Hacia un Modelo de Interacción Abstracto para la Definición de Interfaces Multiplataforma[1]

Francisco Valverde, Jose Ignacio Panach, Nathalie Aquino, Óscar Pastor

Departamento de Sistemas Informáticos y Computación
Universidad Politécnica de Valencia, Camino de Vera s/n, 46022, Valencia.
E-Mail:{fvalverde, jpanach, naquino, opastor}@dsic.upv.es

Resumen

Actualmente la mayoría de las herramientas de producción de software que siguen las directrices MDA, no disponen de las primitivas conceptuales adecuadas para modelar la interacción de forma genérica. En su lugar, cuentan con uno o varios modelos de presentación centrados en una única plataforma concreta (Escritorio, Web, PDA, etc.) Siguiendo esta línea, OO-Method es un método de desarrollo de software dirigido por modelos capaz de generar aplicaciones para distintas tecnologías. Sin embargo, para modelar de forma adecuada la generación de software en ambientes Web, fue necesario introducir la extensión metodológica OOWS. En este trabajo se definen las bases de un nuevo modelo de interacción, que partiendo de los modelos de presentación de OOWS y OO-Method, permita especificar la comunicación usuario-interfaz de forma abstracta e independiente de la plataforma utilizado. De este modo a partir de un único Modelo de Interacción, es posible generar una interfaz de forma automática para diferentes plataformas. Para validar que el nuevo Modelo de Interacción posee la expresividad requerida, se expone un caso de estudio contrastando una aplicación de Escritorio con su equivalente en la Web.

1. Introducción

De un tiempo a esta parte, el tratamiento de las interfaces de usuario ha crecido en complejidad, ya que en muchos casos, las mismas deben ser desarrolladas para múltiples contextos de uso, como diversas categorías de usuarios (con preferencias o idiomas distintos, o con discapacidades), de plataformas (teléfonos móviles, aplicaciones Web o de Escritorio), e incluso de modalidad (interacción gráfica, vocal, 3D). Para afrontar estos desafíos van surgiendo, o evolucionando, diversas aproximaciones, entre ellas técnicas o métodos de modelado de la interacción del usuario con las aplicaciones.

En la actualidad existen multitud de aproximaciones y herramientas MDA (*Model Driven Architecture*) [13] que abordan el modelado de la interacción de sistemas software. Estas herramientas no se limitan únicamente al modelado de sistemas software tradicionales, sino que han surgido distintas especializaciones para modelar la interacción de aplicaciones Web, dispositivos móviles, sistemas domóticos, etc. Esta especialización para un ámbito en concreto, implica que los sistemas modelados sean diseñados teniendo en mente una única plataforma. Por lo tanto su migración a otra plataforma resulta costosa pues implica modelar la interacción del sistema en otro entorno MDA adecuado.

Un ejemplo de metodología basada en los conceptos MDA es OO-Method [17], un método de producción de software que permite la generación automática de sistemas software. Con dicho fin introduce una serie de modelos conceptuales que especifican tanto los aspectos estáticos como dinámicos del sistema. Este método está implementado industrialmente mediante la herramienta OlivaNova [3], permitiendo generar aplicaciones tanto para Escritorio como para ambientes Web en varios lenguajes de programación. OO-Method posee un

[1] Este trabajo ha sido desarrollado con el soporte del MEC bajo el proyecto DESTINO TIN2004-03534 y cofinanciado por FEDER

modelo capaz de representar de forma implícita conceptualmente la interacción entre el usuario y el sistema, denominado Modelo de Presentación. Sin embargo, las interfaces generadas en base a este modelo presentan carencias, tanto a nivel de interacción como estéticas, en entornos que no sean de tipo Escritorio. Estas carencias se han detectado mediante la retroalimentación de los usuarios que han utilizado aplicaciones generadas con OlivaNova. Con la finalidad de suplir la falta de expresividad para modelar la interacción en ambientes Web, surgió la propuesta OOWS [8]. OOWS es una metodología basada en el paradigma MDA para modelar la interacción con el usuario para entornos Web. OOWS se complementa con OO-Method de tal forma que se encarga de generar la interfaz Web del sistema, mientras que OO-Method modela la parte de persistencia y de lógica de negocio.

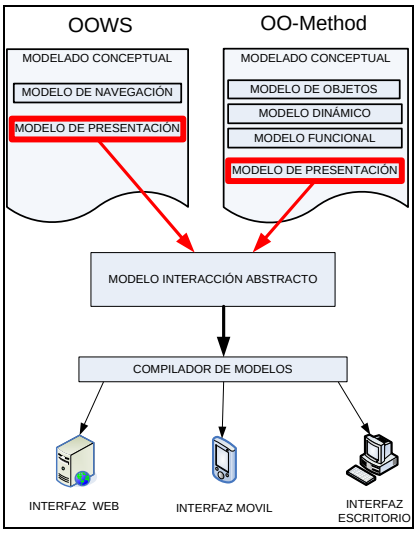

Figura 1. Definición del Nuevo Modelo de Interacción

La contribución principal de este trabajo es definir un nuevo Modelo de Interacción que partiendo de las primitivas conceptuales de OO-Method y OOWS, permita definir la interacción de forma genérica. En vez de definir un nuevo de modelo desde cero, se ha considerado conveniente partir de experiencias previas que ya han sido contrastadas.

El nuevo Modelo de Interacción definido, es totalmente independiente de la plataforma destino en la cual se generará el código. Tal y como muestra la Figura 1, esta independencia permite que a partir de un mismo modelo se pueda generar una interfaz adecuada para una aplicación de Escritorio, una aplicación Web o un dispositivo móvil simplemente indicando la plataforma destino al compilador de modelos. A fin de constatar este hecho, se han implementado sendos casos de estudio, generados en dos plataformas: Web y Escritorio.

Una vez realizadas las validaciones oportunas, el nuevo modelo de interacción será incluido en el proceso de desarrollo de OO-Method. De esta forma será posible generar interfaces que contemplen la interacción con el usuario independientemente de la plataforma destino. Por lo tanto, en el Modelo de Presentación se modelarán únicamente los aspectos relacionados con la apariencia visual de la interfaz.

Este documento se divide de la siguiente forma: en primer lugar se introducen las aproximaciones de otros autores a la hora de modelar aspectos relacionados con la interacción. A continuación se detalla brevemente el marco metodológico en el cual se encuentra nuestro trabajo. En la sección cuatro se analiza qué aspectos de la interacción se han considerado necesarios y qué mecanismos aportan los modelos de presentación (OO-Method/OOWS) actuales para cubrirlos. Una vez realizado dicho análisis, en la sección cinco, se introducen las primitivas conceptuales del nuevo modelo de interacción el cual es ejemplificado, a través de un caso de estudio en la sección seis. Por último se exponen las conclusiones, lecciones aprendidas y las líneas de trabajo futuras.

2. Trabajos Relacionados

A continuación se analizan propuestas de diversos autores a la hora de modelar la interacción desde las primeras fases de desarrollo del software. En el trabajo de Da Silva [5] se proporciona un listado de herramientas para el modelado de las interfaces en un alto nivel de abstracción. Algunos ejemplos son TRIDENT [1] y GENIUS [10]. Las herramientas descritas siguen los principios introducidos en [2], donde se establecen cuatro niveles de abstracción: tares y conceptos, interfaz de usuario abstracta, interfaz de usuario concreta e interfaz final.
Del trabajo de Da Silva también podemos destacar MOBI-D [18]. Esta aproximación utiliza una serie de modelos conceptuales, de dominio, de tareas, dialogo, de presentación y de usuario para definir una interfaz gráfica. Además enfatiza la comunicación del usuario final con el desarrollador de la

interfaz a fin de garantizar un resultado de calidad. Sin embargo la herramienta para construir los modelos se basa en un editor en forma de árbol y una notación textual. Por esta razón la construcción de la interfaz resulta compleja al estar alejada la especificación del resultado final.

Otro método de modelado abstracto de interfaces presentado en el trabajo de da Silva es USIXML (*USer Interface eXtensible Markup Language*) [22]. USIXML es un lenguaje que permite describir interfaces de usuario para múltiples contextos de uso. Las aplicaciones pueden ser descritas de tal manera que se preserva la independencia del diseño, de la interacción de las características peculiares de la plataforma tecnológica escogida. USIXML soporta diversos dispositivos (ratón, pantalla, teclado, sistema de reconocimiento de voz), distintas plataformas (teléfono móvil, *Pocket PC*, *Tablet PC*, *laptop*, *desktop*), y diferentes modalidades de interacción (interacción gráfica, vocal, 3D) a partir de un modelo común. La especificación de la interacción en USIXML se hace a dos niveles:1) Partiendo del Modelo de Tareas y Conceptos, se define un modelo abstracto independiente de la plataforma; 2) El modelo abstracto se refina en un modelo concreto para un contexto de uso específico para obtener la interfaz. Sin embargo, se debe mencionar que USIXML no es un lenguaje de implementación de interfaces de usuario en sí mismo. Necesita de un motor de transformaciones que interprete el modelo y genere el código de la interfaz. Además solo genera la parte de interacción, no la funcionalidad del sistema, tal y como propone OO-Method.

En una línea similar a la de USIXML podemos encontrar TERESA [15]. Esta herramienta es capaz de generar un modelo de interfaz abstracta a partir de árboles de tareas descritos mediante la notación CTT. A partir de esta interfaz es posible generar una interfaz concreta para múltiples dispositivos.

Además de las propuestas ya mencionadas, existen otras basadas en UML, como es el caso de WISDOM. (*Whitewater Interactive System Development with Objetc Models*) [16], un método de ingeniería software basado en UML, especializado para la construcción y el mantenimiento de aplicaciones interactivas para PYMEs. Los distintos modelos conceptuales de WISDOM abarcan el ciclo de desarrollo completo, desde la captura de requisitos, hasta la implementación. Tres de sus modelos están relacionados con el modelado de la interacción: el Modelo de Interacción de la fase de análisis, y los Modelos de Diálogo y Presentación de la fase de diseño. Sin embargo, WISDOM no da soporte a la generación automática de la aplicación software a nivel de lógica de negocio o de interfaz.

Otra propuesta interesante basada en UML es UMLi [6], que es un conjunto de modelos que extienden a UML a fin de proveer mayor soporte para el diseño de interfaces de usuario. Introduce el Diagrama de Interfaz de Usuario que constituye una propuesta basada en UML para la captura formal de los requisitos de interacción con el usuario. El proyecto UMLi contempla un generador de código para las interfaces de usuario. Sin embargo, los modelos involucran demasiados detalles, por lo que el modelado resulta poco práctico. Esto provoca que una especificación de problemas de tamaño mediano resulte bastante complicada de llevar a cabo. Debido a esto UMLi no ha gozado de mucho apoyo en ambientes industriales.

Finalmente, en el ámbito específico de la ingeniería Web, se pueden destacar aproximaciones como OOHDM [19], WebML [4], WSDM [7], UWE [11] y OOH [9]. Al igual que OOWS, estos métodos introducen modelos conceptuales para capturar de forma abstracta los aspectos estructurales, navegacionales y de presentación de las aplicaciones Web. Sin embargo la interacción con el usuario no es contemplada al mismo nivel que el resto de aspectos. Sus modelos de presentación se centran fundamentalmente en la apariencia visual y la disposición de la información en la aplicación Web.

3. Marco Metodológico : OO-Method y OOWS

OO-Method [17] es un método orientado a objetos para la producción automática de software que ofrece un marco para la construcción de sistemas de información organizacionales. OO-Method cubre todas las fases del proceso de desarrollo de software, desde las fases iniciales de recolección de requisitos, pasando por el desarrollo de un Modelo Conceptual orientado a objetos, hasta la generación del producto software correspondiente sobre una plataforma tecnológica específica. La mayor contribución del método reside en su capacidad de generar automáticamente un producto software final, totalmente funcional, a partir de un Modelo Conceptual.

El Modelo Conceptual de OO-Method está dividido en cuatro vistas complementarias:

- El Modelo de Objetos describe la estructura del sistema en términos de sus clases componentes y las relaciones estructurales de las mismas.
- El Modelo Dinámico describe las posibles secuencias de eventos para cada clase, y la comunicación e interacción entre los objetos.
- El Modelo Funcional especifica los efectos de los eventos sobre el estado de los objetos.
- Por su parte, el Modelo de Presentación [14], se estructura en tres niveles:

  Nivel 1: Árbol de Jerarquía de Acciones: establece la manera en la que el usuario interactúa con el sistema, organizando el acceso a la funcionalidad a través de una abstracción en forma de árbol.

  Nivel 2: Unidades de Interacción (UIs): representan las principales operaciones de interacción que pueden ser realizadas con los objetos del dominio. Existen cuatro tipos:

1. La UI de Servicio permite ejecutar servicios para modificar objetos, sus atributos y sus relaciones.
2. La UI de Instancia muestra el estado de un único objeto a partir de su oid.
3. La UI de Población muestra un grupo de objetos pertenecientes a una clase.
4. La UI de Maestro/Detalle presenta dos UI de Población/Instancia que comparten una relación estructural. Al seleccionar una instancia de la UI directora se muestra en el detalle las instancias relacionadas

Nivel 3: Patrones Elementales (PE): permiten restringir y precisar el comportamiento de las diferentes UIs. Los patrones elementales se encargan de tareas como filtrar la información de la IU (Filtro), establecer qué información mostrar (Conjunto de Visualización), qué acciones pueden ser llevadas a cabo en una IU (Acción) o establecer la navegación (Navegación).

Actualmente OO-Method está implementado en la herramienta comercial OlivaNova validada industrialmente en diversos proyectos.

Por su parte, OOWS (*Object Oriented Web Solution*) es un método de producción de aplicaciones Web que extiende OO-Method introduciendo la noción de navegación y de un modelo de presentación más adecuado para entornos Web. Además, define un proceso de producción sistemático adaptado a la Web, usando una estrategia de generación de código basada en modelos.

OOWS introduce un Modelo de Navegación y redefine el Modelo de Presentación del Modelo Conceptual de OO-Method, ya que con las primitivas originales las interfaces generadas para entornos Web padecen de ciertos problemas de usabilidad. A continuación se detallan los modelos mencionados:

- Modelo de Navegación*:* define la estructura de navegación del sistema, es decir, describe la navegación permitida, para cada tipo de usuario, en términos de un Mapa de Navegación. Este mapa se representa por medio de un grafo dirigido en el que los nodos representan Contextos de Navegación y los arcos representan Enlaces de Navegación que definen los caminos de navegación válidos del sistema. Los Contextos de Navegación constituyen Unidades de Información Abstractas (*Abstract Information Units* - AIU). Una AIU define una vista sobre un conjunto de clases del Modelo de Objetos indicando que atributos, operaciones y relaciones van a estar disponibles en el contexto. La información es recuperada a partir de una clase directora y varias clases relacionadas con ella o complementarias, que aportan información adicional.
- Modelo de Presentación: captura los requisitos de presentación de información. Asocia patrones de presentación a los Contextos de Navegación tales como la ordenación, cardinalidad o disposición de la información. Además, incorpora un mecanismo para definir los aspectos estéticos a través de plantillas de presentación reutilizables.

Al igual que OO-Method, OOWS dispone de una herramienta que genera interfaces Web a partir de modelos conceptuales. Esta herramienta está basada en la Eclipse Modelling Plattform, integrándose con la herramienta OlivaNova a fin de generar aplicaciones Web funcionales. Actualmente se ha desarrollado como un prototipo con fines académicos. Más información sobre dicha herramienta puede encontrarse en [21].

4. Especificación de la Interacción en los Modelos de Presentación actuales

En esta sección vamos a detallar qué características a nivel de interacción pueden aportar los Modelos de Presentación de OOWS y OO-Method. La interacción engloba más aspectos que la simple visuali-

zación de las interfaces. Por lo tanto la definición de aspectos estéticos (fuentes tipográficas, colores, tamaño de ventanas etc.) no van a ser analizados. Para más detalles puede consultarse [14] y [20]. Por razones de espacio el análisis se centra en aquellas características que aportan un mayor valor a la hora de definir el nuevo Modelo de Interacción.

Recuperación de la información: a través de estos mecanismos de interacción, el usuario recibe la información que reside en el sistema. En OOWS la recuperación de información se realiza a través de AIUs (ver sección 3). Por otro lado, la misma interacción se modela en OO-Method mediante las UIs de Población, Instancia y Maestro-Detalle. El concepto de AIU es muy flexible y de gran riqueza expresiva de tal forma que engloba a las tres UI de OO-Method. El uso de cada una de estas tres UI en la fase de modelado depende del número de instancias que se van a recuperar. Sin embargo, en OOWS no es necesario hacer esta diferenciación en la fase de modelado. Es el Compilador de Modelos el encargado de llevar a cabo esta tarea. Además, si en una AIU la clase directora se relaciona con "n" instancias de una clase complementaria, el propio compilador asocia la representación de dicha información como un Maestro-Detalle atendiendo a criterios de usabilidad.

*Ejecución de servicios***:** usando esta característica de interacción, definimos cómo el usuario activa el comportamiento del sistema. En OOWS, los servicios que pueden ser ejecutados en un contexto determinado se definen en las clases navegacionales de las AIU. El Compilador de Modelos de OOWS asocia un formulario Web genérico para ejecutar el servicio, acorde a los argumentos del mismo. Aunque se introducen reglas para evitar la introducción de valores incorrectos en función del tipo del argumento, esta validación no elimina todos los errores que pueda tener el usuario. En este aspecto la UI de Servicio definida en OO-Method ofrece un mayor número de posibilidades ya que no solo posee la funcionalidad de evitar la introducción de argumentos de un tipo de dato erróneo. La UI de Servicio permite definir valores por defecto en los argumentos y dispone de reglas de validación basadas en máscaras a través de un Patrón Elemental.

Navegación: la navegación describe la secuencia de pasos (ventanas, páginas Web, diálogos etc.) que sigue el usuario a través de la aplicación. El Modelo de Presentación de OO-Method posee un Patrón Elemental para soportar la Navegación. Este patrón se asocia a una UI a través de las asociaciones que posea la clase en el Modelo de Objetos. Por lo tanto sólo permite definir navegaciones basadas en relaciones estructurales. Aplicando esta propuesta se han detectado problemas de usabilidad a la hora de trasladar este patrón a la Web. La diferencia fundamental entre ambas propuestas, es que en OOWS puede asociarse tanto a una relación estructural, como a la información recuperada o a la ejecución de un servicio. Una aplicación Web requiere expresividad adicional para especificar navegación de manera adecuada. Por ejemplo, es necesario determinar dónde acudir después de ejecutar un servicio (enlaces de servicio), o determinar atributos de enlace a nuevos contextos navegacionales / UI. Todo este tipo de expresividad navegacional se realiza con mayor facilidad en OOWS a través del conjunto de primitivas navegacionales que proporciona.

Además, OOWS dispone de un modelo basado en Roles el cual permite que la navegación asociada a un Rol pueda ser heredada por otro Rol. Esta característica proporciona una mayor sencillez a la hora de definir la navegación en ambientes Web.

*Mecanismos de acceso a la información***:** Entendemos por mecanismos de acceso, aquellas interacciones que facilitan al usuario el acceso a un conjunto de información. El primer mecanismo necesario, es el filtrado de información el cual se encuentra definido tanto en OOWS como en OO-Method. Este mecanismo permite a través de un valor introducido por el usuario, restringir la población a partir de una condición. Otro mecanismo que también se encuentra en ambos modelos, es la definición del criterio de ordenación de la información recuperada. Por último, OOWS introduce el concepto de paginación para poder dividir una población muy elevada en grupos o "páginas". De esta forma se simplifica la interacción del usuario con el conjunto de información.

*Composición de varias unidades de interacción***:** un aspecto que conviene destacar es la posibilidad de definir en OOWS los contextos como una agregación de AIUs. Esta capacidad de composición no se encuentra actualmente disponible en OO-Method. La única composición modelable es la de Maestro-Detalle. Sin embargo, la capacidad de composición es necesaria a la hora de definir interacciones complejas en las cuales participa más de

una UI y la relación entre ambas no es de Maestro-Detalle.

El hecho que el mayor número de las características de interacción que deseamos se encuentren en OOWS, nos ha llevado a definir el nuevo modelo de interacción a partir de éste. Además, actualmente se está trabajando en una herramienta MDA de soporte a OOWS [21] que puede ser fácilmente extendida para soportar el nuevo modelo. A continuación se explican las primitivas de este nuevo Modelo de Interacción.

5. Definición de un nuevo Modelo de Interacción Abstracto

Se define el Modelo de Interacción Abstracto como aquel que representa las interacciones posibles del usuario con el sistema de información. El Modelo de Interacción se enmarca dentro de OO-Method integrándose con el resto de Modelos Conceptuales.

Las unidades principales del Modelo de Interacción Abstracto propuesto son los *Contextos de Interacción (CI)***.** Los CI se utilizan como un medio para encapsular los elementos de interacción necesarios para que el usuario lleve a cabo una tarea específica. Por ejemplo, un CI puede englobar la interacción necesaria para reservar un automóvil. Los CI se asocian a un *Rol* (usuario registrado, administrador, cliente etc.) que define el conjunto de usuarios que tienen privilegios para realizar la interacción definida en el contexto. El conjunto de CI al cual tiene acceso un Rol determinado define su *Mapa de Interacción***.**

Los contextos contienen Unidades de Interacción básicas (UI) que definen una interacción específica con el usuario. En el modelo propuesto de interacción se han considerado tres interacciones fundamentales (ver sección anterior): 1) recuperación de información, 2) ejecución de servicios y 3) navegación. Un CI debe contener al menos una de estas unidades de interacción. El número máximo no se encuentra limitado pero sólo deben incluirse las UI que modelen la interacción que describe el contexto. Las primitivas introducidas se muestran gráficamente en la Figura 2. Esta figura muestra los CI asociados a una tarea como contenedores de las distintas UIs de interacción. Cada Rol tiene determinado en su Mapa de Interacción a cuales CI tiene permitido acceder. También se puede apreciar una relación de herencia entre los dos roles, por lo tanto el Rol Hijo hereda todos las interacciones que tenga definido el Rol Padre...

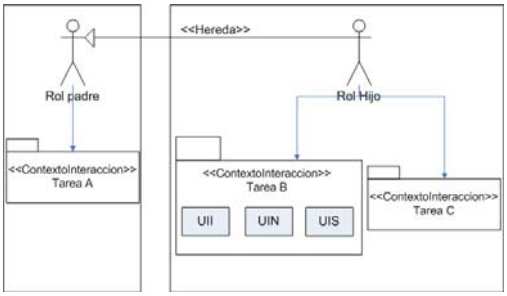

Figura 2. Definición de dos mapas de interacción.

A continuación se explica en detalle cada una de estas UIs.

5.1. Unidad de Interacción de Información (UII)

Una UII representa la interacción del usuario con el sistema para recuperar un conjunto de información relacionada. Conceptualmente es similar a una AIU de OOWS (ver sección 3) pero contemplando nuevos elementos de interacción. Las UIIs representan una vista sobre el Modelo de Objetos, definiendo a través de clases y sus atributos el conjunto de información a recuperar. Toda UII está formada por una clase directora y una o varias clases complementarias relacionadas con ésta. La información a visualizar desde las clases complementarias es aquella que se encuentra relacionada con las instancias de la clase directora.

Las UII además engloban la interacción definida como mecanismos de acceso. Estos mecanismos son los siguientes:

Filtros: Dado que por defecto una UII recupera todas las instancias de población que representa, es necesario incluir algún mecanismo de este tipo. El mecanismo de acceso utilizado ha sido la definición de filtros**.** Los filtros se definen a partir de los atributos definidos en las UII, recuperando aquellas instancias en donde el atributo cumpla la condición de filtro. La gramática para definir las condiciones de filtrado, basada en la utilizada en el Modelo de Presentación de OO-Method, puede consultarse en [14]. Dependiendo de su comportamiento distinguimos dos tipos de filtros:

- *Estáticos:* definen una condición de filtrado que el usuario no puede modificar. Este tipo de filtros puede activarse automáticamente (*Auto*)

para que siempre se apliquen al recuperar la población de la UII o que el usuario decida cuando aplicar el filtro (*Manual*).

- *Dinámicos:* a diferencia de los anteriores, estos filtros implican que el usuario introduzca un valor para los atributos de la UII utilizados en la condición de filtrado.

Paginación: este mecanismo permite establecer el número máximo de instancias o cardinalidad que debe mostrar la UII simultáneamente. Cuando se encuentra definido permite al usuario acceder a las distintas particiones de información o "páginas" de forma ordenada.

En el ejemplo de la Figura 3 se muestra una UII que recupera la información de los automóviles y la complementa con el grupo al cual pertenecen. Además existen dos filtros definidos. El primero permite al usuario introducir el grupo de automóviles que desea ver. El segundo, introducir el nombre del modelo para filtrar aquellos modelos similares al valor introducido.

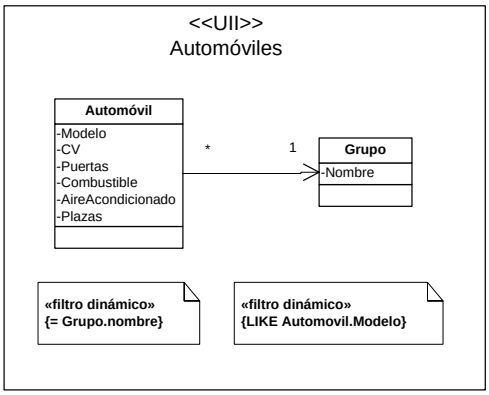

Figura 3. Ejemplo de UII con filtros

5.2. Unidad de Interacción de Servicio (UIS)

Para la definición de la ejecución de los servicios se introduce la UIS. Una UIS se asocia a un servicio concreto del diagrama de clases y permite establecer cómo los argumentos del servicio deben ser introducidos por el usuario. Con este propósito se han establecido las siguientes propiedades para cada uno de los argumentos:

- Valor por Defecto: indica el valor por defecto del argumento a introducir.
- Visibilidad: establece si el argumento se solicitará al usuario o no.
- Regla de Validación: define una regla de validación basada en una máscara, para establecer el formato que debe cumplir el valor introducido como argumento. Por ejemplo, obliga al usuario a introducir una fecha en un formato específico.

En la tabla 1 se muestra la definición para el servicio Realizar reserva. Además de la introducción de los argumentos la UIS se encarga de mostrar al usuario el resultado (satisfactorio/erróneo) de la ejecución del servicio.

Argumento	Visible	DefaultValue	Regla
Nombre	Si		
Apellidos	Si		
Coche	Si		
FechaRecogida	Si	Today()	dd/mm/aa
FechaDevolución	Si	Today()+1	dd/mm/aa

Tabla 1. Especificación del servicio Realizar reserva.

5.3. Unidad de Interacción de Navegación (UIN)

Para definir la interacción de navegación utilizamos esta UI que representa los distintos tipos de navegación disponibles en OOWS. En nuestro modelo, se define la navegación como toda aquella interacción del usuario que provoque un cambio del Contexto de Interacción. Una UIN está asociada a un elemento de interacción del modelo que dispara la navegación y el CI destino que se alcanza a través de la navegación. La navegación al CI destino sólo es posible si está incluido en el Mapa de Interacción del Rol del usuario. La interacción de navegación puede implicar el envío de información desde el CI origen al CI destino dependiendo del elemento de interacción. Los elementos del modelo sobre los cuales puede asociarse una UIN son los siguientes:

- *Atributos de las UII:* dispara la navegación cuando se selecciona el valor del atributo, enviando al CI destino el oid de la instancia seleccionada.
- *Relaciones definidas de la UII:* dispara la navegación cuando se selecciona alguna instancia relacionada con la clase directora. El CI destino recibe tanto el oid de la instancia seleccionada como el identificador de la relación.

- *Servicio de la UIS*: implica la navegación al CI destino cuando la ejecución del servicio asociado es satisfactoria. Envía el identificador del servicio y el resultado de su ejecución.

La información enviada al CI a través de una interacción de navegación se representa en el CI destino como una *variable de contexto***.** Esta variable de contexto permite el acceso al CI destino a la información del elemento de interacción que ha iniciado la navegación. Por ejemplo, en caso que se haya definido una navegación sobre un atributo "nombre vehículo", en el CI destino se tendrá visibilidad a través de la propiedad oid de la variable de contexto, al identificador asociado a dicha instancia de vehículo. Esta información puede ser utilizada en el CI destino, por ejemplo, para filtrar qué información debe ser mostrada en una UII a partir del identificador del vehiculo.

En la Figura 4 se muestra un ejemplo de navegación asociada al atributo *Nombre*. Cuando el usuario seleccione una instancia a través de este atributo, se navegará a otra CI llamada *Selección Alquiler*.

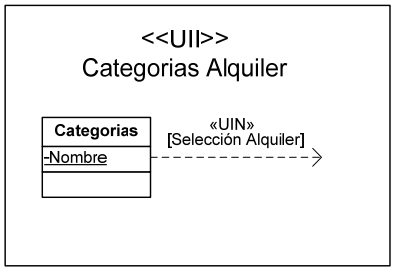

Figura 4. UII con un elemento de navegación definido

6. Caso de estudio

Como caso práctico que muestre un sistema modelado utilizando el nuevo Modelo de Interacción propuesto, se ha elegido el de una agencia de alquiler. Se trata de una aplicación para poder realizar reservas de distintos medios de transporte como yates, barcos de vela, motos, turismos, camiones, furgonetas o bicicletas. A modo de ejemplo se ha seleccionado la tarea Alquiler de Automóvil.

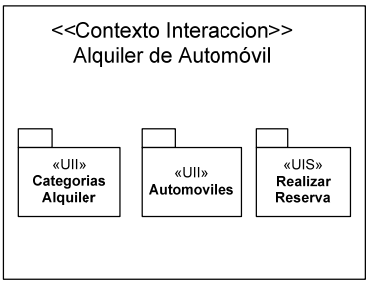

Figura 5. Contexto de Interacción de la tarea Alquiler de Coche

El CI definido para esta tarea, tal y como se muestra en la Figura 5, se compone de las siguientes UI:

- UII Categorías: En esta UI se muestran las distintas categorías de medios de transporte que pone a disposición nuestra empresa: automóviles, motocicletas, autobuses, yates etc. La Figura 4 representa el modelo de esta UI, en el cual existe una UIN asociada al atributo categoría. Al interaccionar con dicho atributo se navegará al contexto "Selección Alquiler". En dicho contexto es posible conocer el oid de la categoría seleccionada a través de la variable de contexto.
- UII Automóviles: En esta AIU se muestran todos los automóviles disponibles para su alquiler. El contexto está representado en la Figura 3, en donde la clase con el rol directora es Automóvil y la clase grupo presenta información complementaria.
- UIS Reservar Automóvil: una vez elegido el automóvil en la UII Automóviles en esta UIS se puede ejecutar el servicio para alquilarlo. Para ello se deben introducir en el sistema los datos para hacer la reserva. Estos datos son el periodo de tiempo durante el que se va a alquilar y el nombre de la persona que hace el alquiler. La tabla 1 representa los argumentos de esta UIS.

Utilizando el nuevo Modelo de Interacción, se puede generar la misma funcionalidad tanto para ambientes Web como para Escritorio, tal y como muestra la Figura 6. La única diferencia entre la aplicación para la Web (izquierda) y para escritorio (derecha) es el aspecto visual. La funcionalidad de ambas y su interacción con el usuario es similar.

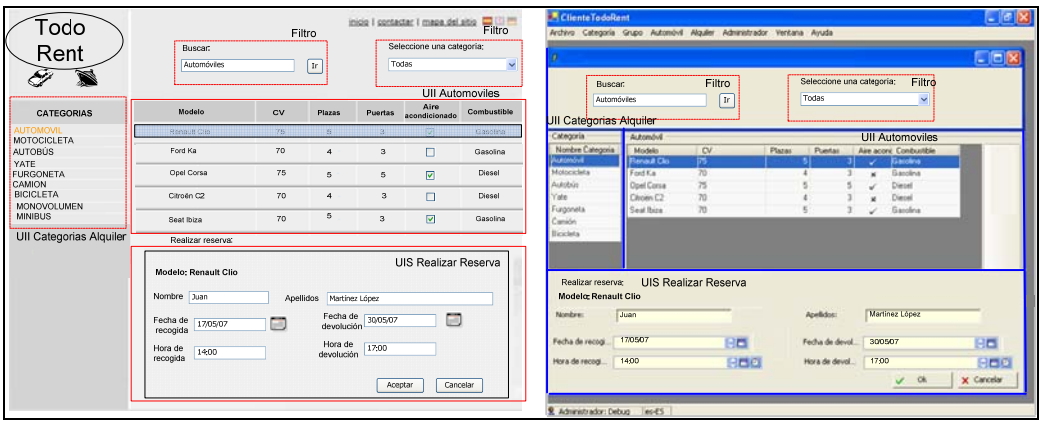

Figura 6. (a) Aplicación para la Web. (b) Aplicación para Escritorio.

7. Conclusiones

En este trabajo se ha definido un nuevo Modelo de Interacción a partir de las experiencias previas en el modelado de entornos de Escritorio y Web. Este modelo a diferencia de los modelos de presentación anteriores presenta las siguientes ventajas:

- Ha sido definido sin tener en mente la interacción para una plataforma concreta.
- Reutiliza conceptos que han sido ampliamente contrastados y validados.
- Se enmarca dentro del método de generación de productos software OO-Method.

El modelo propuesto se ha utilizado en un caso de estudio. Este caso práctico demuestra que la apariencia visual o la plataforma destino, no implica que la interacción deba ser modificada para las mismas tareas. El utilizar un único Modelo de Interacción para todas las plataformas resulta más eficiente. Partiendo del mismo Modelo Conceptual se puede generar automáticamente la aplicación para las distintas plataformas que soporte el Compilador de Modelos.

Para el nuevo modelo se han introducido en este trabajo los mecanismos básicos de interacción considerados más relevantes. En próximos trabajos se introducirán nuevos aspectos de interacción como el *feedback* al usuario o mecanismos para que las distintas UI de un mismo CI puedan interactuar entre sí.

Por otra parte, se ha detectado que determinados aspectos de la interacción pueden verse afectados por restricciones tecnológicas. Claros ejemplos son el tamaño de visualización o el dispositivo de entrada (ratón, táctil, teclas, etc.) en un dispositivo móvil. Sin embargo, la especificación de dichas restricciones debe ser situada a un nivel de abstracción inferior, en un modelo de interacción concreto de la plataforma. Actualmente se está estudiando cómo debe realizarse la transición del modelo abstracto actual al nivel concreto.

Como otras líneas de trabajo futuras se está estudiando qué mecanismos debe contemplar el nuevo Modelo de Interacción para cumplir con determinados criterios de usabilidad. Además, aunque ya existe un prototipo en desarrollo para generar las interfaces a partir de este Modelo de Interacción, se pretende incorporar dicho modelo en el proceso de producción software de OO-Method. Con ello se conseguiría generar las interfaces automáticamente junto con toda la funcionalidad del sistema. Por último, las mejoras que aporta este Modelo de Interacción serán validadas empíricamente a través de las métricas adecuadas.

Referencias

[1] Bodart, F., Hennebert, A., et al. (1994) A Model-Based Approach to Presentation: A Continuum from Task Analysis to Prototype. In Proceedings of DSV-IS'94, pp. 25-39, Bocca di Magra.

[2] Calvary, G., Coutaz, J., Thevenin, D., Limbourg, Q., Bouillon, L. and Vanderdonckt, J. A Unifying Reference Framework for multi-target user interfaces. Interacting with Computers, 15 (3). 289-308.

[3] Care Technologies: http://www.care-t.com Ultima Visita: Abril-2007.

[4] Ceri, S. Fraternali, P., Bongio, et al. (2003). Designing Data-Intensive Web Applications.Morgan Kaufman

[5] Da Silva, P. P. (2001) "User interface declarative models and development environments: A survey". Interactive Systems. Design, Specification, and Verification, 8th International Workshop, DSV-IS 2001, Glasgow, Scotland, Springer-Verlag Berlin.

[6] da Silva, P. P. d. and N. W. Paton (2003). "User Interface Modelling in UMLi " IEEE Softw. 20 (4). pp. 62-69

[7] De Troyer, O. and Casteleyn, S. (2003) Modelling Complex Processes from web applications using WSDM. In IWWOST 2003. Oviedo, Spain. pp 1-12.

[8] Fons J., P. V., Albert M., and Pastor O. (2003). Development of Web Applications from Web Enhanced Conceptual Schemas. ER 2003, LNCS. Springer.pp. 232-245.

[9] Gómez, J., Cachero, C., Pastor, O. (2000) Extending an Object-Oriented Conceptual Modelling Approach to Web Application Design. CAiSE'2000, LNCS 1789, Pags 79-93

[10] Janssen, C., A. Weisbecker, et al. (1993). Generating user interfaces from data models and dialogue net specifications. SIGCHI, Amsterdam, The Netherlands ACM Press: 418-423

[11] Koch, N. (2000) Software Engineering for Adaptive Hypermedia Applications. PhD thesis, Ludwig-Maximilians-University, Munich, Germany.

[12] Markopoulos, P., Pycock, J., Wilson, S. and Johnson, P. (1992) Adept - A task based design

[13] MDA, http://www.omg.org/mda/. Ultima Visita Mayo 2007.

[14] Molina, P. J. (2003). Especificación de interfaz de usuario: de los requisitos a la generación automática. PhD. DSIC. Valencia, Universidad Politécnica de Valencia; 382 páginas.

[15] Mori, G., Paterno, F. and Santoro, C. Design and Development of Multidevice User Interfaces through Multiple Logical Descriptions. *IEEE Transactions on Software Engineering.*

[16] Nunes, N. J. y J. F. e. Cunha (2000). "Wisdom: a software engineering method for small software development companies." Software, IEEE 17(5): 113-119.

[17] Pastor, O., Gómez, J., Insfrán, E. Pelechano, V. (2001) The OO-Method Approach for Information Systems Modelling: From Object-Oriented Conceptual Modeling to Automated Programming. Information Systems, 26(7) 507–534.

[18] Puerta, A. and Maulsby, D. (1997) Management of Interface Design Knowledge with MOBI-D. In Proceedings of IUI'97, pp. 249-252, Orlando, FL, USA.

[19] Schwabe D., Rossi G., and Barbosa. S. (1996) Systematic Hypermedia Design with OOHDM. In ACM Conference on Hypertext, Washington, USA.

[20] Valderas, P., Pelechano, V., Pastor, O, (2006) Towards an End-User Development Approach for Web Engineering Methods. CAiSE 2006, Pags 528-543, Luxemburg.

[21] Valverde, F., P. Valderas, et al. (2007). OOWS Suite: Un Entorno de desarrollo para Aplicaciones Web basado en MDA. IDEAS, Venezuela.

[22] Vanderdonckt, J., Q. Limbourg, et al. (2004). USIXML: a User Interface Description Language for Specifying Multimodal User Interfaces. Proceedings of W3C Workshop on Multimodal Interaction WMI'2004, Sophia Antipolis, Greece.