Revista EIA, ISSN 1794-1237 Número 6, p. 33-42. Diciembre 2006
Escuela de Ingeniería de Antioquia, Medellín (Colombia)

UNA REVISIÓN DE MODELOS Y SEMÁNTICAS PARA LA TRAZABILIDAD DE REQUISITOS

Marta Silvia Tabares*
Fernando Arango**
Raquel Anaya***

RESUMEN

Realizar el seguimiento a los requisitos a lo largo del proceso de desarrollo de software no es tarea fácil. Todo artefacto de software cambia en el tiempo por la evolución en las necesidades de los usuarios. Para minimizar el impacto causado por dicha evolución, la práctica de la trazabilidad ha sido estudiada e implementada con diferentes modelos y técnicas que permiten lograr mayor calidad en los productos de software. En este artículo se muestra una revisión conceptual de algunas de las más representativas aproximaciones de la práctica de la trazabilidad de requisitos y se analiza la forma como contribuyen a la verificación de la consistencia y completitud de los modelos de desarrollo.

PALABRAS CLAVE: ingeniería de software; trazabilidad de requisitos; trazabilidad; requisito.

ABSTRACT

To make the tracking of requirements during software development process is not easy task. The software artifacts change throughout the time because the user requirements evolve. To minimize the caused impact by this evolution, the traceability has been studied and implemented through different models and techniques that allow achieve more quality in the software products. In this paper we show a survey about some requirement traceability approaches and analyze the way how they contribute to both consistency and completeness verifications of the development models.

KEY WORDS: software engineering; requirement traceability; traceability; requirement.

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

** Doctor en Ingeniería de la Programación e Inteligencia Artificial, Universidad Politécnica de Valencia, España. Docente de la Escuela de Sistemas e Informática, Universidad Nacional de Colombia. farango@unal.edu.co

*** Doctor en Ingeniería de la Programación e Inteligencia Artificial, Universidad Politécnica de Valencia, España. Escuela de Ingeniería, Departamento de Sistemas, Universidad EAFIT. ranaya@eafit.edu.co

Artículo recibido 26-IX-2006. Aprobado 14-XI-2006
Discusión abierta hasta junio de 2007

1. INTRODUCCIÓN

La trazabilidad en la Ingeniería de Software es una práctica de control que ayuda a obtener el producto en el dominio de la solución lo más exacto y fiable posible a las necesidades expresadas por el cliente en el dominio del problema. A diferencia de productos en otras ramas de la ingeniería, un producto de software es maleable y pueden cambiar su estructura y composición en espacio y tiempo durante el proceso de desarrollo de software y es aquí donde la trazabilidad adquiere gran importancia. La trazabilidad se refiere a la *"capacidad para describir y seguir la vida de un requisito en ambas direcciones, hacia delante (forward) y hacia atrás (backward) (esto es, desde su origen, durante su desarrollo y especificación, hasta su desarrollo y uso, y a lo largo de todos los períodos de refinamiento en curso e iteración en alguna de estas fases"* [1]. La trazabilidad está condicionada por los cambios y las validaciones que los participantes del proyecto hagan al sistema durante el proceso de desarrollo. Como lo expresa Ramesh, *"muchos participantes diferentes –patrocinadores del proyecto, directores del proyecto, analistas, diseñadores, mantenedores y usuarios finales– están involucrados en el ciclo de vida de desarrollo del sistema. La trazabilidad necesita de estos participantes debido a las diferencias en sus metas y prioridades…" [2]*.

Una de las mejores prácticas para garantizar la exactitud del producto solicitado es que sea rastreado durante las actividades mismas de construcción del producto; no obstante, la mayoría de veces los analistas ven la trazabilidad como una actividad pasiva que adquiere importancia sólo en momentos en que se presentan problemas con los participantes o cuando se hace necesario hacer un mantenimiento al sistema. La importancia de la trazabilidad se evidencia en los diversos estudios acerca de su adopción y uso y en la creación de diferentes modelos y técnicas de trazado con el fin de hacer de ella una práctica más sencilla, intuitiva y necesaria durante el proceso de desarrollo [1, 2, 3, 4, 5].

El objetivo de este trabajo es analizar las características más relevantes de los modelos de trazabilidad, con el fin de conocer qué elementos y qué semánticas de trazado proveen para realizar el trazado y lograr la verificación de otros atributos de calidad, como la consistencia y la completitud en los modelos de desarrollo. Para lograr esto hacemos una revisión de algunos de los modelos de trazabilidad de requisitos más representativos y reconocidos por desarrolladores, ingenieros de calidad de software e investigadores de esta área de la ingeniería de software. Los modelos también serán comparados con respecto a su aplicación y sus ventajas desde el nivel de automatización de la trazabilidad.

La estructura de este artículo es como sigue. En la sección 2 se presentan trabajos relacionados con revisiones alrededor de la trazabilidad de requisitos. En la sección 3 se presentan los modelos de trazabilidad para su estudio. En la sección 4 se hace un análisis de los autores con respecto a la información suministrada por los modelos. Finalmente, en la sección 5 se concluye y se proponen trabajos futuros.

2. TRABAJOS RELACIONADOS

Los estudios que se han realizado alrededor de la trazabilidad están comúnmente orientados al análisis del problema de la adopción y uso de esta práctica en las industrias de software y tecnología. Gotel *et al.* [7] presentan un estudio acerca de las causas más destacadas del problema de la trazabilidad, el significado que ella tiene para diferentes participantes y el conocimiento y la profundización que puede hacerse en la práctica a partir del conocimiento de diferentes tipos de trazabilidad (pretrazabilidad y postrazabilidad). Ramesh *et al.* [8] presentan un estudio donde muestran que el problema de la trazabilidad depende de los diferentes contextos y usuarios de la trazabilidad. Gills [9] presenta el problema desde la falta de cultura, motivación y generación de productos de calidad. De forma muy resumida, en la tabla 1, mostramos los objetivos proyectados, el enfoque

del análisis del problema y los resultados obtenidos de los estudios mencionados.

Estos y otros estudios han servido para fomentar la creación, adopción y uso de modelos y técnicas para la práctica de la trazabilidad. En la siguiente sección se presentan algunos de los modelos de trazabilidad más representativos, el aporte que hacen desde los elementos y semántica que los determinan y cómo desde estos es posible lograr otros atributos de calidad, como la completitud y consistencia de los modelos que se van obteniendo a lo largo del desarrollo. Este análisis será utilizado como un recurso de investigación para revisiones futuras de la práctica de la trazabilidad en empresas de software.

Con el objetivo de brindar mayor comprensión de algunos términos, presentamos las definiciones de semántica de trazado, consistencia y completitud.

Tabla 1. Revisiones acerca de la adopción y uso de la trazabilidad

Evaluador	Objetivo	Enfoque del análisis del problema	Resultados y soluciones
Gotel *et al.* [7]	Analizar el problema de la trazabilidad de los requisitos	Aunque son muchas las técnicas y herramientas para la trazabilidad de requisitos, la persistencia del problema de su práctica y los resultados que de ella se obtiene llevan a revisar factores como: - Definición de trazabilidad: difieren en énfasis y delimitación del alcance. - Problemas subyacentes que están en conflicto: nivel de granularidad, diferentes usuarios, integración débil de tecnología, etc.	- Los problemas se centran en que hay una distinción débil entre pretrazabilidad y postrazabilidad. - Se hace énfasis en el mejoramiento de la pretrazabilidad de la especificación de requisitos por medio de aumentar la conciencia de información, obtener y registrar información para el trazado, organizar y mantener dicha información y proveer el acceso de la información y realizar su presentación necesaria.
Ramesh *et al.* [8]	Conocer qué factores críticos afectan la práctica de la trazabilidad	La adopción de la práctica de la trazabilidad está influenciada por el contexto tecnológico, el de estrategias corporativas y el de las políticas y el personal del sistema de desarrollo. El problema se centra en la forma como se realiza la práctica de la trazabilidad. Para esto se diferencian dos tipos de usuarios: low-end user, que usa esquemas simples de trazabilidad, y high-end user, que captura la información relacionada con el proceso como la razón fundamental detrás de varios artefactos y la evolución progresiva de esos artefactos.	Los factores que afectan la práctica de la trazabilidad son: - Condiciones para la adopción y uso de la trazabilidad: reconocer y articular el problema de la trazabilidad y formular objetivos de trazabilidad. - Adopción y uso de la trazabilidad: desarrollar métodos, adquirir herramientas, desarrollar herramientas y crear políticas de desarrollo del sistema de cambios. - Consecuencias en la adopción y uso de la trazabilidad: reacciones del patrocinador, del administrador o gerente de proyecto, de los desarrolladores y del resto de participantes.
Gills [9]	Estudiar la posibilidad de mejorar la práctica de la trazabilidad en empresas de tecnología de información	No hay cultura para la práctica de la trazabilidad, se considera un asunto difícil de implementar en la vida real.	Muy pocos proyectos tienen una aproximación sistemática para trazabilidad, lo cual hace necesario buscar mecanismos para que sea integrada por medio de herramientas de apoyo al proceso de desarrollo.

- *Semántica de trazado:* corresponde a la definición de los elementos (ítems) y vínculos de trazado dentro de un modelo de trazabilidad. El significado que tiene cada uno de ellos permitirá hacer el seguimiento a un requisito desde su identificación hasta los aspectos de implementación.
- *Consistencia:* un modelo es consistente en un nivel de abstracción Y, con respecto a un nivel de abstracción X, si conserva el mismo significado del nivel de abstracción X.
- *Completitud:* Se refiere al grado en que un modelo cumple las necesidades de los usuarios en cualquier nivel de abstracción. Esto se cumple si todos los significados del nivel de abstracción X se hallan en el nivel de abstracción Y, y viceversa.

3. MODELOS DE TRAZABILIDAD

3.1 Generalidades

Se han creado diferentes técnicas y modelos para soportar la práctica de trazabilidad durante el proceso de desarrollo de software. La técnica más común y aplicable a cualquier modelo de desarrollo es la construcción de matrices de trazado, que hacen posible el análisis de la correlación entre elementos de un mismo modelo y entre modelos en diferentes niveles de abstracción. Por ejemplo, en la figura 1 se muestra una de las matrices más comunes de trazado: la de Requisitos y Casos de Uso, la cual permite verificar en qué casos de uso son representados y especificados los requisitos funcionales elicitados desde el espacio del problema.

Casos de Uso / Requisitos	CU1	CU2	CU3	CU4
Requisito 1	✓			✓
Requisito 2		✓		
….				
Requisito n	✓			

Figura 1. Matriz de trazado, Requisitos y Casos de Uso

Otro ejemplo es la matriz CRUD (Create, Retrieve, Update and Delete). Permite analizar las operaciones que deben realizarse sobre la base de datos a partir de la correlación entre tablas y funciones y entre otros elementos de la base de datos.

La construcción de estas matrices trae beneficios, más allá de un simple registro de la correlación o dependencia entre los elementos de los modelos. A partir de ellas es posible analizar características tales como el nivel de especificación de los requisitos, el nivel de participación de los usuarios, el costo asociado a cada fase de desarrollo, la arquitectura requerida, el plan de las pruebas, etc. Otra de las ventajas de estas matrices es que, si se actualizan continuamente, pueden servir como herramienta de soporte muy útil para el personal encargado del mantenimiento y las pruebas del sistema.

De otra parte, esta práctica puede ser costosa y tediosa para sistemas de gran tamaño o complejos. Algunas herramientas CASE posibilitan la generación automática de estas matrices o de forma automática generan elementos de un modelo a otro en diferentes niveles de abstracción (p. ej. del modelo de análisis al modelo de diseño), para obtener así una correlación *(mapping)* de trazado; sin embargo, no todos los elementos pueden ser correlacionados de forma automática o simplemente no tienen elementos definidos en los modelos destino.

3.2 Revisión de modelos de trazabilidad

Las aproximaciones que hemos seleccionado para la revisión son los siguientes: Lindvall [1], Gotel *et al.* [2], Ramesh *et al.* [3], Letelier [4] y Egyed [5, 6]. De cada una de ellas explicamos su modelo, su semántica para realizar el trazado y los mecanismos utilizados para la verificación de consistencia y completitud.

Lindvall [1] presenta un modelo que ofrece diferentes tácticas para realizar el trazado entre artefactos de software que resultan del proceso software en tres niveles de abstracción: el dominio, el análisis y el diseño. Aplica la trazabilidad hacia

delante *(forward)* y hacia atrás *(backward)*, vertical y horizontal. Distingue entre artefactos permanentes y temporales para jerarquizar el trazado.

Clasifica la trazabilidad en dos dimensiones. En la primera, se identifican los vínculos de trazados entre los diferentes elementos del modelo, por ejemplo: de objeto a objeto *(object-to-object)*, relación a relación *(association-to-association)*, caso de uso a caso de uso (use-case-to-use-case), requisitos a casos de uso, etc. (figura 2). En la segunda, se determina cómo se realiza el trazado: por vínculos explícitos (o relaciones de los modelos de desarrollo), por referencia, por nombre o por la experiencia del desarrollador y el conocimiento que tenga del dominio y del sistema para distinguir otros ítems que estarían relacionados.

Verificación de consistencia: evalúa en el modelo de desarrollo la posibilidad de que haya algún ítem o conjunto de ítems y de asociaciones que se contradigan entre sí o que de una u otra forma hagan el modelo inconsistente. Procura la consistencia a partir del trazado por nombre. La inspección es informal y no tiene forma de validar si todos los ítems y asociaciones están involucrados en la inspección.

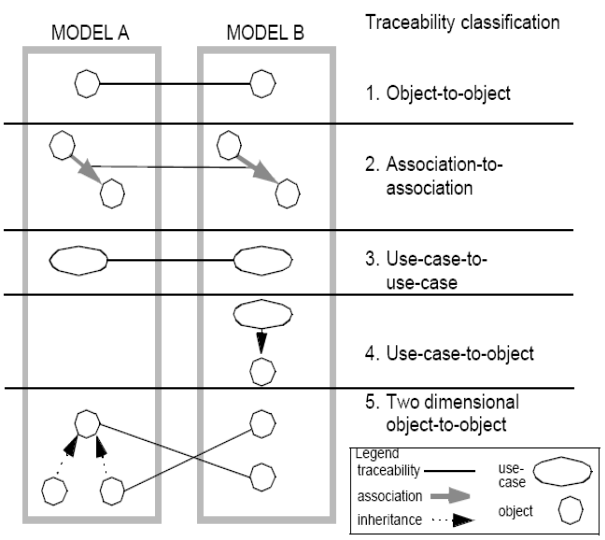


Figura 2. Clasificación de la trazabilidad, según Lindvall

Verificación de Completitud: evalúa si el material de especificación de requisitos utilizados en los diferentes niveles de abstracción está completo. Es una inspección intuitiva que depende de la importancia que cada participante les dé a los documentos producidos en el ciclo de vida.

Gotel y **Finkelstein** [2] presentan un modelo que hace posible extender formas convencionales de trazabilidad de requisitos basada en artefactos. En su propuesta presentan dos tipos de requisitos: basados en artefactos y basados en el personal participante. Crean una "*estructura de contribución*" para realizar la práctica de la trazabilidad, que consiste en hacer trazabilidad de requisitos basada en las contribuciones que hacen los participantes a los artefactos producidos en el proceso de ingeniería de requisitos.

En la semántica asociada a dicha estructura es posible encontrar dos vínculos de trazado entre los artefactos de un modelo de desarrollo: *adopta* para indicar las relaciones basadas en artefactos que son típicamente mantenidas para la trazabilidad de requisitos; *referencia* para indicar las relaciones basadas en artefactos que proporcionan información más allá del contexto básico.

Se definen también para los participantes tres roles de responsabilidad con respecto a un artefacto: *autor* (responsable original), *principal* (responsable corriente) y *documentador*. Esto ayudará posteriormente a verificar la colaboración entre los participantes y sus roles en el trazado.

La naturaleza de la semántica de los vínculos basados en artefactos proviene de la descripción de las relaciones dadas entre un participante y la naturaleza de las relaciones en términos de su cohesión y coherencia. Por ejemplo, *b adopta a* (siendo *a* y *b* dos elementos del modelo de desarrollo) proviene de las relaciones: *b califica a* (por el participante) y *b adiciona a* (por coherencia o cohesión de los artefactos). La semántica de *adopta* está asociada a operaciones tales como adicionar, remover y alterar. Este modelo expone el trazado desde los artefactos y sus participantes conjuntamente con el objetivo

de poder inferir más tarde acerca del modelo y los compromisos de sus participantes, a los cuales son atribuidos roles sociales (p. ej. *Autorverdadero, Devisor/Relayer*[1]*, AutorRepresentativo y AutorFantasma)*.

Verificación de consistencia y completitud: se evalúan a partir de los compromisos de los participantes (rol social). Esto se representa en una matriz donde se relaciona cada artefacto con el rol social del participante, los colaboradores y los artefactos que lo adoptan y lo refieren.

Rashamel y **Jarke** [3] presentan un modelo basado en un estudio acerca de la adopción y uso de la trazabilidad de requisitos. A partir de él se crea el metamodelo de trazabilidad (figura 3a) y se diferencian dos tipos de usuarios de trazabilidad de "bajo perfil *(low-end)*" y de "alto perfil *(high-end)*". El metamodelo representa los tres aspectos más importantes para la práctica de la trazabilidad: las fuentes, los participantes y los objetos para ser trazados. La semántica asociada a los vínculos de trazado (*trace_to, has_role_in,* etc.) está dada por: Object *traces_to* Object, Stakeholder *has_role_in* Object, Stakeholder *manages* Source y Source *documents* Object.

La metaclase *Objeto (Object)* es extendida desde una vista de alto nivel con las siguientes partes que la componen: Requirements Management, Design Allocation y Compliance Verification (figura 3b). Éstas permiten a los usuarios de alto perfil registrar información que les permita hacer inferencias en los modelos de desarrollo.

Además, van acompañadas de los siguientes vínculos de trazado: *satisfied,* que representa la información que se considera crítica y se encarga de asegurar que todos los requisitos de usuarios sean trazados. *Depends_on* se refiere a la dependencia entre objetos trazados vistos desde un alto nivel. *Envolves_to* corresponde al registro de la evolución a partir de los cambios que pueden sufrir los modelos de desarrollo (instancias de este vínculo podrían ser: *modificado, definido, elaborado, derivado)*. *Rationale* representa los fundamentos o definiciones básicas de artefactos, decisiones, etc. Cualquier entidad o relación de este modelo puede ser instanciada.

Los Participantes (Stakeholders) representan agentes involucrados en el sistema de desarrollo y mantienen las actividades del ciclo de vida; tienen diferentes roles o capacidades sobre los objetos y vínculos de trazabilidad. Las *Fuentes* (Sources) documentan los *Objetos* y los *Participantes* manejan las Fuentes. Los *Objetos* representan cualquier ítem de entrada o salida del modelo de desarrollo, como requisitos, diseños, sistema de componentes, suposiciones, decisiones, fundamentos, alternativas, factores críticos, etc.

El vínculo de trazado también puede ser instanciado, por ejemplo: Requisitos *deriva* Requisito, Requisito *depende_de* Suposición, Requisitos *localizado_en* Sistema_de_Componentes, Requisito *satisface* Sistema_de_Componentes, Procedimiento_de_verificación *realizado_en* Sistema_de_Componentes, Sistema_de_Componentes *depende_en* Sistema_de_Componentes, Procedimiento_de verificación d*esarrollado_por* Requisitos, Sistema_de_Componentes *interfaz_con* Sistemas_Externos[2]. En otro nivel es posible encontrar vínculos de trazado tales como:

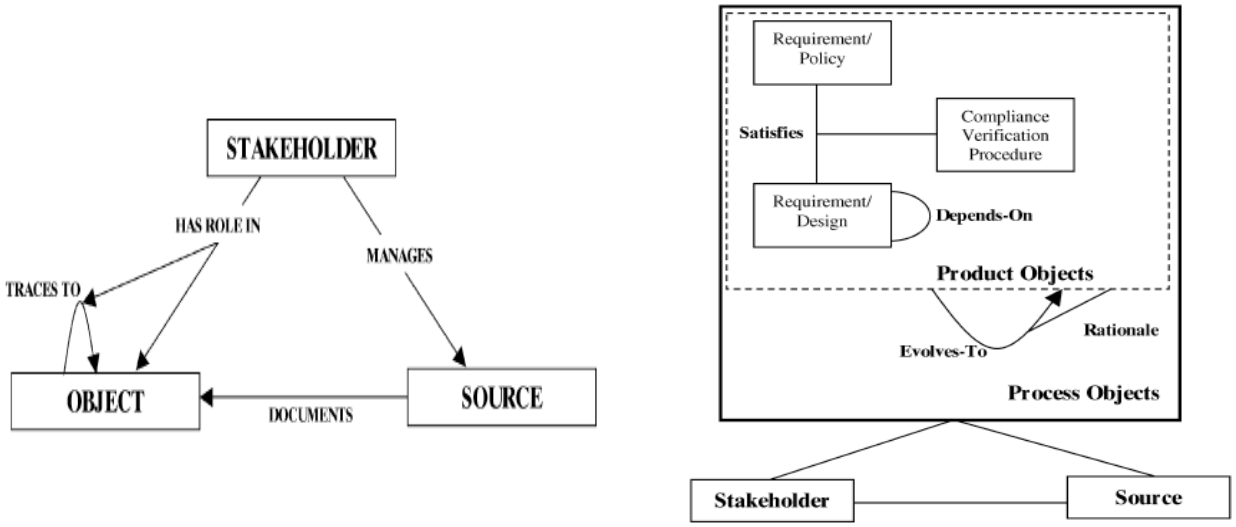
Necesidades_Organizacionales *justifica* Sistema_de_Objetivos, Escenarios *describen* Necesidades_Organizacionales, Sistema_de_Objetivos *genera* Requisitos, Necesidades_Organizacionales *identifica* Factores_Críticos, etc.[3].

Verificación de consistencia: se realiza a partir de las dependencias y vínculos de trazado entre los objetos y se garantiza por medio de un metaadministrador de bases de datos basadas en el conocimiento.

1 *Devisor:* rol social que se atribuye a los participantes que son tanto *principales* como *autores* de un artefacto. *Relayer:* rol social que se atribuye a los participantes que solo actúan como *documentadores*.

2 Este tipo de instancias corresponden a usuarios de "*bajo perfil*" de trazabilidad.

3 Este tipo de instancias corresponden a usuarios de "*alto perfil*" de trazabilidad.



(a) (b)

Figura 3. a) Metamodelo básico de trazabilidad. b) Metamodelo de trazabilidad extendido [3].

Verificación de completitud: parte de la información suministrada por los *Componentes de Verificación de Conformidad*. Estos producen resultados que permiten *verificar* cómo, por ejemplo, los Componentes *satisfacen* Requisitos o ayudan a generar ofertas de cambio para los requisitos, el diseño o la implementación.

Letelier y **Anaya** [4] presentan un metamodelo basado en UML. Para la práctica de la trazabilidad definen entidades y vínculos de trazado (figura 4). *TraceableSpecification* es la entidad que representa los elementos de los modelos de desarrollo y permite cubrir las cuatro perspectivas de información de trazabilidad: requisitos *(RequirementSpecification)*, fundamentos *(RationaleSpecification)*, asignación de requisitos a artefactos de los modelos de desarrollo *(OtherUML_Specification)* y pruebas *(TestSpecification)*. Este modelo está preparado para realizar pretrazabilidad y postrazabilidad.

La semántica asociada a los vínculos de trazado se representa en el metamodelo. Algunos de ellos son: TraceableSpecification *TraceTo* TraceableSpecification, Stakeholder *modifies* TraceableSpecification, Stakeholder *responsibleOf* TraceableSpecification, RequirementSpecification *verifiedBy* TestSpecification.

Verificación de consistencia y completitud: la propuesta no expresa la forma ni los elementos mediante los cuales sea posible lograr estos atributos de calidad.

Egyed [5, 6] expone que el objetivo del modelo propuesto es producir información de trazabilidad generada entre los sistemas de software y sus modelos (figura5a). Está basado en la observación de escenarios de prueba que son ejecutados durante la corrida de un sistema de software.

A partir de dicha observación se establecen vínculos de trazado entre los elementos de modelos (clase y flujos de datos) y su correspondiente código fuente (el sistema).

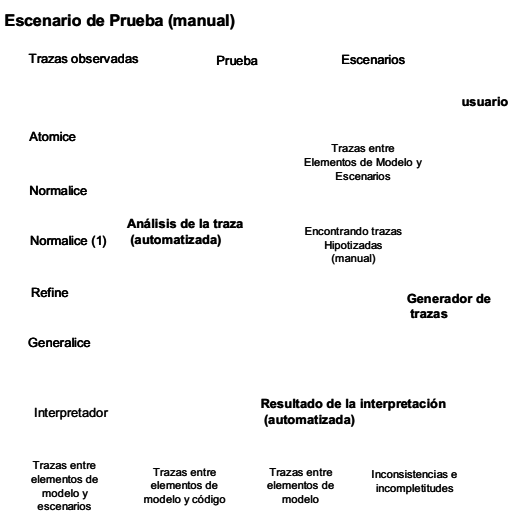
El modelo está orientado a manejar tipos de trazado entre escenarios y el sistema, elementos de los modelos y el sistema, escenarios y elementos de los modelos, elementos de los modelos y elementos similares, entre los mismos elementos de los modelos, entre conjuntos de elementos del modelo, posibles inconsistencias e incompletitudes. Además, provee

Figura 4. Metamodelo de trazabilidad, según Letelier y Anaya

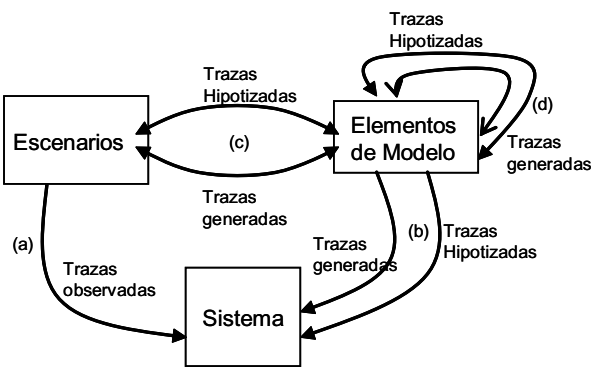
tres dependencias de traza: *hipotetizada, generada* y *observada* (figura 5b). La primera actividad requiere analizar las posibles *trazas hipotéticas* que pueden ser levantadas desde documentación o modelos; las trazas generadas mediante una iteración pueden usarse como trazas hipotéticas en iteración sucesiva; además usa conocimiento corriente de las trazas y lo permuta con trazas observables.

Egyed también provee otros tipos de vínculos que se usan para analizar el grado de incertidumbre del trazado entre las líneas de código y los elementos del modelo; algunos de estos son: *incluido, excluido, esCompartido, esAloSumo, esPorloMenos, NoEs, EsExactamente,* etc.

Verificación de la consistencia y la completitud: son verificadas a partir de las trazas hipotéticas



a)



b)

Figura 5. a) Modelo de trazabilidad general b) Dependencias de traza, según Egyed

excesivas. Por ejemplo, si un elemento del modelo *m* es exactamente alguna huella de trazado *f* (en el código), entonces se reconoce que *m* traza a dicha huella de trazado y no a otra, además que la huella pertenece a *m* y no a otro modelo. Ésta se considera una entrada precisa y completa.

4. ANÁLISIS DE LAS APROXIMACIONES

En la revisión conceptual de la sección anterior, cada autor presenta un modelo que permite a los desarrolladores tener medios o formas para hacer de la práctica de la trazabilidad una actividad necesaria e inseparable durante el proceso de desarrollo. Todas las aproximaciones de alguna u otra forma coinciden en tener en cuenta los tres elementos básicos para la práctica de la trazabilidad: los artefactos, los participantes y las fuentes. Pero, realmente, aunque parezca que tienen diferencias sutiles, el nivel de profundidad y automatización que pueden lograr en estos factores hacen que los diferentes modelos propuestos establezcan patrones o directrices en la práctica de la trazabilidad.

Gotel y Finkelstein [2] profundizan en las responsabilidades de los participantes en cuanto a la creación y el mantenimiento de los artefactos asociados a los requisitos. Ramesh *y* Jarke [3] profundizan en la diferenciación de los tipos de usuarios de la trazabilidad. Así logran distinguir las labores operativas del trazado de las que realmente proveen más información a los modelos de desarrollo. Además, presentan un alto nivel de automatización del trazado. Egyed [6] extiende su propuesta al manejo de incertidumbre desde el trazado, lo cual permite lograr un alto grado de certeza en el modelado a partir de la práctica de la trazabilidad.

Es importante resaltar que algunas de estas propuestas encuentran limitaciones en el trazado de requisitos no funcionales, ya que UML no proporciona elementos de modelo para este tipo de requisitos. De esta forma, consideramos importante probar estas aproximaciones con esquemas de análisis de requisitos no funcionales [9] o aproximaciones de ingeniería de requisitos orientada a aspectos [10].

5. CONCLUSIONES

En la actualidad la práctica de la trazabilidad se conduce por las herramientas CASE que permiten hacer una buena administración de los requisitos. Pero consideramos importante que el personal de desarrollo o calidad dedicado a la validación y verificación de la trazabilidad de requisitos establezca un modelo de trazado con el cual pueda medir el nivel de exactitud de los modelos de desarrollo con respecto a los requisitos planteados por los usuarios.

De igual forma consideramos conveniente realizar la práctica de la trazabilidad teniendo en cuenta factores tales como: complejidad del software, nivel de especificación de los requisitos y de los modelos, identificación de dependencias entre los requisitos, lenguaje de modelado utilizado para representar los requisitos, empleo de artefactos para la reutilización, nivel de soporte de los modelos y las estructuras para las pruebas y los cambios, dependencias entre los elementos de cada modelo y nivel de abstracción, lenguaje de representación utilizado en los modelos, entre otros.

6. REFERENCIAS

[1] Lindvall M. and Sandahl, K. Practical implications of traceability. Journal of Software Practice and Experience, 26(10), p. 1161-1180, 1996.

[2] Gotel O. and Finkelstein A. Extended requirements traceability: results of an industrial case study. In: Proc. 3rd IEEE Inter. Symposium on Requirements Engineering (RE'97), p. 169.

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

[4] Letelier P. y Anaya V. Entregando especificaciones textuales y elementos de modelado UML en un marco de trabajo para trazabilidad de requisitos. JISBD p. 151-162, 2002.

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

[6] Egyed A. Resolving uncertainties during trace analysis. Proc. of the 12th ACM SIGSOFT Symposium on Foundations of Software Engineering (FSE), Irvine, CA, Nov. 2004, p. 3-12.

[7] Gotel O. and Finkelstein A. An analysis of the requirements traceability problem. In: Proc. 1st IEEE Int. Conf. on Requirements Engineering, p. 94-101, Colorado Springs, April 1994.

[8] Ramesh B. Factors influencing requirements traceability practice. Communications of the ACM, 41(12):37-44, 1998.

[9] Gills M. Survey of traceability models in IT projects. Proceedings ECMDA Traceability Workshop (ECMDA-TW) 2005. p. 39-46.

[10] Chung, L., Nixon, B.; Yu, E. and Mylopoulos, J. Non-functional requirements in software engineering, Kluwer Academic Publishers, 2000.

[11] Chitchyan R.; Rashid A.; Sawyer P.; Bakker J.; Pinto M.; García A.; Tekinerdogan B.; Clarke S. and Jackson A. Survey of aspect-oriented analysis and design approaches, AOSD-Europe-ULANC-9, AOSD-EUROPE Network of Excellence. May 2005.