GENERACIÓN DEL DIAGRAMA DE CASOS DE USO A PARTIR DEL LENGUAJE NATURAL O CONTROLADO: UNA REVISIÓN CRÍTICA

USE CASE DIAGRAM GENERATION FROM NAURAL OR CONTROLLED LANGUAGE: A CRITICAL REVIEW

CARLOS ZAPATA
Grupo de Lenguajes Computacionales, Universidad Nacional de Colombia, cmzapata@unalmed.edu.co

PAULA TAMAYO
Grupo de Lenguajes Computacionales, Universidad Nacional de Colombia, patamayo@unalmed.edu.co

Recibido para revisar octubre 4 de 2007, aceptado noviembre 1 de 2007, versión final enero 21 de 2008

RESUMEN: El diagrama de casos de uso es importante en el desarrollo de aplicaciones de software para capturar los requisitos funcionales y para manejar la complejidad de sistemas robustos. En este artículo, se presenta una revisión crítica de los trabajos relacionados con la obtención del diagrama de casos de uso, partiendo de representaciones del discurso del interesado en lenguaje natural o controlado. De esta revisión, se concluye que el proceso suele partir de representaciones difíciles de conseguir en las etapas iniciales del software, que aún se realiza de forma asistida por el analista y que es todavía incompleto, pues no se identifican las relaciones especiales entre los actores y los casos de uso del diagrama.

PALABRAS CLAVE: Diagrama de casos de uso, lenguaje natural, lenguaje controlado, superestructura de UML 2.0.

ABSTRACT: Use case diagram is useful in software application development in order to capture functional requirements and to manage robust system complexity. We present, in this paper, a critical review of works concerned to the use case diagram obtaining from stakeholder discourse representations, in the form of natural or controlled language discourses. We conclude, from this review, that such representations are difficult to obtain in the previous stages of software development, when the analyst must subjectively influence the process, and when the process is still incomplete, due to the lack of special actor-use-case relationship identification.

KEYWORDS: Use case diagram, natural language, controlled language, UML 2.0 superstructure.

1. INTRODUCCIÓN

En la especificación de la Superestructura del *Unified Modeling Language* UML [1], el diagrama de casos de uso se define como el "diagrama que muestra las relaciones entre los actores y el sujeto (sistema) y los casos de uso. Jacobson [2] y Jacobson *et al.* [3] introdujeron el diagrama de casos de uso, que describe los requisitos funcionales del sistema en términos de las secuencias de acciones". En OMG [1], Schach [4] y Fowler [5] se presentan los siguientes elementos de su especificación:

- Casos de uso: son las especificaciones de un conjunto de acciones realizadas por el actor sobre el sistema.
- Actores: son los roles que los usuarios desempeñan respecto del sistema y que emplean los casos de uso.
- Relaciones: identifican la comunicación existente entre actores y casos de uso [6]. Las relaciones pueden ser de cuatro tipos:
  - Asociación: se establece entre los actores y casos de uso.

- <<*include*>>: se presenta cuando el caso de uso origen incluye también el comportamiento descrito por el caso de uso destino [7].
- <<*extend*>>: ocurre cuando el caso de uso origen extiende el comportamiento del caso de uso destino [7].
- <<*inheritance*>>: un caso de uso origen hereda la especificación del caso de uso destino y posiblemente la modifica y/o amplía. Este tipo de relación también se presenta entre los actores.

La representación gráfica de los elementos del diagrama de casos de uso se puede apreciar en la Figura 1. Las principales ventajas de utilizar este diagrama, según Firesmith [8] son:

- La captura de los requisitos funcionales desde el punto de vista del usuario.
- La utilización de los casos de uso para educir y documentar los requisitos funcionales.
- El manejo de la complejidad en sistemas robustos, descomponiendo el problema en funciones más simples.

Figura 1. Elementos del diagrama de casos de uso [5]
Figure 1. Use case diagram elements [5]

La obtención automática o semiautomática del diagrama de casos de uso es un tema importante en la Ingeniería de requisitos puesto que, si se puede acortar el tiempo en la elaboración de estos diagramas, una aplicación de software se podría conceptualizar en un tiempo menor. En este tema, el punto de partida puede ser lenguaje natural o controlado, buscando garantizar que los requisitos del interesado se reflejen en el sistema obtenido. Empero, aún subsisten problemas, tales como:

- El lenguaje de partida es un lenguaje controlado que debe mencionar la funcionalidad de la aplicación de software.
- El Analista completa el diagrama de manera subjetiva en la mayoría de los trabajos.
- No se identifican todos los elementos que componen el diagrama de casos de uso, en especial, las relaciones <<*include*>>, <<*extend*>> e <<*inheritance*>>.

En este artículo, se realiza una revisión crítica de la obtención automática o semiautomática del diagrama de casos de uso a partir de discursos en lenguaje natural o controlado, empleando para ello la siguiente estructura: en la Sección 2, se presentan los principales trabajos en el tópico en estudio; en la Sección 3, se determinan los principales problemas que quedan aún por resolver; finalmente, en la Sección 4, se presentan las conclusiones y el trabajo futuro que se pueden derivar de esta revisión.

2. DIAGRAMA DE CASOS DE USO Y SU OBTENCION A PARTIR DE LENGUAJES NATURALES O CONTROLADOS

Los principales trabajos en este tema, categorizados de acuerdo con las principales falencias identificadas, se discuten en el desarrollo de esta sección.

2.1 El resultado es la descripción textual de los casos de uso

El principal interés en estos trabajos, radica en la obtención de los casos de uso, los actores y las relaciones de asociación.
Ben Achour [9] establece un diálogo interesado-analista que permite especificar los elementos principales de la descripción de los casos de uso. Para ello, Ben Achour [10] define un conjunto de reglas de clarificación, análisis, completitud, mapeo e integración del discurso y, finalmente, obtiene algunos de los actores y los casos de uso, además de la asociación entre estos dos elementos. El-Ramly *et al.* [11 y 12], proporcionan una aproximación dinámica llamada "*CellEST*", que identifica los casos de uso a partir del análisis de una característica particular del comportamiento dinámico de un sistema, llamada "interacción usuario-sistema". Esta aproximación permite identificar los casos de uso, pero no identifica los

demás elementos de este diagrama definidos en OMG [1].
Por último, Insfrán *et al.* [13 y 14] emplean el modelo de requisitos para capturar las características de un sistema. Este modelo emplea tres técnicas complementarias: la Misión del Sistema (MS), el Árbol de Refinamiento Funcional (ARF) y el Modelo de Casos de Uso (MCU). En primer lugar, se organiza un ARF que contiene como nodo raíz la MS y se identifican todas las funciones o interacciones externas del sistema. Luego, cada función elemental del ARF (nodo hoja) se representa como un caso de uso y el analista define los actores (usuarios) que participan en su realización.

2.2 El diagrama de casos de uso no es el objetivo final

En esta sección se describen los principales trabajos que obtienen el diagrama de casos de uso como etapa intermedia para la obtención de otros diagramas
Díaz *et al.* [15] proponen una serie de reglas que, a partir de la descripción textual de los casos de uso, permite obtener el diagrama de secuencias de UML 1.4. Por medio de estas reglas es posible extender el esquema definido para obtener algunos de los elementos del diagrama de casos de uso, como los actores y los casos de uso. Como una continuación de este trabajo, Díaz *et al.* [16] definen patrones para escribir en oraciones simples la especificación de los casos de uso y las relaciones especiales <<*include*>> y <<*extend*>>, por medio de plantillas específicas.
Liu [17] y Liu *et al.* [18 y 19] generan el diagrama de clases a partir de un discurso en lenguaje natural, teniendo como pasos intermedios la construcción del diagrama de casos de uso y la especificación de los mismos. Este proceso incluye el análisis del lenguaje natural, la transformación de los requisitos en sentencias y la representación de cada sentencia mediante una estructura sintáctica. Con base en estas sentencias se derivan los posibles actores desde los sustantivos y los posibles casos de uso desde los verbos que actúan como predicado de los actores. Las principales falencias de este proceso radican en que no se identifican todos los elementos del diagrama y se necesita la intervención del analista para validar y completar el modelo. A partir de estos trabajos, Subramanian *et al.* [20 y 21] implementan una herramienta llamada "Asistente para el desarrollo guiado de casos de uso UCDA" que permite obtener el diagrama de casos de uso y, a partir de él, obtener el diagrama de clases.
Anandha *et al.* [22] obtienen, desde la especificación de los requisitos en lenguaje natural, los actores y casos de uso del diagrama de casos de uso, los nombres de las clases, las operaciones, los atributos y las relaciones de asociación del diagrama de clases, en un proceso realizado por la herramienta "*Requirements Elicitor"*, con los siguientes pasos:

- Fraccionamiento de la frase: la declaración del problema se divide en sentencias.
- Etiquetado: cada sentencia recibe etiquetas que marcan cada una de las palabras.
- Reconocimiento de partes de la oración: se identifican el sustantivo y los sintagmas verbales.
- Resolución de la referencia: Los sujetos y objetos que actúan como pronombres se convierten en sustantivos.
- Normalización de la estructura: El texto se simplifica para llevarlo a la forma sujeto-verbo-objeto.
- Mapeo: se identifican los elementos del diagrama de casos de uso.

Giganto [23] obtiene los diagramas de secuencias y de clases a partir de las especificaciones de los casos de uso. Estas especificaciones se obtienen del documento de requisitos expresado en un lenguaje controlado, el cual posee un vocabulario limitado. El sistema que permite obtener estos elementos se denomina *ReoCASE* y funciona mediante dos módulos:

- Procesamiento del lenguaje natural: Se divide en dos submódulos. El primero, se encarga de producir el árbol de análisis de la sentencia y, el segundo, genera un caso de uso inicial. Luego de generar el caso de uso inicial, se consulta la base de datos de casos de uso para sincronizar las dos versiones, produciendo unos casos de uso finales, que se almacenan en la base de datos para su reutilización.
- Ingeniería de Software: Se compone de un analizador y un diseñador orientado a objetos.

Muestra los diagramas de clases y de secuencias obtenidos.

Esta aproximación no contempla la obtención de las relaciones presentes en el diagrama de casos de uso y requiere un repositorio de estos para llevar a cabo la comparación.

2.3 El punto de partida es código fuente

Algunos investigadores proponen la obtención del diagrama de casos de uso a partir del código fuente de un programa; esta aproximación requiere que el sistema ya esté construido y, por tanto, que el proceso de desarrollo de software se encuentre en sus fases finales.
Qin *et al.* [24] y Zhang *et al.* [25] representan, mediante un grafo llamado "*Branch-Reserving Call Graph* (BRCG)", la estructura de un programa; para construir el grafo BRCG desde el código fuente se realizan dos pasos:

- Construcción de un subgrafo para cada procedimiento.
- Conexión de todos los subgrafos en un sólo.

Posteriormente, visualizan los casos de uso como rastros de una ejecución del programa, mediante la aplicación de un algoritmo. Los casos de uso se identifican desde los diferentes segmentos del código fuente y, las asociaciones de los casos de uso, mediante el llamado a los procedimientos. Un experto en el dominio debe identificar las relaciones especiales entre los casos de uso.
Di Lucca *et al.* [26] y Bernardi y Di Lucca [27 y 28], obtienen el diagrama de casos de uso a partir de un código orientado a objetos. Esta aproximación extrae los métodos y los mensajes intercambiados entre objetos, a través del análisis estático del código, y los convierte en un caso de uso de bajo nivel. Para completar el diagrama obtenido, es necesario que un experto intervenga.
Por último, Elsberry y Elsberry [29] crean, a partir del *Extensible Markup Language* (XML), diagramas UML tales como el diagrama de caso de uso y el diagrama de clases. Los componentes del diagrama, se representan como datos XML y la transformación se detalla por medio de un grafo de vector escalable dinámico (SVG). Para llevar a cabo esta transformación, el desarrollo incluye:

- El Esquema de XML.
- El modelo de los archivos de datos de XML (véase la Figura 2).
- La plantilla del estilo de la transformación.

El documento XML se transforma sincronizando los elementos de este documento con los modelos. Luego, estos modelos se asocian con los modelos definidos en las plantillas del archivo XSLT (*Extensible Stylesheet Language Transformations*). En la Figura 3 se puede observar la implementación del programa que permite sincronizar el elemento "actor". Esta aproximación sólo le ayuda al analista a realizar el diagrama, sin necesidad de que éste conozca acerca de las herramientas CASE que también lo permiten realizar.

```
<actor actorID="customer">
<name>Customer</name>
<summary>Bank ATM  customer </summary>
<initiates>withdrawMoney</initiates>  </actor>
```

Figura 2. Datos XML para el elemento actor [29]
Figure 2. XML data of the actor element [29]

```
<!-- Actor Group -->
<g id="actor{$gid}" onmouseover="showInfo(evt,'actorHover{$gid}')"
                    onmouseout="hideInfo(evt,'actorHover{$gid}')">  <!-- Actor's name -->
 <text x="{$xOffset}" y="{$yOffset + 70}" style="fill:black; stroke: none; font-size:12; text-anchor: middle;">
 <xsl:value-of select="//actor[@actorID = $varInitiator]/name" /> </text>
 <!-- Actor's head -->  <circle cx="{$xOffset}" cy="{$yOffset}" r="10" /> 10  <!-- Actor's body -->
 <line x1="{$xOffset}" y1="{$yOffset + 10}" x2="{$xOffset}" y2="{$yOffset + 35}" />  <!--Actor's arms -->
 <line x1="{$xOffset - 15}" y1="{$yOffset + 20}" x2="{$xOffset + 15}" y2="{$yOffset + 20}" />
```

Figura 3. Código de la transformación para el actor [29]
Figure 3. Source code to transform the actor element [29]

2.4 El punto de partida es un esquema conceptual

Existen diversos diagramas que presentan similitudes con el diagrama de casos de uso, como el diagrama de procesos, el modelo de tareas y el diagrama de actividades, entre otros, que permiten identificar el diagrama de casos de uso.
Dijkman y Joosten [30 y 31] mapean, desde el modelo de procesos del negocio, algunos elementos del diagrama de casos de uso. Esta aproximación define un mapeo inicial basado en las definiciones de los conceptos y relaciones.
En la Tabla 1, se puede observar el mapeo entre los conceptos del modelo de procesos del negocio y los conceptos de los casos de uso. Mediante esta aproximación se obtienen los actores, los casos de uso y las relaciones entre estos dos elementos, sin obtener las relaciones especiales. Esto se debe a que esta aproximación busca obtener la especificación de los casos de uso y no su diagrama.
Garcia *et al.* [32 y 33] proponen otra aproximación, que se basa en el modelo de procesos del negocio además de un esquema conceptual. Los pasos de este proceso son:

- Identificar y delimitar los procesos del negocio. Para cada proceso del negocio, se define un caso de uso.
- Descubrir los roles involucrados en el proceso del negocio y su descripción en el modelo de roles.

Tabla 1. Relación entre los conceptos del modelo de proceso del negocio y los casos de uso [30]
Table 1. Relationship between business process model concepts and use cases [30]

Concepto modelo de procesos del negocio	Concepto del caso de uso
Rol	Actor
Paso	Caso de uso
Asociación entre rol y paso	Asociación entre actor y caso de uso
Tarea	Interacción
Tarea en un paso	Interacción en un caso de uso
Transición entre tareas en el mismo paso	Ordenamiento entre interacciones de un casos de uso
Restricción en una transición	Restricción en una interacción
Ruta alternativa en una rama	Descripción para una ruta alternativa o un caso de uso extendido

- Modelar el flujo del trabajo para cada proceso del negocio mediante el diagrama de actividades.
- Extraer los casos de uso desde las actividades que constituyen el caso de uso del negocio.
- Establecer el modelo conceptual de los datos en el diagrama de actividades.

El principal interés de esta aproximación es la obtención del diagrama de casos de uso del negocio y la identificación de los actores. Las actividades en el diagrama de procesos del negocio que soporta el sistema se transforman en los casos de uso y el rol que realiza la actividad se transforma en un actor.
Štolfa y Vondrák [34 y 35], emplean los métodos "mapeo uno a uno" y "mapeo de varias acciones a casos de uso" para la conversión entre el diagrama de actividades y el diagrama de casos de uso. En esta aproximación, el caso de uso general puede ser el caso de uso origen para una relación <<*include*>> o <<*extend*>> con otro caso de uso, mediante un proceso que se realiza así [36]:

- <<*include*>> si se derivan desde acciones secuenciales.
- <<*extend*>> si se derivan desde acciones opcionale.

Mediante estos trabajos se obtienen manualmente los casos de uso y las relaciones especiales <<*include*>> y <<*extend*>>, sin obtener la relación <<*inheritance*>>.
Zapata y Alvarez [37] realizan la conversión de diagramas de procesos en diagramas de casos de uso utilizando AToM^3®. Las reglas de consistencia que permiten la conversión entre los dos modelos son las siguientes:

- Los actores están presentes en ambos modelos; por lo tanto, un actor o unidad organizacional en el diagrama de procesos se podrá convertir directamente en un actor del diagrama de casos de uso.
- Los procesos del diagrama de procesos, se pueden asimilar a funciones o procesos del caso de uso.
- Los procesos, pertenecen al carril de responsabilidad de un actor en el diagrama de

procesos. Esto genera una asociación entre el actor y el caso de uso que el actor inicia.

Artim [38] reconoce que hay una similitud fuerte entre los casos de uso y el modelo de tareas. Por otra parte, Lu *et al.* [39] establecen que esta similitud también se presenta con el diagrama de casos de uso y el diagrama de interacción, debido a que son semánticamente similares al modelo de tareas. Para el diagrama de casos de uso, Lu *et al.* [40] identifican los actores y las relaciones *<<include>>* y *<<extend>>*, de la siguiente manera: el actor se define implícitamente mediante el atributo "*style*" de la tarea. La relación *<<include>>* se identifica a partir de los enlaces lógicos y temporales del modelo de tareas, mientras la relación *<<extend>>* se identifica mediante los enlaces condicionales del modelo. Las reglas definidas para la obtención de estos elementos, se ligan estrechamente con TAMOT, que es una herramienta de modelado para la notación específica del modelo de tareas, la cual emplea el formalismo *Diane+* [41]. Esta aproximación, depende, en su totalidad, del formalismo utilizado, por lo que limita al usuario a utilizar una herramienta para la elaboración del modelo de tareas.

Lopez *et al.* [42 y 43] obtienen la descripción de los casos de uso a partir de un diagrama documental de tareas. La funcionalidad del sistema se modela a través del grafo de casos GC. Del análisis del GC se obtienen familias de casos de uso del negocio y familias de casos de uso del sistema. A partir de un GC, se pueden extraer, automáticamente, los casos de uso, para lo cual se requiere un proceso algorítmico. Con este proceso, se realiza el refinamiento y la transformación de tareas, que permite que el GC se exprese como la combinación de grafos de casos de uso. Empleando esta aproximación se pueden obtener manualmente los casos de uso, los actores y las relaciones de asociación entre estos elementos del diagrama de casos de uso.

Kösters *et al.* [44-46], modelan los casos de uso a partir de los diagramas de actividades, mediante una descripción informal de la acción. En el diagrama de actividades cada acción se marca con uno de los siguientes estereotipos *<<contextual action>>*, *<<interaction>>*, *<<macro action>>*, *<<actor in action>>*. El estereotipo *<<macro action>>* determina en el diagrama de casos de uso una relación *<<include>>* o *<<extend>>*. El estereotipo *<<actor in action>>* dibuja un actor que se puede conectar a una acción. Una acción con el estereotipo *<<contextual action>>* determina un caso de uso y el estereotipo *<<interaction>>* determina una acción que realiza el sistema. Los estereotipos, y su correspondencia con el diagrama de casos de uso, se pueden observar en la Tabla 2.

Stamper *et al*. [47] y Liu *et al*. [48], definen las normas como el conjunto de reglas o patrones de comportamiento. En particular, para modelar los procesos de negocios se utilizan las normas denominadas de comportamiento, debido a que prescriben qué pueden, deben y no deben hacer las personas. En estas normas, se especifican las condiciones y las acciones que se deben cumplir.

Tabla 2. Estereotipos del diagrama de actividades y su representación en el diagrama de casos de uso [44]
Table 2. Activity diagram stereotypes and their representation in use case diagram [44]

ESTEREOTIPO	DESCRIPCIÓN
<<contextual action>>	Representa la ejecución de una acción que realiza un actor sin la ayuda del sistema.
<<interaction>>	Representa una acción que el sistema soporta o ayuda.
<<actor in action>>	Dibuja un actor que se puede conectar a una acción, es decir, a un caso de uso.
<<macro action>>	Determina un subdibujo que representa una asociación *<<include>>* o *<<extend>>*.

En la Figura 4 se puede observar la regla de comportamiento para la suscripción de un cliente o un hotel en un sistema de reservas.

whenever
<El Cliente/Hotel decide usar el HBS>
if <El Cliente/Hotel inicializa la suscripción>
then < El Cliente/Hotel >
is <obligado>
to <Pagar la subscripción >

Figura 4. Estructura de reglas de comportamiento [44]
Figure 4. Behavioral rules structure [44]

A partir del análisis de las normas de comportamiento, Shishkov [49], Shishkov *et al*. [50] y Shishkov y Dietz [51] derivan los componentes del diagrama de casos de uso.

El análisis de responsabilidades de estas normas, permite identificar y asignar agentes responsables a cada acción. El proceso para obtener el diagrama de casos de uso se realiza a través de cuatro fases definidas en la metodología DEMO [52], en donde se realiza el análisis de las diferentes normas de comportamiento:

- Análisis semántico: Con base en la descripción textual y en la delimitación del dominio del problema, se aplica un mapa ontológico para conducir el análisis semántico.
- Normas de alto nivel: Según Shishkov [49], este proceso se realiza mediante la comparación de los requisitos especificados por el interesado con un repositorio de estándares de procesos de negocio.
- Normas de bajo nivel: se derivan las normas de bajo nivel o las normas de comportamiento para cada instancia.
- Diagrama de casos de uso: Se identifican los casos de uso y se construye el diagrama basado en las normas obtenidas.

3. PROBLEMAS A RESOLVER

En esta Sección se discuten más ampliamente los problemas remanentes en esta tendencia.

3.1 No se identifican todos los elementos del diagrama

Para la elaboración automática del diagrama de casos de uso, a partir de las especificaciones verbales de los requisitos, es necesario identificar cada una de los elementos de este diagrama. A este respecto, ninguno de los trabajos presentados realiza la identificación completa de esos elementos. En especial, las relaciones especiales *<<include>>*, *<<extend>>* e *<<inheritance>>*, correspondientes a la superestructura de UML 2.0 [1], son las que menos se identifican.

3.2 Intervención del analista

Casi todos los trabajos requieren una alta participación del analista para tomar las decisiones pertinentes a la generación del diagrama de casos de uso. La automatización en la generación de un diagrama, se concibe para aliviar la carga de un analista en la generación de conceptos del diagrama y, por ello, la máquina debería ser quien tomara la mayor cantidad de decisiones en este aspecto. En particular, la automatización de la elaboración del diagrama de casos de uso, haría que el analista participara mucho más en labores de análisis y no en labores de trazado asistido por las herramientas CASE actuales, que sólo cumplen la función de "asistentes de dibujo" en este tema en particular. Además, la automatización contribuye a disminuir la cantidad de errores humanos que se involucran en tareas repetitivas, como la identificación de los elementos de un diagrama. De esta manera, la labor del analista se podría especializar hacia tareas menos repetitivas y mucho más analíticas.

3.3 Origen del discurso del problema

En la mayoría de los trabajos, se suele utilizar como punto de partida un discurso en lenguaje natural o controlado que describe el "sistema". Tal es el caso de las especificaciones textuales de los casos de uso o las especificaciones detalladas que se emplean en dos de los trabajos. Si bien, este punto de partida permite la obtención del diagrama, un discurso tal sólo se puede obtener después de transcurrir gran parte de la etapa de análisis del problema e incluso requiere una solución ya definida al mismo. Si se procura la obtención temprana del diagrama, el punto de partida debería ser una descripción del dominio del problema y no de su solución.

4. CONCLUSIONES Y TRABAJO FUTURO

En la educción de requisitos de software cobran fuerza los intentos por automatizar la elaboración de los diferentes esquemas conceptuales, debido a que se reducen los tiempos de entrega y, por tanto, los costos en los que se incurre en el proceso de desarrollo de software. Además, se reduce el riesgo inducido por los errores humanos y se permite realizar seguimiento de los requisitos a lo largo del proceso de desarrollo de software.

En este artículo se evaluó la problemática de la elaboración automática o semiautomática del diagrama de casos de uso, tomando como punto de partida lenguaje natural o controlado. Esta evaluación, permitió identificar tres problemas:

(1) se suele partir de representaciones de la solución y no de representaciones del problema; (2) se requiere aún una alta participación del analista en el proceso; (3) no se identifican todos los elementos del diagrama de casos de uso.

La evaluación de las diferentes propuestas se compendia en la Tabla 3. Allí, se presentan cuatro características de las propuestas: la intervención del analista, si poseen o no un lenguaje orientado a la solución, si generan los diferentes diagramas o no y los elementos que identifican del diagrama de casos de uso. Las convenciones relativas a los elementos identificados son las siguientes: "A" cuando la identificación es automática, "SA" si el elemento se identifica de manera semiautomática con la intervención del analista y "NO" si la propuesta no realiza la identificación del elemento en particular.

Tomando como base la evaluación realizada, se pueden sugerir varios aspectos susceptibles de iniciar trabajos de investigación, tales como:

- La representación del dominio del problema mediante un lenguaje controlado, que incluya características que posibiliten la identificación posterior de los diferentes elementos del diagrama de casos de uso, incluyendo las relaciones especiales, que hasta ahora poco se identifican.
- La definición misma de las reglas heurísticas que permitan la identificación de esos elementos y otros como los *extension points.*
- La construcción o complementación de las herramientas CASE que automaticen la concepción misma de los diferentes diagramas, para propender por la disminución de errores en el proceso de educción de requisitos y posibilitar a los analistas la generación y utilización de buenas prácticas de desarrollo de software, en lugar de emplear el tiempo en las labores repetitivas que hacen parte del proceso de educción de requisitos.

Tabla 3. Resumen de los trabajos en obtención del diagrama de casos de uso
Table 3. A summary of works in use case diagram obtaining

Autores	Inter-vención analista	Leng. orient. solución	Gen. Diag.	Elementos identificados					
				Casos de uso	Actor	Asoc	include	Extend	Inher
Anandha *et al.*	NO	NO	NO	A	A	NO	NO	NO	NO
Ben Achour	SI	NO	NO	A	A	A	NO	NO	NO
Di Lucca *et al* y Bernardi y Di Lucca	SI	SI	SI	SA	SA	SA	NO	NO	NO
Diaz *et al.*,	SI	NO	NO	NO	A	NO	SA	SA	NO
Dijkman y Joosten	NO	SI	NO	A	A	A	NO	A	NO
El-Ramly	NO	SI	NO	A	NO	NO	NO	NO	NO
Elsberry y Elsberry	SI	SI	SI	SA	SA	SA	NO	NO	NO
Garcia *et al.*	NO	SI	SI	A	A	A	NO	NO	NO
Giganto	NO	NO	SI	A	A	NO	NO	NO	NO
Insfrán *et al.*	SI	NO	SI	SA	SA	NO	NO	NO	NO
Kösters *et al.*	SI	SI	SI	A	A	NO	NO	NO	NO
Liu, Liu *et al*, y Subramanian *et al.*	SI	NO	SI	SA	SA	SA	NO	NO	NO
Lopez, *et al.*	NO	SI	NO	A	A	A	NO	NO	NO
Lu *et al.*	NO	SI	SI	SA	SA	NO	NO	NO	NO
Quin *et al.* y Zhang *et al.*	SI	SI	SI	SA	A	A	SA	SA	SA
Shishkov *et al.,* Shishkov Shishkov y Dietz	SI	NO	SI	A	A	A	SA	SA	NO
Štolfa y Vondrák.	SI	SI	SI	SA	SA	SA	SA	SA	NO
Zapata y Alvarez	NO	SI	SI	A	A	A	NO	NO	NO

REFERENCIAS

[1] OMG. Unified Modeling Language: Superstructure version 2.0. Final adopted specification ptc/03-08-02. Needham, US. Object Management Group, Inc., 2003.

[2] JACOBSON, I., Object-Orien. Development in an Industrial Environment, Special issue of SIGPLAN Notices, 22(12), 183–191, 1987.

[3] JACOBSON, I., CHRISTERSON, M., JONSSON, P., AND OVERGAARD, G., Object-Oriented Software Engineering: A Use Case Driven Approach. Add.-Wesley, New York, 1992.

[4] SCHACH, S., Análisis y diseño orientado a objetos con UML y el proceso unificado, McGraw-Hill Interamericana, Mexico, D.F., 2004.

[5] FOWLER, M., UML Distilled: A brief guide to the Standard Object Modeling Language, Addison-Wesley, Reading, 2004.

[6] AMESCUA, A., Análisis y diseño estructurado y orientado a objetos de sistemas informáticos, McGraw-Hill Interamericana, Madrid, 2003.

[7] COCKBURN, A., Writing Effective Use Cases, Addison-Wesley Pub. Co, Reading, 2000.

[8] FIRESMITH, D. G., Use case: the pros and cons. En: Wisdom of the Gurus: A Vision for Object Technology (Ed. Ch. F. Bowman), SIGS Books Inc., New York, 171–180, 1996.

[9] BEN ACHOUR, C., Guiding the construction of textual case use specifications, Data & Know. Eng. Journal, 25(1-2), 125–160, 1998.

[10] BEN ACHOUR, C., Extraction des bensoins par analyse of scenarios textuels, [PhD Thesis], Paris, Universidad de Paris, 1999.

[11] EL-RAMLY, M., STROULIA, E., AND SORENSON, P., Mining system–user interaction traces for use case models, Proceedings of 10th International Workshop on Program Comprehension, Paris, Francia, 21–29, 2002.

[12] EL-RAMLY, M., STROULIA E., AND SORENSON, P., Recovering software requirements from system-user interaction traces, Proceedings of the 14th International Conference on Software Engineering and Knowledge Engineering, Ischia, Italia, 447-454, 2002.

[13] INSFRÁN, E., DÍAZ, I., AND BURBANO, M., Modelado de Requisitos para la Obtención de Modelos Conceptuales, Memorias del V Workshop Iberoamericano de Ing. de Requisitos y Amb. Software (IDEAS), La Habana, Cuba, 2002

[14] INSFRÁN, E., TEJADILLOS, E., MARTÍ, S., AND BURBANO, M., Transformación de Especificación de Requisitos en Esquemas Conceptuales usando Diagramas de Interacción, Memorias del Workshop en Ingeniería de Requisitos, Valencia, España, 91–105, 2002.

[15] DIAZ, I., MORENO, L., AND PASTOR, O., Traducción de casos de uso en patrones de interacción de instancias: una aproximación lingüística, Memorias de las III Jornadas Iberoam. de Ingeniería de Software e Ingeniería del Conocimiento (JIISIC), Valdivia, Chile, 2003.

[16] DIAZ, I., LOSAVIO, F., MATTEO, A., AND PASTOR, O., Specification pattern of use cases. Inform. and Manag., 41, 961-975, 2004.

[17] LIU, D., Automating Transition from Use Cases to Class Model, [MSc Thesis], Calgary, University of Calgary, 2003.

[18] LIU, D., SUBRAMANIAM, K., FAR, B.H., AND EBERLEIN, A., Automating transition from use-cases to class model, Proc. of the IEEE Canadian Conf. on Electrical and Computer Eng., Montreal, Canada, v. 2, 831–834, 2003.

[19] LIU, D., SUBRAMANIAM, K., EBERLEIN, A., AND FAR B., Natural Language Requirements Analysis and Class Model Generation Using UCDA, Lecture Notes in Artificial Intelligence, 3029, 295–304, 2004.

[20] SUBRAMANIAM, K., LIU, D., FAR, B. H., AND EBERLEIN, A., UCDA: use case driven development assistant tool for class model generation, Proceedings of the 16th International Conference on Software Engineering and Knowledge Engineering (SEKE'04), Banff, Canada, 324–329, 2004.

[21] SUBRAMANIAM, I., AND FAR, B.H., Automating Transition from Stakeholder requests to Use cases, Proceedings of the IEEE Canadian Conf. on Electrical and Computer Engineering, Niagara Falls, Canada, 515–518, 2004.

[22] ANANDHA, G., JAYARADIKA, J., AND UMA, G., Restructuring Natural Language Text to Elicit Software Requirements, Proceedings of the International Conference on Cognition and Recognition, Mandya, India, 521-525, 2005.

[23] GIGANTO, R. T., Extracting Use Cases for Class and Sequence Diagrams Generation, Proceedings of the 5th New Zealand Computer Science Research Student Conference, Hamilton, Nueva Zelanda, 2007.

[24] QIN, T., ZHANG, L., ZHOU, Z., HAO, D., AND SUN, J., Discovering use cases from source code using the branch-reserving call graph, Proceedings of the 10th Asia-Pacific Software Engineering Conference, Chiangmai, Tailandia, 60–67, 2003.

[25] ZHANG, L., QIN, T., ZHOU, Z., HAO, D., AND SUN, J., Identifying use cases in source code, Journal of Systems and Software, 79(11), 1588–1598, 2006.

[26] DI LUCCA, G. A., FASOLINO, A. R., AND DE CARLINI, U., Recovering use case models from object-oriented code: a thread-based approach, Proceedings of 7th Working Conference on Reverse Engineering, Brisbane, Australia, 108–177, 2000.

[27] BERNARDI, M. L., AND DI LUCCA, G. A., Supporting the Comprehension of Object-Oriented Software Systems by Extended M-M Graph, Proceedings of the IASTED International Conference on Software Engineering, Innsbruck, Austria, 55–60, 2005.

[28] BERNARDI, M. L., AND DI LUCCA, G. A., UsCaAb: A Tool for Abstracting Use Case Diagrams, Proceedings of the Software Maintenance and Reengineering, CSMR 2005, Manchester, Reino Unido, 194–194, 2005.

[29] ELSBERRY, J., AND ELSBERRY, N., Using XML and SVG to Generate Dynamic UML Diagrams, Technical Report, Ellensburg, Central Washington University, 2003

[30] DIJKMAN, R., AND JOOSTEN, S., Deriving use case diagrams from business process models, Technical Report series 08 (02), CTIT, Enschede, Holanda, 2002.

[31] DIJKMAN, R. M., AND JOOSTEN, S. M., An Algorithm to Derive Use Case Diagrams from Business Process Models, Proc. of the 6th Intern. Conf. on Software Engineering and Applications (SEA), Anaheim, US, 679-684, 2002.

[32] GARCÍA, J., ORTÍN, J., MOROS, B., NICOLÁS, J., AND TOVAL, J., Toward Use Case and Conceptual Models through Business Modeling, Proceedings of the 19th International Conference on Conceptual Modeling, Utah, US, 281–294, 2000.

[33] GARCÍA, J., ORTÍN, J., MOROS, B., NICOLÁS, J., AND TOVAL, J., De los Procesos del Negocio a los Casos de Uso, Memorias de las V Jornadas Ing. de Software y Bases de Datos, Valladolid, España, 103–116, 2000.

[34] ŠTOLFA, S., AND VONDRÁK, I., Using the Business Process for Use Case Model Creation, Proceedings of the international conference on Information Systems Implementation and Modelling ISIM '03. Brno, República Checa, 129–137, 2003.

[35] ŠTOLFA, S., AND VONDRÁK, I., An Explanation of Automatized Transformation Procedure from Business Processes to Use Case Diagrams, Proceedings of the international conference on Information Systems Implementation and Modelling ISIM'04, Rožnov pod Radhoštm, República Checa, 101–107, 2004.

[36] ŠTOLFA, S., AND VONDRÁK, I., Using Business Modeling Methods for Requirements Specification, Proceedings of the SCI 2004, Orlando, US, vol. IV, 298–302, 2004.

[37] ZAPATA, C., AND ALVAREZ, C., Conversión de Diagramas de Procesos en Diagramas de Casos de Uso Usando AToM^3, Revista Dyna, 146, 103–113, 2005.

[38] ARTIM, J. M., Integrating user interface design and object-oriented development through task analysis and use cases, Proceedings of the CHI'97 workshop on Object Oriented User Interfaces, Atlanta, US, 1997.

[39] LU, S., PARIS, C., AND VANDER, L., Integrating task modelling into the object-oriented design process: a pragmatic approach. Proceedings of the CHI'98 workshop on Incorporating Work, Processes and Task Analysis into Industrial Object-Oriented Systems Design, Los Angeles, US, 1998.

[40] LU, S., PARIS, C., VANDER, K., AND COLINEAU N., Generating UML Diagrams from Task Models, Proceedings of the CHINZ'03, 4th Annual International Conference of the New Zealand, chapter of the ACM's SIGCHI, Dunedin, Nueva Zelanda, 9–14, 2003.

[41] TARBY, J. C., AND MARIE F. B., The DIANE+ Method, Proc. of the 2nd Intern. Workshop on Computer-Aided Design of User Interfaces, Namur, Bélgica, 95–119, 1996.

[42] LÓPEZ, O., LAGUNA, M. A., AND MARQUÉS, J. M., Normalización de Assets de Requisitos en el Contexto de la Reutilización Sistemática del Software, Memorias de las V Jornadas de Trabajo MENHIR, Granada, España, 25–35, 2000.

[43] LOPEZ, O., LAGUNA, M., AND MARQUÉS, M., Generación Automática de Casos de Uso para desarrollo de software basado en reutilización. Memorias de las V Jornadas de Ingeniería del Software y Bases de Datos, Valladolid, España, 89–101, 2000.

[44] KÖSTERS, G., PAGEL, B-U., AND WINTER, M., Coupling Use Cases and Class Models, Proceedings of the Workshop on Making Object Oriented Methods More Rigorous, London, UK, 27–30, 1997.

[45] KÖSTERS, G., SIX, H-W., AND WINTER M., Validation and Verification of Use Cases and Class Models, Proceedings of the 6th Requirements Engineering for Software Quality, Stockholm, Suecia, 2000.

[46] KÖSTERS, G., SIX, H.-W., AND WINTER, M., Coupling Use Cases and Class Models as a Means for Validation and Verification of Requirements Specifications, Requirements Engineering, 6(1), 3–17, 2001.

[47] STAMPER, R. K., LIU, K., HAFKAMP, M., AND ADES, Y., Signs plus norms: One paradigm for organizational semiotics, Proceedings of the 1st Intern. Workshop on Computational Semiotics, Paris, Francia, 1997.

[48] LIU, K., SUN, L., DIX, A., AND NARASIPURAM, M., Norm-based Agency for Designing Collaborative Information Systems. Info Systems Journal, 11, 229–247, 2001.

[49] SHISHKOV, B., Business Engineering Building Blocks, Proc. of the 9th Doctoral Consortium on Adv. Information Systems Eng. (CAiSE'02), Toronto y Ontario, Canadá, 2002.

[50] SHISHKOV, B., XIE, Z., LUI, K., AND DIETZ, J., Using norm analysis to derive use case from business processes, Proceedings of the 5th Workshop on Organizations semiotics. Delft, Holanda, 187–195, 2002.

[51] SHISHKOV, B., AND DIETZ, J., Design of Software Applications Using Generic Business Components, Proceedings of the 37th Hawaii International Conference on System Sciences, Big Island, US, 2004.

[52] DIETZ, J., Understanding and Modelling Business Processes with DEMO, Proceedings of the 18th International Conference on Conceptual Modeling, Paris, Francia, 1999.