Revista EIA, ISSN 1794-1237 Número 8, p. 69-82. Diciembre 2007
Escuela de Ingeniería de Antioquia, Medellín (Colombia)

UN MÉTODO PARA LA TRAZABILIDAD DE REQUISITOS EN EL PROCESO UNIFICADO DE DESARROLLO

Marta Silvia Tabares*
Andrés Felipe Barrera**
Juan David Arroyave**
Juan Diego Pineda**

RESUMEN

El Proceso Unificado es un proceso de desarrollo adoptado por gran parte de las empresas desarrolladoras de *software*. Esto lleva a que atributos de calidad como la trazabilidad de requisitos deban estandarizarse para este proceso, con el fin de lograr los niveles de calidad exigidos por los clientes. Por lo general, los modelos de trazabilidad se proponen independientemente del proceso o métodos de desarrollo, y su definición y mantenimiento dependen de los criterios de calidad usados por los desarrolladores. En este artículo se presenta un método para la práctica de la trazabilidad en el Proceso Unificado de Desarrollo. El enfoque propone un flujo de trabajo para el control y soporte a la trazabilidad en las iteraciones del proceso. Dicho flujo establece un conjunto de acciones para generar modelos de trazabilidad que faciliten negociaciones oportunas con los participantes del proyecto.

PALABRAS CLAVE: trazabilidad de requisitos; ingeniería de *software;* trazabilidad; requisito; Proceso Unificado; UP.

* Ph. D(c) en Ingeniería de Sistemas, Universidad Nacional de Colombia. Docente del Área de Ingeniería de Software y Bases de Datos, y Directora del Grupo de Investigación de Ingeniería de Software (GIISEIA), Escuela de Ingeniería de Antioquia. pfmstabare@eia.edu.co

* Estudiantes de Ingeniería Informática, Escuela de Ingeniería de Antioquia. Auxiliares de Investigación del Área de Ingeniería de Software. ifanbar@eia.edu.co; ifjuar@eia.edu.co; ifjuanp@eia.edu.co

Artículo recibido: 24-IX-2007. Aprobado 13-XI-2007
Discusión abierta hasta junio de 2008

ABSTRACT

Unified Process is the development process adopted by many software development companies. Quality attributes, such as requirement traceability, must be standardized for this process, so that the system can achieve the quality demanded by customers. Commonly, traceability models are proposed independently of either the development process or the methodology that are followed, and its definition and maintenance depends on the quality criteria used by the developers. A traceability method for the Unified Development Process is presented in this paper. This approach proposes a workflow to control and support traceability throughout the iterations of the process. This workflow establishes a set of actions capable of generating traceability models that facilitate opportune agreement with the customers.

KEY WORDS: requirement traceability; software engineering; traceability; requirement; unified process; UP.

1. INTRODUCCIÓN

En la Ingeniería de Requisitos, es determinante lograr productos de *software* correctos, fiables y mantenibles. Por lo tanto, es necesario tener buenas técnicas para separar y especificar correctamente los requisitos, controlar su evolución y soportar los cambios. La trazabilidad es el mecanismo que permite lograr este resultado. Esta práctica es la base de la gestión de los requisitos, puesto que brinda la información necesaria para su control y soporte a lo largo del proceso de desarrollo de *software*. En otras palabras, posibilita la verificación de la transformación de los requisitos en elementos de modelo sucesores, así como el análisis y gestión del cambio en ellos, verificando su completitud y coherencia [15].

La trazabilidad permite que los participantes del proyecto logren propósitos claros dentro de la gestión del proceso. Además, proporciona elementos que ayudan a la comunicación entre los equipos de trabajo, ya que brinda mayor información para la comprensión del problema que se está tratando y apoya el control de las actividades y cambios en los productos de trabajo durante todo el ciclo de vida [13].

La mayoría de enfoques de trazabilidad se proponen de forma independiente al proceso o metodología de desarrollo. Sus modelos presentan los elementos básicos que participan en la trazabilidad y la forma como los vínculos de trazado se deben controlar [6, 9-12, 14], Sin embargo, de aquí surgen dos interrogantes fundamentales: ¿Es posible acercar el uso de dichos modelos a los procesos de desarrollo usados cotidianamente por las empresas de desarrollo? y ¿qué proporcionan las metodologías de desarrollo y, en particular, el proceso unificado de desarrollo para lograr que dicha práctica no se convierta en un elemento problemático y costoso, sino en una verdadera herramienta que proporcione un seguimiento certero para minimizar riesgos y asegurar productos de calidad?

Para dar respuesta a los anteriores interrogantes, en este artículo se presenta un método para la trazabilidad de requisitos en el proceso unificado de desarrollo. Este método surge como resultado de la investigación realizada en empresas de desarrollo de *software* acerca de la adopción y uso de la trazabilidad desde el proceso unificado. El método es un flujo de trabajo para el control y soporte de la trazabilidad en las iteraciones del proceso. Dicho flujo establece un conjunto de acciones para generar modelos de trazabilidad que faciliten negociaciones oportunas con los participantes del proyecto. Los modelos conservan la correlación entre los artefactos *software,* pero su definición y mantenimiento dependerá de los criterios de calidad usados por los desarrolladores.

La estructura de este artículo es como sigue. En la sección 2 se presentan las características generales del proceso unificado y los elementos que provee para hacer la práctica de la trazabilidad. En la sección 3 se trata el flujo de control y soporte a la trazabilidad para el Proceso Unificado; en la sección 4, los trabajos relacionados. Finalmente, en la sección 5 se concluye y se propone el trabajo futuro.

2. EL PROCESO UNIFICADO Y LA TRAZABILIDAD

En esta sección, se presentan las características más importantes que proveen el Proceso Unificado [4] y UML *(Unified Modeling Language)* [16] para realizar la trazabilidad de requisitos.

2.1 El Proceso Unificado y sus elementos para el trazado

Los métodos de desarrollo de *software* son variados y tienen características propias que los hacen aptos y específicos para las necesidades de los desarrolladores. Sin embargo, independientemente de cuál se utilice y los productos de trabajo *(workproducts)* o artefactos que de él se deriven, los elementos que apoyan el proceso de desarrollo son susceptibles de ser trazados. El grado de trazabilidad que se puede lograr depende de factores tales como la cantidad y calidad de información que proporcionan los elementos de modelo y las necesidades de los participantes del proyecto en la gestión que se deriva de la traza.

El Proceso Unificado (UP), conocido comercialmente como RUP –*Rational Unified Process*–, constituye un marco de trabajo o metodología estándar de desarrollo ampliamente usado y difundido en las empresas de desarrollo, con características adaptables a las organizaciones y proyectos de *software*. Los productos de trabajo o elementos de modelo que se elaboran durante el proceso se representan en UML. El proceso de RUP es iterativo e incremental, dirigido por requisitos o casos de uso, centrado en la arquitectura y enfocado a la gestión del riesgo [2].

Al tratarse de un marco de trabajo extensible, procura la adaptación y define cuatro fases en la etapa de desarrollo (inicio, elaboración, construcción y transición) que a su vez se cruzan con una serie de disciplinas (requisitos, análisis, diseño, implementación y pruebas), como se muestra en la figura 1.

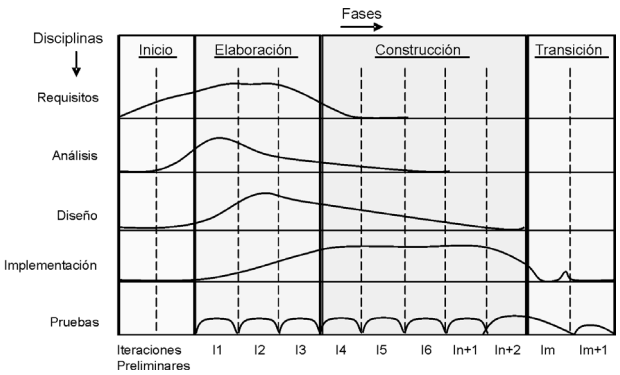

Figura 1. Una adaptación del diagrama del Proceso Unificado presentado en [2]

Según la fase en que se encuentre el proyecto, algunas disciplinas tienen mayor incidencia que otras. El desarrollo iterativo e incremental es versátil y elimina muchos de los errores que otros procesos de desarrollo dejan en el tiempo. Permite identificar y procesar un conjunto de artefactos por fase que se liberan como resultado de una iteración. Así, los participantes de una fase podrán trazar los documentos y modelos de forma sucesiva, ya que el proceso provee liberaciones de completitud creciente por iteración (figura 2).

Para determinar el alcance de la práctica de la trazabilidad con el proceso unificado de desarrollo es necesario conocer: 1) los objetivos que se logran y los productos de trabajo que se deben elaborar en cada una de las fases; 2) la forma como operan los flujos de trabajo de cada disciplina por iteración.

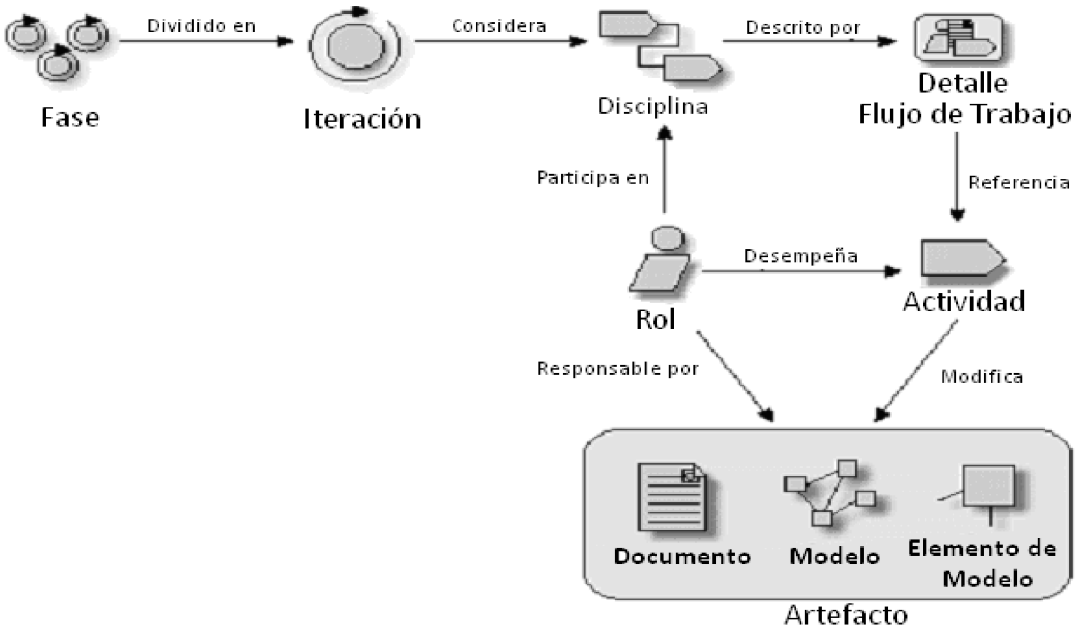

Figura 2. Detalle del proceso iterativo e incremental [8]

- ## Metas y productos de trabajo por fase

En la tabla 1 se ilustran, de forma general, los objetivos de cada fase y algunos documentos/modelos relevantes mediante los cuales se especifican y representan los productos de trabajo que se pueden liberar o entregar en cada iteración.

Por lo tanto, en RUP es posible manejar dos tipos de trazabilidad entre los productos de trabajo o artefactos elaborados por fase o iteración: i) trazabilidad entre los elementos de modelo UML, por medio de las relaciones de trazado (<<trace>>, <<realize>>, etc.); ii) trazabilidad por versionamiento de documentos, tales como visión, evaluación del riesgo, plan de pruebas, gestión y plan del proyecto, etc.

- ## Los flujos de trabajo

Los flujos de trabajo *(workflows)* proveen una secuencia de actividades que permiten lograr metas concretas en cada una de las disciplinas del proceso. Deben estar muy bien definidos con su propósito, actores responsables, tareas y entregables, para que así sea más uniforme y organizado el desarrollo de aplicaciones robustas y complejas. Cada flujo de trabajo cubre una iteración desde el punto de vista de cada disciplina [7].

Una iteración se puede entender como la ejecución de las disciplinas definidas en el proceso de desarrollo, manteniendo el objetivo de cada fase y dejando como resultado un incremento sobre los modelos construidos en las fases anteriores [4]. En otras palabras, cada iteración es una secuencia distinta de actividades enmarcadas en un lapso que tiene como resultado una entrega (interna o externa) de un producto ejecutable [8]. Cada iteración se define durante el proceso, es decir, nunca se deben planear todas las iteraciones desde el principio. Los miembros del grupo de trabajo que tengan mayor experiencia deciden qué actividades de las disciplinas involucradas se deben desarrollar en cada iteración. En la figura 3 se ilustra la participación de

Tabla 1. Objetivos generales y algunos documentos relevantes por fase

Fase	Objetivos generales de la fase	Documento/Modelo Fuente	Documento/Modelo Producto de trabajo
Inicio	Tomar decisiones tecnológicas Modelar el negocio Capturar requisitos Identificar el riesgo crítico	Modelo del negocio	Documento de visión Documento descriptivo del negocio Documento de evaluación del riesgo Modelo de requisitos funcionales Modelo de casos de uso Modelo del dominio Prototipos desechables Arquitectura candidata
Elaboración	Crear arquitectura ejecutable Evaluar el riesgo Especificar los atributos de calidad Especificar, refinar casos de uso Crear el plan detallado de la fase de construcción Analizar el costo-beneficio del sistema solución	Documento de visión Documento de evaluación del riesgo Modelo de requisitos Modelo de casos de uso Arquitectura candidata	Documento de visión refinado Documento de evaluación del riesgo refinado Modelo de requisitos no funcionales Modelo de casos de uso arquitectónicamente significativos Modelo de clases Modelo de componentes Esquema de la base de datos Prototipos definitivos Arquitectura ejecutable Modelo de pruebas
Construcción	Desarrollar los productos para ser entregados Hacer la integración de subsistemas Realizar pruebas de unidad Realizar pruebas de integración	Arquitectura ejecutable refinada Modelo de componentes Esquema de la base de datos	Modelo de despliegue Programas de *software* de la solución Resultados de pruebas de unidad
Transición	Ejecutar pruebas operativas del sistema Corregir errores de construcción Hacer pruebas para entrega de productos de trabajo	Modelo de despliegue Modelo de componentes refinado Esquema de la base de datos refinado Programas de software para ser entregados	Resultado de pruebas funcionales y de la capacidad operativa del sistema Manuales de usuario Manuales de operación del sistema Documento con plan de implantación

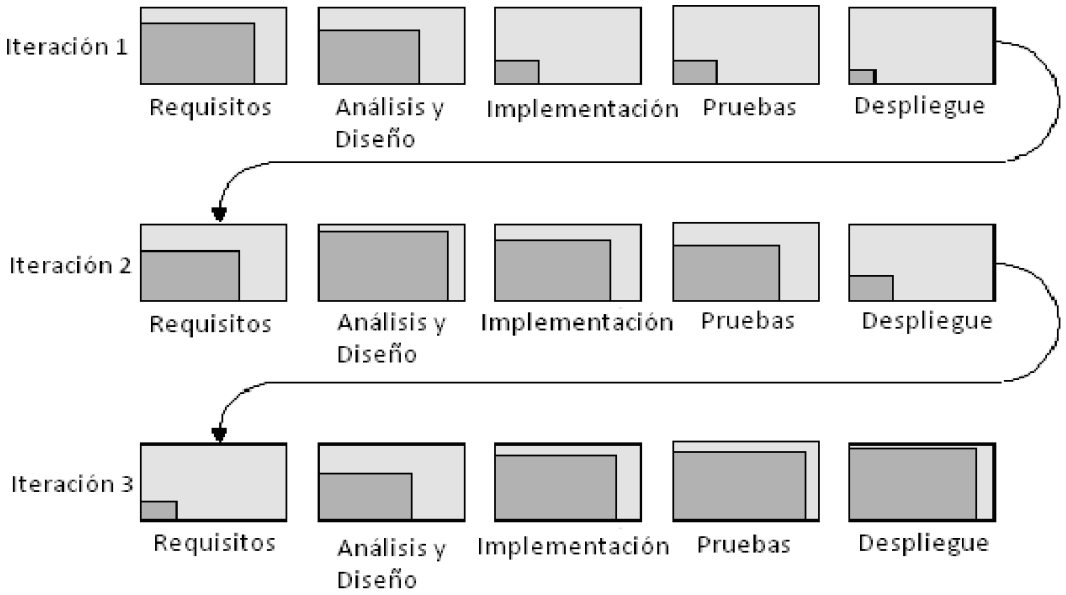

Figura 3. Ciclo de vida de una iteración [8]

las disciplinas iteración tras iteración. En un flujo de trabajo, la trazabilidad permite hacer el seguimiento de los elementos del modelo que evolucionan en cada iteración.

2.2 Relaciones de trazado en UML y modelos de trazabilidad

UML dispone de dos tipos de relaciones para realizar la trazabilidad: Abstracción *(Abstraction)* y Realización *(Realization)*. La relación de *Abstracción* "*relaciona dos o más elementos o conjunto de elementos que representan el mismo concepto en diferentes niveles de abstracción o desde diferentes puntos de vista. En el metamodelo, una Abstracción es una Dependencia en la cual hay una correlación entre el proveedor y el cliente*" [16]. Hay cuatro dependencias de abstracción: <<trace>>, <<substitute>>, <<refine>>, y <<derive>> [2]. Comúnmente, se usan las siguientes relaciones:

<<trace>>: relación donde el proveedor y el cliente representan el mismo concepto en diferentes modelos. Por ejemplo, en la figura 4 se ilustran tres casos: a) un componente o subsistema del diseño traza un paquete en el análisis; b) la clase de diseño y la interfaz trazan la clase del análisis; c) una realización de un caso de uso del diseño traza una realización de un caso de uso del análisis[1].

<<refine>>: relación usada entre elementos del mismo modelo. Por ejemplo, en un mismo modelo se puede tener dos versiones de la misma clase en el modelo de clases.

La relación de *Realización* "*es una relación de abstracción especializada entre dos conjuntos de elementos de modelo, uno representa una especificación (el proveedor) y el otro representa una implementación del último (el cliente). La realización se puede usar para modelar paso a paso refinamiento, optimizaciones,*

[1] Análisis y diseño refieren a niveles de abstracción diferentes.

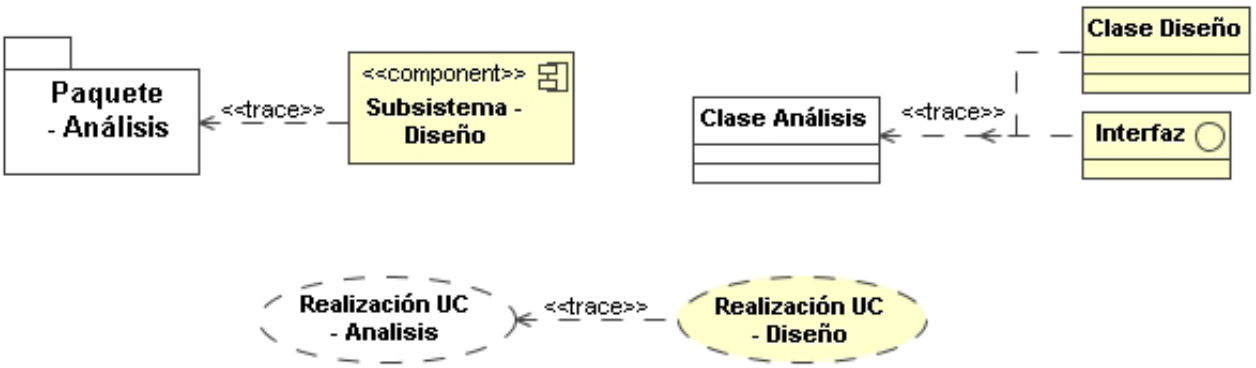

Figura 4. Relaciones entre artefactos del análisis y el diseño en UML [2]

*transformaci*ones, plantillas, síntesis de modelo, composición de marcos de trabajo, etc." [16]. En la figura 5 se ilustra un ejemplo donde el elemento de modelo Caso de Uso realiza los elementos de modelo Requisito 1 y Requisito 2. Otros elementos estereotipados que dispone UML 2.0 para el trazado son: <<call>>, <<send> e <<instantiate>>. Las herramientas CASE para el modelado UML disponen de estos y de otras relaciones de trazado para soportar la trazabilidad.

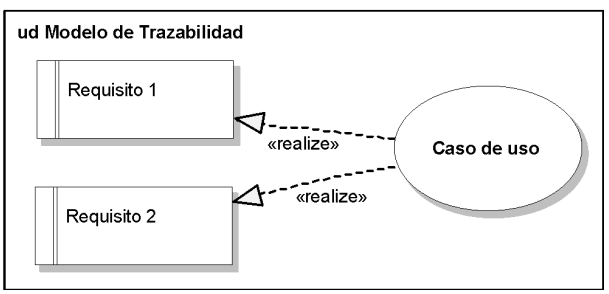

Figura 5. Ejemplo del uso de la relación Realization con el estereotipo <<realize>>

Los modelos de trazabilidad soportan la correlación entre elementos de modelo. En la literatura es posible encontrar que "Modelo de Trazabilidad" (Traceability Model) se refiere al metamodelo que provee un conjunto de elementos abstractos diseñados para establecer criterios acerca de relaciones y elementos que registran el trazado [3, 11, 14].

Para el enfoque de este artículo, los modelos de trazabilidad son aquellos que los desarrolladores crean para controlar la evolución y cambios de los requisitos; dependerán de los modelos de desarrollo (requisitos, casos de uso, clases, etc.) que sean construidos por los desarrolladores. Por lo general, combinan diferentes relaciones de trazabilidad estereotipadas de Abstracción y Realización para correlacionar los elementos de modelo. Estos elementos se construyen con base en las decisiones de calidad que tomen los grupos de trabajo. Además, se complementan con las matrices de trazabilidad que proveen las herramientas de modelado (tabla 2).

Tabla 2. Requisitos y casos de uso

Casos de uso \ Requisitos	CU1	CU2	CU3	CU4
Requisito 1	✓			
Requisito 2		✓		
….			✓	
Requisito n				✓

Cuando se construye un modelo de trazabilidad, las herramientas permiten usar libremente las relaciones de trazabilidad; lo importante es que los desarrolladores hagan buen uso de ellas. El flujo de control y el soporte de trazabilidad que este enfoque proponen ayudarán a estandarizar la realización de dichos modelos.

3. LA TRAZABILIDAD COMO SOPORTE A LOS FLUJOS DE TRABAJO

Los modelos de trazabilidad reconocen tres elementos básicos: los participantes *(stakeholders)*, las fuentes (documentos y modelos) y los objetos o artefactos para ser trazados. Estos elementos y su evolución se deben identificar explícitamente en cada flujo de trabajo para así controlar y soportar el trazado en las fases del proceso. Por lo tanto, es necesario que un flujo de control de la trazabilidad apoye los flujos de trabajo en cada iteración. Los modelos de trazabilidad se deben generar por iteración para que los grupos de trabajo tomen decisiones acerca del alcance del desarrollo y del impacto del cambio. Así, se realizarán negociaciones oportunas con los participantes del proyecto. Además, se proveerán elementos para verificar la consistencia y la completitud de los modelos de la solución.

3.1 El flujo para el control y soporte de la trazabilidad

En este artículo se propone un flujo de trazabilidad orientado a estandarizar el control y soporte de esta práctica en el proceso de desarrollo (figura 6).

En la primera iteración, normalmente no existen los modelos de trazabilidad. Estos se deberán elaborar realizando las siguientes acciones:

a. Establecer criterios para el modelo de trazabilidad. Se refiere a la definición de criterios para determinar qué participantes, modelos/documentos fuente y elementos de modelo participarán en el trazado. Además, se establecen criterios de control del impacto del cambio, tales como operaciones de trazado y método de análisis costo-beneficio. Estos criterios establecen la forma como los participantes elaborarán e interpretarán los modelos de trazabilidad. Así los modelos de trazabilidad lograrán ser estándar para todos los proyectos en una empresa de desarrollo, pero de igual forma podrán variar de acuerdo con el tipo de proyecto o la arquitectura de desarrollo utilizada.

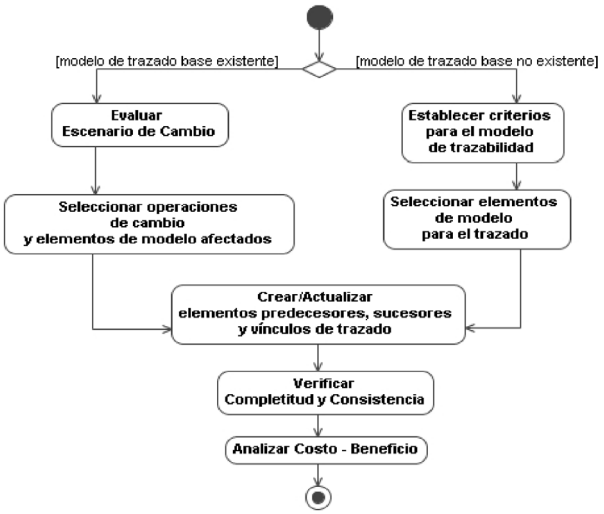

Figura 6. Flujo para el control y soporte de la trazabilidad

b. Seleccionar elementos de modelo para el trazado. Se refiere a la clasificación de los elementos de modelo proporcionados por el flujo de trabajo en una iteración determinada. Aunque los casos de uso son el centro del desarrollo y de la toma de decisiones, es importante determinar qué otros elementos se trazarán conjuntamente con ellos en el modelo de trazabilidad[2].

c. Crear/Actualizar elementos predecesores, sucesores y vínculos de trazado. Se refiere a la creación o actualización de los modelos de trazabilidad. Establece el orden de los elementos (predecesores-sucesores) que serán trazados a partir de los elementos de modelo involucrados en la traza y de los vínculos que permitirán trazarlos. En la figura 7 se ilustra un modelo básico de trazado generado en una primera iteración en el flujo de trabajo de la disciplina de requisitos.

[2] Los elementos de configuración a los cuales se refiere la gestión del cambio [13] son elementos de grano grueso a partir de los cuales se puede determinar qué elementos de modelo serán trazados.

d. Verificar completitud y consistencia. Se refiere a que los modelos de trazabilidad son la base para reconocer incompletitud e inconsistencia en los modelos de desarrollo. Algunas inconsistencias pueden ser: más de un caso de uso realice (<<realize>>) al mismo requisito, un prototipo no realice a ningún caso de uso, un requisito no sea trazado (<<trace>>) a ningún caso de uso, una interfaz no trace ninguna clase del análisis, etc. Realizar verificaciones de este tipo puede disminuir conflictos entre los grupos de trabajo, compensando los problemas con buenas prácticas de gestión del cambio.

En las siguientes iteraciones, los modelos de trazabilidad se actualizarán por los grupos de trabajo con base en decisiones técnicas o cambios solicitados por los usuarios durante el desarrollo. Para lograr esto se deben realizar las siguientes acciones.

e. Evaluar el escenario de cambio. Se refiere a la evaluación del impacto de los cambios solicitados por los participantes. Generalmente, las empresas de desarrollo definen su proceso de gestión del cambio y establecen plantillas específicas para formalizar los escenarios de cambio. En ellos, se debe registrar información referente a la iteración, disciplinas afectadas, participantes del cambio (cliente y grupo de desarrolladores), contexto funcional y casos de uso afectados, los riesgos asociados a los cambios y elementos de configuración afectados, tales como documentos y modelos de desarrollo. A partir de los modelos de trazabilidad existentes, los desarrolladores evaluarán el impacto del cambio (costo-esfuerzo-beneficio) y podrán dar un diagnóstico o presupuesto de las actividades que deberán realizarse sobre modelos de desarrollo, documentos y productos que se entregarán.

f. Identificar operaciones de cambio y elementos de modelo afectados. Se refiere al reconocimiento de las operaciones de cambio que se deben aplicar a los elementos de modelo identificados. Básicamente, se dan tres operaciones: crear nuevos elementos, modificar los existentes o eliminarlos[3]. Un cambio puede propagarse por muchos otros elementos de modelo (identificados de manera general en la acción anterior). La propagación es una "reacción en cadena" que se debe controlar y verificar en consistencia y completitud a partir del modelo de trazabilidad existente.

En la figura 8, se muestra una nueva versión del modelo de trazabilidad de la figura 7. Aquí, la relación de <<*trace*>> se usa para relacionar los cambios con los elementos de modelo afectados. Tanto el *Requisito 3* (proveedor) como el *Cambio 1* (cliente) representan el mismo concepto en diferentes modelos. El *"Cambio 1"* crea el Requisito 3, el cual a su vez realiza la *"Necesidad 2"*. La propagación de este cambio en otros elementos de

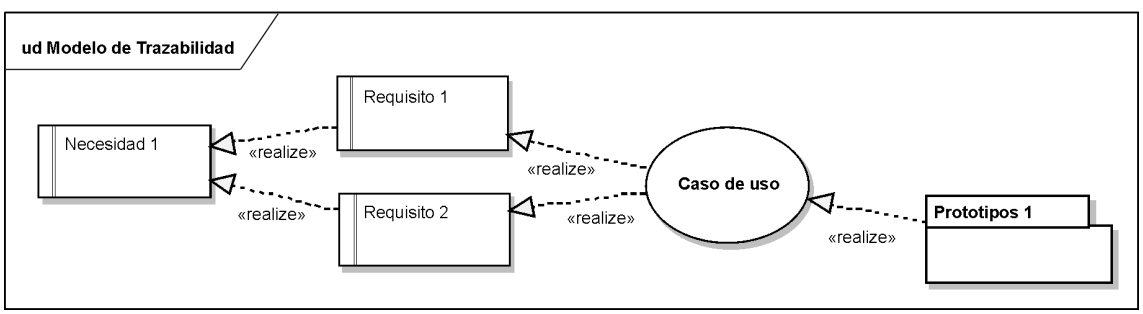

Figura 7. Un modelo de trazabilidad en una primera iteración

[3] En [6] se presentan otros tipos de operaciones de cambio como una especialización de éstas.

modelo provoca la creación de nuevos cambios sobre dichos elementos.

g. **Analizar costo-beneficio.** Se refiere a la estimación del costo y el esfuerzo que requieren los cambios solicitados por los participantes. Con base en el modelo de trazabilidad, se calcula el esfuerzo que implica realizar los cambios. Para los modelos de trazabilidad centrados en casos de uso, comúnmente se aplica el método basado en puntos de casos de uso [5]. Para modelos de trazabilidad que sólo trazan elementos del diseño, el cálculo del esfuerzo se hace desde el número de vínculos de trazado, de clases, de componentes y de métodos. Estos artefactos son situados y contabilizados por el nivel de granularidad a partir del cual se calcula el esfuerzo [1].

3.2 Trazabilidad en el flujo de requisitos

Las actividades que se ejecuten en los flujos de trabajo dependerán de la iteración que se esté llevando a cabo en el proceso de desarrollo. A continuación se analiza la forma como el flujo de control y el soporte de trazabilidad pueden apoyar el flujo de requisitos.

En la figura 9, se ilustra el flujo de trabajo de la disciplina de Requisitos y se incorpora la actividad de *"Control y Soporte de Trazabilidad"*. Además, se reconocen los participantes, los documentos/modelos fuente y los productos de trabajo involucrados en el trazado. Inicialmente, es importante conocer, por cada iteración, qué acciones se van a ejecutar en cada flujo de trabajo (decisión del grupo de trabajo) y qué objetivo del sistema y requisitos se gestionarán. Así, se determinan el alcance y los modelos de trazabilidad que se generarán.

Para la trazabilidad, toda acción que pueda generar o alterar un elemento de modelo o documento debe estar siempre presente en el flujo para facilitar el control del trazado. Por esta razón, las acciones *"Refinar la definición del sistema"* y *"Administrar el cambio en los requisitos"* se deben considerar.

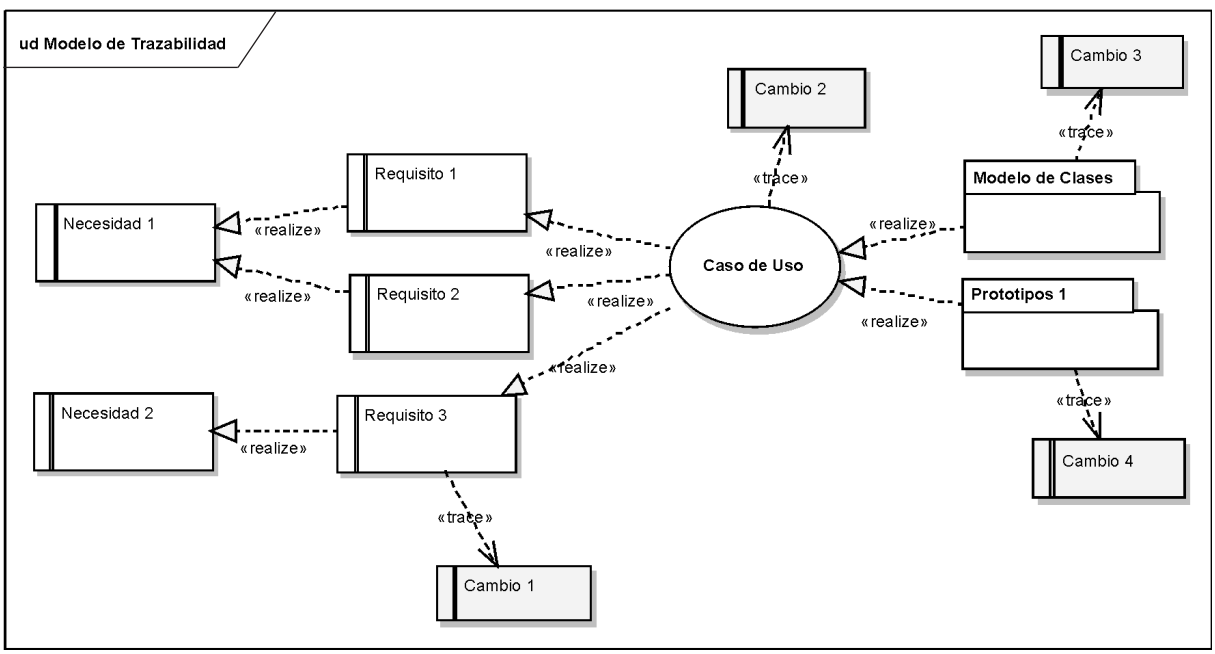

Figura 8. Un ejemplo de modelo de trazabilidad que incluye cambios (generado en el flujo de requisitos)

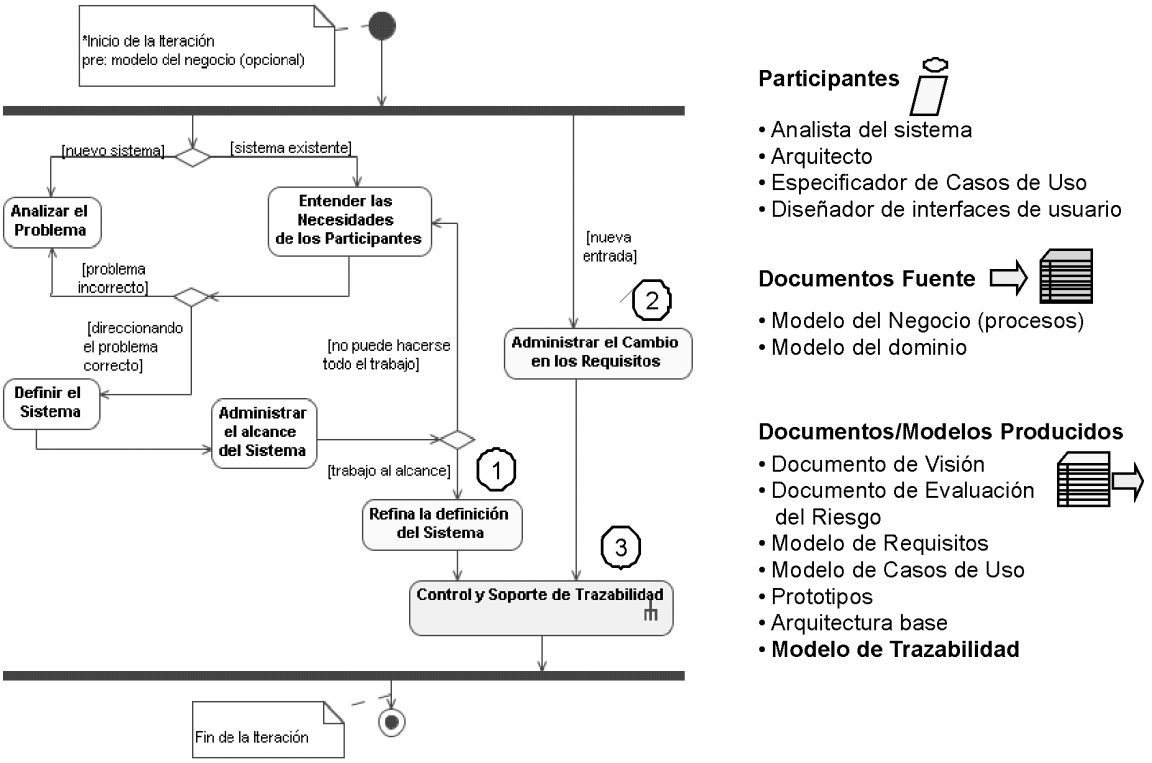

Figura 9. Flujo de trabajo de la disciplina de Requisitos con la acción de trazabilidad (una adaptación de [7] para este enfoque)

Las acciones marcadas como (1) y (2) determinan los elementos de modelo para la acción (3). La primera acción provee los elementos de modelo para crear o refinar el modelo de trazabilidad durante el desarrollo (depende de la iteración). En los productos de trabajo de esta disciplina se incluye el modelo de trazabilidad que será determinante para posteriores fases del ciclo de vida (p. ej., el modelo de la figura 8). En la tabla 3, se muestran algunos de los elementos y relaciones de trazado que se deben controlar durante la ejecución del flujo de trazabilidad.

En este enfoque, el elemento o artefacto de *"Requisitos"* se refiere tanto a requisitos funcionales como no funcionales, pero de igual forma se podrían

Tabla 3. Elementos de trazado en el flujo de Requisitos

Elemento origen	Relación de trazado	Elemento destino
Requisitos	<<realize>>	Necesidades
Casos de uso	<<realize>>	Requisitos
Prototipos	<<realize>>	Casos de Uso
Modelo de clases	<<realize>>	Modelo del Dominio
Arquitectura candidata	<<trace>>	Modelo de Clases

correlacionar con otros elementos independientemente. Dicha decisión dependerá de la estrategia de especificación y modelado usada por el grupo de desarrollo. Algunas buenas prácticas orientan la agrupación de requisitos de acuerdo con los intereses u objetivos del negocio. Así, los casos de uso se separan o agrupan en paquetes funcionales que representan dicho interés.

Al refinar el sistema, un nuevo requisito, caso de uso u otro elemento de modelo se puede crear, modificar o eliminar en un modelo de trazabilidad. Todo cambio debe partir de los requisitos y los casos de uso, pero muchas veces los desarrolladores evitan el flujo de requisitos, y los cambios afectan directamente la arquitectura y elementos de diseño, como los componentes y la base de datos.

En los flujos de trabajo de las otras disciplinas de RUP, los modelos de trazabilidad cambian un poco y es posible que se utilicen otros tipos de relaciones de trazado entre elementos de modelo. Por ejemplo, en la disciplina de análisis y diseño se generan elementos de modelo tales como paquetes y *templates* y las relaciones entre ellos (<<import>>, <<merge>>, etc.) pueden ser usadas como relaciones de trazado. Además, la relación <<*trace*>> se usará directamente para marcar el rastro entre paquetes de clases o colaboraciones y subsistemas de diseño de componentes.

4. TRABAJOS RELACIONADOS

Los trabajos relacionados con el análisis experimental que se presenta en este artículo no son muchos. Entre ellos, resalta el trabajo de Letelier en [11], que presenta un marco de trabajo para la trazabilidad de requisitos en proyectos cuyos modelos estén representados en UML. En este enfoque, hace una configuración de trazabilidad para RUP y aplica cuatro tareas: 1) selecciona los tipos de artefactos que son de interés para la trazabilidad; 2) define relaciones de agregación entre artefactos; 3) establece tipos de vínculos de trazado que son de interés para el proyecto; 4) define criterios para derivar implícitamente vínculos de trazabilidad y su tipo. En relación con esta propuesta, algunas tareas se pueden ver semejantes a las acciones del flujo de trazabilidad. Sin embargo, la diferencia radica en que Letelier desarrolla dichas tareas con base en el metamodelo de trazabilidad de su marco de trabajo, que provee a una semántica de trazabilidad particular. Es decir, selecciona los elementos de modelo que RUP presenta para el trazado y los asocia directamente a clases y relaciones de trazado de su marco de trabajo.

Por el contrario, en el enfoque de este artículo, se usan de forma simple los elementos RUP representados en UML para guiar la elaboración de los modelos de trazabilidad diseñados por el grupo de trabajo. De igual forma, los documentos, como el de visión, se pueden representar en una clase estereotipada en cualquier fase del ciclo de vida. Además, Letelier no presenta un control del trazado explícito a partir de modelos de trazabilidad para verificar completitud y consistencia ni tampoco la factibilidad del impacto de los cambio.

5. CONCLUSIONES Y TRABAJO FUTURO

Formalizar la práctica de la trazabilidad en las empresas de desarrollo es una necesidad sentida. El flujo de control y soporte de trazabilidad propuesto se orienta a estandarizar y automatizar los modelos de trazabilidad. Estos se deben establecer para que los grupos de desarrollo puedan medir fácilmente el impacto de los cambios generados durante el proceso de desarrollo.

Una vez se conoce cómo se puede controlar la práctica de la trazabilidad desde el proceso unificado, es importante empezar una prueba piloto en una empresa de desarrollo. Este flujo está orientado a que los grupos de trabajo puedan establecer medidas o criterios acerca de factores tales como la continua

demanda de cambios por parte de los usuarios, el grado de entendimiento del problema por parte de los desarrolladores y el nivel de intervención de los arquitectos en esta práctica desde etapas tempranas de desarrollo, entre otros.

Las empresas de desarrollo establecen la práctica de la trazabilidad como un elemento base de la calidad del proceso de desarrollo. Sin embargo, su realización durante el proceso se deja, la mayoría de veces, a criterio de los analistas funcionales (líderes) que se apoyan en matrices de trazabilidad ofrecidas por las herramientas. Además, las prácticas ágiles de desarrollo la desechan como una actividad básica de proceso. Para mejorar esto, la alternativa más importante que apoya la trazabilidad es la transformación de modelos. Poder automatizar la evolución de los requisitos y los artefactos en diferentes niveles de granularidad (o fases) hace que la trazabilidad esté totalmente orientada a la propagación de los cambios tanto hacia adelante *(forward)* como hacia atrás *(backward)* en el proceso de desarrollo, a la evaluación del impacto del cambio y a la verificación de la completitud y consistencia de los modelos de desarrollo.

Como trabajo futuro, el grupo de investigación está realizando proyectos en tres frentes importantes. Uno, establecer el grado de la correlación que puede ocurrir entre los modelos de trazabilidad generados en los flujos de requisitos y los generados en los flujos de las etapas de análisis y diseño. Dos, realizar un análisis de los costos y beneficios que implica realizar la práctica de la trazabilidad usando el proceso unificado y otras metodologías de desarrollo. El tercer frente, y más importante, es obtener un patrón de transformación dirigido a generar modelo de trazabilidad con características de propagación del cambio en diferentes niveles de abstracción, para verificar consistencia y completitud de los modelos de desarrollo.

AGRADECIMIENTOS

Este artículo corresponde a producción intelectual generada para el proyecto de investigación "Análisis de la práctica de la trazabilidad de asuntos transversales en una empresa de desarrollo de *software*" patrocinado por la Escuela de Ingeniería de Antioquia. Este proyecto se realiza en el marco de desarrollo de la tesis doctoral del primer autor, que a su vez es apoyado por los coautores como auxiliares de investigación. Además agradecemos a las empresas de desarrollo de *software* de Medellín (Colombia) que nos facilitaron los recursos para hacer el análisis experimental (nos reservamos el nombre por acuerdos de confidencialidad).

BIBLIOGRAFÍA

[1] Ahn, S. and Chong, K. A feature-oriented requirements tracing method: A study of cost-benefit analysis. Proc. in International Conference on Hybrid Information Technology (ICHIT'06) 2006.

[2] Arlow, J., and Neustad, I. UML 2 and the Unified Process: Practical Object-Oriented Analysis and Design (2 ed.). Addison-Wesley Object Technology Series. 2005.

[3] Bondé, L.; Boulet, P. and Dekeyser, J.-L., Traceability and interoperability at different levels of abstraction in model transformations, in Forum on Specification and Design Languages, FDL'05, Lausanne, Switzerland, Sep. 2005.

[4] Booch, G.; Rumbaugh, J. y Jacobson, I. El proceso unificado de desarrollo de software. Pearson Educación, Madrid, 2000.

[5] Carroll, E. R. Estimating software based on use case points. Proc. in OOPSLA'05, Oct. 2005, USA. p. 257-265.

[6] Cleland-Huang, J.; Chang, C. K. and Christensen, M. Event-based traceability for managing evolutionary change, Software Engineering, IEEE Transactions on Volume 29 (9): 796-810, Sept. 2003

[7] Crain A. RUP iteration planning. The Rational Edge: e-zine for the Rational Community. July 2004.

[8] Eeles, P.; Kozaczynski, W. and Houston, K. Building J2EE Applications with the Rational Unified Process. Addison-Wesley, 2003.

[9] Egyed A. A scenario-driven approach to trace dependency analysis. IEEE Trans. Software Eng. 29(2): 116-132, 2003.

[10] Gotel, O. and Finkelstein, A. Extended requirements traceability: results of an industrial case study. In Proceedings of 3rd International Symposium on

Requirements Engineering (RE '97). IEEE Computer Society Press, p. 169-178.

[11] Letelier, P. A framework for requirements traceability in UML-based Projects. 1st International Workshop on Traceability in Emerging Forms of Software Engineering, In conjunction with the 17th IEEE International Conference on Automated Software Engineering, U.K., Sep. 2002.

[12] Lindvall, M. A study of traceability in object-oriented systems development. Licentiate Thesis 462, Dept. of Computer and Information Science, Linkping University, Sweden, 1994.

[13] Pressman R. S., Ingeniería del *software:* Un enfoque práctico. (6 ed.). McGraw-Hill. 2006.

[14] Ramesh B. and Jarke M. Towards reference models for requirements traceability. IEEE Transactions on Software Engineering, Vol. 27, No. 1, January 2001.

[15] Tabares M. S.; Arango, F. and Anaya R. Una revisión de modelos y semánticas para la trazabilidad de requisitos. Revista EIA, número 6, diciembre 2006, p. 33-42.

[16] UML-OMG. Unified Modeling Language: Superstructure. V. 2.0. formal/05-07-04.