Reutilización del Software a partir de Requisitos Funcionales en el Modelo de Mecano: Comparación de Escenarios

Oscar López
Instituto Tecnológico de Costa Rica
olopez@infor.uva.es

Miguel Ángel Laguna
Universidad de Valladolid
mlaguna@infor.uva.es

José Manuel Marqués
Universidad de Valladolid
jmmc@infor.uva.es

Resumen

Este artículo propone una aproximación para incorporar el proceso de reutilización desde las etapas iniciales del ciclo de vida del software. La propuesta se basa en generalizar escenarios del problema y compararlos con escenarios genéricos. Los primeros se obtienen con una herramienta automatizada que está en fase de desarrollo. La generalización se realiza mediante reducción de redes de Petri. Los escenarios genéricos se almacenan en un repositorio asociados a estructuras complejas de reutilización denominadas mecanos que enlazan elementos de análisis, diseño e implementación. La comparación se realiza mediante analogía. De este modo, con una estrategia para reutilizar software desde los requisitos funcionales, se ofrece una alternativa para acelerar el desarrollo de soluciones software con reutilización a la vez que se eleva el nivel de abstracción de los elementos reutilizables. La motivación principal del artículo es que los requisitos representan el conocimiento más abstracto del dominio y un alto porcentaje de ellos son reutilizables. ***Palabras Clave. Reutilización del software, ingeniería de requisitos, escenarios, redes de Petri.***

1. Introducción.

La era de la información y las crecientes presiones para obtener software de calidad y en el menor tiempo posible acentúan la necesidad de aprovechar óptimamente el esfuerzo de desarrollo. Por esta razón se ha propuesto la reutilización como una respuesta para incrementar significativamente la productividad a la vez que se mantiene o se incrementa la calidad del software [28]. Para la adecuada explotación de la reutilización se requieren estrategias maduras, por lo que la investigación actual se centra en mejorar los modelos conocidos de reutilización del software.

Las ventajas potenciales de la reutilización estimulan nuevos planteamientos dentro de la ingeniería del software. La reutilización sistemática requiere una organización apropiada y una cultura idónea [17][5]. Se deben revisar las prácticas de administración, las estructuras de organización y las tecnologías utilizadas para explotar eficientemente los elementos reutilizables [28][8]. Se debe contar con herramientas de apoyo, con desarrolladores entrenados y con una visión de largo plazo para reutilizar el software como producto lógico o simbólico, que no se manufactura, ni se deteriora, más bien se desarrolla y se adapta a los cambios [27].

Un aspecto esencial de la reutilización del software es la delimitación de los artefactos reutilizables. Intuitivamente, la reutilización se asocia de manera directa con la utilización de código fuente para construir un nuevo producto software. Sin embargo, en sentido amplio, un elemento reutilizable es cualquier producto del ciclo de vida del software con potencial de reutilización: modelos y arquitecturas de dominio, requisitos, diseño, código, componentes de bases de datos, documentación, y pruebas [18].

En el Grupo de Investigación en Reutilización y Orientación al Objeto (GIRO) se ha desarrollado el modelo de reutilización MRG, (siglas de Modelo de Reutilización GIRO). El MRG es una alternativa para el desarrollo de software en el marco de la reutilización sistemática y con el soporte de estructuras complejas denominadas mecanos que están formadas por elementos de análisis, diseño e implementación. El modelo técnico, el modelo de proceso y el modelo de cualificación forman los ejes del MRG y encierran los aspectos conceptuales, logísticos y de certificación de mecanos [11]. El MRG es una aproximación para aprovechar las ventajas de la reutilización del software e incorporar artefactos de diferentes niveles de abstracción.

De las opciones abiertas a partir del modelo de mecano aparecen como prioritarias el soporte de dominios y líneas de producto [10], en las cuales la gestión de los requisitos es una actividad esencial. Idealmente, la reutilización consiste en hacer uso del conocimiento en su forma más abstracta [26]. Los requisitos representan el nivel más abstracto de conocimiento sobre un dominio particular por lo que al reutilizar a partir de los requisitos se incrementa el nivel de abstracción de los elementos reutilizables. Por otra parte, de acuerdo con [20], no se dispone de experiencias sistemáticas de reutilización de requisitos a pesar de que aquellos que se refieren a restricciones dentro de un dominio, o se refieren a estilos de presentación de la información o se relacionan con políticas de la organización son potencialmente reutilizables, y pueden representar más del 50% de los requisitos. En consecuencia, la investigación relacionada con requisitos en entornos de reutilización está suficientemente justificada. La selección, especialización e integración de esos elementos reutilizables (también llamados assets) eleva el potencial de la reutilización [10].

De los distintos tipos de requisitos del software nos interesan inicialmente los requisitos funcionales (requisitos de usuario) porque son el punto de inicio para una estrategia completa de articulación de soluciones a partir de elementos reutilizables [3]. Diferentes propuestas de reutilización relacionadas con requisitos funcionales emplean la figura de los escenarios como secuencias de interacción entre el usuario y el sistema representadas mediante estructuras particulares [21][30]. Nosotros hemos trabajado en la obtención automática de los escenarios a partir de un workflow modelado con redes Petri para determinar de forma rápida los requisitos funcionales del sistema y almacenarlos como assets de análisis. La base formal de las redes de Petri permite tratar analíticamente los modelos de requisitos. Los algoritmos para la generación automática de requisitos funcionales se han implementado en una herramienta de software [23].

En el presente artículo continuamos nuestro trabajo planteando una aproximación para generalizar y comparar assets de requisitos asociados a assets de diseño y de implementación dentro de estructuras complejas denominadas mecanos. El proceso para comparar los escenarios del problema con los escenarios genéricos se realiza mediante analogía basada en redes de Petri. La propuesta se enmarca dentro de una estrategia global de reutilización del software desde los requisitos funcionales.

El resto del artículo se organiza de la siguiente manera: La sección 2 describe el proceso para reutilizar a partir de los requisitos funcionales. La sección 3 desarrolla los temas de generalización y comparación de requisitos. La sección 4 hace un recorrido por los trabajos más prominentes en reutilización de requisitos. La sección 5 concluye el artículo y menciona las acciones de trabajo inmediato.

2. Modelo de Proceso para Reutilización a partir de Requisitos Funcionales.

Según el criterio de Prieto-Díaz [26], la reutilización implica cuatro acciones básicas: Codificación de la información de desarrollo, almacenamiento de esta representación para referencia posterior, comparación de situaciones nuevas y antiguas, y adaptación para soportar los nuevos requisitos. Las actividades de comparación y adaptación claramente se relacionan de forma directa con los requisitos funcionales. Rada [27] sostiene que la funcionalidad es el criterio más importante para decidir si un elemento reutilizable se puede emplear en un contexto dado. Se puede interpretar entonces que los requisitos funcionales son el hilo conductor de la reutilización. Por lo anterior, la reutilización del software se apoya en formas estandarizadas de *representación*, de *comparación*, y de *adaptación* de requisitos funcionales.

2.1. Reutilización de Requisitos en los Mecanos.

Representar o documentar los requisitos es una actividad intrínseca a las metodologías de desarrollo. No obstante, Kotonya y Sommerville [20] señalan que los métodos existentes de desarrollo de software, aunque indican cómo producir una especificación de requisitos funcionales, carecen de sintaxis y semántica claras. Por otra parte, según los mismos autores, las aproximaciones formales proporcionan una sintaxis clara y una semántica definida para la representación de requisitos de usuario, pero no hablan de las fases de elicitación, negociación y documentación.

Nuestra propuesta de reutilización a partir de los requisitos se basa en los escenarios como representación de la funcionalidad. Los escenarios se han empleado para apoyar las actividades de la ingeniería de requisitos. Un escenario es una secuencia de acciones que muestran la evolución de un estado a otro [21].

En la figura 1 se muestra el modelo de proceso para la reutilización a partir de los requisitos funcionales generados de forma automática a partir del modelo del workflow. Estos deben ser generalizados para acceder a un mayor nivel de abstracción. Los escenarios generalizados pasan a la fase de comparación con los escenarios recuperados del repositorio. A partir de esta comparación se debe decidir si los escenarios recuperados (modelos genéricos) son adaptables al problema para entonces proceder a generar los mecanos correspondientes en tiempo de reutilización. Estos son mecanos instanciados (estructuras complejas formadas por elementos de análisis, diseño e implementación) que serán la base para el desarrollo de la aplicación final. La adaptación puede dar lugar a nuevas versiones que retroalimentan al repositorio.

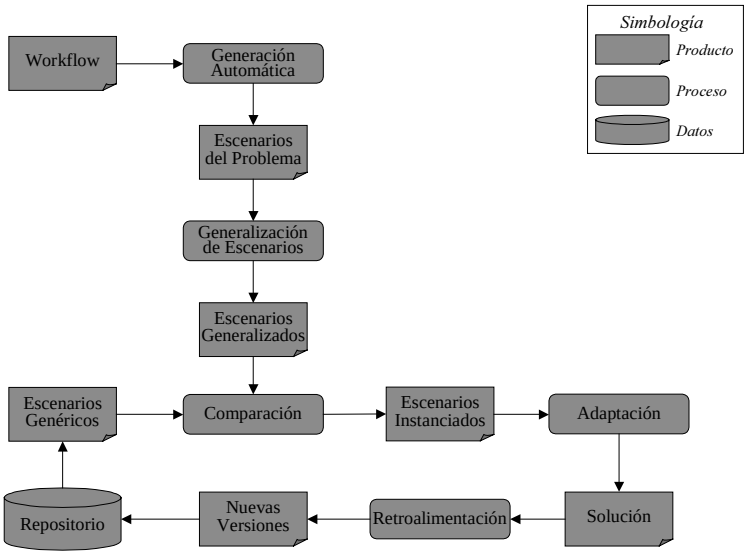


Fig. 1. Modelo de proceso para reutilización de software a partir de requisitos de usuario

Las tareas básicas para el proceso de reutilización de requisitos genéricos son cuatro:

- Representar los escenarios del problema
- Generalizar los escenarios del problema
- Comparar los escenarios generalizados y los escenarios genéricos
- Adaptación de los elementos para que satisfagan los requisitos

El tema de la representación ha sido tratado en nuestro trabajo anterior [23]. La generalización y la comparación constituyen el tema central del presente artículo. La adaptación de los elementos reutilizables relacionados será tema de nuestro trabajo futuro. Debemos aclarar que el término generalizar, en este artículo, se refiere a encontrar un modelo más genérico, y no un supertipo.

2.2. Representación de Requisitos Funcionales en los Mecanos.

Los requisitos funcionales como guías del proceso de reutilización deben estar documentados de forma tal que sea posible el uso efectivo de la información, sin ambigüedad y en estrecha relación con aspectos de reingeniería del negocio y de implementación del sistema [20][15]. Por eso hemos combinado redes de Petri y casos de uso en la representación de requisitos del usuario. Las redes de Petri ofrecen como ventaja el soporte

de herramientas universales, independientes y disponibles. Además, su base formal nos ha permitido la generación de escenarios de manera automatizada y rápida. El empleo de los casos de uso nos proporcionan un punto de partida ampliamente aceptado para el proceso de la ingeniería de requisitos.

Los casos de uso [16][7] permiten recopilar los requisitos del sistema e incluyen aspectos cognitivos, sociales y técnicos referentes al dominio de interés. El modelo de casos de uso permite distinguir entre lo que existe fuera del sistema (actores) y lo que debiera ser realizado por el sistema (casos de uso). Los actores son básicamente los usuarios del sistema que se clasifican en distintas categorías. Un actor, como entidad externa (persona u otro sistema), interactúa con el sistema para alcanzar alguna meta.

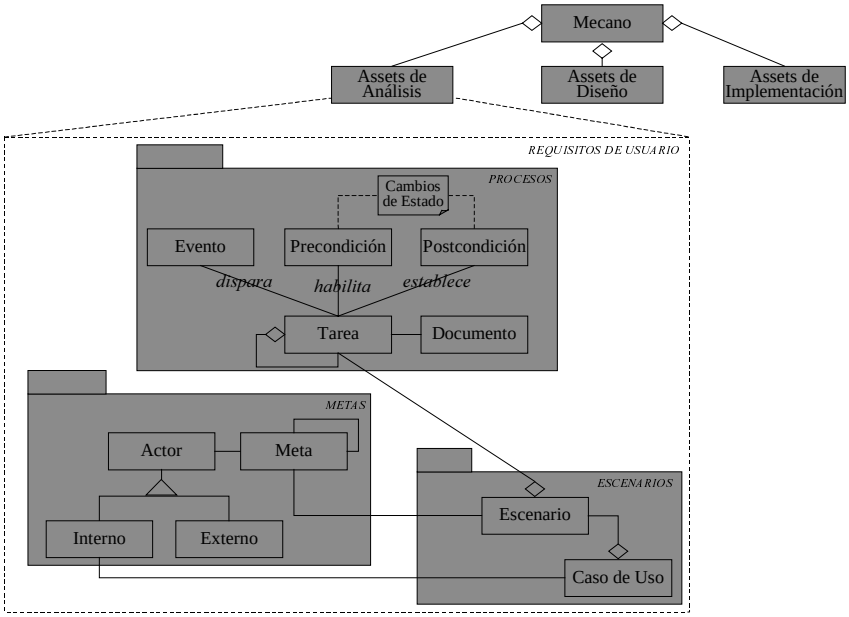


Fig. 2. Esquema en UML para los requisitos funcionales dentro de la estructura del mecano

En la figura 2 se muestra un fragmento del esquema que soporta nuestra propuesta de reutilización de los requisitos. La base de la propuesta es el modelo de mecano como elemento reutilizable complejo que es descrito en [11]. Un mecano es una agregación de componentes reutilizables de menor granularidad en forma de assets (de análisis, de requisitos y de implementación) con relaciones internivel e intranivel, y con la restricción de que debe existir al menos una relación internivel. Tanto el mecano como sus componentes son elementos susceptibles de ser reutilizados. Por lo anterior, es factible incorporar la reutilización desde la fase de los requisitos en el modelo de mecano.

Dentro del mecano, los requisitos de usuario se modelan mediante una estructura de *escenarios, metas y procesos* (figura 2). Los escenarios se recogen en casos de uso que recopilan las posibles interacciones actor-sistema. Una meta es un objetivo significativo y medible desde la óptica de un actor. Las metas se asocian con otras metas y con los escenarios; las asociaciones entre metas definen una jerarquía de metas. Los procesos de la organización son abordados bajo la dinámica de un modelo de comunicación que define el comportamiento del sistema a través de tareas que intercambian documentos (mensajes). Una tarea puede ser una agregación de tareas y se relaciona con un evento disparador, con una precondición y una postcondición, con lo cual las tareas son los elementos que definen los cambios de estado del sistema.

El esquema de requisitos refleja el rol central del *escenario como una secuencia de tareas para el logro de metas*. Tanto las tareas como las metas se representan como transiciones en el modelo de red de Petri, mientras que los documentos se corresponden con los lugares de la red.

En resumen, una agregación de escenarios forma un caso de uso. Un escenario se asocia a una meta y describe una secuencia de interacción entre un actor y el sistema. Con todo lo anterior, el modelado de requisitos mediante escenarios incluye acciones (modeladas como transiciones), reglas (modeladas como precondiciones, postcondiciones y eventos de disparo), metas (modeladas como transiciones puesto que para lograr una meta se requiere la ejecución de tareas del sistema) y actores (asociados a las metas).

3. Generalización y Comparación de Requisitos.

En esta sección proponemos la forma de abordar las labores de generalización y comparación de requisitos funcionales. El modelo del problema debe ser simplificado para recoger la funcionalidad esencial. Por su parte, el modelo genérico del dominio se debe recuperar del repositorio. El modelo simplificado del problema se compara con el modelo genérico del dominio mediante analogía.

3.1. Generalización de requisitos: Análisis por Reducción de Redes de Petri.

El análisis por reducción de redes de Petri pretende simplificar, no minimizar, la complejidad del modelo de forma tal que se preserven las propiedades del comportamiento [29]. Las técnicas de reducción permiten eliminar o sustituir transiciones y/o lugares con base en propiedades específicas, como vivacidad o limitación. El análisis por reducción se realiza mediante los siguientes pasos sucesivos: *eliminar lugares implícitos, realizar fusión de lugares equivalentes*, y *sustituir lugares por grupos de acciones.*

Desde el punto de vista conceptual, la reducción permite agrupar secuencias de operaciones en el sistema y sustituirlas por macroacciones en un modelo reducido. Para garantizar que la simplificación no altera el sentido funcional del modelo, se debe mantener un compromiso entre la completitud y la utilidad del modelo en cuanto a la satisfacción de las metas de los actores. Por eso, el proceso de reducción no debe *ni sustituir ni eliminar metas*.

La eliminación de lugares implícitos es una forma de simplificación que elimina las redundancias de tipo estructural. Intuitivamente, se dice que un lugar implícito es aquel cuyo marcado se puede evaluar en términos del marcado de otros lugares y además, no es el único lugar que impide la sensibilización de sus transiciones de salida. Por lo anterior, la eliminación de los lugares implícitos no altera la relación evento-acción del modelo. Para que un lugar esté implícito se requiere que sus transiciones de entrada tengan más de un lugar de salida y sus transiciones de salida tengan más de un lugar de entrada. Al eliminar un lugar implícito se procede a asignar sus salidas a los lugares implicantes.

Dos lugares p_i y p_j son equivalentes si coinciden los conjuntos de acciones asociados a ellos, y además, para cada transición de salida de p_i existe otra de p_j que tiene el mismo lugar de salida y a ambas se les asocia el mismo par evento - condición externa. Intuitivamente, los lugares equivalentes son aquellos cuyo efecto en el marcado de la red se considera equivalente.

Con la sustitución de lugares se persigue agrupar secuencias y sustituirlas por macroacciones. De esta manera se eliminan lugares que no están implícitos, pero preservando la vivacidad y la limitación. Se elimina el lugar pero preservando de manera condensada todas las secuencias de disparo de la red de Petri original. Un lugar p, más el conjunto de sus transiciones de entrada y salida, se sustituye por un conjunto de transiciones. Para que se mantengan todas las secuencias originales, se deberá introducir transiciones nuevas que reflejen las secuencias, o conjuntos de secuencias, de disparo originales. Estas secuencias se forman exclusivamente por transiciones de entrada y de salida del lugar que se sustituye.

Claramente, la sustitución de lugares es un proceso ascendente. La idea de fondo es que si se logra reducir la red de Petri, pero respetando las características de satisfacción de las metas de los actores, se llegue a un modelo más simple, generalizado, y comparable con la forma genérica del dominio.

3.2. Analogía de escenarios.

La búsqueda de analogías entre escenarios se ha utilizado para comparar los requisitos del software [21][30]. El concepto de analogía proviene de teorías cognitivas y designa al razonamiento mediante el cual se reconocen similitudes entre sistemas individuales y un modelo abstracto común [31]. El proceso de la analogía aplicado a la reutilización de requisitos consiste en *recuperar un caso base* con potencial de reutilización, *derivar características* aplicables al caso destino, *aplicar características* del caso base al caso destino, y *retroalimentar al repositorio* [32].

La analogía presenta como inconveniente, según Lam [21], la necesidad de desarrollar abstracciones para capturar elementos esenciales del problema, lo que es particularmente difícil en dominios industriales complejos. No obstante Sutcliffe y Maiden [31], han identificado una serie de dominios básicos o genéricos.

Los escenarios, tienen el problema de ser instancias, es decir, ejemplos específicos de comportamiento, lo que significa que la reutilización basada en escenarios es difícil [30]. Sin embargo, creemos que, si se cuenta con modelos de redes de Petri para el dominio genérico y para la funcionalidad del problema, se puede establecer la analogía entre los escenarios del problema y los escenarios genéricos.

La comparación de escenarios se debe realizar con base en algún elemento que sea común y significativo entre los modelos. Las metas de los actores pueden ser ese elemento de comparación. De esta manera se puede obtener alguna conclusión sobre la analogía entre dos modelos en términos de la satisfacción de metas.

La comparación de los modelos basada en metas se puede realizar mediante un procedimiento automatizable. A partir del modelo generalizado del problema, la búsqueda de analogías entre escenarios puede realizarse con los siguientes pasos:

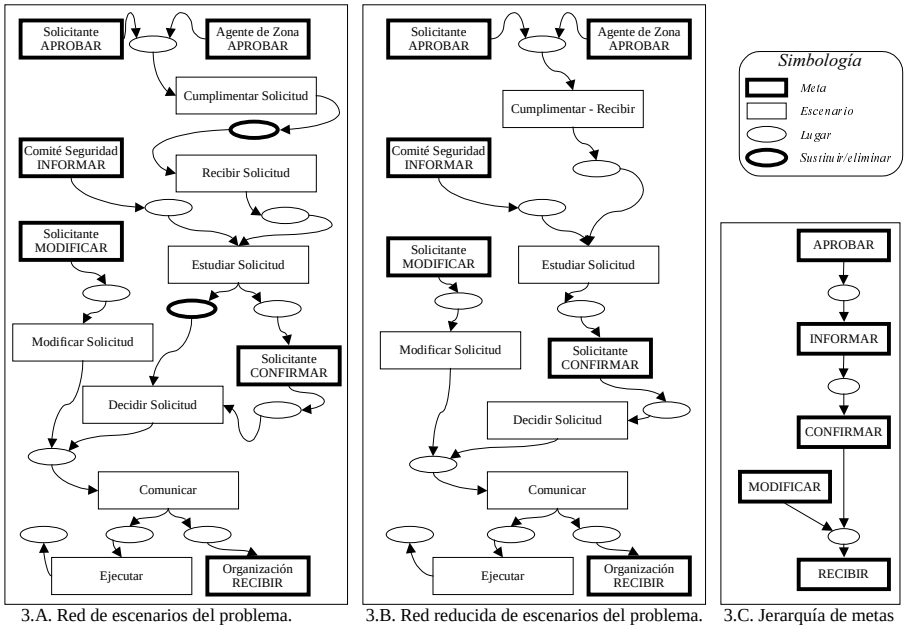
1. *Seleccionar el dominio.* El reutilizador debe identificar en qué dominio de desarrollo se halla. La recuperación del dominio se realiza interaccionando con el repositorio que devolverá los modelos disponibles para ese dominio.
2. *Seleccionar un modelo genérico.* Ese modelo genérico debe tener potencial básico de reutilización para el problema. Un modelo genérico es potencialmente reutilizable en un problema si se cumplen dos condiciones.
    a. Las metas raíz y las metas hoja de la jerarquía de metas del problema están incluidas (inclusión de nodos) en la jerarquía de metas del modelo genérico. Esto supone que las metas del dominio y del problema están expresadas mediante un lenguaje común.
    b. La estructura interna de las metas en el modelo genérico (la jerarquía de metas sin considerar los nodos raíz) satisface el logro de las metas hoja del problema. Esta condición asegura que en el dominio existe al menos un camino lógico para satisfacer las metas finales del problema.
3. *Seleccionar los escenarios genéricos.* Se debe seleccionar los escenarios del modelo genérico (secuencias de transiciones genéricas) que sean aplicables al problema. Un escenario genérico es aplicable al problema si:
    a. Parte de un nodo raíz del modelo genérico y satisface una meta hoja del modelo genérico.
    b. Satisface una meta raíz del problema y una meta hoja del problema.

Las secuencias de transiciones genéricas que satisfacen las metas raíz del problema y las metas hoja del problema se consideran escenarios análogos a los correspondientes del problema. Para probar esa analogía, se pueden expresar las secuencias de transiciones correspondientes a cada meta raíz, tanto en los escenarios análogos como en el modelo del problema, mediante disyunciones y conjunciones lógicas. Sin embargo, la aplicabilidad real sólo es decidible en términos de la semántica del problema. Esto indica que el proceso general de reutilización desde los requisitos debe pasar a la etapa de la adaptación de escenarios.

Por otra parte, la comparación entre el modelo genérico y el modelo simplificado se podría haber realizado en función de características sintácticas de redes de Petri, como alcanzabilidad, limitación, vivacidad, y equidad. Sin embargo la comparación basada en estas características sólo sería útil en términos de la sintaxis del modelo pero no diría nada en términos de la semántica del problema. La comparación sintáctica sólo permitiría saber si los modelos de dominio y de problema están bien formados (lo cual es indispensable). Pero

lo que se requiere para reutilizar a partir de los requisitos de usuario es comparar los modelos de manera que se pueda deducir si ambos ofrecen una funcionalidad análoga en términos de la semántica propia del problema. Por lo anterior, en este artículo la comparación de los modelos se ha centrado en la estructura de metas y en las transiciones entre las metas.

En la siguiente sección presentamos un ejemplo de comparación de modelos. La determinación del potencial de reutilización del modelo genérico para el problema se puede realizar mediante una prueba automática de teoremas, ver cuadro 2. La aplicabilidad de los escenarios genéricos en el problema es expresable en términos lógicos, aunque decidible en relación con la semántica del problema, ver cuadro 3.



3.A. Red de escenarios del problema. 3.B. Red reducida de escenarios del problema. 3.C. Jerarquía de metas

Fig. 3. Red de Petri de escenarios y metas para el proceso de órdenes de una organización.

3.3. Caso Práctico.

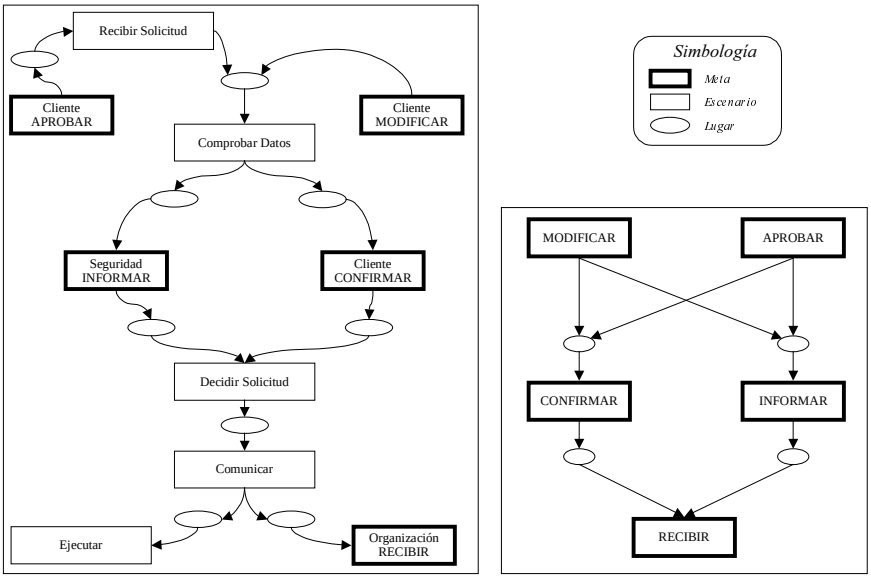
En la figura 3.A se muestra un modelo de los escenarios generados de un caso real [23]. La versión simplificada del modelo del problema se aprecia en la figura 3.B. La figura 3.C muestra la jerarquía de metas del modelo del problema. Nótese que la estructura jerárquica de metas, la meta aprobar sólo se toma en cuenta una vez, aunque existen dos actores (Solicitante y Agente de Zona) que presentan esa misma meta.

Las metas se incorporan como transiciones de la red de Petri (con recuadro más grueso y en su inscripción se antepone el nombre del actor) por dos razones. Primero, por la necesidad de incorporar un elemento esencial para la comparación de los modelos. Segundo, porque las metas representan acción y cohesión dentro del sistema en tanto que la satisfacción de una meta lleva a la ejecución de tareas y eventualmente a la satisfacción de otras metas. Por las mismas razones se abstrae del modelo su jerarquía de metas. Esta jerarquía expresa la relación de precedencia entre las metas.

La figura 4 presenta un posible modelo genérico del dominio de proceso de solicitudes, identificado por Sutcliffe y Maiden [31], en forma de red de Petri. Este es un modelo minimal pues contiene las acciones esenciales dentro del dominio. Las acciones y las metas corresponden a la interacción entre actores y un sistema de proceso de solicitudes (como un cajero automático o un proceso de admisión, entre otros). La dinámica de comunicación entre los actores y el modelo genérico define una jerarquía de metas del dominio. Al igual que en el modelo del problema, la jerarquía establece una relación entre las metas y determina cuales son las principales y cuales las subordinadas. Esta relación entre las metas es información propia del dominio.

En el dominio de proceso de solicitudes existen dos metas principales: *aprobar* y *modificar*. El actor entra en contacto con el modelo porque pretende *aprobar o modificar* una solicitud. Junto a cada una de esas metas existen otras subordinadas que son de apoyo para la satisfacción de las primeras.

Dados los modelos de las figuras 3.B y 4.A, se debe probar que el modelo genérico (4.A) es potencialmente reutilizable en el problema (3.B). Para ello se debe probar que las metas raíz y las metas hoja de la jerarquía del problema están incluidas en la jerarquía genérica. Esto se puede determinar fácilmente a partir de los datos del cuadro 1, además es una decisión automatizable en forma de verificación de inclusión de conjuntos. No obstante, con esta decisión sólo sabremos si existe similitud básica de las metas de los modelos sin determinar si el modelo genérico ofrece soporte básico a la estructura de metas del problema.



4.A. Red de Petri del dominio genérico de proceso de solicitudes

4.B. Jerarquía de metas del dominio de proceso de solicitudes

Fig. 4. Un modelo genérico para el dominio de proceso de solicitudes.

El siguiente paso es probar que la estructura de metas genéricas satisface el logro de la metas hoja del problema. Esto se demuestra por refutación a partir de las proposiciones lógicas de metas internas del modelo genérico y las metas raíz del modelo del problema, ver cuadro 2.

Del cuadro 2 se deduce que la negación de la meta final del modelo del problema (teorema a probar), en el sistema lógico formado por la estructura de metas internas del modelo genérico y las metas raíz del problema, conduce a contradicción. Con lo anterior, se deduce que la parte superior y la parte inferior (entradas y las salidas) del modelo del problema se satisfacen en el modelo genérico. Se puede concluir que el modelo genérico es potencialmente reutilizable en el problema puesto que la misma meta del problema es derivable de la estructura genérica. Y así se muestra que la analogía de los modelos se puede comprobar con fundamento en la lógica proposicional.

Cuadro 1. Metas raíz, metas hoja y estructura lógica de metas internas del modelo genérico y del modelo del problema.

	MODELO GENERICO	**MODELO DEL PROBLEMA**
Metas raíz	Modificar, Aprobar	Modificar, Aprobar
Metas hoja	Recibir	Recibir
Estructura lógica interna	Confirmar ← Modificar ∨ Aprobar Informar ← Modificar ∨ Aprobar Recibir ← Confirmar ∧ Informar	Informar ← Aprobar Confirmar ← Informar Recibir ← Confirmar ∨ Modificar

Los escenarios genéricos aplicables a los correspondientes escenarios del problema, según las metas raíz del problema (Aprobar y Modificar), se muestran en el cuadro 3. Para probar que las secuencias de transiciones de los escenarios análogos son aplicables al problema se puede ver a las secuencias de transiciones como proposiciones lógicas.

Cuadro 2. Prueba por refutación de que la meta hoja del modelo del problema se satisface con la estructura lógica interna del modelo genérico y las entradas del problema.

	MODELO GENERICO		
Cláusulas	---------- Estructura lógica interna del modelo genérico 1. Confirmar ∨ ¬Modificar 2. Confirmar ∨ ¬Aprobar 3. Informar ∨ ¬Modificar 4. Informar ∨ ¬Aprobar 5. Recibir ∨ ¬Confirmar ∨ ¬Informar ---------- Metas raíz del modelo del problema 6. Aprobar 7. Modificar		
Teorema a probar	¬Recibir		
Prueba por refutación	Recibir ∨ ¬Confirmar ∨ ¬Informar ¬Confirmar ∨ ¬Informar ¬Informar ∨ ¬Aprobar ¬Informar ¬Modificar *CONTRADICCIÓN*	¬Recibir Confirmar ∨ ¬Aprobar Aprobar Informar ∨ ¬Modificar Modificar	cláusula 5 cláusula 2 cláusula 6 cláusula 3 cláusula 7

En términos lógicos, la aplicabilidad de los escenarios análogos se puede establecer si se comprueba que *Escenario-Genérico-Aprobar*(Recibir-Solicitud ∧ Comprobar-Datos ∧ Decidir-Solicitud ∧ Comunicar ∧ Ejecutar) es análogo a *Escenario-Problema-Aprobar*(Cumplimentar-Recibir ∧ Estudiar-Solicitud ∧ Decidir-Solicitud ∧ Comunicar ∧ Ejecutar). En otras palabras, se debe comprobar que las secuencias de satisfacción de la meta Aprobar del dominio genérico (*Escenario-Genérico-Aprobar)* es análoga a la meta Aprobar en el problema (*Escenario-Problema-Aprobar)*. Lo mismo se hace con la meta Modificar. De este modo, las secuencias genéricas son aplicables al problema si, y sólo si:

- *Escenario-Genérico-Aprobar*(Recibir-Solicitud ∧ Comprobar-Datos ∧ Decidir-Solicitud ∧ Comunicar ∧ Ejecutar) es aplicable a *Escenario-Problema-Aprobar*(Cumplimentar-Recibir ∧ Estudiar-Solicitud ∧ Decidir-Solicitud ∧ Comunicar ∧ Ejecutar)
- *Escenario-Genérico-Modificar*(Comprobar Datos ∧ Decidir Solicitud ∧ Comunicar ∧ Ejecutar) es aplicable a *Escenario-Problema-Modificar*(Modificación ∧ Comunicar ∧ Ejecutar)

Cuadro 3. Secuencias de transiciones de los escenarios genéricos para las metas raíz del problema (Aprobar, Modificar), y los correspondientes escenarios del problema.

META	ESCENARIO GENERICO	ESCENARIO DEL PROBLEMA
Aprobar	(Recibir Solicitud ∧ Comprobar Datos ∧ Decidir Solicitud ∧ Comunicar ∧ Ejecutar)	(Cumplimentar-Recibir ∧ Estudiar Solicitud ∧ Decidir Solicitud ∧ Comunicar ∧ Ejecutar)
Modificar	(Comprobar Datos ∧ Decidir Solicitud ∧ Comunicar ∧ Ejecutar)	(Modificación ∧ Comunicar ∧ Ejecutar)

4. Trabajos Relacionados.

Existen varias alternativas para iniciar una solución al problema de la reutilización de los requisitos. Mientras algunos trabajos se preocupan por establecer una semántica formal para la expresión de los requisitos, otros intentan proporcionar técnicas de análisis y de comparación de requisitos.

4.1. Los Escenarios y los Casos de Uso.

Los escenarios se utilizan para apoyar actividades de ingeniería de requisitos: descripción de propiedades del dominio de aplicación [4][14], descubrimiento de requisitos [16][19], evaluación de alternativas de diseño de alto nivel [13], y validación de requisitos [25]. Más aún, existen aproximaciones basadas en escenarios [9].

Los casos de uso propuestos por Jacobson [16] y reforzados por Cockburn [7], se han convertido en la estrategia principal de descubrimiento de requisitos en la comunidad de desarrolladores [6]. La idea de los casos de uso es capturar los escenarios de interacción en un sistema. Esas transacciones producen un valor medible para un actor del sistema. La idea subyacente es que mediante los casos de uso es posible especificar la funcionalidad completa del sistema. Sin embargo, los casos de uso presentan la desventaja de ser informales lo que conduce a malas interpretaciones y dificulta la tarea de garantizar que la funcionalidad deseada es satisfecha por el sistema.

Se ha tratado de corregir la informalidad de los casos de uso con el objetivo de expresar con más precisión los requisitos. Los diferentes trabajos varían entre los que intentan desarrollar una semántica formal para los casos de uso, como en [1][12][14], hasta los trabajos que se encaminan hacia el desarrollo de técnicas para integrar y analizar conjuntos de casos de uso como en [22][2].

El aporte de todos estos intentos por tratar de formalizar y de analizar los escenarios y los casos de uso es limitado para la reutilización de requisitos. La limitación principal se basa en que tales intentos toman como punto de partida a los escenarios y casos de uso ya descubiertos.

4.2. Escenarios de Dominio.

Maiden y Sutcliffe [24], proponen una estrategia basada en analogía para la reutilización de requisitos. Los mismos autores hacen una propuesta de comparación a través de un proceso de matching basado en lógica de primer orden y lógica temporal como se expone en [31].

Por otra parte, en el seno del proyecto STARS [8] se propuso el análisis de dominios como un proceso mediante el cual la información necesaria para desarrollar sistemas software se identifica, se captura y se clasifica de modo que sea reutilizable para nuevos desarrollos. El enfoque del análisis de dominios se centra en realizar un análisis de requisitos para aplicaciones típicas del dominio.

Lam [21] se basa en los trabajos de analogías y de análisis de dominios para desarrollar escenarios de dominio como una forma particular de abstracción del problema mediante compartimientos estancos. Plantea que la reutilización de escenarios es útil en aplicaciones que pertenecen a un mismo dominio. Por ejemplo, los escenarios de un sistema de bibliotecas (préstamo de libros, retornar libros, etc) son comparables mediante analogía con los encontrados en un sistema de alquiler de vídeos (préstamo de vídeos, retornar vídeos, etc). Ambas aplicaciones pertenecen al dominio de gestión de recursos y comparten escenarios similares. La adaptación de los escenarios se realiza mediante “walkthrouhs” y discusiones con los clientes.

El trabajo de Lam [21] muestra que la reutilización de escenarios es factible. Y dado que los escenarios permiten capturar conocimiento del dominio, la reutilización de escenarios permite la reutilización de conocimiento del dominio. En el mismo sentido, Tessem *et alt*. [32] y Sutcliffe *et alt.* [30] plantean el soporte a la reutilización de requisitos con base en escenarios que son comparados mediante analogía.

4.3. Escenarios Expresados Mediante Redes de Petri.

De los trabajos mencionados se deduce que efectivamente se requiere de una sintaxis consistente con una semántica precisa y el soporte de herramientas para representar y analizar los escenarios del dominio y del problema en la reutilización de los requisitos. No obstante, es cuestionable que la representación de escenarios se haya basado en estructuras particulares; representaciones tabulares y diagramas de transición de escenarios en el caso de Lam [21], y modelos de sistemas de objetos (OSMs por sus siglas en inglés) en el caso de

Sutcliffe [30]. Nosotros hemos propuesto [23] la normalización de casos de uso basada en redes de Petri como paso inicial para una aproximación de reutilización a partir de los requisitos. El empleo de redes de Petri para expresar casos de uso se ha propuesto en [22], aunque sólo con la intención de formalizar los casos de uso ya descubiertos sin pretender la reutilización de los mismos.

5. Conclusiones y Trabajo Inmediato.

En este artículo se ha presentado una aproximación para generalizar y comparar escenarios dentro del proceso de reutilización de requisitos. La propuesta se enmarca dentro del modelo de reutilización del software soportado por mecanos, que se ha desarrollado en el Grupo GIRO de la Universidad de Valladolid. El procedimiento de generalización y comparación de escenarios propuesto se fundamenta en casos de uso, redes de Petri y la búsqueda de analogía. De esta manera se complementa el trabajo de investigación en reutilización del software que se ha desarrollado.

El trabajo aquí presentado es una continuación de lo planteado en [23] para la obtención automatizada de requisitos funcionales como escenarios a partir de un workflow. La determinación rápida de requisitos funcionales y la comparación de los mismos con modelos de dominio integrados en mecanos son la base de la estrategia de reutilización de requisitos que estamos integrando en un entorno de desarrollo.

El modelado formal de los requisitos con redes de Petri nos ha permitido establecer una comparación mediante analogía entre el modelo del problema y del dominio genérico. El modelo del dominio es minimal y recoge las acciones esenciales y las metas dentro de un dominio particular. El modelo del problema es sometido a un proceso de generalización, basado en el análisis de reducción de redes de Petri, para simplificarlo de manera que represente la funcionalidad esencial requerida por el usuario.

El proceso de comparación toma como punto de partida las metas del sistema modelado. Esto plantea la debilidad de requerir de una sintaxis común para expresar las metas del problema y del dominio. No obstante, nuestro planteamiento presenta la ventaja de ser automatizable. A diferencia de otras aproximaciones de comparación de requisitos que utilizan procesos complejos e interactivos de matching, con heurísticas incorporadas, nosotros proponemos un proceso automático dentro del modelo de mecano.

El ejemplo estudiado ayuda a validar la utilidad práctica de la generación automática de escenarios y de la analogía basada en modelos de redes de Petri. Además, se demuestra que es factible la combinación de escenarios, redes de Petri y analogía como soporte a la reutilización a partir de los requisitos funcionales. No obstante, el trabajo debe continuar hasta integrar una aproximación para adaptar los escenarios genéricos en el desarrollo de soluciones software dentro del marco del mecano.

Nuestro trabajo inmediato consiste en integrar esta propuesta en las herramientas de generación y reutilización de requisitos sobre las que estamos trabajando, y desarrollar una estrategia de adaptación de escenarios genéricos en el desarrollo de soluciones software.

6. Agradecimientos.

Este documento se ha generado con el aporte invaluable y la crítica constructiva de los miembros del Grupo GIRO del Departamento de Informática de la Universidad de Valladolid. Este trabajo ha sido parcialmente financiado por CICYT(TIC2000-1673-c06-05) como parte del proyecto DOLMEN. Oscar López también agradece el apoyo de la Agencia Española de Cooperación Internacional (AECI), el Instituto Tecnológico de Costa Rica (ITCR) y el Ministerio de Ciencia y Tecnología de Costa Rica (MICYT).

7. Referencias.

[1] Andersson, M.; Bergstand, J.; Formalizing Use Cases with Message Sequence Charts. MASTER´S TESIS. Lund Institute of Technology, 1995.

[2] Back, R.; Petre, L.; Porres, I.; Formalising UML Use Cases in the Refinement Calculus. TUCS Technical Report No 279. Turku Centre for Computer Sceince. May 1999. ISBN 952-12-0464-8

[3] Bass, L., et. alt. Third Product Line Practice Workshop Report. Carnegie Mellon Software Engineering Institute. CMU/SEI-99-TR-003. March 1999.

[4] Beck, K.; Cunningham, W.; A laboratory for teaching object-oriented thinking. In Proceedings of OOPSLA'89, Special Issue of ACM SIGPLAN Notices, 24(10):1-6. 1989.

[5] Biggerstaff, T.J. An assessment and analysis of software reuse. Advances in Computers, (34). Academic Press, 1992 Edited by Marshall C. Yovits

[6] Booch, G., Rumbaugh, J., Jacobson, I., "The Unified Modelling Language", Adisson Wesley, 1999.

[7] Cockburn, A. Writing Effective Use Cases. Humans and Technology for Addison-Wesley, Q3 2000. 1999.

[8] Creps, R.; Simos, M.; Prieto-Díaz, R. The STARS Conceptual Framework for Reuse Processes. Defense Advanced Research Projects Agency. November 1992.

[9] Dardenne, A.; Fickas, S.; Lamsweerde, V.; Goal-directed concept acquisition in requirements elicitation. CIS-TR-91-08, Department of computer and information science, University of Oregon, March, 1991.

[10] García, F.J. et alt.; Mecanos: Exposición de resultados y líneas abiertas en la reutilización sistemática del software. Actas Jornadas Iberoamericanas Ingeniería de Requisitos y Ambientes de Software. 1999. pp 193-204.

[11] García, F.J., Modelo de Reutilización Soportado por Estructuras Complejas de Reutilización Denominadas Mecanos. TESIS DOCTORAL, Universidad de Salamanca, 2000.

[12] Glinz, M.; An integrated Formal Model of Scenarios Based on Statecharts. Proc. European Software Eng. Conf.'95, pp. 254-271, Spain, Sept. 1995.

[13] Holbrook, H.; A scenario-based methodology for conducting requirements elicitation. In ACM Sigsoft Software Engineering Notes. 15(1):95-104, 1990.

[14] Hsai, P.; Jayarajn, S.; Gao, S.; Kung, D.; Yasufumi, T.; Chen, C.; Formal approach to scenario analysis. In IEEE Software, March 11(2):33-41, 1994.

[15] IEEE, "Recommended Practice for Software Requirements Specifications", Std 830-1993

[16] Jacobson, I et alt.; Object-Oriented Software Engineering: A Use-Case Driven Approach. Addison-Wesley 1992

[17] Jacobson, Griss, Jonsson; Software Reuse: Architecture, Process, Organization for Business Success.ACM,1997

[18] Karlsson, E. Software Reuse: A Holistic Approach. John Wiley & Sons. 1995 – ISBN 0-41-95819-0

[19] Kavakli, E.; Loucopoulus, P.; Filippidou, D.; Using scenarios to systematically support goal-directed elaboration for information system requirements. In Proc.of Internat. IEEE ECBS'96. Germany, March, 1996

[20] Kotonya, G.; Sommerville, I. Requeriments Engineering: Processes and Techniques. USA, Willey. 1997.

[21] Lam W.; Scenario Reuse: A Technique for Complementing Scenario-Based Requirements Engineering Approaches. IEEE 1997; 0-8186-8271-X/97. 1997.

[22] Lee, W.; Cha, S.; Kwon, Y.; Integration and Analysis of Use Cases Using Modular Petri Nets in Requirements Engineering, IEEE Trans. On Software Eng. (24):12, pp. 1115-1130. Dec. 1998.

[23] López, O; Laguna, M.A.; Marqués, J.M.; Generación Automática de Casos de Uso para Desarrollo de Software con Reutilización. Actas V Jornadas Ingeniería de Software, Valladolid, 2000. ISBN 84-8448-065-8. pp 89-101.

[24] Maiden,N.;Sutcliffe,A.;Exploting reusable specification through analogy. In Commun.ACM. 35(4):55-64. 1993

[25] Potts,C.; Takahashi,K.; Anton A.;Inquiry-based Requirements Analysis. In IEEE Softw.; Mar,11(2):21-32,1994

[26] Prieto-Díaz, R.; Classification of Reusable Modules. In Software Reusability, Vol 1. Concepts and Models. Edited by T. Biggerstaff and Alan J. Perlis. ACM Press, Frontier Series. 1989. pp 99-123.

[27] Rada, R. Reengineering Software, How to Reuse Programming to Build New, State-of-art Software. 2º Edition. USA AMACON. 1999.

[28] Sampat, N.; Luker, P.; Zedan, H. A Model for Software Reuse. Software Technology Research Laboratory, School of Computing Sciences, De Montfort University.

[29] Silva, M., Las Redes de Petri en la Informática y la Automática. Editorial AC. Madrid, 1985.

[30] Sutcliffe, A.; Maiden, N.; Manuel, D.; Supporting Scenario-Based Requirements Engineering In IEEE Transactions on Software Engineering. (24):12. December,1998.

[31] Sutcliffe, A.; Maiden, N.; The Domain Theory for Requirements Engineering. In IEEE Transactions on Software Engineering. (24):3. March,1998.

[32] Tessem, B.; et alt.; ROSA=Reuse of Object-oriented Specifications through Analogy: A Project Framework. Report No. 16. Depart. of Information Science, Unversity of Bergen, Norway. ISSN 08803-6489. March 1994.