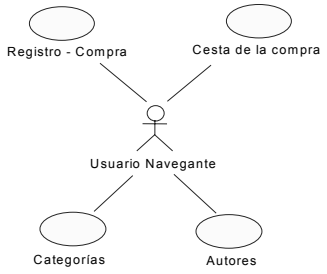

Fig. 7. Caso de uso del agente *Usuario Navegante*

El caso de estudio empieza con la elicitación de los requisitos del sistema en la fase de **Especificación del Problema.** El sistema se divide en la funcionalidad relevante y estos requisitos son descritos mediante un diagrama de casos de uso. En la Figura 7 se muestra el diagrama de caso de uso para el agente Usuario Navegante. Del mismo modo se han descrito los requisitos para el agente Administrador (no aparecen en este artículo por falta de espacio y por carecer de aspectos interesantes). En la fase de modelado conceptual se construyen los siguientes modelos: Modelo de Objetos, Modelo Dinámico, Modelo Funcional, Modelo Navegacional y Modelo de Presentación. Estos dos últimos recogen las extensiones incorporadas a OO-Method.

La Figura 8 muestra el **Modelo de Objetos** del caso de estudio. La *cesta* de la compra participa directamente en el proceso de compra *on-line* del internauta *Usuario Navegante*. Para su construcción se han estudiado los requisitos funcionales del sistema desde un punto de vista OO, y se ha asociado esta funcionalidad a cada usuario, según los casos de uso planteados (esta información no está visible por motivos de espacio del artículo).

Después del Modelo de Objetos, se construye el **Modelo Dinámico** donde se describen las vidas válidas de los objetos representando el comportamiento de cada clase del sistema según la interpretación descrita por OO-Method. La Figura 9 presenta una porción del Modelo Dinámico de la clase *Cesta.*

El **Modelo Funcional** captura la semántica asociada a los cambios de estado de los objetos. El valor de cada atributo es modificado dependiendo de la acción que activó el cambio de estado, de los argumentos de dicho evento y del estado actual del objeto.

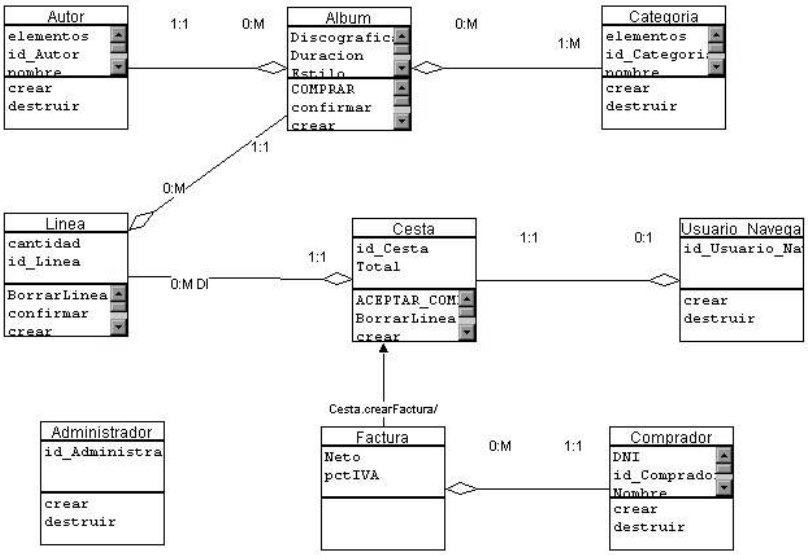

Fig. 8. Modelo de Objetos de *DiscoWeb*

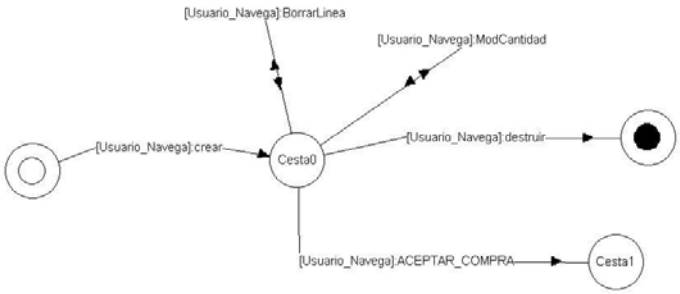

Fig. 9. Porción del Modelo Dinámico de la clase *Cesta*

La Figura 10 muestra un ejemplo para la clase *Linea*, donde se halla definida la siguiente evaluación:

```
Atributo: Cantidad
Categoría: De Estado     Evento: ModCantidad(p_nuevaCantidad)
Efecto: = p_nuevaCantidad      Condición: p_nuevaCAntidad > 0
```

Fig. 10. Parte del Modelo Funcional de la clase *Linea*

A continuación se construye el **Modelo de Navegación** donde se estructurará el acceso de cada usuario al sistema. La Figura 11 presenta el mapa de navegación del agente *Usuario Navegante* con sus contextos de navegación que han sido identificados en las primeras fases de especificación del sistema. También aparecen sobre el mapa los servicios que son ejecutados al iniciar y finalizar una sesión. Cuando el servidor Web recibe una petición de un internauta, ejecuta el servicio *crear* del *Usuario Navegante* asociándole además una *Cesta*. Cuando el *Usuario Navegante* abandona el sistema se ejecuta el servicio *destruir*, eliminando además, si no ha sido confirmada, su *Cesta* asociada.

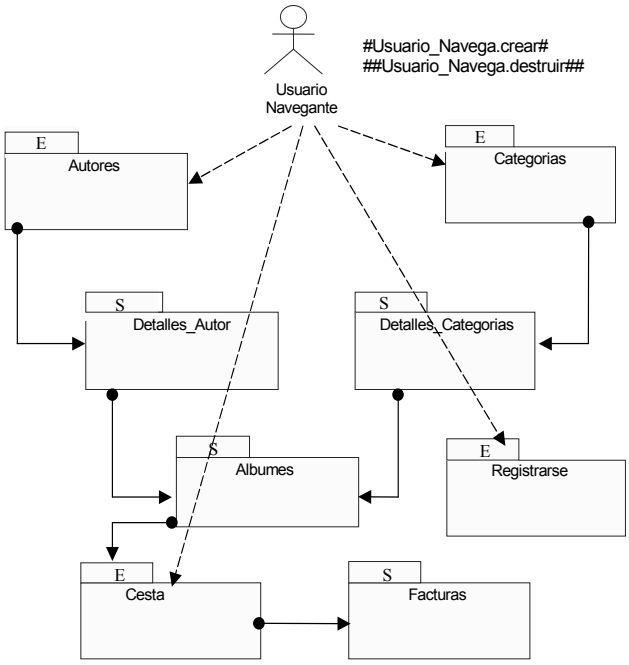

Fig. 11. Mapa de Navegación del agente *Usuario Navegante*

En este mapa de navegación se aprecia que el *Usuario Navegante* siempre tendrá disponibles los contextos (marcados como contextos de exploración) *Autores, Categorías, Cesta* y *Registrarse*. A partir de estos, y siguiendo diferentes caminos navegacionales, podrá alcanzar los demás (*Detalles_Autor, Detalles_Categoría, Albumes y Facturas*).

En la Figura 12 se describe con detalle el contexto *Autores*, donde se recupera la información sobre un autor (su nombre), los álbumes que están disponibles de este autor (título, año y precio) y el nombre de la categoría del álbum. Seleccionando el título de un álbum podremos navegar al contexto *Álbumes*, donde se proporcionará información adicional del álbum y podremos comprarlo. Además, se ha definido una estructura de acceso índice de tipo atributo, que permitirá acceder a los autores por su

letra_inicial (atributo derivado definido en la clase *Autor*). También se ha definido un filtro de tipo aproximado para facilitar la búsqueda de autores por su nombre.

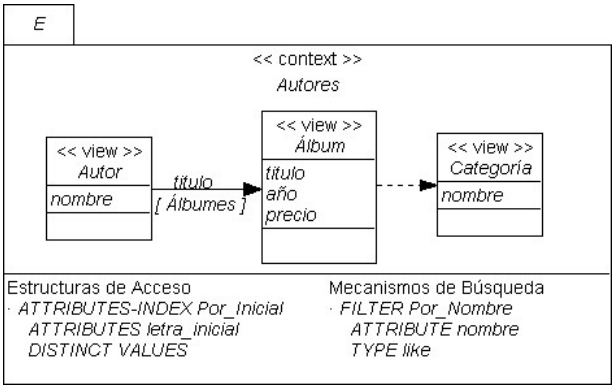

Fig. 12. Contexto *Autores* para el agente *Usuario Navegante*

Finalmente, se construye el **Modelo de Presentación** donde se captan los requisitos de presentación de información para cada contexto del mapa de navegación. En la Figura 13 se muestra como ejemplo la plantilla de presentación asociada al contexto *Autores*. En ella se especifica que los objetos de la clase directora se presentarán en modo *tabular* y el contexto (objetos de la clase directora) estará paginado con una cardinalidad estática de 1 elemento, con posibilidad de acceso *secuencial* y *circular*. El patrón de presentación asociado a la relación de contexto definida entre un *Autor* y sus *Album*es será *maestro-detalle*, con el detalle en modo *tabular* y con una paginación de cardinalidad estática de 5 elementos, con acceso *secuencial*, *circular.* Se ha definido una ordenación de los elementos de la clase *Album* por el *año* de modo *ascendente* y la relación de contexto definida entre la clase *Album* con la clase *Categoría* se presentará en modo *tabular* (relación "1 a 1").

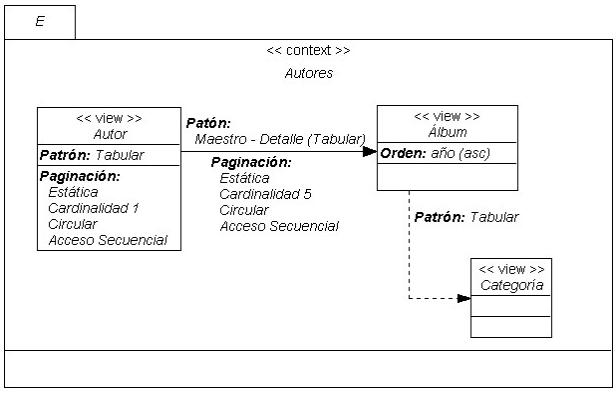

Fig. 13. Información adicional de presentación del contexto *Autores*

Después de construir estos modelos, el proceso llega a su segunda fase, **Desarrollo de la Solución**, donde en una estrategia de compilación de modelos, se obtiene el prototipo software completo de manera automática, siguiendo la especificación realizada del sistema. En la Figura 14 aparece una posible interfaz de usuario que representa correctamente los requisitos, tanto navegacionales como de interfaz, especificados para el agente *Usuario Navegante*.

Fig. 14. Página generada a partir del contexto navegacional *Autores*

Como se puede apreciar en este contexto generado (Figura 14), esta implementación recoge las características navegacionales y de presentación de información que se habían especificado en la definición del contexto (Figuras 12 y 13). Se puede apreciar que existe un enlace desde esta página (en realidad desde cualquier página, ya que se el *frame* izquierdo es común a todas las páginas) a cada uno de los contextos de exploración especificados (*Autores, Categorías, Cesta* y *Registrarse*). Esta página proporciona información sobre un autor y sus álbumes disponibles, mostrando su título, año, precio y categoría. Además, aparece el índice que se había especificado, usando el atributo *letra_inicial* de la clase *Autor* y también el mecanismo de búsqueda de autores por su atributo nombre. Si nos fijamos en aspectos de presentación de información, la paginación del contexto (objetos de la clase directora *Autor*) se realiza elemento en elemento (cardinalidad 1). Es por esto que sólo nos aparece información sobre un autor o grupo (*Queen*) y se nos proporcionan mecanismos para avanzar o retroceder secuencialmente. La información complementaria sobre los álbumes para este grupo aparece en modo *maestro-detalle* (con el detalle en modo *tabular*) y además paginado de 5 en 5 elementos, como había sido explicitado.

4 Conclusiones

En este artículo hemos presentado una propuesta metodológica para la construcción de aplicaciones web dentro un marco de trabajo OO en ambientes de compilación automática de modelos. Este marco definido integra formalmente diferentes modelos para dar un soporte completo a la especificación de este tipo de sistemas software, además de definir un proceso metodológico guiado. El trabajo realizado ha consistido en extender un método formal existente OO (OO-Mehod), dotándolo de la expresividad necesaria para construir soluciones web. Esta expresividad ha sido capturada en: un *Modelo de Navegación* que captura la estructura navegacional asociada a cada usuario del sistema, proporcionando el acceso a la información, y a la funcionalidad, permitiendo definir estructuras de acceso y mecanismos de búsqueda que faciliten el acceso a la información deseada; un *Modelo de Presentación*, que refleja las características esenciales de presentación de información de los nodos navegacionales mediante (unos pocos) patrones básicos de presentación.

Se ha aplicado el método a un caso de estudio, siguiendo el proceso propuesto de construcción de un sistema web. Adicionalmente, se han realizado trabajos orientados a dar soporte a:

- la introducción de características de personalización a nivel de modelado conceptual, para crear sistemas adaptativos
- la definición de operadores conceptuales orientados al reuso conceptual
- la construcción de interfaces web declarativas, basadas en XML

Actualmente estamos aplicando el método a numerosos casos de estudio reales, extendiéndolo incorporando nuevas primitivas navegacionales detectadas, y definiendo sus patrones de traducción software a plataformas destino.

En este ámbito se está se rediseñando la estrategia de generación de código basada en componentes para adaptarla a una arquitectura software destino y completamente distribuida, basada y orientada en servicios web.

Referencias

[1] Abrahao S., Fons J., and Pastor O. *Conceptual Modeling of Personalized Web Applications*. Springer–Verlag (LNCS). Editores: Paul De Bra, Peter Brusilovsky, Ricardo Conejo. 2nd Intenational Conference on Adaptive Hypermedia and Adaptive Web Based Systems. Málaga, España, Junio, 2002

[2] Amazon.com Tienda Virtual de Comercio Electrónico. http://www.amazon.com

[3] Baresi L., Garzotto F., Paolini P. *From Web Sites to Web Applications: New Issues for Conceptual Modeling*. ER'2000 Workshop on Conceptual Modeling and the Web, LNCS 1921. Springer-Verlag, 2000

[4] Baumeister H., Koch N., and Mandel L. *Towards a UML extension for Hypermedia Design.* In UML'99 The Unified Modeling Language, Springer Verlag. LNCS 1723. Fort Collins, USA, Octubre 1999.

[5] Cachero C., Garrigós I., and Gómez J. *Personalización de Aplicaciones en OO-H*. In Proc. V Workshop Iberoamericano de Ingeniería de Requisitos y Desarrollo de Ambientes de Software, IDEAS'02. ISBN: 959-7164-08-6. La Habana, Cuba, Abril, 2002

[6] Ceri S., Fraternali P., Bongio A. *Web Modeling Language (WebML): a Modeling Language for Designing Web Sites*. In WWW9, Vol. 33 (1-6), pp 137-157. Computer Networks, 2000

[7] De Troyer O. and Leune C. *WSDM: A user-centered design method for Web sites*. In Proc. of the 7th International World Wide Web Conference, 1997

[8] Fons J., Valderas P., and Pastor O. *Specialization in Navigational Models*. Anales JAIO, 31st. Argentine Conference on Computer Science and Operacional Research. Iberoamerican Conference on Web Engineering 2002 (ICWE'02), Santa Fe, Argentina, Septiembre, 2002

[9] Garzotto F., Paolini P. and Schwabe D. *HDM - A Model-Based Approach to Hypertext Application Design*. ACM Transactions on Information Systems, vol. 11 (1), pp. 1-26. 1993

[10] Gómez J., Cachero C., and Pastor O. *Extending a Conceptual Modeling Approach to Web Application Design*. Proc. Conference on Advanced Information Systems Engineering (CAiSE'00), Springer- Verlag, LNCS 1789, pp. 79-93, 2000

[11] Isakowitz T., Stohr E. A., and Balasubramanian V. *RMM: A Methodology for Structured Hypermedia Design*. CACM: Communications of the ACM., pages 34-44, Agosto, 1995

[12] Jacobson I., Christerson M., Jonsson P., Overgaard G. *OO Software Engineering , a Use Case Driven Approach*. Reading, Massachusetts. Addison-Wesley, 1992

[13] Lange D. *An Object-Oriented Design Method for Hypermedia Information Systems*. Proc. Hawaii International Conference on System Science, 1994

[14] Mecca G., Merialdo P., Atzeni P. and Crescenzi V. *The Araneus Guide to Web-Site Development*. Technical Report, University of Roma. 1999

[15] Object Management Group. *Unified Modeling Language Specification Version 1.4 draft*. Technical report, www.omg.org, February 2001

[16] Paolini P. and Fraternali P. *A Conceptual Model and a Tool Environment for Developing More Scalable, Dynamic, and Customizable Web Applications*. In Proc. EDBT98 Conference, Valencia, España, Marzo, 1998

[17] Pastor O., Insfrán E., et. al. *OO-Method: An OO Software Production Environment Combining Conventional and Formal Methods*. Proc. Conference on Advanced Information Systems Engineering (CAiSE), LNCS 1250, Springer- Verlag, pp. 145-159, 1997

[18] Pastor O., Abrahao S., and Fons J. J. *An Object-Oriented approach to automate web applications development*. In 2nd International Conference on Electronic Commerce and Web Technologies (EC-Web'01), Springer-Verlag, LNCS 2115. ISBN: 3-540-42517-9. Munich, Germany, Septiembre, 2001

[19] Pastor O., Pelechano V., Insfrán E,. and Gómez J. *From Object Oriented Conceptual Modeling to Automated Programming in Java*. 17th International Conference on Conceptual Modeling (ER'98). Springer-Verlag, LNCS 1507, pp. 183-196. Singapore, November, 1998

[20] Pastor O., Abrahao S., and Fons J. *Building E-Commerce Applications from Object-Oriented Conceptual Models*. Newsletter of the ACM SIGecom Exchanges, volume 2.2, ACM Press, pp. 24-32. June, 2001

[21] Rossi G., Schwabe D., and Guimaraes R. *Designing personalized web applications*. In Proc. of the WWW10, Hong Kong, May 2001

[22] Schwabe D., Rossi G. and Barbosa D.J. *Systematic Hypermedia Application Design with OOHDM*. Proc. ACM Conference on Hypertext. pp.166. 1996

Métricas de Usabilidad y Sistemas Multiagente en Hipermedia Adaptativa

Víctor López-Jaquero y Antonio Fernández-Caballero

Laboratorio de Interacción de Usuarios e Ingeniería del Software (LoUISE)
Instituto de Investigación en Informática de Albacete (I3A)
Universidad de Castilla-La Mancha
02071 - Albacete, España
{victor, caballer}@info-ab.uclm.es

Resumen. En los años más recientes una nueva conceptualización del fenómeno computacional ha dirigido el énfasis hacia la interacción en lugar del procedimiento. La interacción hombre-computador en el desarrollo de aplicaciones tradicional se enfoca en la interacción entre las tareas y una única interfaz de usuario diseñada para un único tipo de usuario. Una evolución lógica debiera llevar la interacción a un modelo de desarrollo en el que las características y las preferencias del usuario fueran tomadas en cuenta [13] [14]. Existen diferentes tipos de usuarios y éste es un hecho que no debe ser ignorado. La sociedad humana está llena de diversidad y esto debe verse reflejado en la interacción hombre-computador. En este artículo se proponen métricas de usabilidad, a saber las de preferencia y de rendimiento, y sistemas multiagente para la adaptación de la interfaz de usuario en entornos de hipermedia adaptativa. La propuesta incluye un ejemplo práctico para el caso del aprendizaje/enseñanza de una asignatura de ingeniería.

1 Introducción

En computación muchas cosas no están cambiando en absoluto [5]. La idea básica de lo que es un ordenador, lo que hace y cómo lo hace, prácticamente no ha cambiado en las últimas décadas. El incremento de la capacidad computacional y el contexto siempre creciente en el que se inserta dicha capacidad, sugieren que hay que buscar nuevas vías de interactuar con las computadoras. Estas vías debieran estar más ajustadas a nuestras necesidades y habilidades. En los años más recientes una nueva conceptualización del fenómeno computacional ha dirigido el énfasis hacia la interacción en lugar del procedimiento [19]. Se enfatiza la diversidad y la especialización en vez de la unidad y la generalidad. De modo que está siendo influida la manera en cómo procesamos los modelos computacionales en nuestras mentes, tal y como reflejan la "Society of Mind" de Minsky [15], la crítica del razonamiento computacional de Agre [1] o la aproximación a la robótica de Brooks [2].

La interacción hombre-computador (HCI, *Human-Computer Interaction*) en el desarrollo tradicional de aplicaciones está enfocada a la interacción entre una tarea y una única interfaz de usuario diseñada para un único tipo de usuario. La masa de usuarios de la aplicación es tratada como una única entidad, sin distinguir entre los diferentes estereotipos de usuarios incluidos en esa gran masa de usuarios (figura 1a). Una evolución lógica debiera llevar la interacción a un modelo de desarrollo donde se tuvieran en cuenta estos estereotipos. Existen diferentes tipos de usuarios y éste es un hecho que no debe ser ignorado. La sociedad humana está llena de diversidad y esto debe verse reflejado en el diseño de la HCI (figura 1b).

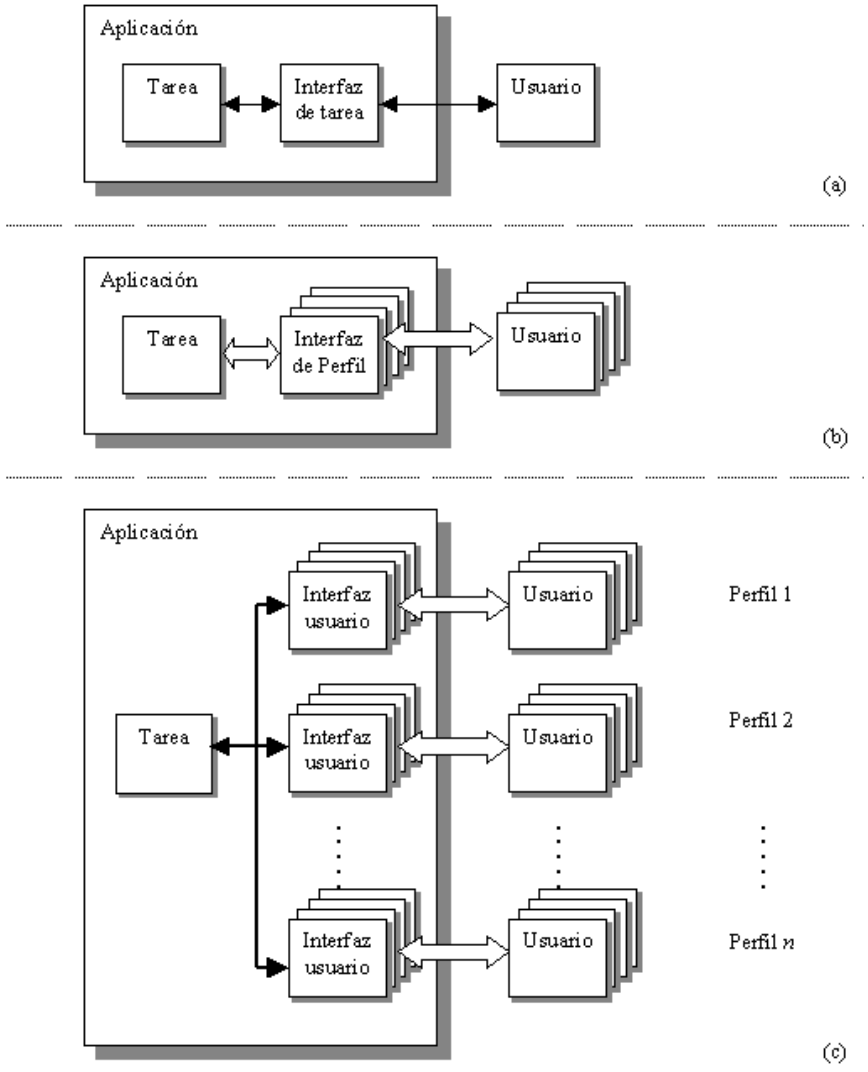

Fig. 1. (a) Unidad, (b) diversidad, y, (c) especialización en interacción.

Sin embargo, es necesario un paso más hacia adelante en el diseño de la interacción para trasladar esta diversidad al desarrollo de aplicaciones. El añadir soporte para distintos perfiles de usuario es, por supuesto, más exacto que desarrollar aplicaciones para un único tipo de usuario. Pero la realidad es que todos los usuarios son algo diferentes. Si tenemos en cuenta las particularidades de cada usuario, nos vamos al concepto de especialización (figura 1c). De este modo, nos adentramos en un nuevo concepto de interacción en el que las interfaces de usuario son hechas a medida para cada usuario, y donde las interfaces de usuario son inteligentes y adaptativas.

Como ya se ha comentado, debemos capturar las características y las preferencias del usuario a partir de su interacción con el sistema. Ello nos va a conducir a un proceso de inspección que nos va a llevar a determinar cómo el usuario "usa" la aplicación. Aquí es donde las métricas de usabilidad pueden trabajar a nuestro favor. Se presenta, por tanto, en este artículo cómo las métricas de usabilidad aplicadas a entornos hipermedia pueden llevar a sistemas de aprendizaje/enseñanza altamente adaptativos.

Con el fin de conseguir el objetivo marcado de crear un entorno de aprendizaje altamente adaptativo, nuestra propuesta es también que será necesario aplicar algunas técnicas de inteligencia artificial que pueden ser modeladas por medio de sistemas multiagente (MAS, *Multi-Agent Systems*) para conseguir un comportamiento inteligente.

2 Métricas de Usabilidad

Las métricas de usabilidad son métricas de calidad del software con una larga historia de aplicaciones exitosas dentro del campo de la ingeniería del software [3] [6] [8]. Pero, las métricas también acarrean riesgos [4]. Ningún simple número puede representarlo todo completamente, y en especial, algo tan sutil y complejo como es la usabilidad de un sistema software. No obstante, los números pueden algunas veces crear cierta ilusión de comprensión.

Hasta este momento, las métricas de usabilidad tienen una gran variedad de usos, pero casi siempre desde el punto de vista del diseñador. Podemos clasificar las métricas de usabilidad en tres grandes categorías: las métricas de preferencia, que cuantifican la evaluación subjetiva de las preferencias de los usuarios, las métricas de rendimiento, que miden el uso real del software, y las métricas predictivas (o métricas de diseño), que evalúan la calidad de los diseños y los prototipos.

2.1 Métricas de Preferencia

Uno de los métodos más populares para valorar la usabilidad es usar métricas de preferencia. La satisfacción del usuario es una componente de la usabilidad y un factor importante en la puesta en el mercado de un producto.

Un buen ejemplo de un conjunto estandarizado de métricas de preferencia es el *Software Usability Measurement Inventory* (SUMI) desarrollado como parte de un proyecto ESPRIT [18]. SUMI es un cuestionario de 50 preguntas que incluye cinco apartados que miden distintos aspectos de la usabilidad: atracción (cuanto le gusta el diseño al usuario), eficiencia (cómo de bien el software permite el uso productivo de la aplicación), capacidad de ayuda (cuanto soporte se incluye en el software y en la documentación), control (cuan consistente y normal es la respuesta del software) y capacidad de aprendizaje (cuan fácil es explorar y hacerse con el software). Otra conocida aproximación es el cuestionario *Subjective Usability Scales for Software* (SUSS), que mide seis elementos clave de la interfaz de usuario y que afectan a la usabilidad: valencia (gustos o preferencias personales), estética (capacidad de atracción), organización (diseño gráfico y formato), interpretación (capacidad de comprensión), adquisición (facilidad de aprendizaje) y facilidad (sencillez general de uso).

Las métricas de preferencia son también uno de los pilares para la personalización de la interfaz de usuario. No obstante, debido a su propia naturaleza, son difíciles de aplicar en tiempo de ejecución. Generalmente, se utilizan cuestionarios para evaluar las métricas de preferencia. Hasta la fecha no se han visto interesantes para la captura de las preferencias de los usuarios en tiempo de ejecución, ni tan siquiera en investigación de interfaces inteligentes. Hasta ahora solamente han mostrado su valía en la fase de diseño. Creemos, no obstante, que algunas métricas de preferencia pueden ser especialmente útiles para capturar las preferencias de los usuarios.

2.2 Métricas de Rendimiento

Las métricas de rendimiento, por su parte, pueden entenderse como índices de varios de los aspectos de cómo los usuarios realizan su trabajo (real o simulado). Los estudios de medidas forman la base de gran parte de la investigación acerca de los factores humanos presentes en la interacción con el ordenador. El rendimiento de los usuarios casi siempre se mide teniendo un grupo de usuarios de prueba realizando un conjunto predefinido de tareas de prueba [17].

Algunas medidas de usabilidad típicas cuantificables serían: el tiempo que un usuario tarda en completar una tarea; el número de tareas de diferentes tipos que pueden completarse en un límite de tiempo dado; la razón entre las interacciones exitosas y los errores cometidos por el usuario; el tiempo transcurrido para recuperarse de los errores; el número de errores de usuario; etc. [17]. Por supuesto, solamente un subconjunto de las medidas ideadas deben recogerse durante cualquier estudio en particular.

Las métricas de rendimiento son especialmente útiles para valorar la usabilidad en general. Una clave en este tipo de métricas es que la mayoría de ellas pueden ser evaluadas en tiempo de ejecución de un modo sencillo. Las métricas de rendimiento son un parámetro más para avanzar hacia las interfaces adaptables al usuario. Nuestra propuesta es dejar atrás las interfaces de usuario en las que es el usuario el que debe adaptarse a una interfaz fija dada.

3 Sistemas Multiagente de Interacción Adaptativos

La necesidad de adaptatividad de las interfaces al usuario tiene también una base importante en la necesidad de motivar al usuario. Para construir un sistema capaz de elevar la motivación del usuario frente a su ordenador, se necesita una vigilancia prácticamente constante del mismo, y trabajar con conceptos largamente estudiados en inteligencia artificial – conocimiento, inteligencia, memoria y capacidad de razonamiento – equivalentes a los que posee un ser humano. Un estudio bastante reciente concluye que una persona que se siente frustrada, rápidamente pierde capacidad de atención [11], de memoria [10], de aprendizaje [12], de pensamiento creativo [9] y de interacción social educada [7], entre otras. De igual modo, un episodio frustrante en la interacción con un computador predispone negativamente al usuario frente al sistema para siempre.

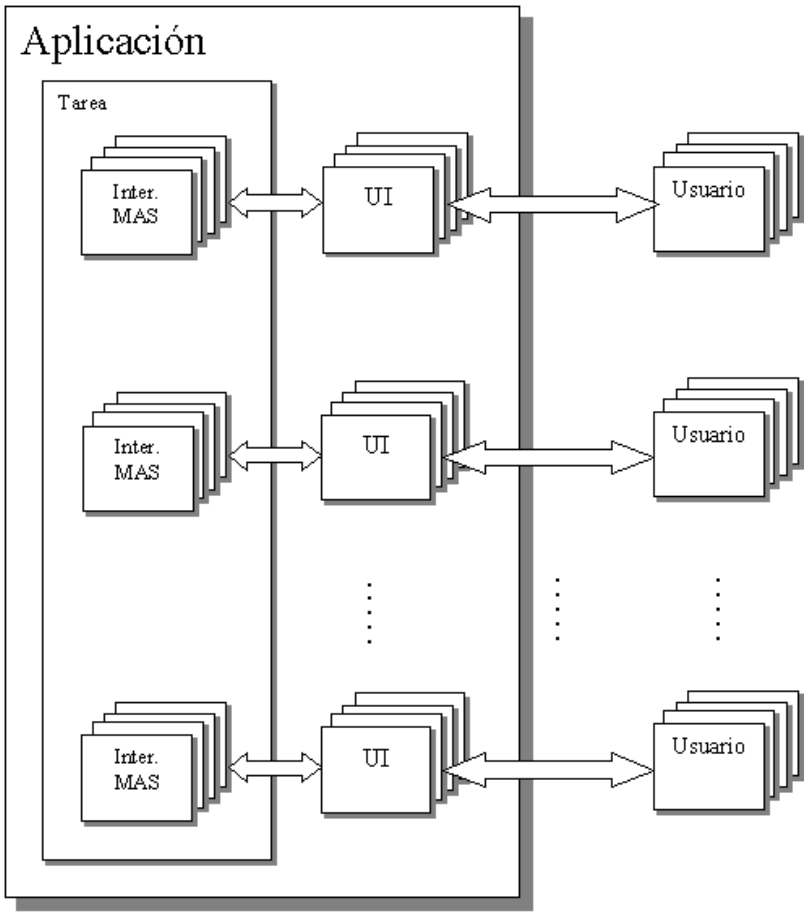

Fig. 2. Una arquitectura general de interacción centrada en el usuario.

Por lo tanto, nuestra propuesta recoge también diseñar y construir sistemas multiagente (MAS) de interacción que ayuden significativamente y positivamente a los usuarios en su relación con los ordenadores. Los humanos son mucho más que procesadores de información. Los humanos son seres afectivos motivados a reaccionar bajo su sistema complejo de emociones, necesidades y condicionamiento exterior [16]. Y el tema clave es que los MAS de interacción de este tipo, capaces de mostrar que el estado emocional de la persona es oído y aceptado, pueden ser construidos usando la tecnología existente.

Al adentrarnos en esta clase de soporte, el sistema debe solicitar activamente información acerca del estado del usuario. El inicio de este diálogo de soporte puede provenir bien de la iniciativa propia del usuario o bien el sistema puede iniciar el diálogo con el usuario de un modo proactivo. Este es posible únicamente si el MAS de interacción es capaz de acceder a medidas de comportamiento del usuario. Aquí aparece, por tanto, la relación entre las métricas de usabilidad descritas y los MAS de interacción.

Volviendo a nuestro razonamiento inicial, diremos que el objetivo último de la HCI debe ser la creación de interfaces de usuario basadas en las preferencias individuales de los usuarios (figura 2). Estas preferencias pueden ser capturadas inicialmente, y hasta cierto punto en los pasos del análisis del desarrollo. Al usar esa información capturada, se pueden crear perfiles de usuario de acuerdo a los estereotipos de usuario identificados. Sin embargo, el usuario avanza y sus preferencias cambian. Así, el entorno debe reflejar los cambios, tanto de las características como de las preferencias del usuario. Insistimos, pues, en que son necesarios MAS inteligentes para capturar esos cambios.

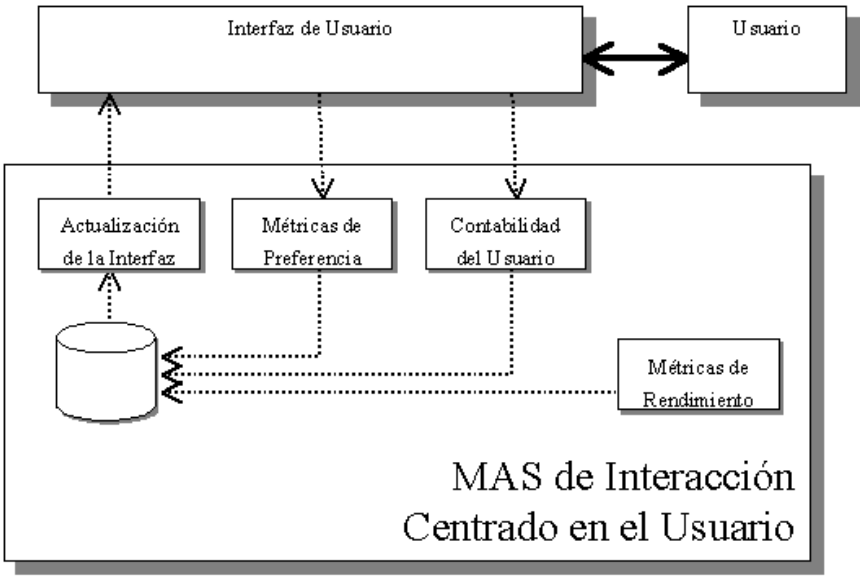

Fig. 3. La arquitectura MAS.

La figura 3 muestra una visión general de la arquitectura multiagente de interacción centrada en el usuario propuesta. En esta figura encontramos tres bloques separados en los que se ha añadido un MAS adaptativo a un modelo tradicional. El agente monitoriza la interacción entre el usuario y la interfaz de usuario capturando diferentes parámetros (o métricas). Por una parte, el MAS graba la historia de la interacción en un proceso de log. Por otra parte, el MAS monitoriza métricas de preferencia, tales como los artefactos preferidos para la interacción (teclado, menú, barra de herramientas, etc.).

En paralelo se van obteniendo las métricas de rendimiento requeridas: el ratio de errores respecto de las acciones de interacción correctas, el tiempo gastado en realizar una tarea determinada, etc. Toda esta información es procesada y almacenada en una base de datos de conocimiento (BDC) que el MAS usa para decidir acerca de las acciones necesarias para conseguir sus objetivos, es decir, para encontrar los posibles cambios a aplicar a la interfaz de usuario según las expectativas del usuario. De este modo, se crea una nueva interfaz de usuario a medida de acuerdo a las características, la experiencia y las preferencias del usuario.

4 Hipermedia Adaptativa basada en Métricas y Sistemas Multiagente

La arquitectura propuesta en las secciones anteriores se está probando en un entorno de hipermedia adaptativa, más concretamente en un sistema de enseñanza on-line en forma de un sistema de tutorización personal inteligente (ITS, *Intelligent Tutoring System*) de una asignatura de ingeniería en la Escuela Politécnica Superior de Albacete.

Uno de los principales objetivos perseguidos es conseguir que los alumnos sean capaces de aprender más, y mejor, es decir, estructurar el aprendizaje de forma que se facilite su aprendizaje. Una de las características que debe plantearse dentro de la enseñanza es el ritmo al que un alumno es capaz de aprender, luego un ITS debe ser capaz de adaptar el ritmo en que se presentan los conceptos al ritmo de aprendizaje de cada alumno (por ejemplo, plantear más o menos ejercicios, plantear más o menos tests, etc. Otro de los aspectos ampliamente considerados en las teorías de aprendizaje es el reforzamiento, premiando el aprendizaje correcto, y penalizando los errores (mensajes, sonidos, etc.).

Otro objetivo en nuestro entorno de hipermedia adaptativo es mejorar la enseñanza, no solamente el aprendizaje. Uno de los principales problemas que el profesor encuentra a la hora de enseñar es que desconoce el nivel de conocimientos de los alumnos. A través de la enseñanza tutorizada se pueden extraer conclusiones que "enseñen a enseñar". Dentro del objetivo de aprender a enseñar se engloba hacerla más comprensible para todos los alumnos.

Por ello, en nuestro sistema de enseñanza (figura 4) hemos incorporado tres MAS: (1) El *MAS Interacción*, que captura las preferencias del usuario por medio de algunas métricas de usabilidad. Los contenidos mostrados al usuario se crean de acuerdo a sus preferencias y comportamientos capturados. (2) El *MAS Aprendizaje* compone los contenidos para el usuario de acuerdo a la información recogida por el *MAS*

Interacción. (3) El *MAS Enseñanza* es una de las más importantes aportaciones en nuestra experiencia, ya que realiza recomendaciones para mejorar las clases de la asignatura.

El usuario (el alumno) interactúa con la interfaz de usuario. A partir de la interacción entre estas dos entidades, modelada por medio del *MAS Interacción*, se recogerán distintas métricas que servirán para ir conformando una *Base de Conocimiento de Perfiles* donde se recojan los distintos perfiles que resultan de la utilización del sistema por parte de distintos alumnos, con distintas aptitudes, motivaciones, etc. El sistemas multiagente para el aprendizaje (*MAS Aprendizaje*), toma los datos obtenidos a partir de los perfiles (análisis de las distintas métricas recogidas) e intenta adecuar los contenidos presentados al alumno concreto que accede al sistema. Por otro lado, será el sistema multiagente de enseñanza (*MAS Enseñanza*) el que recoja medidas que permitan obtener recomendaciones para obtener mejoras en la asignatura.

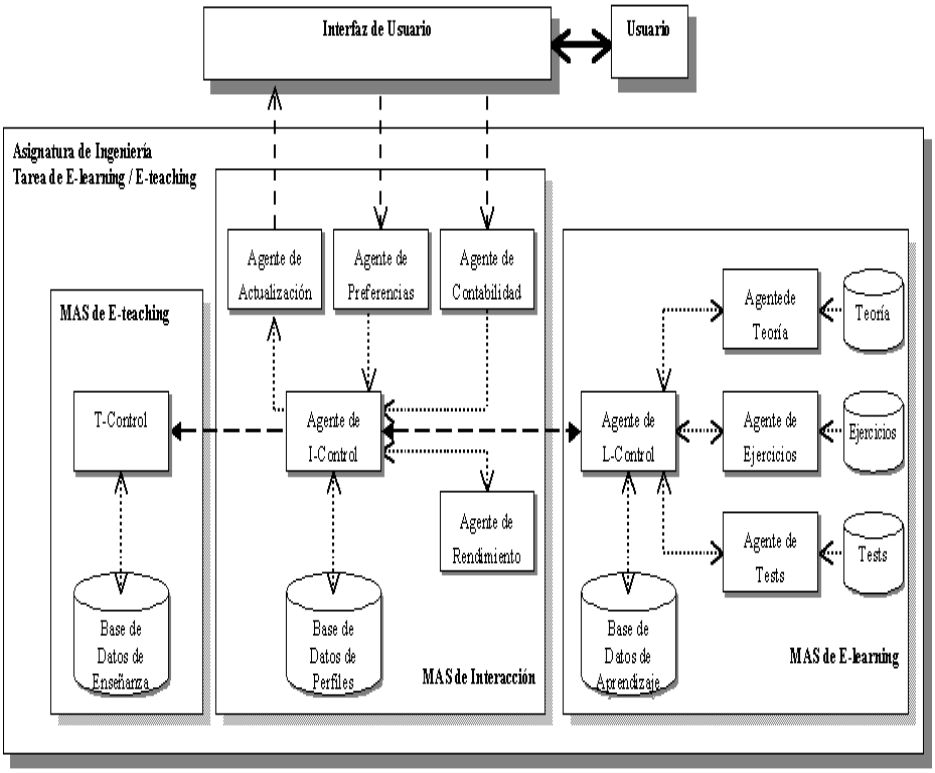

Fig. 4. Una arquitectura adaptativa del sistema hipermedia de aprendizaje/enseñanza.

Con el fin de llevar a buen fin el aprendizaje de la asignatura se propone una descomposición de la misma en teoría, ejercicios y tests (ver figura 5).

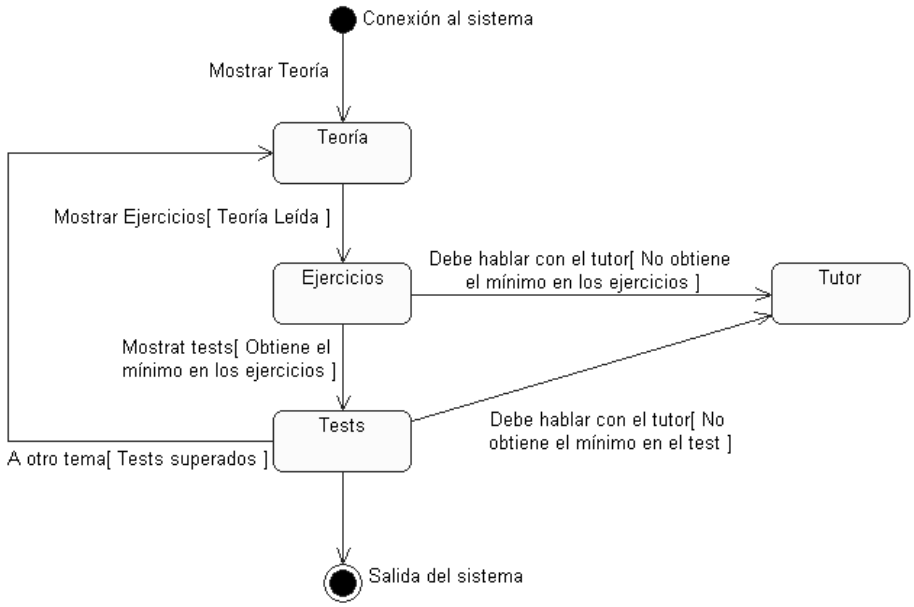

Fig. 5. Descomposición de la asignatura en teoría, ejercicios y tests.

4.1 Sistema Multiagente de Aprendizaje

El *MAS Aprendizaje* aparece a partir del objetivo general de maximizar el aprendizaje de la asignatura. El *agente de control de aprendizaje* tiene una comunicación bidireccional (pide y recibe información) con el *agente de teoría*, el *agente de ejercicios*, el *agente de tests* y con el *agente de control de interacción* (*MAS Interacción*). Este agente pide/recibe páginas de teoría al/del *agente de teoría*, pide/recibe ejercicios al/del *agente de ejercicios*, pide/recibe tests al/del *agente de tests* y comunica (a través del *agente de control de interacción*) al *agente de rendimiento* que vea cual es el rendimiento del alumno para saber si tiene o no que darle un refuerzo. En el caso de que haya que darle un refuerzo al alumno el *agente de control de aprendizaje* elaborará un plan con el material que se deberá mostrar al alumno. Para determinar si se tiene que dar refuerzo o no a un alumno el *agente de rendimiento* tendrá acceso a una BDC donde estarán almacenados los requisitos mínimos de cada tema (cantidad de ejercicios que se muestran inicialmente al alumno, cuantos tiene que hacer bien, en qué tiempo, tiempo máximo en hacer cada ejercicio para considerárselo como bueno).

El *agente de teoría* está a la escucha de que el *agente de control de aprendizaje* le pida una página de teoría. En el momento en que le llega dicho mensaje busca la página de teoría correspondiente y se la envía al *agente de control de aprendizaje*.

El *agente de ejercicios* es autónomo ya que tiene un cierto grado de control sobre las acciones que realiza. El agente por cuenta propia (proactivo) selecciona el conjunto de ejercicios que se propondrán en el tema que está estudiando el alumno y le añade a cada ejercicio los enlaces a las páginas de teoría que explican los conceptos

relacionados con el ejercicio y le envía al *agente de control de aprendizaje* el nombre de los ejercicios que se propondrán. El *agente de teoría* es un proceso que se está ejecutando continuamente para poder conocer cuando tiene que buscar el ejercicio que se mostrará.

El *agente de tests* está a la escucha de que el *agente de control de aprendizaje* le pida un test. El agente por cuenta propia (proactividad) va diseñando para el tema que está estudiando el alumno el conjunto de preguntas tests que se mostrarán al alumno cuando llegue a la fase realizar tests en el tema que estudia. El *agente de tests* es un proceso que se está ejecutando continuamente para poder conocer cuando tiene que buscar el test que se mostrará al alumno.

4.2 Sistema Multiagente de Enseñanza

El *MAS Enseñanza* es fruto del segundo objetivo general fijado, a saber, el de maximizar la enseñanza de la asignatura. El *MAS Enseñanza* irá recogiendo la bonanza o no de los parámetros definidos en el sistema de aprendizaje. EL *MAS Enseñanza* será proactivo en el sentido en que irá dando recomendaciones al profesor acerca de dichos parámetros (ver figura 6).

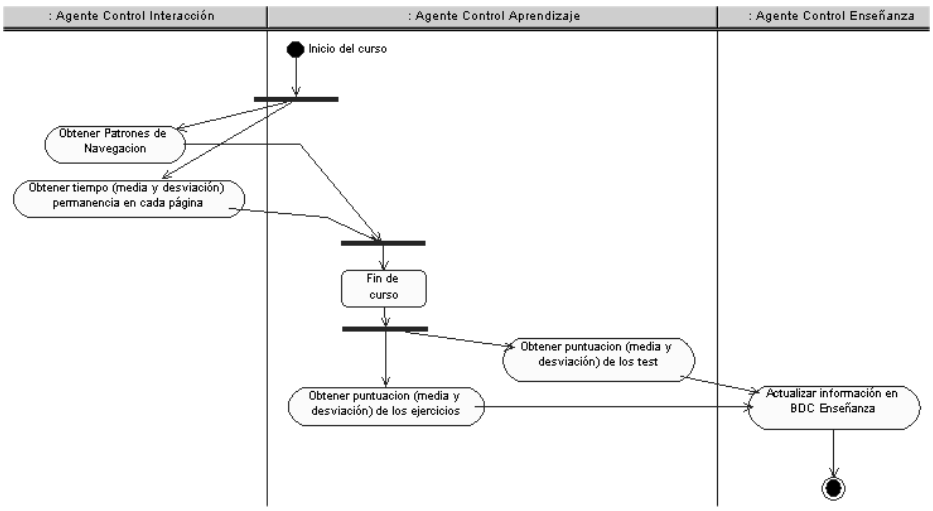

Fig. 6. Diagrama de actualización de la base de conocimiento de enseñanza.

4.3 Sistema Multiagente de Interacción

El *MAS Interacción* está concebido para facilitar la comunicación adaptativa del sistema con el usuario. El *agente de control de interacción* deberá comunicarle al *agente de actualización* cuales son las preferencias del alumno, obtenidas por parte del *agente de preferencias* y la nueva página que debe mostrar (*agente de control de aprendizaje* del *MAS Aprendizaje*). Al hablar de las preferencias del alumno nos

referimos al tipo de letra, color, iconos, etc. que le gustan. La información que recoge irá almacenándose en la *BDC de Perfiles*. Todos los tiempos y ciertos comportamientos del usuario se obtienen a través del *agente de rendimiento* y del *agente de contabilidad*.

El *agente de preferencias* percibe la interacción del alumno con la interfaz de usuario y actúa cuando el alumno cambia sus gustos. El agente de preferencias es un proceso que se está ejecutando continuamente para poder conocer cuáles son las preferencias del alumno en cada momento (ver figura 7).

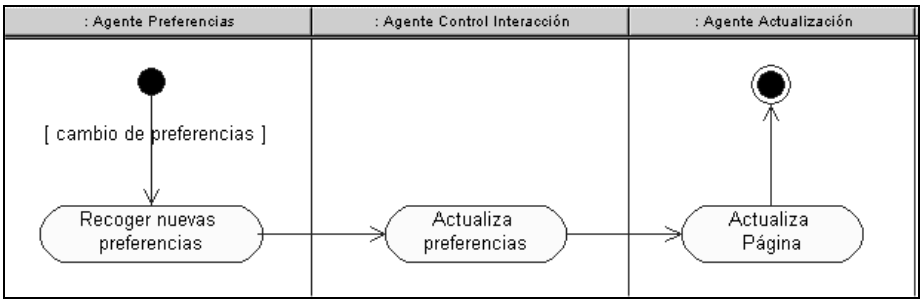

Fig. 7. Actualización de las preferencias de alumno.

El *agente de rendimiento* calcula las métricas de rendimiento cuando el alumno se desconecta (da por finalizada su sesión de trabajo) y va evaluando lo que ha hecho el alumno para saber si hay que darle un refuerzo. Es autónomo y proactivo, pues puede calcular métricas al mismo tiempo que el alumno está realizando otra tarea. Algunas de las métricas que el *agente de rendimiento* maneja son: por cada página de teoría, el tiempo medio que han estado los alumnos en ella; por cada ejercicio calcula puntuación media obtenida, tiempo medio invertido en realizarlo; tiempo medio invertido en realizar un test, y puntuación media obtenida en cada una de las preguntas de test.

El *agente de contabilidad* percibe la interacción del alumno con la interfaz de usuario y actúa (recoge información) cuando el alumno cambia de página, hace scrolling hacia atrás en la página que está leyendo, hace un ejercicio o un test. El agente de contabilidad es un proceso que se está ejecutando continuamente para poder recoger toda la información relevante para el sistema fruto de la interacción del alumno con la interfaz de usuario.

El *agente de actualización* está a la escucha de que el *agente de control de interacción* le comunique que debe actualizar la interfaz de usuario con la información a mostrar al alumno (muestra otra página o la misma página que estaba viendo de acuerdo a los nuevos gustos del alumno). El *agente de actualización* es un proceso que se está ejecutando continuamente para poder conocer cuándo y qué tiene que actualizar en la interfaz de usuario.

5 Conclusiones

La generación de interfaces de usuario se ha convertido en una rama de la Ingeniería del Software de creciente interés. Ello se debe probablemente a la cantidad de dinero y esfuerzo gastados en desarrollar interfaces de usuario que cumplan la exigencia de los usuarios en cuanto a usabilidad y accesibilidad [20]. Por otra parte, los usuarios son cada vez más heterogéneos, y este es un hecho que no podemos ignorar.

En este artículo hemos propuesto una arquitectura que tiene en cuenta la gran diversidad en las características y preferencias de los usuarios: un MAS de interacción centrado en el usuario y adaptativo. La arquitectura toma su inspiración de las métricas de usabilidad y la Inteligencia Artificial, uniendo ambos conceptos de un modo natural.

Agradecimientos

Este trabajo ha sido financiado en parte por los proyectos de investigación CICYT TIC 2000-1673-C06-06 y CICYT TIC 2000-1106-C02-02.

Referencias

[1] Agre, P.: *Computation and Human Experience*.: Cambridge University Press (1997).
[2] Brooks, R.: *Cambrian Intelligence: The Early History of the New AI.*: Massachusetts Institute of Technology (1999).
[3] Card, D.; Glass, R.: *Measuring Software Design Quality.*: Prentice-Hall (1990).
[4] Constantine, L.L.; Lockwood, L.A.D.: *Software for Use: A Practical Guide to the Models and Methods of Usage-Centered Design.*: Addison-Wesley (1999).
[5] Dourish, P.: *Where the Action Is: The Foundations of Embodied Interaction.*: Massachusetts Institute of Technology (2001).
[6] Gilb,T.: *Software Metrics*.: Winthrop Publishers, Inc., Cambridge MA (1977).
[7] Goleman, D.: *Emotional Intelligence*.: Bantam Books (1995).
[8] Henderson-Sellers, B.: *Object-Oriented Metrics: Measures of Complexity*.: Prentice-Hall (1996).
[9] Isen, A.M.; Daubman, K.A.; Nowicki, G.P.: Positive affect facilitates creative problem solving.: *Journal of Personality and Social Psychology,* 52: 6 (1987) 1122-1131.
[10] Kahneman, D.: *Attention and Effort*.: Prentice Hall (1973).
[11] Kitayama, S.; Niedenthal, P.M.: *Heart's Eye: Emotional Influences in Perception and Attention.*: Academic Press (1994).
[12] Lewis, V.E.; Williams, R.N.: Mood-congruent vs. mood-state-dependent learning: Implications for a view of emotion.: En Kruiken, D. (ed.): Mood and Memory: Theory, Research, and Applications (vol. 4) of Special Issue of the *Journal of Social Behavior and Personality*, Vol. 2 (1989) 157-171.
[13] López-Jaquero, V.; Montero, F.; Fernández-Caballero, A.; Lozano, M.D.: Usability metrics in adaptive agent-based tutoring systems.: *Proceedings of HCI International 2003,* (2003).

[14] López-Jaquero, V.; Fernández-Caballero, A.; Montero, F.; Lozano, M.D.: Towards adaptive user interface generation: One step closer to people.: *Proceedings of ICEIS 2003,* Volume 3, pp. 97-103 (2003).
[15] Minsky, M.: *The Society of Mind*.: Simon & Shuster (1988).
[16] Myers, D.G.: *Psychology*.: Worth Publishers (1989).
[17] Nielssen, J.: *Usability Engineering*.: Academic Press (1993).
[18] Porteous, M. ; Kirakowski, J. ; Corbett, M. : *SUMI User Handbook.*: University College Cork, Irlanda (1993).
[19] Wegner, P.: Why interaction is more powerful than algorithms.: *Communications of the ACM*, 40:5 (1997) 80-91.
[20] W3C: http://www.w3.org/WAI/

Aplicaciones Altamente Interactivas para el Aprendizaje de Materias Científico-Técnicas

Roberto Moriyón

Universidad Autónoma de Madrid, Escuela Politécnica Superior
Carretera de Colmenar Viejo, Km 15,
28049 Madrid, España
Roberto.Moriyon@uam.es
http://www.ii.uam.es/~roberto

Resumen. El objetivo de esta exposición es poner de relieve desarrollos tecnológicos recientes en el terreno de la enseñanza asistida por ordenador, con un énfasis especial en la enseñanza de materias científico-técnicas, junto con el contexto en el que han surgido y las principales ideas subyacentes. Más concretamente, se describen los sistemas hipermedia adaptativos para la enseñanza y los sistemas basados en el aprendizaje colaborativo guiado, así como sendas herramientas de autor para la construcción de estos tipos de sistemas.

1 Introducción

El aprendizaje asistido por ordenador es un dominio de aplicación que, aparte de su importancia intrínseca, presenta muchos retos desde el punto de vista tecnológico, siendo un campo de prueba de las tecnologías más avanzadas, tanto desde el punto de vista de la inteligencia de los sistemas como de su capacidad interactiva. Por otra parte, las distintas disciplinas científicas y técnicas tienen características específicas que dan lugar a requerimientos adicionales en las aplicaciones que se utilizan para el aprendizaje de las mismas.

El objetivo de esta exposición es poner de relieve desarrollos tecnológicos recientes en el terreno de la enseñanza asistida por ordenador, con un énfasis especial en la enseñanza de materias científico-técnicas, junto con el contexto en el que han surgido y las principales ideas subyacentes, cubriendo el ámbito del proceso de aprendizaje desde el punto de vista del apoyo a la labor del alumno y del profesor. En este proceso influyen factores de tipo muy diverso, como los didácticos, los de formación y actualización del profesorado, los organizativos e incluso los económicos, aparte de los puramente tecnológicos. Conviene aclarar sin embargo que en este trabajo no trataremos aspectos relevantes para la enseñanza asistida por

ordenador como los relacionados con el apoyo a la gestión del proceso educativo, los mecanismos de valoración mediante el ordenador del trabajo del estudiante, los estándares para la representación de conocimiento y materiales educativos, incluyendo los específicos para la representación de conocimiento científico, etc. En lo que sigue servirán como hilo conductor los requerimientos satisfechos por distintos tipos de software educativo que están siendo propuestos como resultado del trabajo de determinados grupos de investigación, obviando los relativos a las infraestructuras necesarias para su uso; también haré consideraciones sobre los aspectos didácticos relevantes. Por último quiero disculparme de antemano con quienes echen en falta algún tema concreto dentro de la problemática general que acabo de describir. En ningún caso se deberá ello a la intención de marginarlo o minusvalorar su importancia, sino a la necesidad de elegir dentro de un campo muy amplio y a la influencia inevitable del grado de afinidad personal con los distintos temas y desarrollos.

Antes de entrar en cuestiones más específicas parece conveniente hacer algunas reflexiones generales acerca de la utilización del ordenador en el aprendizaje, más próximas a una declaración de principios cuya validez se ha de constatar constantemente que a una exposición rigurosa de hechos. Su objetivo es enmarcar mejor el resto de la exposición. Mi punto de partida en este aspecto es la convicción de que el ordenador será en el futuro una herramienta cada vez más útil como apoyo al aprendizaje en contextos muy diversos. Todas las formas de trabajo habituales se beneficiarán mucho más de la utilización del ordenador de lo que lo hacen ahora, tanto en el aula como en el trabajo individual, tanto en cursos presenciales como en la enseñanza a distancia, tanto en el trabajo de comprensión de conceptos nuevos como en el de memorización de todo tipo de datos, de repaso o de aplicación práctica de los conceptos y habilidades adquiridos. Lo anterior es independiente del hecho hoy día evidente de que este proceso vendrá acompañado de retrocesos debidos a factores muy diversos, incluyendo los económicos, y entre los cuales no hay que despreciar la influencia de determinadas actitudes que pretenden difundir la utilización de las nuevas tecnologías en todos los ámbitos atribuyéndoles posibilidades que están fuera de la realidad. El objetivo principal de esta conferencia es resaltar las expectativas de futuro francamente positivas que he mencionado, mediante una mirada rápida a un par de parajes relevantes pero aislados dentro del campo de la aplicación de las tecnologías de la información y las comunicaciones a la enseñanza de materias científico-técnicas. Pero es fundamental pensar en que los sistemas e ideas que se describen en lo que sigue son simples ejemplos de lo que previsiblemente será una eclosión paulatina pero imparable de aplicaciones cada vez más potentes y útiles para la enseñanza.

También es un objetivo de esta conferencia dar argumentos a favor de la necesidad de que nuestra sociedad se plantee dar un apoyo importante a la labor de I+D en el ámbito de las tecnologías educativas mediante la formación de equipos en los que participen profesores destacados e investigadores en pedagogía y en tecnologías de la

información y las comunicaciones. Solamente de esta forma será posible la aparición de ideas y sistemas que supongan un avance relevante en este campo. Este objetivo, que en otros países como los Estados Unidos e Inglaterra está ya aceptado en sectores importantes, desgraciadamente en otros como el nuestro dista de plasmarse en iniciativas concretas.

El resto de esta presentación se estructura como sigue: en la próxima sección haré una breve exposición de distintos medios (equipos, medios de programación y programas) cuya utilización en la enseñanza de materias científico-técnicas está hoy día más o menos extendida. El apartado siguiente, que es el núcleo de este trabajo, se dedicará a la descripción de los sistemas hipermedia adaptativos y su utilización en la enseñanza, describiendo con un énfasis especial el funcionamiento de herramientas de autor existentes para el desarrollo de cursos de este tipo y algunas experiencias de cursos científico-técnicos, y los sistemas para la enseñanza colaborativa guiada, junto con el funcionamiento de un *framework* que permite su desarrollo sistemático de forma simplificada y una herramienta concreta para el desarrollo de colecciones de problemas de Matemáticas y temas afines basados en esta tecnología.

No quiero dejar de manifestar explícitamente mi agradecimiento a la Universidad de Castilla-La Mancha y a los organizadores del curso de verano que ha dado lugar a este trabajo por la oportunidad que me ha proporcionado su amable invitación de poner juntas ideas dispersas que necesitaban de un estímulo para adquirir una forma más cohesionada y sistemática, y de exponerlas en un entorno académico y multi-disciplinar que en este caso creo muy apropiado para el tema que abordan. Aprovecho también la ocasión para agradecer a distintas personas su ayuda directa o indirecta, incluso en algunos casos inconsciente. Algunas partes de este trabajo serían menos coherentes o menos completas sin la ayuda y la información que me han prestado Pablo Castells, Rosa Carro, José Antonio Macías, Estrella Pulido y Pilar Rodríguez.

2 El uso actual de los ordenadores en la enseñanza

La utilización más habitual hoy día de ordenadores en el proceso educativo, dejando de lado aquéllas materias en que el funcionamiento del ordenador o de equipos o programas asociados a él son parte de la materia a tratar, es mediante los programas e infraestructuras básicas de uso general, como el vídeoproyector, la videoconferencia, los programas para generar presentaciones de uso general, hojas de cálculo y bases de datos. Estos medios se utilizan preferentemente en el aula y ofrecen distintos grados de interactividad. Por supuesto, estos medios se pueden utilizar para la enseñanza de la mayor parte de las disciplinas, incluyendo las científico-técnicas.

También es cada vez más frecuente la utilización de programas para áreas y tareas específicas, como los sistemas de información geográfica, los de diseño asistido por ordenador, los programas de emulación, los especializados para Matemática simbólica

como *Matemática* y *Mapple*, o los de Geometría dinámica. En la mayor parte de las disciplinas científicas hay programas específicos para su enseñanza, especialmente de simulación, cuya utilización está cada vez más extendida. Algunos de ellos tienen aplicación en áreas muy diversas, como los sistemas de información geográfica, utilizados tanto en la enseñanza de la Geografía como en la de Geología y otras especialidades. En muchos casos como el anterior, el de los programas de diseño asistido por ordenador o los programas para la realización de análisis estadístico, estos programas son útiles también en el ámbito profesional, por lo que son objeto de estudio por sí mismos como parte de la labor de capacitación incluida en muchos planes de estudios. Además, en muchos otros casos estos programas aportan la posibilidad de representar de forma virtual entornos de trabajo e información que sin el ordenador serían imposibles de utilizar con facilidad. En otros se aprovecha la inmensa capacidad de los ordenadores para hacer cálculos, permitiendo que los estudiantes realicen tareas de cómputo que de otra forma serían imposibles. La capacidad interactiva suele ser otro de los aspectos más destacados de estos programas. Por ejemplo, los programas de Geometría dinámica permiten hacer construcciones geométricas de gran complejidad, modificando posteriormente aspectos parciales de las mismas, lo que da lugar a la aplicación del mismo proceso constructivo a partir de objetos geométricos diferentes.

Los programas más interactivos actuales para la enseñanza de materias específicas, como los programas de Geometría dinámica, propician el aprendizaje activo y una mayor comprensión de los conceptos relacionados con el escenario en el que se trabaja. Además, pueden estimular la creatividad y la capacidad de iniciativa del alumno al permitirle explorar distintos caminos de actuación de forma libre dentro de su ámbito de trabajo. Para evitar la posibilidad de problemas como la pérdida de perspectiva de los objetivos a alcanzar y de la forma de alcanzarlos puede ser importante su utilización con una planificación adecuada.

Desde el punto de vista de los contenidos, una gran parte de la información digitalizada disponible para la enseñanza actualmente consiste en documentos hipermedia estáticos. Con ello se consiguen grandes ventajas con respecto a su almacenamiento en libros. Así, la utilización de formatos web permite un acceso más universal a la información; igualmente, se posibilita la realización de búsquedas incomparablemente más complejas y la definición y utilización de relaciones conceptuales con mayor facilidad y eficiencia. Algunos de estos documentos contienen animaciones con distinto grado de dinamicidad que permiten ilustrar mejor la materia que enseñan, acercándolos a la capacidad de los sistemas interactivos descritos anteriormente. A cambio, el diseño de estas componentes interactivas es más costoso que el de materiales estáticos, aunque las herramientas y recursos para la programación disponibles para su desarrollo simplifican cada vez más el mismo. Podemos citar dentro de este apartado herramientas como Macromedia Director y Flash, y bibliotecas como VRML, que permiten hacer simulaciones de mecanismos de distinto grado de complejidad de forma comparativamente simple. De nuevo este tipo

de medios tiene aplicaciones mucho más allá del contexto educativo. Hay que reconocer no obstante que la cantidad de materiales disponibles hoy día relativos a materias científicas o técnicas es muy reducida si la comparamos con los que se pueden encontrar en otras ramas del saber. Ello se debe a la mayor complejidad de este tipo de contenidos, que requieren en muchos casos la representación y manipulación de fórmulas, gráficas y simulaciones, a veces incluso de forma simultánea y coordinada.

Por último, conviene resaltar la importancia que en todos los casos anteriores tienen las herramientas que permiten que los profesores puedan desarrollar materiales educativos con facilidad y sin necesidad del dominio de conocimientos informáticos tan complejos y alejados de su área de especialización como requieren los lenguajes de programación genéricos. Este será un leitmotiv que aparecerá de forma recurrente a lo largo de la conferencia.

3 Algunos tipos de sistemas emergentes

En esta sección veremos algunos tipos de sistemas para el aprendizaje asistido por ordenador que han aparecido en los últimos años como resultado del trabajo de determinados grupos de investigación. Nos centraremos en sistemas cuyas características permiten su utilización en la enseñanza de materias científico-técnicas, y describiremos algunas experiencias realizadas en este contexto y los aspectos más relevantes de los sistemas para ello. Más concretamente, estudiaremos en primer lugar la aplicación de sistemas hipermedia adaptativos para la educación, y después veremos los aspectos más sobresalientes de sistemas colaborativos guiados.

3.1 Sistemas hipermedia adaptativos para la educación

Los sistemas hipermedia adaptativos son sistemas informáticos basados en hipertexto e hipermedia que reflejan características del usuario, a quien adaptan diversos aspectos visibles de su funcionamiento. Los sistemas hipermedia adaptativos se utilizan en muchos ámbitos, incluyendo la educación, los sistemas de información y ayuda en línea, de recuperación de datos, de información institucional y para la gestión de vistas personalizadas. En algunos campos como el de los sistemas de información institucional esta tecnología lleva tiempo en el mercado, destacando en este ámbito los portales adaptativos y personalizables.

Brusilowsky, [3], clasifica de forma muy exhaustiva los sistemas hipermedia adaptativos de acuerdo con distintos criterios que los definen. Destacan entre ellos los siguientes:

- Las características del usuario contempladas pueden ser muy diversas y estar relacionadas con el contexto de trabajo y aspectos personales, en particular con sus

objetivos y preferencias, con los conocimientos que tiene y con su procedencia en el trabajo previo y su experiencia en la navegación.

- Los contenidos se pueden adaptar de distintas formas y mediante técnicas diferentes. Se pueden utilizar explicaciones adicionales que se muestran o no en función de las condiciones de adaptabilidad que considera el sistema, explicaciones que se muestran de forma variable dependiendo de dichas condiciones, o materiales cuyo orden de presentación depende de las características del usuario. En cuanto a las técnicas utilizadas, merece la pena citar la utilización de texto y otras componentes condicionales, así como de texto expansible y de bases de datos de variantes de contenidos.
- Es muy habitual la adaptación de las posibilidades de navegación. Esto se hace tanto en las guías globales del curso como en las locales, así como en las ayudas para la orientación global y local. En cuanto a las técnicas utilizadas, destacan la utilización de reglas para decidir qué conceptos deben de estar visibles en cada momento, la determinación del acceso a la información del curso en función de las tareas que está realizando el usuario y de los nodos del mismo que les correspondan, y las basadas en la modificación adaptativa de la relevancia asignada a la información accesible, con una selección y ordenación posterior de las opciones disponibles de acuerdo con la relevancia determinada.

A continuación describiremos algunos sistemas educativos adaptativos, destacando sus principales características.

ELM-ART II (Episodic Learner Adaptive Remote Tutor, [23]) es un entorno inteligente de enseñanza de LISP que permite la programación basada en ejemplos y el análisis inteligente de las soluciones aportadas por los estudiantes y ofrece facilidades para comprobar y depurar programas. En ELM-ART II, ganador del European Academic Software Award en 1998, el conocimiento se representa mediante una red de conceptos donde las unidades se organizan jerárquicamente en lecciones, secciones, subsecciones y páginas. Cada unidad contiene información sobre el texto a ser presentado y sobre las relaciones de esa unidad con otros conceptos (prerrequisitos, conceptos relacionados o conceptos que se suponen conocidos después de trabajar con la unidad considerada). Los textos presentados incluyen ejercicios que el estudiante deberá resolver para que se considere la unidad como aprendida. Las interacciones del estudiante se almacenan en un modelo del mismo. Al presentar los contenidos en cada enlace se indica gráficamente el grado de preparación del estudiante para abordar los contenidos correspondientes así como la recomendación de que el trabajo proceda accediendo al mismo.

Los resultados obtenidos tras la utilización de ELM-ART II por grupos de estudiantes sin conocimientos previos de LISP indican que la combinación de las técnicas de anotación de enlaces adaptativa y la indicación del enlace óptimo a seguir en cada instante es especialmente útil durante la primera fase del aprendizaje.

El sistema AHA (Adaptive Hypermedia Architecture, [2]) permite la generación de cursos adaptativos para la web mediante la instrucción de instrucciones condicionales en los documentos. Estas instrucciones actúan como filtros a la hora de decidir las porciones de cada documento que se muestran al estudiante. Con ello se pueden anotar, deshabilitar, ocultar y eliminar enlaces adaptativos. AHA mantiene un modelo simple del estudiante basado en el conocimiento o desconocimiento de cada concepto. AHA se ha utilizado en cursos de Interfaces de Usuario Gráficas y Bases de Datos. El software se encuentra disponible en la web.

DCG (Dynamic Courseware Generator, [22]) permite la generación automática de cursos que se adaptan a los objetivos de los estudiantes, a su conocimiento previo del dominio y al proceso de aprendizaje. En este sistema se separa la estructura conceptual del dominio de los contenidos del curso, estructurándose los conceptos como un mapa de carreteras que se utiliza para generar el plan del curso. Para ello un planificador busca aquellos subgrafos que conectan los conceptos ya conocidos por el estudiante con el concepto objetivo del curso y le ofrece al estudiante un plan. Si durante el estudio de los conceptos contenidos en dicho plan el estudiante no alcanza el nivel de aprendizaje necesario para continuar, el planificador genera un nuevo plan. El proceso de diseño de un curso en DCG consiste en crear la estructura de conceptos y añadir enlaces que relacionen cada concepto con los documentos disponibles para su enseñanza. DCG ha sido utilizado en distintos dominios, como la estructura y funcionamiento de dispositivos electrónicos y la diagnosis médica.

Por último Tangow (Task-based Adaptive learner Guidance on the Web, [5], [6]) utiliza tareas y reglas docentes para describir y manejar de manera sencilla los conceptos asociados a un curso y las distintas formas de estructurarlos y organizarlos dentro del mismo, así como las posibilidades de asociación de fragmentos de contenidos a los conceptos. El mecanismo que utiliza permite una gestión independiente de la estructura del curso con respecto a los contenidos que forman parte de él, lo que permite que el programa seleccione en cada momento los temas o conceptos más apropiados para cada estudiante y elija por separado de manera dinámica los contenidos que componen los documentos que se le presentan. La Fig. 1 muestra la arquitectura del sistema Tangow.

Las tareas docentes de Tangow, que especifican la estructura de los cursos, pueden ser atómicas o compuestas, formando una jerarquía para cada curso. Las tareas de más alto nivel, unidades, corresponden a capítulos o secciones del curso. Las tareas pueden ser de distintos tipos (teoría, ejemplos y prácticas), y su descomposición (con distintos tipos posibles de secuenciación) en subtareas es adaptativa, dependiendo del contexto. El sistema determina qué descomposición se activa en cada momento entre las que están disponibles para la tarea didáctica que el sistema está llevando a cabo. Lo hace en base a condiciones asociadas a las distintas descomposiciones posibles que especifican requisitos que el estudiante debe reunir para que se le planteen las subtareas correspondientes y dependencias entre tareas. El carácter adaptativo de los cursos de Tangow se basa en este mecanismo. La selección de los fragmentos de

contenido a mostrar para cada tarea atómica es otro aspecto que proporciona mayor adaptatividad al sistema.

Tangow guarda a lo largo del trabajo de un estudiante en un curso, que puede abarcar múltiples sesiones, información dinámica acerca del trabajo realizado por el estudiante, como las tareas en que ha participado, sus resultados en los ejercicios que ha resuelto y los temas en los que ha trabajado. La información acerca de las tareas realizadas forma un árbol que se actualiza constantemente sirviendo como fuente para la extracción del resto de la información que se almacena. El motor de gestión de tareas utiliza el árbol dinámico de tareas y el resto de la información extraída que describe el estado instantáneo del proceso de aprendizaje para contrastarlo con el modelo del curso formado por la estructuración de sus tareas y decidir en cada momento la forma en que debe de continuar la interacción del sistema con el estudiante.

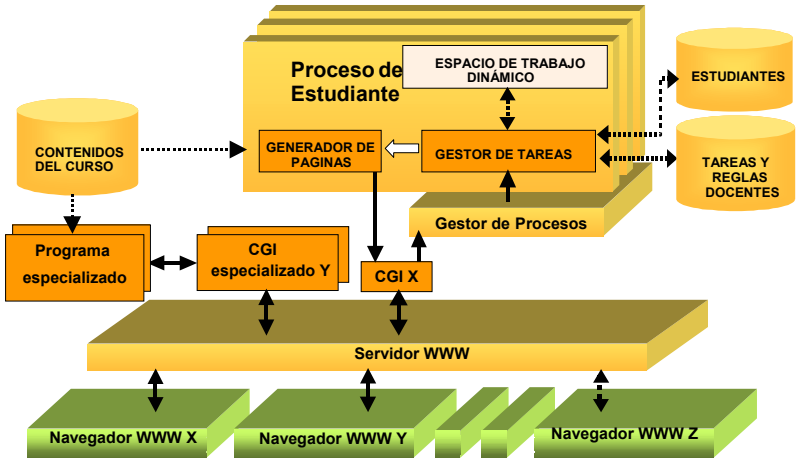

Fig. 1. Arquitectura de Tangow

Tangow se ha utilizado en los últimos años en una amplia variedad de cursos, incluyendo algunos de carácter científico-técnico, como un curso de compiladores como apoyo a la docencia presencial en la titulación de Ingeniería Informática de la Universidad Autónoma de Madrid y otro de Introducción a la Informática, impartido en las universidades Autónoma, Carlos III, Politécnica y Rey Juan Carlos dentro del proyecto ADA-Madrid financiado por la Comunidad Autónoma de Madrid.

Como hemos visto, algunos de los sistemas adaptativos para la enseñanza que hemos descrito son genéricos por cuanto permiten definir sobre ellos cursos arbitrarios con las características de adaptatividad correspondientes. En estos casos es de la mayor importancia conseguir que el diseño de los cursos sea lo más sencillo posible, y en particular que requieran los conocimientos más reducidos fuera del dominio propio del curso que se desea desarrollar. Al igual que en otros apartados de esta exposición, esto nos lleva considerar la disponibilidad de herramientas para el diseño de cursos, una cuestión recurrente de la mayor importancia.

En general, la complejidad inherente a la construcción de herramientas interactivas para el diseño de software crece enormemente en función de la complejidad intrínseca de las interfaces que se desean generar. Las técnicas más avanzadas para la especificación del comportamiento software interactivo sin necesidad de programar son las basadas en programación por demostración, [7], [8], que resultan particularmente útiles en el desarrollo de herramientas de autor para software educativo altamente interactivo, como es el caso de los sistemas educativos adaptativos.

Existen muy pocas herramientas de este tipo que reduzcan la cantidad de programación necesaria suficientemente como para poder considerarse auténticas herramientas de autor. La primera digna de mención es Interbook, [4], una herramienta que permite introducir información adicional a una colección de documentos para transformarlos en un curso adaptativo mediante la ocultación de documentos correspondientes a estados que permiten el acceso de materiales que el estudiante no está en condiciones de comprender.

Con Interbook el profesor o diseñador del curso deberá estructurar el contenido de los documentos en capítulos y secciones, a los que puede asociar listas de prerrequisitos y de conceptos asociados que el documento permite conocer. Esta información es la esencial para la creación del curso.

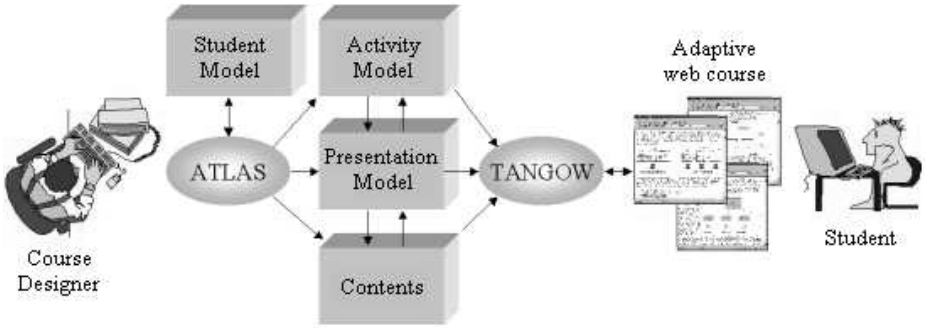

Fig. 2. Arquitectura de Atlas

Atlas, [12], es otra herramienta de autor para el diseño de cursos adaptativos basados en Tangow. Dadas las características del sistema Tangow descritas más arriba, se trata de una herramienta mucho más ambiciosa y potente que Interbook. Atlas permite el diseño integrado, en un mismo entorno, de los distintos aspectos de los cursos y las relaciones entre ellos: estructura, contenido, presentación, y perfiles de estudiante. Para reducir la necesidad de abstracción por parte del autor, la herramienta ofrece la opción de trabajar en tiempo de diseño sobre una representación de los cursos próxima al resultado final, a partir de ejemplos concretos del perfil de usuario, proporcionados por el diseñador. La Fig. 2 muestra la arquitectura de la herramienta Atlas.

El espacio de trabajo de la interfaz de Atlas está formado por tres áreas principales que representan respectivamente el modelo de tareas, el modelo de contenidos y el modelo del estudiante. En la primera el diseñador define la estructura del curso mediante un modelo de las tareas del estudiante, que siguiendo el modelo Tangow y de acuerdo con lo descrito anteriormente, están organizadas jerárquicamente. En el área de contenidos se incluyen las componentes disponibles, fragmentos de HTML, que recogen los contenidos de las distintas partes del curso. En el área del modelo del estudiante se detallan las características de éste que se consideran relevantes para el curso en construcción, y que han de condicionar su estructura. Este modelo consiste en un conjunto de atributos simples y compuestos estructurados en forma de un árbol que el diseñador puede editar.

El diseñador asigna a cada tarea las unidades de contenido que le corresponden, arrastrando con el ratón sobre la tarea objetos del área de contenidos del curso. El árbol de tareas se construye pulsando y arrastrando el ratón entre tareas, asociando cada tarea con sus subtareas. En la Fig. 3, el diseñador ha descompuesto una unidad (tarea de alto nivel) sobre Programación en Java en cuatro subtareas: Tipos Abstractos de Datos, Conceptos de POO, Aspectos Básicos del Lenguaje, y Clases y Objetos en Java. Entre cada tarea y sus subtareas se interpone un nodo para cada ramificación, que representa el elemento adaptativo que permite condicionar la estructura del curso a las características del estudiante y a la actividad realizada por éste en tiempo de ejecución. Su papel consiste en controlar mediante una condición asociada cuál de las descomposiciones disponibles se aplicará en tiempo de ejecución cuando, como en el ejemplo, se ha definido más de una alternativa. Las condiciones se editan seleccionando y arrastrando atributos del modelo de usuario sobre un diálogo de edición de condiciones. Atlas permite combinar predicados simples de comparación entre propiedades del estudiante, parámetros de las tareas y valores literales, en forma normal disyuntiva.

Para simplificar el tratamiento del aspecto adaptativo, ATLAS permite trabajar sobre perfiles concretos del usuario proporcionados por el diseñador, de forma que la herramienta abstrae y generaliza después el modelo de tareas construido para los casos particulares.

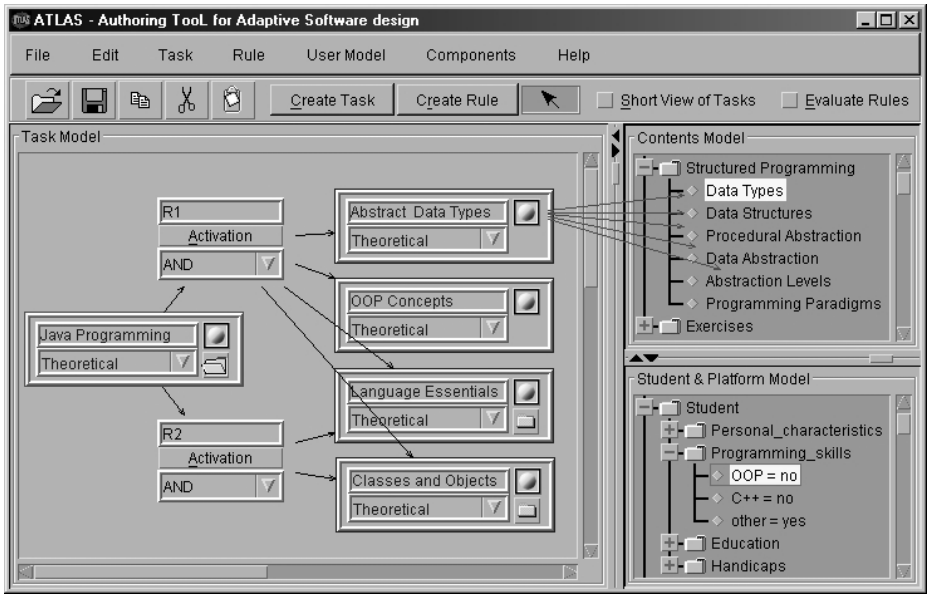

Fig. 3. Interfaz de Atlas

Estos mecanismos permiten al diseñador definir de forma totalmente interactiva, salvo por la especificación de las condiciones, que por su complejidad ha de hacerse utilizando fórmulas que involucran la notación básica de acceso a propiedades de objetos (que se puede especificar interactivamente) junto con las operaciones aritméticas básicas e igualdades y desigualdades.

3.2 Sistemas para la enseñanza colaborativa guiada

En los últimos años se ha producido un considerable avance en la creación de aplicaciones, herramientas y recursos para la programación que permiten la colaboración entre los usuarios. Hoy día el trabajo colaborativo asistido por ordenador (Computer Supported Collaborative Work, CSCW) es un área de investigación consolidada, con congresos de prestigio mundial que se reúnen periódicamente. Además se han desarrollado productos a partir de proyectos de investigación en CSCW que están teniendo un gran éxito, como BSCW, [1].El CSCW tiene como principal objetivo el estudio de la utilización de redes de ordenadores que permiten el trabajo en grupo a través de una interfaz compartida. Esta área de investigación, que tiene muy importantes puntos de conexión con la de las interfaces de usuario, tiene un carácter claramente multidisciplinar; en ella colaboran tecnólogos, psicólogos y sociólogos, cubriendo los distintos aspectos conceptuales que la configuran.

El aprendizaje colaborativo asistido por ordenador (Computer Supported Collaborative Learning, CSCL) es una subárea del CSCW cuyo objetivo es la mejora del proceso de aprendizaje mediante la colaboración de los estudiantes con otros agentes no estáticos. Estos agentes pueden ser de distinta naturaleza, como personas o agentes virtuales, y jugar distintos roles, como profesores o compañeros. También en esta área más restringida hay congresos de relieve especializados que se celebran periódicamente, así como publicaciones internacionales especializadas. Sin embargo, el grado de difusión de los productos resultantes de la investigación llevada a cabo en esta área es menor que en el caso del CSCW.

Jermann, [11], hace una clasificación exhaustiva de los sistemas para el aprendizaje colaborativo basada en el tipo de sistemas (sistemas replicativos, que muestran a los colaboradores las acciones de los demás y sistemas de tutoría, que asesoran al usuario en base a estos indicadores), el tipo de datos de interacción que contemplan, los procesos que utilizan para derivar representaciones de más alto nivel de los datos y el tipo de *feedback* que proporcionan a los usuarios.

El aprendizaje colaborativo admite distintos grados de libertad por parte de los estudiantes que participan en él. El aprendizaje con un tutor es habitualmente un caso extremo de colaboración dirigida, mientras que el aprendizaje en grupo entre estudiantes lo suele ser de colaboración independiente.

Pese a que la colaboración en sí es un concepto que por su propia naturaleza plantea retos tecnológicos de envergadura, como los derivados de los requisitos de sincronización entre los actores del trabajo colaborativo, mientras que el aprendizaje colaborativo es en primera instancia un caso particular del trabajo en grupo en el que parecen predominar las problemáticas interpersonales sobre las exigencias tecnológicas específicas, lo cierto es que cuando se analizan los requisitos que impone el aprendizaje colaborativo en un contexto más específico aparecen necesidades nuevas en el ámbito tecnológico que son dignas de investigación. Más adelante veremos algún ejemplo en que se dan estas circunstancias.

La labor investigadora que se hace en el contexto del CSCL es muy variada. Abarca el desarrollo de teorías sobre el aprendizaje colaborativo, que posteriormente se utilizan como referencias para establecer objetivos y criterios de evaluación en otros trabajos, así como la realización de experiencias de trabajo en grupo, el análisis del proceso de colaboración que tiene lugar en las experiencias anteriores para establecer los resultados obtenidos mediante el aprendizaje colaborativo y las circunstancias en las que éste es más efectivo, y el desarrollo de tecnologías que potencian el aprendizaje colaborativo y simplifiquen el diseño y el desarrollo de los materiales interactivos utilizados.

Hay evidencias empíricas de que el trabajo en grupo proporciona beneficios en el proceso de aprendizaje, sobre todo en temas de Ciencias. Los aspectos en los que estos beneficios son más destacados son la consecución de una mayor motivación y disposición a abordar problemas complejos y la promoción de una participación activa, lo que conlleva la incorporación del conocimiento a un nivel más profundo. El

papel del profesor en este contexto deja de ser el de una simple fuente de conocimiento, convirtiéndose en un experto en aprendizaje, que orienta a los alumnos en su trabajo.

Entre las teorías relacionadas con el aprendizaje colaborativo destaca el Constructivismo, que postula que la adquisición o construcción del conocimiento se realiza mediante la participación activa en procesos relacionados con el mismo. El constructivismo, en su interpretación más pura, propone que el estudiante tenga libertad para explorar alternativas mientras experimenta y para elegir el camino a seguir y asuma la responsabilidad de su propio aprendizaje y que el profesor se responsabilice de darle ayuda. Presupone que con ello el estudiante desarrolla habilidades metacognitivas que le permiten monitorizar el rendimiento de su propio proceso de aprendizaje y dirigirlo. También proclama el desarrollo por parte del estudiante de una capacidad especial de adaptación de los métodos aprendidos a situaciones nuevas.

El constructivismo en el contexto clásico está muy ligado al trabajo experimental y de campo. En este aspecto, las Nuevas Tecnologías permiten extender la praxis constructivista a otros ámbitos mediante la simulación, el hipertexto, etc. Además, el constructivismo tiene pleno sentido en el ámbito colaborativo, en el que los estudiantes no se limitan a trabajar de forma independiente para la consecución de un objetivo, sino que además intercambian entre sí experiencias, resultados y conclusiones e incluso planifican en común el trabajo a realizar.

El constructivismo está muy relacionado también con la teoría del aprendizaje basado en problemas, próxima a su vez a las ideas anteriores, que propone que el proceso de aprendizaje se base en la resolución de problemas representativos de casos reales. Esta teoría se basa en la hipótesis de que con este enfoque el trabajo del estudiante resulta más atractivo, que consigue una mayor profundización en la abstracción de las ideas descubiertas en casos concretos y que se desarrolla de una forma especial la capacidad de reconocer ideas válidas para la resolución de problemas, así como que se generan actitudes de autoaprendizaje. El aprendizaje basado en problemas se planteó por primera vez de forma sistemática como una teoría didáctica en el contexto del aprendizaje de las ciencias biomédicas básicas. Posteriormente se ha estudiado y aplicado en otros contextos, especialmente en el de las Matemáticas, [20]. Evidentemente, el aprendizaje basado en problemas tiene pleno sentido en el ámbito colaborativo.

La teoría del aprendizaje guiado pone énfasis en la importancia de disponer de medios de revisión del trabajo realizado por parte del alumno. Esta revisión puede llevarse a cabo por el propio alumno, por un grupo de alumnos o por un profesor responsable de la misma junto con el alumno o grupo de alumnos. Debe de permitir el análisis de propuestas para mejorar los resultados obtenidos y superar las dificultades y errores cometidos, estudiando las alternativas existentes.

Como veremos en lo que sigue, la utilización del ordenador y de técnicas de trabajo colaborativo da lugar a un incremento de los beneficios del aprendizaje guiado al poner en manos del profesor herramientas más potentes que le permiten seleccionar mejor la forma en que distribuye su trabajo y mantener un intercambio de información más eficiente desde el punto de vista formativo con los alumnos.

El concepto fundamental que permite que el ordenador asista en la enseñanza colaborativa guiada es el de historia de trabajo y revisión, [10]. Una historia de trabajo es una representación de la sucesión de acciones realizadas durante una o varias sesiones de trabajo, bien sea aislado o colaborativo, en una forma que permite su revisión y análisis posterior; por lo tanto viene a ser una grabación de las acciones del estudiante que se puede reproducir posteriormente. Las historias de trabajo y revisión permiten que se les añada nuevas historias elaboradas a partir de instantes determinados de las mismas, en las que se muestran otros caminos que se podrían haber seguido. Un ejemplo muy simple de una historia con revisiones corresponde a un análisis de una partida de ajedrez, en el que se estudia la forma en que la partida podría haber seguido a partir de determinadas posiciones si un jugador hubiera realizado una jugada diferente de la que hizo en la partida que se analiza.

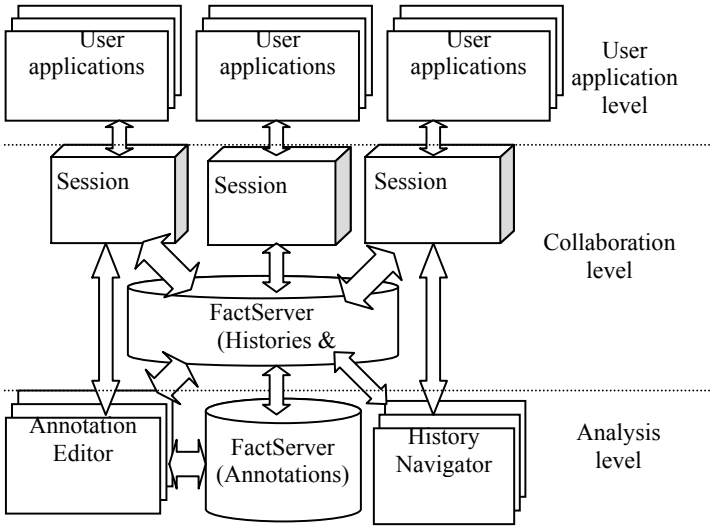

Fig. 4. Arquitectura del framework *FACT*

Uno de los primeros precedentes de la utilización de historias para el análisis de la interacción en procesos se encuentra en el trabajo de B. Shneiderman y sus colaboradores, [18], que desarrollaron un sistema que permite el aprendizaje

colaborativo de los principios básicos de la tecnología de las bombas de vacío a través de su simulación y le incorporaron un sistema de historias de colaboración que permitía su revisión posterior. Por otra parte, *FACT*, [14], es un *framework* para el desarrollo de aplicaciones para el aprendizaje colaborativo guiado que utiliza historias de aprendizaje y revisión que pueden incorporar anotaciones que contienen referencias a otras historias. La Fig. 4 muestra la arquitectura de *FACT*.

Las aplicaciones desarrolladas con *FACT* permiten a varias personas conectadas a una red de ordenadores, incluyendo un profesor, participar simultáneamente o asíncronamente en sesiones de aprendizaje. Los estudiantes pueden trabajar aisladamente o formando grupos que comparten comentarios o análisis de diferentes alternativas que se presentan en su trabajo conjunto.

Cuando trabajan en grupo en una aplicación desarrollada con *FACT*, los estudiantes pueden hacer propuestas para las siguientes acciones a tomar simplemente poniéndolas en efecto y mostrándoles a continuación a sus compañeros la historia correspondiente. La evolución del trabajo final, incluyendo las propuestas de los distintos estudiantes y las decisiones correspondientes, quedan a disposición del profesor y del propio grupo para su revisión. De esta forma, los estudiantes tienen todas las ventajas de las técnicas clásicas de trabajo en grupo, y el profesor tiene la ventaja de poder revisar el trabajo, contemplar las acciones de los estudiantes, corregir los errores más importantes que encuentra en ellas y hacer sus propias propuestas para su revisión posterior. Cuando un estudiante trabaja individualmente el profesor puede proporcionarle una ayuda semejante.

Independientemente de la forma de trabajo, *FACT* permite que los estudiantes en su conjunto reciban una idea más directa y más profunda de lo que han hecho equivocadamente o simplemente de lo que pueden hacer mejor. Esto se consigue especialmente mediante la posibilidad de trabajar con ejemplos de situaciones más simples en las que aparecen problemas similares a los que dificultan su trabajo y practicar con ellos, bien sea por sí mismos o síncronamente con el profesor.

Los profesores pueden enseñar a sus alumnos cómo resolver problemas haciendo que sigan síncronamente su resolución. También pueden guiarles directamente mientras trabajan mostrándoles síncronamente alternativas a las decisiones que hacen o, indirectamente, creando anotaciones que pueden incluir enlaces a ejemplos guardados que ilustran situaciones similares o creando alternativas para que los estudiantes las analicen posteriormente. Además, pueden revisar la labor de análisis de otros estudiantes, leyendo sus anotaciones. Para facilitar el trabajo del tutor, las aplicaciones basadas en *FACT* pueden utilizar agentes que le informan cuando un estudiante o grupo de estudiantes necesitan ayuda.

Hasta el momento se han desarrollado dos aplicaciones basadas en *FACT*, *ChessEdu*, [13], y *ConsMath*, [15], [16], aparte de otros ejemplos sencillos para la validación inicial del *framework*, como una aplicación para el aprendizaje de la resolución de puzzles. *ChessEdu* permite la práctica y el aprendizaje del ajedrez por uno o más estudiantes, mientras un profesor sigue varios juegos e interactúa con ellos.

Para utilizar *ChessEdu*, como cualquier otra aplicación desarrollada con *FACT*, es necesario que corra en una máquina un servidor de sesiones de *FACT*. Cada sesión representa un grupo de participantes síncronos que trabajan sobre una partida determinada. Varias sesiones pueden trabajar simultáneamente sobre el mismo juego, que puede estar jugándose a la vez o no.

FACT es especialmente adecuado para el aprendizaje de tareas y materias en las que los conocimientos adquiridos se utilizan para resolver problemas y en la adquisición de destrezas específicas, situación muy común en las Ciencias clásicas (Matemáticas, Física, Química), la utilización simulada de equipamientos, el aprendizaje del manejo de herramientas informáticas interactivas y el aprendizaje de juegos. Además, *FACT* permite compaginar el paradigma constructivista con el aprendizaje basado en problemas y el aprendizaje guiado colaborativo.

Es posible incorporar *FACT* a una aplicación no colaborativa si ésta cumple unos requerimientos poco exigentes. La aplicación tiene que estar desarrollada en *Java* y permitir operaciones de *deshacer/rehacer*, además de utilizar el protocolo de *Java Beans*. Para adaptar una aplicación de este tipo a *FACT* se ha de implementar la interfaz de *FACT CollaborativeApplication*, que permite crear un agente de colaboración. Los eventos capturados por la aplicación se envían al agente en lugar de ser tratados directamente por la interfaz. El agente los envía automáticamente a las distintas aplicaciones que participan de la misma sesión, y un receptor centralizado de eventos de colaboración los trata en la forma en que antes lo hacía la interfaz inicial de la aplicación.

3.3 Sistemas para la enseñanza de Matemáticas

En esta sección describiré algunas aplicaciones para la enseñanza de las Matemáticas que han surgido en los últimos años, incluyendo la aplicación de *FACT* en este contexto para permitir el aprendizaje colaborativo guiado de materias que involucran la manipulación de fórmulas matemáticas.

Conviene mencionar en primer lugar a *Calculus Wiz*, [21], un curso interactivo desarrollado sobre *Matemática* que consiste en un conjunto de *cuadernos* que ayudan al alumno a resolver una gran cantidad de los problemas típicos de un primer año de cálculo. El alumno sólo necesita rellenar determinados campos, normalmente con fórmulas matemáticas, y pulsar botones para resolver los ejercicios que le plantea el sistema. Desde un punto de vista interactivo, probablemente es actualmente uno de los sistemas de enseñanza de Matemáticas más avanzados. Sin embargo no es una herramienta. Su contenido es fijo y, salvo nuevas versiones de la aplicación, el usuario no puede incrementar el contenido del curso por sí mismo. Posteriormente, *MathEdu,* [9], *Calculus Machina*, [19] y *EGrade*, [17], representan avances en aspectos importantes como la capacidad de extender problemas y métodos de resolución a otros más generales o de resolver problemas que incluyen la resolución

de otros más simples, la inclusión de una herramienta que permite que un profesor pueda definir cursos arbitrarios, la posibilidad de definir distintas estrategias para la resolución de diferentes tipos de problemas y la simplicidad de la interfaz del profesor para definir nuevos cursos, evitando la necesidad de tener conocimientos avanzados de programación. La Fig. 5 muestra una instantánea del diálogo de un estudiante con un curso desarrollado utilizando *MathEdu*.

Más recientemente, *ConsMath*, [15], [16], es una herramienta que permite la creación de conjuntos interactivos de problemas que involucran fórmulas matemáticas y su manipulación a partir de documentos estáticos. *ConsMath* tiene muchas características comunes con *MathEdu*, que en muchos aspectos es su antecesor, añadiendo a los cursos generados con esta herramienta la posibilidad de ser ofrecidos a través de Internet y la capacidad de trabajar en forma cooperativa guiada, para lo cual integra la funcionalidad del framework *FACT* descrito en la sección anterior. Además, *ConsMath* está diseñado para permitir la utilización de una metodología por la cual los conjuntos interactivos de problemas se desarrollan añadiendo capacidades interactivas sucesivas a partir de documentos estáticos, según se explica más adelante. La Fig. 6 muestra la arquitectura de *ConsMath.* En lo que sigue describiremos los aspectos más sobresalientes de *ConsMath*.

El primer concepto esencial que incorpora *ConsMath* es el de la generalización de un documento con fórmulas matemáticas, algo existente en otros sistemas para la creación de documentos con capacidad de manipulación simbólica. Mediante la generalización en un documento que incluya por ejemplo el cálculo de una integral se puede modificar una fórmula, como el integrando de la integral inicial, y el resto de las fórmulas que aparecen en el documento se actualizan automáticamente. En *ConsMath* esto se consigue mediante la utilización de un sistema de restricciones similar a las fórmulas que relacionan las casillas en una hoja de cálculo; el sistema de restricciones de *ConsMath* puede relacionar entre sí campos de texto, fórmulas matemáticas o componentes gráficas, que se actualizan automáticamente cuando cambia cualquier otra componente de la que dependen. La generalización de un documento exige únicamente añadir en cada campo dependiente una fórmula que diga cómo se calcula su contenido. Así, por ejemplo, para generalizar una integral que se calcula mediante un cambio de variable es necesario introducir una fórmula genérica que indica que hay que sustituir la integral de $f(g(x)).g'(x)dx$ por la de $f(u)du$. Además, en los campos que no dependen de otros se pueden indicar condiciones para la aceptación de la fórmula o datos que contienen. Así, en el ejemplo anterior se pondría la condición de que el integrando tenga la forma indicada para poder hacer el cambio de variables. Esto se puede hacer sin necesidad de tener conocimientos de programación gracias a la utilización de patrones de fórmulas y un procedimiento de *pattern matching* potente, algo que proporciona *Mathematica*.

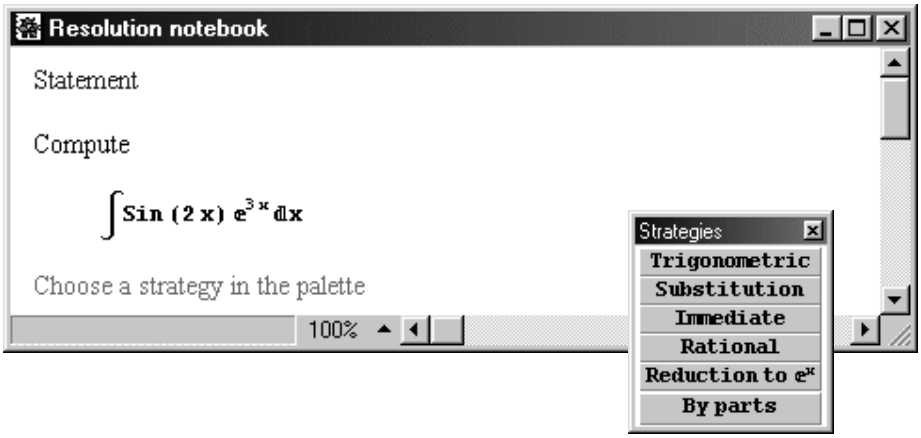

Fig. 5. Enunciado y paleta para la selección de estrategia en MathEdu

ConsMath, al igual que *MathEdu*, permite que el profesor determine un diálogo interactivo que tendrá lugar entre el estudiante y el sistema. El tipo de diálogo se puede ejemplificar mediante la resolución de una ecuación diferencial ordinaria como $y' = (\cos x)\, y - 5 \cos x$. Este problema lo genera automáticamente el sistema tras la generalización de otro semejante introducido por el profesor. Cuando el alumno intenta resolverlo, *ConsMath* le hace sucesivamente preguntas como las que se muestran a continuación.

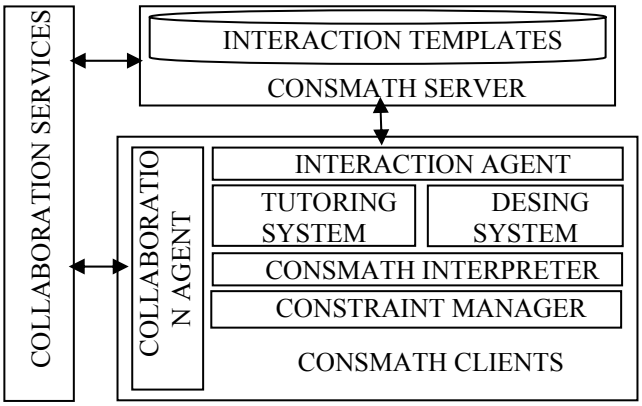

Fig. 6. Arquitectura de *ConsMath*

1. *De qué tipo es la ecuación que tienes que resolver?*
    a. *Ecuación lineal no homogénea*
    b. *Ecuación lineal homogénea*
    c. *Ecuación de Ricatti*

   Respuesta: a
2. *Escribe la ecuación homogénea que tienes que resolver.*
   *Respuesta: y'=(*cos *x) y*
3. *Qué método se utiliza para la resolución de la ecuación homogénea?*
    a. *Utilización de un factor integrante*
    b. *Separación de variables*
    c. *Computación directa*
4. *Escribe la solución general de la ecuación homogénea*
   Respuesta: $y=Ce^{\text{sen } x}$
5. *Cómo se calcula una solución particular de la ecuación original?*
    a. *Derivando la ecuación homogénea*
    b. *Sustituyendo la constante que aparece en la solución general de la ecuación homogénea por una función de x*
    c. *Derivando la solución general de la ecuación homogénea*

   Respuesta: b

A medida que el estudiante va contestando las respuestas acertadas se le van mostrando los distintos pasos de la resolución del problema. En caso de que se equivoque, se le muestran explicaciones que aclaran los conceptos involucrados. El proceso continúa hasta que el estudiante resuelve el problema por completo. El profesor, en el momento en que diseña el problema interactivo, puede decidir el diálogo concreto que tiene lugar, que puede ser diferente del anterior.

La especificación por el profesor de este diálogo se efectúa mediante una interfaz muy sencilla, que tiene la misma apariencia que el entorno en el que el estudiante resuelve el problema, excepto por la disponibilidad de funcionalidad adicional que se activa desde los menús y barras de herramientas. La Fig. 7 muestra la especificación del primer paso en un problema como el anterior.

El profesor tiene que determinar cómo continúa el trabajo del estudiante dependiendo de la respuesta que dé a cada una de las preguntas que se le hacen. Para ello, cuando termina de especificar una pregunta y cuál es la respuesta correcta se le muestra la parte anterior disponible del documento, ocultando la última pregunta; en ese momento el profesor puede editar el documento, añadiéndole más material proveniente del inicial o nuevo texto para esa situación específica. De esta forma, el profesor trabaja en cada momento en el mismo contexto que el estudiante que va a seguir el diálogo especificado. Para poder retroceder y determinar la forma en que reacciona el sistema cuando el estudiante da una respuesta diferente a la correcta, el profesor utiliza las flechas de avance y retroceso situadas en la parte superior de la

ventana de diseño. Con ello activa los mecanismos de *FACT*, que permiten a la interfaz volver atrás en el estado de la interfaz siguiendo su propia historia de diseño. Así pues, la especificación final de la resolución interactiva de un problema con todas sus variantes dependiendo de las acciones del usuario se representa mediante una historia de interacción con alternativas especificada por el profesor.

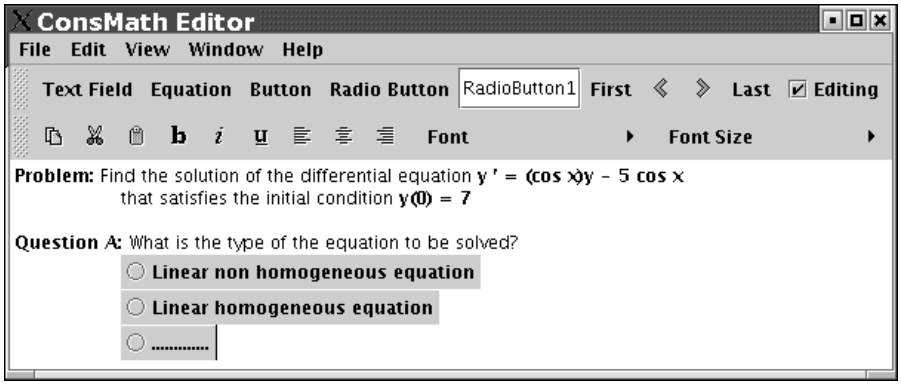

Fig. 7. Editor de ConsMath

El desarrollo de técnicas de diseño de interfaces que permiten al diseñador trabajar en el mismo contexto de los usuarios finales constituye un campo importante de investigación dentro del área de las interfaces de usuario, la programación mediante demostración, [8]. En el párrafo anterior hemos descrito los mecanismos novedosos de programación mediante demostración que utiliza *ConsMath*, basados en la utilización del *framework* para la enseñanza colaborativa guiada *FACT*.

Como ya hemos dicho, *FACT* no solamente proporciona la base para la representación mediante historias de interacción de los documentos interactivos de *ConsMath*, sino que también permite que el trabajo del estudiante se realice en un contexto de colaboración guiada. Ante un problema como el descrito anteriormente, no solamente el sistema hace un seguimiento de la ejecución de los distintos pasos por parte del estudiante, sino que el profesor puede comprobar el proceso que ha seguido, bien sea por iniciativa propia, a petición del estudiante o como consecuencia de la acción de un agente que detecta la existencia de dificultades en el proceso de resolución. Tras ello, el profesor puede proponerle al estudiante otras alternativas para realizar su trabajo.

Referencias

[1] Appelt, W.: What Groupware Functionality do Users Really Use? In Proceedings of the 9th Euromicro Workshop on PDP 2001, Los Alamitos, IEEE Computer Society (2001)

[2] de Bra, P., Calvi, L.: AHA! An open Adaptive Hypermedia Architecture, The New Review of Hypermedia and Multimedia, 4, 115-139, Taylor Graham Publishers (1998)

[3] Brusilowsky, P.: Methods and techniques of Adaptive Hypermedia, en Brusilovsky, P., Kobsa, A., Vassileva, J. (Eds.), Adaptive Hypertext and Hypermedia, Dordrecht, Kluwer Acad. Publ. (1998)

[4] Brusilowsky, P., Eklund, J., Schwarz, E.: Web based education for all: A tool for the development of adaptive courseware, Computer Networks and ISDN Systems, 30 (1-7) 291-300 (1998)

[5] Carro, R., Pulido, E., Moriyón, R., Rodríguez, P.: Teaching Tasks in an Adaptive Learning Environment, In Human-Computer Interaction. Communication, Cooperation and Application Design, Vol. 2, 740-744, Lawrence Erlbaum Ass. Pub. (1999)

[6] Carro, R., Pulido, E., Rodríguez, P.: TANGOW: A model for Internet Based Learning. Intenational Journal of Continuing Engineering Education and Life-Long Learning (IJCEELL), 11, 1/2 (2001)

[7 Castells, P., Szekely, P.. Presentation Models by Example. En D.J. Duke and A. Puerta, eds., Design, Specification and Verification of Interactive Systems '99 100-116, Springer-Verlag, Viena, (1999)

[8] Cypher, A.: Watch what I do. Programming by Demonstration, MIT Press, Cambridge, MA (1993)

[9] Díez, F., Moriyón, R.: Doing Mathematics with MathEdu, proc. IXth Conference of Mathematics/Science Education & Technology , AACE (1999) 247-251

[10] Graham, T.C.N., Grundy, J.C.: External Requirements of Groupware Development Tools. En Engineering for Human-Computer Interaction 363-376, Dordrecht, Kluwer Acad. Publ. (1999)

[11] Jermann, P., Soller, S, Muehlenbrock, M.: From Mirroring to Guiding: A Review of State of the Art Technology for Supporting Collaborative Learning. Proceedings of ECSCL01, Maastricht (2001)

[12] Macías, J. A., Castells, P.: Interactive Design of Adaptive Courses, en Ortega, M. y Bravo, J. (eds.), Computer and Education: Towards an Interconnected Society 235-242, Dordrecht, Kluwer Acad. Publ. (2001)

[13] Mora, M.A., Moriyón, R.: Guided collaborative chess tutoring through game history analysis, Proceedings SIIE'2000, Universidad de Castilla-La Mancha, Spain (2000)

[14] Mora, M., Moriyón, R.: Collaborative Analysis Tutoring: The Fact Framework, in Learning Technology: Issues, Achievements and Challenges, Los Alamitos, IEEE Computer Society Press (2001)

[15] Mora, M., Moriyón, R., Saiz, F.: Mathematics Problem-Based Learning Through Spreadsheet-Like Documents, in Proc. Second International Conference on the Teaching of Mathematics (ICTM2), N. York, John Wiley & Sons Inc. (2002)

[16] Mora, M., Moriyón, R., Saiz, F.: Building Mathematics learning applications by means of ConsMath, preprint EPS UAM (2003)

[17] Orr, J.L., Franklin, B.: Egrade: Student Learning Guide, John Wiley and Sons (2000)

[18] Plaisant, C., Rose, A., Rubloff, G., Salter, R., Shneiderman, B.: The Design of History Mechanisms and their Use in Collaborative Educational Simulations. In Proceedings of the Computer Support for Collaborative Learning (CSCL) 1999 Conference, C. Hoadley & J. Roschelle (Eds.), Lawrence Erlbaum (1999)
[19] Quinney, D.: Calculus Machina: An intelligent tutor providing computer based support for teaching undergraduate calculus. Proc. 2nd Int. Conference on the Teaching of Mathematics. Hersonissos, Crete (2002)
[20] Schoenfeld, A.H.: Learning to Think Mathematically: Problem Solving, Metacognition, and Sense-Making in Mathematics, in D. Grouws, Ed., Handbook for Research on Mathematics
Teaching and Learning 334-370, New York, MacMillan (1992)
[21] Stroyan, K.D.: Will Yesterday's Calculus Education Survive Mathematica?, Worldwide Mathematica Conference, Wolfram Research (1998)
[22] Vasilieva, J.: A Task-Centred Approach for User Modeling in a Hypermedia Office Documentation System. En Brusilovsky, Pl, Kobsa, A., Vassileva, J. (Eds.), Adaptive Hypertext and Hypermedia 209-247, Dordrecht, Kluwer Acad. Publ. (1998)
[23] Weber, G., Specht, M.: User modeling and adaptive navigation support in www-based tutoring systems, Proceedings of User Modeling'97, 289-300 (1997)

Accesibilidad a Interfaces Móviles para Computación Ubicua Relativa al Contexto

Julio Abascal González

Laboratorio de Interacción Persona-Computador para Necesidades Especiales
Universidad del País Vasco-Euskal Herriko Unibertsitatea
Facultad de Informática
Manuel Lardizabal 1, 20018 Donostia-San Sebastián
julio.abascal@si.ehu.es

Resumen. El rápido desarrollo de las diversas tecnologías implicadas en la computación ubicua ha permitido la realización de grandes avances en esa dirección. Los sistemas de computación móviles dotados de dispositivos de interacción adecuados, a veces integrados en la ropa, que permiten la interacción a través de redes inalámbricas con procesadores empotrados en el entorno, pueden ofrecer nuevos servicios en función del contexto. Estos servicios resultan especialmente útiles para las personas con discapacidad que sufren limitaciones en el uso de dispositivos de interacción estándar. Sin embargo, para que estos sistemas puedan ser utilizados por todas las personas es necesario que la tecnología y los procedimientos sobre los que se asientan sean accesibles. En este capítulo se hace un repaso de los aspectos básicos de la computación ubicua para, posteriormente, identificar los problemas de accesibilidad que plantean las interfaces móviles y discutir técnicas de diseño que garanticen la accesibilidad de este tipo de productos y servicios.

1 Introducción

Desde que Mark Weiser publicara lo que podríamos denominar el "manifiesto" de la computación ubicua [1], el rápido desarrollo de diversas tecnologías ha permitido que una gran parte de sus planteamientos sean hoy en día posibles. Algunos de estos avances tienen que ver con la miniaturización de los computadores aplicada al diseño de procesadores empotrados y de ordenadores móviles (palm-tops, PDAs, teléfonos, tarjetas inteligentes, etc.), con los avances de las redes inalámbricas basadas en radiofrecuencia (entre las que destacan Bluetooth[1] e IEEE802.11[2]) o en infrarrojos (tales como IrMC[3]), con los sistemas de localización en interiores y exteriores (como el GPS), etc.

[1] http://www.bluetooth.com/
[2] http://www.computer.org/students/looking/summer97/ieee802.htm
[3] http://www.irda.org/standards/pubs/iMelody.pdf

Evidentemente la computación ubicua no nace con el artículo de Mark Weiser. Anteriormente se habían plantado las bases de una computación integrada en el entorno que obtenía servicios a través de redes inalámbricas de área local. Curiosamente, muchos de estos esfuerzos fueron realizados en el contexto de la Tecnología de la Rehabilitación (denominada en inglés *Assistive Technology*) tratando de paliar los problemas que encuentran las personas con discapacidad para manipular su entorno o para comunicarse con los demás. Así pues, se puede decir que los sistemas inalámbricos para el control de entorno usados en domótica, las redes locales basadas en infrarrojos para acceso a los ordenadores, etc., son precursores de la computación ubicua. Paradójicamente, el nuevo impulso que han tomado estos sistemas puede dejar a un lado a las personas con discapacidad si no se tienen en cuenta los requisitos de accesibilidad a la hora de desarrollar aplicaciones ubicuas.

En los siguientes apartados vamos a hacer una breve revisión de las tecnologías que intervienen en la computación ubicua con especial hincapié en las interfaces para sistemas móviles. Analizaremos las condiciones de accesibilidad y estudiaremos un método para diseñar sistemas usables y accesibles.

2 Computación ubicua

Si atendemos a su etimología, la computación ubicua[4] es aquella que está en todas partes, en oposición a la que está ligada a ubicaciones concretas. Esta última incluiría a un amplio rango de ordenadores que van desde las grandes instalaciones a los ordenadores personales. Sin embargo, los ordenadores típicos de la computación ubicua son pequeños sistemas móviles (PDAs, *handhelds*, teléfonos) comunicados mediante redes inalámbricas y procesadores empotrados en el entorno que, a su vez, pueden estar conectados entre sí mediante redes inalámbricas o cableadas.

La computación ubicua ha evolucionado muy rápidamente incluyendo nuevos enfoques y tecnologías que se apartan de la concepción inicial[5]. Buscando una mayor precisión se han propuesto nuevas denominaciones, tales como *proactive computing* o *autonomic computing,* que no han llegado a arraigar. Sí lo ha hecho la expresión *pervasive computing* que en la literatura anglosajona se propone como sinónimo de *ubicuitous computing*[6]. De todos modos, la palabra inglesa "pervasive" puede ser traducida al castellano por penetrante u omnipresente, lo que se podría interpretar como una sutil ampliación en la idea de computación ubicua, que se referiría a la computación que inunda todo el espacio (computación penetrante, omnipresente, diseminada…).

---

[4] Según el diccionario de la RAE, la palabra "ubicuo" se deriva del latín *ubīque*: "en todas partes".

[5] Según Satyanararayanan [2], el mismo Weiser no parecía estar completamente satisfecho con la expresión "ubiquitous computing" propuesta inicialmente por él [1].

[6] M. Satyanararayanan, editor jefe de la revista IEEE Pervasive Computing, opina que ambas expresiones son equivalentes: “This magazine will treat ubiquitous and pervasive computing as synonymous” [2].

2.1 Computación relativa[7] al contexto

Además de la omnipresencia, la otra característica básica de la computación ubicua es la posibilidad de interactuar con el entorno, de manera que puede ofrecer servicios dependientes de la localización del usuario. Esto no es del todo novedoso, salvo en que la computación ubicua pone el acento en una interacción "sin costuras" (*seamless*) con el entorno. Es decir, los cambios de contexto del usuario pueden requerir cambios en el modo en el que el sistema provee el servicio que está utilizando. Estos cambios se realizarán automáticamente por el sistema, sin necesidad de intervención explícita del usuario (lo que no quiere decir que éste no sea consciente de ellos, ya que en alguno casos pueden significar que se producen variaciones en la calidad del servicio).

La interacción con el contexto requiere la localización de la posición del elemento móvil. Para ello existen múltiples técnicas que se diferencian en el alcance (área de localización que cubren), en la precisión con la que localizan el objeto móvil, en la fiabilidad de sus cálculos, en la necesidad o no de instalaciones especiales, el coste total, etc. [3]. El tipo de localización necesario puede depender del servicio al que se desee acceder. En algunos casos se necesitan las coordenadas espaciales del objeto, mientras que en otros bastaría con detectar si está dentro del área cubierta por el detector.

La localización precisa de personas ha sido desde antiguo uno de los objetivos de la Tecnología de la Rehabilitación. Las personas con limitaciones cognitivas (que pueden ser debidas a diversas causas, tales como discapacidad, envejecimiento, demencia, etc.) pueden desorientarse y perderse fácilmente. Un sistema de localización ayuda a encontrarlas rápidamente cuando esto ocurre. La localización es muy útil también en interiores. Permite aumentar la seguridad de las personas que viven solas mediante la observación de la verticalidad y la existencia o no de movimiento [4]. En los sistemas domóticos avanzados diseñados para personas con discapacidad y ancianas, los sistemas de localización tienen otras funciones añadidas, tales como ayudar al ahorro de energía (por ejemplo apagando las luces de las habitaciones desocupadas) y a la provisión de servicios que dependen de la situación de la persona (por ejemplo para el control de entorno) [5].

3 Interfaces móviles

Una de las características más importantes de los sistemas ubicuos es la interfaz de usuario. Según Mark Weiser, el ordenador, al menos tal como lo conocemos hoy, desaparecerá en un periodo de tiempo no muy largo[8]. En su opinión, el ordenador

[7] A pesar de que algunos autores han propuesto computación consciente (o conciente) del contexto para traducir la expresión *context aware computing*, he preferido usar computación relativa al contexto, ya que las palabras consciente y conciente me resultan demasiado antropomórficas.

[8] "For thirty years most interface design has been headed down the path of the “dramatic” machine. Its highest ideal is to make a computer so exciting, so wonderful, so interesting,

estará integrado en otros dispositivos y no seremos conscientes de su existencia más que por las funciones que ofrezca. Esto parece significar la desaparición de la interfaz de usuario "explícita" y la necesidad de desarrollo de interfaces "implícitas" enfocadas a una tarea concreta, más inteligentes, y capaces de dialogar con otros elementos. De este modo, la computación ubicua puede hacer que los servicios que provea el computador sean tan móviles como sus usuarios e, incluso, aprovechar los constantes cambios del contexto en que son usados. Esto daría lugar a entornos activos en los que los ordenadores interaccionan entre sí y con el usuario de manera inteligente.

3.1 Interacción móvil

Las interfaces de usuario tradicionales presentan diversas dificultades para ser usadas en dispositivos de pequeño tamaño, con elementos de entrada/salida diferente y en situaciones especiales (que pueden incluir el desplazamiento del usuario). El diseño de interfaces para sistemas móviles requiere un esfuerzo creativo[9] para detectar nuevos dispositivos y procedimientos de interacción que permitan una comunicación usuario-ordenador más natural (similar a la que ocurre entre personas) y que no impida la realización simultanea de otras tareas [6].

3.2 Nuevos dispositivos y procedimientos de interacción

En los siguientes apartados vamos a recordar brevemente algunos dispositivos de interacción alternativos adecuados para la interacción móvil.

3.2.1 Computación *vestible*

Hemos visto que la interacción móvil requiere procesadores pequeños y ligeros que no interrumpan las actividades normales del usuario. La mejor forma de no ocupar las manos y la vista del usuario móvil es utilizar dispositivos integrados en la ropa o en los adornos. Existen ya diferentes sistemas de ayuda personal y profesional (agendas, sistemas para navegación en carretera o en ciudad, ayuda para la realización de tareas, acceso a telefonía e internet, etc.) basados en lo que se denomina *wearable computing* (y que aquí llamaremos computación "vestible"). Independientemente de la función de este tipo de dispositivos, está claro que los ordenadores que "se llevan puestos",

---

that we never want to be without it. A less-travelled path I call the "invisible", its highest ideal is to make a computer so imbedded, so fitting, so natural, that we use it without even thinking about it. (I have also called this notion "Ubiquitous Computing" and have placed its origin in post-modernism)" [1].

[9] En los últimos años se vienen celebrado una serie de congresos y *workshops* que tratan de profundizar en la necesaria innovación en el diseño de interfaces para sistemas móviles. La serie de workshops Mobile HCI es una buena muestra de ello (MobileHCI03: http://hcilab.uniud.it/mobilehci/index.html).

entre otras características propias, necesitan un sistema especial de interacción con el usuario.

La investigación en esta área es sumamente activa. Para hacerse una idea de las tendencias actuales se puede consultar el número monográfico dedicado a Wereable Computing de la revista Pervasive Computing [7].

3.2.2 Interfaces de voz

Buscando sistemas móviles que no entorpezcan otras actividades del usuario y que sean más eficientes que teclear o mover el ratón (y al mismo tiempo más naturales), parece lógico pensar en una interacción basada en el uso de la voz. Las interfaces de lenguaje natural para sistemas móviles deberán permitir el acceso a las aplicaciones mediante diálogos similares a los mantenidos por las personas, de manera que el ordenador vaya extrayendo de ellos la información que necesita para saber qué desea el usuario, resuelva las ambigüedades y aclare los posibles malentendidos, posiblemente mediante preguntas.

Este no es un objetivo novedoso. Desde hace tiempo muchos investigadores están empeñados en conseguir "hablar con el ordenador". Pero sabemos que el problema no es fácil. El uso del lenguaje natural requiere el perfeccionamiento de los sistemas de reconocimiento y síntesis de voz y de los métodos de compresión del lenguaje [8].

El reconocimiento de la voz humana está avanzando considerablemente de manera que, en un futuro próximo, será posible disponer de sistemas que reconozcan discurso continuo, independientemente del hablante [9]. Por otro lado, la combinación de diversas técnicas, tales como la percepción a través de los sistemas de procesamiento de la señal, del aprendizaje mediante redes neuronales y del análisis (morfológico, sintáctico y semántico) del discurso en tiempo real mediante técnicas de inteligencia artificial posiblemente permitan resultados parecidos a los que consiguen las personas a la hora de entender el lenguaje hablado, en un plazo no muy largo.

3.2.3 Nuevos dispositivos de interacción

El uso de la voz para la interacción con el ordenador también tiene sus limitaciones. Por ejemplo, no permite salvaguardar la intimidad cuando se usa en sitios públicos y presenta dificultades para la manipulación en interfaces de dos dimensiones (que usualmente se manejan con dispositivos apuntadores). Por ello, además de la voz, pueden ser necesarios otros dispositivos tales como teclados reducidos[10] para entrada de datos o emuladores de ratón de diversos tipos para cuando se necesitan dispositivos apuntadores. En este último caso, si queremos que las manos queden libres, lo más lógico es recurrir al rastreo del movimiento de los ojos [11].

---

[10] Existen ya teclados reducidos que tienen asignada más de una letra a cada tecla (por ejemplo, las 28 letras repartidas entre 12 teclas). Los textos escritos mediante estos teclados, con una sola pulsación por letra, resultan ambiguos pero pueden ser descodificados por el propio sistema. [10].

Cuando la salida se produce mediante voz sintética, posiblemente esté dirigida al oído del usuario mediante auriculares, para no molestar a los demás y garantizar la confidencialidad. Por motivos de seguridad, resulta necesario que los auriculares transmitan también la información sonora que el usuario recibe del exterior (palabras y sonidos) para que el uso del ordenador no lo aísle del entorno. También son previsibles salidas gráficas sobre gafas de cristal líquido que, al igual que los auriculares, permitan percibir simultáneamente la información procedente del entorno y la que sale del ordenador.

La gama de posibles dispositivos de interacción en computación ubicua es enorme, como ha demostrado ampliamente la ciencia ficción, pero para que muchos de esos dispositivos sean posibles es necesario resolver antes problemas básicos, como la disponibilidad de baterías de alta duración[11] y bajo peso.

4 Accesibilidad a las interfaces

Las barreras que los usuarios discapacitados y ancianos encuentran para interactuar con los entornos inteligentes están relacionadas principalmente con la interfaz de usuario e incluyen las dificultades físicas para manipular los dispositivos y las barreras cognitivas para entender los procedimientos y la navegación [12]. Los estudios realizados con usuarios evidencian la necesidad de interfaces adaptables que permitan el control de dispositivos y servicios, de manera homogénea, a través de sistemas ínteroperables integrados en un entorno inteligente.

4.1 Accesibilidad física

Las interfaces estándar se basan en el uso de los dispositivos de interacción más comunes: el teclado y ratón para la entrada de datos y la pantalla (y ocasionalmente los altavoces para señales audibles) para la salida. El uso de estos dispositivos requiere determinadas capacidades físicas. La entrada demanda precisión y coordinación motora, además de coordinación visual-motora para el manejo del dispositivo apuntador. La salida requiere capacidad visual y ocasionalmente auditiva [13].

Los seres humanos presentan una gran diversidad en sus capacidades, de manera que una fracción importante de la población no alcanza los mínimos necesarios para manejar estos dispositivos de manera adecuada. Esto puede ocurrir por diversas causas tales como envejecimiento, discapacidad o por estar realizando simultáneamente otra tarea (como conducir o trabajar). Este último caso se ha incorporado recientemente al conjunto de necesidades especiales debido a la aparición de los sistemas ubicuos, que pueden ser utilizados mientras el usuario se desplaza o realiza actividades diversas.

[11] De hecho la disponibilidad de energía es uno de los principales condicionantes de los sistemas móviles.

4.2 Accesibilidad cognitiva

Las interfaces regulan el diálogo usuario-aplicación mediante una serie de procedimientos que incluyen las órdenes disponibles, los procedimientos de navegación, etc. Estos elementos se encuadran en un modelo de la tarea a realizar que suele ser explicitado como una metáfora de la misma actividad realizada sin la ayuda del ordenador. Para conseguir un uso adecuado la persona debe comprender los procedimientos, las metáforas, la navegación, etc., lo que en definitiva depende del ajuste entre la "visión del mundo" que tiene el usuario y la que tiene la aplicación.

También las capacidades cognitivas de los usuarios son muy diversas [14, 15]. Además del envejecimiento y las discapacidades cognitivas, aspectos tales como el uso de un idioma diferente de la lengua materna o la disminución de la atención al realizar otra tarea simultáneamente pueden influir en la capacidad cognitiva, por lo que también es necesario tener en cuenta esta diversidad a la hora de diseñar métodos de interacción.

4.3 Diseño universal

Usualmente las inveraces se diseñan pensando en una persona estándar con todas las capacidades físicas y cognitivas, lo que frecuentemente deja fuera a los colectivos de personas con "necesidades especiales". El diseño universal, también conocido como diseño para todos, tiene como objetivo el diseño de interfaces que no presenten barreras de accesibilidad [16]. Para ello es necesario que la interfaz admita el uso de dispositivos de interacción alternativos, adecuados a las capacidades físicas de cada usuario. La tecnología de la rehabilitación (*Assistive Technology*) ha desarrollado múltiples dispositivos para las personas con diversas limitaciones motoras, visuales, etc. [17, 18].

Respecto de la necesidad de tener en cuenta la diversidad de capacidades cognitivas, es necesario, por ejemplo, limitar la necesidad de memorizar datos, utilizar diversas metáforas adecuadas a las diversas culturas y experiencias previas de los usuarios y producir sistemas de navegación coherentes e intuitivos. A pesar de que este tipo de barreras afectan a un colectivo muy amplio, que incluye a personas no consideradas como discapacitadas, los estudios de accesibilidad cognitiva están menos desarrollados que los de accesibilidad física.

4.4 Accesibilidad y usabilidad

Parece evidente que las interfaces orientadas a la accesibilidad deben ser también usables. Es difícil concebir una interfaz accesible que no cumpla las condiciones de usabilidad. La cuestión es si las interfaces usables son también accesibles. Algunos autores opinan que accesibilidad y usabilidad no son separables[12]. Para ellos una interfaz que no sea accesible no es usable. Pero lo cierto es que la mayoría de los estándares de usabilidad actuales no cumplen las condiciones de accesibilidad.

[12] Ver http://ww.usablenet.com/accessibility_usability/accessibility_usability.html

5 Aplicaciones de la computación ubicua para personas con discapacidades

Tal como se ha visto previamente, los avances recientemente experimentados por diversas tecnologías posibilitan el diseño de sistemas ubicuos particularmente adecuados para personas con necesidades especiales (principalmente personas con discapacidad y ancianas) [19]. Los diversos tipos de entornos inteligentes que dan servicio a las personas con discapacidad encajan muy bien dentro del concepto de *inteligencia ambiental* adoptado por la Comunidad Europea como objetivo de investigación [20].

5.1 Hogares inteligentes

Tal como hemos comentado, los primeros pasos en computación ubicua aparecen relacionados con la domótica. El objetivo es el desarrollo sistemas de control centralizado de los recursos del hogar que tratan de facilitar la interacción del usuario con el entorno diario. Para ello, fundamentalmente se encargan de la gestión de los recursos (temperatura y luz ambiental…) de las alarmas (escapes de agua, gas, derivaciones eléctricas, detección de intrusos…).

Actualmente, la combinación de redes inalámbricas de área personal con los diversos tipos de redes cableadas existentes en el hogar, interactuando con sistemas de localización y buses de sillas de ruedas de tracción eléctrica, permiten el desarrollo de hogares inteligentes (Smart Homes) que van más allá de los objetivos de la domótica clásica. Al control remoto y centralizado de los diversos elementos del hogar se añade la posibilidad de dotar al contexto de cierta autonomía para tomar decisiones "inteligentes" que permiten la adaptación a las necesidades y gustos de los usuarios. Además se añaden diferentes funciones complementarias de gran utilidad, tales como el acceso inalámbrico a diversos tipos de comunicaciones (mediante texto, voz, multimedia, etc.).

Los avances técnicos en el desarrollo de hogares inteligentes conducen a importantes avances sociales. Las personas con discapacidad y ancianas reivindican frecuentemente su derecho a llevar una vida autónoma fuera de una institución y los hogares inteligentes ofrecen una serie de servicios que facilitan la realización de este objetivo. Por ejemplo, además de facilitar el control remoto de todos los dispositivos y electrodomésticos de la casa (lo que resulta vital para las personas con movilidad restringida), pueden colaborar a garantizar la seguridad del usuario mediante la gestión de las alarmas, la monitorización de determinadas constantes vitales, la supervisión de la localización del usuario, etc.[13]

[13] Por ello, el desarrollo de entornos inteligentes para personas con necesidades especiales en uno de los objetivos estratégico de la CE para el VI Programa Marco http://www.hltcentral.org/usr_docs/call_docs/FP6-call1/wp2003-04_final.pdf.

5.2 Sistemas para personas con movilidad restringida

Curiosamente, la computación móvil presenta características muy adecuadas para las personas con movilidad restringida. Estos usuarios presentan dificultades para acercarse a los diversos dispositivos que utilizan para comunicarse, controlar el entorno, acceder a redes telemáticas, etc. Las redes inalámbricas, las interfaces móviles, ayudan a que estas personas puedan acceder a servicios que de otra manera les sería imposible. Muchas de estas necesidades coinciden con las vistas en el apartado anterior dentro de las casas inteligentes, pero también existen otros contextos de utilización: en el trabajo (acceso al ordenador), en el desplazamiento por el exterior (acceso a transporte urbano, orientación en ciudad…), etc. Una interesante aplicación son las tarjetas inteligentes sin contacto (*contactless smart cards*) que permiten a los usuarios interaccionar a cierta distancia con diversos elementos activos. Si observamos los problemas que presentan, por ejemplo, los cajeros automáticos para las personas con movilidad restringida o con problemas de visión, entenderemos el interés de una tarjeta inteligente que permita acceder a los servicios del cajero sin necesidad de ser introducida por la ranura[14].

Las tarjetas inteligentes [21] permiten almacenar grandes cantidades de información de manera que pueden ser usadas para múltiples servicios, tales como acceso a cuentas bancarias, almacenamiento criptografiado del historial médico de la persona (lo que resultaría especialmente útil en caso de accidente o cambio de médico por cualquier causa), como llave electrónica para el acceso al puesto de trabajo, al hogar y a todo tipo de entornos inteligentes, etc.

Este tipo de tarjetas ha sido adecuadamente estandarizado[15] lo que permitiría, por ejemplo, concentrar en una sola las múltiples tarjetas que utiliza una misma persona para múltiples propósitos. A pesar de las indiscutibles ventajas, las tarjetas inteligentes no se han expandido suficientemente, seguramente debido a problemas comerciales, tales como la competencia entre las diversas empresas de tarjetas de crédito, y a algunos problemas de usabilidad, tales como la resistencia del usuario a la transmisión de información sensible a cierta distancia.

5.3 Acceso a las aplicaciones de la computación ubicua de uso general

Las necesidades de las personas con discapacidad no se restringen a los sistemas especialmente diseñados para ellas. La integración de estos usuarios requiere que todos los sistemas de interacción ubicua sean accesibles. Teniendo en cuenta que se supone al usuario dotado de una interfaz que es capaz de manejar, el problema se restringe a impedir que las aplicaciones ubicuas añadan barreras innecesarias que

[14] El proyecto europeo SATURN se centró en el estudio de las posibilidades y restricciones de las tarjetas inteligentes para las personas con discapacidad y ancianas. [http://phoenix.herts.ac.uk/SDRU/SATURN/saturn.html]

[15] ISO: ISO/IEC 10536 [http://www.sc17.com/wg.cfm?Wg=WG8] y CEN: prEN 1332.4 [http://www.e-europestandards.org/Activities_Smart_cards.htm]

impidan el uso de interfaces adaptadas para acceder a ellas[16]. Para ello es necesario crear pautas de diseño integrador (como las existentes en otras áreas [22]) que permitan el diseño de sistemas ubicuos sin barreras añadidas.

6 Diseño de sistemas ubicuos orientado a la accesibilidad

Como cualquier otra, la interfaz de un sistema móvil debe ser diseñada teniendo en cuenta requisitos de usabilidad y accesibilidad. Para ello es necesario utilizar una metodología de diseño rigurosa que permita recolectar las necesidades de usuario, las tareas que se van a realizar y el contexto de trabajo. Una metodología sencilla y muy adecuada para ser usada en este contexto es *USERfit*.

6.1 *USERfit*: metodología de diseño para la usabilidad y la accesibilidad

USERfit[17] es una metodología para la generación de especificaciones de usabilidad desarrollada para el ámbito de la Tecnología de la Rehabilitación, pero que ha demostrado ser útil para aplicaciones dirigidas a cualquier tipo de usuarios [23]. Debido a que la información que se va generando se consigna en formularios sobre papel, la inclusión y eliminación de nuevos usuarios o contextos de uso y la necesidad de propagar los resultados de unos formularios a otros puede llegar a hacer que su uso sea tedioso. *USERfit Tool*[18], es una aplicación desarrollada con el objetivo de facilitar el uso del entorno de diseño *USERfit,* disminuyendo la carga añadida que supone la gestión de la información que se produce a lo largo del proceso. Además, esta herramienta permite reutilizar material previamente desarrollado y compartir un mismo diseño entre grupos de diseñadores remotos, manteniendo la coherencia y la compatibilidad [24, 25][19].

---

[16] Es el mismo problema que presenta la que la accesibilidad a internet. Se supone que el usuario está dotado de una interfaz que le permite manipular un determinado navegador. Pero para garantizar la accesibilidad es necesario asegurar que ni el contenido ni la estructura de las páginas web se basan en características no identificables por el navegador. Es decir que, por ejemplo, los iconos que sirven de enlace tengan un texto alternativo que los identifique. De otra manera un navegador de sólo texto, usado por muchos invidentes, no podría "leer" ese enlace. Las pautas de accesibilidad de la WAI tratan de definir las características de las páginas web para que sean accesibles. http://w3c.wai.org

[17] La metodología Userfit fue desarrollada dentro del proyecto europeo USER (TIDE-1062) por HUSAT Research Institute (UK), SINTEF Unimed Rehab (Norway) and COO.S.S. Marche scrl (Italy).

[18] El desarrollo de Userfit Tool ha sido parcialmente financiado por la Comisión Europea dentro del proyecto IST-2000-26211 IRIS: *Incorporating Requirements of People with Special Needs or Impairments to Internet-based Systems and Services*. Desarrollado por European Dynamics (GR), University of the Aegean (GR), Fraunhofer Institute for Applied Information Technology (D), Information Society disAbilities Challenge (B), Universidad del País Vasco-Euskal Herriko Unibertsitatea (E) http://www.iris-design4all.org/.

[19] Userfit Tool es una herramienta de uso libre que se puede obtener en la página web del LIPCNE: http://scsx01.sc.ehu.es/acwusfit/

6.2 Accesibilidad a interfaces móviles

Tal como hemos visto, las interfaces móviles actuales presentan problemas de accesibilidad física, en parte debido a su pequeño tamaño. Tanto la entrada, ya sea mediante un teclado reducido, reconocimiento de texto manuscrito [26], reconocimiento de voz, o dispositivo apuntador, como la salida, mediante visualizadores (*displays*) de tamaño reducido o voz sintética, presentan problemas para las personas con discapacidad. Además, la interfaz cognitiva (que, como sabemos, incluye los procedimientos que se explicitan a través de la interfaz física) resulta inaccesible si el usuario no comprende su manejo y no puede desarrollar la tarea a la que el dispositivo está destinado, bien por que no entiende los procedimientos (navegación, manipulación de la información) o bien por que tiene una concepción de la tarea diferente de la del sistema.

6.3 Interfaz TetraNauta

Para dar una visión práctica de los conceptos relacionados con accesibilidad a interfaces móviles, vamos a utilizar algunas ideas procedentes del diseño de la interfaz TetraNauta[20].

El proyecto TetraNauta diseñó una silla de ruedas de tracción eléctrica autónoma, destinada a las personas con severas discapacidades motoras que experimentan dificultades para utilizar los sistemas de guiado estándar. La novedad con respecto de otras sillas de ruedas "inteligentes" (*smart wheelchairs*) era tratar de mantener el sistema dentro de un coste asequible. Esto significa que fue necesario limitar la calidad y cantidad de los sensores típicamente usados para el control de vehículos guiados autónomamente (AGV) que utilizan profusamente las sillas de ruedas inteligentes. Como compensación, se trató de aumentar la inteligencia del sistema de control aprovechando que la capacidad de cálculo es relativamente más barata.

Una de las claves de este sistema es la interfaz de usuario, que debe minimizar el esfuerzo físico y cognitivo del usuario para permitirle manejar la silla aunque su capacidad de movimiento sea muy restringida. Por otro lado, dado que los usuarios experimentan variaciones en sus capacidades debido a factores tales como el entrenamiento y la fatiga, es necesario que la interfaz sea auto adaptable [27]. Además debe colaborar a la rehabilitación haciendo que el usuario tome el máximo de decisiones que pueda, para mantenerle activo e interesado [28].

[20] *TetraNauta* es un proyecto de investigación financiado por la CICYT y desarrollado por el Hospital Nacional de Parapléjicos de Toledo; Bioingeniería Aragonesa S. A.; la Universidad de Sevilla y el Laboratorio de interacción Persona-Computador para Necesidades Especiales de la Universidad del País Vasco-EHU.

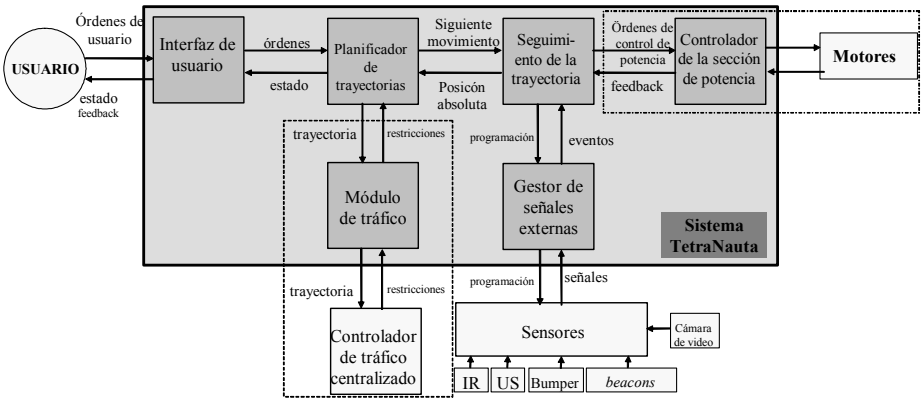

Fig. 1. Arquitectura del sistema TetraNauta

Sin entrar en los detalles del desarrollo de la interfaz TetraNauta[21], vamos a presentar los planteamientos básicos para ilustrar el método de diseño empleado.

6.4 Diseño para la accesibilidad

El diseño de una interfaz móvil accesible tiene que basarse en:

- dispositivos de entrada que el usuario pueda manipular
- dispositivos de salida que el usuario pueda entender
- procedimientos que coincidan con la visión del problema que tiene el usuario
- sistemas de navegación intuitivos

Además, dada la dificultad que puede tener la manipulación de la interfaz y la necesidad de interrumpir las otras actividades del usuario lo menos posible, es necesario que la información requerida al usuario sea mínima. Para ello, la interfaz debe comportarse de manera "inteligente", es decir que tome una actitud activa y, basándose en la información que pueda recopilar sobre el usuario, la tarea que este quiere realizar y el contexto en que la realiza, tome el máximo de decisiones en función de esa información y evite pedir datos al usuario que puede deducir.

La información sobre el usuario, la tarea y el contexto se trata a través del modelado, como veremos en la siguiente sección.

[21] El sistema TetraNauta está siendo ampliado mediante dos nuevos proyectos: DomoSilla, *Estudio, evaluación y diseño de un sistema de interconexión entre red local para el control de sillas de ruedas (Dxbus) con red domótica (EHS).* Subvencionado por el MCYT (TIC2000-0087-P4-03. MP4) y desarrollado por U. de Sevilla, UPV-EHU y Bioingeniería Aragonesa S.L., que está diseñando el control de un sistema domótico mediante la misma interfaz con que el usuario gobierna la silla de ruedas eléctrica y Heterorred, *Estudio y desarrollo de una red heterogénea de área personal/local para la interoperabilidad y el acceso a servicios y comunicaciones inalámbricos,* subvencionado por el MCYT (TIC2001-1868-C03-03.) que trata de facilitar la interoperabilidad entre redes heterogéneas para la integración de TetraNauta en entornos inteligentes.

6.5 Modelado

Como hemos visto previamente, las interfaces de los sistemas móviles pretenden disminuir el esfuerzo del usuario en la interacción, para lo que tratan de actuar de manera "inteligente". El sistema hace deducciones a partir de la información de que dispone, de modo que su comportamiento se adecue a lo que supone que el usuario trata de hacer en ese momento y en ese lugar. Por lo tanto, el sistema debe ser capaz de recolectar información relevante sobre el usuario, la tarea que está realizando y el contexto dónde se realiza, para poder hacer suposiciones que le permitan simplificar el diálogo con el usuario. Así pues, es necesario disponer de modelos del usuario, la tarea y el contexto.

6.5.1 Modelado de usuario

Para sistemas móviles se pueden usar las técnicas tradicionales de modelado de usuario, teniendo en cuenta que en este caso basta con un modelo simple. En primer lugar se seleccionan los parámetros relevantes que son observables para el sistema y se definen diversos perfiles de usuario en función de los valores posibles de dichos parámetros. Resulta útil disponer de estereotipos de usuarios, a los que se les asignan unos valores concretos de los parámetros. Esto permite asignar un tipo de diálogo a los usuarios de los que no se tiene información completa. También ayudan a resolver contradicciones entre los valores de los parámetros. La comparación de los valores reales de estos parámetros para un determinado usuario con los perfiles definidos, permite al sistema determinar qué tipo de interacción debe llevar adelante.

En el caso de la interfaz de TetraNauta, los parámetros utilizados son

- *Reacción* (velocidad de selección de un ítem): Lenta, Media, Rápida
- *Precisión* (en el manejo del dispositivo apuntador): Baja, Alta
- *Habla*: Normal, Disártrica
- *Tecleo* (nº de teclas que puede manejar): 1, 2, 9
- *Orientación*: se Orienta, se Pierde

Algunos de estos parámetros son permanentes para cada usuario (por ejemplo el tipo de habla), mientras que otros varían con el tiempo. Así, la velocidad de reacción y la precisión mejoran a medio plazo con el entrenamiento pero, a corto plazo, empeoran con la fatiga. Determinados parámetros son directamente observables por el sistema, como por ejemplo la velocidad de reacción y la precisión, mientras que otros se recogen en una fase previa al uso.

De este modo se dispone de 5 parámetros, cada uno de ellos con dos o tres posibles valores, por lo que el número de perfiles de usuario diferentes es de 72. Podemos definir a cada usuario por una tupla:

$$\text{Usuario}_i\ (\text{reacción}_j, \text{precisión}_k, \text{habla}_l, \text{tecleo}_m, \text{orientación}_n)$$

Por ejemplo, (L, B, N, 9, O) sería una persona de reacción lenta, de baja precisión, con voz normal, capaz de usar un teclado de 9 teclas y con capacidad de orientación.

Aunque todas las combinaciones son posibles, algunas son muy infrecuentes. Además suelen existir correlaciones entre parámetros que caracterizan a determinados

grupos de usuarios. De esta manera podríamos hablar de estereotipos de usuarios que comparten un subconjunto de valores determinado. Por ejemplo, (L, B, D, 1, X), serían las personas con grandes dificultades motoras. (R, A, N, 9, O), personas con buen control motor, habla y orientación y (X, X, X, X, O) serían las personas con buena orientación.

Según los valores de los parámetros, a cada usuario se le asigna uno o varios estereotipos y a cada estereotipo le corresponderá un tipo de interfaz.

6.5.2 Modelado de tarea

Se identifica y detalla el conjunto de tareas que pueden ser desarrolladas. El sistema usa esta información para determinar el diálogo de usuario que es posible en cada estado.

En el caso de la silla de ruedas TetraNauta, las tareas previstas son:

- *conducción automática*, que incluye las subtareas: buscar faro; seleccionar destino; ir al destino seleccionado; parar; pasar a guiado manual.
- *conducción manual*, que incluye las subtareas: parar; seguir una ruta mediante el *joystick*; pasar a conducción automática.
- *control de entorno*, que incluye las subtareas: encender y apagar luces; abrir y cerrar puertas, ventanas y persianas; manipular electrodomésticos y aparatos de TV, video, audio; modificar la temperatura.
- *comunicación remota mediante voz*, que incluye las subtareas: establecer la comunicación; mantener una conversación; finalizar.
- *comunicación remota mediante texto* (e-mail, chat), que incluye las subtareas: establecer la comunicación; leer un texto; escribir un texto; enviar un mensaje.
- *acceso a la web*, que incluye las subtareas: abrir el navegador; introducir la URL; leer el contenido de la página; seleccionar un enlace dentro de la página.

La estructura del modelo permite incluir nuevas tareas y subtareas si se detecta su necesidad.

6.5.3 Modelado de contexto

La información sobre el contexto debe ser almacenada en una estructura que permita la identificación de la posición actual y las funciones disponibles en ese punto [29]. El sistema puede comparar esa información con la posición actual para determinar los servicios disponibles al usuario y el modo de interacción que les corresponde.

Así, en la interfaz de la silla de ruedas TetraNauta, el sistema dispone de información de localización absoluta procedente de faros situados en el entorno, de un mapa del hospital (en forma de grafo) donde puede localizar la posición actual de la silla de ruedas y de una estructura de datos en la que constan los servicios accesibles y el modelo de interfaz de usuario adecuado a ese punto. Si, por ejemplo, la silla de ruedas entra en un entorno en el que hay acceso a Internet a través de una conexión inalámbrica de banda ancha, la interfaz de usuario incluye los elementos necesarios para utilizar ese servicio.

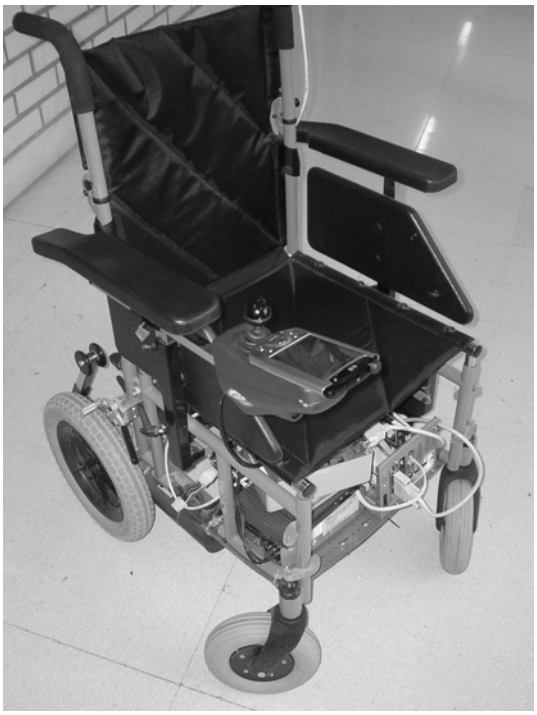

Fig. 2. Prototipo de TertraNauta

6.5.4 Modelo de la interacción

En cada momento el comportamiento del sistema se modela según una tripleta:

$$\text{Interacción}_i\ (\text{usuario}_j, \text{tarea}_k, \text{contexto}_l)$$

De este modo, el sistema decide qué servicios hay disponibles en el contexto_l para que el usuario_j pueda realizar la tarea_k y, lo que es más importante, el tipo de diálogo que establece con ese usuario. Un cambio de contexto hace que el sistema verifique los cambios en los servicios disponibles. Esto puede afectar a la tarea que el usuario esté desarrollando de tres modos: que pueda continuar igualmente, que sea necesario cancelarla o que se pueda realizar a través de otros servicios. Si fuera necesario utilizar otros servicios para continuar con la tarea, los cambios se realizan de manera que el usuario no tenga que hacer nada.

Así, si el usuario está hablando con otra persona a través de una comunicación de banda ancha de radiofrecuencia y sale del área de alcance del enlace de red, el sistema buscará un enlace alternativo que puede utilizar el mismo, u otro soporte, (por ejemplo GMS) para mantener la comunicación. En caso de que no exista ninguna alternativa, se informa al usuario de la situación.

6.6 Pruebas de usabilidad y accesibilidad

El diseño de toda interfaz requiere la evaluación por parte de una muestra representativa de usuarios reales utilizando técnicas rigurosas y fiables. Existen técnicas de evaluación de la usabilidad que pueden ser ampliadas a la detección de problemas de accesibilidad [30, 31]. Estas evaluaciones se realizan en diferentes fases del desarrollo y permiten detectar tempranamente fallos que no podrían ser corregidos una vez que el producto está finalizado. Así, el sistema TetraNauta ha sido evaluado en el Hospital Nacional de Parapléjicos de Toledo. Los resultados de evaluación de los diversos prototipos han servido para eliminar fallos de accesibilidad y realizar mejoras en la usabilidad.

7 Otros aspectos que intervienen en el diseño

La aparición de los sistemas ubicuos permite el desarrollo de aplicaciones con características muy diferentes de las aplicaciones informáticas tradicionales. La computación ubicua aporta, tal como hemos visto, una serie de ventajas indudables pero también trae consigo algunos inconvenientes que el diseñador no puede ignorar.

7.1 Seguridad y protección de la intimidad

El uso de redes inalámbricas hace que los datos que se transmiten puedan ser captados por otros sistemas a los que no van destinados. Aunque esto no parece preocupar a algunos usuarios (por ejemplo los de la telefonía móvil), la computación ubicua debe enfrentar el problema cuanto antes ya que algunas de las aplicaciones ubicuas manejan datos personales, de salud, números de tarjetas de crédito, etc. que deben ser protegidos.

Además, los sistemas de localización permiten hacer un seguimiento detallado de las personas, invadiendo su intimidad. Para evitarlo es necesario garantizar el anonimato en los sistemas de localización [32] y ofrecer la necesaria protección social y legal [33].

Existen grupos de investigadores que ya están trabajando en este tipo de problemas. Para obtener información actualizada, se puede consultar el número monográfico sobre "Security & Privacy" de la revista Pervasive Computing [34].

7.2 Aspectos éticos

El nivel de miniaturización alcanzado por la microelectrónica permite el diseño de chips que pueden ser implantados fácilmente en el cuerpo humano. Existen propuestas para realizar este tipo de implantes para diversos casos, tales como la monitorización de enfermos crónicos, localización de niños pequeños, personas con demencia o en libertad vigilada, etc. Además de la indudable utilidad que estos

sistemas podrían tener es necesario tener en cuenta los aspectos éticos relacionados con la invasión de la vida privada y la grave restricción de libertad que suponen.

El tecnólogo no puede inhibirse de las consecuencias éticas y sociales [35] de sus diseños, dejándolas manos de expertos que, una vez desarrollado el producto, tiene muy pocas posibilidades de actuación [36].

8 Conclusiones

La computación ubicua requiere el diseño de interfaces móviles que usualmente presentan problemas de accesibilidad para las personas con discapacidad. Dado que este colectivo de usuarios es precisamente uno de los que más se puede beneficiar de la tecnología ubicua, es necesario diseñar las interfaces de manera que no incluyan barreras de accesibilidad innecesarias. En este artículo se han revisado algunos de los problemas de accesibilidad que típicamente encuentran las personas con discapacidad y se han expuesto métodos de diseño que permiten crear interfaces accesibles. Estos métodos han sido ilustrados con ejemplos desarrollados dentro del los proyectos TetraNauta, DomoSilla y Heterorred.

Referencias

[1] Weiser M. “The Computer for the 21st Century”. Scientific American (1991). Reprinted in IEEE Pervasive Computing, vol.1, no. 1, pp. 19-25. Jan.-Mar.(2002).

[2] Satyanararayanan M. “A Catalyst for mobile and Ubiquitous Computing”. IEEE Pervasive Computing, vol.1, no. 1, pp. 1-5. (2002).

[3] Hightower J. and Borriello G.. Location systems for ubiquitous computing. IEEE Computer, Vol. 34 No. 8. Pp. 57-66. August (2001). Ampliado en: Hightower J. and Borriello G. A Survey and Taxonomy of Location Sensing Systems for Ubiquitous Computing. UW CSE 01-08-03 August (2001). http://www.cs.washington.edu/homes/jeffro/pubs/hightower2001survey/hightower2001survey.pdf.

[4] Falcó J. Entornos adaptado para ancianos y discapacitados: localización y alarma para personas con demencia. Tesis Doctoral. Depto. de Ingeniería Electrónica y Comunicaciones. Universidad de Zaragoza (1997).

[5] Ekberg J., Routio R. Smart Homes for supporting independent living. The European context for assistive technology. Placencia I., Puig R. (Eds.). IOS Press (1995).

[6] Abascal J., Civit A. Universal Access to Mobile Telephony as a Way to Enhance the Autonomy of Elderly People. WUAUC'01. EC/NSF Workshop on Universal Accessibility of Ubiquitous Computing: Providing for the Elderly. Jorge J., Heller R., Guedj R. (eds.).ACM Press (2001). Pp. 93-99.

[7] "Wearable Computing" IEEE Pervasive Computing. Vol 1, No. 4. Oct.-Dec. (2002).

[8] Rosenfeld R., Olsen D., Rudnicky A. Universal Speech Interfaces. ACM Interactions. Vol 8, No. 6 Nov.-Dec. (2001). Pp. 34-44.

[9] Srinivasan S., E. Brown (eds.). Special Issue on Speech Recognition. Computer. Vol. 35, no. 4, pp. 38-67, April (2002).

[10] Lesher, G.W., Moulton, B.J., & Higginbotham, D.J. Optimal character arrangements for

ambiguous keyboards. IEEE Transactions on Rehabilitation Engineering, vol. 6, no. 4, (1998). Pp. 415-423. [http://www.enkidu.net/downloads/papers/LeMoHi98b.pdf]

[11] Sibert L.E., Jacob R.J.K. Evaluation of Eye Gaze Interaction. Proc. ACM CHI 2000 Human Factors in Computing Systems Conference, pp. 281-288, Addison-Wesley/ACM Press, (2000). [http://www.cs.tufts.edu/~jacob/papers/chi00.sibert.pdf]

[12] Abascal J., Civit A. Opportunities and Risks of the Information and Communication Technologies for Users with Special Needs. 2nd IEEE International Conference on Systems, Man and Cybernetics. Hammamet (Tunisia), (2002). IEEE Cat. No. 02CH37349C.

[13] Abascal J. y Garay N. Capítulo 6. Dispositivos. En J. Lorés (editor) La interacción persona-ordenador. AIPO, Asociación Interacción Persona-Ordenador (2001).

[14] Cañas J. J., Salmerón L., Gámez P. Capítulo 2. El factor humano. En J. Lorés (editor) La interacción persona-ordenador. AIPO, Asociación Interacción Persona-Ordenador (2001).

[15] Cañas J. J., Waerns Y. Ergonomía Cognitiva. Aspectos Psicológicos de la Interacción de las Personas con la Tecnología de la Información. Editorial Médica Panamericana (2001).

[16] Stephanidis C., Savidis A. Universal Access in the Information Society: Methods, Tools, and Interaction Technologies. International Journal of Universal Access in the Information Society. June, vol 1. Pp. 40-55 (2001).

[17] Cook A. M., Hussey S. M. Assistive Technologies: Principles and Practice. Mosby - Year Book, Inc. (1995).

[18] Abascal J., Gardezabal L.. Technology to support alternative and augmentative communication. In A. Casals (Ed.): “Technological Aids for the Disabled". Societat Catalana de Technologia/Institut d'Estudis Catalans. Barcelona (1998). Pp. 233-261.

[19] Abascal J., Civit A., Fellbaum K., Hampicke M. "Telecommunications and Smart Homes. Technology for the future. J. Engelen (ed.) Proceedings of the COST219bis Semminar “Telecommunications: Access for all?” Leuven December 3-4 (2001).

[20] Ducatel K. et all. “Scenarios for ambient intelligence in 2010”. European Commission. Brussels, (2001). ftp://ftp.cordis.lu/pub/ist/docs/istagscenarios2010.pdf

[21] Hansmann U., Merk L., Nicklous M. S., Stober T. Pervasive Computing Handbook. Springer (2001). [3.1 Smart cards, pp. 53-62]

[22] Nicolle C., Abascal J. (eds.) Inclusive Design Guidelines for HCI. Taylor & Francis, London (2001).

[23] Poulson, D., Ashby, M., Richardson, S. USERfit: A practical handbook on user centred design for assistive technology. ECSC-EC-EAEC, Brussels-Luxembourg (1996). http://www.stakes.fi/include/1-4.htm

[24] Abascal J., Arrue M., Garay N., Tomás J. USERfit Tool. A tool to facilitate Design for All. In Carbonell, N., Stephanidis, C., (eds.) Universal Access: Theoretical Perspectives, Practice, and Experience. LNCS Springer, Berlin (2002).

[25] Abascal J., Arrue M., Garay N., Tomás J., Velasco C. Accessibility- and usability-oriented design through USERfit Tool. Upgrade, Vol. IV, no. 1, February (2003). http://www.upgrade-cepis.org/issues/2003/1/up4-1Abascal.pdf publicado junto con: Abascal J., Arrue M., Garay N., Tomás J., Velasco C. USERfit Tool. Una herramienta para el diseño orientado a la accesibilidad y a la usabilidad. NOVATICA: «Interacción Persona-Computador: superando barreras». Nº 161, enero-febrero (2003).

[26] Lee S.-W. (ed.) Advances in Handwriting Recognition, World Scientific Publishing Co., pp. 489-497 (1999).

[27] Malinowski J. U., Schneider-Hufschmidt M., Kühme T. (eds.). Adaptive user interfaces: principles and practice. North Holland (1993).

[28] Abascal J., Cagigas D., Garay N., GardeazabalL. Mobile Interface for a Smart

Wheelchair. In F. Paternò (ed.) Human Computer Interaction with Mobile devices. LNCS 411, Springer, Berlin (2002). Pp 373-377.

[29] Dix A., Rodden T., Davies N., Trevor J. Friday A., Palfreyman K. Exploiting Space and Location as a Design Framework for Interactive Mobile Systems. ACM Transactions on Computer-Human Interaction, Vol. 7, No. 3, September (2000).Pp. 285–321.

[30] Rubin J. Handbook of Usability Testing. Jhon Whiley, New York (1994).

[31] Nielsen J., Mack R. L. Usability Inspection Methods. John Wiley, New York (1994).

[32] Beresford A., Stajano F. "Location Privacy in Pervasive Computing". IEEE Pervasive Computting. Vol. 2, No. 1 (2003). Pp 46-55.

[33] Myles G., Friday A, Davies N. "Preserving Privacy in Environments with Location-Based Applications". IEEE Pervasive Computting. Vol. 2, No. 1 (2003). Pp 56-64.

[34] Davies N. (ed.) "Security & Privacy". IEEE Pervasive Computting. Vol. 2, No. 1 (2003). Pp 20-64.

[35] Jessup L. M., Robey D. The Relevance of Social Issues in Ubiquitous Computing Environments. Communications of the ACM. Dec. (2002). Vol. 45 No. 12, pp. 88-91.

[36] Abascal J. G. "Ethical and social issues of “teleservices” for disabled and elderly people. In J. Berleur and D. Whitehouse (eds). “An Ethical Global Information Society. Culture and democracy revisited” Chapman & Hall (1997).

La Generación de Interfaces Post-WIMP y su Desarrollo en el Laboratorio de Interacción Persona-Ordenador de Albacete

José Pascual Molina Massó y Pascual González López

Grupo de investigación LoUISE – Laboratory of User Interaction and Software Engineering
IIIA – Instituto de Investigación en Informática de Albacete
Universidad de Castilla-La Mancha
Escuela Politécnica Superior, Campus Universitario
Edif. I.D. Juan Manuel, Avda. España s/n
02071 Albacete, España
{jpmolina, pgonzalez}@info-ab.uclm.es

Resumen. Dejando atrás aquellas primeras décadas de la computación en las que el esfuerzo se centraba en obtener ordenadores más y más potentes, la informática actual dirige su atención a la otra parte contratante: el usuario. El protagonismo que ha ganado la Interacción Persona-Ordenador persigue la creación de interfaces más fáciles de manejar y de aprender, en definitiva interfaces de mayor calidad. Pero a pesar del énfasis actual en el diseño centrado en el usuario, el campo de las interfaces se encuentra estancado en el paradigma WIMP surgido al abrigo de la tecnología de los años 60. Calidad e innovación son pues dos retos que asume el grupo de investigación LoUISE, centrándose este artículo en el segundo, en las interfaces post-WIMP y su desarrollo.

1 Introducción

La creciente introducción en nuestras vidas de sistemas software hace cada vez más necesario que, utilizando un lenguaje coloquial, dichos sistemas sean fáciles de usar. Lo que todos los diseñadores de sistemas software pretendemos es que la interfaz sea invisible, es decir, el usuario no sea consciente de que existe un elemento que le facilita o dificulta su manejo. La idea es que el usuario utilice de manera casi intuitiva el sistema software, reduciendo de este modo el tiempo de aprendizaje y obteniendo un alto nivel de efectividad, eficiencia y satisfacción en el uso del citado sistema.

En todo caso, aunque la disponibilidad de "interfaces invisibles" parece hoy por hoy inalcanzable, no es menos cierta la necesidad de disponer de "interfaces de calidad". El objetivo básico que se ha propuesto el Laboratorio de Interacción con el Usuario e Ingeniería del Software (LoUISE) del Instituto de Investigación en Informática de la UCLM, es mejorar la calidad de los sistemas interactivos. Para ello parte de una visión metodológica que permita garantizar el correcto diseño de este tipo de sistemas.

En la figura 1 se muestra la disposición de las diferentes piezas que formas los ámbitos de investigación del grupo LoUISE tendentes a asegurar la calidad de los sistemas software interactivos. Como vemos la parte central de la propuesta es un entorno metodológico creado para guiar el diseñado y desarrollo de este tipo de sistemas interactivos. A esta pieza se unen otros cuatro elementos que permitirán obtener un nuevo entorno de desarrollo más robusto y completo. Como vemos el primero de los elementos se corresponde con la inclusión buenas reglas o prácticas de diseño que recogen la experiencia de los diseñadores. La principal novedad reside en el modo de formular la citada experiencia de los diseñadores, ya que en este caso se utilizan "patrones de interacción" en vez de "guías de estilo", lo cual ofrece un mejor modo de expresar dicha experiencia.

Fig. 1. Ámbitos de investigación del grupo de investigación, conformado alrededor de la metodología de desarrollo IDEAS.

El segundo de los elementos se asocia con la necesidad de medir y evaluar, para de este modo poder determinar el nivel de calidad, medida fundamentalmente en función de la usabilidad, de un diseño y si tras ciertas modificaciones dicha calidad mejora o no. La inclusión de estás métricas dentro del entorno metodológico serán esenciales para que los agentes de interacción puedan alcanzar otro de los objetivos propuestos que es la consecución de interfaces adaptativos. Esta necesidad de adaptatividad es otro de los elementos que nos permitirán hacer más usables a un mayor número de usuarios los sistemas software. Para ello se analizan y se explotan los beneficios que los agentes software tienen para alcanzar mayores cotas de adaptatividad, considerando no sólo adaptación a distintos usuarios, sino a distintos dispositivos o entornos.

Por último estamos convencidos que los actuales sistemas WIMP no solo no son la única alternativa para diseñar la interacción de los sistemas software sino que tampoco son la mejor opción para ello. Por tanto, dentro del grupo LoUISE vemos la necesidad de analizar, desde la misma perspectiva metodológica, nuevos modos de interacción basados principalmente en sistemas tridimensionales, multimodales y con nuevos dispositivos de interacción. Estos sistemas pretenden acercar los distintos

tipos de interacción de los sistemas software al modo en que los seres humanos interaccionamos con otros objetos o utensilios en la vida diaria.

En este artículo vamos a centrarnos básicamente en el estudio de esta nueva generación de sistemas interactivos, ya que el resto de líneas de investigación han sido abordadas en ponencias anteriores dentro de esta Escuela de Verano. Así, el presente artículo comenzará con una introducción a las interfaces post-WIMP en general, para seguidamente centrar la atención sobre las interfaces tridimensionales y su usabilidad. Después pasará a tratar los dispositivos utilizados en las interfaces post-WIMP y el papel que juega en el desarrollo de estas interfaces la ingeniería del software y de la usabilidad. Los temas de investigación de mayor interés para el grupo LoUISE serán resaltados oportunamente en cada uno de los apartados siguientes, intereses que serán recogidos de nuevo en las conclusiones finales de este artículo.

2 Interfaces de usuario post-WIMP

Las interfaces gráficas de usuario (GUI's, *Graphical User Interfaces*) basadas en ventanas, menús, iconos y el ratón (WIMP, *Windows, Icons, Menus and Pointing device*) así como en el "apunte y oprima" (*point and click*), han dominado y dominan la interacción persona-ordenador por más de veinte años. A pesar de los beneficios que han traído consigo estas interfaces WIMP, como por ejemplo servir de estándar "de facto" de las interfaces de aplicación y conseguir con ello una mejor reutilización de la experiencia del usuario, existe sin embargo un sentimiento generalizado de que, a pesar de las mejoras en su aspecto que estas interfaces han incorporado a lo largo de todos estos años, el estancamiento que sufre el campo de las interfaces de usuario no nos permite afrontar los retos que nos plantea el futuro. Las posibilidades que ofrecen los ordenadores hoy en día y las que ofrecerán en un futuro no muy lejano precisan de nuevos planteamientos, de la llegada de una nueva generación de interfaces, una generación a la que Andries van Dam llama "post-WIMP" [27][28].

Estas nuevas interfaces no se basarían únicamente en widgets 2D como menús, formularios o barras de herramientas, sino también, por ejemplo, en el reconocimiento de gestos y del habla para especificar operaciones y sus operandos. Estas interfaces tardarán aún muchos años en ser totalmente desarrolladas, aunque algunos resultados de las investigaciones comienzan ya a salir de los laboratorios, aunque no todos con la difusión deseada. Así, uno de los ejemplos más comunes de interacción post-WIMP disponibles hoy en día en el mercado es el reconocimiento de escritura usado en los ordenadores de bolsillo o PDA'a (*Personal Digital Assistant*), como es el caso del Palm Pilot y los dispositivos Windows CE, los cuales mezclan de manera más o menos exitosa técnicas WIMP y post-WIMP para la realización de tareas 2D. Otro ejemplo bastante claro de interacción persona-ordenador que no hace uso de dispositivos o técnicas WIMP son los juegos que podemos encontrar en los salones recreativos, como los simuladores de conducción con volante de dirección y palanca de cambio de marcha.

En cualquier caso, la interfaz es el medio para realizar un fin, no un fin en sí mismo, y de esta manera la interfaz post-WIMP debería ser transparente al usuario en la realización de sus tareas. De manera ideal, la interfaz debería proporcionar un

medio de adaptación entre las señales de salida del ordenador y los sentidos humanos, al tiempo que traduce de manera eficiente las intenciones del usuario. Es lo que Thomas A. Furness llama en lengua inglesa "matching machines to humans" [10], y cuyo objetivo último es el de crear un amplísimo canal de comunicación entre el ser humano y la máquina.

Sin embargo, lejos aún de la interfaz ideal, hoy por hoy resulta ineludible la presencia de este intermediario, imponiendo una carga cognitiva al usuario. Es por ello que investigadores como van Dam o Furness proponen marcar metas más alcanzables, imaginando el ordenador del futuro como un mayordomo perfecto, quien conoce mi entorno, mis gustos y mi manera de ser, y de forma discreta se adelanta a mis necesidades sin precisar órdenes explícitas. Cuando el usuario se comunica con este mayordomo, lo hace principalmente mediante el habla, gestos, expresiones faciales y otras formas de comunicación humana, como el dibujo de bosquejos. Todo ello sin impedimento de que, si es preciso, el usuario aún pueda seleccionar opciones de un menú o incluso introducir información a través del teclado.

El objetivo de esta visión de la interfaz post-WIMP del futuro no es otro que el de minimizar el trabajo de manipulación y la distancia cognitiva entre las intenciones del usuario y la ejecución de las mismas, de modo que el usuario pueda concentrarse más en la tarea y, con ello, aumentar su productividad. En un primer momento, las interfaces post-WIMP proporcionarán al menos una técnica de interacción que no dependa de los clásicos widgets 2D como menús o iconos. Al final estas interfaces involucrarán todos nuestros sentidos en paralelo en un lenguaje de comunicación natural donde participarán múltiples usuarios. La interacción en esta nueva generación de interfaces post-WIMP será, entonces, multimodal y multiusuario.

La interacción multimodal, demostrada en un primer momento por los investigadores del MIT con su demo "Put That There" (Pon eso ahí) [3] y objeto de mucha reciente investigación, permite que varios canales paralelos proporcionen información correlacionada, de forma que los algoritmos de reconocimiento puedan reforzarse mutuamente. Las interfaces WIMP actuales no están preparadas para acomodar esta nueva y cada vez más demandada forma de interacción.

La interacción multiusuario está produciendo en la actualidad un cambio en las preocupaciones de los investigadores en interfaces de usuario. Si antes las investigaciones se centraban en usuario individuales realizando tareas en una o varias aplicaciones, el interés se dirige ahora hacia múltiples usuarios interactuando simultáneamente empleando diferentes dispositivos informáticos en diferentes entornos. Resulta interesante apuntar que muchos de esos múltiples usuarios serán reales, pero otros podrán ser "virtuales", apreciándose la importancia que tendrá la tecnología de agentes en las interfaces del futuro.

En este nuevo escenario, veremos múltiples participantes interactuando desde lugares dispares en una tarea compartida, con equipos de visualización caracterizados por ofrecer un amplio campo de visión y mostrar imágenes estéreo de acuerdo a la posición del usuario, proporcionándole sensación de inmersión. El sistema monitoriza las manos del usuario, reconoce su voz y sus gestos y facilita además la manipulación por medio de dispositivos de interacción con más de los dos grados de libertad que proporciona un ratón convencional.

Las piezas que componen este escenario futurista están siendo investigadas por muchos grupos de investigación en interacción persona-ordenador alrededor del

mundo, piezas que incluyen a las interfaces 3D, el reconocimiento del habla y de gestos, interfaces táctiles y un amplio abanico de entornos virtuales basados en su mayoría en la tecnología de los cañones proyectores, desde pantallas semi-inmersivas como la ImmersaDesk hasta cubículos totalmente inmersivos como el sistema CAVE, sin olvidar entornos que tratan de aprovechar todas las superficies de una habitación como la "oficina del futuro" imaginada por Henry Fuchs [20].

Tantas y variadas piezas nos permite hacernos una idea del enorme trabajo de investigación que es necesario efectuar en todos los frentes para lograr la construcción de las interfaces post-WIMP del futuro. Conscientes entonces de la labor que supone, desde el laboratorio de interacción persona-ordenador de Albacete pretendemos prestar una mayor atención a un reducido número de ellas, entre las que se encuentran las tecnologías de agentes y las interfaces 3D, siendo esta última la que centrará en gran medida el resto del presente texto.

3 Interfaces 3D

Loa gráficos 3D por ordenador y las interfaces de usuario tridimensionales han sido objeto de estudio durante décadas. Hasta hace no muchos años, la tecnología de síntesis de gráficos 3D era considerada como campo exclusivo de las carísimas estaciones de trabajo que ofrecían el poder de cómputo y la memoria necesaria para tal labor. Sin embargo, los avances que la tecnología nos ha brindado desde entonces ha conseguido que el hardware especializado para la aceleración de gráficos 3D sea cada vez más rápido y, sobre todo, más barato [14]. Tan barato que hoy en día ha puesto al alcance de cualquier consumidor el disponer de un subsistema gráfico que supera con creces las prestaciones de las antiguas estaciones gráficas. De hecho, resulta difícil encontrar un PC nuevo que no incluya una tarjeta de video con aceleración de gráficos 3D. No cabe duda que, aparte de las exclusivas aplicaciones de diseño e ingeniería, este arrollador avance de los gráficos 3D en el mundo PC es debido en gran medida a la enorme popularidad de los videojuegos 3D. Este hecho no debe extrañar, habida cuenta del gran mercado que supone el entretenimiento, superando las ganancias por videojuegos incluso al tradicional líder de este campo, el cine.

El éxito de estos videojuegos tridimensionales nos hace preguntarnos si no sería posible que la interacción con el ordenador fuera tan fascinante como jugar a Wolfenstein 3D, Doom o Quake [13], por poner algunos ejemplos, sin olvidar la importancia del primer videojuego 3D, "The Colony", lanzado en 1987 y precursor de todos los demás juegos de pistoleros en primera persona (*first person shooters*) pero que no pudo alcanzar igual popularidad por estar sólo disponible para ordenadores Macintosh.

Con el empleo de interfaces tridimensionales se busca entonces ofrecer un estilo de interacción que sea más natural e intuitivo para el usuario, quien al fin y al cabo es una persona que vive en un mundo tridimensional, y utiliza sus especializados sentidos para desenvolverse en ese medio. Es por ello que, durante años, muchos entusiastas de los gráficos 3D por ordenador pensaban que este tipo de interfaces de usuario no tardarían mucho tiempo en revolucionar la forma en que trabajamos con

los ordenadores. De hecho, los primeros intentos por desarrollar interfaces de usuario 3D iban dirigidos hacia una transformación radical de la metáfora de escritorio que puede encontrarse en prácticamente todos los ordenadores o la telaraña mundial que ha logrado conectar todos esos ordenadores a través de un espacio hipermedial. Sin embargo, esos años han pasado y nos han enseñado que el éxito que tienen los gráficos 3D en los videojuegos no puede extrapolarse sin más al resto de aplicaciones de ordenador. Uno de los errores que frecuentemente se comenten es la mera adaptación al mundo 3D de las soluciones de interacción empleadas en las interfaces WIMP, lo cual no suele proporcionar un resultado satisfactorio. En su lugar, deben plantearse nuevas metáforas que permitan aprovechar las singulares ventajas que ofrecen las interfaces de usuario tridimensionales.

Pero antes de abordar ese tema, realizaremos un breve recorrido en el tiempo para mostrar los esfuerzos que se han realizado a lo largo de los últimos años para llevar la tercera dimensión al mundo de la informática personal, desde aquellas interfaces que buscaban explotar ese tercer eje del espacio para mejorar la visualización de la información, como es el caso de 3D Rooms, hasta propuestas tan recientes como es el sistema operativo Croquet, pasando por soluciones que tratan de establecer un puente desde los actuales sistemas de ventanas 2D hacia entornos 3D más ricos, como Win3D o 3DNA.

Empezaremos entonces hablando de 3D Rooms, una interfaz de usuario 3D presentada por los investigadores de Xerox a principios de los 90. En 1993, la revista "Communications of the ACM" publicaba un artículo [21] escrito, entre otros, por George Robertson, quien por entonces desarrollaba su trabajo en la cuna de las interfaces WIMP, los laboratorios Xerox PARC. Este artículo describía el sistema "Information Visualizer", un sistema que exploraba un paradigma de interfaz de usuario que iba más allá de la metáfora del escritorio aprovechando el impulso de la tecnología de gráficos para dar solución a la creciente demanda en almacenamiento, recuperación, manipulación y comprensión de grandes cantidades de información. Este sistema se presentaba como una evolución del anterior sistema Rooms hacia la tercera dimensión, incluyendo un buen número de visualizaciones 3D, entre las cuales podemos citar el árbol de conos (*cone tree*) para representar la estructura jerárquica de directorios y ficheros, o paredes en perspectiva (*perspective wall*) para acomodar información que se extiende mucho más en el eje horizontal que en el vertical. El objetivo de todas esas visualizaciones no era otro que el de extender el escritorio WIMP haciendo uso de los gráficos 3D, así como maximizar el uso efectivo del escaso espacio del que disponen la pantallas que suelen acompañar a PC's y estaciones de trabajo.

En este punto es interesante comentar dos trabajos relacionados con el sistema propuesto por los investigadores de Xerox. Uno es anterior en el tiempo, y se trata del sistema FSN (*File System Navigator*) de Silicon Graphics [24], el cual presenta el sistema de ficheros como un paisaje urbano (*Information Landscape*), representando con cajas los ficheros, cajas con un tamaño proporcional al del fichero y cuyo color representa la antigüedad del mismo. Muchos recordarán este sistema por aquella escena de la película "Parque Jurásico" en la que la muchacha trata de cerrar las puertas de seguridad del recinto a través de un ordenador y exclama "¡Hey, esto es UNIX, lo conozco!". Otro trabajo relacionado, en este caso algo posterior, es el de

Iván Fernández, quien recogía muchas de estas ideas en la creación de su sistema "Information Walktrough" [9].

Un artículo contemporáneo del publicado por G. Robertson *et al.* es el que escribió Jim Leftwich acerca del sistema InfoSpace y su evolución, InterSpace [15]. Leftwich pretendía con este sistema aprovechar el hardware que equipaba muchas estaciones gráficas y que permitía la visualización de mundos virtuales a través de imágenes estéreo. Basándose en estas capacidades, Leftwich imaginaba un nuevo nivel en la interacción con la información, estableciendo una relación en el espacio tridimensional entre el usuario y la información con la que está interactuando, con el fin de explotar la memoria espacial del usuario y otros recursos del ser humano para absorber, procesar y sintetizar la información. Así, encontraremos pantallas bidimensionales (*Screens*) que rodean al usuario orientadas siempre hacia la posición de éste, y que pueden contener información tanto 2D como 3D. En este último caso, el usuario podría sustituir el actual espacio de interacción y sus objetos por el contenido de ese otro subespacio. InfoSpace introduce conceptos interesantes, como el de hiper-conector, que permite establecer relaciones trans-espaciales entre elementos, o el de maleta (*briefcase*), que permite al usuario llevar consigo elementos de un espacio a otro. Con InterSpace, Leftwich realiza una segunda iteración sobre los conceptos de InfoSpace, evolucionando y expandiendo sus conceptos, como por ejemplo su organización en cuanto a espacios y objetos, según la cual el interior de cualquier objeto es un espacio y, recíprocamente, cualquier espacio es representado externamente como un objeto.

Con todas estas ideas en la mente de muchos entusiastas de los gráficos 3D por ordenador, no es extraño que cuando apenas se había comenzado a tejer la teleraña mundial ya hubiera quien imaginara la posibilidad de navegar e interactuar con mundos tridimensionales en la Web. Así, Mark Pesce y Tony Parisi fueron invitados por Tim Barnes-Lee para presentar Laberynth en la primera conferencia internacional del W3C. Laberynth fue el precursor del lenguaje VRML, en cuya definición se volcaron desde entonces un extraordinario número de personas, para finalmente convertirlo en 1997 en el primer estándar para la descripción de mundos virtuales en la Web. La utilización de este lenguaje es cada vez más madura, con un abanico de aplicaciones que crece cada día más, incluyendo simulación para el entrenamiento de médicos o bomberos, visitas virtuales a ciudades turísticas o exposiciones virtuales de productos en portales de comercio electrónico, entre otras. En la actualidad, aquella especificación aceptada por la ISO sigue siendo el único estándar oficial, si bien el consorcio Web3D está cerca de conseguir la estandarización de la siguiente generación de ese lenguaje, bajo el nombre de X3D.

Sin embargo, durante estos últimos años son muchos quienes han tachado a VRML de ser un lenguaje muerto en comparación con otras tecnologías surgidas al abrigo de Internet. Y en referencia a las interfaces gráficas 3D antes mencionadas, no han logrado el éxito que sí acompaña en cambio a los videojuegos 3D. Tanto la interfaz de nuestro sistema operativo favorito como el ciberespacio que representa la World Wide Web sigue siendo en su inmensa mayoría bidimensional. La popularización de hardware especializado en aceleración de gráficos 3D no se ha traducido por ahora en una mayor aceptación de otros usos de la tecnología 3D, lo que ha llevado a que las propuestas actuales en el campo de las interfaces 3D no traten de revolucionar la

interacción persona-ordenador con alternativas radicalmente diferentes a las actuales interfaces WIMP. Como apuntaba Leftwich en su artículo de 1993, resulta crucial crear un camino o puente para que los usuarios familiarizados con las actuales interfaces bidimensionales puedan reutilizar ese conocimiento adquirido con las nuevas interfaces que han de llegar.

Por esa razón, el objetivo que persiguen las nuevas propuestas es el de mejorar los actuales entornos de ventanas con la adición de la tercera dimensión. Un buen ejemplo de esta tendencia es el conjunto de directrices IBM RealPlaces [12], las cuales permiten describir un entorno tridimensional en el que la metáfora es el empleo de los objetos del día a día. El entorno está formado por lugares (*Places*, en la lengua de Shakespeare) que contienen grupos de objetos que permiten la realización de una o varias tareas. Esta organización recuerda las relaciones entre espacios y objetos que Leftwich establecía en InfoSpace e InterSpace. También existen otras similitudes, como el empleo de un espacio en pantalla (*scrim*) para el transporte de elementos de un lugar a otro, labor desempeñada por la maleta en el caso de InfoSpace. Sin embargo, una de las características destacadas de RealPlaces es la navegación. Para desplazarse de un lugar a otro, el usuario puede mover el ratón, utilizar las teclas de cursor situadas en el teclado o hacer uso de unos controles sobreimpresionados en pantalla. Más interesante aún, el usuario puede desplazarse simplemente haciendo un único clic de ratón sobre el objeto destino, y el sistema se encarga entonces de mover el punto de vista del usuario en el entorno hasta situarlo a una distancia adecuada del objeto. Con ello, se pretende simplificar la navegación siguiendo además la idea de que el sencillo clic de ratón debería dominar la interacción con el ordenador, en contraposición con el usual doble clic, como bien afirma Tony Temple. Aunque este conjunto de directrices fue escogido con la idea de crear un entorno que pudiera sustituir el escritorio de los actuales entornos de ventanas, también se facilita que las aplicaciones existentes para esos entornos puedan ejecutarse igualmente junto a las nuevas.

Otro ejemplo es "The Task Gallery" [22], un entorno tridimensional basado en la metáfora de la galería de arte y desarrollado por equipo liderado por G. Robertson, en esta ocasión trabajando como parte del grupo de investigación de usuario de Microsoft, lo que demuestra el interés que también muestran las grandes compañías en el desarrollo de esta clase de interfaces. En esta galería de tareas, el usuario puede ejecutar cualquiera de las actuales aplicaciones Windows y colocar las ventanas de aplicación en las paredes, el suelo o incluso en el techo. Su diseño parte de la premisa de que los entornos virtuales 3D pueden explotar mejor nuestra percepción y conocimiento del espacio, incrementando así la productividad, premisa similar a la tomada por Leftwich para la concepción del sistema InfoSpace.

En relación a los escritorios 3D, podemos citar también en nuestro recorrido a los productos comerciales Win3D y 3DNA (figura 2). El primero es un front-end para Windows creado por la compañía ClockWise [6], en cuyo desarrollo han contado con el consejo y la ayuda del profesor Ben Shneiderman, y donde podemos encontrar varias habitaciones 3D, cada una con una apariencia particular y un propósito específico: aplicaciones de oficina, Internet, juegos y multimedia. El usuario puede navegar por el entorno usando las teclas del cursor o haciendo clic sobre la base del lugar hacia el que quiere desplazarse –imitando de esta manera las directrices IBM RealPlaces–, pero también puede saltar rápidamente a otra habitación con un simple

clic de ratón sobre el mapa sobreimpreso en la parte inferior de la interfaz. Por su parte, el escritorio 3DNA Desktop es comercializado por la compañía 3DNA Corp. [1] y está dirigido de manera especial hacia una audiencia concreta, el de los aficionados a los videojuegos 3D, tratando para ello de transformar el escritorio 2D en un entorno 3D más atractivo donde el usuario pueda navegar usando el teclado de la misma forma que lo hace en cualquier videojuego 3D. Sin embargo, este producto no ofrece otras alternativas para la navegación, como sí lo hace el anterior.

Fig. 2. Front-ends para Windows, Win3D a la izquierda y 3DNA Desktop a la derecha.

Por último, el ejemplo más significativo de hasta qué punto existe un enorme interés por el desarrollo de interfaces de usuario 3D lo encontramos en Croquet [18]. No se trata de un simple front-end que corre sobre un sistema operativo como pueda ser Windows. Se trata de un sistema operativo concebido como un verdadero entorno tridimensional colaborativo, y en cuyo desarrollo participa Alan Kay, uno de los fundadores del centro de investigación Xerox PARC y que en aquella época lideró el desarrollo del actual paradigma WIMP. Dejando a un lado los pormenores de Croquet como sistema operativo, el objeto de su desarrollo no es otro que el de aprovechar la tecnología existente para ir lo más lejos posible. En este entorno encontramos de nuevo ventanas, pero sólo como un tipo más de los diferentes objetos con los que podemos interaccionar. El usuario puede trasladar, redimensionar y rotar las ventanas, y cuando están muy lejos de las posición del usuario, éste solo tiene que hacer clic sobre ellas para que el sistema lo lleve hasta una distancia que permita su manipulación, lo cual recuerda también a otros entornos que ya hemos citado aquí. Pero lo interesante es que esas ventanas también pueden ser portales a otros mundos, a través de los cuales el usuario puede asomarse y pasar de un mundo a otro simplemente avanzando un poco más. Se dibuja así un verdadero ciberespacio tridimensional, en el que otros usuarios aparecen representados por sus correspondientes avatares, y cualquiera de ellos puede interactuar con cualquier objeto de cualquiera de esos mundos, constituyendo así un entorno verdaderamente colaborativo.

Con todo, aprender de experiencias anteriores y extraer las características que hacen de las interfaces 3D algo útil para el usuario resulta imperativo si se pretende que esta clase de interfaces sean las preferidas por los usuarios en los próximos años, sucediendo a las actuales interfaces 2D de la misma forma que las interfaces gráficas

WIMP sucedieron al estilo de interacción basado en línea de comandos. A pesar de los esfuerzos realizados a lo largo de todos estos años, uno de las principales obstáculos que impiden una mayor difusión de las interfaces 3D es su dificultad de uso. Este problema de usabilidad es uno de los temas que, dentro de este campo de las interfaces tridimensionales, se está estudiando en el laboratorio de interacción persona-ordenador de Albacete. En el siguiente punto abordaremos este tema en mayor profundidad.

4 Usabilidad de la interfaz 3D

La mayoría de las críticas que se han vertido estos últimos años sobre la tecnología de gráficos 3D tienen que ver con la dificultad para crear contenidos e interactuar con ellos. En el caso de entornos 3D interactivos para la Web, podemos añadir un tercer argumento en su contra, relacionado éste con la espera que tiene que soportar el usuario para descargar los modelos 3D y los recursos asociados, así como la potencia que se exige a la máquina cliente para poder representar el entorno descargado con la suficiente suavidad. Este tercer argumento que se esgrime en contra de la tecnología 3D es en realidad un problema común a muchas otras tecnologías de Internet, y depende del ancho de banda y la potencia de los ordenadores clientes, cuestiones ambas tecnológicas que comienzan a superarse gracias a la proliferación de accesos cada vez más rápidos a Internet y de hardware especializado para la aceleración de gráficos 3D, por lo que es de esperar que desaparezcan en el futuro. En cambio, la dificultad de creación y de uso de las interfaces 3D precisa de una más detallada discusión.

La interfaces 3D suelen valorarse en la mayoría de la ocasiones comparándolas con las interfaces 2D. Jacob Nielsen argumentaba recientemente que el 2D es mejor que el 3D simplemente porque no somos ranas y nuestros ojos miran hacia al frente [17]. Esta sencilla afirmación parece olvidar la gran ventaja que nos proporciona tener una visión binocular, el sentido de la profundidad, que nos permite desenvolvernos con facilidad en el mundo tridimensional en el que vivimos. Pero aunque dejemos a un lado argumentos como el anterior, fáciles de rebatir, no podemos ignorar que existe una opinión bastante extendida de que las interfaces 2D son mejores que las interfaces 3D, argumentando esta vez que el 2D es preferible por tener una dimensión menos, mientras que en 3D la interacción se complica de manera innecesaria.

Un ejemplo podría ser la visualización de la estructura de directorios y ficheros. Existen visualizaciones 3D alternativas al clásico árbol 2D, como pueden ser el paisaje urbano utilizado en el sistema FSN o el árbol de conos usado en el sistema "Information Visualizer". Sin embargo, y aunque ya han pasado años desde la propuesta de esas alternativas en 3D, seguimos utilizando el clásico árbol 2D. Pero ello no nos debe hacer caer en la tentación de generalizar. Tan cierto es que existen muchas tareas cuya realización es más sencilla en 2D como cierto es que existen otras muchas que podrían mejorarse con la adecuada interfaz 3D.

La usabilidad, al fin y al cabo, es medida en muchas ocasiones como la rapidez con la que puede realizarse una tarea cometiendo el menor número de fallos, trabajando de forma continuada durante un cierto periodo de tiempo. El reto es entonces la

creación de mejores interfaces 3D, no la simple adaptación de técnicas 2D a una tercera dimensión, tal y como apuntamos anteriormente. El reto es reducir la carga cognitiva del usuario añadiendo una nueva dimensión, del mismo modo que esa carga cognitiva se redujo al pasar del estilo de interacción de línea de comandos –que podríamos calificar como de una única dimensión– a un paradigma de dos dimensiones como son las interfaces WIMP. Por ejemplo, el trazo sobre una superficie plana de la proyección de un objeto no es tarea sencilla, pues obliga al dibujante a pensar acerca de la perspectiva, la dirección de la luz y las sombras, mientras que dada una herramienta de modelado con la adecuada interfaz de usuario, esa tarea puede simplificarse en gran medida.

Por otro lado, a Jacob Nielsen no le falta razón al afirmar que resulta difícil interactuar con modelos tridimensionales con los dispositivos y las técnicas de interacción que se usan actualmente, pues se trata de dispositivos 2D que han sido diseñados para la manipulación de interfaces 2D. Los dos grados de libertad con los que cuenta un dispositivo tan popular como es el ratón limita por un lado nuestra forma de pensar con respecto a la interacción persona-ordenador, confinándola al 2D, obligando por otro lado a plantear las soluciones más diversas para conseguir abarcar los seis grados de libertad del espacio tridimensional, pues no es posible establecer un mapeo directo.

Un ejemplo muy ilustrativo es la navegación e interacción con contenido 3D en la Web. Una de las soluciones podría consistir en utilizar uno de los botones del ratón para conmutar el modo en que se mapean los dos grados de libertad del ratón sobre dos de los seis grados de libertad del modelo en pantalla. Otra opción es recurrir al teclado como ocurre en la mayoría de los videojuegos 3D, complementando las teclas del cursor con otro conjunto de teclas o el ratón para abarcar todos los posibles movimientos del personaje. El empleo del teclado ha sido también adoptado por los browsers VRML, así como por los escritorios 3D citados anteriormente. Sin embargo, la solución más ampliamente extendida en la mayoría de las aplicaciones de modelado y representación en tres dimensiones consiste en proporcionar un conjunto de widgets 2D que rodean la escena 3D (*dashboard*), lo que van Dam denomina el "modelo de televisión". Estos elementos 2D permiten seleccionar entre varios modos de mapeado del movimiento del ratón –como por ejemplo andar, volar o examinar– o actuar directamente sobre el modelo 3D –como por ejemplo acercándolo o alejándolo–. Esta solución también ha sido adoptada por todos los browsers VRML.

En cualquier caso, el empleo del ratón, el teclado y los widgets 2D se hace complejo para cualquiera que no sea un aficionado a los videojuegos 3D o un entusiasta de los gráficos tridimensionales. Esta complejidad se ve amplificada por la diversidad de interfaces que ofrecen los distintos browsers VRML, haciendo más difícil al usuario trasladar su experiencia de uso de un browser a otro. Más aún, muchas compañías que trabajan en el desarrollo de contenido 3D para la Web optan por ocultar la barra de herramientas del browser VRML, sustituyéndola por otro conjunto de widgets 2D que se espera se ajusten mejor al objetivo del producto.

En cambio, el ratón convencional resulta perfecto para navegar por el contenido 2D que reina en la telaraña mundial, basándose la interacción en el simple "apunte y oprima" para ir de una página a otra, y los usuarios reconocen fácilmente en cada browser los botones básicos, como por ejemplo el botón de "vuelta atrás". Sin

embargo, un diseño más acertado de los entornos 3D puede ayudar a eliminar esos problemas de usabilidad que le separa de la interacción 2D en la Web. Por ejemplo, la navegación de un lugar a otro en el espacio 3D por medio del simple clic del ratón que puede encontrarse en las directrices RealPlaces, el front-end Win3D o el sistema operativo Croquet, puede equipararse con la navegación de página a página basada en hiperenlaces. Por otro lado, el botón de "vuelta atrás", tan útil cuando se navega por la red podría tener su contrapartida en el espacio 3D a través de un botón en el browser VRML que deshaga el último movimiento efectuado por el usuario; esta funcionalidad ya está presente en el browser X3D desarrollado por Media Machines, llamado Flux [16].

Con todo, la creación interfaces tridimensionales para los ordenadores actuales no deja de ser un completo desafío debido a que la interfaz hardware con la que cuentan estos ordenadores fue diseñada en su momento para aplicaciones bidimensionales como procesadores de texto u hojas de cálculo, utilizando widgets 2D usualmente representados en pantallas con proporción 4:3, similar a las televisiones de entonces. Pero con una pantalla así y un ratón, la interfaz hardware de estos ordenadores resulta perfecta para seres con un único ojo y una mano con un solo dedo, como apuntaba Bill Buxton, científico jefe de Silicon Graphics. Y, aunque en estos últimos años la tecnología ha evolucionado lo suficiente como para ofrecernos pantallas de gran profundidad de color, mejorando así el aspecto visual de la interfaz WIMP, y el mercado de consumo se ha poblado con pantallas con mayores resoluciones y un tamaño de 17", 19" e incluso 21", solucionando en parte el problema de falta de espacio en pantalla que tienen los entornos de escritorio, ninguno de estos avances conseguiría que Buxton cambiara de opinión.

Fig. 3. Algunos ejemplos de pantallas mejor adaptadas al sistema visual humano. A la izquierda la pantalla Panoram formada por tres paneles LCD, en el centro la pantalla autoestereoscópica SynthaGram, y a la derecha el monitor 3D Perspecta.

En cambio, las interfaces tridimensionales tratan de acercarse más al ser humano, de proporcionarle una interacción más natural en un medio en el que está acostumbrado a desenvolverse, el espacio 3D. Pero para lograrlo también es necesaria una interfaz hardware mucho más acorde con este objetivo, mucho más adaptada al ser humano. Por ejemplo, si nos paramos a pensar por un momento que el campo de visión que nos proporciona nuestro sistema visual abarca 180º en horizontal y 120º en vertical, parece claro que las dimensiones y proporciones de las actuales pantallas no son las más adecuadas, y que sería mucho más apropiado disponer de pantallas de mayor tamaño pero con visión panorámica, como por ejemplo las pantallas de plasma con 32" o 42" y razón de 16:9 que comienzan a tener cada vez más éxito en el

mercado de consumo, o las pantallas multipanel como las que comercializa la compañía Panoram [19], con un panel de cristal líquido central a cuyos lados izquierdo y derecho se disponen dos paneles más ligeramente girados hacia el usuario. Si pensamos además que nuestro sistema de visión es binocular, podríamos pensar en sacarle mucho más partido utilizando una representación estéreo de las imágenes, pero en lugar de utilizar las usuales gafas activas o pasivas, podríamos pensar en usar algún tipo de pantalla autoestereoscópica como las que comercializa StereoGraphics [25], con tecnología multiplanar y lenticular, o un monitor 3D como el Perspecta [2], capaz de representar volúmenes en el espacio a base de puntos (figura 3).

En general, si pretendemos explotar las ventajas de las interfaces 3D liberándonos de los dispositivos hardware que emplean las interfaces WIMP, resulta inevitable echar un vistazo a las tecnologías utilizadas en los nuevos paradigmas, como por ejemplo la realidad virtual, la realidad aumentada o mixta y la computación ubicua. En el caso de la realidad virtual, la idea que suele venir a la mente enseguida es la de un usuario con un casco en la cabeza y unos guantes de datos en las manos. Sin embargo, las tecnologías que abarca la realidad virtual van mucho más allá de esa imagen, combinando también interfaces táctiles y de retorno de fuerzas, gracias al empleo de tactores y complejos mecánicos, así como sonido que puede ser localizado en el espacio tridimensional gracias al uso de la función de transferencia asociada a la cabeza o HRTF. Este conjunto de tecnologías que reúne la realidad virtual está siendo estudiado y probado en el laboratorio de interacción persona-ordenador de Albacete, y de ellas nos ocupamos en el siguiente apartado.

5 Dispositivos para interfaces post-WIMP

En los anteriores apartados se ha expuesto la necesidad de huir de las actuales interfaces WIMP haciendo evolucionar el campo de la interacción entre personas y máquinas hacia interfaces más naturales, mejor adaptadas al ser humano, interfaces que al mismo tiempo deberán permitirnos exprimir la tecnología a nuestro alcance para poder hacer frente a los retos que nos plantea el futuro. Pero de poco sirve plantear nuevas interfaces, como por ejemplo las interfaces tridimensionales, si la interacción con el ordenador sigue desarrollándose bajo la tiranía del teclado y del ratón. Esta situación nos obliga a explorar nuevas tecnologías, como por ejemplo las pantallas de plasma con dimensiones panorámicas o las pantallas de tres paneles, con las que buscamos una mejor adaptación de la interfaz visual al sentido de la vista humano. Aunque el coste actual de estos dispositivos es aún excesivo para el consumidor medio, la creciente demanda y el aumento de producción están reduciendo sensiblemente el valor final de estos productos. Sin embargo, también podemos recurrir a dispositivos más económicos para construir interfaces post-WIMP, como por ejemplo cámaras y micrófonos para, a través del procesamiento de la imagen y de la voz, reconocer gestos faciales o el movimiento de las manos del usuario, así como los comandos vocales.

Como se ha mencionado en el apartado precedente, las tecnologías asociadas a los nuevos paradigmas, como la realidad virtual, la realidad aumentada o mixta y la computación ubicua, representan una fuente muy rica de nuevas técnicas de

interacción basadas en dispositivos que poco o nada tienen que ver con el ratón o el teclado a los que estamos acostumbrados. En este punto conviene establecer una clara separación entre los dispositivos y las técnicas de interacción empleadas en estas nuevas interfaces de usuario, y es que los dispositivos de entrada y salida que nos encontraremos son solamente las herramientas físicas usadas para la implementación de diferentes técnicas de interacción. En general, es posible llevar diferentes técnicas de interacción sobre un mismo dispositivo. Lo que realmente importa es si realmente conviene usar un determinado dispositivo para una cierta técnica de interacción, evaluando tanto la naturalidad de esa correspondencia como su empleo eficiente por parte del usuario.

En particular, en el laboratorio de interfaces de usuario de Albacete estamos muy interesados en las tecnologías asociadas a la realidad virtual. Un ejemplo de dispositivo con el que contamos en nuestro laboratorio y que es muy utilizado en entornos de realidad virtual son los guantes Pinch Gloves (figura 4), fabricados por la compañía FakeSpace [7].

Fig. 4. Los guantes Pinch Gloves (a la izquierda) no requieren calibración y el reconocimiento de gestos es más sencillo que en el caso de guantes de datos que, como el CyberGlove (derecha) miden la flexión y abducción de los dedos.

Se trata de unos guantes que, a diferencia de los demás modelos de guantes utilizados en realidad virtual, no sirven para medir la flexión o abducción de los dedos. En su lugar, estos guantes detectan el contacto entre dos o más dedos (*pinch gestures*), gracias a la tela conductora que se encuentra en las yemas de los dedos del guante. Esta singularidad los convierte en unos dispositivos de control bastante útiles en entornos inmersivos en los que el usuario no puede bajar la vista hacia el teclado pues se lo dificulta el visiocasco o las gafas estereoscópicas que lleva puestas. Su mayor ventaja es que no precisan proceso alguno de recalibración, tan sólo establecer una correspondencia entre los datos discretos que envía el guante y las acciones en el mundo virtual. Usados normalmente en combinación con un sistema de localización electromagnético, son muchas las técnicas de interacción que se han desarrollado para estos guantes [5], que incluyen la navegación basada en las dos manos, la selección de

objetos, el envío de órdenes de manera directa o a través de menús y la introducción de texto, entre otras aplicaciones. Sin embargo, su principal desventaja es que existen pocos gestos que puedan resultar naturales o intuitivos al usuario, como por ejemplo juntar las yemas del dedo índice y el pulgar para expresar conformidad, por lo que los gestos deben ser aprendidos por el usuario.

6 La ingeniería y el desarrollo de interfaces post-WIMP

Como se ha expuesto en el apartado anterior, los diseñadores de interfaces post-WIMP deben enfrentarse a una gran variedad de dispositivos de entrada y de salida y a una tecnología que puede experimentar cambios muy rápidamente, con nuevas pantallas, sensores y dispositivos de control que requieren nuevas y apropiadas técnicas de interacción y una consecuente reevaluación del conocimiento y experiencia ganado hasta el momento. Se está aún lejos, por tanto, de alcanzar el mismo estado de madurez que presentan actualmente las interfaces WIMP.

Ben Shneiderman [23] identifica tres pilares para un diseño exitoso de la interfaz de usuario: la documentación acerca de las guías de diseño actuales, el empleo de herramientas software para la creación de interfaces de usuario, y una revisión por parte de expertos junto con la realización de test de usabilidad. Estos tres pilares nos permiten determinar las fuentes que debemos utilizar en el diseño de la interfaz de usuario. Un ejemplo son las guías de diseño de interfaz para Macintosh y la API o interfaz de programación que forma parte del sistema operativo Macintosh, que en conjunto nos informan sobre el qué y el cómo de los elementos más básicos de la interfaz de usuario basada en la metáfora de escritorio, definen su funcionalidad, propósito y apariencia visual. Guías de diseño y herramientas como las de este ejemplo proporcionan a los diseñadores no sólo los bloques básicos de construcción de la interfaz, liberándoles de la necesidad de inventarlos e implementarlos ellos mismos, sino también una clara idea acerca de cómo esos bloques encajan unos con otros y pueden ser utilizados para crear la interfaz de un productor particular.

Sin embargo, frente a la gran experiencia acumulada en el diseño de interfaces WIMP, las interfaces post-WIMP no suelen basarse en ningún tipo de guía de diseño, sino que su creación suele llevarse a cabo más bien por intuición, involucrando a los desarrolladores en ciclos de creación de prototipos y de evaluación. La razón es que estas interfaces post-WIMP, como por ejemplo las interfaces tridimensionales, suelen ser mucho más ricas en interacción, y conllevan el desarrollo de un considerable número de elementos que deben ser creados por medios bien distintos.

El desarrollo de estos sistemas interactivos suele interpretarse desde dos enfoques básicos [4][8]. El primero está más relacionado con el usuario y la interacción con el ordenador, y comprende el desarrollo de todos los medios, incluyendo texto, gráficos, audio y video, así como la integración de los dispositivos que el usuario usará para comunicarse con el ordenador. Este primer enfoque representa en mayor medida la visión del artista y el diseño de las respuestas del sistema a las acciones del usuario, y suele ser abordado por diseñadores y evaluadores especialistas en campo de la interacción persona-ordenador.

Ese enfoque sistemático es precisamente el segundo enfoque básico de los que se han apuntado. En general, este enfoque es más lento y muy metódico, con un desarrollo incremental más que a saltos. Representa la visión del ingeniero del software, y se caracteriza por el estudio de las tareas de usuario así como la programación tanto de la interfaz de usuario como del resto de la aplicación. Otro importante componente de esta filosofía es, como es de imaginar, la evaluación. Existen muchas y bien conocidas técnicas de evaluación de la componente software de la interfaz de usuario, que tienen como propósito comprobar la fidelidad de la implementación al diseño y los requisitos. Sin embargo, la usuabilidad no suele ser uno de sus objetivos, y de hecho la ingeniería de la usabilidad emplea un conjunto de métodos bien diferentes, pues no es el componente software lo que se evalúa sino la interacción usuario-máquina [11].

No cabe duda por ello que la cooperación entre ingenieros de software e ingenieros de la usabilidad resulta esencial para conseguir que el desarrollo de nuevas interfaces alcance un estado de madurez similar al que gozan las actuales interfaces WIMP. Y este objetivo está claramente asumido por nuestro grupo de investigación, no en vano las siglas de LoUISE hacen referencia a la interacción con el usuario y a la ingeniería del software.

7 Conclusiones

En clara oposición a la lenta evolución que están experimentando las interfaces de usuario, año tras año la industria del sílice nos sorprende con procesadores más y más rápidos. La Ley de Moore nos dice que los ordenadores continuarán doblando su capacidad de procesamiento cada año y medio sin ningún aumento en el coste. Siguiendo esta tendencia, los ordenadores del futuro pondrán a nuestro alcance una tremenda cantidad de información que hace ineludible el desarrollo de nuevos conceptos en el campo de las interfaces de usuario que nos ayuden a encontrar la información precisa en este océano de información.

No es extraño, por tanto, que se anime a dejar atrás las actuales interfaces WIMP para evolucionar hacia interfaces más ricas, naturales e intuitivas, interfaces que nos permitirán emplear nuestra vista, nuestro oído y nuestro tacto como medio de interacción, y nos facilitarán la comunicación a través del diálogo y los gestos, tal y como ocurre en la vida real. En una década podremos ver a los usuarios interactuando con los ordenadores más como competentes compañeros de tareas que como incansables trabajadores que ejecutan sólo aquellas tareas que se les ordena de manera explícita.

Pero para que las interfaces post-WIMP que se investigan en los laboratorios logren el mismo grado de madurez que las actuales interfaces, es preciso realizar una ingente labor de investigación en muchos frentes. Como se ha expuesto en este artículo, el laboratorio de interacción persona-ordenador está centrando sus esfuerzos en dos componentes principales de las interfaces post-WIMP, como son la tecnología de agentes y las interfaces tridimensionales. La segunda ha copado la mayor parte de nuestra atención, presentando el estado actual de esas interfaces, sus problemas de usabilidad y la complejidad de su desarrollo. Sin duda, es este desarrollo nuestra

mayor preocupación dentro del grupo de investigación, cuyas siglas recuerdan la colaboración que debe existir entre la ingeniería del software y la ingeniería de la usabilidad.

Referencias

[1] 3DNA Corp.: 3DNA Desktop. URL: http://www.3dna.net
[2] Actuality Systems: Perspecta 3D System. URL: http://www.actuality-systems.com
[3] Bolt, R.A.: Put that there – voice and gesture at the graphics interface. Proceedings of SIGGRAPH'80, ACM Press (1980) 262-270
[4] Bowman, D.A., Kruijff, E., LaViola, J.J., Poupyrev, I.: An Introduction to 3-D Interface Design. Presence, Vol. 10, No. (2001) 96-10
[5] Bowman, D.A., Wingrave, C.A., Campbell, J.M., Ly, V.Q., Rhoton, C.J.: Novel Uses of Pinch Gloves for Virtual Environment Interaction Techniques. Virtual Reality Journal, Vol. 6, No. 3 (2002) 122-129
[6] ClockWise Tech.: Win3D. URL: http://www.clockwise3d.com
[7] FakeSpace Systems: Pinch Gloves. URL: http://www.fakespace.com/worktools1.shtml
[8] Fencott, C.: Towards a Design Methodology for Virtual Environments. User Centered Design and Implementation of Virtual Environments (1999) URL: http://www.cs.york.ac.uk/hci/kings_manor_workshops/UCDIVE/
[9] Fernández Lobo, I.: Information Walkthrough - Visualización 3D de Sistemas de Información. I Jornadas Interacción (2000) 231-237
[10] Furness III, T.A.: Toward Tightly Coupled Human Interfaces. En Frontiers in Human-Centred Computing, Online Communities and Virtual Environments, Springer Verlag, London, R. Earnshaw, R. Guedj, A. van Dam and J. Vince, Eds., (2001) 80-98
[11] Gabbard, J., Hix, D., Swan, J.: User-centered design and evaluation of virtual environments. IEEE Computer Graphics & Applications, Vol. 19, No.6 (1999) 51-59
[12] IBM: RealPlaces Design Guide. IBM Ease-of-Use Website: http://www-3.ibm.com/ibm/easy/eou_ext.nsf/Publish/580
[13] Id Software. URL: http://www.idsoftware.com
[14] Leavitt, N.: 3D Technology: Ready for the PC? IEEE Computer Magazine, Vol. 34, No. 11 (2001) 17-20
[15] Leftwich, J.: InfoSpace – A Conceptual Method for Interacting with Information in a Three-Dimensional Virtual Environment. Third International Conference on Cyberspace (1993)
[16] Media Machines: Flux X3D browser. URL: http://www.mediamachines.com/products.html
[17] Nielsen J.: 2D is better than 3D (1998) URL: http://www.useit.com/alertbox/981115.html
[18] Open Croquet. URL: http://www.opencroquet.org
[19] Panoram Inc. URL: http://www.panoram-tech.com
[20] Raskar, R., Welch, G., Cutts, M., Lake, A., Stesin, L., Fuchs, H.: The office of the future: a unified approach to image-based modelling and spatially immersive displays. SIGGRAPH'98, Orlando, FL (1998)
[21] Robertson, G., Card, S. K., Mackinlay, J. D.: Information visualization using 3D interactive animation. Communications of the ACM, Vol. 36, No. 4 (1993) 57-71
[22] Robertson, G., van Dantzich, M., Robbins, D., Czerwinski, M., Hinckley, K., Risden, K., Thiel, D., Gorokhovsky, V.: The Task Gallery: A 3D Window Manager. Proceedings of CHI 2000, ACM Press (2000) 494-501

[23] Shneiderman, B.: Designing the User Interface – Strategies for Effective Human-Computer Interaction. Pearson Education (1998)
[24] Silicon Graphics: FSN – File System Navigator. URL: ftp://ftp.sgi.com/sgi/fsn
[25] StereoGraphics Corporation: SynthaGram Monitor. URL: http://www.stereographics.com/ products/synthagram/synthagram.htm
[26] Tebbutt, D.: Desktop Evolution. IBM Ease-of-Use Website. http://www-3.ibm.com/ibm/easy/eou_ext.nsf/Publish/582
[27] van Dam, A.: Post-WIMP User Interfaces. Communications of the ACM, Vol. 40, No. 2 (1997) 63-67
[28] van Dam, A.: Post-WIimp User Interfaces: the Human Connection. En Frontiers in Human-Centred Computing, Online Communities and Virtual Environments, Springer Verlag, London, R. Earnshaw, R. Guedj, A. van Dam and J. Vince, Eds., (2001) 163-178

Nuevos Mecanismos para el Desarrollo de Sistemas Interactivos de Calidad

María Dolores Lozano y Francisco Montero

Departamento de Informática
Escuela Politécnica Superior de Albacete
Universidad de Castilla-La Mancha
ALBACETE
{maria.lozano, francisco.montero}@uclm.es

Resumen. Uno de los grandes retos a la hora de desarrollar sistemas software es la calidad. Dentro de este concepto se encuentran diferentes apreciaciones que permitirán definir un sistema como de calidad. En este artículo se presenta, desde una perspectiva amplia, el concepto de calidad y, a partir de él, el de usabilidad. Éste se plantea como medida de la calidad percibida por el usuario de un sistema software. Tras ello se describe cómo las metodologías software deben recoger e integrar dentro del proceso de desarrollo de software algunas ideas y técnicas procedentes de la disciplina HCI (Human Computer Interaction) para alcanzar diseños de mayor calidad. Por último, se aborda el estudio de la evaluación de la calidad. En este caso, junto a la definición de evaluación de la calidad en el ámbito del diseño de la interfaz, se presenta algunas de las técnicas y métodos más representativos propuestos hasta el momento.

1 Introducción

Aunque el término calidad parece autoexplicativo y es utilizado de manera habitual, en la práctica suele resultar un tanto difícil presentar una única visión de lo que significa y de cómo debemos ser capaces de desarrollar software de calidad.

En este ámbito, podemos encontrar ciertas recomendaciones internacionales, como la ISO 9000 [1], en la que se asocia la calidad con la certeza de que el producto a desarrollar o producir satisfará los requisitos establecidos. Esta definición acerca la consecución de calidad a las personas que realizan las tareas de producir y especificar los requisitos y puede darse la paradoja de que un producto se considere con calidad aunque parta de unos requisitos incorrectos. Esta definición de calidad es a la que Garvin [2] denomina calidad de manufacturación.

Otro acercamiento a la definición de calidad la podemos encontrar en la ISO 8402 [3] la cual define calidad como la totalidad de las características de una entidad que afectarán a su habilidad para satisfacer necesidades establecidas e implícitas. Dentro de esta línea se mueve la propuesta de la ISO 9126 [4] en la cual se describen

las características que permiten describir la calidad de un producto software como: funcionalidad, eficiencia, usabilidad, fiabilidad, mantenibilidad y portabilidad.

Si consideramos que las necesidades de los usuarios son bien conocidas y comunes a todos los usuarios implicados podríamos considerar que la calidad es una característica inherente al producto. Sin embargo, si distintos grupos de usuarios tienen diferentes necesidades, ellos requerirán ciertas características específicas del producto para considerarlo como un producto con calidad en base a sus propósitos. Por ello la calidad se convierte, en cierto sentido, en una calidad que depende de la percepción de los usuarios y puede surgir, por tanto, el concepto de calidad percibida. Este término se asocia al conjunto de las características de un producto que aportan una gran satisfacción a unos usuarios específicos. De este modo podría decirse que un producto sólo tendrá calidad en relación a sus propósitos establecidos. Así las características o atributos que medirán la calidad de un editor de texto no coincidirán plenamente si dicho editor es utilizado por programadores/as o secretarios/as. Del mismo modo la funcionalidad, usabilidad y eficiencia requerida al procesador de texto serán diferentes para personas expertas o usuarios ocasionales.

En este entorno la ISO 14598-1 [5] define la calidad externa como el instante en el cual un producto satisface las necesidades establecidas e implícitas cuando éste es utilizado bajo ciertas condiciones específicas. Con ello se mueve el foco del término calidad del producto como algo aislado, a la satisfacción de unas necesidades de unos usuarios en unas circunstancias particulares. De este modo llegamos al concepto calidad en el uso [1], definido por la ISO 14598-1 [5] como "la efectividad, eficiencia y satisfacción con la cual ciertos usuarios específicos pueden alcanzar ciertas metas en entornos concretos". Este concepto está muy próximo a la definición de usabilidad que el comité de ergonomía del software aporta dentro de la ISO 9241-11 [6]. En este caso se define usabilidad como "el instante en el cual un producto puede ser utilizado por usuarios específicos para alcanzar metas específicas con efectividad, eficiencia y satisfacción en un contexto de uso específico".

2 Definición de Usabilidad

Como ya hemos visto, la definición de la calidad en el ámbito del diseño de la interacción con el usuario se asocia con el término "usabilidad". Éste sustituye a otros menos formales como "interfaces amigables" y, como vimos anteriormente, puede definirse atendiendo a la ISO 9241-11 [6] como "el instante en el cual un producto puede ser utilizado por usuarios específicos para alcanzar metas específicas con efectividad, eficiencia y satisfacción en un contexto de uso específico". Esta definición nos llevaría a definir tres posibles valores a medir: efectividad, eficiencia y satisfacción. Asimismo es interesante de esta definición la importancia de realizar las medidas de dichos parámetros en "un contexto específico", lo cual nos lleva a pensar en la necesidad de realizar adaptaciones en función de dicho contexto.

En todo caso dentro del área de HCI (Human Computer Interaction) podemos encontrar otras definiciones que intentan precisar en la definición y que, en dicho intento, aportan nuevas características que nos ayudan a comprender qué se esconde

detrás de la "usabilidad". En este sentido J. Nielsen [7] introduce los siguientes parámetros para definir la "usabilidad": aprendizaje; eficiencia; necesidad de recordar; errores; satisfacción. Como vemos en esta nueva aproximación al término de usabilidad encontramos, junto a eficiencia y satisfacción, otros conceptos que nos ayudan a pensar hacia donde debemos dirigir nuestra mirada cuando tratamos de mejorar la calidad de nuestro sistema en el capítulo de la interacción con el usuario. Así debemos pensar que el sistema debe facilitar su aprendizaje, reducir la necesidad de recordar cosas (reducir la memorización necesaria para manejar el sistema) y evitar que el usuario cometa errores en su manejo y ayudarle a resolverlos en caso de que los cometa. Por último Fabio Paternò [8] describe la "usabilidad" como un concepto multidimensional que contendría por lo menos las siguientes características: efectividad o relevancia (cómo de bien el sistema atiende a las necesidades de los usuarios); eficiencia (con qué eficacia pueden realizar los usuarios las tareas); actitud del usuario (sentimientos subjetivos en el manejo del sistema); comprensibilidad (nivel de facilidad con el que los usuarios pueden utilizarlo por primera vez y con el que pueden utilizarlo después de algún tiempo); seguridad (aportar la posibilidad de que los usuarios puedan deshacer acciones y no permitir que el sistema realice acciones destructivas). En estas dos últimas definiciones los conceptos manejados para describir la usabilidad son muy parecidos y por tanto pueden considerarse complementarias.

Pero en todo caso estas definiciones también consideran que la usabilidad debe ajustarse a un contexto dado y, por tanto, plantearán soluciones para cada uno de los entornos posibles. Esta idea, que puede ser válida en el diseño de una aplicación para una organización concreta, deja de serlo tanto si planteamos el diseño de una aplicación para la venta masiva o incluso planteamos el diseño de sistemas para ser explotados en entornos Web dónde, en muchos casos, se desconocen a priori los usuarios, o mejor dicho los usuarios potenciales son todos. A su vez, dentro de este término "todos", no podemos olvidar a personas con ciertas discapacidades. En este ámbito surge, por tanto, la necesidad de extender el concepto de usabilidad y hablar de "accesibilidad", vista ésta como "usabilidad universal" [9]. Este concepto puede definirse como "diseñar productos que sean usables por el rango más amplio de personas, funcionando en el rango más amplio de situaciones".

La idea de diseñar sistemas para personas con discapacidad no debe considerarse como una "obra social", pues debemos ser conscientes de que todos en algún momento o situación somos discapacitados (ver tabla 1) y en dichas situaciones deseamos ser capaces de manejar los sistemas.

Discapacidad	Personas Discapacitadas	Personas sin Discapacidad
Sin visión.	Ciegos.	▪ Con ojos ocupados (conduciendo). ▪ En la oscuridad.
Poca visión.	Con limitaciones Visuales.	▪ Manejando un visualizador pequeño.
Sin audición.	Sordos.	▪ Oídos ocupados. ▪ Silencio forzado (biblioteca).
Oído limitado.	Con poca audición.	▪ En entornos ruidosos.
Manipulación Limitada.	Movilidad limitada; falta de extremidades.	▪ Con vestidos especiales. ▪ En un vehículo inestable (tren).
Limitaciones Cognitivas.	Con problemas cognitivos.	▪ Distraídas. ▪ Consumo de alcohol. ▪ Sueño.
Lectura Limitada.	Con problemas cognitivos.	▪ Desconocen la lengua.

Tabla 1. Situaciones Habituales de Discapacidad.

Para entender el término "accesible" podemos desglosarlo en los siguientes factores:

- *Uso equitativo*.- El diseño debe ser usable y su manejo debe tener un coste razonable para personas con diferentes habilidades.
- *Uso flexible*.- El diseño debe acomodar a un rango amplio de personas con distintos gustos y habilidades.
- *Uso simple e intuitivo*.- El uso del sistema debe ser fácil de entender, independientemente de la experiencia del usuario, conocimiento, habilidades del lenguaje y nivel de concentración.
- *Información perceptible*.- El diseño comunica al usuario la información necesaria de manera efectiva, independientemente de las condiciones ambientales y las habilidades sensoriales del usuario.
- *Tolerancia para el error*.- El diseño minimiza posibles incidentes realizados por azar y las consecuencias adversas de acciones no previstas.
- *Esfuerzo físico mínimo*.- El diseño se ha de poder usar eficientemente y confortablemente con un mínimo de fatiga.
- *Tamaño y espacio para poder aproximarse y usar el diseño*.- El diseño ha de tener un espacio y un tamaño apropiado para la aproximación, alcance y uso del sistema.

Como hemos indicado, el ámbito donde tiene mayor importancia el diseño accesible, ya que se desconoce a los usuarios de los sistemas, es en el diseño de aplicaciones y sistemas Web. En este entorno es interesante referenciar a los trabajos que se están llevando a cabo dentro del Web Accessibility Initiative del W3C [10] cuyo objetivo es definir unas normas para alcanzar Web accesibles.

Realmente el objetivo a perseguir para alcanzar la máxima calidad en el resultado final del diseño de la interacción es conseguir verdaderos "interfaces invisibles". Esto es, ser capaces de definir interfaces que no sean percibidos como tales por los

usuarios y por lo tanto el usuario no tenga que pensar en cómo utilizarlos. Debemos ser conscientes de que todos los objetos del mundo que nos rodean tienen una interfaz y que el objetivo de estos es conseguir que el usuario no sea consciente de ello, sino que los utilice de manera satisfactoria, eficiente e "intuitiva", es decir, sin entrenamiento previo.

3 Usabilidad en el Proceso de Desarrollo de Software

Para obtener un sistema con las características de usabilidad y accesibilidad comentadas en anteriores apartados, es necesario incorporar esos criterios durante el ciclo de vida del desarrollo de software desde las etapas más tempranas de recopilación de requisitos hasta las etapas finales de implementación, pruebas y mantenimiento.

Por tanto, el gran reto es cambiar la concepción del término "usabilidad" para pasar de ser un factor de evaluación de diseños, a ser realmente un conductor o guía del diseño. Actualmente nos encontramos en una fase donde la usabilidad se utiliza principalmente para mejorar un mal diseño, pero el siguiente paso es no empezar a hacer el diseño hasta no haber hecho previamente un estudio minucioso y profundo de la usabilidad esperada del sistema en función de las necesidades y características de los usuarios de dicho sistema. Es decir, se debería hacer primero un análisis de los usuarios y sus necesidades antes de empezar a hacer cualquier desarrollo.

Como plantea B. Shneiderman [9], dentro de lo que se denomina "ingeniería de la usabilidad", una de las bases para alcanzar altas cotas de calidad es utilizar métodos de diseño iterativo que permitan realizar pruebas tempranas sobre prototipos, revisiones basadas en la retroalimentación de las observaciones y reacciones de los usuarios y los refinamientos sucesivos propuestos por los expertos en usabilidad.

A su vez, J. Nielsen [7] plantea que el modo más sencillo de aplicar con éxito los criterios de usabilidad es realizar el mayor esfuerzo posible antes de que el proceso de diseño de la presentación de la interfaz comience. En este aspecto lo más importante es realizar un estudio previo que permita "conocer a los usuarios". Este conocimiento no sólo se basa en estudiar sus características o habilidades individuales sino que también es necesario conocer las tareas que necesitan realizar. Igualmente dentro del ámbito de la ingeniería de la usabilidad es importante que los usuarios puedan realizar evaluaciones (test de usabilidad) sobre prototipos del sistema final, con lo que pueden aportar mejoras que se puedan incorporar en la siguiente versión.

Landauer [11] destacó que el 80% de los costes de mantenimiento (lo que representa el 80% del coste total del desarrollo del software) se deben a problemas de interacción entre el usuario y el sistema y no a problemas técnicos. Además comenta que el 64% de esos costes están relacionados con problemas de usabilidad.

Esta situación resalta la importancia de la usabilidad de los sistemas y justifica la necesidad de incorporar criterios de usabilidad durante todo el ciclo de vida del proceso de desarrollo de software. Esto implica, además, que el desarrollo de aplicaciones informáticas es una actividad multidisciplinar, que debe incorporar

también técnicas ergonómicas y de factores humanos con el objetivo de mejorar las condiciones de trabajo de los usuarios. Esto supone la participación de diferentes expertos en varias disciplinas, no sólo ingenieros del software, sino además expertos en disciplinas como psicología, ergonomía, factores humanos, sociología, etc.

Retomando la definición dada anteriormente del término "Usabilidad" de acuerdo a la normativa estándar ISO 9241 [6], la Usabilidad es *"la forma en la que un producto puede ser utilizado por usuarios concretos para conseguir objetivos concretos con efectividad, eficiencia y satisfacción en un contexto de uso concreto"*. Entendiendo estos tres términos como:

- *Efectividad*: La precisión y completitud con las que el usuario alcanza objetivos concretos.
- *Eficiencia*: Los recursos necesarios empleados en relación con la precisión y completitud con la que los usuarios alcanzan los objetivos.
- *Satisfacción*: La aceptación del sistema por parte de los usuarios, es decir, la actitud positiva de los usuarios hacia la utilización del producto.

Los distintos componentes junto con las relaciones que se establecen entre ellos se ilustran en la figura 1.

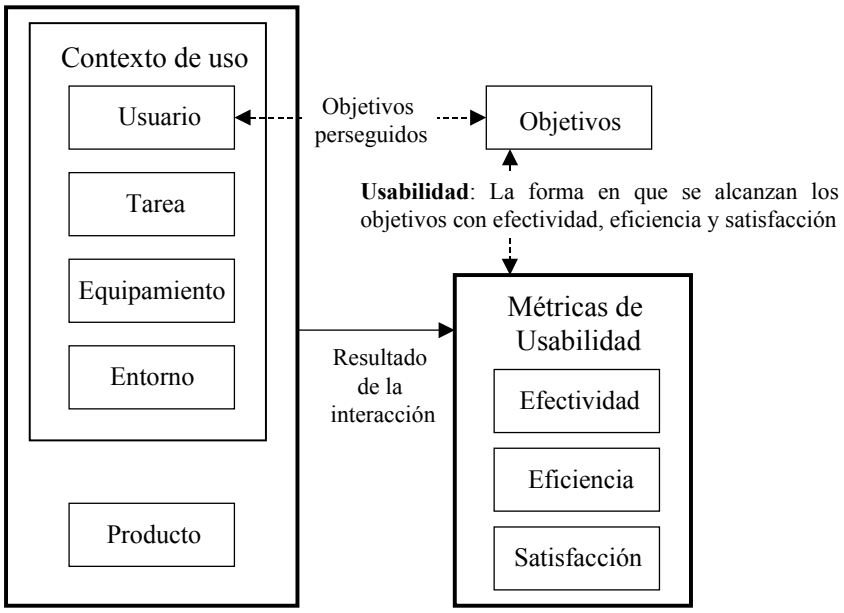

Fig. 1. Marco de Aplicación de Usabilidad de acuerdo a la ISO/DIS 9241.

El estándar establece, además, que cuando se especifica o mide la usabilidad de un sistema, es necesario tener en cuenta la siguiente información:

1. Una descripción de los objetivos que pretende cubrir el sistema que se va a construir.
2. Una descripción de los componentes del contexto de uso incluyendo usuarios, tareas, equipamientos y entornos. Esto puede ser la descripción de un contexto existente en la realidad, o por el contrario, puede ser la especificación del contexto que se desea obtener. Los aspectos relevantes del contexto y el nivel de detalle requerido dependerán del alcance y ámbito del tema de que se trate. La descripción del contexto debe ser suficientemente detallada como para que aquellos aspectos del contexto que puedan tener una influencia significativa en términos de usabilidad, puedan ser reproducidos sin dificultad.
3. Los valores reales o perseguidos de efectividad, eficiencia y satisfacción para el contexto deseado.

3.1 Contexto de Aplicación de la Usabilidad

El contexto de uso definido por el estándar mencionado, incluye los siguientes factores:

Descripción de los Usuarios: Son aquellas características de los usuarios del sistema que deben ser tenidas en cuenta. Dichas características pueden incluir desde su nivel de conocimiento informático, experiencia, estudios, habilidad, educación, entrenamiento, atributos físicos y capacidades motoras y sensoriales. En este sentido, puede ser necesario definir las características de los diferentes tipos de usuarios, por ejemplo, usuarios con diferentes niveles de experiencia o que desempeñen roles diferentes en el sistema.

Descripción de las Tareas: Las tareas son las actividades que se deben llevar a cabo para alcanzar un objetivo concreto. En este sentido es necesario describir aquellas características de las tareas que pueden influir en términos de usabilidad, por ejemplo, la frecuencia o la duración de las tareas.

También puede ser necesario describir de forma detallada las actividades y procesos involucrados en la realización de las tareas en caso de que la descripción del contexto se utilice como base para el diseño o la evaluación de los detalles de interacción con el sistema. Esto puede incluir la descripción de la localización de las actividades y los pasos entre los recursos humanos y tecnológicos. Además, las tareas no se deben describir solamente en términos de las funciones o características proporcionadas por el sistema, cualquier descripción de las actividades y pasos a seguir en la ejecución de la tarea deben relacionarse con los objetivos que se desean alcanzar.

Descripción del Equipo: La descripción del hardware, software y demás material que pueda ser utilizado, es también necesaria ya que la especificación o evaluación de la usabilidad también puede depender de ellos, es decir, se pueden establecer criterios

de usabilidad en términos de un conjunto de atributos o características de rendimiento del hardware, software o cualquier otro material.

Descripción del Entorno: Es necesario describir también las características relevantes del entorno social y físico. Entre los aspectos que se deben describir en este sentido destacan ciertos atributos que rodean el entorno tecnológico tales como el entorno físico de trabajo (p.e. el espacio de trabajo, mobiliario, etc.), el entorno ambiental (p.e. temperatura del lugar de trabajo, humedad, etc.) y el entorno social y cultural (p.e. grupos de trabajo, estructura organizacional, etc.)

Métricas de Usabilidad: Las métricas de usabilidad incluyen efectividad, eficiencia y satisfacción, definidas anteriormente. Estos factores son medidos en las pruebas del sistema realizadas por los usuarios finales.

El objetivo de las pruebas del usuario final es ayudar en la definición de los requisitos de usuario, validar que la tecnología creada funciona correctamente en condiciones reales y evaluar la actitud del usuario tras la utilización del sistema.

En resumen, usabilidad significa que las personas que utilizan el sistema lo puedan hacer rápida y fácilmente para llevar a cabo sus tareas. Esta definición se basa en cuatro principios básicos:

- *Usabilidad significa centrarse en el usuario*, es decir, evaluar desde el punto de vista del usuario final. Para desarrollar un sistema usable, se tiene que conocer, comprender y trabajar con las personas que representan los usuarios actuales o potenciales del sistema. En definitiva, el que decide si un sistema es fácil de utilizar es el usuario, no el desarrollador.
- *Las personas utilizan productos para ser más productivos.* El usuario considera que un sistema es "fácil de aprender y usar" en términos del tiempo que les lleva hacer lo que tienen que hacer, el número de pasos que tienen que recorrer y el éxito que tienen a la hora de predecir correctamente la siguiente acción que tienen que ejecutar. Para desarrollar sistemas usables, por tanto, hay que comprender los objetivos de rendimiento de los usuarios del sistema.
- *Los usuarios son gente ocupada que intenta llevar a cabo sus tareas.* Los usuarios relacionan usabilidad con productividad, porque el objetivo final de un sistema informático es ayudarles a llevar a cabo su trabajo de forma más rápida y fácil que si lo hacen a mano. El hardware y el software son, en definitiva, herramientas para ayudar a la gente ocupada a hacer su trabajo.
- *Son los usuarios los que deciden cuándo un sistema es fácil de utilizar.* Los usuarios, no los analistas ni los desarrolladores, determinan cuándo un producto es fácil de utilizar. Podemos encontrar muchos ejemplos en la vida cotidiana, muchas de las funciones que nos ofrecen, por ejemplo, un procesador de textos, un microondas, una cámara de vídeo no se usan porque sencillamente no son fáciles de aprender, usar y sobre todo recordar la próxima vez que se necesiten utilizar.

3.2 Integración de Criterios de Usabilidad en Metodologías de Desarrollo de Software. IDEAS, un Ejemplo de Integración

Teniendo en cuenta todos los factores expuestos anteriormente, hay que destacar que la mayoría de las metodologías de desarrollo de software no proporcionan los mecanismos adecuados para identificar y especificar las necesidades y requisitos de los usuarios, así como la validación de esos requisitos durante todo el proceso de desarrollo del software, desde el principio hasta el final. Como resultado de esta deficiencia de las metodologías actuales, los sistemas desarrollados siguiendo tales métodos puede que satisfagan todos los requisitos técnicos, ser internamente correctos, completos y robustos pero, aún así, puede que el usuario final no sea capaz de utilizar toda la potencialidad de la aplicación a través de su interfaz, por no haber sido ésta desarrollada siguiendo una metodología específica que incorpore criterios de usabilidad durante el proceso de desarrollo. Este problema explica en parte que la mayoría de las peticiones de cambios para modificar los servicios de una aplicación sean solicitados después de su puesta en marcha, y en la mayoría de los casos, por problemas en la interacción del usuario con el sistema. Ni siquiera el desarrollo más reciente, UML [3] y el Proceso Unificado asociado [12] aportan soluciones a esta situación.

La solución a esta situación básicamente consiste en adoptar una aproximación de desarrollo orientada al usuario que se caracteriza por [13]:

- La participación activa del usuario así como una clara comprensión del contexto de uso y de las tareas y requisitos del usuario final.
- Una distribución apropiada de funciones entre los usuarios y la tecnología, especificando qué funciones deben ser llevadas a cabo por los usuarios y cuáles son de índole tecnológica.
- La iteración de las soluciones de diseño en las que la retroalimentación del usuario es esencial como fuente de información.
- Una perspectiva de diseño multidisciplinar que requiere de gran variedad de especialidades o disciplinas (desarrolladores de software, expertos en usabilidad, especialistas en ergonomía, etc.).

Para cubrir estos principios, surge la necesidad de definir nuevos modelos que permitan recoger los requisitos y especificar las diferentes características de la interfaz asociada a una determinada aplicación. Estos modelos deben permitir que se defina una especificación de la interfaz independiente del resto del sistema. Es decir, los modelos utilizados para especificar la interfaz deben ser distintos de los utilizados para especificar la aplicación.

En la literatura podemos encontrar varias propuestas en este sentido, que fundamentalmente se basan en la construcción de nuevos modelos declarativos. Concretamente podemos destacar el entorno IDEAS [14], un entorno de desarrollo y especificación de interfaces de usuario integrado en el proceso de desarrollo de software global, que consiste básicamente en la introducción de nuevos modelos de interfaz de usuario a lo largo del proceso de desarrollo que recogen la información necesaria para, finalmente, generar de forma automática la interfaz. El proceso de desarrollo del entorno IDEAS se resume en la figura 2.

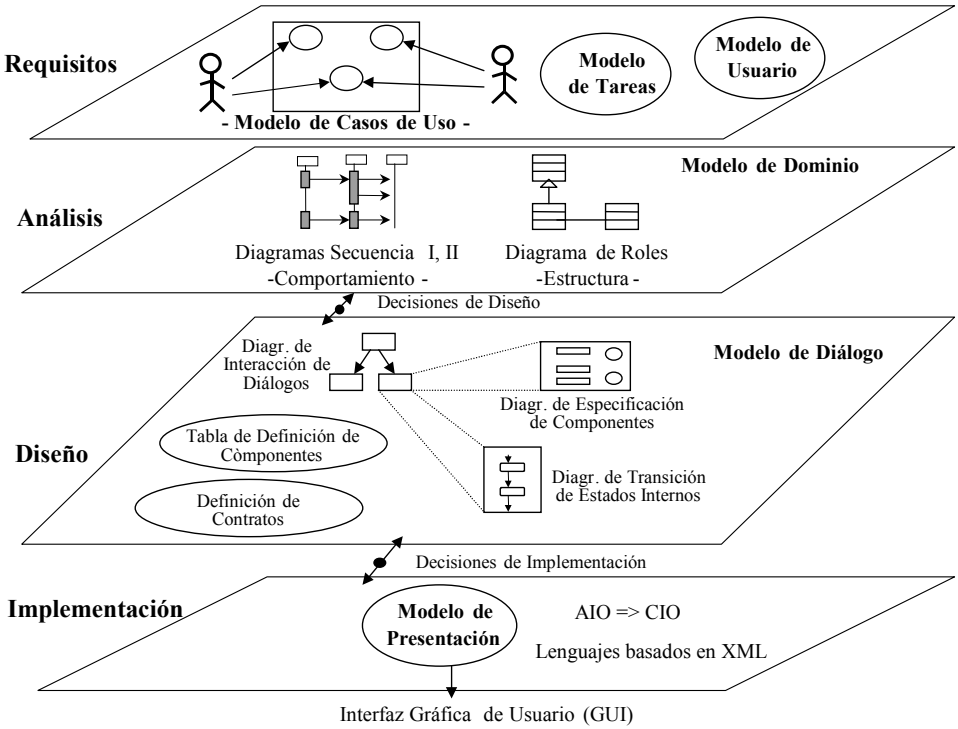

Fig. 2. IDEAS: Entorno de Desarrollo de Interfaces de Usuario.

En definitiva, IDEAS es un entorno metodológico que incorpora lo que se conoce como "ingeniería de la usabilidad" [7] en su proceso de desarrollo tal y como se presentará a continuación. Sin entrar en detalles metodológicos, destacaremos principalmente aquellos aspectos del entorno que incorporan los criterios de usabilidad establecidos en anteriores apartados.

De acuerdo a los criterios definidos por Nielsen, el primer paso en el proceso de desarrollar un producto con criterios de usabilidad es estudiar los usos y usuarios potenciales de dicho producto. Establece, además, que los dos factores con mayor impacto en el ámbito de la usabilidad son las características de los usuarios particulares y la variabilidad de las tareas. Por tanto, son estos dos factores los que hay que estudiar con mayor cuidado.

En este sentido, IDEAS incorpora en su primera fase del proceso de desarrollo, un modelo de tareas y un modelo de usuario.

El modelo de tareas realiza una caracterización y definición detalla de todas las tareas que el usuario final puede realizar con el sistema. Esta identificación de tareas se obtiene a partir de la funcionalidad definida por los casos de uso identificados previamente.

El modelo de usuario nos permite "conocer al usuario". En este modelo se identifican, en primer lugar, los diferentes tipos de usuarios del sistema. Una vez definidos los tipos de usuarios se establece la relación entre usuarios y tareas, es decir, se tiene en cuenta que no todas las tareas pueden ser accesibles para todos los usuarios del sistema. De este modo se implementa la idea de crear interfaces de usuario adaptadas y personalizadas a los distintos tipos de usuarios que un mismo sistema puede tener. Pero dentro de una misma tarea, puede haber acciones que ciertos usuarios con acceso a esa tarea no puedan ejecutar o que ejecuten de forma diferente a lo establecido en la tarea inicialmente. Para contemplar esta posibilidad se añade a este modelo una segunda caracterización que define, para cada usuario y tarea a la que tiene acceso, la proyección sobre las acciones que dentro de esa tarea puede realizar. Finalmente se establece una tercera caracterización en el modelo que se refiere al tipo de visualización establecida por usuario y tarea. Es decir, se define la forma más adecuada de mostrar la información relacionada con la tarea en función de las características particulares del usuario. Por ejemplo, es posible definir dos visualizaciones diferentes para una misma tarea y dos tipos de usuario con acceso a esa tarea.

En esta etapa inicial del desarrollo es esencial establecer las metas de usabilidad que se desean alcanzar. En definitiva, tras conocer a los usuarios y las tareas que realizan, se propone un estudio que permita establecer las metas o criterios básicos de usabilidad que el sistema debe alcanzar. Estos criterios se establecen a diferentes niveles. En primer lugar se establecen criterios de usabilidad a nivel de sistema, es decir, de forma global independientemente del usuario. En segundo lugar, se establecen criterios de usabilidad a nivel de tarea, es decir, la forma en la que una tarea debe llevarse a cabo de forma efectiva, eficiente y satisfactoria independientemente del usuario que la realice. En tercer lugar, se establecen criterios de usabilidad a nivel de usuario concreto, es decir, en función de las características particulares de cada usuario se establecen ciertos criterios personalizados puesto que lo que para un determinado tipo de usuario puede ser satisfactorio, para otro puede no serlo. Y finalmente se establecen otros criterios de usabilidad a nivel de usuario y tarea, puesto que la ejecución de una misma tarea se puede plantear de forma diferente en función del usuario que la ejecute, buscando siempre satisfacer al usuario final en función de sus características particulares. Esta definición permitirá más adelante realizar tests de usabilidad asociados a dichos criterios en cada uno de los niveles expuestos. Es decir, si al principio no se establecen los objetivos concretos de usabilidad que se pretenden alcanzar, no se pueden hacer evaluaciones para comprobar si se han alcanzado o no. Es necesario tener claro desde el principio los objetivos a cubrir.

Otro aspecto importante en el proceso de usabilidad es el prototipado, es decir, la posibilidad de realizar tests empíricos con usuarios reales. En este sentido, al ser IDEAS un proceso de desarrollo iterativo e incremental favorece este principio tal y como se describe en la figura 3. No es necesario haber desarrollado el sistema completamente para poder realizar ya pruebas sobre la funcionalidad desarrollada. El proceso de desarrollo se realiza de forma vertical, es decir, se toma un caso de uso definido en la etapa de requisitos, y se desarrolla pasando por las distintas etapas, hasta llegar a la implementación de la funcionalidad definida por ese caso de uso, materializada por las distintas tareas que pueden realizar los usuarios. Esta

funcionalidad puede ser probada por los usuarios, y en función del resultado y de la retroalimentación obtenida en este proceso volver al principio y mejorar dicha funcionalidad o añadir nuevas características.

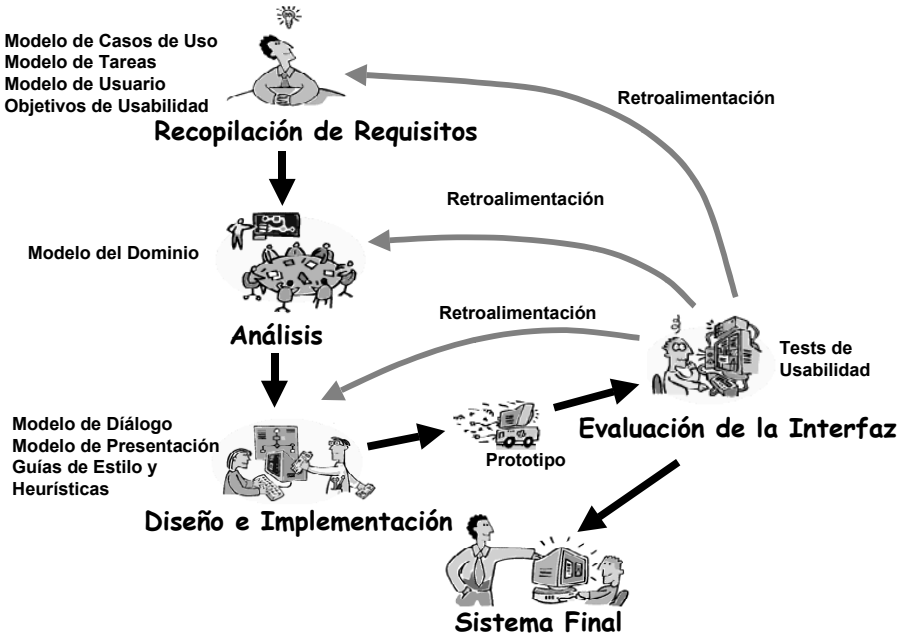

Fig. 3. Proceso de Desarrollo Iterativo Incorporando Criterios y Test de Usabilidad

Este proceso de retroalimentación permite introducir en las siguientes iteraciones del proceso de desarrollo, la información recogida en las pruebas realizadas por los usuarios, mejorando notablemente la usabilidad de las interfaces diseñadas.

En la fase de diseño, a la hora de construir el modelo de diálogo, se tienen en cuenta las diferentes caracterizaciones establecidas en el modelo de usuario, de forma que al diseñar las diferentes pantallas, se le ofrecen al usuario las herramientas más adecuadas para que pueda ejecutar las acciones permitidas dentro de las tareas a las que tiene acceso. Estos diseños, que se realizan de forma abstracta y mediante Objetos de Interacción Abstractos (AIOs), serán finalmente implementados en el modelo de presentación, seleccionando para cada uno de los AIOs el objeto de interacción concreto (CIO) más adecuado de acuerdo a las guías de estilo y heurísticas seguidas en esta fase.

Nielsen aboga por realizar diseños paralelos, en los que diferentes diseñadores realizan diseños preliminares de forma independiente con el objetivo de explorar distintas alternativas de diseño antes de seleccionar uno concreto. Esta interesante idea puede ser materializada en IDEAS estableciendo diferentes modelos de presentación a partir del mismo diseño. Dichos modelos de presentación serán evaluados por los usuarios y en función de esa evaluación se decidirá la implementación final de la interfaz.

Sin embargo hay que destacar que la posibilidad de realizar diferentes modelos de presentación a partir de un mismo modelo de diálogo, no sólo nos permite probar y decidir finalmente la implementación más adecuada para la ejecución de cada tarea, sino que nos permite además generar interfaces de usuario para distintos entornos. Es decir, esta metodología no está condicionada a la plataforma final de implementación ni a un lenguaje de implementación concreto, los diseños realizados en el modelo de diálogo son aplicables a cualquier plataforma final de implementación, ya sea una interfaz para un entorno web, una interfaz Windows en un PC o cualquier otro dispositivo de última tecnología, PDA, WAP, etc.

4 Medida de la Usabilidad de una Interfaz de Usuario

Llegado a este punto del artículo disponemos de un mapa proporcionado por la metodología propuesta en el apartado anterior. Un mapa en el que se nos dice que el diseño de interfaces de usuario es un proceso iterativo y centrado en el usuario. Pero de poco nos servirá dicho mapa, es decir, saber que tenemos que iterar en el diseño y desarrollo de un producto informático y de su interfaz, si desconocemos en qué punto nos encontramos. No podremos mejorar algo a menos que podamos evaluarlo.

El atributo que nos sirve de referencia para realizar la evaluación del sistema que se pretenda diseñar o desarrollar es la usabilidad. Ésta debe ser sistemáticamente tenida en cuenta durante el proceso de desarrollo [7] con el fin de garantizar la calidad final del sistema, manteniéndose presente, además, la satisfacción de requisitos de los diferentes perfiles de usuario y el contexto de uso. Para tal fin, se han de considerar, dentro del ciclo de vida de desarrollo de sistemas, aquellas actividades, metodologías y procedimientos que conduzcan a asegurar el control de la calidad [15]. Estas actividades no son obligatorias sino que representan un marco conceptual, donde los distintos métodos, procedimientos y herramientas se pueden aplicar.

El potencial de aplicación de técnicas de evaluación no queda circunscrito al proceso de desarrollo de un producto, también puede utilizarse durante la fase operativa para ponderar la mejora obtenida respecto a versiones anteriores del mismo producto o respecto a productos competidores.

En los siguientes puntos se presentarán métodos, procedimientos y criterios de medida utilizados para ponderar la usabilidad que presenta o puede presentar un producto y se relacionarán con la metodología presentada en apartados anteriores (Figura 3).

Antes de nada será preciso incluir algunas definiciones relacionadas con los procesos de evaluación de calidad. Por **evaluación** se entiende el proceso por el que se estima, se aprecia, se calcula el valor de una cosa. En nuestro caso lo que pretendemos evaluar es la usabilidad. En este sentido, por **métrica** nos vamos a referir al arte que trata de medir la usabilidad bajo un determinado criterio o parámetro, utilizando un método, un procedimiento o una técnica determinada o cualquier combinación de ellas.

En nuestro caso, se trata de ponderar la usabilidad que ofrece un sistema informático en función de su interfaz, éste será nuestro criterio. El problema que se presenta es que el hecho de medir directamente la usabilidad, la aceptación de un

sistema, o su grado de calidad es demasiado complejo debido a que son atributos demasiado abstractos y subjetivos, son lo que se denominan criterios últimos. Muchos de esos atributos fueron presentados en la introducción de este artículo, son atributos que dependen del usuario final que pretenda utilizar el producto que se quiere desarrollar o ya esté desarrollado, son criterios difícilmente ponderables directamente. Frente a ellos y como estimadores de los mismos deben establecerse otros criterios que si que deben ser ponderables, y así permitir medir la usabilidad de una interfaz, estos se denominan criterios actuales.

Los **métodos de evaluación de usabilidad** (UEM) propuestos van asociados a estos criterios actuales y deben integrarse en la metodología de desarrollo de cualquier sistema en todas sus fases, y cuanto antes mejor. Con ellos se pretende determinar diferentes aspectos [8]:

- Si el diseño es lo suficientemente bueno.
- Si estamos comparando diferentes productos: poder determinar cuál es mejor.
- La fidelidad con la que el producto refleja el mundo real.
- Si el producto se ajusta a los estándares disponibles.

5 Métodos de Evaluación de Usabilidad

El tema que nos interesa es el desarrollo de interfaces de usuario de calidad, y lo que pretendemos es saber cuándo hemos alcanzado determinado grado de usabilidad en el producto que estamos diseñando o poder comparar con un producto competidor en función de ese mismo criterio, para ello se han propuesto diferentes métodos, técnicas y procedimientos englobados dentro de lo que se conocen como métodos de evaluación de la usabilidad.

5.1 Características que debe cumplir un Método de Evaluación

Las características propias de las que debe hacer gala un método de evaluación pasan por [16]:

- Ser fácil de interpretar y de calcular.
- Poder aplicarse en diferentes fases de desarrollo, a prototipos y a modelos.
- Tener una base conceptual simple, junto con un análisis razonado fuerte.
- Tener el suficiente grado de detalle y la posibilidad de discriminar entre diseños.
- Dar soporte a actividades de diseño (la métrica o su proceso de computo deberían aportar información que sugiera formas para mejorar el diseño del producto).
- Permitir una predicción eficiente de la usabilidad.
- Permitir indicar la calidad relativa de los diseños realizados.

5.2 Clasificación de los Métodos de Evaluación de la Usabilidad

La taxonomía de los métodos de evaluación puede ser muy variada en función del criterio de clasificación elegido. Wixon, en [17], nombra cinco posibles criterios de clasificación de los métodos de evaluación:

- **Métodos formativos frente a sumativos:** Los primeros se utilizan para la generación de nuevas ideas y detección de posibles problemas de usabilidad de forma temprana durante el proceso de análisis, diseño y posterior desarrollo del sistema. Los segundos se utilizan para evaluar sistemas ya existentes.
- **Métodos de descubrimiento frente a métodos de decisión:** Los de descubrimiento, también denominados métodos cualitativos, se utilizan para determinar cómo trabaja, piensa o qué problemas encuentra el usuario. Los de decisión sirven para seleccionar un diseño de entre varias alternativas, estos también se denominan cuantitativos.
- **Métodos formales e informales:** Muchos métodos se describen formalmente, pero en la práctica los evaluadores los adaptan en función de sus necesidades de manera informal.
- **Métodos que involucran al usuario y métodos que no lo hacen:** Los métodos de evaluación se pueden clasificar en función de que precisen de la presencia del usuario o no durante la evaluación, análisis y el diseño del producto informático.
- **Métodos completos frente a métodos componente:** Algunos métodos cubren todos los pasos necesarios para completar un diseño centrado en la usabilidad. La ingeniería de usabilidad como tal es un método completo. La mayoría de los métodos son métodos componente, así que representan sólo parte de un proceso completo de usabilidad.

Estos son algunos de los criterios que aparecen en la bibliografía relacionada con la presentación de métodos de evaluación de la usabilidad de una forma más o menos explícita. Para la presentación, en el siguiente apartado, de algunos métodos de evaluación representativos y de las fases del ciclo de vida en la que pueden utilizarse, se atenderá a la distinción en función de la presencia o no del usuario.

5.3 Algunos Ejemplos de Métodos de Evaluación de la Usabilidad

Como se ha apuntado en la sección anterior, presentaremos ahora algunos de los métodos que pueden utilizarse para mejorar la usabilidad en un contexto que ofrece un producto. La presentación agrupará los métodos en función de la participación del usuario final (*user testing*) o de que éste sea sustituido por expertos o por guías de estilo y heurísticas (*usability inspection*). Al final, lo que ocurre es que en función de las necesidades de tiempo y de dinero se utilizan varios de estos métodos conjuntamente y en diferentes fases del ciclo de desarrollo de cualquier producto [7].

El grupo de técnicas y métodos agrupados bajo el epígrafe de *user testing* se caracteriza por otorgar al usuario un papel principal. En este caso, se le proporcionan al usuario prototipos del producto final y se le observa (*observation*, *contextual inquiry* [18]), se le pide que comente sus pensamientos en voz alta (*thinking aloud* [Eri80]) utilizando el producto, se le sigue mientras trabaja con el prototipo o el producto (*usability test*, *informal walkthrough*, *visual walkthrough*, *pluralistic usabiilty walkthrough* [2]), se le hacen preguntas sobre sus percepciones al utilizar el producto (cuestionarios, entrevistas) y sobre los posibles problemas de usabilidad que haya podido encontrar. Los cuestionarios permiten recoger información sobre percepciones del usuario relacionadas con la satisfacción lograda utilizando el prototipo. Ejemplos de este tipo de cuestionarios pueden ser:

- **El SUMI (Software Usability Measurement Inventory)** [19] es un cuestionario utilizado para la evaluación de la calidad de un conjunto software, puede utilizarse tanto para nuevos productos como para efectuar comparaciones con versiones previas y establecer objetivos para desarrollos futuros. Con él se pueden establecer escalas sobre percepciones de sentimientos emocionales del usuario, facilidad de aprendizaje, eficiencia y control. Las respuestas que se esperan del usuario son de tres niveles: "de acuerdo", "no sé", "en desacuerdo". Preguntas ejemplo de este tipo de cuestionario son: "Las instrucciones y advertencias son de ayuda", o "El modo en que se presenta la información del sistema es clara y comprensible". Requiere de una decena de usuarios para la obtención de datos significativos y no más de treinta si se pretende llevar a cabo una encuesta.

- **El MUMMS (Measuring the Usability of Multi-Media Systems**) [19] tiene el mismo objetivo que el anterior, está ideado para medir percepciones de satisfacción del usuario al utilizar aplicaciones multimedia y se caracteriza por la consideración de cinco subescalas:
  - La medida en que el producto capta las respuestas emocionales del usuario.
  - El grado de control con el que el usuario siente que él, y no el producto, va procediendo paso a paso.
  - El grado de eficiencia con el que el usuario siente que puede conseguir los objetivos de su interacción con el producto.
  - El nivel de ayuda y asistencia que el producto parece prestar al usuario.
  - La facilidad de aprendizaje con la que se puede empezar y aprender con un producto.

- **El WAMMI (Website Analysis and Measurement Inventory)** [19] herramienta para la evaluación de sitios web. Se basa en el cuestionario que cumplimentan los lectores, lo que proporciona una medida de la utilidad y facilidad de uso que encontraron en el sitio en cuestión. Puede utilizarse con fines de predicción, monitorización y/o referencia.

- **El SUS (System Usability Scale)** [4] su propósito era proporcionar un test fácil de completar, fácil de puntuar y que permitiera establecer comparaciones

cruzadas entre productos. Se utiliza después de que un usuario ha tenido la oportunidad de utilizar un sistema pero antes de que cualquier informe o discusión tenga lugar. La escala SUS es un único número representando una medida compuesta de la usabilidad del sistema global sometido a estudio.

Con este tipo de cuestionarios se identifican problemas de usabilidad que se le presentarán al usuario cuando haga uso del producto informático. Con ellos podemos evaluar aspectos como pueden ser la apariencia estética, el control, la eficiencia, la utilidad, la facilidad de aprendizaje relacionados todos ellos con la satisfacción del usuario, uno de los factores principales de la usabilidad. El hecho de involucrar al usuario y de la necesidad de que éste haga uso de prototipos o del producto ya desarrollado hace que este tipo de métodos de evaluación sea costoso en tiempo y en dinero, además, requieren de mayor preparación y planificación, selección de grupos de usuario, recreación del contexto de utilización y demás factores que influyen en el terreno de las percepciones del usuario.

Como contrapartida a los métodos anteriores, se definen los englobados dentro del término *usability inspection*, que prescinden, en principio, del usuario. En ellos se utilizan expertos en el diseño de interfaces de usuario, así como la consulta de guías de estilo o heurísticas [20], [21], [9]. Estas últimas no son sino formas de recopilar el conocimiento. Otra forma de realizar esta recopilación es mediante patrones. Desde hace unos años este término, el de *interaction design patterns* [22], está apareciendo asociado al diseño de interfaces de usuario, como un intento de salvar los problemas que lleva aparejado el uso de guías de estilo al utilizarlas para el diseño de interfaces [23]. Estos métodos pueden aplicarse antes en el ciclo de desarrollo de cualquier producto, precisan de menos recursos temporales y económicos y son útiles a la hora de ponderar factores de usabilidad que tienen que ver más con la efectividad y la eficiencia que con la satisfacción del usuario. Dentro de ellos destacan los de evaluación heurística [7] (*heuristic evaluation*), recorrido cognitivo [24] (*cognitive walkthrough*), GOMS [25], evaluación automática o inspecciones formales de usabilidad.

Todos estos métodos presentados pueden incluirse en una metodología de análisis, diseño y desarrollo de interfaces de usuario como la propuesta en la Figura 3. En la tabla 2 se presentan métodos que permiten alcanzar mayores cotas de usabilidad y la fase del ciclo de vida donde pueden utilizarse.

Aparte de los métodos presentados que permiten obtener información, fundamentalmente cualitativa, sobre la usabilidad que puede esperarse de un producto en desarrollo y poder así identificar y eliminar problemas de usabilidad que pudiera presentar el producto final desarrollado, son también interesantes otras métricas de índole cuantitativa. Estas métricas permiten obtener información sobre criterios actuales que tienen relación directa con la usabilidad final que presenta el producto, así se puede ponderar la bondad de otros criterios últimos. Ejemplos de criterios actuales son los propuestos por [7], véase tabla 3.

Fase del ciclo de vida	Métodos de mejora de la usabilidad
Recopilación de Requisitos y Análisis	- Indagación - Observación (estudio etnográfico) - Brainstorming - Encuestas, Cuestionarios, Entrevistas - Categorización (por tarjetas) - Prototipado (papel, rápido, Mago de Oz)
Diseño e Implementación	- Técnicas de composición de interfaces - Diseño paralelo - Secuencias de escenarios - Construcción de escenarios - Cuadros de asignación de tareas - Análisis de tareas - Matriz de funcionalidad - Guías de estilo, heurísticas, patrones
Evaluación de la interfaz	- Inspecciones formales de usabilidad - Evaluación heurística - Revisiones cognitivas - Guías de comprobación - Cuestionarios, Entrevistas - *Thinking aloud*

Tabla 2. Relación Métodos de Evaluación y Ciclo de Desarrollo de un Producto Software

Medidas típicas cuantificables de usabilidad
- Tiempo que los usuarios se toman para completar una tarea específica. - El número de tareas de diversos tipos que pueden ser completadas dentro de un tiempo límite dado. - La relación entre las interacciones exitosas y los errores. - El tiempo empleado en la recuperación de los errores. - El número de errores de usuario. - Porcentaje de competidores que consiguen una mejor marca. - El número de acciones erróneas inmediatamente posteriores. - El número de comandos u otras características que fueron utilizadas por el usuario. - El número de comandos y otras características que en ningún momento fueron utilizadas por el usuario. - El número de características del sistema que el usuario puede recordar durante una sesión informativa después del test. - La frecuencia del uso de manuales y/o ayuda del sistema y el tiempo empleado en el uso de estos elementos del sistema. - La frecuencia con la que el manual y/o ayuda del sistema resolvieron el problema del usuario. - La proporción de aseveraciones del usuario de carácter positivo durante el test frente a aquellas críticas hacia el sistema. - El número de ocasiones en que el usuario presenta una frustración evidente. - La proporción de usuarios que dicen preferir el sistema antes que otro de la competencia. - El número de ocasiones en que tuvo que volver el usuario sobre un problema irresoluble. - La proporción de usuarios haciendo uso de estrategias de trabajo eficientes en comparación con los usuarios que utilizan estrategias ineficientes. - La cantidad de "tiempo muerto" durante el cual el usuario no interactúa con el sistema. El sistema puede estar instrumentado para distinguir entre dos tipos de tiempo muerto: los retrasos asociados a los tiempos de respuesta donde es el usuario el que espera al sistema y los tiempos de reflexión en los que el sistema espera al usuario. - El número de ocasiones en que el usuario se desvía de la tarea real.

Tabla 3. Criterios Cuantificables de Usabilidad de Nielsen

Otros criterios que tienen influencia en la usabilidad del producto diseñado pueden ser los propuestos por [16] recogidos en la tabla 4 y que están inspirados en trabajos previos que pretenden cuantificar numéricamente la complejidad (*complexity*) o adecuación (*appropriateness*) del diseño, o la cohesión de los datos. Esta información es útil para estimar la facilidad de aprendizaje o memorización que tendrá la interfaz y, por tanto, lo más o menos proclive a errores y a dificultades en su utilización por parte del usuario.

Nombre	Formula	Criterio
Essential Efficiency	$EE = 100 * \frac{S_{essential}}{S_{enacted}}$ donde $S_{essential}$ es el nº de pasos ideal para completar la tarea y $S_{enacted}$ es el nº de pasos reales necesarios para completar dicha tarea.	Simplicidad
Task Concordance	$TC = 100 * \frac{D}{P}$ donde D es el nº de parejas de tareas clasificadas por orden en función de su extensión menos el nº de parejas fuera de dicho orden y P es el nº de parejas de tareas posibles.	Eficiencia, simplicidad
Task Visibility	$TV = 100 * \left(\frac{1}{S_{total}} * \sum_{\forall i} V_i \right)$ donde S es el número total de pasos necesarios para completar la tarea y V_i es la característica de visibilidad (entre 0 y 1) para el paso i-ésimo.	Visibilidad
Layout Uniformity	$LU = 100 * \left(1 - \frac{(N_h + N_w + N_t + N_l + N_b + N_r) - M}{6 * N_c - M} \right.$ donde N_i son los valores de las diferentes dimensiones de anchura, altura y alineamiento (superior, inferior, derecho e izquierdo) de los componentes visuales. N_c es el número total de componentes visuales en la pantalla y M es un valor de ajuste para el mínimo número de posibles alineamientos y tamaños necesarios para que LU se ajuste a un rango comprendido entre 0 y 100	Regularidad, uniformidad
Visual Coherence	$VC = 100 * \left(\frac{\sum_{\forall k} G_k}{\sum_{\forall k} N_k * \frac{(N_k - 1)}{2}} \right) \quad G_k = \sum_{\forall i, ji \neq j} R_{i,j}$ donde N_k es el número de componentes visuales en una agrupación de componentes k, y $R_{i,j}$ es un valor entre 0 y 1 que se asocia a la relación semántica entre los componentes i y j en la agrupación k.	Comprensibilidad

Tabla 4. Criterios Cuantificables de Usabilidad de Constantine

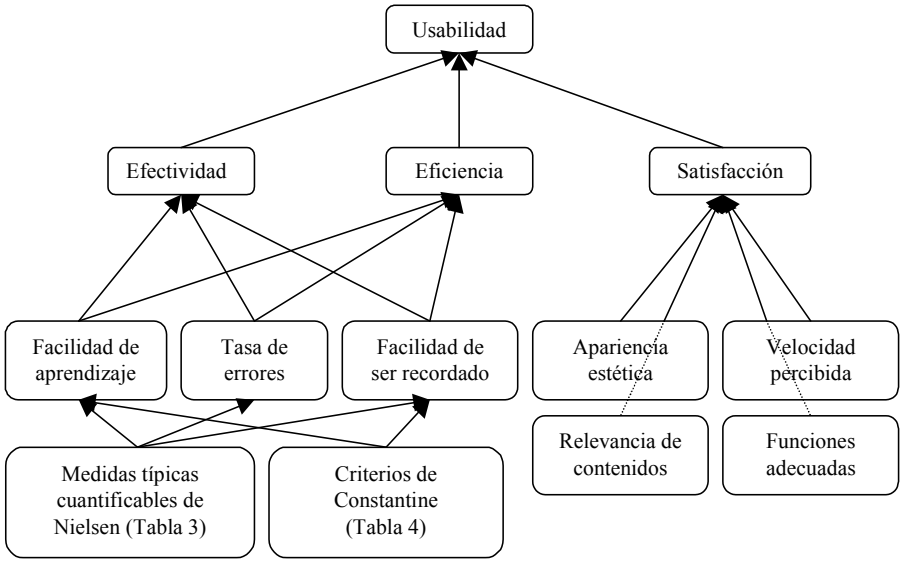

Fig. 4. La Usabilidad y sus Criterios de Evaluación

La figura 4 presenta un resumen de criterios últimos y actuales que se tienen en cuenta para evaluar el grado de usabilidad logrado en el análisis, diseño y desarrollo de interfaces de usuario.

El criterio último es la usabilidad, según las definiciones este criterio depende de la efectividad, la eficiencia y la satisfacción. La satisfacción del usuario se puede evaluar mediante el empleo de métodos que involucran al usuario; entrevistas, cuestionarios y/o observación son los métodos más utilizados. Los factores que la determinan son la apariencia estética, la velocidad percibida, la relevancia de contenidos, la adecuación de las funciones a la funcionalidad deseada, la tolerancia a errores y la protección que la interfaz preste al usuario también repercutirán en dicha satisfacción.

Por otro lado, la efectividad y la eficiencia tienen una vertiente más relacionada con las tareas que el usuario tiene que llevar a cabo, la facilidad con que las puede realizar, lo que le cuesta aprender a manejar la interfaz y la tolerancia con la que la interfaz responderá a errores del usuario. En este sentido la visibilidad, la comprensibilidad, la uniformidad, la regularidad, los tiempos de realización de las tareas, el número de errores, en general las medidas mostradas en las tablas 3 y 4 son factores mensurables que redundan en los criterios de los que depende la usabilidad, será con ellos con los que ésta pueda estimarse.

6 Conclusiones

En este artículo se ha presentado la idea de la calidad desde la perspectiva y la percepción que el usuario tiene del sistema. En este ámbito de la interacción hablar de calidad es hablar de "usabilidad". Este concepto tiene asociado una descripción multidimensional que incluye distintos atributos o propiedades asociadas. Una de las características básicas de este concepto es la fuerte dependencia de criterios subjetivos de los usuarios, lo que introduce cierta complejidad adicional.

Otro aspecto importante son las adaptaciones necesarias en el propio proceso de desarrollo de software para mejorar la calidad en el uso del sistema o usabilidad. Por ello en la segunda parte del presente trabajo se plantea la integración de propuestas provenientes del ámbito HCI dentro del campo de la ingeniería del software, pues para alcanzar la calidad necesaria a nivel de usabilidad el propio proceso de desarrollo de software debe ser alterado para recoger desde el principio ideas y técnicas que permitan describir al usuario y al uso que se le dará al sistema. En este ámbito se presentan ideas básicas de la "ingeniería de la usabilidad" y cómo una metodología debe integrar dichas propuestas básicas en el ámbito del desarrollo de sistemas software, pues la calidad no es algo que solamente deba analizarse al final sino que debe guiar todo el proceso de desarrollo. Para servir de ejemplo se presenta IDEAS, una metodología que pretende integrar el diseño de la interfaz dentro del propio proceso de desarrollo de software y la cual incorpora en su planteamiento las ideas propuestas entorno a la ingeniería de la usabilidad.

Finalmente se aborda uno de los aspectos más relevantes dentro del ámbito de la calidad que es el de la evaluación e incluso medida "cuantitativa" de la calidad de la interfaz. En este punto se presentan diferentes modos de clasificar los distintos métodos propuestos hasta el momento. Se describen desde métodos donde la subjetividad del usuario es un factor básico hasta otros métodos que incluyen métricas que permitirán realizar una valoración cuantitativa de la calidad final. Al mismo tiempo, como la evaluación de la calidad no es algo que deba dejarse para el final, se presenta una clasificación que permite ver en qué fases del ciclo de vida deben utilizarse cada uno de los métodos y técnicas propuestas.

Para terminar, acentuar la cada vez mayor relevancia de la calidad de la interfaz dentro del desarrollo de productos software, ya que es ésta la que permitirá a los usuarios descubrir otras características de calidad ocultas en el software. En este caso, es interesante remarcar también la necesidad de ampliar el concepto de "usabilidad" y llegar a la definición de "usabilidad universal" o "accesibilidad", el cual nos permitirá abordar un nuevo reto que es el de hacer "software usable" desde la perspectiva de cualquier usuario, problema que ya se nos presenta al desarrollar aplicaciones en entornos Web donde los usuarios potenciales son "todos". El gran reto consiste en poder alcanzar verdaderos "interfaces invisibles" para cualquier usuario. Esto es, ser capaces de definir interfaces que no sean percibidas como tales por los usuarios y por lo tanto el usuario no tenga que pensar en cómo utilizarlas.

7 Referencias

[1] ISO 9001. Quality systems- Models for quality assurance in design/development, production, installation and servicing. 1987.
[2] Garvin. What does "product quality" really mean?. Sloane Management Review, Fall. 1984.
[3] ISO 8402. Quality Vocabulary. 1994.
[4] ISO 9126. Software products evaluation – Quality characteristics and guidelines for their use. 1991.
[5] ISO 14598-1. Information Technology – Evaluation of software Products- Part 1 general guide. 1997.
[6] ISO 9241-11. Ergonomics requirements for office work with visual display terminal (VDT). Part 11 Guidance on usability. 1996.
[7] J. Nielsen. Usability Engineering. Academic Press. 1993.
[8] Paternò, F. Model-Based Design and Evaluation of Interactive Applications. Springer. 2000.
[9] B. Shneiderman. Designing the User Interface: Strategies for Effective Human-Computer Interaction. Addison-Wesley Publishers. 1998.
[10] Chisholm, W., Vanderheiden, G., Jacobs, I. "Techniques for Web Content Accessibility Guideline 1.0". W3C, WAI. www.w3.org/TR/WCAG10/. Mayo 1999.
[11] T. K. Landauer. The Trouble with Computers: Usefulness, Usability and Productivity. MIT Press, 1995.
[12] I. Jacobson, G. Booch, J. Rumbaugh. El Proceso Unificado de Desarrollo de Software. Addison Wesley. 2000.
[13] ISO-DIS 13407. Human-Centered Design Processes for Interactive Systems. 1998.
[14] M. Lozano. Entorno Metodológico Orientado a Objetos para la Especificación y Desarrollo de Interfaces de Usuario. Tesis Doctoral. Universidad Politécnica de Valencia. Valencia, 2001.
[15] D. J. Mayhew. The usability engineering lifecycle. A practitioner's handbook for user interface design. Morgan Kaufmann Publishers. 1999.
[16] L. Constantine, L. Lockwood. Software for use. A practical guide to the models and methods of usage-centered design. Addison-Wesley. 1999.
[17] D. Wixon, C. Wilson. The usability engineering framework for product design and evaluation. Handbook of human-computer interaction, 2nd ed. Elsevier Science. Pp. 653-688. 1997.
[18] K. Holtzblatt, H. Beyer. Making customer-centered design work for teams. Communications of the ACM, Vol. 36, nº 10. pp. 93-103. 1993.
[19] The Human Factors Research Group. University College Cork (UCC). http://www.ucc.ie. 1990
[20] R. Molich, J. Nielsen. Improving a human-computer dialogue. Communications of the ACM. Vol. 33, nº 3, pp. 338-348. 1990.
[21] J. Nielsen, R. L. Mack. Usability Inspection Methods. John Wiley & Sons, New York. 1994.
[22] M. Welie, H. Trætteberg. Interaction Patterns in User Interfaces. 7th. Pattern Languages of Programs Conference, 13-16 August 2000, Allerton Park Monticello, Illinois, USA. 2000.
[23] A. Dix, G. Abowd, R. Beale, J. Finlay. Human-Computer Interaction. 2nd edition, Prentice Hall Europe. 1998.

.

[24] C. H. Lewis, P.G. Polson, C. Wharton, J. Rieman. Testing a walkthrough methodology for theory-based design of walk-up-and-use interfaces. Proceedings of the ACM CHI'90. Conference on Human Factors in Computing Systems. pp. 235-242. 1990.

[25] S.K. Card, T.P. Moran, A. Newell. The psychology of human-computer interaction. Lawrence Earlbaum Associates, Hillsdale, N. J. 1983.

La Ingeniería de la Usabilidad aplicada al Diseño y Desarrollo de Sitios Web

Jesús Lorés, Toni Granollers

Departamento de Informática
Universidad de Lleida
Campus de Cappont
25001 Lleida
jesus@eup.udl.es; tonig@diei.udl.es

Resumen. En este documento se presenta una metodología de desarrollo de sitios web que pretende priorizar la incorporación de la usabilidad en un sitio web desde un principio utilizando conceptos y métodos que proceden de un espectro amplio de disciplinas relacionadas con la interacción persona-ordenador que se adaptan a los modelos mentales de los implicados a través de diversas técnicas de prototipado y evaluación. Esta metodología se ha validado a través de la realización de numerosos proyectos de sitios web que nos han permitido analizar, revisar y consolidar esta metodología.

1 Introducción

El Modelo de Proceso de la Ingeniería de la Usabilidad (MPIU) [1,2] especifica una metodología que guía al equipo de desarrollo de aplicaciones interactivas con altos niveles de usabilidad.

El mero hecho de conseguir aplicaciones que cumplan con los objetivos funcionales para las que han sido propuestas y desarrolladas no es una tarea fácil. Conseguir, además, que estas aplicaciones cumplan con todos los principios que permiten etiquetar a las mismas como "usables" es aún un proceso más laborioso si no se realiza siguiendo un disciplinado y riguroso procedimiento.

El MPIU tiene sus cimientos, por una parte, en la Ingeniería del Software (IS) y, por otra, en la disciplina de la Interacción Persona-Ordenador, que contribuye —entre otras— con toda una sólida base de conocimiento y un conjunto de técnicas y experiencias conocidas para el diseño de interfaces centradas en sus usuarios. Pretende ser una herramienta de trabajo para ayudar metodológicamente a los equipos de desarrollo. No especifica ni el uso de un determinado lenguaje de programación, ni ninguna tecnología específica, ni ningún factor que pueda determinar la aplicación, sino todo lo contrario, está pensado para todo tipo de aplicaciones y tecnologías —actuales y futuras. En definitiva, es independiente de los dispositivos y la tecnología.

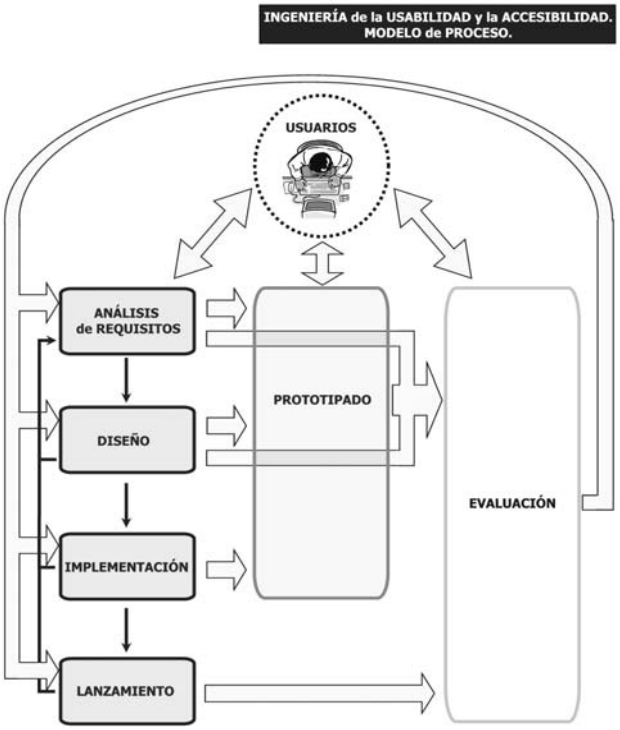

Fig. 1. Modelo de Proceso de la Ingeniería de la Usabilidad (MPIU)

Este modelo se adapta a una característica fundamental en los equipos de desarrollo de entornos interactivos, la interdisciplinariedad. En un estudio publicado en Ainda.info [3], web española especializada en usabilidad, se describe que sólo un 18% de los especialistas en usabilidad y Arquitectura de la Información son ingenieros, el resto son, por orden, periodistas, diseñadores, sociólogos, humanistas y psicólogos, hecho que restringe el uso excesivo de métodos formales informáticos que impedirían su uso tanto en miembros no informáticos como de los propios usuarios que deben implicarse en el proceso.

En este artículo centraremos los esfuerzos en explicar la aplicación del MPIU durante el desarrollo de aplicaciones o sitios web, entornos para los que deben tenerse en cuenta ciertas características que les hacen diferentes de cualquier otro tipo de aplicaciones.

La validación basada en proyectos reales es la base del trabajo de nuestro grupo de investigación. El mencionado modelo de proceso ha sido aplicado para desarrollar los siguientes sitios web (algunos de ellos en desarrollo):

Web de la entidad y del centenario del Centro Excursionista de Lleida (http://www.lleida.org/cel, http://www.lleida.org/cel100), web dinámica en JSP de la Asociación Interacción Persona-Ordenador (http://www.aipo.es), web dinámica en ASP y FLASH de la infancia del ayuntamiento de Lleida (http://www.paeria.org/infancia), web basada en XML del congreso de Interacción 2004 (http://griho.udl.es/I2004), web en XML del programa de doctorado en

Interacción Persona-Ordenador (http://griho.udl.es/ipo/doct), web del Laboratorio de Usabilidad de Griho, GLU (http://glu.griho.net), web de la estantería virtual de la Universidad de Lleida, web local de Amnistía Internacional, web de la Asociación de Moros y Cristianos, web de la ONG Lleida Solidaria y web del área de servicios sociales del ayuntamiento de Lleida, que cubren un amplio espectro de tecnologías y temáticas.

2 Algunas características de los sitios web

El perfil de los usuarios de las aplicaciones informáticas ha cambiado mucho desde la aparición de Internet; el abanico de posibilidades se amplía enormemente. La inmediatez y globalización hacen que el perfil genérico del usuario de un sitio web sea diferente: existe una gran diversidad de conocimientos, necesidades y formas de acceder a los servicios.

Con la aparición y enorme explosión de los sistemas web apareció un nuevo conjunto de conceptos, como los enlaces (links), las páginas (web), los navegadores (browsers), la inmediatez, la audiencia, la arquitectura de la información, la navegación, etc., que se han ido añadiendo a nuestro vocabulario diario. Las concepciones tradicionales de distancia y tiempo de repente desaparecen para generar un espacio en el que la separación espacial y temporal no existen.

Los sitios web constituyen hoy en día la mejor interfaz de integración de servicios, a la vez que conecta a las personas u organizaciones formando las denominadas *redes sociales* [4].

Hay que tener en cuenta que aproximadamente el 50% del código se dedica a la interfaz [5], aunque en el paradigma web este factor se incrementa enormemente (hay un gran número de sitios web en el que el 90% de los mismos no son más que elementos visuales puramente estéticos, o sea, elementos de la interfaz).

3 ¿Por qué es importante la usabilidad de los sitios web?

Todos somos conscientes de que la web se está convirtiendo en un elemento clave, tanto en el desarrollo de las empresas como de las instituciones, ofreciendo información y una amplia gama de servicios a través de la misma.

A pesar de ello, la web (o Internet) sigue sin ser indispensable para un extensa parte de la población y conseguir que se conviertan en internautas y/o futuros clientes on line dependerá directamente de su facilidad de uso, es decir, de su usabilidad.

Dicha usabilidad aporta el enfoque imprescindible para que las páginas de una empresa o entidad tengan el suficiente atractivo para que el visitante no sólo se quede y las visite, sino que regrese en el futuro. Para ello el diseño de las páginas, sus funciones, mensajes y contenidos deben estar diseñados e implantados para que lo pueda usar cualquier persona.

3.1 Problemas de usabilidad en la web

Aunque la web está basada en principios de interfaces relativamente simples —enlaces, botones, texto, menús, cajas de texto y gráficos—, los problemas de usabilidad serios son bastante frecuentes [5]:

- Problemas de *percepción humana*: Aparecen cuando, por ejemplo, un conjunto de páginas está diseñado de acorde a como la información está fisicamente almacenada en lugar de cómo ésta debe ser presentada para su comprensión.
- Frustrantes problemas de *navegación*: Desorientan al navegante motivando preguntas como ¿dónde estoy ahora?, ¿cómo he llegado aquí? o ¿qué debo hacer para...? son demasiado frecuentes. Todo ello se agrava cuando hay ambigüedad en la comprensión de los enlaces o no se siguen los estándares de los elementos de navegación.
- Deben tenerse en cuenta aspectos importantes acerca de la *memoria humana*. Si los usuarios tienen que recordar un elevado número de ítems seguro que alguno de ellos se perderá, agravándose si además deben recordar ciertos ítems de una página a otra.
- Gran parte de la información que las webs muestran provienen, cada vez más, de información almacenada en bases de datos, conllevando inconvenientes de usabilidad para el usuario final derivados de la no concordancia de la información mostrada con los datos reales que la base de datos asociada dispone —problema derivado de la sincronización de las páginas.
- Contenidos pobres, la lentitud en las descargas, los enlaces rotos, las opciones y menús confusos, el abuso de ventanas emergentes, la "moda de la letra muy pequeña", etc.

3.2 Beneficios que aporta la usabilidad a un sitio web

Los beneficios que la usabilidad puede aportar a la implementación de sitios web deben mirarse desde dos ópticas distintas:

Desde el punto de vista del desarrollador, implica una reducción de los costes de producción, mantenimiento y soporte (desarrollo), disminución de los costes de uso. Los sistemas fáciles de usar permiten una mayor productividad y una reducción del esfuerzo, mientras que los sistemas difíciles de usar disminuyen la salud, bienestar y motivación y pueden incrementar el absentismo. También produce menores costes de desarrollo al establecerse pautas generalizadas de diseño, reutilizables en diferentes aplicaciones departamentales (uso interno) y incremento de ventas —un producto más usable permite un mejor marketing—, un producto de mejor calidad garantiza aplicaciones más competitivas, menor soporte al cliente y facilidad en sustituciones y rotación de personal (ventas).

Desde el punto de vista del usuario, la confianza que produce la facilidad de uso facilitará su "fidelización" (el visitante volverá y posiblemente recomendará nuestro sitio a sus conocidos y amistades).

4 Aplicación del modelo de proceso a la web

Como vemos, parece evidente afrontar el desarrollo de una aplicación en el paradigma web teniendo objetivos de usabilidad claramente definidos, que no sería viable sin nuevas metodologías que nos permitan implementar la usabilidad a lo largo del proceso de desarrollo, siendo el MPIU anteriormente referenciado nuestra propuesta a dicha necesidad.

Aunque existen otras propuestas [5,6,7], nuestro modelo basa todo el proceso de desarrollo en una constante aplicación iterativa de las actividades de prototipaje y evaluación aplicadas a todas etapas considerando la usabilidad desde un principio, a la vez que integra, a diferencia de aquellas, actividades propias de la IS.

En este artículo veremos aquellos aspectos más importantes de cada etapa del MPIU cuando tratamos de aplicarlo para implementar un sitio web.

5 Análisis de requisitos

Para que una aplicación finalice con éxito depende de manera crítica de esta fase [8,9] y aunque este aspecto es ampliamente compartido por los desarrolladores de software lo cierto es que habitualmente esta fase no se realiza con el rigor que merece.

En el paradigma web los requisitos se centran en los grandes temas de estudio de la audiencia y las necesidades de los usuarios [5].

5.1 Estudio de la audiencia y de la plataforma

El objetivo del análisis de la audiencia es estudiar quienes serán nuestros usuarios y el entorno de software y hardware que vamos a utilizar.

Audiencia: Cuando empezamos un nuevo proyecto de sitio web lo hacemos pensando en una audiencia determinada. Realizando el análisis de la audiencia pretendemos conocer cómo son realmente estos futuros usuarios y cuáles son sus necesidades. Para ello, podemos basarnos en información procedente de fuentes empresariales [10,11] o, si la información no está disponible, deberemos proceder a la realización de nuestros propios estudios realizando encuestas y/o entrevistas.

Para ello, organizaremos la audiencia en categorías, para cada una de las cuales estableceremos el perfil, sus necesidades y sus metas.

A la hora de entender la audiencia no sólo tendremos en cuenta los atributos personales, sino que también deberemos analizar los distintos dispositivos y plataformas que utilizan para acceder a la información.

Escenarios: El objetivo de los escenarios —que son historias de usuarios que experimentan con el sitio para realizar un objetivo— es asegurarse que resuelve las necesidades de personas específicas en situaciones reales y nos asegura que hemos considerado todos los detalles necesarios.

Aspectos a tener en cuenta:

- Perfil del usuario.
- Planificación de un episodio interactivo.
- Una foto del usuario donde se realiza el escenario.

Ejemplo: Para la web de la infancia del ayuntamiento de Lleida se determinó una audiencia muy joven que, debido a sus peculiaridades, debía estructurarse en tres grupos: (1) de 3 a 6 años, (2) de 7 a 10 y (3) de más de 10 años.

5.2 Diseño para la diversidad

Como se ha mencionado en el párrafo anterior, otro aspecto a tener en cuenta es la enorme diversidad en cuanto a plataformas y velocidades de acceso, preferencias personales, navegadores, etc., factores que analizados con la audiencia deberían influir en el diseño. A estos aspectos hay que añadirle, además, las diferencias internacionales que directamente condicionan el modelo mental del usuario, sus costumbres y, por tanto, sus interpretaciones de la información.

Diferencias individuales: Segmento de mercado —edad, género, educación, ocupación, aficiones...—, discapacidades —visuales, auditivas, motrices...— y nivel de experiencia son tipos de diferencias entre individuos que deben ser especialmente consideradas al diseñar webs accesibles.

Diferencias en HW & SW: En este apartado se analizan los diferentes ordenadores, sistemas operativos, resolución de los monitores, navegadores y versiones de navegador, diferentes sistemas de acceso a la red y velocidades.

Internacionalización: Diferencias culturales que puedan existir para cada categoría de audiencia, idiomas, localizaciones.

5.3 Necesidades de los usuarios

En esta etapa, y partiendo del trabajo previo de análisis de la audiencia y de la plataforma, definiremos objetivos del negocio o entidad, los objetivos de usabilidad, definir los implicados, analizar la competencia y fijar las metas de éxito a conseguir.

5.3.1 Objetivos

Aunque este es uno de los puntos clave de cualquier análisis de requisitos, sea web o no, en el caso que nos ocupa son los objetivos de negocio más que los funcionales —aún así los objetivos funcionales son los más importantes, puesto que sin éstos normalmente la aplicación carece de sentido— los que deseamos identificar, ya que un parámetro que especialmente marcará el futuro de nuestro sitio será la medición del éxito en relación a dichos objetivos de negocio.

Objetivos de negocio (de la empresa)

En este apartado hemos de establecer por qué los usuarios visitarán este sitio, para entretenerse o para trabajar, para aprender o para aportar información, para conocer o para comprar. Si no sabemos establecer estas razones probablemente no visitarán el sitio.

Ejemplo: En la web de la infancia los objetivos de *negocio* son:

- La página debe servir de vínculo de comunicación entre la juventud y la Oficina de la Defensora de los Derechos de los Niños y Adolescentes.
- Tiene que ser un espacio educativo en el que a partir de diferentes actividades los jóvenes que participen desarrollen un proceso de educación en valores y respeto hacia los demás.
- Que sea una herramienta que permita conocer la ciudad de Lleida, sus costumbres, su cultura, los lugares más emblemáticos, etc.
- Que sea, además, un elemento de interacción entre los niños/niñas y las TIC.

Objetivos de usabilidad

Sin objetivos es difícil poder decidir cómo diseñar un sitio web. Sin objetivos cuantificables de usabilidad resulta imposible medir y valorar si el sitio es usable o no lo es.

Los objetivos de usabilidad necesitan ser identificados para después ser medidos [12]. Asignar un valor (o varios) como meta para cada objetivo proporciona al diseñador una línea base de medida, comparación y análisis.

En este apartado analizaremos los objetivos que consideramos más importantes en cuanto a usabilidad del sitio [5].

En la definición de estos objetivos debemos evitar ser demasiado simplistas o irrealistas. Por ejemplo, la clásica regla de los tres clics, que indica que el diseño de la estructura de un sitio web debe permitir al usuario llegar en tres clics de ratón a la información que desea, es en muchas webs un objetivo irreal. No deben ser 3 clics, sino los justos y necesarios. Usando este objetivo como punto de partida es responsabilidad de cada equipo de diseño junto con los usuarios determinar "qué" elementos de información del sitio son "importantes" y cuántos clics son aceptables para llegar a ellos [13].

En la tabla siguiente definimos una serie posible de objetivos de usabilidad:

Tiempo aprendizaje/tiempo tarea	Usar el sitio por primera vez sin entrenamiento Encontrar un tema por primera vez en menos de 2 minutos Usuarios expertos (5 visitas) menos de 30 segundos
Facilidad de aprendizaje	Medible por el tiempo que se tarda en la consecución de las tareas habituales
Número de errores	No visitar más de tres páginas erróneas para visitar una página No hacer errores fatales menos del 99% del tiempo
Impresión subjetiva	En una escala de 1 a 10 en cuanto a que el sitio sea atractivo como mínimo de 7 (medible con una encuesta)
Tareas realizadas	Como mínimo un 75% de los usuarios serán capaces de realizar una compra (en el caso de una web de compra on line)

Objetivos o especificaciones funcionales

Este apartado correspondería a la parte más tradicional del análisis de los requisitos de la IS, durante el cual se describe cada subsistema del software y todos los requisitos dentro de cada subsistema. Dado que es un tema suficientemente tratado en la IS no insistimos en él, salvo que es preciso comentar que puede darse el caso que algún objetivo de usabilidad entre en contradicción con alguna especificación funcional. Si esto sucede, el equipo de desarrollo junto a los encargados de mantener la calidad del sistema decidirán las acciones pertinentes.

5.3.2 Implicados

Los implicados son aquellas personas u organizaciones que se verán afectadas por el sistema a desarrollar y que tienen influencia directa o indirecta en los requisitos del mismo [14].

En un sitio de comercio-e los implicados pueden incluir a los vendedores, los distribuidores, la empresa de transportes, socios de negocio, publicistas, inversores, personas de otros departamentos relacionados (marketing, compras, ventas...).

Los usuarios finales del sistema, que evidentemente son los implicados en el mismo, no suelen englobarse en esta categoría porque su nivel de implicación respecto a los demás, por razones evidentes, es distinto. Por ello, los implicados suelen etiquetarse también como "usuarios indirectos" [5].

Una vez identificados los implicados, debe procederse a obtener toda la influencia relativa al proyecto que éstos pueden aportar al mismo. Uno de los métodos más habituales de obtener esta información es realizando una serie de reuniones de implicados planificadas conocidas como "*Stakeholders Meetings*" [14].

5.3.3 Análisis de la competencia

Seguramente nuestra web, sobre todo si es comercial, no será única y tendrá que competir contra otras similares. Deberemos, por tanto, realizar en esta etapa del desarrollo un análisis de todas las fuentes secundarias para conocer las fortalezas y debilidades de la competencia [15,16] encarado principalmente a la generación de ideas que deberán ser corroboradas por nuestros usuarios finales (que no tienen porque ser exactamente los mismos que los de dicha competencia).

Analizar la competencia sirve también para ver las buenas ideas que tienen los demás y que pueden ser aplicadas a nuestro negocio, pero debe procederse con mucho cuidado para no ultrapasar los límites de la propiedad intelectual.

Las etapas básicas a cubrir en esta actividad son: (1) Realizar un listado de la competencia correspondiente, (2) crear una tabla comparativa con la evaluación de cada sitio y (3) realizar una presentación (*Focus Group*, etc.) para revisar los resultados.

5.3.4 Establecer una medida de éxito

Un aspecto importante a tener en cuenta en todo sitio web es establecer unos criterios que puedan determinar su éxito, o lo que es lo mismo, si cumple con los objetivos para los que fue desarrollado. En la mayor parte de casos, la medida estará directamente relacionada con el objetivo del sitio, siendo el número de visitas, de solicitudes, de ficheros descargados o de ventas realizadas los parámetros habituales por los que se rigen.

En el caso del libro digital el éxito vendría dado por el número de descargas de los capítulos; en el de un congreso, el número de personas que finalmente se registrarán en el mismo; en el caso de la web de la infancia sería el número de niños de la ciudad de Lleida que la consultan y participan en sus juegos; en el caso de la estantería virtual, el número de profesores que han depositado material docente correspondiente a sus asignaturas y el número de alumnos que lo utilizan; en el centro excursionista será en el número de personas que acceden a la parte pública, y en el número de socios que consultan tanto la parte pública como la privada y la frecuencia con que lo hacen, etc.

Otra manera de considerar el éxito de un sitio es su clasificación en los buscadores y directorios de sitios web más importantes.

5.4 Prototipado en los requisitos

Maquetas: Las maquetas son básicamente una sola página de representaciones estáticas del espacio de diseño. Se usan para mejorar el diseño y facilitar la comunicación entre los diseñadores, usuarios e implicados. Consideraremos tres tipos:

- Boceto: Los bocetos son maquetas rápidas y pequeñas para desarrollar un amplio espectro de ideas de diseño. Se usan en la primera etapa del diseño, muchas veces antes de que se haya acabado la fase de análisis de requisitos. La clave de los bocetos es su velocidad. Cada boceto no puede costar más de 15 o 20 segundos, de esta manera se pueden generar una gran cantidad de bocetos en poco tiempo.
- Maqueta de papel: Son representaciones de una mayor calidad que los bocetos. Permite explorar el diseño de la página y los aspectos estéticos. Permite una comprensión más realista de los límites del tamaño de la página, del espacio de diseño, identificar posibles dificultades en el proceso de diseño y comenzar la evaluación con los usuarios.
- Maqueta digital: El prototipo digital constituye un paso más en la elaboración de prototipos realizados con herramientas de diseño gráfico, siendo, por tanto, más costosos y elaborados. Permiten afinar aspectos importantes del diseño como los colores, los contrastes, las fuentes tipográficas, etc. que el prototipo de papel no proporciona.

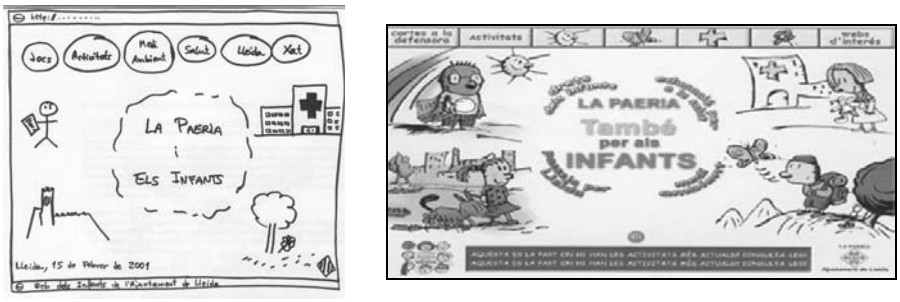

Fig. 2. Maquetas de papel y digital de la web de la Infancia de Lleida

5.5 Evaluación en la fase de requisitos

Realizar encuestas es especialmente útil en las etapas más iniciales del proyecto. Esta técnica está especialmente indicada para conseguir una definición precisa de la audiencia y está siendo, además, una técnica con una buena relación coste-beneficio [13]. Éstas, además, en función de la audiencia a la que va destinada, casi siempre pueden realizarse a través del correo electrónico y tecnología basada en Internet (alcanzando así un espectro de población muy amplio y diverso).

Entrevistas y/o grupos de discusión (*Focus Groups*) [17] con implicados (*Stakeholders*) [8,18] haciendo uso de maquetas o prototipos de papel nos proporcionarán reacciones subjetivas acerca de nuestras suposiciones que nos ayudaran a entender su entorno y cómo tratan de resolver sus problemas. El carácter individual de las entrevistas hace que los resultados obtenidos carezcan de influencias externas. Por el contrario, las influencias del grupo ensalzan aquellas ideas que un miembro destapa. Vemos, por tanto, que combinar ambas técnicas suele ser altamente beneficioso.

Evaluaciones a partir de *descripciones formales de escenarios de casos de uso* [19,20] describen los requerimientos del sistema en el contexto de las especificaciones funcionales mostrando cómo se efectúan los procesos de negocios y qué actores o perfiles de usuario intervienen en éstos a través de las secuencias de tareas descritas para cada uno de los escenarios.

Evaluaciones del tipo *recorrido cognitivo* [21] donde los usuarios evalúan preferentemente maquetas digitales son especialmente útiles cuando nos encontramos en una iteración un poco adelantada del modelo de proceso.

Fig. 3. Focus Group con usuarios y implicados realizado para evaluar la maqueta digital la web del CEL

6 Diseño

Durante la fase de análisis y especificación de los requisitos el objetivo es documentar lo que el sitio debe hacer. En la fase de diseño es importante determinar cómo será el sitio estructuralmente y gráficamente. En esta parte presentamos dos aspectos importantes en el diseño de un sitio web, el análisis de tareas y la arquitectura de la información. En los aspectos de prototipado y evaluación se presentan la ordenación de tarjetas, la maqueta digital y el Storyboard de uso.

6.1 Análisis jerárquico de tareas (HTA)

El análisis jerárquico de tareas (HTA Hierarchical Task Analysis) desarrollado por Annett y Duncan [22] es la técnica de análisis de tareas más conocida y más antigua, que sigue siendo válida aunque hayan aparecido nuevas metodologías [23,24].

En HTA se realiza una descripción de tareas en términos de operaciones y planes. Las *operaciones* (descomposición en subtareas) son actividades que realizan las personas para alcanzar un objetivo, y los planes son una descripción de las condiciones que deben darse cuando se realiza cada una de las actividades. Las operaciones se pueden descomponer de forma jerárquica y se asigna un plan a cada una de las subtareas que aparecen. Se define un objetivo como un estado determinado del sistema que puede alcanzar el usuario. Aunque se habla de objetivos y tareas, la representación que se realiza describe únicamente la descomposición jerárquica en subtareas de las tareas que aparecen en el sistema.

El formato gráfico se parece a un árbol con ramas y subramas en función de las necesidades. A la hora de describir la descomposición de unas tareas en subtareas podemos representar cuatro tipos de descomposiciones:

- *Secuencia*: Descomposición en un conjunto ordenado temporalmente de una secuencia de tareas.
- *Selección*: Conjunto de tareas de las que se tendrá que elegir una de ellas.
- *Iteración*: Repetición de un subconjunto de tareas.
- *Tarea unitaria*: Actividad indivisible (según el nivel de detalle dado).

El análisis de tarea implica tres etapas enlazadas: recogida de información, diagramación y análisis. Los procedimientos de recogida de información incluyen la revisión de la documentación existente (por ejemplo, manuales de funcionamiento, procedimientos, informes de seguridad, estudios de análisis de tareas previos, diseños, imágenes, prototipos, etc.), que permitan establecer lo que hacen las personas en circunstancias especificas (normales y anormales), entrevistas y cuestionarios (descripciones por parte de personas experimentadas de cómo hacen las cosas, qué informaciones necesitan y cómo determinan si la tarea se puede realizar satisfactoriamente).

Algunas tareas se pueden desglosar con mayor detalle en secuencias. Un plan describe el conjunto de operaciones necesarias para llevar a cabo una actividad, o bien, muestra las circunstancias por las que una operación es realizada antes que otra. Estos planes se añaden a la tabla jerárquica.

La descripción de la información se realiza en forma de tabla o en forma de diagrama de árbol que describa las relaciones entre tareas y subtareas.

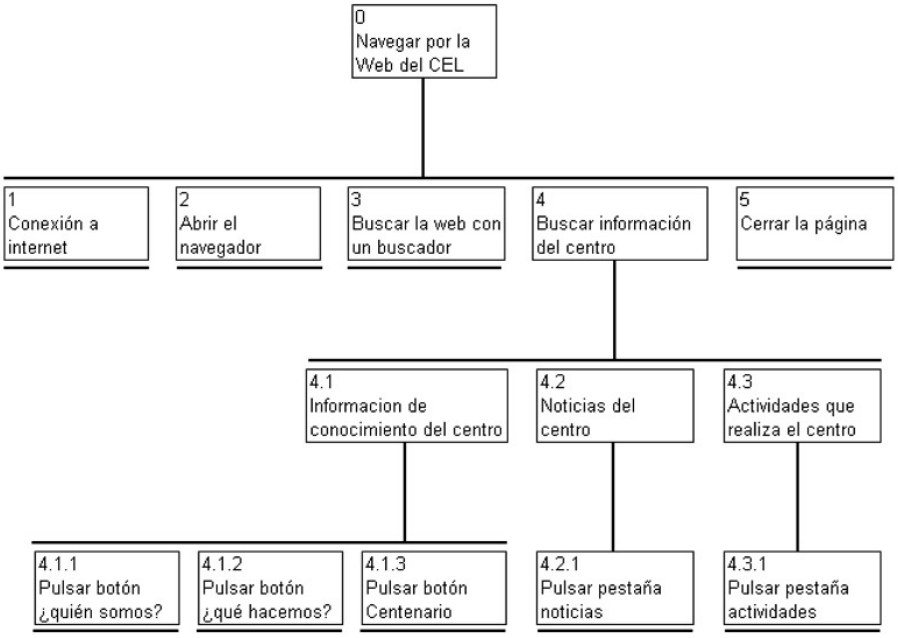

Fig. 4. Descripción en HTA de una tarea del Centro Excursionista

6.2 Arquitectura de la Información (AI)

La información es una fuente de conocimiento. Pero si no está organizada, procesada y disponible para las personas en un formato que les permita tomar decisiones, más que un beneficio es un estorbo.

La arquitectura de la información se refiere a la estructura de la organización del sitio, especialmente en como las diferentes páginas del sitio están relacionadas entre si. Implica opciones como la planificación y el análisis de los contenidos, organización de las páginas, proporcionando indicaciones para ayudar a los usuarios a orientarse, etiquetado, técnicas de búsqueda y diseño de la navegación [25,26].

La misma información puede tener diferentes estructuras razonables dependiendo de cómo la gente piense, hable de ella o la use.

Partiremos de la información aportada por el análisis de requisitos y el análisis de tareas. Se pueden revisar otras versiones del sitio web que estamos desarrollando y los de la competencia. Esto nos permitirá disponer de piezas de contenidos potenciales, etiquetas y esquemas de organización.

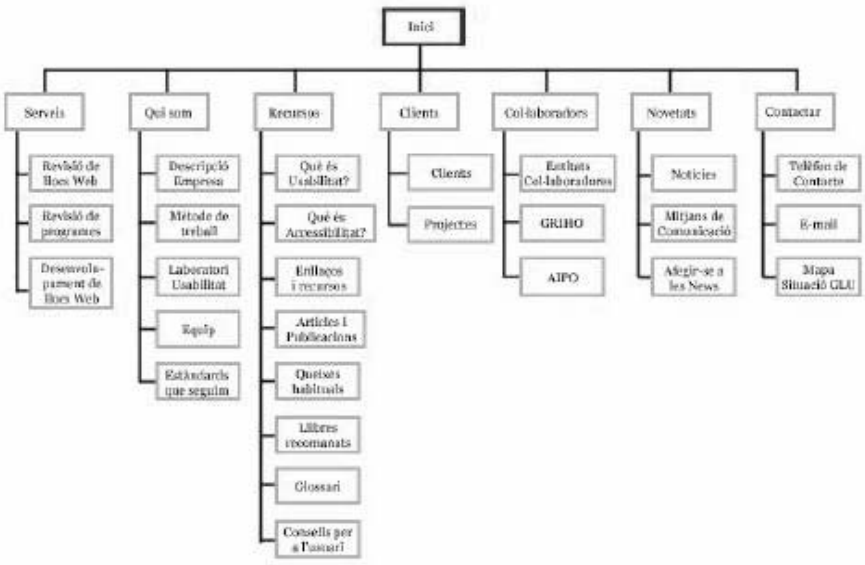

Fig. 5. Representación en topología jerárquica de la web de GLU

6.2.1 Identificación de objetos

En la elaboración de los contenidos de un sitio web primero debe procederse a la identificación de los objetos o unidades de información que contendrá la web en particular [13].

El proceso de identificación es un poco diferente si se trata de un sitio nuevo o de uno ya existente. El primero no cuenta con elementos preestablecidos y en el segundo una gran parte del esfuerzo se dedicará a mejorar la organización de la información actual.

Existen diferentes métodos para realizar esta actividad y el método a elegir dependerá, en gran parte, del tiempo y presupuesto disponibles. Los métodos más apropiados son [13]:

Grupos de Discusión (*Focus Group*)
Encuestas estructuradas (*Structured Survey*)
Encuestas de tanteo (*Exploratory Survey*)

6.2.2 Ordenación de tarjetas (*Card Sorting*)

Una vez identificados los objetos nos encontramos frente al reto de organizarlos de manera que sea útil y comprensible para los usuarios del sistema.

Aunque es cierto que realizando el análisis de la información pueden revelar algunas pistas, difícilmente podrá determinarse qué tópicos deben agruparse entre ellos, y menos aún imaginar cómo los usuarios los agruparían.

La dificultad de organizar el contenido procede de la falta de conocimiento sobre cómo usan los usuarios reales este contenido. Y sin este conocimiento cualquier intento de organizar dicha información no deja de ser un puro ejercicio teórico [27].

La técnica conocida como ordenación de tarjetas o *Card Sorting* [28] resulta altamente útil para conocer cómo los usuarios visualizan la organización de la información. El diseñador utiliza las aportaciones de los usuarios para decidir cómo deberá estructurarse la información en la interfaz.

Se trata de una técnica simple —fácil de entender y aplicar—, barata, rápida y que involucra a los usuarios, lo que está especialmente indicado cuando disponemos de una serie de ítems que precisen ser catalogados, así como para decidir la estructura organizativa de los sitios web.

Los pasos a seguir para implementar una ordenación de tarjetas son los siguientes:

- *Determinar la lista de tópicos*: Esta lista, que procede de la actividad anterior, no debería ser muy extensa (debe ser manejable), a la vez que debería ser comprensible para todos los participantes de la sesión. El evaluador no debe poner ningún tipo de indicación que pueda influenciar a los usuarios en su decisión, así como ningún tópico que induzca a la agrupación de términos (ejemplo: archivo, edición, FAQs).
- *Crear las tarjetas*: Cada tópico deberá escribirse en una tarjeta (papel, cartón) que ocasionalmente puede adjuntar algún tipo de explicación (aunque no es muy recomendable). Deberá, además, proporcionar tarjetas en blanco a los participantes.
- *Seleccionar a los participantes*: Los participantes preferentemente serán usuarios finales (gerentes y otros implicados no son usuarios finales) y deberemos estar seguros que representan fielmente a grupos de usuarios potenciales del sistema.
- Proceder con la/s *sesión/es de ordenación*: Cada sesión debe comenzar con una explicación del método y de los objetivos animando a todos los participantes a organizar las tarjetas y a etiquetar los grupos según sus criterios personales. El organizador de la sesión debe tomar nota de todo aquello que pueda resultar relevante para la evaluación final.

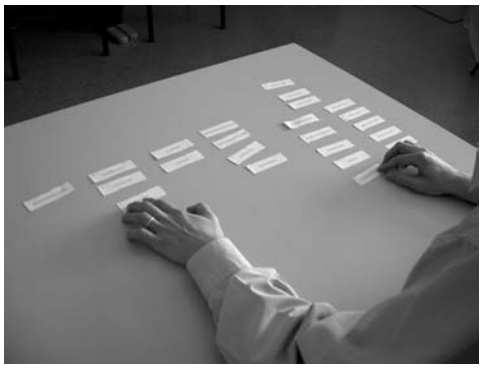

Fig. 6. Usuario realizando una ordenación de tarjetas para la web del congreso de Interacción 2004

- *Analizar las agrupaciones*: Una vez han concluido todos el evaluador deberá analizar todas las agrupaciones en un ejercicio de *análisis democrático* para identificar aquellas agrupaciones más frecuentes para poder decidir la estructura final. Existen programas informáticos específicos [29] que dan soporte a esta labor de síntesis.

6.2.3 Estructuras. Modelos o tipologías de organización de contenidos.

El estudio de los modelos de navegación permite determinar y entender cómo navegan los usuarios para definir cómo queremos que naveguen por nuestro sitio.

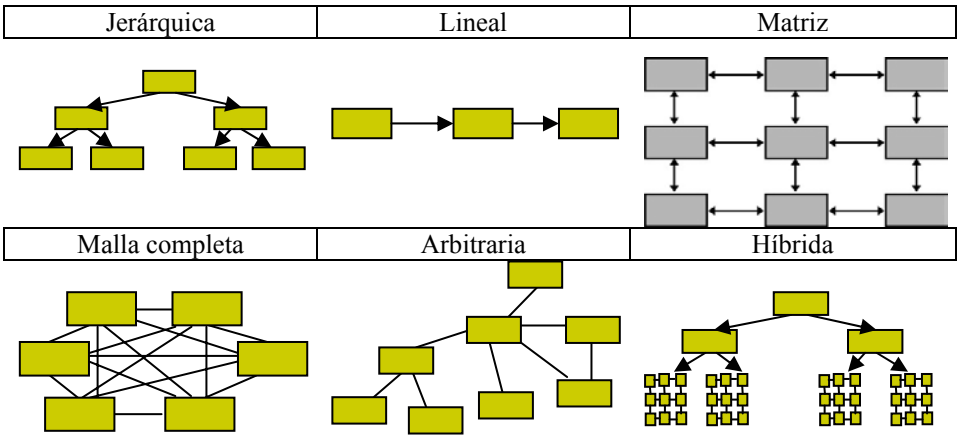

Fig. 7. Diferentes topologías o estructuras de modelos de navegación

Varias son las topologías, aunque casi todas, por no decir todas, parten de una página de inicio y de ahí dan acceso al resto del sitio. De todas formas, no hay que olvidar nunca que los navegantes pueden entrar en nuestro sitio a través de cualquier otra página.

La topología constituye la primera vía de definición acerca de cómo estarán enlazadas las diferentes páginas del sitio web.

6.2.4 Navegación

Una vez definida la topología organizada a partir de la estructura aportada por la ordenación de tarjetas completaremos el trabajo añadiendo atajos aportados por el análisis de tareas, desarrollaremos la barra de navegación y las señales de orientación.

Tipos de navegación

En los apartados siguientes recogemos los estilos de navegación más comunes que se encuentran actualmente en uso. Evidentemente, otros estilos de navegación pueden complementar los actuales o crearse de nuevos:

- *Navegación mínima*: La página de inicio puede enlazar con todas las páginas del sitio.
  Inicio. Mostrar los enlaces a las páginas de nivel inferior
  Inicio
  Productos: Conectores- electrónica- decoración
- *Migas*: Las migas muestran la jerarquía de páginas de la forma más directa desde la página inicial a la página actual. Es una representación de la estructura del sitio y la posición actual en la estructura.

Fig. 8. Migas de la web de GRIHO (http://griho.udl.es)

- *Categorías principales*: Una lista de las categorías principales ayuda a definir el ámbito del sitio al usuario y permite un acceso rápido a la información primaria.
  Ejemplo: http://www.nngroup.com
- *Esquema expandible*: Visualiza una lista de opciones y permite la expansión de la opción seleccionada.
  Ejemplo: Inicio
  Acerca de
  Productos
  Muebles
  Armarios
  Sillas
  Mesas
  .....
  Baños
  Iluminación
  Contactar

- *Barra de progreso*: Es especialmente interesante para resultados de motores de búsqueda

  Ejemplo: 1 2 3 4 **5** 6 7 8 9 | Anterior Siguiente
- *Mapa del sitio:* Visualiza la estructura del sitio. Obligatorio para todos los sitios, ya que refuerza un modelo mental del sitio.
- *Mecanismos de búsqueda*: Imprescindibles para sitios grandes y recomendables para cualquier sitio, ya que facilitan el acceso a la información. Además, el uso de estas herramientas se ha convertido en un estándar de la navegación web.
- Algunos indicadores de orientación.
  - Botón de inicio (Home Button). Es importante disponer siempre de un botón de inicio claro y explicito.
  - Usted está aquí. Importante marcar la página actual y que ésta no sea un enlace a ella misma.
  - Títulos de página. Procura títulos de página que se correspondan con el texto del enlace marcado.

6.3 Prototipado en el diseño

6.3.1 Maquetas digitales

Las maquetas digitales son representaciones de calidad en formato digital, por lo que se puede ya visualizar de una manera muy aproximada a la versión final el diseño de la página, colores, forma de la estructura de navegación, botones. Las maquetas de papel se perciben como menos pulidas y más conceptuales, mientras que los usuarios e implicados ven las maquetas digitales como versiones finales que no se pueden cambiar y, por tanto, es más adecuado utilizarlas en las fases finales del diseño.

Fig. 9. Maquetas digitales del sitio web de AIPO

6.3.2 Storyboard de uso

Son secuencias de pantallas que se centran en las posibles acciones o movimientos que el usuario puede realizar en el sitio.

Fig. 10. Ejemplo de Storyboard de uso de la web del centenario del CEL

Desarrollar Storyboards de uso puede requerir más tiempo que el uso de maquetas por el número de pantallas que tienen que realizarse, pero proporciona un material más comprensible para muchos usuarios.

Las pantallas realizadas se despliegan sobre un soporte y se visualizan enlaces que describen los caminos que puede recorrerse al interaccionar.

7 Implementación

Dos aspectos primordiales deben considerarse: el puramente tecnológico y el de los contenidos.

El primero incluye los lenguajes de programación a utilizar, las hojas de estilos e incluso el seguimiento de las normas WAI para disponer de páginas accesibles a personas con discapacidades. En este aspecto, el MPIU, como ya se ha mencionado, deja libertad al equipo de desarrollo, puesto que debemos recordar que el MPIU lo que hace es guiar en el proceso de desarrollo para la consecución de productos usables, sin entrar en la tecnología que hay que utilizar, puesto que ello dependerá de cada proyecto concreto.

En cuanto a los contenidos, debe tenerse en cuenta que escribir para la web es totalmente distinto a hacerlo para otro medio, puesto que el texto tiene que redactarse de manera que soporte las tareas y los objetivos del usuario y deberá adaptarse a la audiencia prevista. El proceso de escritura deberá, por tanto, estar orientado en función de los parámetros de lectura fácil, legibilidad, escaneabilidad (los navegantes no leen sino que escanean la información en busca de un contenido), la paginación, los enlaces y los principales hitos de cada página. Esta conducta a la hora de leer condicionara la manera en la que deberemos escribir para la web. Utilizaremos el modelo de escritura en pirámide invertida, que consiste en empezar cada página por la conclusión [30]. El usuario debe poder discernir al primer vistazo si lo que hay dentro le interesa lo suficiente para continuar leyendo.

8 Lanzamiento

Todo lo realizado anteriormente entra en escena definitivamente en esta fase. Es el momento en el que todo el trabajo se ha desarrollado y se pondrá a disposición del usuario final.

Cuando se trata de una aplicación web esta fase consta de tres partes bien diferenciadas:

- Prelanzamiento: Fase en la que el registro del dominio, el alojamiento y los tests de tareas, de código y de carga del sistema deben realizarse.
- Lanzamiento: Donde además de ubicar el sitio será de vital importancia realizar una buena promoción de éste, el proceso de alta en ciertos buscadores, gestionar enlaces en sitios afines y para controlar si la web es visitada o no debemos proveer de los mecanismos de control visitas.
- Poslanzamiento: No debemos confundir esta etapa con el mantenimiento pues su misión es muy distinta. Esta etapa en el paradigma web es muy determinante, ya que el objetivo principal es analizar el uso real de nuestro sitio por parte de los usuarios. Nos interesará saber si acceden a él, qué páginas son las más accedidas, que caminos son los más recorridos, desde dónde acceden nuestros usuarios, que motivaciones tienen, etc. Una técnica muy adecuada para evaluar esta información es la conocida con el nombre de *Logging* que, partiendo de la información que los usuarios van dejando en nuestro servidor, podemos encontrar respuesta a muchas de las preguntas anteriormente formuladas y con ello mejorar la usabilidad del sitio.

8 Métodos de evaluación generales

8.1 Evaluación heurística

La evaluación heurística fue desarrollada por Nielsen y Molich [31]. La evaluación heurística consiste en analizar la conformidad de la interfaz con unos principios reconocidos de usabilidad (heurística) mediante la inspección de varios evaluadores expertos. Se recomienda normalmente utilizar de tres a cinco evaluadores, ya que se consideran suficientes y la inclusión de un mayor número de los mismos no garantiza una mejora en el resultado.

La evaluación heurística se lleva a cabo realizando por parte de cada evaluador una revisión de la interfaz. Cuando se han terminado todas las evaluaciones se permite a los evaluadores comunicar los resultados y sintetizarlos. Este procedimiento es importante para asegurar evaluaciones independientes e imparciales de cada evaluador. Los resultados de la evaluación se pueden registrar como informes escritos de cada evaluador o haciendo que los evaluadores comuniquen verbalmente sus comentarios a un observador mientras inspeccionan la interfaz.

Fig. 11. Experta en usabilidad realizando la evaluación heurística de la web del CEL

Típicamente, una sesión de evaluación heurística debe durar de 1 a 2 horas, en caso de que se realice el test de una interfaz muy compleja se puede dividir en varias sesiones que aborden diferentes aspectos de la interfaz.

El resultado de una evaluación heurística es una lista de problemas de usabilidad que han sido encontrados en el diseño en opinión de cada evaluador.

8.2 Recorrido de la usabilidad plural

Este método es debido a Bias [32] y fue desarrollado en los laboratorios IBM. Comparte algunas características con los recorridos tradicionales, pero tiene algunas características propias que lo definen. Estas características son las siguientes:

1) Participantes: Este método se realiza con tres tipos de participantes, usuarios representativos, desarrolladores y expertos en usabilidad, que conforman todos los actores implicados en el producto.

2) Las pruebas se realizan con prototipos de papel u otros materiales utilizados en escenarios. Cada participante dispone de una copia del escenario de la tarea con datos que se puedan manipular

3) Todos los participantes han de asumir el papel de los usuarios, por tanto, aparte de los usuarios representativos que ya lo son, los desarrolladores y los expertos en usabilidad también lo deben asumir.

4) Los participantes deben escribir en cada panel del prototipo la acción que tomarán para seguir la tarea que están realizando, escribiendo las respuestas lo más detalladas posibles.

5) Una vez que todos los participantes han escrito las acciones que tomarían cuando interactuaban con cada panel, comienza el debate. En primer lugar deben hablar los usuarios representativos y una vez estos han expuesto completamente sus opiniones, hablan los desarrolladores y después los expertos en usabilidad.

8.3 Análisis de logs

¿Qué son los logs?

Cada visitante que accede a nuestro sitio web deja un rastro físico de su visita, estos rastros quedan almacenados en el servidor para poder ser consultados, filtrados y recopilados con la intención de obtener información de lo que está sucediendo.

El formato típico de un log es el de una cadena de texto que almacena la información sobre las peticiones que realiza el usuario en el servidor. Por tanto, cuando un navegante accede a nuestra página de inicio, se escribirá una línea por cada elemento que solicite, es decir, una línea al solicitar la página de inicio, otra para una imagen y así sucesivamente para cada elemento de la página solicitada.

Por tanto, efectuando un análisis de logs correcto dispondremos de una información muy valiosa que nos permitirá desde detectar problema de usabilidad hasta determinar perfiles de usuario o fidelizar clientes.

Características del análisis de logs

El análisis de logs es más económico que realizar un método de evaluación, ya que no se necesita la presencia física de los usuarios. Los datos se extraen en un formato estándar. Podemos comparar los datos entre meses, días, semanas o países. Se realiza sobre muestras amplias de usuarios y en un término amplio a lo largo en el tiempo. Se detecta fácilmente el uso verdadero del sitio (páginas más vistas, palabras más buscadas, etc).

¿Que necesito para hacer un análisis de logs?

Para hacer un análisis de logs es preciso disponer del fichero que contiene los logs que se encuentra en el servidor de nuestro sitio web y de una aplicación que analice este fichero. Las aplicaciones que permiten hacer análisis de logs son Analog [33] y WebTrends Log Analyzer [34], entre otras.

9 Actividades de protección

La Ingeniería del Software (IS) proporciona unas actividades de protección que dan soporte al proceso de desarrollo con la finalidad principal de obtener un producto de calidad demostrable [35]. Utilizaremos las actividades de protección, por tanto, como metodología para garantizar y planificar la usabilidad dentro del ciclo de vida. En este aspecto el MPIU utiliza:

- La Gestión de la Configuración (GC) en cuanto a que es una actividad de protección que permite gestionar el cambio a lo largo de todo el ciclo de vida. Será necesario identificar, organizar, controlar y documentar todas las modificaciones a las que son sometidos tanto los programas. La GC empieza en el mismo momento

en el que se inicia el desarrollo y sólo puede darse por finalizado en el momento en el que el producto es retirado del mercado [35].
- Todo ello envuelve al MPIU junto a la correspondiente planificación, que debe ser permanentemente revisada (en las RTFs, por ejemplo).

Toda nueva propuesta debe validarse de una forma u otra para probar que realmente sirve para aquello lo que ha sido pensada.

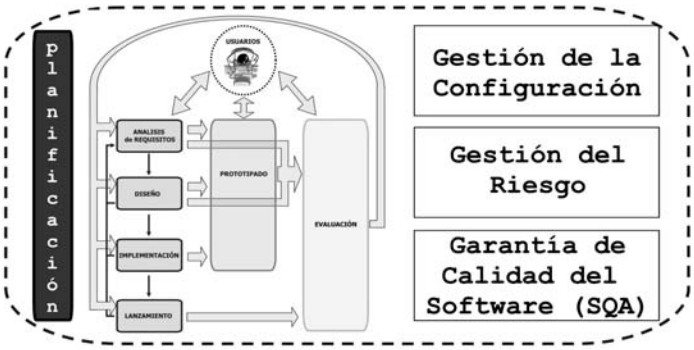

Fig. 12. Actividades de protección de la IS que envuelven el MPIU

La metodología incorpora un apoyo de la calidad del modelo basada en la evaluación del proyecto en cuestión y un esquema para mejorar el modelo basado en las mejoras adquiridas como resultado de su aplicación.

Para ello, nos basamos en actividades de protección de la ingeniería del software para dar soporte metodológico al proceso de desarrollo con el propósito principal de obtener un producto de calidad demostrable [35]. La manera como el MPIUA integra la ingeniería del software es extendiéndola hacia la gestión de la calidad de la usabilidad y la accesibilidad usando:
- La Gestión de la Configuración (GC) es el mecanismo de protección que permite gestionar el cambio durante toda la vida de servicio de un sistema interactivo. Será necesario identificar, organizar, controlar y documentar todas las modificaciones que se irán sucediendo. Esta GC empieza en el mismo instante en el que lo hace el desarrollo y sólo puede finalizarse cuando el producto es retirado del mercado [34].
- El Aseguramiento o la Garantía de la Calidad del Software (GCS) que sobre la base de diseñar planes específicos que consisten en la inspección, revisiones y evaluaciones realizadas, también, durante todo el ciclo de vida de la aplicación permitirá asegurar que cada parte del producto cumple perfectamente con la totalidad de los requisitos para la que fue pensada, diseñada y implementada. Esto permitirá asegurar la calidad efectiva del producto final. Las Reuniones Técnico Formales (RTFs) son una vía de probada efectividad para llevar a cabo esta revisión y mejora constante de la calidad del proyecto.

Este conjunto de actividades que *cubren* el MPIUA debe estar correctamente sincronizado con la planificación del proyecto, por lo que debe estar permanentemente en constante revisión (las RTFs son una buena estrategia para ello) certificando la correcta ejecución del proceso global para que se cumpla.

Una de las bases del MP, como se ha ido repitiendo, es la iteración de las actividades premiando aquellas facetas orientadas a la obtención de los requisitos tanto funcionales como de usabilidad y accesibilidad, los prototipos y sus posteriores evaluaciones contribuyen en cada ciclo del MP en un número de cambios, que deberán reflejarse donde les corresponda. Para poder visualizar la evolución de los cambios producidos, así como poder reflejar qué actividades del MP se están llevando a cabo y cuando el MP propone una particularización de la GC que consiste en una hoja de trabajo denominada Hoja de Trabajo de la Gestión de la Configuración (HTGC) que debe reflejar, en orden cronológico, todas las actividades realizadas.

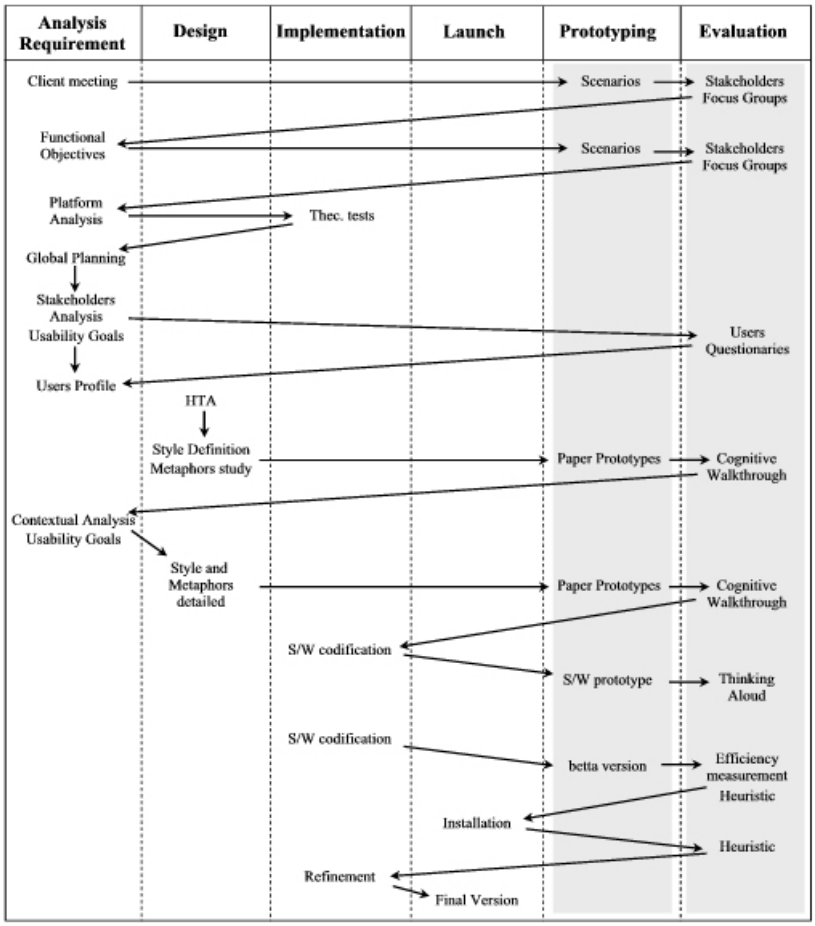

Fig. 13. Ejemplo de la HTGC correspondiente a un proyecto determinado

La figura 13 anterior muestra la HTGC de un determinado proyecto. Podemos observar que esta HT tiene tantas columnas como fases del MPIUA. La idea es que debajo se van indicando las tareas realizadas en cada una de estas fases y qué repercusión sobre las otras tienen.

En ella podemos ver a simple vista el número de prototipos y de evaluaciones realizadas y en que fase del proyecto se han llevado a cabo, así como acerca de que actividades notan repercusión como resultado de estas actividades.

10 Conclusión

Hemos descrito las ideas generales de nuestra propuesta metodológica para desarrollar aplicaciones interactivas basadas en la ingeniería del software y en la disciplina de la interacción persona-ordenador aplicadas al paradigma web. Se ha focalizado la explicación en aquellos aspectos más importantes a considerar en cada etapa del modelo. Asimismo, se han introducido las actividades de protección de la ingeniería del software que aseguran el éxito de un proyecto.

Con este trabajo no pretendemos únicamente proporcionar un método *usable* y comprensivo, sino que también pretendemos definir y resolver un Índice de Calidad de la Usabilidad (ICU) con la finalidad de que en la etapa del lanzamiento seamos capaces de indicar un nivel de usabilidad de dicha aplicación. Éste es uno de los objetivos más interesantes que pretendemos conseguir de nuestras investigaciones.

Referencias

[1] Lorés, J. 2002. *Introducción a la interacción persona-ordenador*. AIPO

[2] Granollers, T.; Lorés, J.; Perdrix, F. 2003. *Usability Engineering Process Model. Integration with Software Engineering*: Procs. of HCI-Intl'03, Creta (Grecia)

[3] Manchón, Eduardo. 2002. *Resultados encuesta perfil profesional AI y usabilidad iberoamericanos: España, Portugal y Latinoamérica.* http://www.ainda.info

[4] Garton, L.; Haythornthwaite, C.; Wellman, B. 1997. "Studying on line Socialnetworks", *Journal of Computer Mediated Communication*. Available and re-reviewed on April 2002. http://ascusc.org/jcmc/vol3/issue1/garton.html

[5] Brink, T.; Gergle D.; Wood, S.D. 2002. *Design Web Sites that Work: Usability for the Web,* Morgan-Kaufmann

[6] Nielsen, J. 1993. *Usability Engineering. AP Professional*, Boston, MA

[7] Mayhew D.J. 1999. *The Usability Engineering Lifecycle: A practitioner's Handbook for User Interface Design*, Morgan Kaufman

[8] Kotonya, G.; Sommerville I. 1997. *Requirements Engineering. Processes and Techniques*, JohnWiley

[9] Duran, T. 2002. *Un entorno metodológico de ingeniería de requisitos para sistemas de información*, Doctoral PhD. Universidad de Sevilla (España)

[10] Asociación para la Investigación de los Medios de Comunicación. Audiencia en Internet http://www.aimc.es/aimc/html/inter/net.html

[11] U.S. Census Bureau (http://www.census.gov/)

[12] Hix, Deborah and Harston; H. Rex. *Developing User Interfaces: Ensuring Usability Through Product & Process*, John Wiley and Sons
[13] Fuccella, J. "Using user centered design methods to create and design usable Web sites", *Proceedings of the 15th annual ACM international conference on Computer documentation*
[14] Bevan, N.; Serco Usability Services Research Manager, http://www.usability.serco.com/trump/methods/recommended/stakeholder.htm
[15] Wilson, R. F. 2001. *Planning Your Internet Marketing Strategy: A Doctor Ebiz Guide*, John Wiley & Sons
[16] Sterne, J. 2002. *Web Metrics: Proven Methods for Measuring Web Site Success*, John Wiley & Sons
[17] Nielsen, N. 1994. *Usability Engineering*, Morgan-Kaufman
[18] Pouloudi, A. 1999. "Stakeholder Analysis as a Front-End to Knowledge Elicitation", *At & Society*, XI, 122-137
[19] Schneider, G.; Winters, J.P. 1998. "Applying Use Cases: A Practical Guide: Reading", *MA: Addison-Wesley*
[20] Constantine L.L. 1995. "Essential Modeling: Use Cases for user Interfaces": *ACM Interactions*
[21] Wharton C. et al. 1994. "The Cognitive Walkthrough Method: a Practitioner's Guide", *Usability Inspection Methods*. John Wiley & Sons, Nueva York, 105-140
[22] Annet, J.; Duncan, K. 1967. "Task analysis and training in design", *Occupational Psychology*, 41
[23] Paternó F. 2000. *Model–based design and evaluation of interactive application. Springer–Verlag*, http://giove.cnuce.cnr.it/ConcurTaskTrees.html
[24] Gerrit, C; Van der Veer, Chris Stary. "Task analysis meets prototyping: seeking seamless UI-development". *Tutorial Session of Conference on Human Factors and Computing Systems CHI*, 99
[25] Rosenfeld, L.; Morville, P. 1998. *Information Architecture for the World Wide Web*, Ed. O'Really
[26] Rosenfeld, L. 2002. *The Emergence of Information Architecture*, Yggdrasil, Sandefjord, Norge
[27] Roberston J. 2001. "Intranet Design Series: Information design using card sorting", *Information & Design*.
[28] McDonald, J.; Dearholt, D.; Paap, K. and Schvaneveldt, R. 1986. "A Formal Interface Design Methodology Based on User Knowledge". *Procs. of CHI'86*, 285-290
[29] Toro, J.A. *CardZort: computer application that runs card sorting exercises. Adailable at http://condor.depaul.edu/~jtoro/cardzort/cardzort.htm*
[30] Nielsen, J. 1996. *Inverted Pyramids in Cyberspace: Jakob Nielsen's Alertbox for June 1996*, www.useit.com/alertbox/9606.html
[31] Nielson, J. & Molich, 1990. *Heuristic evaluation of user interfaces*
[32] Bias, R.; Rietmeyer, P.B. 1995. *Usability Support Inside and Out*. Interactions ACM Press
[33] Analog log file analyser tool available at http://www.analog.cx
[34] Web Analyser Series tool available at http://www.analog.cx
[35] Pressman, R. 2001. *Software Engineering: A Practitioner's Approach*, 5ª edición. Mc Graw-Hill

Herramientas Avanzadas para la Producción de Interfaces de Usuario Basados en Tecnología Web [1]

Jaime Gómez

Grupo de Ingeniería Web y Almacenes de Datos
Departamento de Lenguajes y Sistemas Informáticos
Universidad de Alicante
03690 - ALICANTE
http://www.dlsi.ua.es/iwad/
jgomez@dlsi.ua.es

Resumen. El desarrollo de aplicaciones web es una tarea compleja que requiere del uso de una amplia variedad de conocimientos de tecnología, organización y comunicación. Los sistemas de información basados en web son mucho más complejos que las aplicaciones tradicionales debido a que han de construirse sobre componentes tecnológicos que se encuentran en continua evolución, han de encajar en la infraestructura existente en la empresa y el interface de usuario debe de ofrecer un nivel de calidad hasta ahora no exigido. Este artículo describe los fundamentos ingenieriles de VisualWADE, una herramienta CASE que automatiza la producción de interfaces de usuario de aplicaciones web. VisualWADE es una aproximación dirigida por el modelo (model-driven) que hace énfasis en el análisis de requisitos, el diseño de alto nivel, y el prototipado rápido. De esta forma una aplicación evoluciona de forma suave desde el primer prototipo al producto final y su mantenimiento es consecuencia natural del desarrollo. Conforme aparecen nuevos requisitos o cambios en los actuales, simplemente hace falta revisar el modelo conceptual, establecer los cambios y regenerar la implementación.

1 Introducción

La rápida evolución de Internet en general y del WWW en particular ha propiciado en los últimos años la investigación intensiva en el campo del modelado conceptual de aplicaciones web. Esta circunstancia ha dado lugar a la aparición de una nueva línea de investigación dentro de la Ingeniería del Software conocida como Ingeniería Web (Web Engineering). En este contexto se han propuesto distintos métodos, lenguajes, herramientas y patrones de siseño [15, 17] de modelado hipermedial. Algunos de los más relevantes estudiados hasta el momento son HDM [9], HDM-lite [7], WebML [4], OOHDM [19], RMM [12], ADM [1, 10, 14], Strudel [6] u OO-H [11, 3,2]. Estos métodos se centran en su mayor parte en la definición de los aspectos de navegación y presentación respecto a la semántica de los modelos para capturar la problemática del

[1] Este artículo ha sido patrocinado por el proyecto MCYT ref TIC2001-3530-C02-02

desarrollo en entorno web. Sin embargo, son pocas las propuestas que han intentado aplicar su método en la resolución de casos reales complejos para comprobar la validez y eficacia de sus constructores de modelado. Y mucho menores han sido los intentos (existosos o no) de construir herramientas de propósito específico para la Ingeniería Web como por ejemplo, WebRatio [21] desarrollada en el marco de un proyecto europeo de V programa marco en el Politécnico de Milano bajo la dirección técnica del Prof. Piero Fraternali.

Este artículo describe los fundamentos ingenieriles de VisualWADE una herramienta CASE para el diseño y producción automática de interfaces de usuario de aplicaciones web que esta basada en el método OO-H. La idea que subyace en VisualWADE consiste en que con una combinación adecuada de conceptos simples (constructores de modelado), el diseñador puede modelar y automáticamente generar cualquier tipo de sistema basado en web, desde un portal e intranet de una compañía hasta un sitio web seguro de comercio electrónico. VisualWADE es una aproximación dirigida por el modelo (model-driven) que hace énfasis en el análisis de requisitos, el diseño de alto nivel, y el prototipado rápido. De esta forma una aplicación evoluciona de forma suave desde el primer prototipo al producto final y su mantenimiento es consecuencia natural de desarrollo. Conforme aparecen nuevos requisitos o cambios en los actuales, simplemente hace falta revisar el modelo conceptual establecer los cambios y regenerar la implementación.

VisualWADE explota un conjunto de conceptos bien conocidos para capturar la complejidad de las aplicaciones web reales, el método subyacente OOH (object oriented hypermedia) proporciona constructores de modelado específicos para modelar espacios de navegación basándose en una notación compatible con UML. La especificación capturada se compila haciendo uso de técnicas de generación basadas en modelos y produce como resultado una presentación por defecto que puede refinarse desde la propia herramienta para obtener la apariencia final de interface deseada con el consiguiente incremento de productividad de desarrollo.

El artículo esta organizado como sigue: la sección 2 proporciona una breve descripción del método OO-H para familiarizar al lector con la notación de modelado. La sección 3 proporciona una descripción detallada de VisualWADE. Primeramente se presenta cómo se especifica un modelo de dominio, introduciendo los constructores de modelado para un sistema de correo eléctronico basado en web. A continuación se describe la forma en la que se especifica la vista de navegación y el proceso de compilación hasta llegar a la presentación por defecto. La descripción de la funcionalidad necesaria para refinar el interface generado y la evaluación de la consistencia entre los distintos modelos terminan esta sección. Finalmente, la sección 4 presenta las conclusiones derivadas del uso de VisualWADE durante estós últimos 2 años y los trabajos en perspectiva.

2 El método OO-H

El método OO-H (Object Oriented Hypermedia) es un método genérico que está basado en el paradigma orientado a objetos y que ofrece al diseñador una notación y una semántica específica para el desarrollo de interfaces basados en web, su conexión

con funciones CRUD[2] sencillas o con módulos de lógica preexistentes (legacy software).

OO-H define un conjunto de diagramas, técnicas y herramientas que juntos constituyen una propuesta completa para el modelado de interfaces web. El método incluye:

- Proceso de Diseño
- Diagrama de Acceso Navegacional (DAN)
- Diagrama de Presentación Abstracto (DPA)
- Una herramienta CASE que soporta y automatiza gran parte del proceso de desarrollo.

Con OO-H una aplicación tradicional de escritorio puede ser extendida añadiendo dos nuevas vistas (diagramas) complementarias a las vistas de estructura (diagrama de clases) y comportamiento (diagramas de interacción y de transición de estados). La primera de ellas es el DAN con la cual se puede especificar una vista de navegación, la segunda, el DPA captura conceptos relacionados con los detalles de presentación del interface final.

El DAN enriquece un modelo de dominio (constituido por el diagrama de clases y los casos de uso [18]) con características de navegación e interacción. También define restricciones sobre la navegación y la información que se muestra por cada clase. Para ello, OO-H hace uso del lenguaje de restricción de objetos proporcionado por UML (OCL [16, 20]) . De esta forma se consigue obtener un diagrama de navegación suficientemente preciso. Por otro lado, la definición de páginas abstractas (independientes de la implementación) está basada en un conjunto de plantillas XML que capturan las propiedades relevantes respecto a los aspectos de apariencia del interface en construcción.

A continuación veremos con un mayor nivel de detalle estos diagramas.

2.2 El diagrama de acceso navegacional

El modelo de navegación se captura a través de uno o más DAN´s. El diseñador deberá construir tantos NAD´s como escenarios de navegación quiera representar. El diagrama está basado en cuatro tipos de constructores de modelado: clases, destinos y enlaces navegacionales, y agrupadores de enlaces. Además, cuando se está definiendo la estructura de navegación es necesario tener en cuenta aspecto relacionados con el comportamiento de la navegación, la selección de población de objetos navegables, el orden en que serán navegados, y la forma en que se visualizarán las poblaciones de objetos navegadas, entre otros. Estas características se capturan según diferentes tipo de patrones de navegación y filtros asociados sobre los enlaces. Los constructores de modelado de OO-H son:

[2] Create, Read, Update, Delete

- **Clases Navegacionales** (CN): se corresponden con clases del dominio cuya visibilidad de atributos y métodos se restringe de acuerdo a los permisos de acceso de usuario y/o requisitos de navegación.
- **Destinos Navegacionales** (DN): agrupan elementos del modelo que trabajan en la cobertura de un requisito navegacional genérico.
- **Enlaces Navegacionales** (EN): definen los caminos de navegación que un usuario puede seguir a través del sistema. Pueden tener asociados patrones y/o filtros de navegación para enriquecer la semántica del modelo de navegación. OO-H propone 6 tipos de enlaces:
    - **Ei** (enlace simple) define caminos de navegación entre elementos del modelo.
    - **Er** (enlace punto de entrada) punto de inicio de la navegación por cada destino navegacional.
    - **Ex** (enlace punto de salida) para salir del espacio de navegación del sistema.
    - **Es** (Enlace de servicio) muestras los servicios disponibles en la vista de navegación. También capturan la forma en la cual se introducen los parámetros para la invocación del método asociado al servicio y la posibilidad de representar por donde continua la navegación una vez ejecutado el servicio.
- **Agrupadores de enlaces**: proporcionan al usuario con nuevas formas de acceso a la información. En este caso, como un mecanismo de abstracción para agrupar enlaces de navegación.

Como ya comentamos anteriormente los **filtros de navegación** se capturan mediante expresiones OCL. OO-H distingue entre dos clases de filtros; filtros en origen (o precondiciones) y filtro en destino. Los primeros capturan restricciones de navegación en origen, es decir, condiciones aplicadas sobre el/los objeto/s origen de un enlace, si estas condiciones no se cumplen se inhibe la navegación. Los segundos, son restricciones que acotan la población de objetos que se visualizará.

Los **patrones de navegación** en OO-H caracterizan la forma en la cual se visualiza la información que se va recorriendo en un camino de navegación, la mayoría de los patrones propuestos por OO-H son bien conocidos puesto que provienen del campo de investigación del multimedia. Entre otros se encuentran: Indexación, mostrar todos y tour guiado.

Además de los filtros y patrones de navegación, OO-H asocia información complementaria sobre los enlaces para capturar propiedades avanzadas de navegación. Entre éstas tenemos:

- **Visualización** (mostrar en origen | mostrar en destino): cuando un enlace conecta dos clases navegacionales. Con esta propiedad se puede indicar si se desea que la información de ambas clases aparezca en un mismo origen (misma página abstracta) o con un salto de navegación implícito (distinta página abstracta).
- **Interacción de Usuario** (manual | automático): esta propiedad permite que un enlace se active o no de forma automática al alcanzar una determinada

clase navegacional. Algunas veces es útil que el usuario no se vea obligado a hacer un click en un enlace para acceder a la información destino.

- **Ambito de Aplicación** (simple | múltiple | universal): Esta propiedad establece el número de objetos que un determinado link abarca en origen cuando es atravesado. Tal origen puede referirse a un objeto simple, a un conjunto de objetos (múltiple) o a todo el conjunto de objetos que se visualizen en la vista actual (universal)

2.3 El diagrama de presentación abstracta

El DPA permite el refinamiento de la estructura de interfaz y de los rasgos de visualización capturados en las etapas previas del modelo. Concretamente, los patrones capturados a nivel de DAN, junto con las características de objetos y atributos, proporcionan la información mínima necesaria para generar de forma automática un DPA por defecto.

En OO-H se adopta una aproximación basada en plantillas [1, 6, 7, 14] para la especificación tanto de la apariencia visual como de la estructura de página de la interfaz hipermedial. El uso de plantillas favorece la reutilización de las experiencias de diseño y facilita la consistencia tanto dentro de los distintos módulos que componen la aplicación como entre distintas aplicaciones.

OO-H define cinco tipos de plantillas, expresadas como documentos XML (eXtensible Markup Language) [5, 13]. XML permite al diseñador mantener información acerca del significado semántico de los distintos datos capturados en OO-H. Para ello especifica un marco estandarizado dentro del cual definir las etiquetas necesarias para capturar dicha información. Las páginas resultantes pueden ser directamente visualizadas o traducidas a cualquier otro lenguaje para su publicación. Con esta aproximación, la adición de nuevos tipos de plantillas es muy sencilla.

Las etiquetas y la estructura del documento se han definido mediante un conjunto de DTD's (Document Type Definition), uno para cada tipo de plantilla. El DTD especifica el conjunto de reglas que debe cumplir un documento XML para ser considerado válido. Como las páginas abstractas son documentos XML, el DTD especificará las etiquetas que puede contener ese tipo de página, las anidaciones válidas de etiquetas, sus atributos y los valores permitidos. También contiene otro tipo de información, como son instrucciones de proceso, comentarios, declaración de tipo de documento etc. Los tipos de plantilla definidos hasta el momento en OO-H son:

- tStruct: estructuran la información que debe aparecer en la página materializada.
- tStyle: definen los rasgos de visualización como son el emplazamiento físico de los elementos, la tipografía o la paleta de color.
- tForm: definen las estructuras de información que debe introducir el usuario para interactuar con el sistema.
- tFunction: capturan funcionalidad de cliente mediante funciones definidas
- basándose en el modelo DOM (Document Object Model) [5].

- tWindow: su uso implica la existencia de dos o más vistas de la información activas en un mismo instante de tiempo.

3 VisualWADE

3.1 Modelado del dominio

El punto de partida para abordar el diseño de una aplicación web es el modelo de dominio representado con un diagrama de clases (ver Figura 1). La notación de este diagrama está basada en UML con lo cual es bastante intuitivo su uso por parte de un diseñador familizarizado con técnicas de análisis y diseño orientado a objetos. La aplicación te asiste en todo el proceso de edición mediante un simple y a la vez potente e intuitivo interface gráfico. La creación de clases, atributos, relaciones jerarquías de herencia, … se realizan con un simple click. La disponibilidad en el entorno de botones de zoom y vistas generalizadas ayudan a organizar y manejar diagramas complejos compuestos por varias clases.

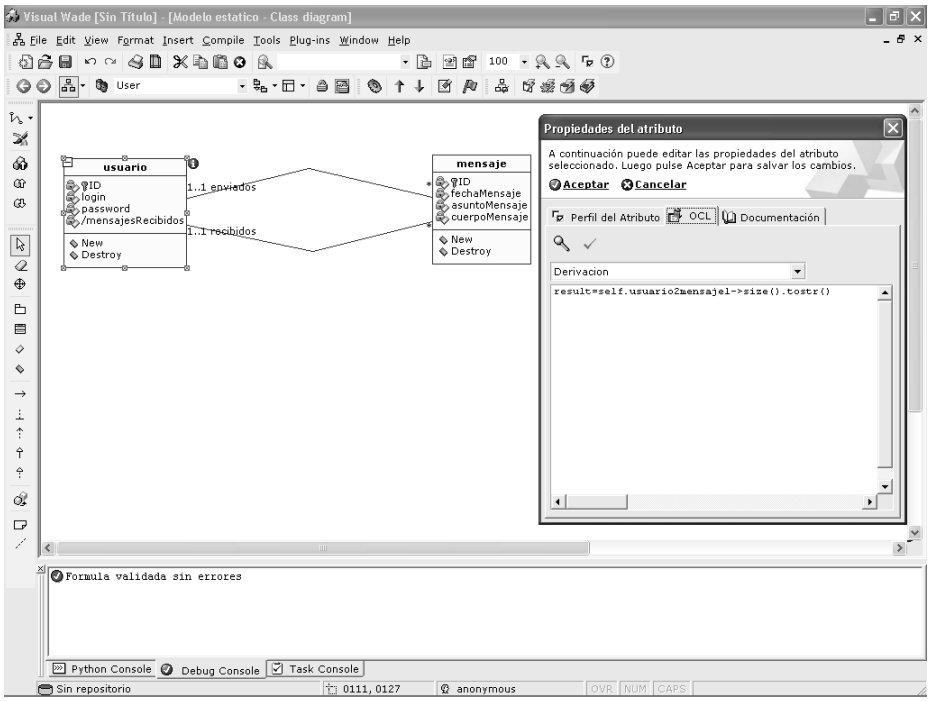

Fig. 1. Diagrama de clases en VisualWADE

En la Figura 1 podemos observar el diagrama de clases correspondiente a un sistema de correo electrónico basado en web (`webMail`). Tenemos la clase `usuario` y la clase `mensaje` y entre ellas relaciones de asociación para capturar la información correspondiente a los mensajes `enviados` y `recibidos` por parte de un usuario.

Además podemos hacer uso en el entorno de derivaciones. Podemos estereotipar atributos como `derived` y asociarles una fórmula OCL que especifique la obtención del valor de ese atributo. Este es el caso del atributo `mensajesRecibidos` de la clase usuario, cuyo valor se obtiene navegando por el rol `usuario2mensaje1`, calculando el numero de instancias de la clase mensaje (aplicación de la función `size()`) y finalmente pasándolo a cadena (aplicación de la función `str()`). Una de las características del entorno es que integra un compilador de OCL que además permite validar sintáctica y semánticamente las fórmulas introducidas.

3.2 Modelado del Espacio de Navegación

Una vista de navegación refleja la forma en la que se accede al contenido de información del modelo de dominio y a los servicios proporcionados por las clases. En la Figura 2 el lector puede observar una vista parcial del diagrama DAN para un usuario del sistema.

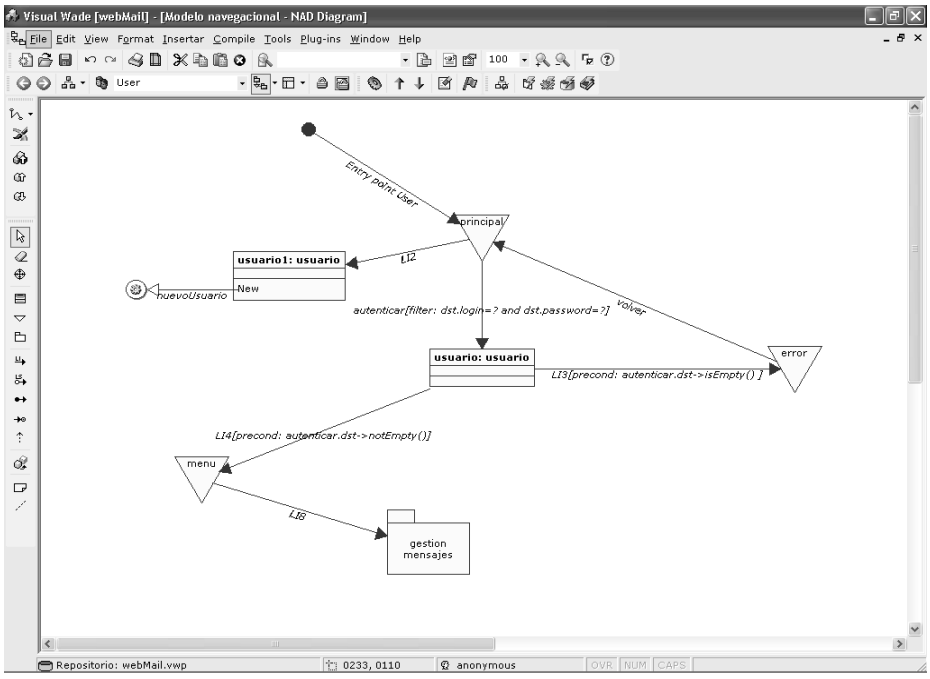

Fig. 2. Diagrama Navegacional en VisualWADE

El punto de entrada es un agrupador de enlaces principal (que representa una página abstracta de inicio). Desde `principal` se puede navegar a la clase navegacional `usuario` a través del enlace `autenticar`. La activación del enlace `autenticar` requiere que el usuario proporcione información respecto a su login (`dst.login`) y su password (`dst.password`) con el objetivo de comprobar si es un usuario válido en el sistema. Dependiendo de si este proceso de validación tiene éxito o no las precondiciones asociadas a los enlaces `LI2` y `LI3` determinan el enlace que se activará y por tanto el agrupador de enlace destino (`menu` o `error`). Las funciones OCL `notEmpty()`, `isEmpty()` proporcionan la información necesaria para conocer si un usuario está o no registrado en el sistema. Además desde el agrupador `principal` y a través del enlace `LI2` se accede a una clase navegacional `usuario` que ofrece el servicio `nuevoUsuario` (primitiva proporcionada por el contructor de la clase). Finalmente, indicar que desde el agrupador `menu` se continua la navegación por el destino navegacional `gestión mensajes` mientras que con el agrupador `error` se vuelve a la página de inicio `principal`.

La barra de herramientas de la parte izquierda contiene todos los constructores de modelado que se pueden insertar en un NAD. Haciendo doble click se accede a las propiedades internas de cada elemento. Los enlaces permiten construir los caminos de navegación indicando si la información se mostrará en la misma o distinta página abstracta así como el conjunto de operaciones que están disponibles para su activación. El entorno visual, además incluye muchas funciones muy úitles como copiar/pegar, hacer/deshacer y zoom in/zoom out, búsqueda rápida de elementos y finalmente reglas para comprobar la validez del modelo (model checking) y mensajes de aviso aspectos que se presentan más adelante.

3.3 Modelado de la Presentación

Una de las características más reconocidas del entorno y que hasta ahora no se ha incluido en ninguna de las herramientas CASE actuales es la posibilidad de compilar el diagrama de navegación, generar el conjunto de páginas abstractas mínimo que cumple con las especificaciones de navegación y ofrecer un diagrama abstracto de presentación para modificar de forma visual la representación final de interface. Los aspectos que se pueden refinar son los correspondientes con las propiedades de estilos, localización, colores, inclusión de componentes finales (imágenes, presentaciones flash,…) tal y como podría hacerse con cualquier herramienta de autor (ej. frontpage, dreamweaver) pero con la posibilidad de mantener en todo momento la consistencia con las vistas anteriores (diagrama de clases y de navegación). De esta forma se consigue un desarrollo incremental de la aplicación web.

La Figura 3 muestra el diagrama de presentación abstracto resultado de compilar el DAN de la Figura 2. Se observan varias zonas, en la parte derecha tenemos el `visor de páginas` que contiene las páginas abstractas que se han generado ordenadas alfabéticamente (`error`, `mensajes`, `menu`, `principal`, `usuario`, `usuario1`). En la parte derecha tenemos el `área de edición` donde se visualiza el contenido de las páginas abstractas. En este caso se está mostrando la

información de la página `principal`. Como resultado del proceso de compilación se ha generado un formulario con los campos `login` y `password` y el botón correspondiente para ejecutar la acción (`OK`). Además se observa el enlace `usuario1` que habilita la navegación para dar de alta un usuario.

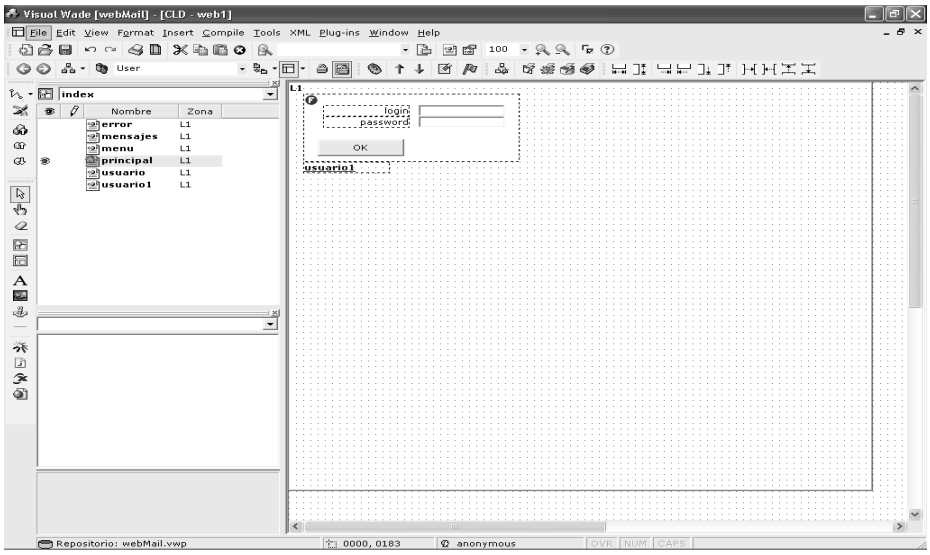

Fig. 3. Diagrama de Presentación en VisualWADE

El entorno ofrece la posibilidad de animar el interface generado con el objetivo de comprobar que efectivamente se está diseñando la solución requerida. Cuando la herramienta de animación está activa los enlaces y botones del interface resultante se vuelven sensibles a los clicks de ratón produciéndose en el entorno el salto de navegación correpondiente. Mientras se está animando un modelo es posible seleccionar la herramienta de edición para modificar cualquier propiedad de los elementos del interface. En la Figura 4, se pueden observar estas características.

La Figura 4 es el resultado de haber activado la herramienta de animación, haber pichado en el enlace `usuario1` de la Figura 3 y finalmente haber usado la herramienta de edición para modificar el aspecto de esta página. Se puede observar que entre estos refinamientos se ha incluido un texto de ayuda y se han cambiado las posiciones tamaño y color del formulario de creación de usuarios.

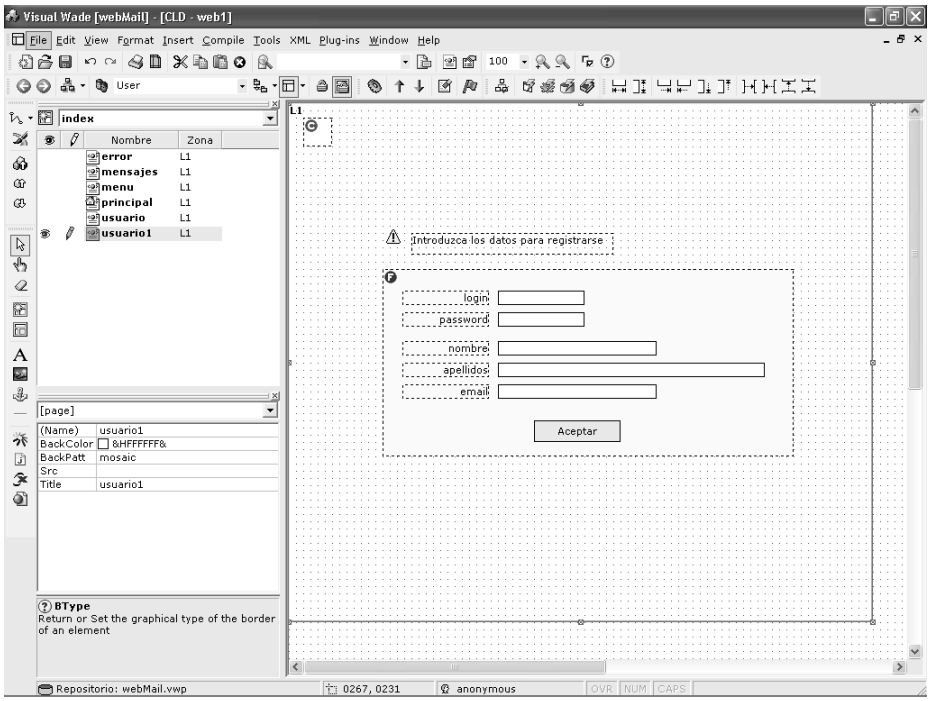

Fig. 4. Refinamiento y Animación en el DPA

3.4 Comprobación Automática de Modelos

VisualWADE incluye una potente funcionalidad para comprobar de forma automática la validez del modelo que se está diseñando. De esta forma se verifica tempranamente la especificación capturada con el consiguiente ahorro de tiempo en el proceso de generación de código. El proceso de comprobación se produce a tres niveles:

- Comprobación del modelo (model checking): esta función valida la construcción de las vistas de estructura y navegación y presenta los problemas detectados de consistencia así como lo que hay que hacer para arreglarlos.
- Representación del modelo (mapping checking): esta función comprueba si los elementos del diagrama de clases están correctamente representados en la base de datos generada debido a cambios en la especificación conceptual o el renombrado de atributos de las clases.
- Representación de la presentación (presentation checking): Este proceso comprueba si la representación de las páginas abstractas se corresponde con la especificación capturada en el diagrama de navegación proporcionando los avisos correspondientes en el entorno para corregir el error.

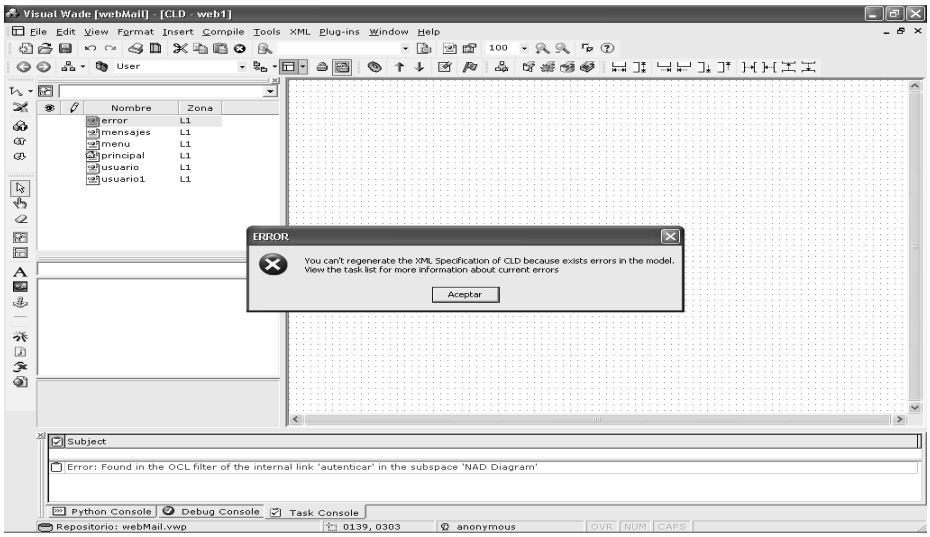

Fig. 5. Comprobación del Modelo en VisualWADE

En la Figura 5 se puede observar un aviso del entorno en el proceso de comprobación del modelo. Concretamente, se está informando que no se puede generar el DPA por defecto debido a que se ha encontrado un error en la fórmula OCL asociada al enlace `autenticar` de la vista NAD. Observando con más detalle este error comprobaremos que la fórmula hace referencia a atributos que no han sido declarados en el diagrama de clases.

Con todas estas funcionalidades presentadas creemos que VisualWADE constituye una apuesta seria como herramienta avanzada para el diseño y generación automática de interfaces de usuario de aplicaciones web. VisualWADE y su notación subyacente (OO-H), han sido el resultado de 4 años intensos de investigación en el campo de la Ingeniería Web en la Universidad de Alicante.

4 Conclusiones

Los desarrolladores web deben mejorar la productividad y calidad para satisfacer las necesidades del mercado y reducir los tiempos de entrega y costes. Lamentablemente, los métodos y técnicas estándar que se utilizan para el desarrollo web todavía presentan varias deficiencias:

- Los modelos y herramientas de análisis y diseño carecen de conceptos adecuados para capturar las propiedades de desarrollo en este tipo de entornos como consecuencia la mayoría del código de la aplicación es escrito a mano y difícil de reutilizar.
- La documentación es escasa y de baja calidad, especialmente para el interface de usuario.

- Los costes y el tiempo de desarrollo son difíciles de predecir y rápidamente escapan de nuestro control durantes las fases de mantenimiento y evolución de la aplicación.

Todavía las herramientas de desarrollo web limitan su campo a la fase de implementación o están centradas en soluciones verticales concretas (a menudo basadas en componentes) que son difíciles de personalizar y encajar juntas.

Desde nuestra experiencia, el uso de técnicas CAWE (Computer-Aided Web Engineering) en el diseño/implementación de aplicaciones web proporciona los siguientes beneficios:

- *Incremento de la productividad*: hasta casi un 80% gracias a los potentes compiladores de modelos que generan el código de forma automática a partir de la especificación conceptual.
- *Prototipado rápido*: de uno a tres días son suficientes para entregar un prototipo de alta calidad. De esta forma el "time to market" se ve mejorado considerablemente.
- *Satisfacción del cliente*: el intercambio de información con el cliente es mucho más ágil y la respuesta ante cambios exigidos puede mostrarse en poco tiempo gracias a prototipado rápido. De esta forma se consigue transmitir confianza al cliente captando su fidelidad.
- *Disminución de la curva de aprendizaje*: en cinco días un diseñador puede producir aplicaciones web completas de alta calidad sin necesidad de conocimientos de programación.
- *Esfuerzo repartido a lo largo del ciclo de vida*: la posibilidad de dedicar más tiempo al usuario se traduce en un incremento de la calidad de la aplicación respecto a los requisitos del sistema.

Como consecuencia de estas reflexiones, y de nuestra experiencia a lo largo de los últimos 4 años, creemos que en un futuro no muy lejano empezaran a proliferar herramientas de desarrollo que ofrezcan al mismo tiempo:

- La posibilidad de modelar visualmente una aplicación web haciendo uso de constructores de modelado de alto nivel, junto con compiladores de modelos que faciliten la generación automática de código portable y eficiente a partir de la especificación conceptual capturada.
- Soporte de todo el ciclo de vida de la aplicación, incluyendo, análisis, prototipado, diseño, despliegue y mantenimiento.
- Integración de módulos de lógica preexistentes (legacy software) en un proceso de desarrollo optimizado.

VisualWADE, en su estado actual proporciona alguna de estas demandas y actualmente se está utilizando en la resolución de casos reales complejos en organismos como Diputación de Alicante, Caja de Ahorros del Mediterráneo y el Colegio de Ingenieros Técnicos Industriales de la Provincia de Alicante.

VisualWADE y OO-H están en continua evolución, nuestro próximo reto es integrar en el entorno el nuevo paradigma de desarrollo basado en servicios web. De esta forma aplicaciones desarrolladas con VisualWADE podrían fácilmente integrarse con servicios web de terceros e incluso ofertar la funcionalidad de navegación como un servicio web propio.

Referencias

[1] P. Atzeni, G. Mecca, and P. Merialdo. Design and Maintenance of Data-Intensive Web Sites. In Advances in Database Technology - EDTB'98, pages 436–449, 03 1998.

[2] C. Cachero. The OO-HMethod Pattern Catalog. Technical report, Universidad de Alicante, 12 1999.

[3] C. Cachero, J. Gómez, and O. Pastor. Object-Oriented Conceptual Modeling of Web Application Interfaces: the OO-HMethod Presentation Abstract Model. In EC-Web 2000. 1st International Conference on Electronic Commerce and Web Technologies. Springer-Verlag. Lecture Notes in Computer Science, 09 2000.

[4] S. Ceri, P. Fraternali, and A. Bongio. Web Modeling Language (WebML): a modeling language for designing Web sites. In Position Paper, Web Engineering Workshop, WWW9, 05 2000.

[5] eXtensible Markup Language (XML). http://www.w3.org/XML/.

[6] F. M. Fernández, D. Florescu, J. Kang, A. Levy, and D. Suciu. Catching the Boat with Strudel: Experiences with a Web-Site Management System. In Proceedings of ACM SIGMOD International conference on Management of data, pages 414–425, 10 1998.

[7] P. Fraternali and P. Paolini. A Conceptual Model and a Tool Environment for Developing more Scalable, Dynamic, and Customizable Web Applications. In Advances in Database Technology - EDBT'98, pages 421–435, 1998.

[8] E. Gamma, R. Helm, R. Johnson, and J. Vlissides. Design Patterns. Elements of Reusable Object-Oriented Software. Addison Wesley, 1995.

[9] F. Garzotto and P. Paolini. HDM A Model-Based Approach to Hypertext Application Design. ACM Transactions on Information Systems (TOIS), 11(1):1–26, 01 1993.

[10] D.M. Germán and D.D. Cowan. Three Hypermedia Design Patterns. In Proceedings of the HT99 Workshop on Hypermedia Development: Design Patterns in Hypermedia, 02 1999.

[11] J. Gómez, C. Cachero, and O. Pastor. Conceptual Modeling of Device-Independent Web Applications. IEEE Multimedia 8(2): 20-32. 2001.

[12] T. Isakowitz, E. A. Stohr, and V. Balasubramanian. RMM: A Methodology for Structured Hypermedia Design. CACM: Communications of the ACM., pages 34–44, 08 1995.

[13] S. McGrath. XML by Example. Building ecommerce Applications. Prentice Hall, 1998.

[14] G. Mecca, P. Merialdo, P. Atzeni, and V. Crescenzi. The ARANEUS Guide toWeb-Site Development. Technical report, Universidad de Roma, 03 1999.

[15] M.Nanard, J. Nanard, and P. Kahn. Pushing Reuse in Hypermedia Design: Golden Rules, Design Patterns and Constructive Templates. In HYPERTEXT '98. Proceedings of the ninth ACM conference on Hypertext and hypermedia: links, objects, time and space—structure in hypermedia systems, pages 11–20, 1998.

[16] OMG Unified Modeling Language Specification Version 1.3. http://www.omg.org/cgi-bin/doc?ad/99-06-08.pdf, 06 1999.

[17] P. Paolini and F. Garzotto. Design Patterns for WWW Hypermedia: Problems and Proposals. In Proceedings of the HT99 Workshop on Hypermedia Development: Design Patterns in Hypermedia, 02 1999.
[18] G. Rossi, D. Schwabe, and A. Garrido. Design Reuse in Hypermedia Applications Development. In Proceedings of the eight ACM conference on HYPERTEXT '97, pages 57–66, 1997.
[19] D. Schwabe, G. Rossi, and D. J. Barbosa. Systematic Hypermedia Application Design with OOHDM. In Proceedings of the the seventh ACM conference on HYPERTEXT '96, page 166, 1996.
[20] Jos Warmer and Anneke Kleppe. The Object Constraint Language. Precise Modeling with UML. Addison-Wesley, 1998.
[21] WebRatio Web Site http://www.webratio.com.