INSPECCIÓN DE ESCENARIOS

Jorge Horacio Doorn1 Gladys Kaplan2 Graciela Hadad2 Julio Cesar Sampaio do Prado Leite3

1 Universidad de Belgrano y Univ. Nac. del Centro de la Prov. de Bs. As.,
email:jdoorn@exa.unicen.edu.ar

2 Departamento de Investigación, Universidad de Belgrano,
email:gkaplan@ub.edu.ar , ghadad@ub.edu.ar

3 Departamento de Informática, Pontificia Universidad Católica de Río de Janiero,
email:julio@inf.puc-rio.br

www.ub.edu.ar

Resumen. La cercanía de los escenarios con situaciones concretas y vitales ayuda a los clientes/usuarios y a los ingenieros de software en el desarrollo de una comprensión compartida de las necesidades. Este fenómeno se incrementa significativamente al construir los escenarios utilizando lenguaje natural. La flexibilidad que ofrece el uso del lenguaje natural mejora la interacción de los participantes en el proceso de captura de los requisitos, pero agrega una importante dificultad a la hora de validar los mismos, ya que la presencia de errores, discrepancias u omisiones es más difícil de detectar. Por otra parte, el contar con un conjunto de requisitos lo más consistente y completo posible es una necesidad creciente en la ingeniería de software. En el presente artículo se propone un conjunto de heurísticas para facilitar la detección de errores. Estas heurísticas consisten básicamente en la realización de verificaciones cruzadas entre los escenarios y el Léxico Extendido del Lenguaje y entre los escenarios mismos.

Palabras clave: Escenarios, Léxico, Requisitos, Elicitación, Inspección.

Abstract.The closeness of the scenarios with concrete situations helps clients/users and requirements engineers to develop a shared understanding of the necessities. This phenomenon is significantly increased when scenarios are built using natural language. The flexibility offered by the natural language improves the interaction of the participants in the process of requirements capture, however it adds one important difficulty when the time to validate them arrives, since the discrepancies, omissions and errors are hard to detect. On the other hand to rely on requirements as much consistent and complete as possible is a growing necessity in the current software engineering. In the present article a set of heuristics to ease the detection of errors in scenarios is proposed. These heuristics basically consist on a set of cross checks between the scenarios and the Language Extended Lexicon, and between the scenarios themselves.

Keywords: Scenario, Lexicon, Requirements, Elicitation, Inspection.

1. INTRODUCCIÓN

Generalmente la clave del éxito o fracaso de un proyecto de software depende de resolver el problema correcto, por lo que es muy importante capturar los requisitos correctos. Estos deben contemplar las necesidades del cliente, las reglas de la organización y las reglas externas próximas al macrosistema. Habitualmente cuando se hace referencia a errores en los requisitos de los sistemas, se desliza en forma tácita la idea que el profesional de software o el proceso de captura han sido la fuente y origen de los mismos. Esta suposición, si bien muy difundida, probablemente sea falsa ya que muchos de los errores, discrepancias y omisiones que aparecen en los requisitos y naturalmente se propagan al sistema final, ya existían en la fuente de información; el verdadero problema es la falencia de herramientas, procedimientos y estrategias que permitan detectarlos a tiempo. Los errores en los requisitos incrementan fuertemente los costos de desarrollo y mantenimiento de los sistemas. Cuanto más tardía es su detección, más alto es el costo de corrección. La incidencia del costo de corrección de requisitos erróneos ha sido ampliamente estudiada por autores como [Davis 93], [Bell 76] y otros.

En respuesta a la baja calidad de los productos de software con altos costos de corrección y mantenimiento, se tornó obvia para muchos desarrolladores de software e investigadores la necesidad de profundizar en el proceso previo a la definición del software. Ellos apuntan a garantizar la producción de software de alta calidad y bajo costo basándose en el establecimiento de un punto de partida de alta confiabilidad, el que se logra mediante la detección lo más temprana posible de errores, omisiones, y discrepancias.

Por otra parte la ingeniería de requisitos ha enfatizado la necesidad de que los requisitos no sólo sean de la mayor calidad posible en el comienzo del proceso de desarrollo, sino que también se conserve la calidad a lo largo de todo el desarrollo. Para lograr esto, es necesario que los requisitos puedan evolucionar registrando los cambios que la realidad impone al desarrollo de la aplicación. Por esta razón, se pone énfasis en especificaciones dinámicas de requisitos precisos y claros que acompañen los cambios del macrosistema y que permitan el seguimiento de los mismos desde cualquier punto del proceso de producción del software hasta sus orígenes. Se puede decir que la sistematización del proceso de detección de errores ayuda considerablemente en el proceso de especificación, y esto se refleja en todo el proceso de construcción del software.

Varios mecanismos se han propuesto para formalizar la obtención de requisitos tales como los presentados en [Jacobson 92], [Reubinstein 91], [Bubenko 93], [Macaulay 93] y otros. El modelo [Leite 95] está basado en la existencia de un andarivel denominado Requirements Baseline, que utiliza una estructura dinámica basada en el dominio del problema. La misma permite conocer y registrar los orígenes de los requisitos y evoluciona durante todo el proceso de desarrollo del producto de software. El Requirements Baseline utiliza varios modelos tales como los Escenarios, el Léxico Extendido del Lenguaje [Leite89] y el Modelo Básico.

El presente artículo está basado en el uso de los Escenarios dentro del marco del Requirements Baseline. Los escenarios permiten describir situaciones de un problema, utilizando, en este marco, el lenguaje natural. Varios autores han estudiado esta técnica, entre ellos [Potts 94] [Booch 91] [Jacobson 92] [Carrol 95] [Zorman 95].

Recientes estadísticas [Weidenhaupt 98] acerca del uso de escenarios en la realidad productiva han

comprobado que los mismos requieren esfuerzos significativos y que es necesario disponer de guías y herramientas para su administración.

Por otro lado, la ingeniería de software ha utilizado ampliamente las inspecciones como técnica de verificación y validación, primero de programas y luego de otros documentos creados durante el proceso de desarrollo de software [Fagan 76][Barnard 94]. En el presente trabajo se proponen procedimientos y métodos para guiar la inspección de un conjunto de escenarios y reducir en el mayor grado posible los errores, discrepancias y omisiones de los mismos. Estos procedimientos fueron elicitados a partir de experiencias y casos reales como parte de una investigación más amplia que se encuentra en desarrollo.

La sección 2 detalla el Requirements Baseline. La sección 3 describe las posibles fuentes de dificultades implicadas en el proceso de construcción de escenarios. La sección 4 define la semántica de la validación de escenarios. La sección 5 especifica la inspección de escenarios propuesta y, finalmente en la sección 6 se incluyen las conclusiones a las que se ha arribado.

2. REQUIREMENTS BASELINE

El Requirements Baseline contiene descripciones sobre el dominio del problema y el artefacto de software ha ser construido dentro de ese dominio. El mismo es una estructura dinámica generada durante el proceso de ingeniería de requisitos, que evoluciona junto al proceso de desarrollo del software y que acompaña las tareas de mantenimiento.

Se debe enfatizar la importancia de la capacidad de referencias cruzadas entre el Requirements Baseline y el proceso de desarrollo y mantenimiento, así como la existencia de una referencia cruzada dentro del propio Requirements Baseline. Esto permite el seguimiento de los requisitos en cualquier punto durante el desarrollo y el mantenimiento hacia sus orígenes.

El Requirements Baseline está compuesto por cinco vistas que complementariamente satisfacen la información necesaria en cada etapa del proceso de desarrollo. Estas cinco vistas son: la vista del modelo Léxico, la vista del modelo de Escenarios, la vista del modelo Básico, la vista de Hipertexto y la vista de Configuración.

Es importante resaltar que las vistas de Hipertexto y de Configuración son ortogonales a las otras tres. Son un soporte de servicios indispensable para garantizar el seguimiento de los requisitos (vista de Configuración) y el acceso a la información almacenada (vista de Hipertexto).

La vista Léxica está basada en el Léxico Extendido del Lenguaje (LEL) que es un meta modelo diseñado para ayudar a la elicitación del lenguaje usado en el macrosistema. La vista de Escenarios describe situaciones del comportamiento de la aplicación en un momento específico. La vista Básica está soportada por diagramas de entidad-relación que representan los requisitos externos propuestos por el cliente del macrosistema. La vista de Hipertexto trabaja como un integrador de la vista Léxica, la vista de Escenarios y la vista Básica, habilitando la definición de vínculos dentro de la misma vista y entre las tres vistas. Las descripciones de los escenarios tienen un fuerte vínculo con el LEL, ya que adoptan este léxico como referencia. La vista de Configuración es un sistema de versiones que lleva la historia de los cambios en la vista Léxica, en la vista de Escenarios y en la vista Básica durante el desarrollo y el mantenimiento. Esta vista muestra la evolución del sistema de software.

2.1 LEL, la vista Léxica

La vista Léxica se basa en el Léxico Extendido del Lenguaje que es un modelo que ayuda a conocer el lenguaje propio del dominio del problema. Este léxico está construido utilizando el lenguaje natural y está compuesto por símbolos que representan palabras o frases que el cliente repite o enfatiza, también se incluyen las palabras o frases relevantes para el dominio del problema más allá de su frecuencia de repetición. Durante la recolección de símbolos el ingeniero de software identifica el nombre de cada símbolo y sus sinónimos, luego describe en la noción "qué es" y en el impacto "cómo repercute en el sistema" [Leite 90]. El léxico debe cumplir con los principios de "circularidad" y de "vocabulario mínimo".

2.2 La vista de Escenarios

El principal objetivo de los escenarios, durante la etapa de elicitación de requisitos, es comprender el problema en su totalidad. En las siguientes etapas acompaña el proceso de desarrollo del software describiendo aspectos de diseño y de código y la representación de modificaciones a requisitos existentes y la representación de requisitos nuevos.

Entre los escenarios pertenecientes a un mismo problema existe una relación semántica [Booch 94] y es esta relación semántica la que es aprovechada por la técnica de inspección que se propone.

Los escenarios tienen una estructura compuesta por el título, el objetivo, el contexto, los recursos, los actores y los episodios. El objetivo, el contexto, los recursos y los actores son sentencias declarativas, mientras que los episodios son un conjunto de sentencias con un lenguaje muy simple que hace posible la descripción operativa de comportamientos.

Un episodio puede concebirse como un escenario en sí mismo, esto posibilita la descomposición de un escenario en subescenarios. Para la descripción de escenarios se utilizó el esqueleto presentado en la Figura 1 basado en [Leite 98].

En estudios anteriores [Hadad 97] se han propuesto heurísticas para derivar los escenarios desde el LEL. Estas heurísticas permiten generar una primera versión de los escenarios. Se detectan los actores en el LEL y se utilizan sus impactos y sus vínculos para describir parcialmente los escenarios. Luego, se completan los escenarios validándolos con el usuario y ampliando la información faltante o confusa.

3. CONSTRUCCIÓN DE ESCENARIOS

Durante la construcción de escenarios, existen factores personales de los participantes del proceso, de la naturaleza del problema y de las fuentes de información, que dificultan, facilitan o sesgan el resultado obtenido. A los participantes les resulta difícil percibir sus propios errores o desvíos, por lo que la existencia de procedimientos de verificación y validación es de gran ayuda.

Los principales factores implicados en el proceso de construcción de escenarios son:

* El punto de vista del ingeniero de requisitos
* El conocimiento y experiencia de los participantes
* El tipo de fuentes de información
* El número de fuentes de información
* Los mecanismos de adquisición de la información
* Los niveles de abstracción envueltos en el proceso

* Las características del macrosistema

Los puntos antes mencionados pueden producir diferentes dificultades, tales como:

* Escenarios con diferentes niveles de detalle
* Escenarios con información faltante
* Escenarios con información errónea
* Escenarios con información ambigua
* Escenarios con conflictos
* Situaciones nunca descriptas
* Escenarios parcial o totalmente superpuestos
* Carencia de una visión integral de la aplicación
* Carencia de escenarios relacionados

En las secciones siguientes, se analizan estos problemas y se proponen mecanismos de detección.

4. SEMÁNTICA DE VALIDACIÓN DE ESCENARIOS

Un análisis de la terminología usada [Loucopoulos 95] [Jackson 95] [Zave 97], tales como contradicciones, discrepancias, errores y otros, requiere una definición más precisa antes de proponer mecanismos de detección y corrección.

En este artículo se han utilizado las siguientes definiciones:

Discrepancia: los ítems a comparar difieren.

Inconsistencia: los ítems a comparar establecen hechos diferentes.

Omisión: falta de información en uno o más ítems.

Error: al menos uno de los ítems establece algo falso.

Las palabras Conflicto y Contradicción son interpretadas con el mismo significado que Inconsistencia. La palabra Equivocación y la frase Información Errónea son utilizadas como sinónimo de la palabra Error. Las frases Información Ausente e Información Faltante son utilizadas como Omisión.

MODELO FORMAL DE ESCENARIOS

Título: nombre que identifica al escenario

Actores: agente que tiene un rol en el escenario.

Sintaxis: Frase|([Actor|Recurso] + Verbo + Predicado)

Sintaxis: Nombre.

Objetivo: finalidad a ser alcanzada

Episodios: acciones que detallan el escenario.

Sintaxis: [Sujeto] + Verbo + Predicado

Sintaxis:

Contexto: compuesto por los siguientes ítems:

<episodios> ::= <series>

Ubic. Geográfica: lugar donde se produce el escenario.

<series> ::= <sentencia> | <series> <sentencia>

Sintaxis: Nombre

<sentencia> ::= <sentencia secuencial> |

Ubic. Temporal: momento en que ocurre el escenario.

< sentencia no secuencial> |

Sintaxis: Nombre

<sentencia condicional> | <sentencia optativa>

Precondición: estado inicial del escenario.

<sentencia secuencial> ::= <sentencia episodio>CR

Sintaxis: [Actor | Recurso] + Verbo + Predicado +

{Restricción}

<sentencia condicional> ::= Si <condición>

entonces <sentencia episodio>CR

Recursos: elementos pasivos necesarios.

<sentencia no secuencial> ::= # <series> #

Sintaxis: Nombre + {Restricción}

<sentencia optativa> ::= [<series>]

<sentencia episodio> ::=[Actor | Recurso] + Verbo +

Predicado + {Restricción} + {Excepción}

+ composición, {x}0 a n ocurrencias de x, () para agrupar, | para or y [x] significa que x es opcional.

Figura 1 – Modelo Formal de Escenarios

Se debe prestar especial atención a la palabra Ambigüedad, ya que la misma muestra la existencia de una falta de detalles. Una ambigüedad es una sentencia que puede ser entendida de diferentes formas; la misma desaparece cuando se agrega información adicional. En las siguientes secciones, ambigüedad será tratada como un caso especial de Omisión.

Es útil conocer algunas propiedades de los errores, discrepancias, inconsistencia y omisiones. Siendo:

D: conjunto con todas las discrepancias

E: conjunto con todos los errores

I: conjunto con todas las inconsistencias

O: conjunto con todas las omisiones

se tiene que: D Ç O = Æ e I Ç O = Æ , es decir, no pueden existir ni discrepancias ni inconsistencias cuando hay una omisión.

Además: D Ç I = I , D – I = Æ , es decir, cada inconsistencia es una discrepancia.

Además: E Ç I = I , E – I = Æ , es decir, cada inconsistencia es un error.

Entonces: E Ç D Ç I = I , es decir, cada inconsistencia es una discrepancia y un error.

Además: E Ç D - O = I , es decir, cada discrepancia no causada por una omisión y que contiene un error es una inconsistencia. Este artículo propone heurísticas para la detección de problemas pertenecientes al conjunto U = D È E È O.

5. MECANISMO DE INSPECCIÓN DE ESCENARIOS

Es bien conocido en la comunidad de ingeniería de software que, durante la preparación de inspecciones de los diferentes documentos producidos en el proceso de desarrollo de software, los revisores encuentran muy útil conocer los tipos de errores más comunes en la clase de producto que está siendo inspeccionado. Esto permite producir una especie de lista de pasos a realizar que son necesarios para cada tipo de inspección y que se espera sean actualizados en forma continua con su uso. Con este objetivo es que se proponen heurísticas preliminares para la verificación y validación de escenarios escritos en lenguaje natural. Estas heurísticas constituyen lo que Fagan denomina "checklist/entry" y "exit points".

Pese a la gran flexibilidad con que se pueden escribir los escenarios, éstos son enormemente ricos en cuanto a su posibilidad de contrastar unos contra otros y ellos con el LEL, es así que se pueden definir dos tipos de consistencias posibles aplicables a un conjunto de escenarios de un macrosistema:

* Consistencia interna de escenarios
* Consistencia entre escenarios

La Consistencia interna de escenarios se ocupa de contrastar las distintas partes constitutivas de los mismos. La Consistencia entre escenarios se ocupa de analizar la integridad del conjunto de escenarios. En la Figura 2, se esquematizan ambos procesos.

Figura 2 – Proceso de Consistencia de escenarios

Los métodos que se describen a continuación han sido diseñados para su utilización experimental en forma manual a los efectos de permitir una rápida revisión de los mismos y se planifica la transformación de éstos en una herramienta computacional en un futuro cercano. Este uso manual significa que se han diseñado formularios que deben ser completados durante el proceso de inspección.

Los formularios están acompañados por un conjunto de guías y recomendaciones acerca de cómo deben ser llenados y de qué aspectos del o de los escenarios que están siendo inspeccionados deben ser

estudiados.

Tanto en el caso de la consistencia interna de escenarios, como en el de la consistencia entre escenarios se utiliza un formulario resumen que está siempre presente a lo largo de todo el proceso y que contiene, por un lado un área para registrar las correcciones que se han detectado y que corresponden realizar sobre los escenarios y por el otro, un área para registrar toda aquella información a ser recabar para poder determinar qué corrección corresponde hacer en los escenarios. Es decir, a lo largo de ambas consistencias se clasificarán todas los errores, discrepancias y omisiones en dos grandes grupos, aquellos que pueden ser resueltos con los restantes elementos disponibles y aquellos otros que requieren retornar a las fuentes de información para dilucidar la naturaleza del inconveniente detectado.

Finalmente, estas heurísticas han sido planificadas con la precondición de que los responsables de la inspección conozcan en profundidad el LEL utilizado.

5.1 Consistencia interna de escenarios

Se ocupa de analizar cada escenario en forma individual. El proceso de consistencia interna de escenarios utiliza para su desarrollo los siguientes elementos adicionales: el modelo formal de escenarios descripto en la sección 2 y el LEL. Al finalizar el proceso, se obtiene un conjunto de escenarios mejorados y un grupo de dudas que se deben resolver en el macrosistema, a partir de las cuales se genera una nueva versión de escenarios y eventualmente un nuevo proceso de consistencia interna de escenarios.

Como se muestra en la Figura 2, el análisis de cada escenario comprende la verificación sintáctica, la interrelación con el LEL y la interrelación de los componentes. La verificación sintáctica comprueba que cada componente haya sido correctamente escrito, por ejemplo el título puede ser: frase|([sujeto] +verbo+predicado). La interrelación con el LEL verifica que los símbolos del LEL estén correctamente utilizados y que toda frase destacada como símbolo del LEL efectivamente sea parte del mismo. La interrelación de componentes verifica entre otras cosas, que todos los actores de la lista de actores cumplan un rol en algún episodio y que todo sujeto de los episodios esté en la lista de actores.

5.1.1 Verificación sintáctica

La verificación sintáctica se basa en la existencia de formularios y heurísticas para el llenado de los mismos. En este caso particular se utilizan tres formularios, con los siguientes propósitos:

* Verificación sintáctica de título, objetivo, contexto, actores, recursos y excepciones.
* Verificación sintáctica de episodios.
* Verificación sintáctica de indicadores de tipos de episodios.

En el Cuadro 1 se presenta una versión resumida de la heurística asociada con el primero de los formularios y en la Figura 3 se incluye dicho formulario.

VERIFICACIÓN SINTÁCTICA DE COMPONENTES

1º Para cada componente del escenario verificar la sintaxis utilizada.

2º Registrar los elementos faltantes en el casillero correspondiente.

3º Registrar los elementos sobrantes en el casillero correspondiente.

4º Asegurarse que corresponde eliminar todos los sobrantes detectados.

5º Registrar en mejoras a realizar todas las eliminaciones confirmadas.

6º Registrar todos los faltantes como información a recabar.

Cuadro 1: Heurística correspondiente a la verificación sintáctica de componentes.

COMPONENTE

SINTAXIS

FALTANTE

SOBRANTE

Título

frase | ([sujeto] + Verbo + Predicado)

Objetivo

[sujeto] + verbo + Predicado

Ubic. Geográf.

Nombre

Ubic. Temporal

Nombre

Precondiciones

[sujeto] + verbo + Predicado

Restricciones

Sujeto + debe + verbo + predicado

Actores

Nombre

Recursos

Nombre

Figura 3 - Comprobación sintáctica de componentes

5.1.2 Interrelación con el LEL

La verificación de la interrelación de los escenarios con el LEL se realiza mediante la utilización de un

formulario con su correspondiente heurística. En el Cuadro 2 se presenta una versión resumida de la heurística asociada con esta verificación, en la Figura 4 se incluye el formulario utilizado, mostrando un ejemplo aplicado al caso de estudio: Sistema de Plan de Ahorro.

IDENTIFICACIÓN DE SÍMBOLOS DEL LEL

1º Transcribir al formulario todas las palabras o frases marcadas como símbolos del LEL.

2º Verificar la pertenencia al LEL de cada palabra o frase transcripta.

3º Registrar en la columna LEL el tipo de símbolo o N si no existe.

4º Revisar el escenario buscando usos sin marca de pertenencia al LEL.

5º Registrar en la columna SIN NEGRITA el componente donde se detectó.

6º Revisar el escenario buscando símbolos usados con significado diferente.

7º Registrar en la columna MAL USO el componente en que se detectó.

8º Para las líneas con tipo N, asegurarse que no deben ser símbolos del LEL.

9º Para las líneas con indicación SIN NEGRITA, asegurarse que son símbolos del LEL.

10º Para las líneas con indicación MAL USO, asegurarse que se debe usar un sinónimo.

11º Registrar en mejoras a realizar todos los cambios confirmados.

12º Registrar en información a recabar todo aspecto no discernible.

Cuadro 2: Heurística correspondiente a la identificación de símbolos del LEL.

PALABRAS

LEL

SIN NEGRITA

MAL USO

Adherente

S

objetivo, 2

Bien Tipo

O

Sorteo

V

contexto

Figura 4 - Ejemplo del formulario "Identificación de símbolos del LEL"

5.1.3 Interrelación de componentes

La consistencia entre componentes se ocupa de verificar que los actores y los recursos estén correctamente enumerados en los componentes correspondientes y a su vez que los mismos tengan participación en el resto del escenario. Para lograr este efecto se utilizan tres formularios con el siguiente propósito:

* Verificación de ocurrencia de actores en el escenario.
* Verificación de ocurrencia de recursos en el escenario.
* Comprobación pragmática de episodios.

Los dos primeros formularios comprueban si todos los actores y recursos enumerados tienen participación en los restantes componentes del escenario, tales como título, objetivo, contexto y episodios. Además se verifica si toda entidad que merece ser registrada como actor o recurso por su participación en el resto del escenario se encuentra incluida en el componente correspondiente. El tercer formulario tiene por objetivo detectar episodios que no contienen ni actores ni recursos ni símbolos del

LEL, esta situación es de por sí muy sospechosa y probablemente permita detectar alguna omisión en el componente actores o en el componente recursos o en ambos. En el Cuadro 3 se presenta una versión resumida de la heurística asociada con el primero de los formularios y en la Figura 5 se incluye dicho formulario con un ejemplo aplicado al caso de estudio.

VERIFICACIÓN DE OCURRENCIA DE ACTORES EN UN ESCENARIO

1º Registrar todo candidato a actor presente en el título, en el objetivo, en el contexto o en los episodios.

2º Registrar la cantidad de ocurrencias en cada componente mencionado.

3º Registrar si el candidato a actor está presente o no en la lista de actores.

4º Registrar en la columna LEL el tipo de símbolo o N si no existe.

5º Recorrer la lista de actores y transcribir aquellos no incluidos aún.

6º Asegurarse que los candidatos a actores fueron seleccionados apropiadamente.

7º Asegurarse que los actores fueron incluidos en la lista apropiadamente.

8º Asegurarse que los actores con N en la columna LEL, no deben ser símbolos del LEL.

9º Registrar en mejoras a realizar todos los cambios confirmados.

10º Registrar en información a recabar todo aspecto no discernible.

Cuadro 3: Heurística correspondiente a la verificación de actores en un escenario

OCURRENCIA

EXISTENCIA

CANDIDATO

TÍTULO

OBJETIVO

CONTEXTO

EPISODIOS

ACTOR

LEL

Adherente

1

1

0

2, 3

Sí

S

Administradora

0

1

0

1, 2, 3

Sí

S

Reemplazante

0

0

0

Sí

No

Figura 5 - Ejemplo del formulario "Verificación de ocurrencia de actores en un escenario"

5.2 Consistencia entre escenarios

Como se muestra en la Figura 2 este proceso debe realizarse luego de finalizado el proceso de consistencia interna de escenarios. Al finalizar el proceso, se obtiene un conjunto de escenarios mejorados y un grupo de dudas que se deben resolver en el macrosistema, a partir de las cuales se genera una nueva versión de escenarios y eventualmente un nuevo proceso de consistencia interna de escenarios o de consistencia entre escenarios.

La Consistencia entre escenarios se encarga de verificar las relaciones entre los escenarios, de analizar la superposición entre los mismos y de estudiar el grado de cubrimiento del LEL.

5.2.1 Relación entre los escenarios

La verificación de las relaciones entre los escenarios se realiza mediante la utilización de cuatro formularios con el siguiente propósito:

* Comprobación de los escenarios integradores.
* Comprobación de la existencia de subescenarios.
* Verificación de la consistencia de precondiciones en el contexto.
* Verificación de las precondiciones de los subescenarios.

El primer formulario comprueba el nivel de jerarquía de los episodios que componen los escenarios integradores. El segundo tiene por objetivo detectar la inexistencia de subescenarios identificados como tales. El tercero permite detectar errores o discrepancias en las precondiciones de los escenarios u omisión de información para satisfacer dichas precondiciones. El cuarto formulario permite detectar errores en las precondiciones de los subescenarios u omisión de información en los escenarios que contienen dichos subescenarios. En el Cuadro 4 se presenta una versión resumida de la heurística asociada con la cuarta verificación, en la Figura 6 se incluye el formulario utilizado con un ejemplo aplicado al caso de estudio.

VERIFICACIÓN DE LAS PRECONDICIONES DE LOS SUBESCENARIOS

1º Identificar escenarios que contengan subescenarios.

2º Registrar cada precondición del subescenario.

3º Verificar que la precondición lo sea también del escenario padre.

4º Verificar si los episodios del escenario padre previos al episodio subescenario satisfacen la precondición.

5º Identificar como omisiones las precondiciones que no cumplen ni 2º ni 3º.

6º Determinar si es discernible cada omisión a partir del resto de la información.

7º Determinar la posible existencia de errores en la precondición del subescenario o del escenario padre.

8º Registrar en mejoras a realizar todas las omisiones resueltas.

9º Registrar en información a recabar toda omisión pendiente de resolución.

Cuadro 4: Heurística correspondiente a la verificación de precondiciones en subescenarios

ESCENARIO

SUBESCENARIO

PRECONDICION

RELACION

8

9

El adherente debe tener las cuotas mensuales abonadas a la fecha.

episodio 2

11

12

Se tienen tantas solicitudes de ahesión como miembros tiene el grupo de aherentes.

episodios 1, 2

Figura 6 - Ejemplo del formulario "Verificación de precondiciones en subescenarios"

5.2.2 Superposición entre los escenarios

La verificación de las superposiciones entre los escenarios se realiza mediante la utilización de un formulario con su correspondiente heurística. En el Cuadro 5 se presenta una versión resumida de la heurística asociada con esta verificación, en la Figura 7 se incluye el formulario utilizado. Si bien se ha desarrollado un formulario para detectar la superposición entre escenarios, debe quedar claro que éste no es un procedimiento muy apto para ser realizado manualmente por lo que su inclusión sólo se justifica para el futuro desarrollo de una herramienta computacional que automatice todo el procedimiento. La dificultad consiste en que es necesario calcular un índice de cercanía entre escenarios, para facilitar el apareamiento, el que se define como:

Iij = (a * CÇ ij + b * AÇ ij + g * RÇ ij) / (a * CÈ ij + b * AÈ ij + g * RÈ ij)

donde:

CÇ ij = | Cont(Ei) Ç Cont(Ej) |, AÇ ij = | Act(Ei) Ç Act(Ej) |, RÇ ij = | Rec(Ei) Ç Rec(Ej) |,

CÈ ij = | Cont(Ei) È Cont(Ej) |, AÈ ij = | Act(Ei) È Act(Ej) |, RÈ ij = | Rec(Ei) È Rec(Ej) |,

a , b , g , factores de peso,

Cont(Ek) = Conjunto de enunciados del contexto del escenario k,

Act(Ek) = Conjunto de actores del escenario k,

Rec(Ek) = Conjunto de recursos del escenario k,

i, j Î N Ù i ¹ j,

N = Conjunto de números naturales.

Este índice se utiliza para reducir el espacio de búsqueda de los pares de escenarios candidatos a ser revisados buscando superposiciones.

VERIFICACIÓN DE SUPERPOSICIONES ENTRE ESCENARIOS

1º Determinar el conjunto de Escenarios excluyendo los Escenarios Integradores.

2º Confeccionar la lista de combinaciones de Escenarios tomados de a dos sin repeticiones.

3º Para cada par, calcular el índice de cercanía.

4º Seleccionar todos los pares cuyo índice supere un valor prefijado.

5º Para los pares seleccionados analizar los objetivos y los episodios.

6º Seleccionar todos los pares con objetivos o episodios comunes.

7º Confirmar y registrar las superposiciones detectadas.

8º Registrar en mejoras a realizar los reordenamientos de escenarios detectados.

9º Registrar en información a recabar las cuestiones no discernibles.

Cuadro 5: Heurística correspondiente a la verificación de superposiciones entre escenarios

ESCEN.

ESCEN.

ACTOR

RECURSO

CONTEXTO

INDICE

OBJETIVO

EPISODIO

OVERLAP

Figura 7 - Superposición entre escenarios

5.2.3 Cubrimiento del LEL

La verificación del grado de cubrimiento del LEL se realiza mediante la utilización de cuatro formularios, con el siguiente propósito:

* Verificación de la ocurrencia de actores.
* Verificación de la ocurrencia de recursos.
* Identificación de símbolos del LEL no utilizados.
* Contrastación de impactos del LEL con escenarios y episodios.

Los dos primeros formularios permiten detectar actores y recursos no incluidos en el LEL y el nivel de participación de los mismos en los escenarios. El tercero permite específicamente determinar el grado de cubrimiento del LEL por parte del conjunto de escenarios. El cuarto tiene por objetivo verificar que todos los impactos de los símbolos pertencecientes a la clasificación sujeto del LEL estén representados en algún escenario o episodio. En el Cuadro 6 se presenta una versión resumida de la heurística asociada con la cuarta verificación, en la Figura 8 se incluye el formulario utilizado.

CONTRASTACIÓN DE IMPACTOS DEL LEL CON ESCENARIOS Y EPISODIOS

1º Registrar todos los símbolos del LEL pertenecientes a la clasificación Sujeto.

2º Registrar todos los impactos de los símbolos seleccionados.

3º Identificar en los escenarios todas las menciones a los símbolos sleccionados.

4º Revisar cada mención buscando detectar qué impacto está siendo considerado.

5º Registrar los episodios o escenarios que se relacionan con impactos del LEL.

6º Registrar como omisiones los impactos no relacionados con los escenarios.

7º Determinar si es discernible cada omisión, a partir del resto de la información.

6º Registrar en mejoras a realizar todas las omisiones resueltas.

7º Registrar en información a recabar toda omisión pendiente de resolución.

Cuadro 6: Heurística correspondiente a la Contrastación de impactos del LEL con escenarios y episodios.

SUJETO

IMPACTO

ESCENARIOS/EPISODIOS

ACTORES

Figura 8 - Contrastación de impactos del LEL con escenarios y episodios

6. CONCLUSIONES Y TRABAJOS FUTUROS

En el presente artículo se ha enunciado un conjunto de guías para llevar a cabo la inspección de escenarios tanto en sí mismos como en su relación con el Léxico Extendido del Lenguaje. Esta inspección ya ha mostrado en estudios preliminares que logra una importante mejora en la calidad de los escenarios producidos, lo que genera a su vez una mejora en la comprensión del macrosistema y por lo tanto, una mayor y mejor elicitación de requisitos.

Para la difusión del uso de escenarios por parte de los desarrolladores de aplicaciones reales y por consiguiente, para el aprovechamiento en casos prácticos de sus beneficios, se requiere disponer de métodos, técnicas y/o herramientas, tanto para su construcción como para la gerencia de proyectos que incluyan el uso de escenarios.

Las heurísticas propuestas y las herramientas que se desarrollarán a partir de ellas constituyen una contribución efectiva al uso de escenarios y a la administración de los mismos.

Se encuentra en ejecución una experiencia sobre casos reales de aplicación de las inspecciones de escenarios a los efectos de obtener resultados cuantitativos y refinar las heurísticas propuestas. Paralelamente, se están analizando las heurísticas a utilizar para la comparación de escenarios obtenidos a partir de diferentes puntos de vista.

REFERENCIAS

[Barnard 94] "Managing Code Inspection Information", Barnard, J., Price, A., IEEE Software, pp 59-69, March 1994.

[Bell 76] "Software Requirements: are they really a problem?", Bell, T.E., Thayer, T. A., Seccond International Conference on Software Engineering, pp. 61-68, 1976.

[Booch 91] "Object Oriented Design with Applications", Booch, G., The Benjamin Cumming Publishing Company, Inc., Redwood City, 1991.

[Booch 94] "Scenarios", Booch, G., Report on Object Oriented Analysis and Design, Vol. 1, 3, pp 3-9, Sep-Oct 1994

[Bubenko 93] "Objectives driven capture of bussiness rules and information systems requirements", Bubenko, J. A., Wrangler, B., IEEE Conference on Systems, Man and Cybernetics.

[Carrol 95] "Scenario Based Design: Envisioning Work and Technology", Carrol, J. (ed.), System Development, Wiley, New York, 1995.

[Davis 93] "Software Requirements: Objects, Functions and States", Davis, A. M., Prentice Hall, Englewood Cliffs, NJ, 1993.

[Fagan 76] "Design and Code Inspections to reduce Errors in Program Development", Fagan, M.E., IBM Systems Journal, Volume Fifteen, Number three, 1976.

[Hadad 97] "Construcción de Escenarios a partir del Léxico Extendido del Lenguaje", Hadad, G., Kaplan, G., Oliveros, A., Leite, J.C.S.P., 26 JAIIO, Buenos Aires, pp 65-77, Agosto 1997.

[Jackson 95] "Software Requirements & Specifications", Jackson, M., Addison Wesley, ACM Press, 1995.

[Jacobson 92] "Object-Oriented Software Engineering - A Use Case Driven Approach" Jacobson, Y., Christerson, M., Jonsson, P., Overgaard, G., Reading, MA: Addison Wesley, New York: ACM Press, 1992.

[Leite 89] "Application Languages: A Product of Requirements Analysis", Leite, J.C.S.P., Departamento de Informática, PUC-/RJ, 1989.

[Leite 90] "O Uso de Hipertexto na Elicitaçao de Linguagens da Aplicaçao", Leite, J.C.S.P., Franco, A.P.M., Anais de IV Simpósio Brasilero de Engenharia de Software, SBC, pp 134-149, October 1990.

[Leite 95] "A Client Oriented Requirements Baseline", Leite, J.C.S.P., Oliveira, A.P.A., Proceedings of the Second IEEE International Symposium on Requirements Engineering, IEEE Computer Society Press, pp 108-115, 1995.

[Leite 98] "Enhancing a Requirements Baseline with Scenarios", Leite, J.C.S.P., Rossi, G., Balaguer, F., Maiorana, V., Kaplan, G., Hadad, G., Oliveros, A., Requirements Engineeering Journal, Vol 2, No. 4, 1998, pp 184-198.

[Loucopoulos 95] "System Requirements Engineering", Loucopoulos, P., Karakostas, V., McGraw-Hill, London, 1995.

[Macaulay 93] "Requirements capture as a cooperative activity", Macaulay, L., IEEE International Symposium on Requirement Engineering, IEEE Computer Society Press, San Diego, CA, pp 174-181.

[Potts 94] "Inquiry-Based Requirements Analysis", Potts, C., Takahashi, K., Antón, A.I., IEEE Software, Vol. 11, n.2, Mar. 1994.

[Reubenstein 91] "The requirements apprentice: Automated assistance for requirements acquisition", Reubenstein, H. B., Waters, R. C., IEEE Transaction on Software Engineering, 17(3)1991, pp 226-240.

[Weidenhaupt 98] "Scenarios in System Development: Current Practice", Weidenhaupt, K., Pohl, K., Jarke, M., Haumer, P., IEEE Software, pp 34-45, March/April 1998.

[Zave 97] "Four Dark Corners of Requirements Engineering", Zave, P. Jackson M., ACM Transaction on Software Engineering and Methodology, Vol. 6, No 1, January 1997, pp 1-30.

[Zorman 95] "Requirements Envisaging by Utilizing Scenarios (Rebus)", Zorman, L., Ph.D. Dissertation, University of Southern California, 1995.