HeAR, Una herramienta de Adquisición de Requisitos

Laura Petersen[1], Stella Tornabene[1], María Carmen Leonardi[2], Jorge Doorn[2,3]

[1] Univ. Nacional del Centro de la Prov. De Bs.As.
[2] INSUC-INTIA, Univ. Nacional del Centro de la Prov. Buenos Aires
[3] Universidad de Belgrano
e-mail:{ lpeter, storn, cleonard, jdoorn}@exa.unicen.edu.ar

Resumen. La Ingeniería de Requisitos cumple un papel primordial en el proceso de producción de software, ya que enfoca un área fundamental: la definición de lo que se desea producir. Obtener documentación de los requisitos implica la sucesión de las actividades de elicitar, modelar y analizar los fenómenos propios del Universo del Discurso, involucrando una combinación de métodos, personas y herramientas. *Requirements Baseline*[1] es una metodología basada en lenguaje natural para registrar el vocabulario del macrosistema y escenarios para modelar el comportamiento, aumentada con verificaciones y validaciones tendientes a producir documentación de mejor calidad. En este artículo se presenta una herramienta de soporte a esta metodología, que tiene como objetivos primordiales asistir durante el proceso de documentación de requisitos automatizando algunas verificaciones. El metamodelo en que se basa el diseño de la herramienta permite adaptarla con facilidad a los cambios producidos en la documentación y su proceso de producción, hecho frecuente en una metodología en estudio y evolución.

Palabras Claves: Ingeniería de Requisitos, lenguaje natural, Léxico Extendido del Lenguaje, escenarios, Herramienta

1 Introducción

El desarrollo de un sistema de software requiere de la comprensión del macrosistema en el que será utilizado. Diversos investigadores han identificado que una de las principales causas de falencia en el producto final está en la especificación de los requisitos. La definición de requisitos no es una tarea trivial; es en sí difícil y tediosa, de allí la necesidad de contar con instrumentos adecuados que tornen más fácil trabajar con ella. Por esta razón, los ingenieros de software necesitan técnicas y herramientas que faciliten la adquisición y el modelado de los elementos en ese macrosistema (Mamani and Leite 1999).

La metodología de Leite (Leite and Oliveira 1995; Leite et al. 1997) propone trabajar con documentos en lenguaje natural altamente vinculados entre sí. Estos documentos están integrados en una estructura llamada *Requirements Baseline*, que

1 Una posible traducción de Requirements Baseline sería Andanivel de Requisitos, sin embargo, este término no es utilizado en la comunidad de requisitos. Por esta razón se utilizará la frase Requirements Baseline sin traducir.

acompaña al proceso de desarrollo de software. El documento central para esta metodología es el Léxico Extendido del Lenguaje (LEL), un diccionario que registra el vocabulario utilizado en el macrosistema. El comportamiento del macrosistema es descripto por la SMV, un conjunto de escenarios que evoluciona mientras se desarrolla el proceso de construcción de software. La vinculación entre los documentos está soportada por una vista de hipertexto. El desarrollo de casos de estudio con un seguimiento manual de esta metodología resulta dificultoso a la hora de mantener los vínculos, por lo que se hace necesario contar con una herramienta que automatice estas tareas. En este artículo se presenta HeAR, un prototipo de herramienta de adquisición de requisitos. HeAR permite la autoría y navegación de los documentos de requisitos que se hayan especificado para un determinado proyecto, soportando actualmente parte de los modelos que componen la *Requirements Baseline*. El prototipo automatiza algunas tareas de edición tediosas como el seguimiento de reglas sintácticas, la identificación de vínculos entre la información y la inspección de calidad y completitud de la información relevada.

HeAR es una instanciación de MENDoR, un metamodelo de representación de documentos de requisitos propuesto en (Petersen and Tornabene 1999), que permite especificar la estructura de los documentos soportados así como su forma de presentación al usuario, sin requerir código especial que los administre. Por ser instancia de este metamodelo, HeAR es un prototipo extensible, al que pueden agregarse nuevos documentos y servicios con un mínimo tiempo y esfuerzo de programación.

En la sección 2 se introduce brevemente la *Requirements Baseline*. La sección 3 describe el metamodelo MENDoR. En la sección 4 se describe la instanciación de MENDoR para la creación de HeAR. La sección 5 presenta la funcionalidad de HeAR, así como algunos detalles de su implementación. Finalmente, en la ultimas secciones se compara este prototipo con otros similares y se comentan conclusiones y trabajos futuros.

2 Requirements Baseline

Esta metodología se centra en una estructura orientada al cliente llamada *Requirements Baseline*, siendo ésta un metamodelo que contiene descripciones sobre el dominio del problema y el artefacto de software que ha de ser construido dentro de ese dominio. Estas descripciones, relacionadas entre sí, son escritas en lenguaje natural siguiendo patrones determinados. El uso de lenguaje natural posibilita la intervención de los clientes a la hora de validar el conjunto de especificaciones obtenidas.

La idea básica detrás de la *Requirements Baseline* es que es persistente: se genera durante el proceso de ingeniería de requisitos, evoluciona junto al proceso de desarrollo del software y acompaña las tareas de mantenimiento. Está compuesta por cinco vistas:

- LEL, vista del modelo léxico
- BMV, vista del modelo básico
- SMV, vista del modelo de escenarios

- HV, vista de hipertexto
- CV, vista de configuración

La vista léxica es un metamodelo diseñado para ayudar a la elicitación del lenguaje usado en el macrosistema del cual el sistema de software formará parte. La vista básica está soportada por diagramas de entidad-relación que representan los requisitos externos propuestos por el cliente del macrosistema. La vista de escenarios describe situaciones del comportamiento del macrosistema. Las vistas de Hipertexto y de Configuración son ortogonales a las otras tres vistas. Son un soporte de servicios indispensable para garantizar el acceso a la información almacenada y el seguimiento de los requisitos y sus revisiones.

A continuación se describe en detalle las vistas LEL, SMV y HV por ser las implementadas actualmente por HeAR.

2.1 LEL, la Vista del Modelo Léxico

El LEL(Leite and Franco 1990) es una representación de los símbolos en el lenguaje del dominio del problema, que intenta capturar el vocabulario de un macrosistema. Su objetivo principal es que el ingeniero de software entienda el lenguaje que habla el usuario, entendiendo los términos que utiliza, sin preocuparse por entender el problema. Los símbolos del LEL definen objetos (entidades pasivas), sujetos (entidades activas), verbos y estados[2]. En (Hadad et al. 1997) se expresa la utilidad de adaptar esta clasificación al Universo del Discurso de la aplicación en estudio o la de adoptar una procedente de este. Ambas clasificaciones ayudan a homogeneizar las descripciones. Cada símbolo tiene uno o más nombres que lo identifican, llamados sinónimos. El significado de los símbolos se representa mediante dos tipos de descripciones: nociones e impactos. La noción indica qué es, describiendo la denotación de la palabra o frase. El impacto describe cómo repercute en el sistema, es decir su connotación. Estas descripciones se construyen siguiendo dos principios importantes: maximizar el uso de términos del LEL utilizados (principio de circularidad) y minimizar el uso de símbolos externos al dominio destino (principio de mínimo vocabulario). Mediante estos dos principios se logra un conjunto autocontenido de símbolos, altamente vinculado, y por lo tanto plausible de ser representado como un documento hipertextual.

2.2 SMV, la Vista del Modelo de Escenarios

Los escenarios, tal y como son considerados en esta metodología, son descripciones que evolucionan de situaciones en el entorno. Su objetivo principal es comprender el problema en su totalidad. Expresados en lenguaje natural, tienen un fuerte vínculo con el LEL, al que adoptan como referencia.

Cada escenario tiene la siguiente estructura: un Título que lo identifica, el Objetivo a lograr en el macrosistema y un Contexto que describe la ubicación geográfica y temporal del escenario, así como un estado inicial o precondición. También se especifican Recursos (medios de soporte, dispositivos que se necesita estén disponibles en el escenario) y Actores (personas o estructuras de organización

[2] En adelante, nos referiremos a estas categorías como categorías básicas.

que tienen un rol en el escenario). La descripción de todos estos componentes son sentencias declarativas. Los Episodios son una serie ordenada de sentencias escritas de manera simple, que posibilitan la descripción de comportamiento. Pueden ser opcionales, condicionales o simples, y estar agrupados según su forma de ocurrencia en grupos secuenciales o no secuenciales. Un aspecto importante es la posibilidad de que un episodio pueda ser un escenario en sí mismo; entonces, se tienen relaciones de jerarquía entre escenarios y subescenarios, y relaciones de integración entre escenarios (Leite et al. 2000). Esto induce una estructura jerárquica en la vista de escenarios que facilita su comprensión y especificación. Para cada escenario se prevé la descripción de excepciones: causas y soluciones a situaciones que discontinúan su evolución natural, e impiden el cumplimiento del objetivo. Estas reflejan la falta o mal funcionamiento de un recurso. El tratamiento de la excepción puede ser realizado por otro escenario. Para algunos componentes del escenario pueden describirse restricciones, es decir requerimientos de ámbito o calidad referidos al componente. Las restricciones permiten especificar aspectos referidos a requisitos no funcionales.

Algunos trabajos han profundizado en el estudio de las vistas LEL y SMV de la *Requirements Baseline* aplicándolas a varios casos de estudio (Hadad et al. 1996; Rivero et al. 1998), lo que ha permitido desarrollar heurísticas de construcción y verificación de ambas vistas, y probar su uso. Se han propuesto heurísticas de construcción del LEL, a partir del cual se deriva luego un primer conjunto de escenarios (Hadad et al. 1997). Las heurísticas propuestas en (Doorn et al. 1998) guían la inspección del conjunto de escenarios para reducir en el mayor grado posible los errores, discrepancias y omisiones de los mismos. En (Leite et al. 2000) se describe el proceso global.

2.3 HV, la Vista de Hipertexto

Esta vista es ortogonal a las anteriores. Trabaja como un integrador de las mismas, habilitando la definición de vínculos dentro de la misma vista y entre ellas. Estos vínculos están determinados por las relaciones naturales entre las vistas y por el uso de un vocabulario controlado. En esta red hipertexto(Leite et al. 1997), los nodos corresponden a cada elemento de las distintas vistas antes mencionadas: símbolos del LEL y escenarios en la SMV. En cuanto a los vínculos, en el caso del LEL, están asegurados por la utilización del principio de circularidad. En la SMV, los vínculos permiten explorar las relaciones entre escenarios, y entre los componentes de cada escenario y con el LEL. Los vínculos pueden originarse a partir de relaciones estructurales existentes, a partir del uso de información en el LEL, o explícitamente. Dada la gran cantidad de nodos y relaciones existentes, el modelo de hipertexto está organizado en contextos navegacionales, esto es, en conjuntos de nodos que están muy relacionados y que típicamente se recorrerán juntos. Estos contextos pueden estar constituidos por los nodos conectados por los tres tipos de vínculos antes mencionados, o bien porque los nodos tienen características comunes, por ejemplo, utilizan un mismo recurso, involucran a un mismo actor, o son distintas versiones de un mismo escenario.

3 MENDoR, el Metamodelo de Base

MENDoR, el metamodelo de Especificación y Navegación de Documentos de Requisitos, permite especificar la estructura de los documentos, el modo en que se muestran y las navegaciones entre ellos. Estas especificaciones se realizan mediante la implementación de las clases de los cuatro modelos que lo componen. Estos modelos son:

- MAB, Modelo Abstracto de la Baseline
- MAP, Modelo Abstracto del Proyecto
- MAN, Modelo Abstracto de la Navegación
- MAI, Modelo Abstracto de la Interface a Usuario

El MAB soporta la definición del conjunto de documentos que modelan los requisitos. El MAP administra la información relevada sobre el dominio que se está estudiando, siguiendo el patrón definido en el MAB. El MAN modela el acceso a los documentos. El MAI modela la interacción entre los documentos definidos y el usuario, tanto a nivel de vistas del sistema como de eventos generados por el usuario.

3.1 MAB, Modelo Abstracto de la Baseline

Este modelo especifica los objetos, relaciones y colaboraciones necesarias para representar la estructura de un documento de requisitos, que se muestran en la Fig. 1. En este modelo, se llama Vista de Baseline a cada tipo de documento que se modela, y Baseline a la estructura que los agrupa[3]. Una Vista de Baseline puede estar compuesta por más de un documento llamado Elemento. Cada documento individual se describe en términos de los componentes. Un componente es la unidad más pequeña identificada en este modelo. Un ejemplo ayudará a entender cómo modelar un documento completo utilizando el MAB. Si se desea modelar el LEL, será necesario definir: a) una vista, la vista léxica, b)un elemento, el símbolo, y c) varios componentes, para el nombre, la categoría, las nociones y los impactos que componen a un símbolo del LEL.

En MAB, cada uno de los componentes es de un tipo de dato, que indica su rol en el hipertexto que constituye la baseline (Texto o Ancla). En el caso de los textos, puede especificarse si responden a algún estilo de escritura en particular (Formato), y si la sintaxis indica que es precedido o seguido por terminales especiales (Delimitador). También se puede especificar si el componente es de valor único o soporta una lista de posibles valores. Además, es posible indicar jerarquías entre los componentes, a través de relaciones de composición e inclusión. En una relación de composición, un componente puede estar compuesto por otros sin límite en el nivel de anidamiento (Grupo); en una relación de inclusión, el valor de un componente (Contenido) está supeditado a la existencia de un valor para el componente que lo contiene.

[3] Los nombres que identifican las clases del meta-modelo tienen una fuerte influencia de los utilizados en la *Requirements Baseline*, ya que el mismo fue creado, en primera instancia, para soportar a la misma.

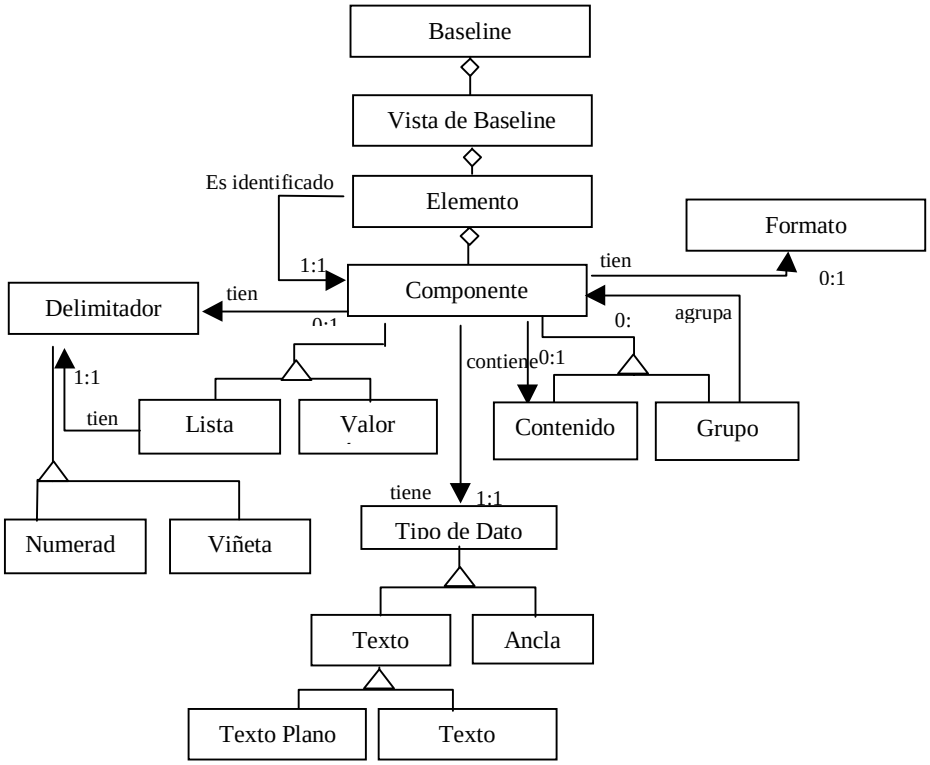

Fig. 1 - MAB, Modelo Abstracto de la Baseline

Volviendo al ejemplo, todos los componentes de un símbolo del LEL son textuales, y con excepción de la categoría, admiten una lista de valores. En el caso de la lista de nombres del símbolo, la metodología presentada en la sección 2, indica que deben separarse mediante "/" (Delimitador) y que se escriben en minúsculas (Formato).

3.2 MAP, Modelo Abstracto del Proyecto

MAP define el conjunto de clases que almacenan la información contenida en los documentos de requisitos. Las clases en este modelo están estructuradas de forma similar a las del MAB. Así, hay un proyecto que agrupa todos los datos que se tienen para un cierto dominio. Cada proyecto está compuesto por vistas de proyecto que se corresponden unívocamente con las vistas definidas en el MAB. Cada vista del proyecto está compuesta por un conjunto de entradas, donde cada entrada representa una instancia de documento que contiene datos del dominio. Por ejemplo, toda la información relevada para el caso del Circulo de Ahorro para Adquisición de un automóvil(Rivero et al. 1998) constituye un proyecto, y los símbolos del LEL adjudicatario, solicitante y cuota son entradas de la vista de proyecto LEL.

3.3 MAN, Modelo Abstracto de la Navegación

MAN estructura la navegación en contextos de navegación, adoptando una versión reducida de los contextos navegacionales presentados en OOHDM(Rossi 1996). En esta propuesta, un contexto de navegación organiza el espacio de navegación en conjuntos consistentes de entradas, que pueden ser recorridas siguiendo un orden en particular. Puede ser definido indicando la condición que deben cumplir sus miembros o enumerando cada uno de ellos. Cada contexto de navegación definido permite explorar un tipo de documento, por lo que está asociado a una vista del MAB. Para evaluar un contexto de navegación se recorrerán las entradas de una vista del proyecto actual (MAP) para determinar si verifican una propiedad que puede ser referida por estar definida en la vista MAB de referencia. Así, el contexto de navegación "Sujetos del LEL" está compuesto por todos los símbolos del LEL en el Proyecto actual, cuya componente categoría indique que es un sujeto.

3.4 MAI, Modelo Abstracto de la Interface

El MAI modela la forma en que se muestran los documentos al usuario. Está compuesto por clases que definen esa presentación y clases que administran la interacción entre el usuario y los componentes gráficos de la presentación. Estas clases permiten desde la definición de ventanas complejas que permiten editar documentos completos hasta indicar qué acciones se llevan a cabo al presionar un botón. Entre las clases que definen la presentación se encuentran las Ventanas, compuestas por jerarquías de componentes gráficos que actúan de nexo entre los componentes del MAB y el usuario. Las acciones que pueden ejecutarse vía esta interface pueden estar relacionadas con las vistas, elementos y componentes del MAB, así como con el proyecto en el MAP.

4 Instanciación de MENDoR para la Creación de HeAR

El metamodelo MENDoR fue pensado para administrar documentos de requisitos en forma genérica. Adaptarlo para que se ajuste a un tipo de documento en particular requiere de un proceso de instanciación que consiste en la asignación de valores a los atributos de las clases que componen cada modelo. Las actividades de definición del documento, su presentación y navegación están estrechamente relacionadas unas con otras y pueden secuenciarse en un proceso que consta de siete pasos.

La definición de la estructura de un documento instancia clases del MAB. Esto se lleva a cabo mediante los siguientes pasos:

1. Definición de la vista que contendrá al documento.
2. Definición de cada documento.
3. Definición de los componentes de cada documento.

Las navegaciones y verificaciones previstas sobre el documento también deberán ser indicadas específicamente, lo que producirá nuevas instancias de clases del MAN. Entonces, será necesario:

4. Definición los contextos de navegación deseados.

La definición de la presentación e interacción con el usuario instancia clases del MAI. Para ello deberán cumplirse los siguientes pasos:

5. Definición de las ventanas que presentan el documento y sus componentes.
6. Definición de los servicios (acciones) a los que responderán las vistas, los documentos y sus componentes.
7. Asociación entre las acciones y navegaciones definidas en los pasos 6 y 4 con sus accesos vía componentes gráficos definidos en 5.

HeAR es una herramienta de adquisición de requisitos que permite documentar la etapa de requisitos de un sistema a desarrollar, de acuerdo a la metodología presentada en la sección 2. HeAR es una instanciación de MENDoR. En este caso particular, se eligió que la instanciación la realizara el mismo prototipo, tomando los valores particulares previamente almacenados en el repositorio de datos. En las tablas del repositorio se encuentran tanto la especificación de los documentos soportados (datos para la instanciación del MAB) y sus interfaces a usuario (datos para la instanciación del MAI), como la información relevada para el dominio de aplicación (datos correspondientes al MAP que la misma herramienta registra). Las instancias de las clases que definen los documentos y sus presentaciones se crean dinámicamente, en tiempo de ejecución, a partir de su especificación en las tablas del repositorio de datos.

5 El Prototipo HeAR

En esta sección se describe la funcionalidad de HeAR y algunos datos de su implementación. El primer paso para usar HeAR es la definición del nombre de un proyecto, ya que la información almacenada para un determinado dominio de aplicación constituye un proyecto. Cada proyecto está compuesto por un conjunto de símbolos y un conjunto de escenarios, vinculados en una red hipertextual. Para documentar un nuevo dominio, el usuario deberá crear un nuevo proyecto e ingresar sus símbolos y escenarios. A medida que ingresa información, la herramienta detectará las referencias a símbolos y escenarios ya existentes y las agregará como vínculos. Estos vínculos podrán ser navegados dentro del ámbito de la herramienta. También se proveen facilidades de consulta y navegación entre los símbolos y escenarios ingresados en forma de contextos de navegación.

En cualquier momento el usuario podrá exportar la información ingresada en forma de documento hipertexto, compatible con cualquier browser o navegador. Para ilustrar esta sección, se utiliza como ejemplo el caso de estudio del Circulo de Ahorro para Adquisición de Automóvil. Cuando se inicia HeAR, luego de la pantalla de presentación aparece la ventana principal o menú principal que se muestra en la Fig. 2. La barra de título de la ventana muestra el nombre del proyecto abierto. Desde esta ventana se pueden administrar los proyectos, las vistas que los componen y realizar consultas sobre ellas.

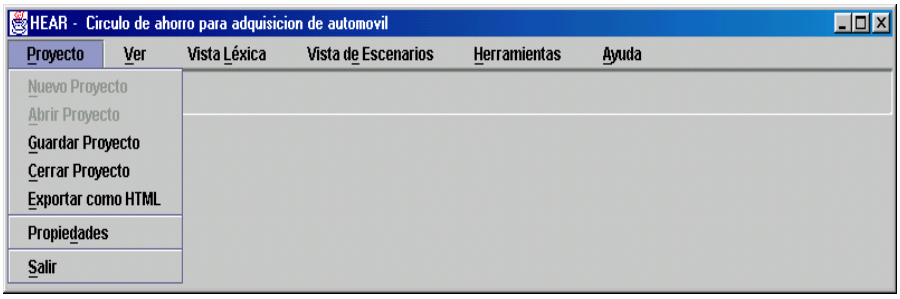

Fig. 2 - Menú Principal de HeAR.

La información para las vistas LEL y SMV se presenta de modo similar, a través de un par de ventanas como las que se muestran en la Fig. 3. En esta figura puede verse una ventana de índice (a la izquierda) que muestra la lista de símbolos que han sido ingresados para el proyecto actual, que podrán ser navegados, ordenados según distintos criterios y seleccionados para ser mostrados en detalle. La ventana de Detalle asociada (a la derecha) muestra la información ingresada para el símbolo seleccionado.

5.1 Capacidades de Edición

Estas capacidades comprenden la creación de símbolos y escenarios, la definición de los vínculos no automáticos entre ellos, así como el ingreso, modificación o eliminación de información sobre ellos. Mientras se esté en modo de edición, no será posible navegar los vínculos entre las entradas. Para salir del modo de edición, el usuario deberá aceptar o cancelar la edición en curso. Toda la edición se accede a través de menús contextuales. Uno de ellos se muestra en la Fig. 3, y corresponde a la Ventana Indice. La edición propiamente dicha se realiza en ventanas de edición separadas para cada uno de los componentes. La Fig. 4 muestra un ejemplo de ventana de edición, en este caso correspondiente a una noción. A medida que el usuario ingresa el texto, la herramienta va detectando automáticamente las referencias a símbolos y escenarios existentes y las identifica como vínculos. En la misma figura se observa que se han reconocido dos símbolos.

La herramienta también permite seleccionar una frase de texto e indicarla como símbolo o escenario pendiente de edición, que podrá completarse luego de finalizada la edición en curso. Otra facilidad provista es la edición de vínculos definidos por el usuario entre el elemento mostrado en la ventana de detalle y cualquier otro elemento existente. Para la creación de estos vínculos se dispone de una ventana especial, que permite seleccionar el destino entre todos los posibles.

5.2 Capacidades de Navegación

Abarca la navegación entre las entradas que componen la red hipertextual. Siempre que no haya una edición en curso, es posible navegar entre los vínculos que relacionan a los símbolos y escenarios existentes. Hay distintos tipos de navegaciones posibles. Una de ellas es a través de clicks sobre las cadenas de texto que se comportan con el estilo clásico de los vínculos hipertextuales en un browser. En este

caso, la navegación se realiza hacia el elemento cuyo identificador es la cadena de texto en el vínculo.

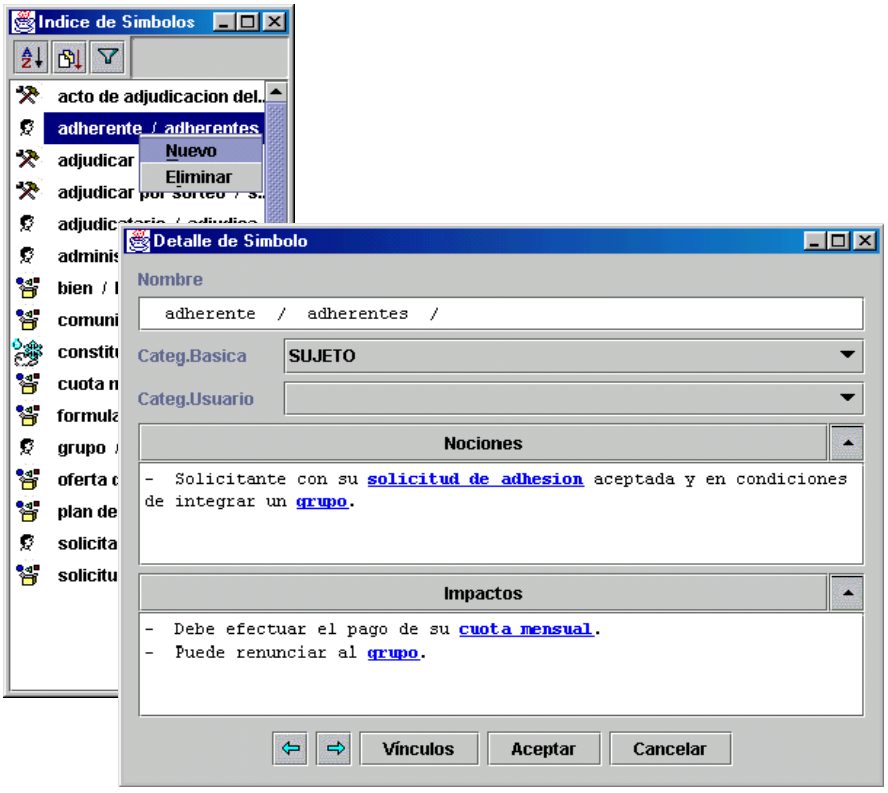

Fig. 3 - Ventana Indice de Símbolos del LEL y Ventana de Detalle del símbolo seleccionado.

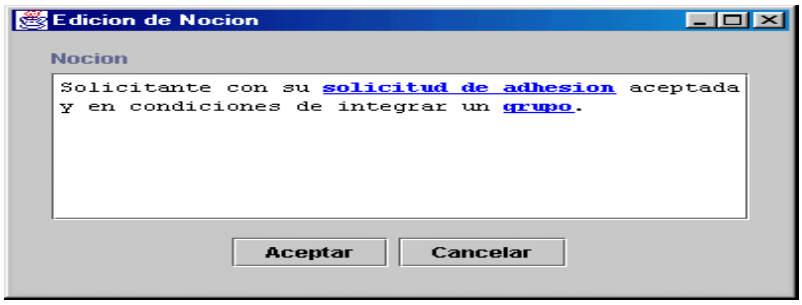

Fig. 4 - Ventana de Ingreso y Edición de Nociones.

Los vínculos definidos especialmente a pedido del usuario se muestran en una ventana especial asociada a la ventana de detalle y su estilo indica que se los navega como si fueran referencias al destino.

Otro tipo de navegación disponible es la que se da entre símbolos o escenarios en un índice o entre los resultantes de una consulta predefinida en el ambiente, ya que éstos se muestran igualmente en una ventana índice con la indicación de que su contenido está filtrado por algún criterio.

En todos los casos la secuencia de navegación es registrada, constituyendo un contexto de navegación, y pueden utilizarse los botones anterior/siguiente presentes en las ventanas de detalle para recorrerla. La selección de opciones que permiten ingresar un nuevo símbolo o escenario, o alguno de sus atributos, lleva al modo de edición abandonando el modo navegación.

5.3 Consultas Predefinidas

Las consultas predefinidas permiten evaluar la completitud de la información contenida en las distintas entradas, detectar errores u omisiones, identificar entradas parcialmente definidas e inspeccionar el conjunto de escenarios siguiendo las heurísticas de inspección de escenarios. Las Fig. 5 y 6 muestran algunas de las consultas y verificaciones disponibles para cada una de las vistas.

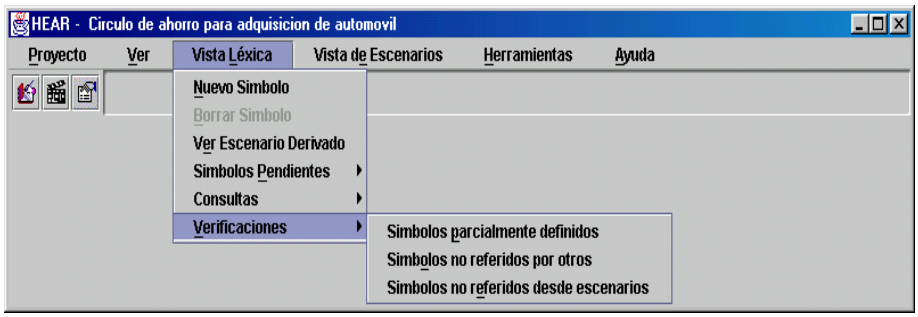

Fig. 5 - Verificaciones disponibles para la vista léxica.

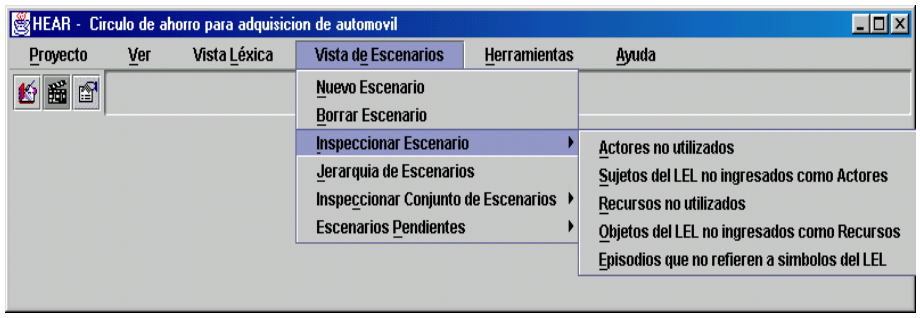

Fig. 6 - Consultas que inspeccionan la composición de un escenario.

5.4 Otras Funciones

La herramienta provee además servicios especiales que implementan heurísticas sugeridas en distintos trabajos de investigación relacionados con la metodología adoptada. Es el caso de la definición de categorías especiales para ciertos dominios y la derivación de escenarios a partir de la vista léxica. Otro servicio permite disponer de la información relativa a un proyecto fuera del ámbito de la herramienta, esto se hace exportando las vistas como archivos HTML.

Definición de Categorías Especiales para los Símbolos del LEL. Esta opción permite agregar categorías especiales de acuerdo al Universo del Discurso que se esté relevando para clasificar los símbolos de la vista léxica. Estas categorías pueden ser refinamientos de las categorías básicas, o bien ser completamente ortogonales a dicha clasificación. Las categorías definidas de esta manera para un proyecto no están disponibles para otros proyectos.

Derivación de Escenarios a Partir del LEL. La herramienta realiza la tarea de sugerir escenarios candidatos a partir de los impactos de los símbolos en el LEL clasificados como sujetos, siguiendo las heurísticas de la metodología presentada en la sección 2. Los nuevos escenarios se muestran en la ventana índice correspondiente y estarán disponibles para ser editados, al igual que cualquier otro escenario de la vista.

Exportación del Conjunto de Documentos como Archivos HTML. Es posible exportar el proyecto actual en formato HTML. Esta operación genera dos archivos con ese formato, uno para cada una de las vistas, y un tercero más general que contiene los datos del proyecto y actúa de índice para los otros. La Fig. 7 muestra que los archivos obtenidos pueden ser navegados con cualquier browser.

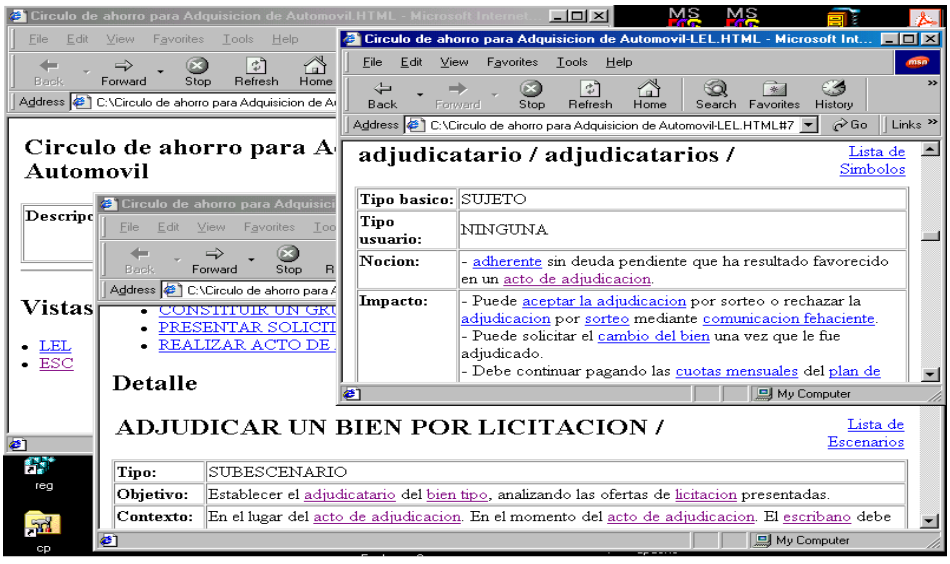

Fig. 7 - Navegación de los documentos exportados desde un browser.

5.5 Implementación

El prototipo HeAR fue implementado en lenguaje Java; en particular, para la interface se utilizó componentes gráficos del paquete Swing. Los datos que definen la metodología y la interface a usuario se almacenan en un repositorio de datos, en este caso MySQL, un servidor de bases de datos relacionales. Las clases Java acceden a los datos mediante lenguaje SQL y el protocolo de comunicación de bases de datos JDBC.

La Fig. 8 esquematiza la arquitectura de la aplicación. En ella pueden verse, además de la plataforma seleccionada, los principales subsistemas y sus interacciones, que siguen el patrón definido por el metamodelo MENDoR, descripto en la sección 3.

El subsistema Baseline implementa las clases propuestas en el MAB, que se instancian a partir de la especificación de los documentos almacenada en la base de datos. Por lo tanto, HeAR soportará un nuevo tipo de documento con sólo definirlo en los lugares necesarios de la base de datos. Este subsistema también contiene las clases que componen el MAN, es decir, cada uno de los contextos de navegación especificados. Cuando se desee incluir un nuevo contexto de navegación deberá proveerse una mínima cantidad de código que defina la condición a cumplir.

El subsistema Proyecto implementa el MAP y se encarga de leer y guardar en la base de datos la información correspondiente al dominio.

El subsistema Interface implementa el MAI, proveyendo ventanas, barras de menú y las acciones que se ejecutan para cada opción. La composición de cada ventana y los componentes gráficos que contiene, están especificados en el repositorio de datos, y existen clases que se ocupan de crearlos a partir de esta especificación.

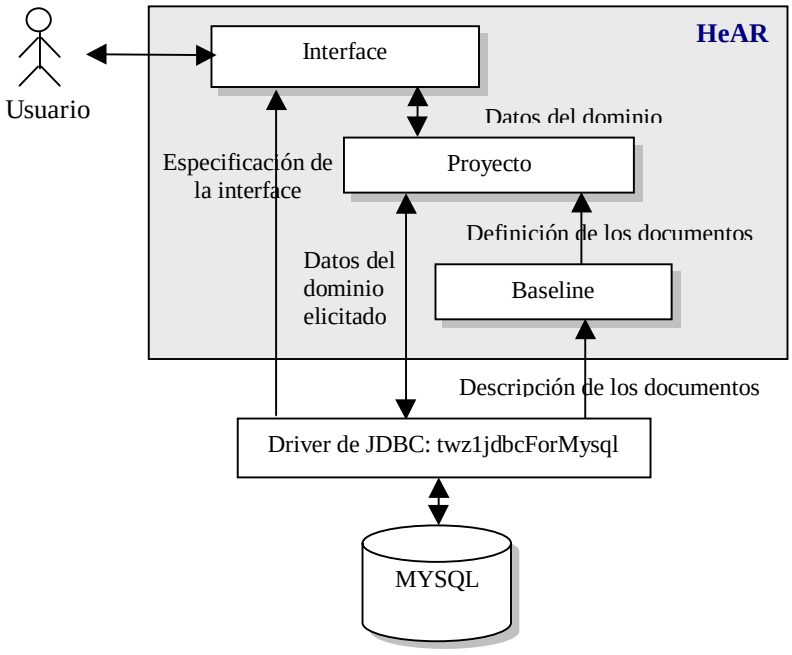

Fig. 8 - Arquitectura del prototipo HeAR.

HeAR fue desarrollada bajo MS Windows 98. Se compone de 6 packages Java más los datos de instanciación de MENDoR que están almacenados en tablas en el repositorio de datos. El número de clases es 216, incluyendo todo el metamodelo MENDoR, clases específicas de HeAR, y ciertas clases que fueron necesarios para presentación de textos, y presentación de la vistas en HTML. En total consta de aproximadamente 25000 líneas de código Java, más 22 tablas en el repositorio de datos que contienen los datos de esta instanciación en particular. El repositorio contiene 8 tablas adicionales destinadas a contener la información de los proyectos que se procesen.

6 Comparación con otras Herramientas Similares

Se estudiaron dos prototipos de ambiente para la adquisición de requisitos: el del proyecto ReCoRD, desarrollado en la Universidad de Umeå, Suecia (Boman and Sigerud 1996), y la herramienta Baseline Mentor (Antonelli et al. 1999), desarrollada en la Universidad de La Plata.

La primera herramienta, llamada Requirements Collection Reuse and Documentation, utiliza casos de uso similares a los propuestos por (Jacobson et al. 1992) para relevar los requisitos, que se complementan con un diccionario de términos. Mediante un análisis sintáctico de los casos de uso se genera un modelo de objetos. No siendo esta la finalidad de HeAR, se estudió la interface de la herramienta. Esta, que se realiza a través de formularios que soportan el ingreso estructurado de los requisitos, resulta muy compleja ya que presenta muchos formularios juntos en pantalla. Por otra parte, aunque el prototipo genera documentos hipertextuales, no es posible navegar entre los casos de uso, ni hacia palabras del diccionario durante la edición de los mismos.

La herramienta Baseline Mentor está basada en la misma metodología en que se basa HeAR. Sin embargo, esta herramienta tiene como propósito la derivación de especificaciones orientada a objetos a partir de los escenarios y LEL. El prototipo permite la gestión de proyectos compuestos por un conjunto de símbolos LEL, un conjunto de escenarios y un conjunto de tarjetas CRC, y sus versiones. Las entradas para los dos primeros conjuntos deben ser ingresadas manualmente en formularios, mientras que las tarjetas CRC (no editables) son obtenidas automáticamente mediante la aplicación de heurísticas de derivación propuestas en (Leonardi et al. 1997). Por tratarse de una experiencia anterior a algunos trabajos de investigación realizados sobre la metodología de referencia, Baseline Mentor no incluye resultados que sí han sido considerados en HeAR.

7 Conclusiones y Trabajo Futuro

Este artículo presenta HeAR, un prototipo de herramienta de adquisición de requisitos que asiste al ingeniero de software en la tarea de documentar la etapa inicial del desarrollo de software. HeAR acompaña al usuario en la aplicación de la *Requirements Baseline*, ayudándole a seguir las reglas de escritura que ésta prescribe, relevándolo de las verificaciones sintácticas más tediosas, manteniendo vínculos entre los documentos obtenidos que podrá navegar, y permitiéndole acceder a la

información según diversos criterios. HeAR aún no se ha probado exhaustivamente. El caso de estudio mencionado (Rivero et al. 1998) y utilizado como ejemplo en el artículo está integrado por una vista léxica de 49 símbolos principales, con un promedio de 2 nociones y 4 impactos cada uno; una vista de escenarios compuesta por 18 escenarios principales y 16 subescenarios. La media de episodios por escenario principal es de 5, mientras que el número de actores y recursos involucrados es de 3 en ambos casos.

MENDoR no trata aspectos de gerencia de configuración. La misma será soportada a través de la definición de la vista de configuraciones ortogonal a las demás vistas(sección 2), que tendrá características que la distinga de las otras vistas. La visualización del versionado de una vista se realizará a través de contextos navegacionales predefinidos. Para realizar un soporte completo de traceability, se debe instanciar MENDoR con modelos que correspondan a las etapas posteriores de desarrollo de software. La única traceability que actualmente soporta HeAR es entre el modelo de escenarios y el de LEL (a partir de la estrategia de creación de los escenarios) y esta totalmente implementada través de un contexto navegacional predefinido.

MENDoR puede incluir fácilmente otros documentos textuales de especificación de requisitos, diferentes de aquellos con los que cuenta en este momento. En este punto, pueden distinguirse mejoras y extensiones por realizar al prototipo presentado:

- Una de ellas involucra el desarrollo de una herramienta complementaria para definir y/o modificar la especificación de los documentos y sus interfaces a usuario. Aunque esto es posible actualmente, para ello es necesario insertar la información en la base de datos usando SQL.
- Otro objetivo es contar con una herramienta que permita estudiar aspectos de la adquisición de requisitos en forma colaborativa. Este objetivo fue tenido en cuenta parcialmente al seleccionar la plataforma de implementación; sin embargo, es necesario estudiar el tema y realizar las modificaciones necesarias.

En síntesis: HeAR es un prototipo de soporte de especificación de documentos hipertextuales de requisitos. Se espera que contribuya, por una parte, a facilitar la tarea rutinaria de mantenimiento de la información relevada y modelada de acuerdo al patrón de los documentos especificados, y, por otra parte, que por sus características de adaptabilidad con bajo costo de esfuerzo y tiempo, permita a los investigadores explorar nuevas posibilidades.

Referencias

Antonelli L., Rossi G., Oliveros A., “Baseline Mentor, An application that derives CRC Scenarios.” Anales Anales de WER 99, 28 JAIIO, septiembre de 1999, pp. 5-16.

Boman T., Sigerud K. “Requirements Elicitation and Documentation Using T-Red” Master’s Thesis, Umeå University, May 1996. http://www.cs.umu.se/~record/T-red/master.html)

Doorn J., Kaplan G., Hadad G., Leite J., “Inspección de escenarios” Anales de WER98,

Sbes98, Maringa, Brasil, 13 de Octubre de 1998. Pp. 57-69.

Hadad G., Kaplan G., Oliveros A., "Uso del Léxico Extendido del Lenguaje y de Escenarios para la elicitación de Requerimientos: Aplicación a un Caso Real." Informe de investigación, Departamento de Investigación, UB, 1996.

Hadad G., Kaplan G., Oliveros A., Leite J., "Construcción de Escenarios a partir del Léxico Extendido del Lenguaje" 26 JAIOO, Buenos Aires, 1997. pp.65-77.

Jacobson I., Christerson M, Jonnson P, Overgaard G. "Object Oriented software engineer: A Use-Case driven approach" Addison Wesley, 1992.

Leite J.C.S.P, Franco A "O Uso de Hipertexto na Elicitaçao de Linguagens da Aplicaçao" Anais de IV Simpósio Brasileiro de Engenheria de Software, octubre de 1990. pp.134-149.

Leite J.C.S.P., Oliveira A., "A Client Oriented Requirements Baseline" Proceedings of the Second IEEE International Symposium on Requirements Engineering, IEEE Computer Society Press, 1995. pp. 108-115.

Leite J.C.S.P., Rossi G., Maiorana V., Balaguer F., Kaplan G., Hadad G., Oliveros A., "Enhancing a Requirements Baseline with Scenarios". Requirements Engineering Journal: Vol. 2, Nº4 1997, Springer-Verlag, London. pp. 184–198.

Leite J, Hadad G, Doorn J, Kaplan G. "A Scenario Construction Process" a aparecer en Requirements Engineering Journal, Springer-Verlag.

Leonardi, C., Maiorana, V., Balaguer, F., "Una estrategia de Análisis Orientada a Objetos basada en escenarios" Actas de II Jornadas de Ingeniería del software, JIS97, Dpto. de Informática, Universidad del País Vasco, San Sebastián 1997, pp. 87-100.

Mamani Macedo N., Leite J.C.S.P., "Elicit@99: Um Prototipo de Ferramenta para a Elicitação de Requisitos". Anales de WER 99, 28 JAIIO, septiembre de 1999, pp. 45-55.

Petersen L., Tornabene S., "HeAR: Una Herramienta de Adquisición de Requisitos". Trabajo de Graduación, Facultad de Cs. Exactas, UNCPBA, Tandil, Argentina, diciembre de 1999.

Rivero L, Doorn J, del Fresno m, Mauco V, Ridao M, Leonardi C. "Una Estrategia de Análisis Orientada a Objetos basada en Escenarios: Aplicación en un Caso Real" Anales de WER 98, Sbes98, Maringa, Brasil, 13 de Octubre de 1998. Pp 79-88.

Rossi, G., "An Object Oriented Method for Designing Hypermedia applications". PHD Thesis, Departamento de Informatica, PUC-Rio, Brasil, julio de 1996.