Comprendiendo el Universo de Discurso Futuro con Escenarios

Jorge H. Doorn[1,2], Graciela D.S. Hadad[2,3], Gladys N. Kaplan[2,3]

[1] Universidad del Centro de la Provincia de Buenos Aires - INTIA
Pinto 399 (7000) Tandil, Bs. As., Argentina
[2] Universidad Tecnológica Nacional, Facultad Regional Buenos Aires
Medrano 951, Capital Federal, Argentina
[3] Universidad Nacional de La Plata - LINTI
Calle 50 esquina 115 (1900) La Plata, Bs. As., Argentina
jdoorn@exa.unicen.edu.ar, ghadad@ub.edu.ar, gladyskaplan@hotmail.com

Resumen. La definición de requisitos es meramente el producto final de la fase de requisitos, pero es el punto de partida basamental para el diseño del software y para la tarea de seguimiento de requisitos, y además en muchos casos es un documento contractual con los clientes. Para lograr una correcta definición, no sólo es necesario comprender el Universo de Discurso presente sino también comprender el Universo de Discurso futuro. Se plantean en este artículo alternativas para la construcción de Escenarios Futuros, utilizando los mismos como medio de representación parcial del Universo de Discurso futuro en el que se asume la presencia del sistema de software que se planifica desarrollar.

1 Introducción

La frase de von Neumann: "No tiene sentido en ser preciso cuando aún no se conoce de lo que se está hablando"[1] expresa con claridad el motivo por el cual muchos proyectos de software han fracasado. *Resolver el problema correcto* depende en gran medida de conocer cabalmente cuál es el problema. Es por ello que la Ingeniería de Requisitos apunta a mejorar la forma en que se comprende un problema para luego definir el sistema de software adecuado.

La estrategia utilizada por los autores para comprender el problema se basa en el uso de Escenarios, el que es un modelo de representación ampliamente difundido en la literatura [1] [2] [3] [4] y utilizado exitosamente en la práctica [5].

Dicha estrategia consiste en comprender el Universo de Discurso (UdeD), para ello se construye primero un Léxico Extendido del Lenguaje (LEL) y luego los Escenarios que modelan situaciones en el mismo [6].

El proceso de construcción del Léxico Extendido del Lenguaje y luego de los Escenarios es sólo el comienzo de una secuencia de actividades cuyo propósito consiste en elicitar, primero conocimiento del UdeD y luego el conjunto de los requisitos del sistema de software a ser desarrollado.

[1] "There is no sense in being precise about something when you don't know what you are talking about." es una frase expresada verbalmente por John von Neumann, citada por Michael Jackson en [JACK95], p.65.

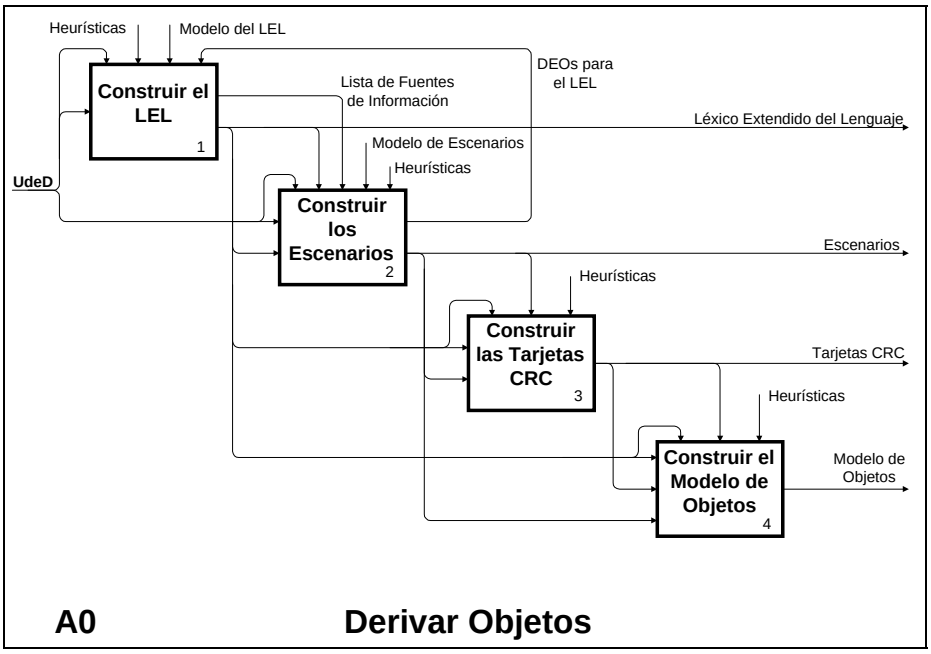

Fig. 1. Diagrama SADT del proceso de derivación de objetos.

Varios mecanismos se han utilizado para definir los límites y las responsabilidades del sistema de software. Por ejemplo la secuencia de actividades propuestas en [7], permite obtener las tarjetas CRC y el diagrama de objetos del UdeD. Este proceso, esquematizado en el SADT de la Figura 1, es eficaz en lograr su objetivo pero tiene como principal inconveniente la administración de los requisitos en forma implícita a lo largo de todo el proceso. Este inconveniente puede resultar menor en algunas circunstancias, pero resulta significativo en otras, como por ejemplo cuando se deben negociar contratos comerciales que contienen apéndices con listas de los requisitos a cumplir por el software o cuando se pretende una adecuada gestión de requisitos incluyendo la identificación de los componentes de diseño y de programas que los atienden [8].

En este artículo se propone un refinamiento del proceso de ingeniería de requisitos, expuesto en la Figura 2, que básicamente consiste en: primero comprender el UdeD presente, para ello se construyen el LEL y los Escenarios que representan situaciones actuales, segundo, comprender el UdeD futuro, construyendo Escenarios que representan situaciones del futuro, y finalmente, definir los requisitos del sistema de software basándose en el conocimiento adquirido y registrado en las etapas previas.

La etapa inicial de comprender el UdeD actual se basa en un enfoque consolidado [9], ampliamente probado y aplicado en la práctica [6]. En este artículo se presentan estrategias alternativas para la comprensión del UdeD futuro.

Estas estrategias presuponen siempre la construcción de un Léxico y de un conjunto de escenarios actuales como actividad indispensable para la comprensión del UdeD. Esto a su vez presupone que se está frente a un problema cuya envergadura o

complejidad hacen necesaria la comprensión del UdeD actual, para luego recién poder adentrarse en aspectos futuros. Es decir, el planteo de alternativas de estrategias para la construcción de escenarios futuros se basa en la siguiente hipótesis:

> *"El problema a solucionar es tal que se requiere la comprensión de la realidad actual."*

La discusión en este artículo se centra en casos que revisten esta característica.

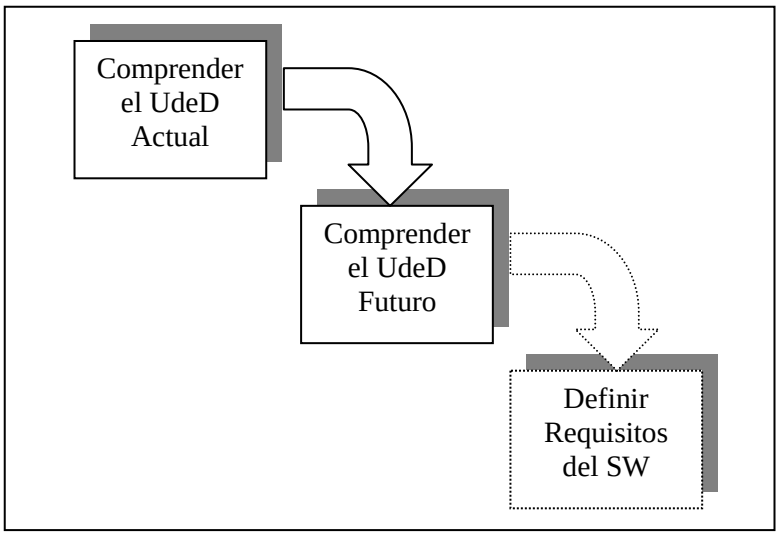

Fig. 2. Estrategia de definición de Requisitos.

El presente artículo está organizado como sigue: en la sección 2 se presenta el enfoque para modelar el UdeD actual, para ello se resumen sucintamente los procesos de construcción del Léxico Extendido del Lenguaje y de Escenarios; en la sección 3 se analizan distintas estrategias para la comprensión del UdeD futuro usando escenarios; finalmente en la sección 4 se indican los trabajos futuros basados en las estrategias alternativas propuestas y se exponen las conclusiones.

2. Modelado del UdeD actual

El enfoque para la comprensión del UdeD actual [10] no es ni top-down ni bottom-up ya que no se mira todo el UdeD en forma global construyendo unos pocos escenarios generales para luego refinarlos progresivamente, pero tampoco se procede a la búsqueda de episodios que adecuadamente sistematizados den lugar a escenarios, los que a su vez se puedan integrar en escenarios de mayor alcance. El enfoque utilizado es denominado middle-out [11] porque partiendo de un nivel medio avanza y retrocede de lo específico a lo general aprovechando las ventajas de los enfoques top-down y bottom-up. Se aplica el enfoque top-down cuando las situaciones

identificadas utilizando el LEL son refinadas y detalladas mediante escenarios. Es bottom-up cuando diferentes situaciones se reúnen en procesos o secuencias más amplias denominadas Escenarios Integradores. La estrategia para modelar el UdeD actual se basa en aplicar la técnica del "Léxico Extendido del Lenguaje" para la comprensión del vocabulario del problema y la técnica de "Escenarios" para la comprensión del comportamiento del problema actual.

2.1 Léxico Extendido del Lenguaje

La idea básica del léxico [12] es conocer el vocabulario de la aplicación y su semántica, dejando para un siguiente paso la comprensión del problema. El LEL está compuesto por un conjunto de símbolos que conforman el lenguaje de la aplicación y que representan, en general, palabras o frases que el cliente/usuario repite o enfatiza o son relevantes para el dominio más allá de su frecuencia de repetición.
La representación de los símbolos se realiza identificando cada símbolo con uno o más nombres y registrando para cada símbolo la *noción* (oraciones que lo definen) y el *impacto* (oraciones que indican cómo repercute en la aplicación).
La construcción del léxico parte de la identificación de las fuentes de información, luego se identifican los símbolos del dominio, se los clasifica, se los describe y, finalmente, el léxico sufre procesos de verificación y validación que retroalimentan el proceso de construcción en sí mismo.

2.2 Escenarios

Los Escenarios son narrativas estructuradas de situaciones del UdeD. Al igual que en el caso del LEL, los Escenarios se construyen utilizando el lenguaje natural con un mínimo de formalismo. Este formalismo limita y ordena el uso del lenguaje natural de una manera que exige un cierto esfuerzo del ingeniero de requisitos en su construcción pero que no limita la lectura y comprensión por parte del cliente/usuario, reduciendo de esta manera en forma sensible la brecha semántica entre ambos.
El modelo de Escenarios utilizado [6] está compuesto de título, objetivo, contexto, recursos, actores, episodios y excepciones. El título, el objetivo, el contexto, los recursos, los actores y las excepciones son declarativos, mientras que los episodios son un conjunto de sentencias que muestran descripciones operacionales de comportamiento de los actores en el escenario. El contexto, los recursos y los episodios pueden contener restricciones, las cuales están habitualmente referidas a requisitos no funcionales. Los episodios pueden ser en sí mismos Escenarios, posibilitando relaciones de jerarquía entre Escenarios y Subescenarios. Esta relación de jerarquía surge muy naturalmente cuando al descubrir una situación se prefiere no detallar otra situación de menor cuantía que es parte de la primera. Asimismo esta relación es utilizada para construir Escenarios Integradores, los cuales describen grupos de situaciones relacionadas. Estos Escenarios Integradores no tienen una unidad temporal o geográfica como los Escenarios particulares y se utilizan para adquirir una visión global del UdeD. Los Escenarios Integradores no deben confundirse con los Escenarios Generales, *"Escenarios Primarios"* [13], *"Escenario*

Fundamental" [14], *"Casos de Uso del Sistema"* [15], *"Casos de Uso de Nivel Alto"* [16], que se construyen como primera etapa con la intención de refinarlos posteriormente. Es decir, estos Escenarios Generales se construyen cuando se está utilizando una técnica top-down.

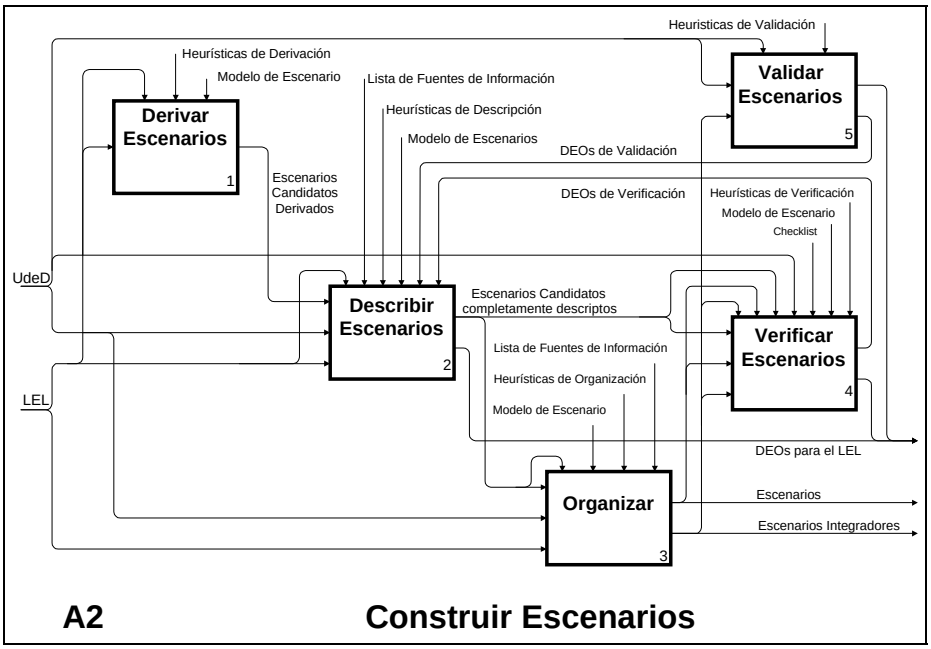

Fig. 3. Construcción de los Escenarios.

La Figura 3 ilustra, mediante un modelo SADT, las actividades del proceso de construcción de los Escenarios [10] [6]. La idea general de este proceso es "anclar" la descripción de Escenarios en el vocabulario del UdeD. Por tanto, el proceso de construcción de Escenarios parte del léxico del dominio de la aplicación, produciendo una primera versión de los Escenarios, derivados exclusivamente del LEL. Estos Escenarios son completados y mejorados utilizando otras fuentes de información y organizados con el fin de obtener un conjunto de Escenarios que representen el dominio de la aplicación. Durante o después de estas actividades, los Escenarios son verificados y validados con los clientes/usuarios.

3. Modelado del UdeD del futuro

Una vez que se ha concluido con la construcción del LEL actual, se procede a la construcción de los EA, tal como se esquematiza en la Figura 3. Concluida esta primera etapa, se debe proceder a la comprensión y modelado de los aspectos futuros del UdeD. La Figura 4 muestra una posible estrategia para la construcción de los EF y la definición de requisitos.

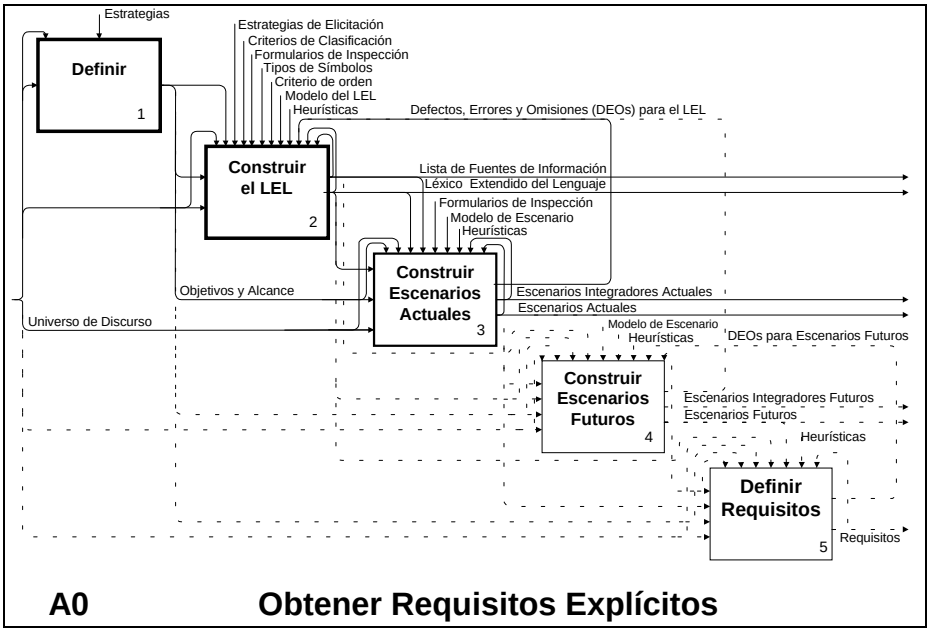

Fig. 4. Hipótesis acerca de la construcción de Escenarios Futuros y la obtención de Requisitos Explícitos

Es decir, en la fase de Ingeniería de Requisitos se propone la construcción de dos conjuntos separados de Escenarios denominados Escenarios Actuales (EA) y Escenarios Futuros (EF) para luego poder obtener requisitos en forma explícita. La obtención de los EA es parte del modelado del UdeD actual, descripto en la sección anterior. Los EA registran las situaciones del UdeD del presente e integran en forma coordinada y armónica lo que ocurre en él. Por otro lado, los EF modelan lo necesario en el futuro, y pueden representar también alternativas de solución al problema en estudio. El concepto de EF es utilizado en la literatura con variadas aplicaciones. El uso de EF para administrar cambios en sistemas actuales es analizado por Jarke et al. en [17]; un concepto similar para analizar la adquisición de sistemas COTS es presentado en [18]; otra propuesta de uso de EF para el proceso de diseño en combinación con objetivos y funciones es mostrada en [4].

Los EF tienen una relación estrecha con los EA ya que se refieren al mismo UdeD sobre el que se conjeturan evoluciones de las características del presente y se pretenden introducir modificaciones. Los EF son de fundamental importancia pues permiten modelar no sólo las características del sistema de software a ser desarrollado sino también la influencia de éste sobre los restantes componentes del UdeD y el contexto esperado para el mismo.

Muchos de los EA tienen una relación directa con los EF, pudiendo interpretarse éstos, en algunos casos, como modificaciones de situaciones actuales. Naturalmente que existen EF originados en situaciones no apareables en el UdeD presente, al mismo tiempo que situaciones del presente pierden vigencia y no aparecen como EF.

Es decir, los escenarios *evolucionan*, empiezan con una visión del UdeD tal cual es hoy (que puede o no tener un sistema de software) y se transforman para considerar la existencia de un sistema de software o el reemplazo o adaptación de uno existente. Esto implica:

- que generalmente en el futuro estará muy presente la palabra *Sistema*, y
- que hay una gran posibilidad de cambio en la interacción del sistema con el ambiente.

La evolución de los EA a los EF, cualquiera sea la estrategia constructiva, debe ser cuidadosa y meditada. Esta actividad es muy relevante ya que en ella se definen las propiedades esenciales del sistema de software a ser construido.
La decisión de incorporar un sistema de software en una organización puede deberse a una gran variedad de razones. Estas razones por las que se decidió la incorporación del sistema se deben constituir en el principio generador de los EF ya que cada uno de ellos debe ser juzgado desde el punto de vista de estas razones, las que se constituyen en meta-objetivos.
Estrictamente entonces el sistema de software debe concebirse para un contexto que no siempre es exactamente el observable en la actualidad y modelado utilizando los EA. Aún cuando no existan supuestos de cambios en el UdeD[2], la mera transformación directa de los EA en EF conlleva en algunos casos al importante riesgo de automatizar lo que existe, perdiéndose de ese modo la oportunidad de reemplazar procedimientos y actividades manuales por otras equivalentes pero más eficientes desde el punto de vista del procesamiento automático de información. La fuerza de esta afirmación se centra en la palabra "equivalentes". Un EA y un EF son equivalentes si ambos tienen el mismo objetivo y ambos lo satisfacen plenamente.
Si por el contrario se desea introducir modificaciones en el UdeD, es deseable que los objetivos del EA y del EF difieran, ya que en esa diferencia estará materializándose las razones por las que se planea implantar el sistema de software.
Naturalmente que en la realidad no siempre es posible lograr un apareo uno a uno entre EA y EF por lo que la noción de equivalencia se debe aplicar en algunos casos a grupos de escenarios. Esto implica que los apareos podrán ser del tipo *n* EA frente a *m* EF. Por otro lado, debe tenerse en cuenta que pueden surgir nuevos EF no apareables con ningún EA ya que se deberán cumplir objetivos no existentes en el UdeD actual o el caso inverso quedar algún EA sin ningún apareo con situaciones futuras, los cuales serán descartados para el UdeD futuro.
Retomando el caso simple en el que un EA evoluciona para producir un EF equivalente, se debe enfatizar que no hay un solo EF posible. Por el contrario, en la mayoría de las situaciones más de una solución puede obtenerse para el mismo problema. Definir el EF más apropiado para un cierto contexto es la responsabilidad esencial de la Ingeniería de Requisitos.
La evolución de los EA a los EF requiere necesariamente la aplicación de mecanismos de generalización o de abstracción en algunos momentos y de detalle o especialización en otros. Cuando por primera vez el ingeniero de requisitos se enfrenta a una situación del UdeD casi siempre observa actores realizando actividades

[2] Cambios en el UdeD pueden deberse por ejemplo a cambios en la organización, mejoras en la producción, expansión de actividades, etc.

en un cierto contexto. Habitualmente el propósito de las mismas no es suficientemente explícito para poder enmarcarlo claramente en el UdeD. Es aquí donde interviene la abstracción ya que el ingeniero de requisitos debe precisar el objetivo durante la construcción del EA que describe la situación. Al construir el EF el ingeniero de requisitos debe imaginarse un futuro posible haciendo jugar un rol preponderante al sistema de software. Es más, si resulta posible más de un EF, el ingeniero de requisitos junto con el cliente deben seleccionar el mejor.

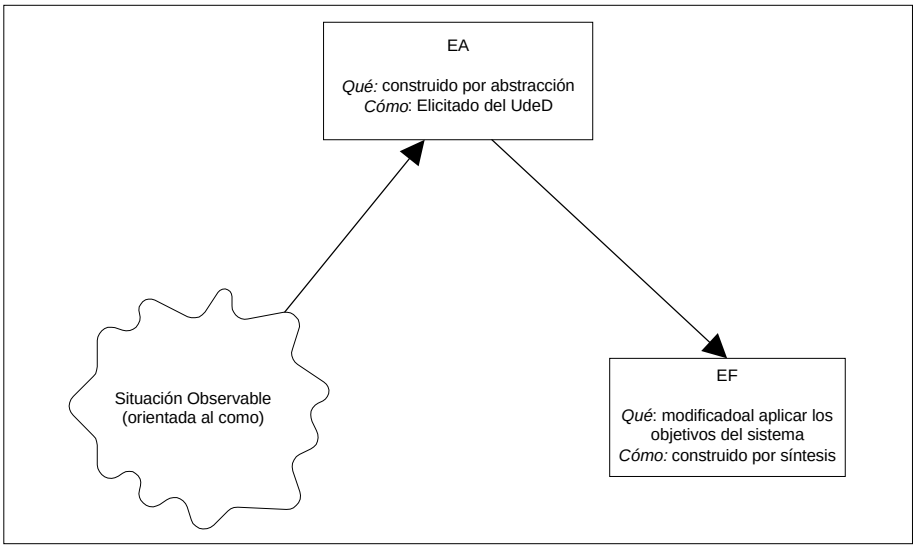

Fig. 5. Abstracción y concreción en la construcción de EF

Es importante resaltar en este punto que la situación futura se analiza desde el punto de vista de QUÉ debe hacerse y CÓMO el sistema interactúa con el resto del UdeD. Debe notarse especialmente que el punto de vista del CÓMO se desarrollan las actividades está aplicado al contexto del sistema y no al sistema mismo. La definición de CÓMO el sistema de software proveerá los servicios que se definen en esta etapa se realiza con posterioridad en el momento del diseño del software. La línea divisoria entre la definición de los servicios que brindará el sistema y la estructura de su implementación es sutil pero su importancia es innegable, debiéndose evitar el diseño prematuro tanto como sea posible.
Resumiendo, cuando el sistema de software es el instrumento para llevar a cabo un meta-objetivo (objetivo del sistema), la relación entre los objetivos del EA y EF son potencialmente diferentes y el desafío es definir el objetivo del EF utilizando el objetivo del EA y el meta-objetivo del problema. En la Figura 5 se esquematizan las nociones anteriores. Nuevamente, se indica que se está mostrando una evolución de un EA en un EF. Si la transformación fuera más compleja, el apareo de los objetivos se cumple sobre un conjunto de escenarios y no sobre cada escenario individual.

3.1 Estrategias para construir Escenarios Futuros

Para la construcción de EF existen varios caminos viables que se pueden clasificar según dos dimensiones: enfoque y estrategia. El enfoque se refiere a la aplicación de un método *top-down*, *bottom-up* o *middle-out*. En cuanto a la estrategia, ésta indica la forma concreta de realizar la construcción y puede ser: *en bloque*, *en paralelo* o *incremental*. Considerando que la dimensión enfoque [19] [20] es ampliamente conocida en la comunidad informática, se describirá brevemente sólo la dimensión estrategia.

La estrategia *en bloque* o *global* consiste en construir los EF partiendo directamente de las necesidades acerca del UdeD futuro, sin usar como base los EA; éstos se consultan sólo como fuente de información acerca del presente. Esta estrategia tiende a presentar el menor anclaje posible con lo actual y dejar librado a la "imaginación"[3] la construcción del futuro.

La estrategia *en paralelo* consiste en la construcción de los EF al mismo tiempo que los EA. Es decir, se debe recabar información del futuro, simultáneamente con la información del presente. Esta estrategia pretende reducir los tiempos de la fase de Ingeniería de Requisitos.

La estrategia *incremental* construye los EF por transformaciones sucesivas de los EA incorporando paulatinamente información acerca del futuro. Esta estrategia surge de la idea de aprovechar lo más posible lo existente para mejorarlo; es decir, su fundamento es anclarse en lo actual promoviendo mejoras a partir de ello.

La Tabla 1 muestra en forma esquemática las combinaciones posibles entre enfoque y estrategia. Esta tabla se analiza en el contexto presupuesto en la introducción, es decir, el ingeniero de requisitos requiere la comprensión del UdeD actual.

Se debe observar que las celdas grisadas en la Tabla indican combinaciones inviables o contradictorias. Tal es el caso de una estrategia en paralelo, ya sea con un enfoque top-down o bottom-up, resulta totalmente incompatible con nuestro enfoque middle-out para la comprensión del UdeD actual, pues se estarían aplicando simultáneamente dos enfoques opuestos para la construcción de dos conjuntos de escenarios: los EA y los EF. Por otro lado, esta estrategia indica la elicitación y modelado concurrente de información de la realidad actual junto con las necesidades futuras, lo que exige una importante dispersión de la atención del ingeniero de requisitos en el momento menos propicio, es decir, cuando realiza su primera aproximación al UdeD. Por lo tanto, la estrategia en paralelo parece ser la menos viable y, por ende, no será considerada como una opción a analizar.

Tanto la estrategia en bloque como la incremental se caracterizan por preservar el mecanismo ya desarrollado y probado para la construcción de los EA, y sólo plantea diferencias para la construcción de los EF. Es decir, en todos los casos se tiene la misma secuencia temporal ya que se construyen los EF cuando ya existen los EA, con lo cual, el ingeniero de requisitos comienza a ocuparse de un problema cuando el anterior está razonablemente concluido y, por ende, tiene una comprensión acabada del UdeD presente.

[3] Se refiere a abrir el campo de opciones posibles acerca de cómo podría ser el UdeD, disminuyendo la influencia de lo existente.

Tabla 1. Combinaciones posibles entre enfoques y estrategias

	TOP-DOWN	MIDDLE-OUT	BOTTOM-UP
BLOQUE	**1ª Etapa** LEL Actual Escenarios Actuales **2ª Etapa** **Propone procesos globales** **EF generales** **EF particulares** **EF integradores**	**1ª Etapa** LEL Actual Escenarios Actuales **2ª Etapa** **Propone situaciones** **EF particulares** **EF integradores**	**1ª Etapa** LEL Actual Escenarios Actuales **2ª Etapa** **Propone activ. especificas** **Episodios Futuros** **EF particulares** **EF integradores**
PARALELO		**LEL** **Escenarios Actuales** **Escenarios Futuros**	
INCREMENTAL	**1ª Etapa** LEL Actual Escenarios Actuales **2ª Etapa** **Modifica EA integradores** **EF generales** **EF particulares** **EF integradores**	**1ª Etapa** LEL Actual Escenarios Actuales **2ª Etapa** **Modifica EA particulares** **EF particulares** **EF integradores**	**1ª Etapa** LEL Actual Escenarios Actuales **2ª Etapa** **Modifica episodios act.** **Episodios Futuros** **EF particulares** **EF integradores**

Por otro lado, el enfoque middle-out puede interpretarse desde dos concepciones diferentes, una se basa en la construcción del LEL con la visión del vocabulario del futuro y la otra encara la construcción directamente de EF analizando situaciones posibles del futuro. La primera se descarta pues el LEL no es un instrumento generador sino de registración, es decir, permite registrar un vocabulario en uso pero no generarlo. Luego, la construcción del LEL Futuro podría estar involucrando un vocabulario ficticio, ya que se estaría avizorando el vocabulario a utilizar con un sistema futuro; esto viola los principios intrínsecos del LEL. En lo que sigue, cuando se menciona el enfoque middle-out, éste se basa en la construcción inicial de EF a un nivel medio de especificidad, agrupando, de ser necesario, los detalles en subescenarios y, finalmente, se construyen los EF Integradores (EFI), que son aquellos de más alto nivel en la jerarquía de escenarios.

En lo que sigue, cada combinación enfoque-estrategia se la denomina *alternativa*. Habiéndose descartado la estrategia paralelo, se tiene que todas las alternativas construyen el LEL y luego los EA. A partir de este punto las alternativas divergen proponiendo cada una de ellas una secuencia de actividades diferente. No todas las alternativas son igualmente promisorias por lo que se procede a un breve análisis de las mismas.

Alternativa top-down y en bloque: por su enfoque top-down consiste en un avance de lo general a lo específico y por su estrategia en bloque propone construir los EF sin

basarse en los EA. Es decir, esta variante consiste en revisar el alcance del sistema de software, revisar cada objetivo del mismo e imaginar grupos de situaciones que permitan cumplir con los objetivos, construyendo EF generales. Luego, precisar las situaciones que cumplan con las demandas y necesidades elicitadas, construyendo los EF particulares (EFP). Esta alternativa da lugar a heurísticas muy diferentes de las utilizadas en la primera etapa, y si bien desaprovecha la guía proporcionada por los EA, evita la influencia de lo "actual" permitiendo ampliar el espectro de propuestas sobre el futuro.

Alternativa top-down e incremental: por su estrategia incremental se propone modificar los EA paso a paso para obtener los EF. Por su enfoque top-down se debe partir de los EA integradores (EAI) y sólo proceder a las transformaciones de los niveles inferiores de la jerarquía cuando se haya completado razonablemente el nivel superior. Esto indica que los EAI se transforman en EF generales, y los EA particulares (EAP) en EF particulares. Durante estas actividades se elicita cómo se modifica cada EA para cumplir con los objetivos del sistema de software. Naturalmente que, no todo EF tiene su correlato en un EA y viceversa.

Alternativa middle-out y en bloque: por su adhesión al enfoque middle-out y su independencia de los EA, permite basar la construcción de los EF en la elicitación de situaciones futuras, donde el ingeniero de requisitos se imagina actores futuros (nuevos o existentes) y sus actividades futuras, gestando entonces los EFP. Un actor principal de estos EFP será el "sistema de software" que puede o no existir en el UdeD actual pero obviamente cobra mayor relevancia en el UdeD futuro. Luego, se construyen los EFI aplicando las heurísticas de integración utilizadas en la primera etapa [6].

Alternativa middle-out e incremental: la estrategia incremental impulsa la evolución paso a paso desde los EA a los EF. Luego, consiste en proponer objetivos para los EF que surgen de los EA y, en función de ellos, revisar cada EA. Si ningún EA cumple con dicho objetivo, se identifica una nueva situación que se volcará en un EF. Si existe algún EA que cumple parcialmente con el objetivo, se modifica el EA obteniendo el EF. Se debe considerar que pueden existir relaciones *n a m* en las transformaciones entre EA y EF.

Alternativa bottom-up y en bloque: el enfoque bottom-up indica avanzar desde lo específico a lo general, lo que presupone también una técnica de captura de información diferente. Por lo tanto, comienza capturando información detallada del futuro (demandas específicas o acciones futuras de los actores dentro de los límites y objetivos planteados) volcándola en episodios futuros que se agrupan en EF de situaciones específicas y, luego se construyen EFI aplicando las mismas heurísticas de integración expuestas en la primera etapa. Para obtener información del futuro, se utilizan fuentes de información con demandas concretas.

Alternativa bottom-up e incremental: utiliza la misma técnica de captura de información expuesta en la alternativa anterior pero las preguntas guía se basan en el seguimiento de los EA. Por ende, consiste en la revisión de los episodios de los EA y su transformación según la visión de lo futuro (objetivos del sistema) en episodios de EF, luego se integran los EF para obtener una visión global de la aplicación futura. Naturalmente que esto da lugar a heurísticas muy diferentes de las utilizadas en la primera etapa, aunque se aprovecha la misma heurística de integración.

3.2 Comparación de las alternativas

La opción entre la estrategia en bloque, independizándose de los EA, y la estrategia incremental, justamente basándose en los mismos, no es inmediata, ya que las ventajas de cualquiera de ellas pueden ser superadas por sus inconvenientes. Es más, es concebible que diferentes problemas impulsen la adopción de estrategias diferentes.
La estrategia incremental establece una mayor adhesión al presente del UdeD, por lo que se propugna básicamente desarrollar sistemas de software con menor grado de innovación, mejores posibilidades de implantación y menor perturbación del UdeD. Por el contrario, la estrategia en bloque se independiza en mayor medida del UdeD actual, por lo que naturalmente puede generar un mayor grado de innovación, con sus consecuentes riesgos de implantación.
Desde el punto de vista de la reingeniería de procesos, una estrategia incremental podrá ser más adecuada cuando se requiere menos reingeniería; en el caso inverso, la necesidad de una mayor reingeniería, induce a seguir una estrategia en bloque.
Dado que las alternativas analizadas no requieren como anclaje un LEL en la descripción de los EF, deberá evaluarse, mediante casos de estudio, la conveniencia de construir en forma retrospectiva un "glosario del futuro" con una estructura similar a la del LEL. En este glosario se incorporarían nuevos términos surgidos recién en los EF, se modificarían, agregarían o eliminarían nociones e impactos en términos ya existentes en el UdeD actual y se descartarían términos existentes no empleados en los EF. Esta propuesta sólo se justifica de requerirse un glosario que clarifique los términos empleados en la descripción de los EF. De no ser este el caso, una opción apropiada, que cumple con los principios del LEL, sería construir un LEL una vez implantado el sistema de software, con lo cual el LEL reflejaría el vocabulario del nuevo UdeD.
En cualquiera de las alternativas presentadas, los EF serán verificados y validados para asegurar su calidad, de manera tal de avanzar a la tercera etapa de definición de requisitos basándose en documentos confiables.

3.3 Casos de estudio

Se han construido los EF en tres casos reales:

- *Sistema de Control de Calidad de productos de un laboratorio farmacéutico.(A)*
- *Sistema de Compras, Producción e Instalación de Cajeros Automáticos.(B)*
- *Sistema de Mesa de Ayuda a servicio técnico de equipamiento electrónico.(C)*

Los dos primeros se realizaron utilizando la alternativa top-down e incremental y en el tercero se aplicó middle-out y en bloque.
En la Tabla 2 se resumen los principales datos que caracterizan los tres conjuntos de escenarios.

Tabla 2. Resumen de los casos reales.

Caso	EFP	Sub-escenarios	Episodios	Excepciones	Actores	Sujetos LEL	Recursos	Objetos LEL	EFI
A	49	30	275	9	11	6	81	74	5
B	25	21	167	11	12	10	28	17	4
C	37	25	181	5	4	3	16	13	3

El uso de la estrategia incremental en los dos primeros sistemas mostró claramente sus ventajas, ya que la construcción de los EF se pudo realizar en forma ordenada. Dado que los objetivos presentados por el cliente eran de muy alto nivel y que existía una exigencia explícita de automatizar los procesos existentes, la guía ofrecida por los EA facilitó la construcción de los EF. Los EF generales construidos a partir de los EAI fueron un documento intermedio utilizado como guía para la construcción de los EFP y luego fueron descartados. La construcción de los EFI al final de la etapa evitó el costoso mantenimiento de los EG obtenidos inicialmente.

Fig. 6. Evolución de escenarios particulares para el caso Control de Calidad

En la figura 6 se presenta la evolución desde los escenarios actuales particulares a escenarios futuros para el caso Control de Calidad, y en la figura 7 se presenta la evolución para los escenarios integradores en el mismo caso.

4. Conclusiones

Se ha presentado un marco teórico que integra las diversas visiones del UdeD y se han presentado formas concretas de incorporar las novedades producidas por este marco teórico en la técnica de representación utilizada hasta el momento sin alterar en modo alguno ni su propósito, ni su sintaxis, ni su contenido semántico.
Se ha avanzado de esta manera en un camino que tiene muy buenas posibilidades de suministrar los requisitos del sistema de software en forma explícita. Este hecho es en opinión de los autores especialmente relevante.
Los tres casos realizados no son suficientes para categorizar las ventajas de las alternativas unas respecto de las otras, aunque evidencian que las utilizadas son apropiadas en el entorno en el que fueron utilizadas. En futuros trabajos se estudiarán y compararán las alternativas en forma más amplia.
Mucho más análisis de casos de estudio debe ser realizado para contrastar las hipótesis introducidas y para dilucidar la viabilidad de la obtención de requisitos explícitos y la cuantificación de las ventajas y los inconvenientes mencionados.

Por otro lado, cabe mencionar que la fase de definición de requisitos de software es un tema en curso preliminar de investigación.

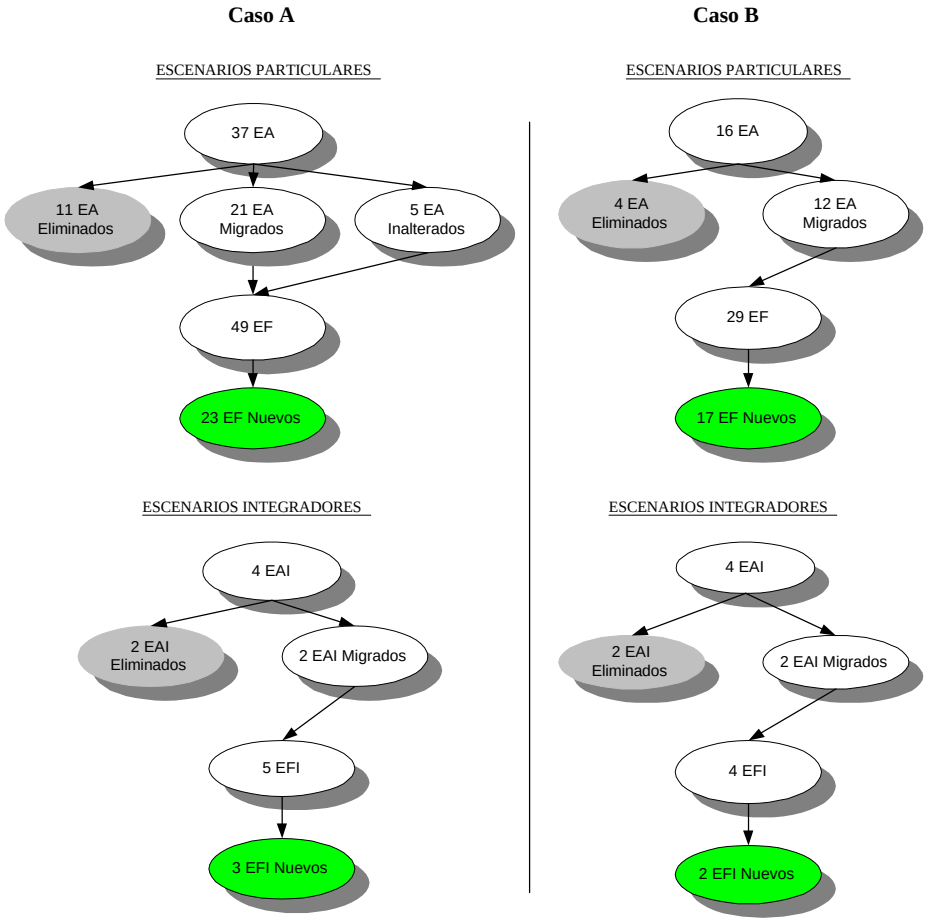

Fig.6. Evolución de los Escenarios Particulares y los Escenarios Integradores en los Casos A y B

Referencias

1. Potts, C.: Using Schematic Scenarios to Understand User Needs, Proceedings of DIS'95 - Symposium on Designing Interactive Systems: Processes, Practices and Techniques, ACM Press, University of Michigan, 1995, pp. 247-256.
2. Sutcliffe, A.G., Maiden, N.A.M., Minocha, S., Manuel, D.: Supporting Scenario-Based Requirements Engineering, IEEE Transactions on Software Engineering, Vol. 24, N° 12, 1998, pp. 1072-1088.
3. Rolland, C., Souveyet, C, Ben Achour, C.: Guiding Goal Modeling Using Scenarios, IEEE Transactions on Software Engineering, Vol. 24, N° 12, 1998, pp. 1055 – 1071.
4. Kaindl, H.: A Design Process Based on a Model Combining Scenarios with Goals and Functions, IEEE Transactions on Systems, Man and Cybernetic, Vol. 30, Nº 5, September 2000, pp. 537-551.
5. Weidenhaupt, K., Pohl, K., Jarke, M., Haumer, P.: Scenarios in System Development: Current Practice, IEEE Software, 1998, pp. 34-45.
6. Leite, J.C.S.P., Hadad, G.D.S., Doorn, J.H., Kaplan, G.N.: A Scenario Construction Process, Requirements Engineering Journal, Vol.5, N° 1, 2000, pp. 38-61.
7. C. Leonardi, V. Maiorana, F. Balaguer Una Estrategia de Análisis Orientada a Objetos Basada en Escenarios, Actas de II Jornadas de Ingeniería de Software, JIS97, Dpto de Informatica, Universidad del pais vasco, San Sebastían, España, 1997, pp. 87-100.
8. Paulk, M., Weber, C., Curtis, B., Chrissis, M.: The Capability Maturity Model: Guidelines for Improving the Software Process, Reading, MA: Addison-Wesley, 1995.
9. Leite, J.C.S.P., Rossi, G., Balaguer, F., Maiorana, V., Kaplan, G., Hadad, G., Oliveros, A.: Enhancing Requirements Baseline with Scenarios, Requirements Engineering Journal, Vol. 2, Nro. 4, (1997), pp. 184-198.
10. Hadad, G.D.S., Kaplan, G.N., Oliveros, A., Leite, J.C.S.P.: Construcción de Escenarios a partir del Léxico extendido del Lenguaje, 26 JAIIO, Buenos Aires (1997), pp 65-77.
11. Hadad, G.D.S., Doorn, J.H., Kaplan, G.N., Leite, J.C.S.P. Enfoque middle-out en la Construcción e Integración de Escenarios, Anales del Wer'99, Argentina, 1999, pp 79-94.
12. Leite, J.C.S.P., Franco, A.P.M.: O Uso de Hipertexto na Elicitaçao de Linguagens da Aplicaçao, Anais de IV Simpósio Brasilero de Engenharia de Software, SBC, Brazil, 1990, pp. 134-149.
13. Booch, G.: Scenarios, Report on Object Analysis and Design, Vol. 1, N° 3, 1994, pp. 3-6.
14. Sutcliffe, A.: A Technique Combination Approach to Requirements Engineering, Proceedings of the Third IEEE International Symposium on Requirements Engineering, IEEE Computer Society Press, Los Alamitos, Colorado, 1997, pp. 65-74.
15. Wirfs-Brock, R.: Designing Objects and Their Interactions: A Brief Look at Responsibility-Driven Design, Scenario-Based Design: Envisioning Work and Technology in System Development, J. Carroll (ed.), John Wiley, New York, 1995, pp. 337-359
16. Schneider, G., Winters, J.: Applying Use Cases, A Practical Guide, Addison-Wesley, Reading, MA, 1998.
17. Jarke, M., Tung Bui , X., Carroll, J.: Scenario Management: An Interdisciplinary Approach, Requirements Engineering Journal, Vol.3, N° 3 y 4, 1998, pp. 155-173.
18 Feblowitz, M.D., Greenspan, S.J.: Scenario-Based Analysis of COTS Acquisition Impacts, Requirements Engineering Journal, Vol.3, N° 3 & 4, 1998, pp. 182-201.
19. Jourdon, E.: Modern Structured Analysis, Englewood Cliffs, NJ: Yourdon Press, 1989
20. Jackson, M., Software Requirements & Specifications. A lexicon of practice, principles and prejudices, Addison Wesley, ACM Press, 1995.