Explicitar Requisitos del Software usando Escenarios

Graciela Hadad[(1)]
gracielahadad@gmail.com

Jorge Doorn[(1)(2)]
jdoorn@exa.unicen.edu.ar

Gladys Kaplan[(1)]
gladyskaplan@gmail.com

(1) Departamento de Ingeniería e Investigaciones. Tecnológicas, UNLaM, Argentina
(2) INTIA, Facultad de Ciencias Exactas, UNICEN, Argentina

Abstract

Los escenarios describen situaciones del proceso del negocio, tanto del proceso observable actual como del proceso proyectado o futuro. En este último caso los escenarios resultan ser contenedores de la mayoría de los requisitos del sistema de software, pero no son los requisitos propiamente dichos. Muchas de las buenas prácticas o prácticas recomendadas en el proceso de desarrollo de software se basan en la existencia de un documento de requisitos en el que éstos se individualizan en forma precisa. En el presente artículo se presenta una estrategia para confeccionar un documento de requisitos a partir de escenarios ya construidos. La particularidad que presenta la misma es que los requisitos individualizados tienden a ser libres de conflictos y de ambigüedad a consecuencia del propio proceso de producción.

1. Introducción

Las situaciones observadas en el universo de discurso (UdeD) [20] son la base para el conocimiento del problema y las situaciones deseadas son el bosquejo para los requisitos del sistema de software. Estas situaciones pueden representarse a través de escenarios. A los escenarios que plasman las situaciones observadas, los denominamos escenarios actuales. Escenarios futuros (EF) son aquéllos que modelan las situaciones proyectadas como solución a los problemas, demandas y necesidades planteadas en el UdeD y de acuerdo a los objetivos propuestos para el sistema [6]. Estos EF incorporan muchos de los requisitos del software en sus descripciones, especialmente aquellos requisitos de naturaleza funcional.

Los EF son construidos desde el punto de vista del proceso del negocio, incorporando las acciones del actor "sistema de software". Es así, que el mundo descripto por estos escenarios no es observable sino que describen la forma en que se espera que se desenvuelva el negocio cuando se disponga del sistema de software que se planifica desarrollar. No toda la información incluida en los EF está directamente relacionada con el sistema de software, por el contrario muchos de ellos describen situaciones en las que no participa el sistema de software. Estas situaciones se constituyen en una eficaz descripción del contexto en el que el sistema se habrá de desenvolver.

Naturalmente al describirse situaciones en las que el sistema de software realiza acciones ya sea en forma espontánea o como respuesta a estímulos de otro actor, se está hablando en forma implícita de las funcionalidades que deberá tener este sistema de software. En otras palabras, básicamente cada aceptación de un estímulo externo y cada respuesta del sistema está asociada con un requisito que se podría enunciar como "El sistema debe aceptar tal estímulo" o "El sistema debe proveer tal respuesta".

Un aspecto a destacar es que la calidad de los requisitos obtenidos de los EF está garantizada pues ellos han sido previamente verificados, validados y acordados, y además estos requisitos, por su forma de obtención, no presentan conflictos.

Por otro lado, muchas organizaciones solicitantes de un sistema de software estipulan la necesidad de un documento que contenga todos los requisitos del software como parte del contrato con la empresa desarrolladora. Es frecuente que partes de este documento sean utilizadas como parte del pliego de licitación o de un anexo técnico en el caso de sistemas de software de cierta envergadura.

De todo lo anterior surge en forma evidente que muchos requisitos del software se pueden obtener en forma inmediata de los EF meramente redactando de manera diferente las acciones en las que participa el sistema de software. Pero esto no se puede extender más allá de cierto límite ya que existen en los EF otros requisitos que no son tan evidentes ni tan fáciles de explicitar. Es precisamente a esto a lo que está abocado este artículo, es decir a describir las diferentes actividades que se deben realizar para explicitar lo más posible los requisitos presentes en los EF.

Cabe mencionar que en el Proceso Unificado [13] se reemplaza el documento SRS (Software Requirements Specification) por un conjunto de modelos: casos de uso, modelo del dominio o del negocio y lista de requisitos adicionales. Es decir, no pone en evidencia cada requisito

y descarta la creación de un documento.

En general, se puede decir que muchas estrategias para resolver conflictos parten de un documento o lista de requisitos, cuyo origen no es explicado, tales como [26] [34]. Una vez resueltos los conflictos, se dan prioridades, se gestionan cambios en los requisitos y se aparean con los componentes del software a desarrollar. Es decir, existen muchas actividades de gran importancia en el desarrollo del software que están motivadas a partir de un conjunto de requisitos cuya elicitación y definición no están descriptas ni justificadas y donde dicho conjunto pareciera surgir espontáneamente en el proceso de desarrollo.

En la sección siguiente se analiza el tratamiento de los requisitos funcionales y no funcionales, se refieren estrategias para resolver conflictos entre requisitos, y se describen las posibles estructuras del documento de definición de requisitos. En la sección 3 se muestran en forma resumida los modelos de escenario y de especificación de requisito que luego se utilizarán en la sección 4, donde se presenta una estrategia para la extracción de los requisitos de un conjunto de EF. En la sección 5 se presentan estadísticas obtenidas a partir de casos desarrollados, y finalmente se exponen conclusiones y se proponen trabajos futuros.

2. Los requisitos del software

Para arribar a un conjunto de requisitos que sea útil para su posterior uso en el diseño e implementación del software, es necesario comprender su naturaleza, avizorar las principales fuentes de conflictos entre ellos, los atributos que se debe conocer acerca de los mismos y disponer de una forma de representación efectiva. Mucho se ha investigado y publicado acerca de estos aspectos [1] [2] [4] [8] [11] [12] [28] [32] [35] [36] [37] [38]. Las siguientes sub-secciones resumen en forma parcial algunos de estos esfuerzos que están relacionados con las estrategias y propuestas del presente artículo.

2.1. Taxonomía de Requisitos

Existen en la literatura diversas formas de clasificar los requisitos, las que atienden distintos propósitos [2] [17] [29]. Una clasificación ampliamente difundida corresponde a la de requisitos funcionales (RF) y requisitos no funcionales (RNF). Los requisitos no funcionales (RNF) poseen una naturaleza abstracta e intangible en comparación con los requisitos funcionales (RF) y esto hace que sean más difíciles de especificar o documentar formalmente. Los RNF frecuentemente tienden a entrar en conflicto entre sí [22] [3], lo cual debe poder identificarse y resolverse. Pero, por otro lado, el cumplimiento de uno o más RNF puede incidir positivamente en el cumplimiento de otros RNF [22]. Recién en los últimos años se han propuesto métodos para considerar adecuadamente los RNF [1] [3] [15] [16] [23].

En la práctica es muchas veces difícil distinguir si un requisito es funcional o no funcional; muchas veces esto depende del grado de detalle con que se describe al requisito. Si se lo describe en forma más abstracta o general fácilmente podrá considerarse como un RNF pero al describirlo con más detalle podrá parecer un RF [17]. Este es el caso en que un RNF es descompuesto en uno o varios RF más específicos, transformaciones denominadas operacionalizaciones (son consecuencias funcionales que surgen por la necesidad de cumplir con cierto RNF [3]).

2.2 Conflictos entre Requisitos

Un conflicto ocurre cuando dos o más requisitos prescriben cosas opuestas; Robinson [26] denomina al conflicto como interacción negativa entre requisitos, otros autores lo denominan inconsistencia o contradicción [10] [25] [35]. La presencia de un conflicto no es algo inusual: el UdeD es per se contradictorio, los clientes y usuarios suelen tener diferentes intereses, ya sea por su posición en la organización o por objetivos personales.

Una de las actividades del proceso de análisis de requisitos es la detección y resolución de conflictos [28]. Es considerada una actividad costosa en tiempo y esfuerzo [16]. Algunos autores denominan a la resolución de conflictos como negociación de requisitos [16] [37].

Existen diversas estrategias para la identificación y análisis de conflictos, como por ejemplo: uso de matrices de interacción [16] para encontrar las interacciones entre requisitos y sus conflictos; Root Requirements Analysis: técnica que identifica requisitos claves y su interacción propuesto por William Robinson [26]; Systematic Tradeoff Analysis: técnica basada en lógica difusa propuesta por Yen y Tiao [31]; Viewpoint-Oriented System Engineering (VOSE): método que utiliza templates de puntos de vista [25], y se complementa con QC Logic (Quasi-Classical Logic): método formal para identificar, registrar y tomar acciones frente a inconsistencias en especificaciones de múltiples puntos de vista [35]; un enfoque basado en atributos del software y rastreabilidad automatizada [39]; un enfoque basado en múltiples puntos de vista y aplica técnicas de probador de teoremas y verificador de modelos sobre especificaciones en lenguaje natural [34], entre otras.

2.3. Documento de Definición de Requisitos

Los requisitos son generalmente documentados con una narrativa explícita para comunicarlos a los involucrados en el proyecto. También sirve como contrato entre los clientes y el equipo de desarrollo sobre lo que va a proveer el nuevo sistema de software. Además, el SRS puede ser usado para registrar y gestionar cómo los requisitos son satisfechos y qué fragmento del sistema de software fue originado en un cierto requisito (alocación de

requisitos), como un ancla para pre y post rastreo de los requisitos [40], como ancla para la gestión de cambios en los requisitos, como herramienta de validación de los requisitos con los usuarios, como puente de comunicación entre el equipo de analistas y el equipo de diseñadores, como instrumento para probar el sistema de software.

En diferentes organizaciones y en la literatura misma, este documento se genera con diversas estructuras [17] [18] [8], aunque existen estándares internacionales que especifican su estructura, tales como IEEE Std 830-1998 [11] (ver Figura 5 parte izquierda), IEEE Std P1233/D3-1995 [12], DOD 5000.2-R [5], ESA PSS-05-0 [7], NCC 87 [24] entre otros.

Como se observa en la Figura 5 parte izquierda, la estructura que sugiere la IEEE es una guía de lo que debería contener este documento, dando pautas generales de contenido para cada sección. No siempre se crea el SRS, los requisitos pueden quedar empotrados en los modelos construidos, o definirse con alguna plantilla donde se incluyen ciertos atributos [12] [4] [17] [28] [30] [32]. En general los atributos que se asocian a los requisitos se adaptan según el equipo de desarrolladores, cada aplicación en particular y la herramienta de gestión utilizada. Estos atributos deben procurar brindar, primordialmente, ayuda en la comprensión del requisito, adicionalmente facilidad en la búsqueda o selección de requisitos y en la administración del propio proceso de definición de requisitos [10]. En [32] además de presentar un ejemplo de atributos, se aconseja no sobrecargar la especificación de requisitos sino definir sólo aquellos atributos que realmente se van a ingresar y administrar. Frecuentemente se observa que la especificación de un requisito excede sus atributos intrínsecos, pues involucra adicionalmente información asociada a la rastreabilidad y a la gestión de cambios.

3. Los Meta-Modelos de Escenario y de Especificación de Requisito

El proceso de explicitar requisitos que se describe en la sección 4 produce un conjunto de especificaciones de requisitos y un SRS, básicamente a partir de la información contenida en los EF. Se describen a continuación los modelos de estos modelos exceptuando el SRS (ver sub-sección 2.3) considerando la variedad de formatos que puede adoptar.

El meta-modelo de escenario es una estructura compuesta por las siguientes entidades: título, objetivo, contexto, recursos, actores, episodios y excepciones, y el atributo restricción. Actores y recursos son dos componentes enumerativos. El título, el objetivo, el contexto y las excepciones son componentes declarativos, mientras que los episodios son un conjunto de sentencias en un lenguaje simple que dan una descripción operacional de comportamiento. La Figura 1 presenta el meta-modelo de Escenario, adaptado de [20].

Título: identificación del Escenario. En el caso de un sub-Escenario, el título es el mismo que la sentencia episodio, sin las restricciones.
Objetivo: finalidad a ser alcanzada. El Escenario describe la forma de lograr el objetivo.
Contexto: compuesto por al menos uno de los siguientes ítems:
- **Ubicación Geográfica**: lugar físico donde se produce el Escenario.
- **Ubicación Temporal**: especificación de tiempo para el desarrollo del Escenario.
- **Precondición**: estado inicial del Escenario.

Recursos: elementos físicos o información relevantes que deben estar disponibles en el Escenario.
Actores: personas, dispositivos o estructuras organizacionales que tienen un rol en el Escenario.
Episodios: conjunto de acciones que detallan al Escenario y proveen su comportamiento.

Los episodios pueden ser de tres tipos: simples, condicionales u opcionales. Los *episodios simples* son aquellos necesarios para concluir el desenvolvimiento del escenario. Los *episodios condicionales* son aquellos cuya ocurrencia dependen de una condición específica. La condición puede ser interna o externa al escenario. Las condiciones internas pueden deberse a ubicaciones o precondiciones alternativas y a episodios previos. Los *episodios opcionales* son aquellos que pueden o no ocurrir dependiendo de condiciones que no pueden ser explicitadas.

Independientemente del tipo, un episodio puede ser expresado como una acción simple o puede ser concebido en sí mismo como un escenario (denominado *sub-escenario*). Además, se pueden expresar episodios secuenciales y de orden indistinto.

Cada episodio puede llevar un atributo **Restricción**: limitación o condición de calidad respecto a la realización del episodio, habitualmente estas restricciones se asocian a RNF.

Excepciones: generalmente reflejan la falta o mal funcionamiento de un recurso. Una excepción impide el cumplimiento del objetivo del Escenario. Se indica el tratamiento de la excepción a través de un Escenario (denominado *Escenario Excepción*) o de una acción simple.

Figura 1. Meta-modelo de Escenario

El meta-modelo de Especificación de Requisito (ver Figura 2) es simple aunque perfectamente puede reemplazarse por otro también escrito en lenguaje natural. Incluye la descripción del requisito; el tipo de requisito según la clasificación adoptada, que sirve para dar contexto al requisito; el fundamento, que es de vital importancia cuando se solicita un cambio en el requisito; y el estado, que se va actualizando a medida que avanza el proceso de desarrollo del software. Opcionalmente, pueden considerarse en este meta-modelo los atributos: volatilidad, prioridad, criticidad, factibilidad, riesgo y

costo de implementación.

Requisito: descripción del requisito.
Tipo: indica si se trata de un RF o un RNF, o el tipo de RNF al que corresponde, o el tipo según alguna otra clasificación utilizada.
Fundamento: razón de la existencia del requisito.
Estado: condición bajo la cual se encuentra el requisito, puede ser por ejemplo propuesto, aceptado, implementado, validado, etc.
Volatilidad: grado de estabilidad del requisito.
Prioridad: importancia relativa que tiene el requisito para los clientes y usuarios.
Criticidad: necesidad relativa de implementación, puede indicarse si es obligatorio, deseable u opcional, o mediante un ranking de necesidad.
Factibilidad: posibilidad de implementación en el proceso del negocio, ya sea por razones sociales, tecnológicas, ambientales, económicas, etc.
Riesgo: calificación en función de las consecuencias en el proceso del negocio por su implementación.
Costo de implementación: esfuerzo para implementar el requisito en el sistema de software.

Figura 2. Meta-modelo de Especificación de Requisito

La propuesta de este meta-modelo obedece a que considera exclusivamente atributos propios del requisito, independiente del sistema de versionado y de los métodos de trazabilidad utilizados, es por ello que no se han incluidos atributos como autor, fechas de creación y modificación, origen (fuente de información), versión, vinculación con modelos (modelos de requisitos, modelos de diseño, modelos de implementación, etc.), entre otros. Tampoco se han incluido las dependencias con otros requisitos ya que las mismas están claramente explicitadas en el conjunto de EF y serán específicamente consideradas en la asignación de prioridades.

4. El proceso de explicitar los requisitos

Este proceso comienza buscando RF y algunos RNF en los EF. La gran mayoría de los RF se recolectan de los episodios donde el sistema de software es un actor involucrado. Los RNF pueden capturarse de las restricciones y excepciones de los EF.

La lista de requisitos obtenida no requiere ser verificada ni validada dado que se construye a partir de un conjunto de EF que ha pasado por procesos de verificación y validación [21]. Además se supone que en esta etapa los requisitos no necesitan ser acordados porque ello ya fue realizado a través de los EF. Si no se han dado prioridades a los EF, entonces será necesario realizar reuniones con los clientes y usuarios para dar prioridades a los requisitos.

Se elicitan y acuerdan los atributos de cada requisito para completar su especificación. Finalmente, se redacta el Documento de Definición de Requisitos (SRS), basándose en algún estándar que la organización utilice o que se proponga de acuerdo al proyecto en particular. Este documento debe respetar el vocabulario utilizado en el UdeD, manteniendo vínculos a un glosario denominado Léxico Extendido del Lenguaje (LEL, modelo que representa el lenguaje del dominio de la aplicación [19]). Luego el SRS es un documento de fácil lectura para los usuarios, y que además presenta nula o escasa ambigüedad.

Debe destacarse que se está partiendo de un conjunto de EF, el cual no sólo fue verificado y validado, sino que los RNF elicitados fueron muchos de ellos operacionalizados y, por lo tanto, incluidos como acciones en los EF, o permanecieron como tales, por implicar decisiones de diseño o propiedades intrínsecas del software, y fueron adosados a escenarios o episodios de escenarios.

Las actividades involucradas en el proceso, que se presenta mediante un modelo SADT (Structured Analysis and Design Technique) [41] en la Figura 3, son: 1) Generar la lista de requisitos, 2) Dar atributos a los requisitos, 3) Organizar los requisitos en un SRS y 4) Verificar el SRS.

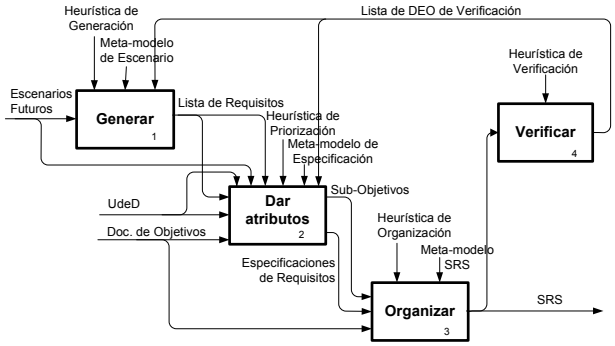

Figura 3. SADT del proceso de explicitar Requisitos de Software

(1) GENERAR

Esta actividad tiene por objetivo generar una lista de requisitos únicos del sistema de software, lo más completa posible. Incluye extraer los requisitos de cada EF y eliminar requisitos repetidos de la lista. Los requisitos se obtienen de varios componentes del escenario, principalmente de los episodios donde el sistema de software actúa, éstos últimos serán del tipo RF. No se considerarán para este propósito los episodios que sean sub-escenarios. Además, pueden extraerse RF de las excepciones, donde la solución es una sentencia donde actúa el sistema de software, no siendo la solución un escenario excepción. En algunos casos las condiciones de los episodios condicionales pueden involucrar datos, estados de datos o acciones, y por lo tanto, pueden obtenerse RF. Esto mismo ocurre en la causa de excepciones.

Cabe aclarar respecto a las condiciones, que los datos o estados de datos que participan en la misma serán aquellos que el sistema deberá administrar, y en el caso de una condición que involucra una acción debe participar en ella el sistema. Se ha comprobado a través del estudio de varios casos en los que una condición que involucra una acción implica que se ha abreviado el discurso. Es decir:

Si *acción1* ***entonces*** *acción2*

⇒ *acción1*
Si *verdadera* ***entonces*** *acción2*

Ejemplo A (Ver caso EQUIP en Tabla 3):

Si *el sistema verifica que el equipo no tiene número de pedido* ***entonces*** *ASIGNAR NÚMERO DE PO*

⇒
El sistema verifica que el equipo tiene número de pedido.
Si *no tiene* ***entonces*** *ASIGNAR NÚMERO DE PO.*

En el caso que la causa de una excepción involucre un RF, con frecuencia, significa que el sistema debe realizar algún tipo de verificación. Por ejemplo:

Ejemplo B:

Causa de excepción: *El sistema no reconoce el número de formulario.*
RF: *El sistema debe verificar la existencia del número de formulario.*

Ejemplo C:

Causa de excepción: *El pasaporte emitido no es retirado antes de los 60 días.*
RF: *El sistema debe verificar que la fecha de emisión del pasaporte no supere 60 días de la fecha actual cuando el solicitante pase a retirar el pasaporte.*

(Ejemplos B y C extraídos del caso "Sistema Nacional para la Obtención de Pasaporte" desarrollado en UB, 1996)

Se pueden extraer RNF básicamente de las restricciones de los episodios donde participa el sistema de software, adicionalmente pueden encontrarse en la causa de excepciones. También a partir de los recursos involucrados, pueden inferirse algunos requisitos de calidad. Si el recurso es un símbolo tipo Objeto del LEL, puede buscarse en la noción del mismo alguna característica particular que pueda indicar la necesidad de un RNF. Por ejemplo:

Ejemplo D (Ver caso AGEN en Tabla3):

Recurso: *Medio de comunicación.*

RNF: *El sistema debe poder conectarse por algún medio de comunicación (e-mail) con todos los convocados a una reunión.*

A continuación se presentan dos ejemplos de causa de excepción que implica un RNF:

Ejemplo E (Ver caso CLASIF en Tabla 3):

Causa de excepción: *El sistema no se puede conectar al servidor donde se encuentran las estadísticas.*
RNF: *El sistema debe establecer conexión con el servidor de estadísticas.*

Ejemplo F (ver caso PREST en Tabla 3):

Causa de excepción: *Problemas en el servicio de internet.*
RNF: *El sistema debe poder accederse desde internet.*

Esto no significa que cada uno de estos componentes del escenario sea siempre un requisito, es justamente en esta actividad en la que se determina cuáles de las partes presentes en los EF son efectivamente requisitos.

La Figura 4 muestra un ejemplo de lista de requisitos extraída de un solo escenario futuro, donde se aprecian cuatro RF y un RNF (extraído del caso "Sistema Nacional para la Obtención de Pasaporte"). En la Tabla 1 se resumen los posibles tipos de requisitos a extraer de los EF, dando ejemplos de cada uno de ellos. Se puede observar en los ejemplos de esta Tabla que existen palabras o frases subrayadas, ellas representan vínculos a símbolos del LEL. Se debe tener presente que dependiendo del componente o ítem puede extraerse directamente un requisito, puede evaluarse si es un requisito o puede inferirse un requisito. En la Tabla 2 se identifica el modo de extracción de requisitos desde los componentes o ítems de un escenario.

El objetivo del EF no se ha utilizado como posible contenedor de requisitos. Estos objetivos pueden representar las grandes funcionalidades del sistema, siempre y cuando el actor sistema de software tenga una participación importante en los episodios del escenario. Por lo tanto, ellos pueden abarcar varios requisitos del sistema de software, pudiendo utilizarse esta relación para establecer dependencias entre ellos.

Una vez extraídos los requisitos de cada EF, debe controlarse la existencia de redundancia de los mismos. Es muy frecuente que un requisito esté presente en más de un escenario. Por ejemplo el requisito "El sistema debe permitir la selección de un cliente por apellido, por tipo o por localidad" puede presentarse en más de un escenario donde debe realizarse alguna tarea vinculada a un cliente. Luego a partir, de esta sub-actividad, se obtiene una lista de requisitos únicos.

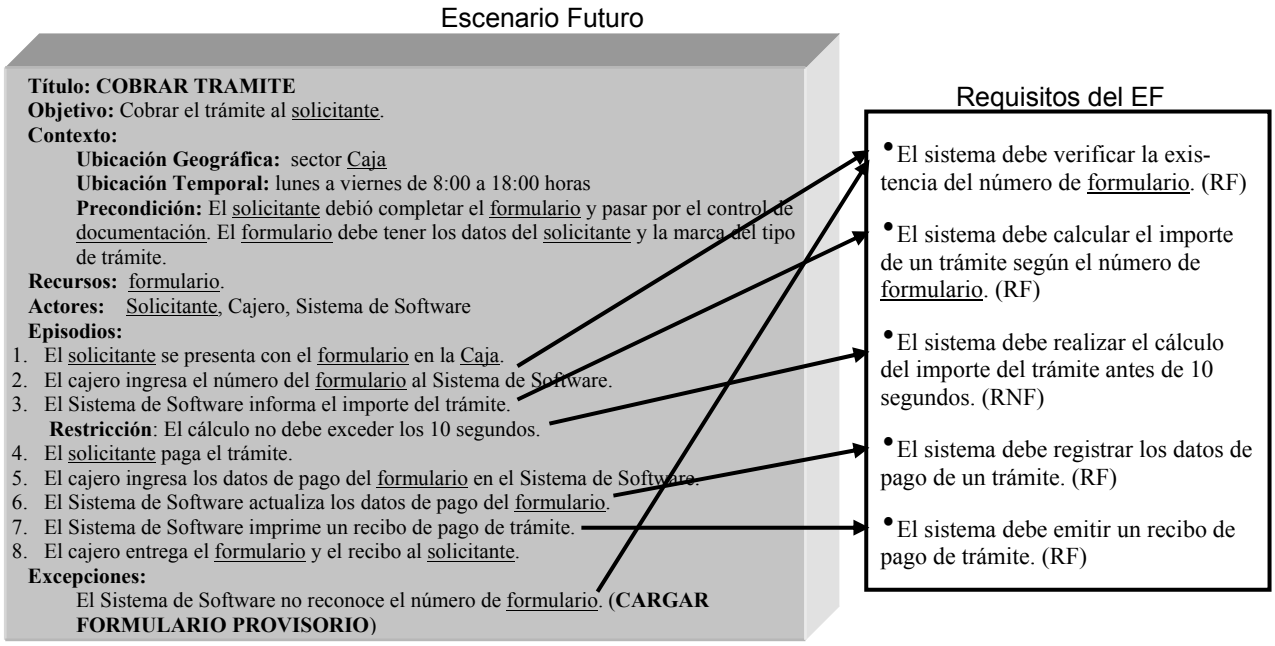

Figura 4. Ejemplo de lista de requisitos de un escenario futuro

Tabla 1. Tipos de requisitos extraídos de escenarios futuros

Tipo	Componente	Ejemplo
Requisito Funcional	*Episodios* donde participa el Actor Sistema de Software	El Sistema de Software imprime un recibo de pago del trámite.
	Solución de Excepción donde participa el Actor Sistema de Software	El Sistema de Software envía mensaje sobre el límite de fecha para la incineración de un pasaporte.
	Condiciones donde en el episodio participa el Actor Sistema de Software	**Si** <<el tipo de trámite es pasaporte original>> **entonces** el Sistema de Software … **Si** <<es el segundo chequeo del equipo>> **entonces** el Sistema de Software … **Si** <<el Sistema de Software verifica que el equipo no tiene asignado número de pedido>> **entonces** el Sistema …
	Causa de Excepciones	El Sistema de Software no reconoce el número de formulario. Algún dato del convocado es inválido o nulo. El pasaporte emitido no es retirado antes de los 60 días. Administración modifica en el Sistema de Software algún dato cargado por Comercio Exterior.
Requisito No Funcional	*Restricción* donde en el episodio participa el Actor Sistema de Software	El Sistema de Software debe realizar el cálculo del importe del trámite antes de 10 segundos.
	Causa de Excepciones	Existen problemas en el servicio de internet. El Sistema de Software no se puede conectar al servidor donde se encuentran las estadísticas.
	Recursos	e-mail archivo de presentación

Tabla 2. Tipo de relación entre componente y requisito

Componente	Relación con requisito
Episodios	ES
Solución de Excepciones	ES
Condiciones	PUEDE INFERIRSE o PUEDE SER (caso acción)
Causa de Excepciones	PUEDE INFERIRSE
Restricción	PUEDE SER
Recursos	PUEDE INFERIRSE

(2) DAR ATRIBUTOS A LOS REQUISITOS

En esta actividad se genera, a partir de la lista de requisitos, las especificaciones de los requisitos necesitándose entonces obtener información del universo de discurso. Estas especificaciones presentan vínculos al LEL, pues por un lado parte de la información proviene de los EF descriptos usando el LEL y la descripción de la nueva información debe respetar también el uso de los símbolos del LEL.

Los clientes y usuarios dan prioridades a los requisitos con apoyo de los ingenieros de requisitos. También se suele establecer la criticidad de cada requisito. Se realizan reuniones para determinar la importancia relativa que tiene un requisito para los clientes y usuarios, y para organizar aquellos requisitos que deben implementarse inicialmente frente a aquellos que pueden postergarse. Se deben tener en cuenta, además de la dependencia de los requisitos, la diversidad de intereses de los clientes y usuarios, las limitaciones de recursos, las necesidades del negocio y las imposiciones del mercado, entre otros factores. Existen variadas técnicas de priorización de requisitos, como por ejemplo: AHP (Analytic Hierarchy Process) [27] [14], QDF (Quality Function Deployment) [33] y EVOLVE [9]. En [42] se presenta una heurística de asignación de prioridades que se basa en una división de objetivos del sistema, donde se priorizan los sub-objetivos de menor nivel y por transitividad se asocia cada requisito a la satisfacción del sub-objetivo.

La tarea de dar prioridades puede realizarse previamente al describirse los EF, dando prioridad a los mismos, pues puede ser una tarea más fácil que priorizar cada requisito individualmente, ya que muchos de ellos están interrelacionados y deben considerarse sus prioridades en simultáneo.

Se debe tener en cuenta que no sólo se le da prioridad a los requisitos sino también puede establecerse su criticidad, factibilidad y riesgo, que son algunos de los atributos que presentan los requisitos. El fundamento surge a partir del origen del requisito, es decir del o de los EF en donde se resuelve. En cuanto al estado, en principio es "propuesto", pero una vez que se le asigna su prioridad pasa a estar "aprobado". Actualizaciones al estado irán surgiendo durante el resto del proceso de desarrollo del software. Posteriormente, estas especificaciones de requisitos son validadas mediante reuniones con los usuarios.

(3) ORGANIZAR

Los requisitos se describen en un documento según el formato de SRS impuesto por el cliente o adoptado por el equipo de desarrollo (ver sección 2.3), a partir de las especificaciones generadas en el paso anterior. El propósito principal es organizar los requisitos de manera tal de obtener el máximo de legibilidad.

Deben mantenerse en el SRS vínculos al LEL, como una forma de reducir la ambigüedad del documento escrito en lenguaje natural. Adicionalmente se mantienen vínculos a los EF para mantener la pre-trazabilidad de los requisitos. Debe además considerarse que muchos requisitos están relacionados con otros, en muchos casos se trata de RNF con RF. El mantenimiento de estos vínculos permite la inter-trazabilidad.

Recomendaciones para describir cada requisito (algunas extraídas de [10]):

- ✓ Redactar utilizando la forma *"El sistema debe …"*, con lo cual se identifica claramente la capacidad o la condición que el sistema de software brindará o acatará.
- ✓ No incluir más de un requisito en cada sentencia.
- ✓ Evitar el uso de la frase "de ser necesario", pues debe quedar claro bajo qué condiciones opera el requisito.
- ✓ Evitar palabras ambiguas como "generalmente", "frecuentemente", "rápido", "flexible", "amigable", "preferible".
- ✓ Evitar deseos ilimitados en las capacidades del sistema de software, por ejemplo "bajo cualquier plataforma", "independiente del motor de base de datos", "100% confiable", "sin fallas".
- ✓ Para RNF, incluir valores de medición. Se puede describir el valor deseado y el valor máximo o mínimo aceptable.

Si se considera el formato presentado por [11], entonces gran parte de sus secciones pueden completarse con la información obtenida, como se describe a continuación (ver Figura 5). La sección 1.2 del SRS "Alcance del Proyecto" incluye el nombre del proyecto que puede obtenerse del título del escenario futuro integrador de nivel 0, y el objetivo del producto que se obtiene del documento de objetivos del sistema, incluyendo sub-objetivos obtenidos al priorizar, o de los objetivos de los escenarios integradores. En la sección 1.3 del SRS "Definiciones, acrónimos y abreviaturas" puede hacerse referencia al LEL. En la sección 1.4 "Referencias" se indicarán referencias al conjunto de EF

y al LEL. La sección 2.2 “Funciones del producto” se puede completar con los objetivos de los EF con el sistema como actor importante. En la sección 2.3 “Características de los usuarios” se puede completar con los actores de los EF que interactúan con el software y vincularlos con los respectivos símbolos tipo Sujeto del LEL. La sección 3 “Requisitos Específicos” incluye todos los requisitos extraídos de los EF, donde podrán organizarse por su tipo, por el EF origen, por prioridad, por Actor del EF.

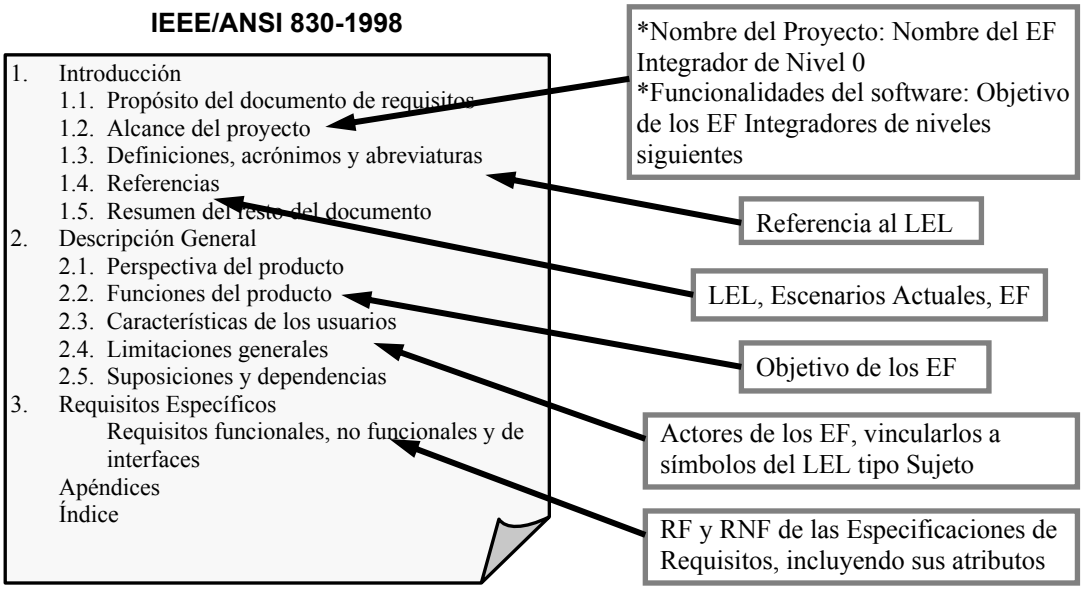

Figura 5. Completar un SRS con la información obtenida

(4) VERIFICAR

La actividad de verificación debe comprobar la correcta generación del SRS, determinando si la extracción de requisitos desde los EF fue realizada exhaustivamente, si la priorización es consistente y si el SRS contempla todos los requisitos explicitados y priorizados. La técnica de verificación sugerida por su alta efectividad [21] es la de inspección. Se genera a partir de esta actividad una lista de DEO (Discrepancias, Errores y Omisiones) para corregir el SRS. Con la lista generada se vuelve a las actividades Generar la lista y Dar Atributos, según el tipo de defectos encontrados. Los aspectos principales que cubre la verificación son:

- ⇨ Verificar la sintaxis:
  - ✓ Detectar el uso de más de un verbo en la descripción del requisito.
  - ✓ Detectar el uso de adjetivos subjetivos.
  - ✓ Detectar el uso de frases ambiguas o subjetivas (cuyo significado dependa del interlocutor).
- ⇨ Verificar contra los componentes del EF:
  - ✓ Controlar que todo requisito proviene de un componente del EF: objetivo, episodio, restricción, causa de excepción, solución de excepción, recurso o condición de episodio.
- ⇨ Verificar dependencias entre requisitos:
  - ✓ Detectar relaciones no establecidas según sus orígenes en EF.
- ⇨ Verificar atributos:
  - ✓ Controlar que prioridad, criticidad, factibilidad, riesgo y costo asignados a un requisito estén dentro de los rangos establecidos.
  - ✓ Controlar que prioridad, criticidad, factibilidad, riesgo y costo del requisito sean consistentes según su dependencia con otros requisitos.
  - ✓ Controlar que se haya fundamentado y clasificado el requisito.
- ⇨ Verificar referencias al LEL:
  - ✓ Controlar que todo símbolo del LEL es identificado al mencionarse en un requisito.
  - ✓ Controlar el uso correcto de símbolos del LEL.

5. Datos observados

Para obtener información del proceso de explicitar requisitos desde escenarios, se evaluaron 14 casos de estudio. La Tabla 3 presenta una breve descripción de cada caso de estudio, los que habían sido desarrollados con anterioridad al comienzo de este estudio.

Estos casos fueron realizados algunos por profesores y otros por alumnos de grado avanzados pertenecientes a distintas universidades, estudiando organizaciones argentinas. El caso EQUIP correspondió al desarrollo completo de un software real, mientras que los otros casos fueron realizados sólo abarcando la fase de requisitos. Los casos de estudio seleccionados fueron previamente evaluados por un ingeniero de requisitos

sobre la calidad de los mismos y sobre la variedad de información contenida en los componentes de los escenarios (ver Tabla 4).

Tabla 3. Descripción de los casos de estudio

Caso	Nombre	Univer-sidad (**)	# de Perso-nas	Año
AGEN	Agenda de Reuniones (*)	UB y UNLaM	2	2007
EQUIP	Administración y Adquisición de Equipos (cajeros automáticos)	Empresa	1	2003
CAJA-1	Seguimiento de Producción de Cajas de Cartón	UNICEN	1	2006
PREST	Administración de Préstamos	UNLaM	3	2006
ENCUA	Administración del Almacén de MP y artículos de Encuadernación	UNLaM	3	2005
SERV	Seguimiento de Service de equipos para radiocomunicación	UNLaM	2	2005
CAJA-2	Seguimiento de Producción de Cajas de Cartón	UNLaM	2	2005
CLASIF	Sistema de Clasificados de un diario	UTN	2	2005
QUIM	Gestión de Producción de Químicos	UTN	2	2005
SOPOR	Administración de Soportes con contenidos de programación por cable	UTN	2	2005
ECPDG	Servicio de Ecodoppler y Ecocardiografía	UTN	3	2005
LAB FAR	Planificación y Seguimiento en un Laboratorio Farmacéutico	UTN	2	2004
CONVE	Gestión de convenios universitarios	UNLaM	2	2007
PRESU	Servicio de Presupuesto y Seguimiento de Pedidos	UNLaM	2	2007

(*) "Sistema de Agenda de Reuniones" desarrollado en UB 1997 y actualizado en UNLaM 2007.
(**) UB: Universidad de Belgrano, UNLaM: Universidad Nacional de La Matanza, UNICEN: Universidad Nacional del Centro de la Provincia de Buenos Aires, UTN: Universidad Tecnológica Nacional.

La idea motora de analizar estos 14 casos de estudio fue determinar en qué componentes de los EF se encontraban realmente los requisitos, es por ello que se revisó cada componente de cada caso y se contabilizaron RF y RNF encontrados en cada uno. Se consideró también la aparición del mismo requisito en distintos componentes, y la ampliación en la contabilización de los componentes según características internas. Por ejemplo, no sólo se estudió el componente Episodios, sino que se evaluó si cada episodio era del tipo simple o condicional. Es por ello, que los ítems que se contabilizaron en cada caso de estudio fueron:

- ✓ Escenarios
- ✓ Objetivo
- ✓ Precondición (cada sentencia individualmente)
- ✓ Recursos
- ✓ Episodios simples, condiciones y opcionales
- ✓ Condiciones de episodios
- ✓ Excepciones
- ✓ Causas de excepciones
- ✓ Tratamiento (Solución) simple de excepciones
- ✓ Restricciones de Episodios
- ✓ Restricciones de Recursos

Se descartaron los componentes: Título, Ubicación Geográfica, Ubicación Temporal y Actores. El primero no fue considerado debido a que el Título es parte del Objetivo, siendo este más descriptivo. Las Ubicaciones Geográfica y Temporal son declaraciones de contexto sin posibilidad de contener requisitos de software. El componente Actores es enumerativo, y aunque Recursos también lo es, este puede estar condicionado por su uso mediante el sistema de software, y puede a partir de su relación con un símbolo del LEL del tipo Objeto inferirse alguna limitación relativa al software.

Tabla 4. Tamaño de los casos de estudio

Caso	# Escenarios	# Episodios	# Restricciones	# Excepciones	# Precondición	Total Requisitos
AGEN	24	112	14	21	45	**119**
EQUIP	38	195	0	11	16	**167**
CAJA-1	8	43	1	0	16	**286**
PREST	20	103	0	5	16	**50**
ENCUA	17	86	0	5	18	**67**
SERV	12	55	0	8	16	**117**
CAJA-2	10	58	0	2	4	**52**
CLASIF	23	94	0	21	25	**27**
QUIM	29	171	0	16	45	**79**
SOPOR	20	81	0	7	9	**24**
ECPDG	22	103	0	2	9	**54**
LABFAR	20	142	0	8	17	**78**
CONVE	22	91	4	11	21	**45**
PRESU	38	171	8	21	52	**84**

Se observa en la Tabla 4 que para cada caso de estudio se desarrolló en promedio 22 escenarios con un

promedio 107 episodios. Sólo cuatro casos describieron restricciones en sus episodios y un promedio de 10 excepciones por caso. Se contabilizó individualmente cada sentencia en el sub-componente Precondición como una precondición.

En la Tabla 5 se observa que se extraen en promedio 50 RF provenientes de episodios por cada caso de estudio. Para estudiar las condiciones que afectaban a los episodios condicionales, se comprobó que en los casos observados se extrajeron RF del 33% de las condiciones y ningún RNF, el resto correspondía a condiciones externas al sistema de software.

Tabla 5. Total de requisitos extraídos de episodios

Caso	# Excepciones	# Excep. Simples	# Excep. Escen.	RF en Soluc. Simple	RF en Causa	RNF en Causa
AGEND	21	15	6	12	11	0
EQUIP	11	4	7	2	4	0
CAJA-1	0	--	--	--	--	--
PREST	5	5	0	2	1	2
ENCUA	5	2	3	0	1	0
SERV	8	8	0	2	3	0
CAJA-2	2	1	1	1	2	0
CLASIF	21	15	6	0	0	4
QUIM	16	14	2	0	3	0
SOPOR	7	6	1	3	3	0
ECPDG	2	2	0	1	1	0
LABFAR	8	8	0	0	1	0
CONVE	11	11	0	4	6	0
PRESU	21	21	0	0	8	0

En la Tabla 6 se observan datos referidos a las excepciones. Se extrajeron 27 RF del tratamiento simple de las excepciones (cuando no se especifica un escenario excepción). Del total de causas que provocan una excepción (138), se obtuvieron 44 RF y 6 RNF; el resto corresponden a causas externas al sistema.

En la Tabla 7 se analizan las restricciones y precondiciones para la obtención de requisitos. Sólo 4 casos de estudio han considerado restricciones. Del total de precondiciones, algunas representaban RF y el resto eran precondiciones externas al sistema. Este sub-componente del escenario no fue tenido en cuenta en la estrategia de extracción de requisitos desde los EF por ser redundante. Es decir, estos RF provenían a su vez de episodios de otros escenarios o de causas de excepciones. Las precondiciones pueden ser internas o externas al macrosistema. Por lo tanto, las segundas no son manejables por el sistema. Las internas son satisfechas dentro del macrosistema por algún escenario o episodio o causa de excepción, luego si involucran algún requisito, éste ya queda incluido al evaluar los episodios y las causas de excepción.

Tabla 6. Total de requisitos por excepciones

Caso	# Episodios	# RF en Episod	# Episod. Condicionales	# RF en Episod Condic	# RNF en Episod Condic
AGEND	112	73	23	10	0
EQUIP	195	147	34	14	0
CAJA-1	43	32	7	0	0
PREST	103	40	12	5	0
ENCUA	86	58	9	8	0
SERV	55	26	10	8	0
CAJA-2	58	44	6	5	0
CLASIF	94	17	20	6	0
QUIM	171	72	27	4	0
SOPOR	81	17	2	1	0
ECPDG	103	50	3	2	0
LABFAR	142	24	21	0	0
CONVE	91	27	13	5	0
PRESU	171	71	26	2	0

Tabla 7. Total de requisitos por restricciones y precondiciones

Caso	# Restricciones	# RF en Restr.	# RNF en Restr.	# Precondiciones	# RF en Precon	# RNF en Precon
AGEND	15	9	4	45	35	0
EQUIP	0	--	--	16	8	0
CAJA-1	1	0	0	16	0	0
PREST	0	--	--	16	12	0
ENCUA	0	--	--	18	13	0
SERV	0	--	--	16	5	0
CAJA-2	0	--	--	4	2	0
CLASIF	0	--	--	25	4	0
QUIM	0	--	--	45	8	0
SOPOR	0	--	--	9	5	0
ECPDG	0	--	--	9	1	0
LABFAR	0	--	--	17	2	0
CONVE	4	3	0	21	11	0
PRESU	8	3	0	52	23	0

Las restricciones que sugieren RF se deben en su mayoría a controles que el sistema debe realizar o modos restringidos en que debe operar. Es decir, estas restricciones son acciones que el sistema debe hacer y deberían expresarse en un episodio (caso controles) o como información contenida en el episodio (caso de modos restringidos). La escasa obtención de RNF a partir de las restricciones se debe a la poca atención en la elicitación de los mismos, dado que los escenarios promueven principalmente la elicitación de comportamientos.

En la Tabla 8 se ha contabilizado el total de requisitos extraídos de los EF de cada caso de estudio, considerando: episodios, condiciones de episodios, restricciones, soluciones simples de excepciones y causas de excepciones, que representasen algún tipo de requisito. Se observa la diferencia de requisitos obtenidos en 9 casos de estudio del total de 14, donde con anterioridad a la evaluación ya se había generado una lista de requisitos a partir de los EF sin disponer de heurísticas ni guías. Los requisitos obtenidos siguiendo la estrategia detallada en este artículo se obtuvieron sin el conocimiento de la lista ad-hoc ya existente. En los casos estudiados en los que se había obtenido previamente una lista de requisitos, el proceso descripto en este artículo superó la lista preexistente en un 44%.

Tabla 8. Total de requisitos por caso de estudio

Caso de estudio	Total Requisitos Ad-hoc	Total Requisitos Derivados	Diferencia
AGEN	**81**	**119**	**+38**
EQUIP	--	167	**--**
CAJA-1	--	286	**--**
PREST	**21**	**50**	**+29**
ENCUA	**42**	**67**	**+25**
SERV	**32**	**117**	**+85**
CAJA-2	--	52	**--**
CLASIF	**22**	**27**	**+5**
QUIM	**45**	**79**	**+34**
SOPOR	**19**	**24**	**+5**
ECPDG	**49**	**54**	**+5**
LABFAR	--	78	**--**
CONVE	--	45	**--**
PRESU	**37**	**84**	**+47**

Luego del estudio cuantitativo de los casos de estudio, se analizó semánticamente los resultados alcanzados y se determinó:

- ✓ El componente Objetivo no fue considerado en la estrategia. Los objetivos de los EF describen funcionalidades del sistema de software si en ellos interviene el sistema como Actor, pero no corresponden a requisitos por su amplitud de alcance y falta de especificidad.
- ✓ La Precondición tampoco fue incluida debido a que todos los requisitos incluidos en ella ya estaban incluidos en otros componentes de otros EF.
- ✓ El análisis semántico de los Recursos estableció que 3 casos (AGEN, PRESU y PREST) dieron origen a RNF y, por lo tanto, esta característica debe ser tenida en cuenta en la estrategia.
- ✓ Respecto a los RF hallados en las restricciones de episodios, se decidió mejorar la heurística que interviene en la construcción de este componente durante la creación de los EF.

6. Conclusiones

Existen en la literatura muchas propuestas sobre los escenarios y casos de uso y, por otro lado, mucho se dice sobre los requisitos de software, pero poco se trata sobre extraer requisitos de los escenarios y cómo redactar un SRS no ambiguo.

Otro punto importante es que los requisitos extraídos en los 14 casos de estudio no presentan conflictos entre sí. Se entiende que esto se debe a que ya en el proceso de construcción de los EF, los conflictos se identifican con técnicas de verificación y validación y se solucionan a través de la negociación. Es decir, el tema de requisitos contradictorios está ampliamente difundido en la literatura [25] [31] [26] [34], pero en general poco se menciona sobre la forma en que fueron elicitados y definidos los requisitos. Es probable que los conflictos entre requisitos surjan cuando éstos han sido levantados en una modalidad ad-hoc sin ningún procedimiento que los respalde y, por lo tanto, no han soportado la consistencia implícita que se produce al aplicar las heurísticas de construcción de los EF.

En principio se propone generar una lista de requisitos clasificándolos sólo en RF y RNF, pero los cuales están validados y acordados por los clientes y usuarios. A partir de ella, puede redactarse un documento de definición de requisitos organizando los requisitos de acuerdo al estándar utilizado, ya sea éste elegido por la organización cliente o por los desarrolladores. La ambigüedad del SRS se reduce pues estará redactado utilizando los términos empleados en el UdeD y con trazas a los EF que les dieron origen.

En próximos pasos de la investigación se abordarán dos aspectos:

i) Se estudiarán más detalladamente las características necesarias para describir en forma precisa y no ambigua a los RNF, donde tal vez sea necesario extender el meta-modelo de Especificación de Requisitos propuesto.
ii) Se estudiará el impacto que ha introducido el proceso de desarrollo llevado a cabo hasta ese punto sobre la semántica de los términos del LEL, y en consecuencia la posible disminución de la legibilidad del SRS.

Referencias

[1] B.W. Boehm, y H. In, "Identifying Quality-Requirement Conflicts", *IEEE Software*, Vol.13, Nº2, 1996, pp. 25-35.

[2] L. Chung, B.A. Nixon, y E. Yu, *Non-Functional Requirements in Software Engineering*, Kluwer Academic Publishers, 2000.

[3] L.M. Cysneiros, y E. Yu, "Non-Functional Requirements Elicitation", *Perspectives on Software Requirements*, Kluwer Academic Publishers, cap. 6, 2004, pp.115-138.

[4] A. Davis, y D. Leffingwell, "Making Requirements

Management Work For You", *Crosstalk, The Journal of Defense Software Engineering*, Vol.12, Nº4, 1999.
[5] US Department of Defense, *DoD Regulation Guidance 5000.2-R: Mandatory Procedures for Major Defense Acquisition Programs (MDAPs) and Major Automated Information System (MAIS) Acquisition Programs*, 2002.
[6] J. Doorn, G. Hadad, y G. Kaplan, "Comprendiendo el Universo de Discurso Futuro", *WER'02*, Valencia, España, 2002, pp.117-131.
[7] ESA PSS-05-0, *European Space Agency Board for Software Standardisation and Control. Software Engineering Standards*, DFl. 50, Issue 2, Febrero 1991.
[8] R.E. Fairley, y R.H. Thayer, "The Concept of Operations: The Bridge from Operational Requirements to technical Specifications", *Software Requirements Engineering*, R. Thayer & M. Dorfman eds., IEEE Computer Society Press, 2º edición, Los Alamitos, CA, 1997, pp. 73-83.
[9] D. Greer, "Requirements Prioritisation for Incremental and Iterative Development", *Requirements Engineering for Sociotechnical Systems, Information Science Publishing*, Maté & Silva eds, cap.VII, 2005, pp. 100-118.
[10] E. Hull, K. Jackson, y J. Dick, *Requirements Engineering. Springer Science*, 2º edición, 2005.
[11] IEEE Std 830-1998, *IEEE Recommended Practice for Software Requirements Specifications (ANSI)*, NY, 1998.
[12] IEEE Std P1233/D3-1995, *IEEE Guide for Developing System Requirements for Specifications*, IEEE, NY, 1995.
[13] I. Jacobson, G. Booch, y J. Rumbaugh, *The Unified Software Development Process*, Addison-Wesley,1º ed,1999.
[14] J. Karlsson, C. Wohlin, y B. Regnell, "An evaluation of methods for prioritizing software requirements", *Information and Software Technology*, Vol.39, Nº14-15, 1998, pp. 939-947.
[15] D. Kerkow, J. Dörr, B. Paech, T. Olsson, y T. Koenig, "Elicitation and Documentation of Non-Functional Requirements for Sociotechnical Systems", *Requirements Engineering for Sociotechnical Systems*, Information Science Publishing, Maté & Silva (eds), cap.XVII, 2005, pp.284-302.
[16] G. Kotonya, y I. Sommerville, "A Framework for Integrating Functional and Non-Functional Requirements", *IEEE International Workshop on Systems Engineering for Real Time Applications*, UK, 1993, pp.148-153.
[17] G.Kotonya, y I. Sommerville, *Requirements Engineering: Processes and Techniques*, John Wiley & Sons, 1998.
[18] D. Leffingwell, y D. Widrig, *Managing Software Requirements - A unified approach*. Addison-Wesley Object Technology Series, 2º edición, 2003.
[19] J.C.S.P. Leite, y A.P.M. Franco, "A Strategy for Conceptual Model Acquisition", *First Intl Symposium on Requirements Engineering, RE'93*, IEEE Computer Society Press, Los Alamitos, CA, 1993, pp.243-246.
[20] J.C.Leite, G.Hadad, J.Doorn, y G.Kaplan, "A Scenario Construction Process", *Requirements Engineering Journal*, Springer-Verlag, Vol.5, Nº1, 2000, pp.38-61.
[21] J.C. Leite, J.H. Doorn, G.D.S. Hadad, y G.N. Kaplan, "Scenario Inspections", *Requirements Engineering Journal*, Vol.10, Nº1, Springer-Verlag, 2005, pp. 1-21.
[22] P. Loucopoulos, y V. Karakostas, *System Requirements Engineering*, McGraw-Hill, Londres, 1995.
[23] J.Mylopoulos, L.Chung, y B.A. Nixon, "Representing and Using Nonfunctional Requirements: A Process-Oriented Approach", *IEEE TSE*, Vol.18, Nº6, 1992, pp. 483-497.
[24] NCC 87, National Computing Centre, *The STARTS Guide: A Guide to Methods and Software Tools for the Construction of Large Real-Time Systems*, 1987.
[25] B. Nuseibeh, J. Kramer, y A. Finkelstein, "A Framework for Expressing the Relationship between Multiple Views in Requirements Specification", *IEEE TSE*, Vol.20, Nº10, 1994, pp.760-773.
[26] W. Robinson, "Surfacing Requirements Interactions", *Perspectives on Software Requirements*, Kluwer Academic Publishers, capítulo 4, 2004, pp.69-90.
[27] T.L. Saaty, *The Analytic Hierarchy Process*, McGraw-Hill, 1980.
[28] P. Sawyer, y G. Kotonya, "Software Requirements", *SWEBOK, Guide to the Software Engineering Body of Knowledge*, Bourque y Dupuis (eds.), IEEE Computer Society, Los Alamitos, CA, cap.2, 2004, pp.2-1 -- 2-17.
[29] I. Sommerville, *Ingeniería de Software*, Addison-Wesley, 6º edición, cap. 5 y 6, 2002.
[30] J. Whitten, L.Bentley, y K.Dittman, *Systems Analysis and Design Methods*, Mc Graw-Hill/Irwin, 6º ed, cap.6, 2003.
[31] J. Yen, y W. Tiao, "A Systematic Tradeoff Analysis for Conflicting Imprecise Requirements", *RE'97, Third IEEE ISRE*, IEEE Computer Society Press, Enero 1997, pp.87-97.
[32] R.R. Young, *The Requirements Engineering Handbook*, Artech House, Norwood, 2004.
[33] R. Zultner, "Quality Function Deployment (QFD) for Software: Structured Requirements Exploration", *Total Quality Management for Software*, eds Schulmeyer & McManus, Van Nostrand Reinhold, 1992, pp. 297-317.
[34] V. Gervasi, "Reasoning about Inconsistencies in Natural Language Requirements", *ACM TSE and Methodology*, Vol.14, Nº3, 2005, pp.277-330.
[35] A. Hunter, y B. Nuseibeh, "Managing Inconsistent Specifications: Reasoning, Analysis and Action", *ACM TSE and Methodology*, Vol.7, Nº4, 1998, pp.335–367.
[36] A. Davis, "The Art of Requirements Triage", *IEEE Computer*, Vol.36, Nº3, 2003, pp. 42-49.
[37] S. Easterbrook, "Handling conflict between domain descriptions with computer-supported negotiation", *Knowledge Acquisition: An International Journal*, Vol.3, 1991, pp. 255-289.
[38] ISO 9126:2001, International Standard ISO/IEC 9126, Information technology - Software product evaluation - Quality characteristics and guidelines for their use, 2001.
[39] A. Egyed, y P. Grünbacher, "Identifying requirements conflicts and cooperation: How quality attributes and automated traceability can help", *IEEE Software*, 2004, pp.50-54.
[40] F. Pinheiro, "Requirements Traceability", *Perspectives on Software Requirements*, Kluwer Academic Publishers, cap.5, 2004, pp.91-113.
[41] D. Ross, y A. Schoman, "Structured analysis for requirements definition", *IEEE TSE, Special Issue on Requirements Analysis*, Vol.3, Nº1, 1977, pp. 6-15.
[42] G. Hadad, J. Doorn, M. Ridao, y G.Kaplan, "Facilitando la Asignación de Prioridades a los Requisitos", *WER'09*, aceptado para publicación, Valparaíso, Chile, Julio 2009.