Departamento de Informática
Universidad de Valladolid
Valladolid - España

Transformación de requisitos

Epifanio Ecube, Miguel A. Laguna, Oscar López

Departamento de Informática, Universidad de Valladolid
epifecu@lab.fi.uva.es, {mlaguna, olopez}@infor.uva.es

Resumen La diversidad de formas de expresar requisitos es un freno para la reutilización de los mismos. Para resolver este problema se utiliza un modelo común de requisitos que permite la transformación de unos tipos de requisitos en otros. Para ello se utiliza una representación intermedia, basada en redes de Petri que aporta rigor a los modelos y permite la comprobación de su consistencia. Se definen una serie de algoritmos de transformación desde workflows y diagramas de actividades hasta diagramas de casos de uso, flujos de datos o escenarios. Estos algoritmos se han implementado como parte de una herramienta de gestión y reutilización de requisitos.

Informe Técnico DI-2004-1

1 Introducción

La Reutilización del software es una alternativa tecnológica para lograr el incremento de la productividad. La reutilización de los productos iniciales del proceso de desarrollo del software estimula la reutilización a lo largo del resto del ciclo de vida y permite aprovechar mejor el esfuerzo de desarrollo.

La ingeniería de dominios permite capturar, organizar y representar la información útil para el desarrollo de sistemas software de manera que esta pueda ser reutilizada para crear nuevos sistemas software. Para que la reutilización esté presente a lo largo de todo el ciclo de vida del software, se requieren herramientas que faciliten y fomenten tanto la reutilización, como el diseño para la reutilización en las distintas etapas que componen el desarrollo de un producto.

El completar la especificación de requisitos de una nueva aplicación con elementos reutilizables calificados, mejora la calidad y agiliza el proceso de ingeniería de requisitos, a la vez que propicia el ahorro de recursos en el contexto de la producción empresarial del software. Dado que la ingeniería de requisitos desencadena el proceso del ciclo de vida, su reutilización puede provocar un impacto positivo en este proceso.

La reutilización desde las etapas de captura y análisis de requisitos de la aplicación, requiere de un marco adecuado con el soporte de estructuras y herramientas. El carecer de este marco equivale a poca efectividad en la estrategia de reutilización.

Para reutilizar requisitos, al igual que cualquier otro producto del ciclo de vida, es necesario representar, clasificar, almacenar, seleccionar y adaptar los assets. Estas labores requieren el soporte de estructuras para interrelacionar los requisitos de manera coherente y así reflejar la realidad del dominio en el repositorio. Las estructuras de reutilización, por ejemplo un mecano, pueden adoptar tal nivel de complejidad que se hace necesario el empleo de herramientas automatizadas para las labores de gestión de los elementos reutilizables, tanto en la fase de *ingeniería del dominio* como en la de *ingeniería de la aplicación.*

En el caso de los sistemas software, sus capacidades han ido en aumento en función del soporte a una mayor cantidad y complejidad de actividades de la organización. El aumento en la complejidad ha llevado a la necesidad de emplear diferentes técnicas de modelado de los requisitos para incorporar los puntos de vista de diferentes «stakeholders». La reutilización de requisitos debe tener en cuenta la complejidad de los mismos y la consecuente diversidad de técnicas de modelado de requisitos.

Aunque la reutilización de requisitos es una alternativa para obtener especificaciones software de mejor calidad y en menor tiempo que si se desarrollan desde cero, existen algunos obstáculos que superar para abordar con satisfacción el desarrollo con reutilización

Distintos niveles de descripción: Aparecen distintos niveles de descripción en el nivel de abstracción de los requisitos.

Diversidad de técnicas de modelado: Los requisitos actúan como medio de comunicación y negociación entre las diferentes personas involucradas en el proceso

de desarrollo del software. Por esta razón se utilizan diversas técnicas y formatos para mostrar distintas vistas de los sistemas.

Bajo nivel de formalidad: Las necesidades de comunicación y de acuerdo entre las partes involucradas impiden por lo general utilizar elevados niveles de formalidad en las etapas iniciales de la ingeniería de requisitos.

Especificidad de los requisitos: Los diagramas de requisitos responden a las necesidades especificas de una aplicación software. Esto plantea la necesidad de acciones especiales a fin de reutilizar requisitos en la especificación de nuevas aplicaciones.

Además, se requieren herramientas para automatizar el propio proceso de reutilización de artefactos software, pero si no se solventasen los obstáculos previamente expuestos, la complejidad de esa herramienta podría ser excesiva.

Para afrontar esos obstáculos se pueden realizar dos acciones generales. La primera acción consiste en restringir el ámbito de estudio a los requisitos que se representan mediante seis técnicas de modelado ampliamente conocidas como: escenarios, casos de uso, diagramas de actividades, flujos de datos, documentos-tareas y workflows. La segunda acción consiste en establecer un marco de trabajo para la reutilización de requisitos. Este marco ha de estar conformado por etapas claramente definidas, como por ejemplo:

- Describir los diagramas de requisitos mediante un modelo común.
- Realizar la proyección de los diagramas integrados por ese modelo común hacia un lenguaje formal para realizar un análisis de validez de los diagramas.
- Organizar los requisitos dentro de un repositorio.

La descripción de los diagramas mediante un modelo común se aborda traduciendo esos diagramas a un Metamodelo de Representaciones de Requisitos (que a partir de ahora y por comodidad denotaremos simplemente por metamodelo) que los unifique. La proyección hacia un lenguaje o modelo formal es en parte el objetivo de este trabajo. La etapa 3 se aborda mediante la organización los requisitos en un repositorio basado en un modelo de familia de diagramas. En el repositorio se catalogan los requisitos para facilitar su comprensión e identificación.

En el proceso de la ingeniería de requisitos, las necesidades de comunicación entre los «stakeholders» y de integración de los «viewpoints» plantean la necesidad de buscar una estrategia que permita que las diversas técnicas de modelado de requisitos de carácter semiformal, puedan ser integradas en un proceso de análisis, verificación y transformación. Distintas visiones han llevado a una explosión de técnicas de modelado que comparten múltiples características.

- Los ingenieros informáticos manejan, sobre todo, casos de uso y en general, escenarios para representar interacciones usuario-sistema.

- Los técnicos en organización de empresas utilizan herramientas para la reingeniería de procesos de negocio (BPR).

- Los gestores de flujos de trabajo se imponen en las grandes empresas como soportes del trabajo cooperativo y de seguimiento y automatización de procesos complejos.

- La *WfMC* propone un marco estándar para la definición de modelos de flujo de trabajo y su implantación [6].

- UML deriva un diagrama de estados especializado,el diagrama de actividades para, precisamente, representar los procesos de negocio 0.

La situación actual nos lleva a destacar dos niveles extremos: El nivel de negocio global, donde encajarían la BPR, los flujos de trabajo o los BUC, y el nivel de interacción usuario-computadora, donde se aplican los escenarios o casos de uso.

La hipótesis de partida este trabajo consiste en que todas estas técnicas de modelado son estructuralmente similares y se pueden integrar de forma homogénea en un proceso de análisis, de modo que, por ejemplo, una actividad dentro de un flujo de trabajo se refine como un workflow, de forma recursiva, hasta llegar a una actividad elemental que representa una interacción simple entre un actor y un sistema.

Por tanto, la propuesta del presente trabajo consiste en desarrollar a través de una definición precisa, una herramienta para la transformación o traducción automática de requisitos funcionales de un sistema de información, expresados mediante diagramas de un tipo, en requisitos expresados mediante otro tipo de diagrama y almacenarlos en forma de assets de análisis.

El disponer de una herramienta que posibilite la transformación de una técnica de modelado en otra, resulta imprescindible, y aporta un alto beneficio a la empresa, facilitando la comunicación entre los stakeholders y la integración de los sistemas de negocios con los que trabaja. Además permitirá encontrar en cada momento una representación apropiada para cada dominio de información y para cada nivel de descripción de los requisitos.

En este trabajo se continúa y se amplia el entorno de Reutilización de requisitos $\boldsymbol{R^2}$, añadiéndole nuevas funcionalidades como la posibilidad de transformar requisitos de un tipo en otro. El usuario, a partir de este momento, podrá transformar sus diagramas, previamente editados y almacenados en un repositorio, en otros tipos de diagramas. Esto redunda en una mejora de la comunicación entre los usuarios del sistema, y en la posibilidad de expresar los requisitos de un sistema de información con la representación que describe mejor el modelo de negocio que encapsula dicho sistema. La sección 2 presenta el entorno $\boldsymbol{R^2}$ básico, la sección 3 hace referencia al metamodelo de requisitos que subyace en la herramienta y la sección 4 incide en los formalismos que se utilizan. En las secciones 5 y 6 se presentan de forma detallada los mecanismos de transformación implementados como complemento a la herramienta $\boldsymbol{R^2}$. Finalmente, la sección 7 establece las conclusiones del trabajo y plantea posibles ampliaciones del mismo.

2 El Entorno R^2

En la figura 1 se propone una arquitectura para un entorno de reutilización de requisitos (R^2). Este entorno pretende facilitar las labores de reutilización de requisitos. El entorno esta formado por cinco elementos principales, con sus correspondientes subelementos: Interfaz de usuario, Editor de requisitos, Gestor de datos, Base de datos e Intercambio de datos. Posteriormente se han incorporarado otros elementos importantes como un traductor de diagramas, un analizador léxico, un analizador de consistencia y un gestor de repositorio. El transformador o traductor de diagramas es el objeto de este trabajo.

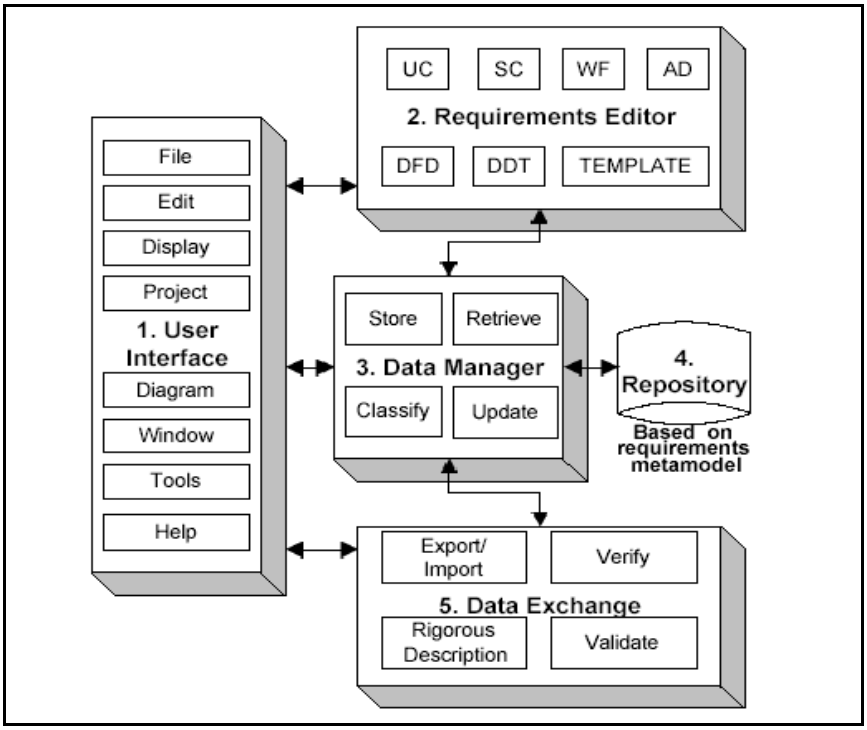


Figura 1. Arquitectura del entorno de reutilización de requisito

La diversidad de técnicas de modelado de requisitos y la existencia de diferentes niveles de descripción obstaculizan el establecimiento de una aproximación a la reutilización de requisitos. Para afrontar estos obstáculos se realizan varias acciones generales.

Una de esas acciones consiste en restringir el ámbito de estudio a los requisitos que se representan mediante seis técnicas de modelado ampliamente conocidas: escenarios, casos de uso, diagramas de actividades, flujos de datos, documentos-tareas y workflows.

Otra de las acciones consiste en integrar los diagramas en un modelo común o metamodelo. El metamodelado es un tipo de modelado que consiste en producir *metamodelos*. El modelado, en términos generales, consiste en representar elementos de la realidad mediante abstracciones. A dicho modelo del lenguaje de modelado se le llama «metamodelo». De esta manera, el metamodelo describe la información que puede ser expresada mediante estructuras del lenguaje que se emplea para el modelado. Si se eleva aún más el nivel de abstracción, el lenguaje del metamodelo puede a su vez ser modelado mediante un meta-metamodelo.

3 Metamodelo para Reutilización de Requisitos

Dado que los requisitos se representan mediante diversas técnicas dentro de diferentes paradigmas de modelado, se requiere un esquema conceptual para describir y gestionar esas representaciones en una estrategia de reutilización de requisitos. Cada técnica de modelado muestra una perspectiva diferente de los requisitos.

Para integrar esos diferentes diagramas dentro de una estrategia de reutilización de requisitos, se propone un Metamodelo de Diagramas de Requisitos, que en lo sucesivo llamaremos *metamodelo de requisitos*.

Los elementos centrales de este metamodelo son la *Representación de Requisitos*, utilizada para describir los diferentes diagramas de requisitos, la *Unidad de Modelado*, utilizada para describir las unidades contenidas en los diagramas de requisitos, y el *Objetivo del Dominio* que permite caracterizar a las Representaciones de Requisitos.

Los modelos de requisitos que representan el *Universo del Discurso* están formados por Unidades de Modelado. Las Unidades de Modelado incluidas en las Representaciones de Requisitos se clasifican en seis categorías [4]:

- **Actividad**: Representa una labor o proceso que se realiza dentro del dominio. Se distinguen dos tipos de actividades:
  - **Tarea**: Es una unidad formada por la agregación de otras actividades.
  - **Acción**: Representa una actividad atómica, es decir una actividad que no presenta subdivisiones.
- **Sujeto**: Representa a los actores que intervienen en el sistema y pueden ser personas, unidades organizacionales o sistemas autónomos.
- **Objetivo**: Representa las intenciones específicas de los usuarios y sistema en el contexto de la interacción entre ellos.

- **Limitación**: Información que restringe la funcionalidad del sistema y que puede ser de tipo temporal o de disponibilidad de algún recurso.

- **Estado**: Representa una situación dinámica (un espacio o período de tiempo) durante el cual se satisface alguna condición, se realiza alguna labor, o se espera a por un evento.

- **Conector**: Representa un criterio de ordenamiento semántico entre Unidades de Modelado y puede ser de tres tipos:

    - **Lineal**: Indica un ordenamiento secuencial entre dos Unidades de Modelado, es decir, una a partir de la otra.

    - **Unión**: Representa la confluencia de un número N > 0 de flujos para producir un flujo de llegada a una Unidad de Modelado. La Unión puede ser de dos tipos, **Unión Total** (indica que se requiere que el control llegue a los N flujos de entrada para generar un flujo de salida) y **Unión Parcial** (indica que el control debe llegar a P flujos, P < N, para generar un flujo de salida)

    - **Bifurcación**: Indica la generación de N > 0 flujos de salida a partir de un flujo de entrada. La Bifurcación puede ser de dos tipos: **Bifurcación Total** (indica que cuando el control llega al flujo de entrada se produce los N flujos de salida) y **Bifurcación Parcial** (Indica que al llegar el control al flujo de entrada se generan P flujos de salida, P < N. Un caso particular de dos salidas en una bifurcación parcial refleja una situación de exclusión mutua lo que permite modelar situaciones de opcionalidad como en los diagramas de flujos)

La clase Representación de Requisitos representa los diferentes diagramas de requisitos que se emplean en el proceso de ingeniería de requisitos. Específicamente, se instancian diagramas que modelan la interacción entre el sistema y entidades externas (por ejemplo, los Diagramas de Escenarios y Casos de Uso), diagramas que modelan los requisitos haciendo énfasis en las funciones de transformación de datos, (como por ejemplo los Diagramas Documentos-Tareas y Flujos de Datos), y diagramas que permiten reflejar la gestión de recursos de la organización, (por ejemplo los Diagramas de Workflow y de Actividades).

Una vez definido el metamodelo, para soportar una estrategia de reutilización con transformación, tenemos que contar con técnicas robustas y formales que permitan eliminar las deficiencias inherentes a los modelos de requisitos. La carencia de una base formal suficiente en los modelos de requisitos limita las posibilidades de soporte de herramientas para su análisis sistemático en el desarrollo con reutilización. Por tanto, es necesario contar con técnicas para expresar los modelos de requisitos en una representación que permita la comprobación automática de ciertas propiedades

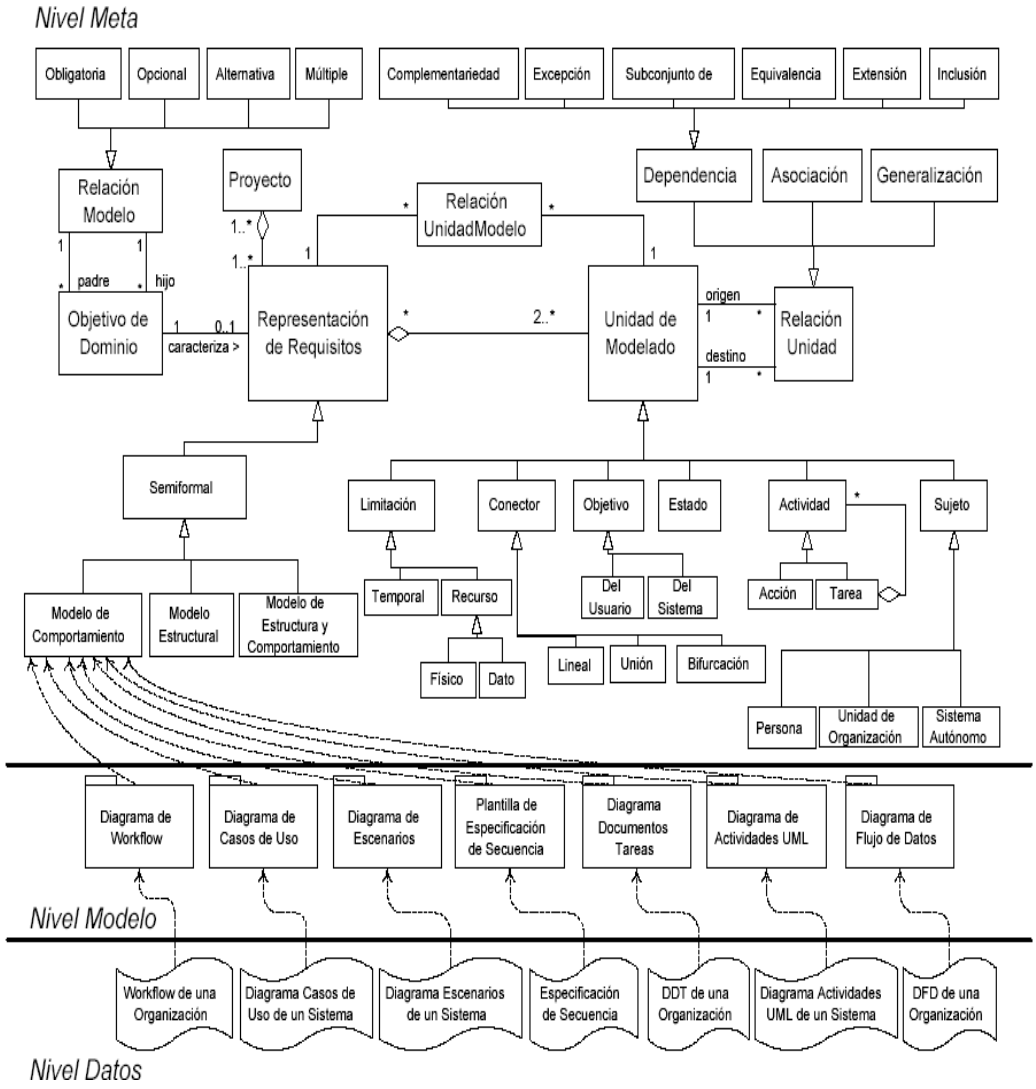


Figura 2. Metamodelo de Representaciones de Requisitos expresado en UML.

4. Modelos Formales para Reutilización de requisitos

La expresión de requisitos mediante el lenguaje natural no satisface las necesidades de especificación para desarrollo con reutilización. Los problemas de ambigüedad, escalabilidad y trazabilidad son inherentes al lenguaje natural Además, la documentación de los requisitos de usuario en lenguaje natural presenta una escasa conexión con la reingeniería del negocio y con la implementación del sistema.

El objetivo de este trabajo de software surge de la necesidad de normalizar la captura de requisitos y su posterior transformación en otros tipos de representaciones. Este proceso se desarrolla en varias etapas, planteando un mecanismo formal y automático para derivar los requisitos funcionales a partir de varios modelos de flujos de trabajos.

Se ha comentado antes que la carencia de una *base* formal suficiente en los modelos de requisitos limita las posibilidades de soporte de herramientas para su análisis sistemático en el desarrollo con reutilización. Por tanto, es necesario contar con técnicas y modelos que permitan representar a las distintos modelos de requisitos con una sintaxis formal y rigurosa.

En consecuencia, se debe buscar un soporte lógico y estructural que permita traducir los modelos de requisitos a una expresión formal, para definir, primero un marco de análisis y posteriormente una estrategia de transformación de requisitos.

En este trabajo para la transformación de requisitos de software, se hace uso del modelo de Grafos de Casos [5][4] al que se le añade el formalismo de las Redes de Petri. La idea consiste en representar los requisitos mediante Grafos de Casos y añadirle otros elementos para que cumplan las propiedades que definen las Redes de Petri. Con las Redes de Petri se logra una descripción formal de los modelos originales de requisitos, sirviendo como soporte al análisis y organización de requisitos para la transformación.

Las redes de Petri (PN) han sido ampliamente utilizadas en el modelado de sistemas cuyo comportamiento es guiado por un patrón de reglas complejas como es el caso de los sistemas software. Gráficamente una red de Petri es un Grafo orientado que contiene dos tipos de nodos. Mediante círculos se representan los lugares y mediante rectángulos las transiciones. Los arcos conectan un lugar con una transición o viceversa, pero no a dos lugares o a dos transiciones. Los lugares y las transiciones permiten anotaciones con las que describir estáticamente un sistema.

Se debe establecer una correspondencia entre el metamodelo de requisitos y los elementos del Grafo de Casos. Los Grafos de Casos proporcionan un buen soporte para la gestión automática de diagramas de requisitos.

4.1 El Modelo de Grafos de Casos.

La labor de ingeniería de este trabajo consiste en modelar la funcionalidad de un sistema representado por un diagrama de requisitos a través de un Grafo de Casos(GC).

Un Grafo de Casos es una estructura en Grafos, que corregido su informalidad mediante el uso del formalismo de las redes de Petri, permite representar los requisitos de software de una manera sistemática y sin ambigüedad. Al corregir la

informalidad de los Grafos de Casos, estaremos en condiciones de obtener mecanismos para comparar requisitos. El marco establecido es claro y permite modelar el negocio con un mínimo de información.

Dado que las redes de Petri se han utilizado para expresar la semántica de los workflow [1], podemos utilizar los Grafos de Caso con este formalismo para modelar diagramas que representen flujos de tareas y cumplan las especificaciones de workflow establecidas por Workflow Managment Coalition (WfMC).

Tomaremos como casos particulares de workflow los diagramas documentos-tareas, diagramas de Actividad UML y workflow y seremos capaces de modelar estos diagramas mediante Grafos de Casos para su posterior análisis y procesamiento.

Los casos de uso (UC) y los casos de uso del negocio (BUC) son una estrategia de captura de requisitos ampliamente aceptada frente a las deficiencias del lenguaje natural [3]. Los UC resuelven las deficiencias de escalabilidad y trazabilidad, y proporcionan facilidad para la descripción.

La primera tarea que realizaremos será modelar la funcionalidad de un sistema a través de un Grafo de Casos (GC). Del análisis del GC se obtendrán familias de casos de uso del negocio (Grafos BUC) y familias de casos de uso (Grafos UC). Las actividades de validación y verificación permitirían corregir la versión inicial del GC. De ser necesario, el proceso puede solicitar una nueva versión del GC , Grafos BUC y los Grafos UC. El modelo de proceso escogido aquí establece dos niveles relacionados para la elicitación de los requisitos: El nivel del usuario y el nivel del Ingeniero de Software. El primero con una visión externa (caja negra) del sistema, y el segundo con la vista interior de esa caja negra. Dentro del nivel del Ingeniero del Software existe la vista del Ingeniero de Requisitos la cual actúa como una interfaz entre los dos niveles anteriores.

A partir del GC verificado, podremos generar los assets que pueden ser expresados en forma de plantillas de casos de uso, diagramas de flujos de datos, diagramas de Actividad UML, diagramas documentos tarea, diagramas de casos de uso, diagramas de WorkFlow, diagramas de Escenarios. Estos pueden ser recibidos por un gestor de Datos para interactuar con el repositorio.

4.2 Grafos de Casos.

El GC es un modelo de la teoría de grafos, y se puede utilizar para describir la funcionalidad de un sistema de información . En él se describen las formas de interacción entre los usuarios y el sistema, especificando un flujo de tareas que se deben ejecutar en un orden determinado.

El GC consta de los siguientes elementos: Tareas, Documentos, Puestos y Arcos. Las Tareas y los Documentos enlazado con Arcos son la base de un Grafo de Casos. En el GC encontramos también los Agentes externos, que emiten los documentos que entran al sistema y reciben los generados por él. En los Puestos se ejecutan las Tareas del sistema.

Definición 1. Grafo de Casos:

Un Grafo de Casos es una cuádrupla $G = (D, T, A, E)$, donde:

D es un conjunto finito de documentos.

T es un conjunto finito de tareas (D ∩ T = ∅).

T es una partición tal que $T = T(AA) \cup T(AO) \cup T(OO) \cup T(OA)$.
Esto es, en un GC existen cuatro tipos de tareas. T(AA), T(AO), T(OO), T(OA) son disjuntos.

A es un conjunto de arcos, $A \subseteq ((DxT) \cup (DxT))$

E: D ∪ T→ ∑ + *es una función que asocia una etiqueta distinta a cada documento y a cada tarea.* ∑ + *es el alfabeto finito de donde se originan las etiquetas.*

En Figura 3 podemos ver un ejemplo de la representación de un proceso de negocio mediante Grafos de Casos. Podremos observar que este consta básicamente de Tareas y documentos enlazados. Además a los GC se le ha añadido otros elementos como Agentes Externos y Puestos, para que puedan dar soporte de manera coherente a las representaciones de requisitos y a todas las unidades de modelado que estos contienen.

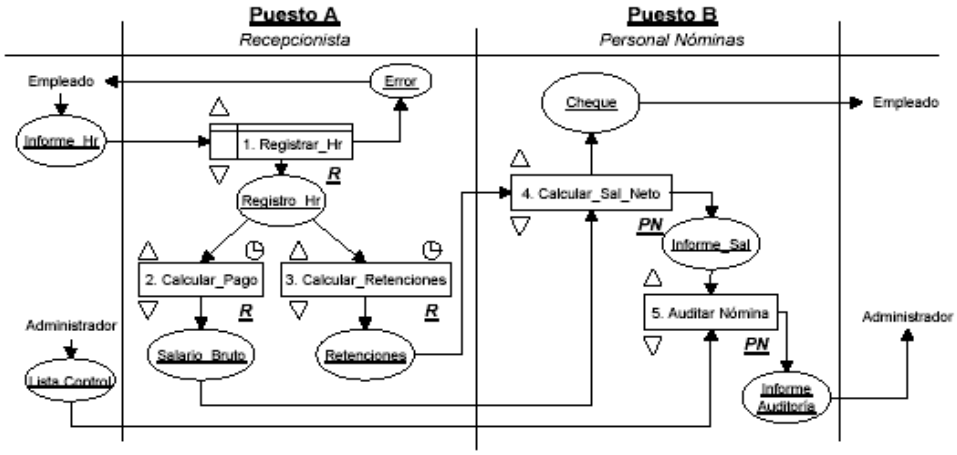
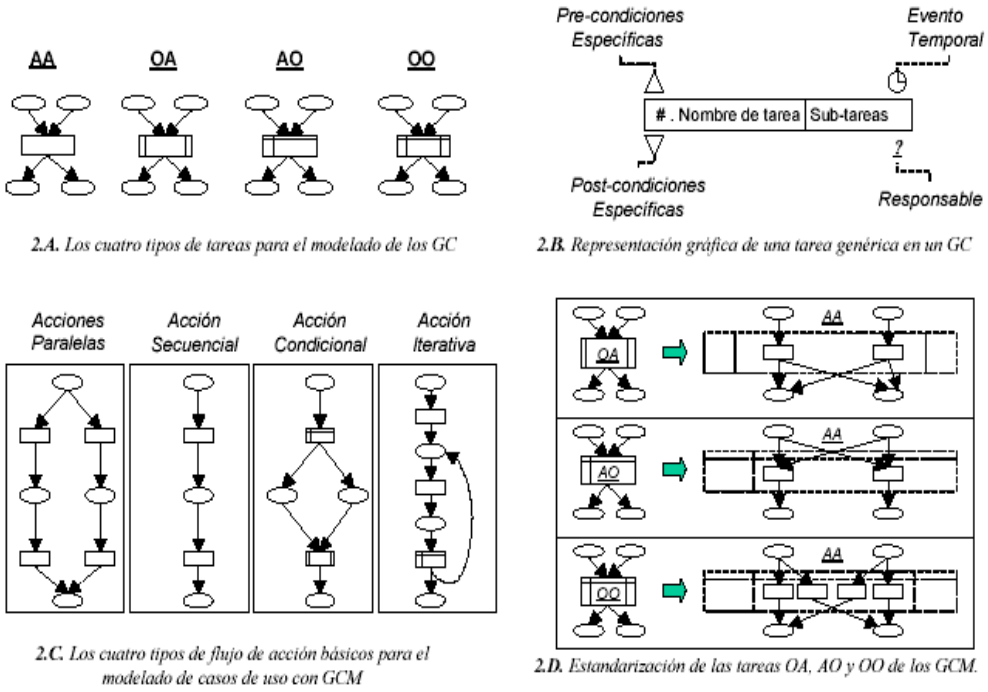


Figura 3 Representación de un proceso de negocio mediante GC.

Según la definición 1, existen cuatro tipos diferentes de tareas (AA, OA, AO, OO) cuya representación gráfica se muestra en el grafico 2.A de la figura 4. Las tareas son los puntos de transición de los documentos. Cada tarea puede tener una especificación de las subtareas que la conforman



2.A. Los cuatro tipos de tareas para el modelado de los GC

2.B. Representación gráfica de una tarea genérica en un GC

2.C. Los cuatro tipos de flujo de acción básicos para el modelado de casos de uso con GCM

2.D. Estandarización de las tareas OA, AO y OO de los GCM.

Figura 4 Representaciones de tareas, transformaciones y flujos de acción para el modelado de grafos de casos.

Se ha escogido una representación particular que refleja los requisitos para que se ejecute una tarea en un contexto organizacional. Para que se ejecute una tarea no sólo es necesario que existan las entradas requeridas (en este caso los documentos de entrada) sino que puede haber pre-condiciones especificas.

Además, la ejecución de una tarea no sólo resulta en la producción de algún documento de salida sino que puede haber postcondiciones específicas. En algunos casos, aún cuando se cuente con las entradas y las recondiciones, las tareas requieren que se produzca un evento temporal para ser ejecutadas. Los diferentes tipos de tareas presentan un comportamiento diferenciado

- Una tarea de tipo AA requiere de la presencia de todos sus documentos de entrada y su disparo genera todos sus documentos de salida.

- El tipo OA se refiere a una tarea que se puede disparar con sólo la presencia de alguno de sus documentos de entrada y su disparo generaría todos sus documentos de salida.

- El tipo AO es aquella tarea que se dispararía ante la presencia de todos sus documentos de entrada y produciría únicamente a alguno de sus documentos de salida.

- El tipo OO es la que se puede disparar con sólo la presencia de alguno de sus documentos de entrada y genera sólo alguno de sus documentos de salida.

Finalmente, cada tarea tiene un responsable directo dentro de la organización. La representación de las tareas y su transformación se muestra en el grafico 2.D de la figura 4. Se han representado los cuatro tipos básicos de flujo de acción mediante Grafos de Casos. Los flujos de acción se muestran en el grafico2.C de la figura 4. Las acciones secuenciales y las paralelas pueden ser descritas mediante tareas del tipo AA. Sin embargo, las acciones condicionales y las iterativas requieren de la adecuada combinación de los tipos AA, OA, AO y OO.

Para realizar un análisis formal de los casos de uso obtenidos, necesitamos homogenizar el comportamiento de las tareas del Grafo de Casos. Las redes de Petri, que han sido utilizadas para expresar la semántica de los workflows, nos pueden proporcionar el soporte formal para analizar un Grafo de Casos. De esta manera las tareas del Grafo de Casos son visualizadas como las transiciones de la red Petri.

Los documentos representan los lugares de la red de Petri. El marcado de la red de Petri podría representar la interacción que muestra la secuencia de estados y no la evolución específica de los casos de uso en conjunto. Para expresar el Grafo de Casos como una Red de Petri se transforman las tareas OA, AO y OO a tareas del tipo AA.

4.3 Grafos BUC y Grafos UC.

El GC es una representación de la funcionalidad del sistema. Por tanto, en él se hallan las formas de interacción entre los usuarios y el sistema. Se distingue dos tipos de interacción de casos de uso: los Bussines Use Cases (BUC) y los Use Cases (UC). Y según el marco de trabajo establecido, a partir del GC se obtienen los BUC y los UC en forma de Grafos BUC y de Grafos UC, respectivamente.

Los Grafos BUC y los Grafos UC reflejan las posibilidades de interacción entre los actores y el sistema. Un Grafo BUC se corresponde con un actor externo. Un Grafo UC se corresponde con un actor interno. Tanto los Grafos BUC como los Grafos UC contienen caminos que puede seguir el flujo de acción del sistema en estudio.

Definición 2. Camino:

Sea $G = (D, T, A, E)$ *Grafo de Casos*, $N = \{n_i \setminus \mathrm{i} = 1..\mathrm{n}, \text{tal que } \mathrm{n_i} \in D \cup T\}$,

Un camino R desde un nodo n_1 a n_k *es una secuencia* $(n_1, n_2, , , n_k)$ *tal que*

$(n_j, n_{j+1}) \in$ A, $\forall$ j $\in$ {1..k-1} .

Esta definición señala que un camino es cualquier secuencia lógica de acción dentro del sistema en estudio. Esta secuencia está compuesta por documentos y tareas los cuales deben estar unidos por arcos dentro del Grafo de Casos. Las tareas son los

puntos de transición de los documentos. Cada tarea tiene asociados dos conjuntos de documentos, los documentos previos y los documentos siguientes. Formalmente se definen estos conjuntos de documentos:

Definición 3. Documentos Previos y Documentos Siguientes:

Sea $G = (D, T, A, E)$ *un Grafo de Casos, el conjunto de documentos previos de la tarea* $t(t \in T)$ *se define* ${}^{o}t = \{d_i \in D \mid \exists x \in A, x\ conecta d_i\ con\ t\}$
Análogamente, el conjunto de documentos siguientes de la tarea t está definido $t^{o} = \{\forall d_i \in D \mid \exists x \in A,\ x\ conecta\ t\ con\ d_i\}$

La consistencia de los Grafos BUC y de los de Grafos UC se puede garantizar si todos sus nodos son alcanzables. Se define un Grafo de Casos fuertemente conectado cuando existe un camino que conecta a cualesquiera dos puntos del grafo:

Definición 4. Tareas Previos y Tareas Siguientes:

Sea $G = (D, T, A, E)$ *un Grafo de Casos, el conjunto de tareas previos del documento* d $(d \in D)$ *está definido por* ${}^{o}d = \{t_i \in T \mid \exists x \in A, x\ conecta t_i\ con d\}$.
Análogamente, el conjunto de tareas siguientes del documento d está definido por $d^{o} = \{\forall d_i \in D \mid \exists X \in A, x\ conecta\ t\ con\ d_i\}$.

Al conjunto de tareas previas se le puede interpretar también como las tareas que generan este documento. Y el conjunto de tareas siguientes como todas las tareas que para su ejecución requieren de dicho documento.

Definición 5. Fuertemente conexo:

Sea $G = (D, T, A, E)$ *un Grafo de Casos. G es fuertemente conectado* $sii\ \forall x \in N, \forall y \in N,\ N = \{n_i \mid i = 1 \ldots n, tal\ que\ n_i \in D \cup T\}$, *existe un camino que conduce desde x hasta y.*

Los casos de uso describen la forma en la cual los usuarios utilizan el sistema. Nuestros Grafos de Casos deben describir los posibles flujos de interacción usuario y sistema. Para esto definimos:

Definición 6. Secuencia de Caso:

Sea $G = (D, T, A, E)$ un Grafo de Casos. G es una Secuencia de Caso (SC) *sii* :

D tiene dos documentos especiales i *y* o.

El lugar ***i*** *es un lugar fuente,* $\alpha(i) = 0$.

El lugar ***o*** *es un lugar sumidero,* $\beta(o) = 0$. *Donde*

$\alpha : DxT \rightarrow \{1,2,3\ldots\}$ *y* $\beta : TxD \rightarrow \{1,2,3\ldots\}$

Si se agrega una tarea t' a T, la cual conecta los documentos i y o (esto es,{i, t',o}es el camino desde i hasta o), entonces el Grafo de Casos resultante es fuertemente conectado.

No existen asociaciones simétricas entre documentos y tareas, es decir, se cumple que, $\forall t_i \in T, {}^{o}t_i \cap t_i^{o} = \varnothing$

Según esta definición y otros estudios, una Secuencia de Caso es sólida si satisface las siguientes definiciones:

- Posee una condición de inicio y una condición de parada.
- Para cualquier documento de entrada, el proceso modelado eventualmente terminara
- No deben existir tareas muertas en el modelo.

A partir de la definición de secuencia de Caso, se define Grafo podemos definir un Grafo de BUC y un Grafo UC. Ambos tipos de grafos están formados por secuencias de casos. Un Grafo BUC es un Grafo de Casos que contiene todas las posibles Secuencias de Caso para un documento de entrada que proviene de actor externo. Un Grafo UC es un Grafo de Casos que contiene todas las posibles secuencias de casos para un actor interno dentro de un Grafo BUC.

Definición 7. Grafo BUC:

Sea $G = (D, T, A, E)$ *un Grafo de Casos. G es un Grafo BUC* sii *contiene todas las posibles secuencias de caso para un documento de entrada que proviene de un actor externo.*

Definición 8. Grafo UC:

Sea $G = (D, T, A, E)$ *un Grafo de casos. G es un Grafo UC* sii *:*

- *Contiene secuencias de caso que corresponden a un actor interno dentro de un Grafo BUC.*
- *Todas la tareas de G son del tipo AA. Esto es,* $T = T(AA)$ *con lo cual se cumple que* $(T(AO) \cup T(OO) \cup T(OA)) = \varnothing$.

Un GC contiene un conjunto de Grafos BUC_i. Cada Grafo BUC_i es un conjunto de Grafos UC_i. Tanto un Grafo BUC_i como un Grafo UC_i consiste de estructuras internas e interfaces externas. Los documentos y las tareas compartidos se consideran parte de las interfaces externas. Las estructuras internas son las mismas que un Grafo de Casos.

4.4 El Grafo de Caso Modular. Refinamiento y Transformación del Grafo de Casos.

Los documentos y las tareas pueden ser compartidos por diferentes secuencias de caso. Esto ocurre porque un mismo documento puede ir a, o provenir de, diferentes tareas. Por tanto, puede haber intersecciones entre las diferentes secuencias de caso.

Los Grafos BUC pueden compartir documentos y tareas, pero los Grafos UC sólo pueden compartir documentos. Cada Grafo UC recoge el tratamiento de la información aplicable a diferentes secuencias de caso dentro de un Grafo UC. Dado que el tratamiento de la información se realiza en las tareas, estas no son compartibles entre los Grafos UC.

Los documentos compartidos por los Grafos UC actúan como puntos de conexión entre diferentes grafos de dicho tipo. Esta situación nos ofrece la posibilidad de tratar al Grafo de Casos como una estructura modular según la definición siguiente:

Definición 9. Grafo de Casos Modular:

Un Grafo de Casos Modular (GCM) es un conjunto $\{G_i = (D_i, T_i, A_i, E_i) \mid i = 1..n\}$ *donde:*

- *Cada* G_i *es un Grafo UC*
- *Los* T_i *deben ser disjuntos para todos los* G_i
- *Una misma etiqueta no debe ser utilizada para documentos y tareas a la vez, esto es:* $\forall G_i, \forall G_j, \neg\exists d \in D_i, \neg\exists t \in T_j \; tal\ que\ E_i(d) = E_j(t)$

Se debe realizar un proceso sobre los GC para obtener el GCM. Este proceso consiste en el refinamiento y transformación de las tareas del Grafo de Casos , factorizacion del GC. El proceso de factorizacion permite descubrir los bloques que son comunes a diferentes grafos BUC.

A partir de un GC se pueden extraer automáticamente los casos de uso. Para esto se requiere un proceso algorítmico para el refinamiento y la transformación de tareas que permita que el GC sea expresado como la combinación de Grafos UC.

La generación automática de los Grafos UC debe considerar la especificación de cada tarea, su tipo, su precondición y post-condición. Además se debe conocer la cantidad de documentos de entrada y de salida. El refinamiento queda reflejado como indica la figura 5. Procediendo de la manera que indica el siguiente algoritmo, todas las tareas del Grafo de Casos quedaran expresado en termino de Tareas de Tipo AA, y por tanto el Grafo de Casos quedará expresado en términos del estándar AA.

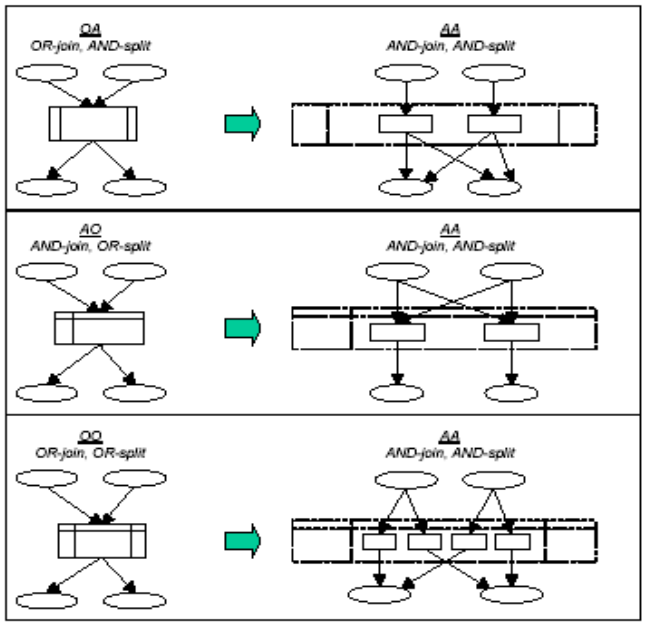


Figura 5. Representación de la transformación de las tareas.

Algoritmo 1. Refinamiento y Transformación de Tareas.

Sea $M = G(D,T,A,E)$ un Grafo de Casos.

Para cada $t \in T$ *hacer*

En caso de $t \in T(OA)$

- *Transformar a t en* $\{t_{(AA)S}\}$ *de modo que s = 1,2,..j donde* $j = |^o t|$, $t_{(AA)S}$ *,es una tarea de tipo AA.*
- *Conectar a cada* $d \in t^o$ *con cada* $\{t_{(AA)S}\}$ *tal que* $^o\{t_{(AA)S}\} = 1$
- *Conectar a cada* $d \in t^o$ *con cada* $\{t_{(AA)S}\}$ *tal que* $\{t_{(AA)S}\}^o = |t^o|$
- *Hacer* $T = T \mid \{t\} \cup \{t_{(AA)S}\}$

En caso de $t \in T(AO)$

- *Transformar a t en* $\{t_{(AA)S}\}$ *de modo que s = 1,2,..j; donde* $j = |t^o|$, $t_{(AA)S}$ *, es una tarea de tipo AA.*
- *Conectar a cada* $d \in {}^o t$ *con cada* $\{t_{(AA)S}\}$ *tal que* $^o\{t_{(AA)S}\} = |^o t|$
- *Conectar a cada* $d \in t^o$ *con una distinta* $\{t_{(AA)S}\}$ *tal que* $\{t_{(AA)S}\}^o = 1$
- *Hacer* $T = T \mid \{t\} \cup \{t_{(AA)S}\}$.

En caso de $t \in T(OO)$

- *Transformar a t en* $\{t_{(AA)S}\}$ *de modo que s = 1,2,..j; donde* $j = |^o t \mid x \mid t^o, \{t_{(AA)S}\}|$ *, es una tarea de tipo AA.*
- *Conectar a cada* $d \in {}^o t$ *con* $|t^o|$ *distintas* $\{t_{(AA)S}\}$ *tal que* $^o\{t_{(AA)S}\} = 1$
- *Conectar a cada* $d \in t^o$ *con* $|^o t|$ *distintas* $\{t_{(AA)S}\}$ *tal que* $\{t_{(AA)S}\}^o = 1$ *y* $\forall t_{(AA)S1}, t_{(AA)S2}$ *si* $^o t_{(AA)S1} = {}^o t_{(AA)S2}$ *y* $t_{(AA)S1}{}^o = t_{(AA)S2}{}^o$ *entonces* $t_{(AA)S1} = t_{(AA)S2}$.
- *Hacer* $T = T \mid \{t\} \cup \{t_{(AA)S}\}$.

Tomar a cada $t \in T$ *como una cadena de tareas genéricas.*

4.5. Factorizacion del Grafo de Casos.

La factorizacion permite proceder a la especificación de los bloques comunes en el GC. La identificación de esos bloques comunes produce una expresión factorizada del GC conservando intacta la estructura particular de cada uno de los Grafos BUC.

Si se expresa la factorizacion mediante tareas del tipo AA, se satisface la definición de un Grafo UC. Con lo anterior, se garantiza que los casos de uso modelados serán los estrictamente necesarios para especificar la funcionalidad del sistema. Este proceso de factorizaron se desarrolla según el algoritmo siguientes.

Algoritmo 2. Factorizacion del GC.

Sea $M = G(D, T, A, E)$ un Grafo de Casos.

Obtener $S = \{SC_d, d \in D \; tal \; que SC_d$ *es una Secuencia de Caso correspondiente al documento externo d }*

- *Para cada* $SC_d \in S$
- *Obtener* T_{SCd} *como* $\bigcup_{j=1}^{n} T_{SCdi}$ *tal que* T_{SCdi} *es el conjunto de tareas de la SC que corresponde al documento d y al actor interno i*
- *Obtener* $T_{SCdi}^{\sim}$ *que se define recursivamente como sigue:* $T_{SCd1} = T_{SCd1}$.
  $\forall i = 2 \ldots n, \forall d = 1 \ldots k, T_{SCdi}^{\sim} = T_{SCdi} - \bigcup_{j=1}^{i-1} T_{SCdj}^{\sim}$.

Tomar a cada $T_{SCdi}^{\sim}$*, y los documentos, arcos, y etiquetas asociadas como un Grafo UC Preliminar.*

Finalmente, cada Grafo UC correspondería a una red de Petri y también todo el GCM sería una red de Petri. Las labores de validación y de verificación determinadas por el marco de trabajo permitirían realizar los ajustes necesarios en las tareas. Con estas bases se puede plantear un proceso general de transformación de requisitos que se presenta en las siguientes secciones.

5 Proceso General de Transformación

En este apartado y el siguiente se analiza el proceso de transformación de requisitos de software en todas sus etapas. Este proceso consta de una secuencia de actividades

que dependen de la transformación que estemos realizando, es decir, del diagrama inicial y final.

Como punto de partida se ha escogido utilizar los workflows o flujos de trabajo modelados mediante Grafos de Casos. Debido a la informalidad que rodea a las representaciones de requisitos, a los Grafos de Casos se le añade la formalidad de las redes de Petri. Es necesario contar con estas técnicas y estructuras (como el formalismo robusto proporcionado por las redes de Petri) para permitir un análisis sistemático y comprobación de propiedades.

Después de expresar los requisitos funcionales en un modelo formal y realizar sobre este un análisis sistemático y comprobación de propiedades, se presenta una estrategia para la generación automática de requisitos funcionales (casos de uso, escenarios, workflow, flujos de datos, docflow, y diagramas de actividad) y de otros elementos reutilizables de nivel de análisis.

En síntesis, se presenta una propuesta para la normalización del proceso de captura de requisitos de usuario mediante un workflow o flujo de trabajo, su representación en modelo de Grafos de Casos con la formalidad de las redes de Petri y su posterior análisis y utilización para producir otros requisitos funcionales. Abajo se enumeran los pasos y etapas de este proceso.

1. Obtener una representación de Requisitos desde un proyecto.

2. Modelar esta representación de requisitos mediante Grafo de Casos.

3. Analizar el Grafo de Casos obtenido, comprobando su validez para aplicar sobre éste los algoritmos de transformación. Este análisis deberá demostrar la consistencia del Grafo de Casos.

4. Generar los Grafos BUC y los Grafos UC a partir del Grafo de Casos. Este proceso es un proceso algorítmico que recorre el Grafo Casos(GC), descubriendo las secuencias de casos para cada documento de entrada

5. Refinamiento y transformación del Grafo de Casos, transformando todas las tareas de tipo AO, OA, OO a tareas de tipo AA.

6. Factorizar el Grafo de Casos para descubrir las estructuras comunes entre los grafos BUC y los Grafos UC. Este proceso concluye con la obtención del grafo de Caso Modular (GCM).

7. A partir del Grafo de Caso Modular extraer los assets de Requisitos que pueden posteriormente ser visualizados, editados o almacenados en un repositorio.

El proceso anteriormente descrito puede en algunos casos no llevarse a cabo en su totalidad. Las similitudes estructurales entre varios diagramas de requisitos, pueden no hacer necesario ejecutarlo en su totalidad. Eso implica que algunos assets de requisitos se podrán extraer durante algunas etapas de este proceso, antes de llegar al Grafo de Caso Modular (GCM).

En Figura 6 se puede observar las etapas y actividades que integran este proceso de transformación.

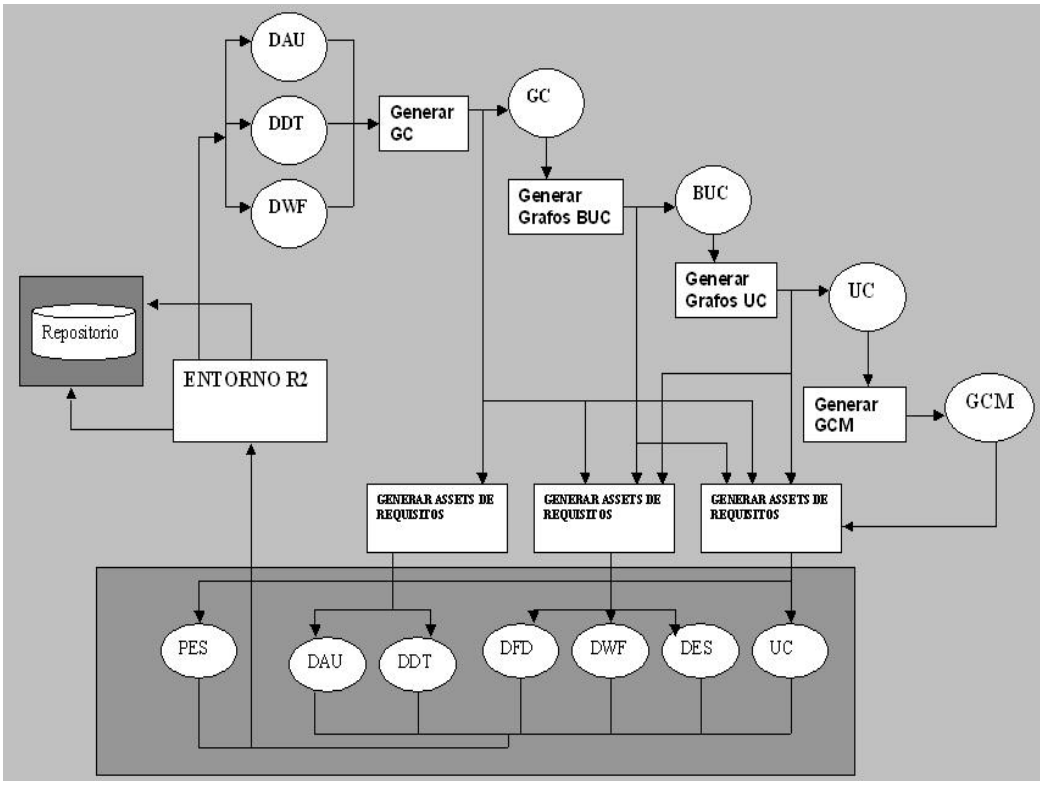


Figura 6 Marco general de trabajo para la transformación de los assets de requisitos.

6 Fases del Proceso de Transformación

6.1 Obtener una Representación de Requisitos

En esta fase se elige desde la herramienta de reutilización de requisitos el diagrama que se va a transformar, en qué tipo se desea transformar y demás información que se quiera incluir en el nuevo diagrama. En este trabajo solo se han tratado las

transformaciones que tienen como diagrama inicial uno que representa un modelo de workflow o flujo de actividades. Por ello solo podremos transformar diagramas documento-tareas (DDT), Actividad UML (DAU) y WorkFlow (DWF).

Como diagramas a obtener podemos elegir cualquiera de las siguientes representaciones de requisitos: diagrama documentos-tarea (DDT), diagrama Actividad UML (DAU), diagrama de workflow (DWF), diagrama de escenarios (DES), diagrama de flujos de datos (DFD), diagrama de casos de uso (DUC) y también obtendremos las Plantilla de Especificación de Secuencia (PES) para cada actividad que contiene el diagrama.

6.2 Generar el Grafo de Casos

Para aprovechar el soporte estructural y formal de los Grafo de Casos en nuestro marco de reutilización se debe establecer una correspondencia entre las metaclases que conforman el metamodelo de requisitos y los elementos del Grafo de Casos. La propuesta que se ha definido establece esta correspondencia en los siguientes términos

Unidades de modelado:

- ***Sujeto***:se corresponden con los ***Puestos*** y ***Agentes*** del Grafo de Casos. En esta categoría están incluidas las unidades de modelados Puesto de trabajo docflow, Participante de workflow, Aplicación workflow, y Swimlane de Actividad UML y Actor externo del diagrama DDT.

- **Actividad** (tarea o acción): se corresponde con ***Transiciones*** o tareas del Grafo de Casos. En esta categoría están incluidas las unidades de modelado Actividad de workflow, Actividad UML y Task del diagrama documentos-tarea.

- **Limitación:** se corresponden con ***Lugares*** o documentos del Grafo de Casos. En esta categoría quedan incluidas las unidades de modelado Document del diagrama DDT y Datos relevantes de workflow.

- **Conector:** Se corresponden con ***Lugares*** o documentos del Grafo de Casos. En esta categoría quedan incluidas las unidades de modelado: Información de Transicion de los diagramas de workflow y Actividad UML.

Relaciones Unidad

Las relaciones entre Unidades de Modelado se corresponden normalmente con Los *Arcos* del Grafo de Casos, pero algunas relaciones pueden asignarse a propiedades y atributos de los elementos del Grafos de Casos.

Se puede observar que algunas metaclases y unidades de modelado no están reflejados en esta propuesta. Solo se han reflejado las unidades de modelado que forman parte de los diagramas que podemos tener como inicio en una transformación. Por eso las metaclases como Estado y Objetivo no se están reflejados. Los tres diagramas elegidos como punto de partida no las instancian.

Aunque ya hemos descrito el proceso general de traducción de una representación de requisitos en Grafo de Casos (GC), cada diagrama tiene sus particularidades. Por lo que pasaremos a describir este proceso para cada tipo de diagrama inicial.

6.2.1 Transformar el diagrama documentos-tareas en GC

Para transformar un diagrama documentos-tareas (DDT) en un Grafo de Casos(GC), se recorren las unidades de modelado del diagrama y se crean elementos del grafo de casos, según la correspondencia que se ha establecido. En este caso en particular el proceso es el siguiente:

1. Se crean los *Puestos* del Grafo de Casos a partir de las unidades de modelado Puesto trabajo docflow, instancias de la metaclase Sujeto.

2. Se crean las *Transiciones* del Grafo de Casos a partir de las unidades de modelado Tarea de docflow, instancias de la metaclase Actividad.

3. Se crean los *Lugares* del Grafo de Casos a partir de las unidades de modelado documento docflow, instancias de la metaclase Limitación.

4. Se crean los *Agentes* del Grafo de Casos a partir de las unidades de modelado Actor Externo de DocFlow , instancias de la metaclase Sujeto.

Obtenemos los *Arcos* del Grafo a partir de las relaciones unidad entre unidades de modelado del diagrama DDT. El arco tiene como origen y destino los nodos del grafo que se obtienen de las unidades origen y destino de esta relación.

6.2.2 Transformar el diagrama Actividad UML en GC

Para transformar un diagrama Actividad UML en un Grafo de Casos, recorremos las unidades de modelado del diagrama y Se crean los elementos del GC según la correspondencia que se ha establecido.

1. Se crean los *Puestos* del GC a partir de las Unidades de modelado Swimlane, instancias de la metaclase Sujeto. Además se crea de manera automática un

Puesto que hace las funciones de puesto externo, necesario para los agentes externos que interactúan con el sistema en el Grafo de Casos.

2. Se crean las *Transiciones* del Grafo de Casos partir de las unidades de modelado Actividad UML primaria y Actividad UML compleja, instancias de la metaclase Actividad.

3. Se crean los *Lugares* del Grafo de Casos a partir de las unidades de modelado conector lineal, conector bifurcación parcial, conector bifurcación total, instancias de la metaclase Conector. La transformación de los conectores en lugares del Grafo se realiza de la siguiente manera:

    - Un *Conector Lineal* representa un flujo lineal en el diagrama Actividad UML y se transforma en un lugar en el Grafo de Casos . Este se conecta con una transición origen y una destino.

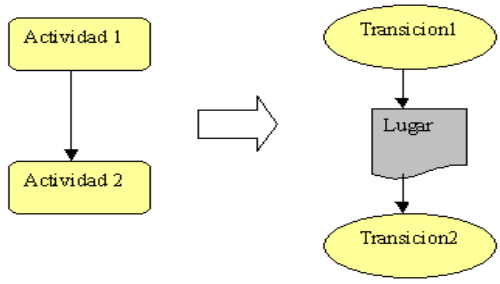


Figura 6 Conector Lineal

    - Un *Conector Bifurcación* parcial o total representa un flujo optativo en el diagrama Actividad UML y se transforma en un lugar en el GC. Este se conecta a una transición fuente y a dos destinos.

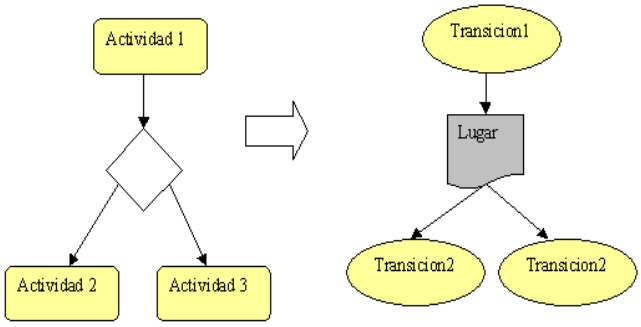


Figura 7 Conector Bifurcación parcial

- Un *Conector Unión* parcial o total representa un flujo concurrente en el diagrama Actividad UML, y se transforma en tantos lugares como entradas tenga el flujo unión.

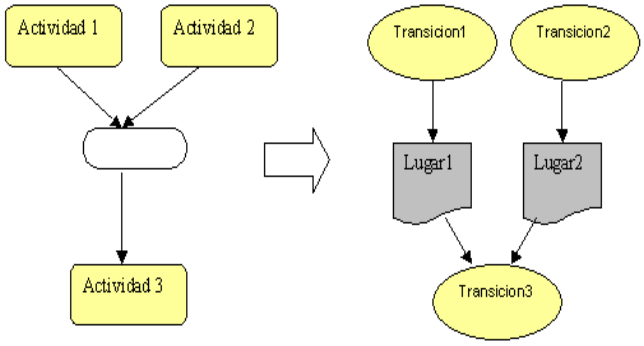


Figura 8 Conector Unión parcial

4. Puesto que el diagrama de Actividad UML no tiene unidades de modelado que se pueden asociar con agentes del GC, en este trabajo se ha propuesto crear dos agentes de manera automática. Uno se encarga de emitir documentos al sistema y el otro recibirá todos los documentos que el sistema envía al exterior.

5. Se crea un arco en el GC por cada relación unidad entre unidades de modelado en el diagrama de Actividad UML

6.2.3 Transformar el diagrama de WorkFlow en GC

Para transformar un Diagrama de WorkFlow en un GC, recorremos las unidades de modelado del diagrama y Se crean los elementos del Grafo de Casos, según la correspondencia que se ha establecido.

1. Se crean los *Puestos* del GC a partir de las Unidades de modelado del tipo Participante de workflow, instancias de la metaclase Sujeto. Además se crea de manera automática un Puesto que hace las funciones de puesto externo, necesario para los Agentes externos que interactúan con el sistema en el Grafo de Casos.

2. Se crean las *Transiciones* del Grafo de Casos a partir de las unidades de modelado del tipo Actividad primaria y Actividad compleja de workflow, instancias de la metaclase Actividad.

3. Se crean los *Lugares* del GC a partir de las unidades de modelado del tipo dato relevante workflow, conector bifurcación parcial, conector bifurcación

total y conector lineal, instancias de la metaclase Limitación y Conector. La transformación de los conectores en lugares del grafo se realiza según el algoritmo:

- Un C*onector Lineal* representa un flujo lineal en el diagrama workflow y se transforma en un Lugar en el GC. Este se conecta a luna transición de inicio y una final.

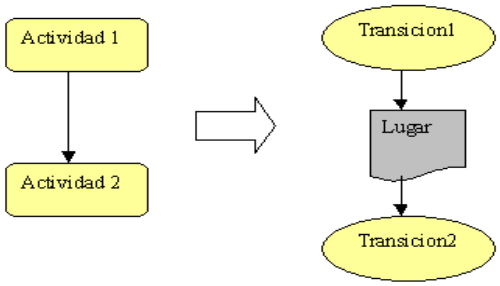


Figura 9 Conector Lineal

- Un *Conector bifurcación parcial o bifurcación total* representa un flujo optativo en el diagrama e workflow y se transforma en un lugar en el GC, y éste se conecta con las transiciones correspondientes.

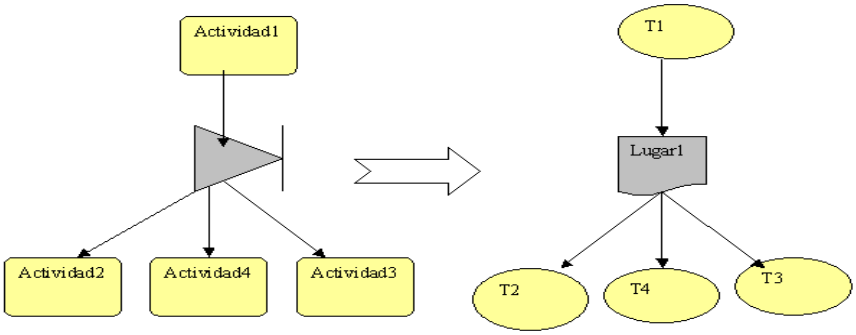


Figura 10 Conector bifurcación parcial

- Un *Conector Unión Parcial o Unión total* representa un flujo concurrente en un workflow y se transforma en tantos lugares como actividades de entrada tenga el conector. Luego se une a cada lugar con una transición fuente y una transición destino.

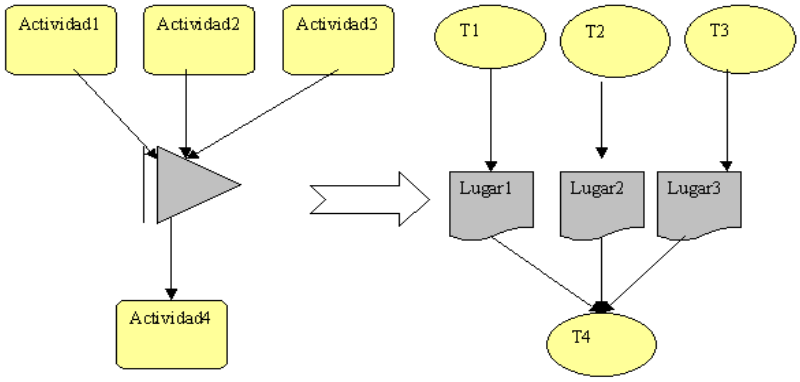


Figura 11 Conector Unión Parcial

4. Puesto que el Diagrama de workflow no tiene unidades de modelado que se pueden asociar con Agentes del GC, en este trabajo se ha propuesto crear dos agentes de manera automática. Uno se encarga de emitir documentos al sistema y el otro recibe todos los documentos que el sistema envía al exterior.

5. Se crea un arco en el GC por cada relación unidad entre unidades de modelado en el diagrama de workflow.

6.3 Analizar la consistencia del Grafo de Casos

Posteriormente a la obtención del Grafo de Casos (GC) se procede a analizarlo para determinar si es consistente y válido. La consistencia del GC se puede garantizar si se cumplen las siguientes condiciones:

- El Grafo de Casos no tiene ciclos: Esta condición se impone para garantizar la integridad de las Secuencias de Caso, y por tanto de los Grafos BUC. En un GC con ciclos no se cumple la definición de las Secuencias de Caso.

- Todos sus nodos son alcanzables: La consistencia del GC requiere que no existan elementos aislados o inaccesibles en el GC. El análisis verifica si todos los lugares y transiciones son alcanzables.

- Todas las Actividades deben tener asociado un Sujeto: Por tanto, si en un diagrama existen actividades sin sujeto que las realice, este diagrama dará lugar a un grafo inconsistente. Pero esta situación se puede resolver asignado un sujeto a la actividad.

También como parte del análisis del GC, se comprueba si todas las actividades tiene asignado un tipo. Las actividades pueden ser de tipo AA, AO, OA o OO. Para algoritmos posteriores se requiere que las actividades del Grafo de Casos tengan asignado un tipo.

6.4 Generar Grafos BUC y Grafos UC

A partir del Grafos de Casos analizado y consistente se obtienen los BUC y los UC . Merece ser resaltado que la generación de los BUC y de los UC a partir del GC es un proceso algorítmico. Es decir, teniendo el GC debidamente verificado, los BUC y los UC se obtienen automáticamente.

Para la obtención de los BUC el procedimiento consiste en tomar cada uno de los documentos de entrada y obtener todos los caminos posibles a través de la red para volver al punto de origen del mismo documento. Mediante este proceso vamos construyendo las Secuencias de Caso del Grafo BUC. Para obtener los UC a partir de un BUC se procede a agrupar para cada actor interno las transiciones y los correspondientes documentos relacionados de entrada y salida.

6.5 Generar el Grafo de Caso Modular.

Los documentos y las tareas pueden ser compartidos por diferentes secuencias de caso. Esto ocurre porque un mismo documento puede ir a, o provenir de, diferentes tareas. Por tanto, puede haber intersecciones entre las diferentes secuencias de caso. Los Grafos BUC pueden compartir documentos y tareas, pero los Grafos UC sólo pueden compartir documentos. Cada Grafo UC recoge el tratamiento de la información aplicable a diferentes secuencias de caso dentro de un Grafo BUC. Dado que el tratamiento de la información se realiza en las tareas, estas no son compartibles entre los Grafos UC.

Los documentos compartidos por los Grafos UC actúan como puntos de conexión entre diferentes grafos de dicho tipo. Esta situación nos ofrece la posibilidad de tratar al Grafo de Casos como una estructura modular. Para obtener el GCM se debe hacer un análisis del GC para descubrir los bloques de acción comunes a diferentes Grafos BUC. La identificación de estas estructuras comunes permite hacer una factorización del GC conservando intacta la estructura particular de cada uno de los Grafos BUC. Para obtener el Grafo de Caso modular tenemos que refinar el Grafo de Casos y luego factorizarlo.

6.5 Refinar y Transformar las tareas del Grafo de Casos

A partir de un GC se pueden extraer automáticamente los assets de requisitos pero para esto se requiere un proceso algorítmico para el refinamiento y la transformación de tareas que permita que el GC sea expresado a través de Tareas de tipo AA. La generación automática de los Grafos UC debe considerar la especificación de cada

tarea, su tipo, su precondición y post-condición. Además se debe conocer la cantidad de documentos de entrada y de salida. Este proceso se realiza mediante el algoritmo 1.

6.6 Factorizar el Grafo de Casos.

La factorización permite proceder a la especificación de las fracciones comunes entre Grafos BUC. Si se expresa la factorización mediante tareas del tipo AA, se satisface la definición de un Grafo UC. Con lo anterior, se garantiza que los casos de uso modelados serán los estrictamente necesarios para especificar la funcionalidad del sistema.

Finalmente, cada Grafo UC correspondería a una red de Petri y también todo el GCM sería una red de Petri. Las labores de validación y de verificación determinadas por el marco de trabajo permitirían realizar los ajustes necesarios en las tareas.

6.7 Generar los Assets de Requisitos

El proceso de obtención de assets de requisitos a partir del grafo de Casos, grafos BUC, grafos UC, y Grafo Caso Modular, depende del tipo de requisitos que queremos generar. Por tanto, para cada tipo de requisitos a obtener utilizaremos determinados algoritmos y grafos.

6.7.1 Generar diagrama documento-tareas (DDT)

El diagrama DDT, debido a su semejanza estructural con el Grafo de Casos, se obtiene directamente de éste. Para ello, establecemos una correspondencia de conversión entre los elementos del Grafo de Casos y los elementos del DDT. Nótese que el diagrama DDT se obtiene después del primer paso del proceso de transformación, la generación del GC.

Esta es la propuesta de correspondencia que se ha establecido.

- Los *Puestos* de Trabajo del diagrama documentos-tareas, se obtienen a partir de los Puestos del Grafo de Casos.

- Las Actividades del diagrama documentos-tareas, se obtienen a partir de las Transiciones del Grafo de Casos.

- Los Documentos del diagrama documentos-tareas, se obtiene a partir de los Lugares del Grafo de Casos.

- Los Agentes externos del diagrama documentos-tareas, se obtiene a partir de los Agentes del Grafo de Casos.

- Los arcos del Grafo de Casos, establecen las relaciones entre los elementos del diagrama documentos-tareas.

En términos generales el proceso de transformación de un GC en un diagrama DDT es sencillo debido a que los dos modelos poseen estructuras similares.

6.7.2 Generar diagrama Actividad UML (DAU)

Se ha propuesto que el diagrama Actividad UML se obtenga a partir del Grafo de Casos y de los Grafos BUC, después de la etapa de generación de estos últimos. Se ha establecido que todos los diagramas Actividad UML a obtener constaran de dos niveles: un diagrama Actividad UML de nivel 1 y varios diagramas de segundo nivel, explosiones de los procesos complejos del nivel 1.

1. En primer lugar se crea un diagrama Actividad UML de primer nivel. Este diagrama estará constituido solo por actividades complejas, se crea una actividad por cada Grafo BUC que tenga el grafo de Casos.

2. Se explosiona cada actividad compleja del diagrama anterior en otro diagrama de Actividad. Este nuevo diagrama tendrá como unidades, los que se obtengan de los elementos del Grafo BUC.

Para obtener este diagrama a partir del Grafo de Casos y del grafo BUC, debemos establecer una correspondencia entre los elementos del Grafo de Casos (GC) y los elementos del diagrama Actividad UML. Esta es la propuesta de correspondencia que se ha establecido.

- Los Unidades del tipo Swimlane del diagrama de Actividad, se obtiene a partir de los Puestos del Grafo de Casos.

- Las Actividades primaria y compleja del Diagrama de Actividad UML , se obtiene a partir de las Transiciones del Grafo de Casos.

- Los Agentes Externos del Grafo de Casos, no se utilizan, porque el diagrama de Actividad UML, no tiene unidades de modelado que representen a este concepto.

- Los conectores Lineal, bifurcación y Unión del Diagrama Actividad UML, se obtiene a partir de los Lugares del Grafo de Casos. Dependiendo del tipo de Lugar, podremos obtener un conector lineal, conector bifurcación, conector unión. Las siguientes variantes detallan mas este proceso.

  Sean las funciones ${}^{o}T(L) = x$ y $T(L)^{o} = y$ que definen el numero de transiciones previos y el numero de transiciones siguientes de un lugar L en el GC, respectivamente.

1. Si el Lugar tiene un ${}^{o}T(L)=1$ y $T(L)^{o}=1$, es decir, es generado y por una transición y consumida también por una transición. Este lugar determina un flujo lineal y se transforma en un conector lineal del diagrama de Actividad UML.

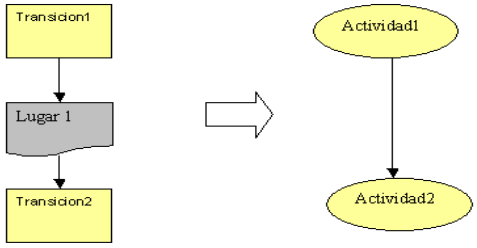


Figura 12 Conector lineal del diagrama de Actividad UML

2. Si el Lugar tiene ${}^{o}T(L)=1$ y $T(L)^{o}=2$,es decir, es generado por una única transición y consumida por dos transiciones. Este lugar puede determinar un flujo optativo o dos flujos lineales.

   - Un bifurcador si este lugar determina un flujo optativo. Se obtuvo a partir de una bifurcación en el diagrama original.

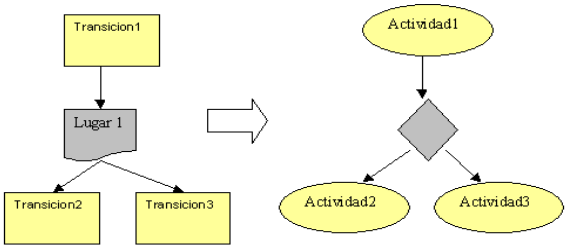


Figura 13 flujo optativo

   - En dos conectores lineales, si este lugar determina dos flujos lineales.

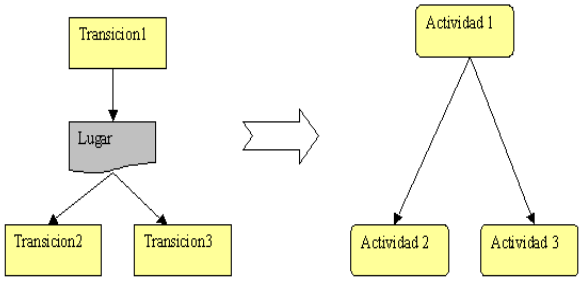


Figura 14 flujos lineales

3. Si el lugar tiene ${}^{o}T(L) = x > 1$ y $T(L)^{o} = y > 1$,se transforma en tantos flujos como pares (x, y) se puedan dar. Y se conecta cada flujo con una transición origen y una final.

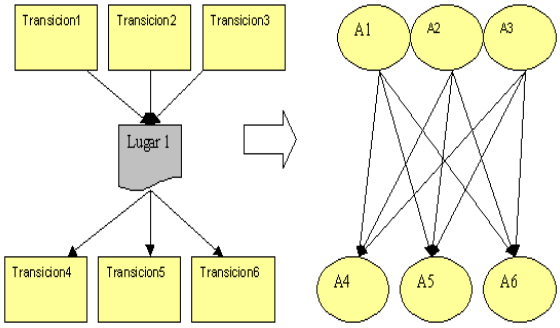


Figura 15 Se conecta cada flujo con una transición origen y una final

4. Finalmente, si una transición recibe varios documentos y es de tipo AND de entrada, se establece un flujo concurrente. Se crea un conector unión para sincronizar la recepción de todos los documentos e inicio de la ejecución de esta actividad.

6.7.3 Generar diagrama de WorkFlow (DWF)

Se ha establecido que el Workflow a obtener constara de dos niveles, un primer nivel constituido solo por actividades complejas y un segundo nivel formado por todos los workflow explosión de cada actividad compleja de primer nivel. Los pasos de este proceso de transformación son:

1. En primer lugar se crea un Diagrama Workflow de primer nivel. Este diagrama estará constituido por actividades complejas, se crea una actividad por cada Grafo BUC que tenga el grafo de Casos.

2. Se explosiona cada Actividad compleja del diagrama anterior en otro diagrama de workflow. Este nuevo diagrama tendrá como unidades, los que se obtengan de los elementos del Grafo BUC.

Para ello se debe establecer una correspondencia entre los elementos del GC, Grafo BUC y los elementos del workflow. La propuesta de correspondencia establecida es la siguiente:

- Los elementos del tipo *Puestos* del Grafo de BUC se convierten en Participante de workflow y Aplicación de workflow.

- Los elementos del tipo *Transición* del Grafo BUC se convierten en Actividades del workflow.

- Los elementos del tipo *Agente* del Grafo BUC, no se utilizan en este caso porque el workflow no tiene unidades de modelado que representen este concepto.

- Los elementos del tipo *Lugar* del Grafo BUC se convierten en Datos relevantes y Conectores del workflow.

  Se deben analizar las distintas situaciones en las un lugar en el Grafo puede convertirse en cada una de estas unidades de modelado.

  Sea la función definida antes: ${}^{o}T(L) = x$ y $T(L)^{o} = y$

  1. Si el lugar tiene ${}^{o}T(L) = 0$ y $T(L)^{o} = y \geq 1$,que corresponde con un documento que viene del exterior, entonces se transforma en un Datos relevantes de workflow.

  2. Si el lugar tiene ${}^{o}T(L) = x \geq 1$ y $T(L)^{o} = 0$, que corresponde a un documento final ,que sale al exterior del sistema o se queda en el interior (documento sumidero), se transforma también en datos relevantes de workflow.
  3. Si el lugar tiene ${}^{o}T(L) = 1$ y $T(L)^{o} = 1$, este lugar determina un flujo secuencial, y se transformara en un Conector simple del diagrama de workflow.

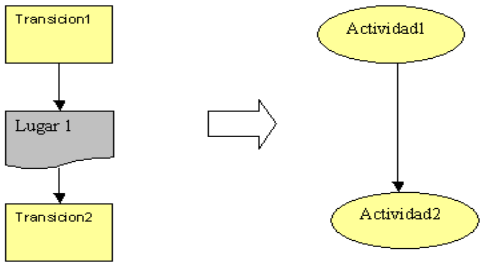


Figura 16 Conector simple

4. Si el lugar tiene ${}^{o}T(L)=1$ y $T(L)^{o}=y\geq 1$, se transformarse en:

   - Un conector bifurcador, si el lugar determina un flujo optativo

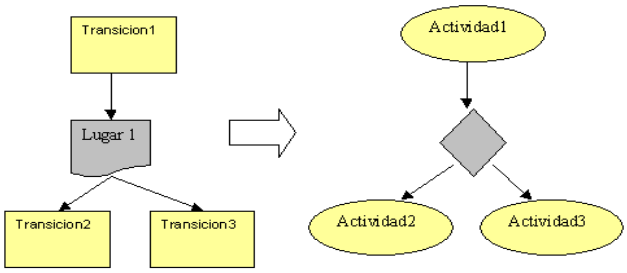


Figura 17 Conector bifurcador

   - Tantos flujos como pares (x, y) se pueden dar, si el lugar determina múltiples flujos lineales.

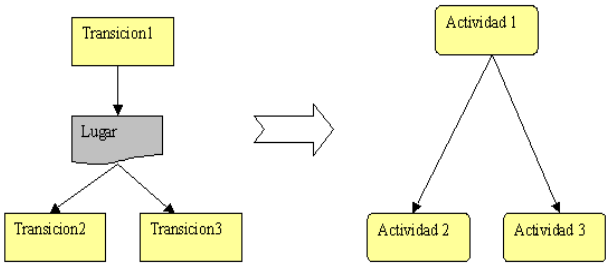


Figura 18 Múltiples flujos lineales

5. Si el lugar tiene ${}^{o}T(L)=x\geq 1$ y $T(L)^{o}=y\geq 1$, determina múltiples flujos lineales, y se transforma en tantos flujos como pares (x, y) se puedan dar.

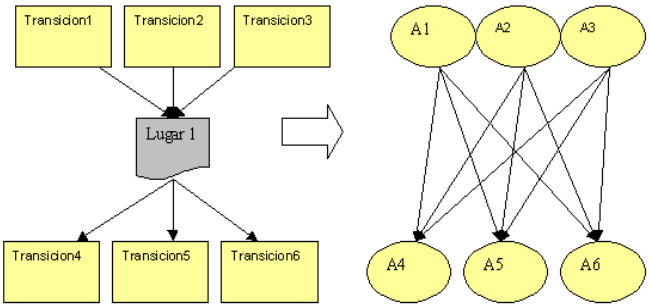


Figura 19 Múltiples flujos lineales

6. Finalmente, si una transición recibe varios documentos y es de tipo AND de entrada, se establece un flujo concurrente. Se crea un conector unión para sincronizar la recepción de todos los documentos e inicio de la ejecución de esta actividad.

6.7.4 Generar diagrama Flujo de datos (DFD)

Se ha establecido que los DFD que se obtengan de este proceso de transformación tengan tres niveles. El primer nivel constituye el diagrama de Contexto, el segundo nivel o subsistemas constituye la explosión del único proceso del nivel y el tercer y ultimo nivel será la explosión y el detalle de los procesos del nivel subsistemas.

Los pasos de este proceso de transformación son:

1. En primer lugar se crea un diagrama de Contexto. Este diagrama tendrá como unidades los Actores externos, el único proceso que represente a todo el sistema y los flujos de información entre el sistema y los agentes externos.

2. Después se construye el diagrama DFD de nivel de subsistemas. Este DFD estará constituido por Agentes externos, Actividades complejas y almacenes donde se comparten documentos entre los procesos. Por cada Grafo BUC que tenga el Grafo de Casos añadimos una Actividad compleja a este diagrama. Si un Grafo BUC utiliza documentos creados en otro BUC, esto se muestra como una relación entre el proceso que representa al BUC y el almacén del BUC que genera el documento.

3. El tercer nivel esta formado por todos los diagramas que resultan de explosionar las Actividades complejas del diagrama de subsistemas. Los diagramas de este ultimo nivel tendrán como elementos todas las unidades de modelado que se obtengan de transformar los elementos de cada Grafo BUC.

La generación de los diagramas DFD de tercer nivel con los elementos de su Grafo BUC requiere establecer una correspondencia entre los elementos de este grafo y los elementos del diagrama de flujos de datos. La propuesta de correspondencia establecida es la siguiente:

- Los elementos del tipo *Puesto* del Grafo BUC no se utilizan porque los DFD no tiene unidades de modelado que representen este concepto.

- Los elementos del tipo *Transición* se convierten en procesos del DFD de ultimo nivel.

- Los elementos del tipo *Agente* se transforman en Agentes externos del diagrama DFD, a donde van a parar los documentos que emite el sistema. Esto sirve para los todos los niveles.

- Los elementos del tipo *Lugar* se convierten en Almacenes DFD y Flujos de datos, según se determina el siguiente algoritmo.

  Sean las funciones ${}^{o}T(L) = x$ y $T(L)^{o} = y$ que definen el numero de transiciones previas y el numero de transiciones siguientes de un lugar respectivamente, y sea las funciones ${}^{o}A(L) = x$ y $A(L)^{o} = y$ que definen el numero de agentes que generan este documento y el numero de agentes que lo reciben.

  1. Si un lugar tiene ${}^{o}T(L) = 0$, $T(L)^{o} = y \geq 1$ y ${}^{o}A(L) = x \geq 1$,es decir, documento que viene del exterior del sistema, entonces se transforma en un flujo de datos entrante.

  2. Si un lugar tiene ${}^{o}T(L) = x \geq 1$, $T(L)^{o} = 0$ y $A(L)^{o} = y \geq 1$, se transforma en un flujo de datos que sale del sistema hacia el exterior.

  3. Si un lugar tiene un T(L)=(x>=1,0) y A(L)=(x, 0), es un documento que queda en el sistema, entonces se transforma en un Almacén de datos.

  4. Si un lugar tiene ${}^{o}T(L) = x \geq 1$ y $T(L)^{o} = y \geq 1$, entonces se transforma en tantos flujos entre actividades como pares (x, y) se puedan dar. Además si tiene $A(L)^{o} = y \geq 1$, también se crean los flujos hacia el exterior.

6.7.5 Generar diagrama de Casos Uso (DUC)

Para obtener el diagrama de Caso de Uso a partir del Grafo de Caso Modular, debemos partir de la identificación entre los elementos de los distintos Grafos y las unidades de modelados del diagrama de casos de uso.

La actividad caso de uso se obtiene a partir de los Grafos UC del Grafo de Caso Modular. En este sentido asociamos los Grafos UC con casos de uso del diagrama Casos de Uso. Todas las transiciones o actividades del Grafo UC pasaran a formar parte de las subactividades en las se especifica este caso de uso.

- Los *Grafos UC* del Grafo de Casos Modular(GCM) se transforman en casos de uso del diagrama de Casos de Uso. Los casos de uso son unidades de modelados instancias de la metaclase Actividad.

- Los *Puestos* del Grafo Caso Modular, se transforman en Actores de caso de uso del diagrama de Casos de Uso.

- Los *Transiciones* de cada Grafo UC del Grafo Caso Modular, pasan a ser las subactividades de la plantilla en la que se especifica o se detalla el caso de uso obtenido de dicho Grafo UC.

- Los *Agentes* del Grafo Caso Modular, no se utilizan en este caso, porque el diagrama DUC no tiene unidades de modelado que representen este tipo de elemento.

6.7.6 Generar Diagrama de Escenarios (DES)

Se ha propuesto que el DES se obtenga a partir del Grafo de Casos después de la etapa de generación de Grafos BUC. Este diagrama estará constituido por Escenarios, Actores de escenarios y episodios de escenarios. Para la obtención del diagrama de escenario realizamos los siguientes pasos:

1. En primer lugar se crea el diagrama de Escenario a partir de la información del GC.

2. Cada Grafo BUC que tenga el Grafo de Casos se transforma en un Escenario al diagrama de Escenarios.

3. Cada Puesto que tenga el grafo de Casos, se transforma en un Actor de escenario del diagrama de Escenario.

4. Comprobamos si un Grafo BUC utiliza documentos generado por otro, en este caso se establece una dependencia de complementación entre el escenario obtenido del Grafo BUC que genera el documento y el escenario obtenido del BUC que consume este documento.

5. Las Actividades de cada Grafo BUC pasan a constituir los Episodios de escenario del Escenario que se obtiene de este Grafo BUC.

6.7.7 Generar Plantilla Especificación de Secuencia

Las plantillas de especificación de secuencia están asociadas a las actividades de cada tipo de diagrama. Por ello su obtención depende en parte del tipo de diagrama al que están asociados.

1. Para la obtención de las plantillas de las Actividades de los diagramas documentos-tareas, actividad UML, workflow, flujo de datos, se crea una nueva plantilla si la actividad en el diagrama inicial tenia una plantilla.

2. Para obtener las plantillas de los escenarios, partimos de que los escenarios se obtienen de los Grafos BUC. Entonces las actividades de este BUC se transforman episodios de escenario que se especifica en su plantilla. Dicho de otro modo: el BUC pasa a ser escenario y sus actividades los episodios de escenario en la plantilla.

3. Para obtener las plantillas de los casos de uso, partimos de que los casos de uso se obtienen de los grafos UC del GCM. Entonces las actividades de este UC se transforman etapas para la plantilla que especifica este caso de uso.

7 Conclusiones, discusión y trabajo futuro

Este trabajo se enmarca dentro del desarrollo de un entorno de reutilización de requisitos software. Para ello, y dada la complejidad asociada al desarrollo de dicho entorno, se planteó su desarrollo mediante el trabajo conjunto de varios proyectos. Los primeros trabajos consistieron en el desarrollo de un editor de diagramas de requisitos, que al tiempo que permitía la creación y modificación de diagramas de requisitos de varias técnicas de modelado distintas, integraba las distintas técnicas representando la información de requisitos a través de un metamodelo. Con el presente trabajo, se ha dotado al entorno de reutilización de una nueva funcionalidad, que consiste en la posibilidad de traducir diagramas de un tipo a otro. A partir de este momento los usuarios pueden transformar sus diagramas a otras representaciones de requisitos. Eso facilitará la visión del mismo proceso de negocio desde varias perspectivas, para su integración en el sistema.

La reutilización de requisitos debe tener en cuenta la complejidad de los mismos y la consecuente diversidad de técnicas de modelado de requisitos. Disponer de una herramienta que soporte la transformación una técnica de modelado en otra, resulta muy interesante y aporta un alto beneficio a la organización facilitando la comunicación entre los usuarios. Los requisitos actúan como medio de comunicación

y negociación entre las diferentes personas involucradas en el proceso de desarrollo del software. Por esta razón se utilizan diversas técnicas y formatos para mostrar distintas vistas de los sistemas. Estas vistas pueden ser traducidas con la funcionalidad que aporta este trabajo, en cualquier momento, a vistas de otra representación.

Después de haber puesto en marcha y observar los beneficios que aporta a la herramienta el servicio de transformación de requisitos, se plantean una serie de posibles mejoras a todo el proceso. Estas mejoras se refieren tanto a la ampliación de las funcionalidades de este proceso de transformación, como a su mejor diseño para una realización más eficiente.

Las tareas de transformación se han visto en ocasiones condicionadas por el nivel de soporte que permite la herramienta y el metamodelo de requisitos en que se basa el diseño de la herramienta. El hecho de que la herramienta no de soporte total y completo a los diagramas como se indica en el metamodelo y otros modelos teóricos, ha sido un aspecto que ha hecho agudizar el ingenio para acomodar el proceso a esas limitaciones. Por otro lado, en este trabajo no se ha tratado la transformación de las 36 posibilidades existentes. Se ha optado por transformar solo aquellos diagramas que representen flujos de actividades o trabajos. Eso hace que se reduzcan las transformaciones posibles a sólo diagramas documentos-tareas, Actividad UML y workflow, en general diagramas que representan flujos de tareas.

Las razones que han llevado a limitar este proceso a los diagramas indicados arriba son variadas y podemos citar:

- **La magnitud del trabajo:** El esfuerzo que conlleva realizar 36 transformaciones aconsejaba limitar este proceso, eligiendo un conjunto de diagramas que pudieran servir de origen de la transformación, para obtener el resto de diagramas.

- **Punto de vista semántica y estructural:** Tanto desde el punto de vista semántico como estructural, no es aconsejable la traducción de unos diagramas en diagramas de otro tipo especifico. La perdida de información, la falta de elementos para completar la especificación, etc pueden hacer inviable este proceso.

- **El modelo de los Grafos de Casos:** En este proceso de transformación, se ha utilizado los Grafos de Casos, grafos BUC y grafos UC como elementos centrales. Los grafos casos se utilizan, junto con la formalidad de la redes de Petri para modelar diagramas que representación flujos de trabajo.

Por ello es evidente que una mejora sobre este proceso de transformación, sería ampliarlo para que soporte la traducción de otros diagramas que no están orientados al modelo de flujos de trabajo, como casos de uso, escenarios, etc. Esta ampliación o mejora se puede enfocar desde dos perspectivas diferentes. Plantear otros modelos de transformación, utilizando otras estructuras y otros formalismo como las redes de Petri coloreadas u optar por, empleando el mismo modelo estructural de Grafos de Casos aquí utilizados, crear nuevos algoritmos que completen este proceso. Otra

ampliación podría consistir en aportar una interfaz visual a los casos de usos y casos de usos de negocio, que se obtienen de un diagrama de representación de requisitos. En este trabajo se ha diseñado un pequeño interfaz en forma de árbol para mostrar esa información.

En otro orden de cosas, se explico en apartados anteriores que el proceso de transformación aquí desarrollado, se basa en los modelos de Grafos de Casos , Casos de Uso de Negocio (BUC) y Casos de Uso(UC). Ello supone y conlleva a que algunas limitaciones de estos afecten a este proceso y a los diagramas.

- **Orden de las Actividades:** Las Actividades en los Grafos no están necesariamente ordenados como se ejecutaban en los diagramas originales, por ello el orden de las etapas en las plantillas de los casos de usos que se obtienen de esas actividades, no refleja el orden secuencial en que estas eran realizadas.

- **Ciclos en los diagramas:** Los caso de uso de negocio (BUC) y los UC están basados en el elemento Secuencia de Caso. En un diagrama con ciclos no se puede cumplir la definición de Secuencia de casos como camino va desde un documento de entrada hasta otro de salida, por ello se ha optado por no transformar aquellos diagramas que tengan ciclos o bucles de realimentan. Esto se puede mejorar eliminado ciclos en los diagramas.

- **Traducción recursiva:** Los diagramas pueden contener actividades o etapas que se especifican a su vez en otros diagramas. Aquí se ha optado por no seguir recursivamente el proceso de transformación a los diagramas que especifican las actividades complejas del diagrama que estamos traduciendo, limitándose únicamente a traducir el diagrama original. También se puede mejorar traduciendo recursivamente.

Referencias

[1] W.M.P. van der Aalst. Three Good reasons for Using a Petri-net-based Workflow Management System. In Proceedings of the International Working Conference on Information and Process Integration in Enterprises (IPIC'96), pages 179–201, 1996.

[2] G. Booch, J. Rumbaugh, and I. Jacobson. The Unified Modeling Language User Guide.Addison–Wesley, 1999.

[3] Ivar Jacobson, Martin Griss, and Patrik Jonsson. Software Reuse. Architecture, Process and Organization for Bussiness Success. ACM Press. Addison-Wesley, 1997.

[4] Oscar López. Reutilización de Requisitos para Incremento de la Calidad y Productividad en el Desarrollo de Especificaciones. PhD thesis, Universidad de Valladolid, Spain, 2003.

[5] Oscar López., Miguel A. Laguna, and Francisco J. García. Automatic Generation of Use Cases from Workflows: A Petri Net Based Approach. In Fundamental Approaches to Software Engineering, FASE 2002 LNCS 2306, pages 279–293, 2002.

[6] WfMC. Workflow management coalition terminology and glosary. Technical Report WFMC-TC-1011, Workflow Management Coalition, Brussels, http://www.aiim.org/wfmc/standards/docs/glossy3.pdf, 1996.