Parametrización de las transformaciones horizontales en el modelo de herradura

Jesús Sánchez Cuadrado[1], Orlando Avila García[2],
Javier Luis Cánovas Izquierdo[3], Adolfo Sánchez-Barbudo Herrera[2]

[1] Universidad Autónoma de Madrid (jesus.sanchez.cuadrado@uam.es)
[2] Open Canarias, S.L. (orlando@opencanarias.com, adolfosbh@opencanarias.com)
[3] AtlandMod, École des Mines de Nantes – INRIA – LINA, Francia
(javier.canovas@inria.fr)

Resumen En los procesos de modernización o reingeniería de software se aplica generalmente el denominado *modelo de herradura*. En este modelo hay transformaciones verticales entre artefactos software de diferente nivel de abstracción y transformaciones horizontales en el mismo nivel de abstracción. A pesar de ser un modelo conocido y usado en numerosos trabajos todavía no se ha explorado completamente cómo automatizarlo. En este artículo se discuten los problemas existentes para su automatización, y se motiva la problemática con un caso real de modernización. Se describe una aproximación basada en la parametrización de transformaciones horizontales con información descubierta en las transformaciones verticales y los cambios realizados en los modelos de niveles superiores.

1. Introducción

En los procesos de modernización o reingeniería de software se aplica generalmente el denominado *modelo de herradura* [1], un marco de trabajo para integrar los diferentes niveles de abstracción y las actividades que caracterizan este tipo de procesos. La Modernización Dirigida por la Arquitectura (*Architecture-Driven Modernization*, ADM) [2] es una iniciativa de la OMG (*Object Management Group*) que, basándose en el paradigma del Desarrollo de Software Dirigido por Modelos (DSDM), busca proponer y estandarizar técnicas, métodos y herramientas para la modernización en el marco del *modelo de herradura*.

El objetivo principal del modelo de herradura es obtener una representación abstracta del sistema origen que facilite su comprensión y transformación para obtener el sistema destino. Al aplicar este modelo se pueden identificar transformaciones verticales entre artefactos software de diferente nivel de abstracción y transformaciones horizontales en el mismo nivel de abstracción. El modelo no prescribe que las transformaciones sean automatizadas, aunque en ADM se promueve su automatización utilizando técnicas del DSDM.

A pesar de que el modelo de herradura cuenta con un importante bagaje dentro de la comunidad científica y el respaldo de numerosos proveedores de herramientas de modernización, no existen técnicas contrastadas para su automatización ni herramientas que den un soporte integral al modelo. El único

A. Ruíz, L. Iribarne (Eds.): JISBD'2012, pp. 551-556, ISBN:978-84-15487-28-9.
Jornadas SISTEDES'2012, Almería 17-19 sept. 2012, Universidad de Almería.

punto del modelo en herradura que parece formalmente resuelto es la "extracción" de los llamados "modelos de aplicación" a partir del sistema origen, donde la disciplina de la ingeniería inversa ha trabajado durante décadas. Sin embargo, el resto de etapas permanecen escasamente resueltas.

Algunas aproximaciones tratan de aplicar las transformaciones horizontales, encargadas de la reingeniería, únicamente a alto nivel de abstracción y, a continuación, utilizan transformaciones verticales para generar los artefactos software (ingeniería directa). Como se argumentará en la Sección 2, esta aproximación no es realista porque la información requerida para regenerar el sistema no está disponible en los modelos de más alto nivel de los que se parte.

Este problema se pone de manifiesto en el caso de estudio presentado en la Sección 3, el cual aplica una refactorización del código donde la información de bajo nivel del sistema origen debe prevalecer en el sistema destino, por lo que una aproximación como la aplicada normalmente no es posible. En su lugar, se planteó realizar una transformacón horizontal a bajo nivel de abstracción, pero parametrizada con información de los modelos de más alto nivel.

Nuestro objetivo a largo plazo es estudiar cómo automatizar la aplicación del modelo de herradura. En particular, en este trabajo nos centramos en cómo parametrizar las transformaciones horizontales, ya que pensamos que la clave de la automatización del modelo está en este tipo de transformaciones, puesto que permiten manejar la información descubierta en el proceso de ingeniería inversa a diferentes niveles de abstracción. La Sección 4 presenta dos aproximaciones diferentes, que son aplicables a escenarios de refactorización y migración.

2. El modelo de herradura

En un proceso de reingeniería el objetivo es transformar un sistema software para que cumpla los nuevos requisitos. El modelo de herradura [1] ofrece un marco de trabajo para la aplicación de procesos de reingeniería, facilitando la integración de los diferentes niveles de abstracción involucrados. En este modelo se definen dos tipos de transformaciones: (1) las verticales, que se aplican entre artefactos software de diferente nivel de abstracción, y (2) las horizontales, que se aplican en el mismo nivel de abstracción. La Figura 1a muestra una visión general del modelo, tal y como es propuesto originalmente en [1].

El modelo divide el proceso de reingeniería en tres fases: (1) fase de ingeniería inversa, donde se aplican transformaciones verticales que recuperan las decisiones de diseño tomadas en cada nivel de abstracción; (2) fase de reestructuración, donde se aplican transformaciones horizontales para adaptar o migrar los artefactos existentes al nuevo sistema; y (3) fase de ingeniería directa, donde normalmente se aplica un desarrollo basado en la arquitectura. Es importante observar que la definición del nuevo sistema implica refinar los artefactos de más alto nivel (para añadir nuevas características), pero también integrar los artefactos existentes de bajo nivel a través de transformaciones horizontales, que de alguna forma deben ser informadas por los modelos de nivel superior.

Sin embargo, el modelo normalmente es mal interpretado y simplificado, como se muestra en la Figura 1b. En estos casos se considera que las transformaciones horizontales de la fase de reestructuración se aplican solamente a los

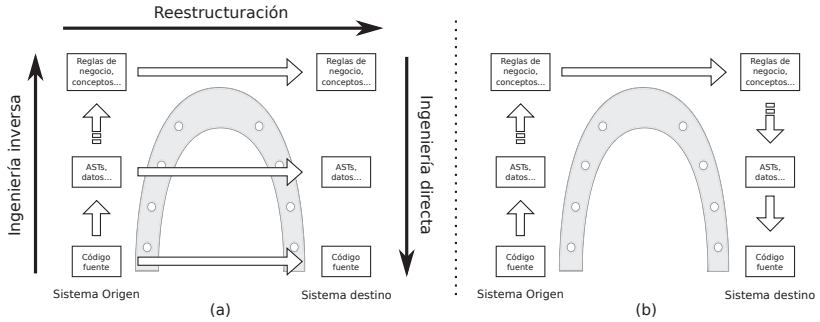

Figura 1. Modelo de herradura (a) original y (b) sin transformaciones horizontales.

artefactos de más alto nivel de abstracción, y que los artefactos resultantes son utilizados para generar el sistema mediante ingeniería directa. Este es el proceso aplicado en trabajos como [3,4] y recomendado por ADM en [5]. Sin embargo, esto no es factible en muchos casos de reingeniería (la sección 3 presenta un ejemplo concreto). El principal problema es la pérdida de información que se produce en las transformaciones verticales (las decisiones de diseño recuperadas en un nivel se pierden al abstraer al nivel superior), que imposibilita "bajar por el lado derecho de la herradura" mediante transformaciones automáticas. Denominaremos a esta versión del modelo "modelo de herradura directo".

Nuestro objetivo es idear un mecanismo basado en técnicas del DSDM para la automatización de procesos de reingeniería donde se consideren las transformaciones horizontales a diferentes niveles de abstracción, respetando el modelo de herradura original. De esta forma, la información recogida en los modelos de más alto nivel, obtenidos tanto en la ingeniería inversa como en la restructuración, guiarán las transformaciones horizontales de los niveles inferiores. El principal reto, por tanto, es cómo parametrizar estas transformaciones horizontales.

3. Caso de estudio

En 2010, el Grupo Banca Cívica decide modernizar su "core bancario" o plataforma tecnológica básica de una entidad financiera. Este escenario requería convertir los programas en "neutros", esto es, independientes de la entidad financiera. Para ello la refactorización debía eliminar cualquier tipo de literal (*hard-coded*) que hiciera referencia al nombre o identificador para convertirlo en una variable que los programas recibieran en tiempo de ejecución.

Las actuaciones a-priori eran sencillas: quitar ciertos literales del código fuente para sustituirlos por variables. Sin embargo, existía un problema importante: el nombre y expresión de la variable por la que que sustituir cada literal dependía del contexto y tipo de programa donde aparecía. Además, existían al menos seis tipos de programas y más de diez contextos de aparición. El proceso de identificación y sustitución dependiente del contexto debía realizarse para las cerca de 100 aplicaciones bancarias, constituidas por más de 14 mil programas COBOL, que sumaban un total de 26 millones de líneas de código fuente.

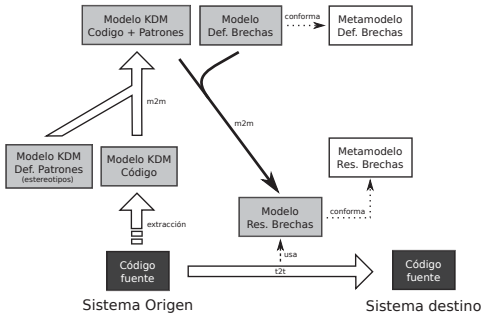

Figura 2. Proceso de transformación llevado a cabo en el proyecto con Banca Cívica

3.1. Aplicación del modelo de herradura directo

El modelo de herradura propone una abstracción progresiva de la descripción de las aplicaciones, de manera que en cada nivel nos podamos centrar en ciertas características (decisiones de diseño) mientras eliminamos otras. Para nuestro problema se percibía necesario abstraer el nivel textual para alcanzar un nivel superior que nos permitiera realizar el *pattern-matching* de literales y sus contextos de aparición, esto es, era necesario un proceso de ingeniería inversa.

Una vez identificadas las apariciones que debían ser sustituidas, había que modificar los programas originales. En este punto, el modelo de herradura directo, tal y como se muestra en la Figura 1b (ingeniería inversa, reestructuración e ingeniería directa) no era factible. Por un lado, debía mantenerse intacto el texto de todo el código no directamente involucrado en la transformación de dichos literales, como los comentarios o las columnas. Por otra parte, incluso sin estas restricciones, regenerar el sistema original siguiendo el modelo directo implicaba trasladar todas las decisiones de diseño e implementación recuperadas del sistema original a través de los diferentes niveles de abstracción. Esta aproximación no es ni eficaz ni eficiente, y por tanto fue descartada.

3.2. Modelo de herradura con parametrización

La solución que se planteó fue hacer una transformación a bajo nivel de abstracción, de texto a texto, de sólo aquellos elementos relevantes. Para conseguirlo, hubo que informar (parametrizar) la transformación con datos inferidos a un mayor nivel de abstracción. Estos datos fueron básicamente dónde y qué tipo de actuación (cambio) realizar en el texto del códido fuente. Esta variación del modelo clásico de herradura se plasma en la Figura 2.

El primer paso fue extraer modelos de código de los ficheros fuente, conformes al metamodelo *KDM* (Knowledge Discovery Metamodel). A partir de este modelo se realizó un nuevo análisis que permitía identificar aquellos literales de interés que se manifestaban en alguno de los contextos definidos, localizando en dichos modelos, las apariciones de los patrones (literal-contexto) planteados.

A continuación se definió el modelo de definición de brechas (*MDB*), que permitía establecer el conjunto de acciones necesarias a acometer sobre el código

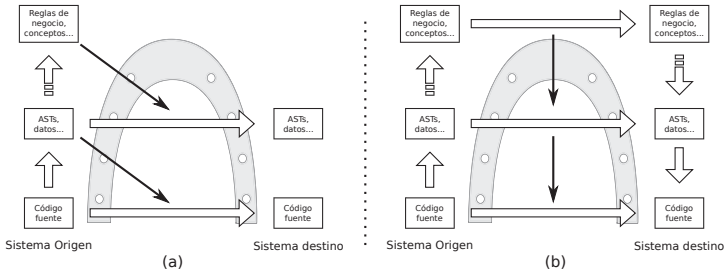

Figura 3. Modelo de herradura con parametrización (a) explícita e (b) implícita.

fuente (nivel de abstracción inferior) para eliminar la aparición de un patrón dado en el modelo de literales identificados (nivel de abstracción superior), esto es, sustituir el literal por la variable correspondiente. De esta forma, por medio de una transformación de modelos, se analizaba el modelo con los literales identificados y en base a lo establecido en el *MDB* se generaba el denominado modelo de resolución de brechas (*MRB*) que es un modelo que especificaba las acciones a realizar por la transformación horizontal de bajo nivel encargada de obtener un fichero fuente con los literales originales sustuidos por variables. El *MRB* se convirtió por tanto en el parámetro para una transformación horizontal a nivel del texto fuente de los programas.

4. Parametrización de las transformaciones horizontales

Dada la problemática explicada en las secciones anteriores, nuestro objetivo es estudiar las diferentes maneras de parametrizar las transformaciones horizontales, permitiendo aumentar el grado de automatización del modelo de herradura. Hasta el momento hemos identificado dos escenarios diferentes.

Escenario 1: Parametrización explícita. Este escenario corresponde al caso de estudio, y la Figura 3a muestra una generalización de este esquema. Como se puede observar, no se producen transformaciones horizontales en los modelos de más alto nivel, sino que éstos sirven para deducir información (de forma automática o asistida por el usuario), que guiará las transformaciones en los niveles inferiores. Para ello, se generan modelos de parámetros (*MRB* en el caso de estudio) utilizando transformaciones modelo-modelo.

Creemos que esta aproximación será útil en escenarios de modernización que no producen cambios en la arquitectura o funcionalidad del sistema. La principal desventaja, es que introduce un nivel adicional de complejidad al tener que definir nuevos metamodelos para los parámetros, así como la creación de nuevas transformaciones que actúan transversalmente. A cambio, es explícito de dónde proviene la información necesaria para las transformaciones horizontales.

Escenario 2: Parametrización implícita. En este escenario sí existen transformaciones horizontales en los modelos de más alto nivel, por ejemplo para realizar cambios en la arquitectura o en la funcionalidad del sistema. Consid-

erese por ejemplo una migración de código Cobol a Java, en la que se pasa de un estilo procedural a uno orientado a objetos.

En este tipo de escenarios los cambios realizados por transformaciones en un nivel de abstracción superior deben tenerse en cuenta en las transformaciones de niveles inferiores para que éstas sean coherentes. Llamamos a este tipo de parametrización *implícita* puesto que debe derivarse del resultado una transformación de modelos (las trazas), como se muestra en la Figura 3b.

En este trabajo no se ha incluido ningún ejemplo de este tipo de escenario, sin embargo los casos de estudio presentados en [6] y [7] sugieren que tiene sentido su aplicación. En ellos los modelos de nivel superior contienen referencias hacia los elementos de nivel inferior. El problema de esta aproximación es que la gestión de las referencias debe realizarse manualmente tanto en las transformaciones verticales como en las horizontales. Además, los metamodelos deben modificarse para tener en cuenta estas referencias.

5. Conclusiones

En este trabajo hemos presentado algunos problemas que aparecen al automatizar el modelo de herradura usando transformaciones verticales y únicamente una transformación horizontal aplicada en el nivel de abstracción superior. Nuestras experiencias en casos de estudio de reingeniería sugieren que el uso generalizado de transformaciones horizontales, a diferentes niveles de abstracción, permiten evitar estos problemas. Sin embargo, estos estudios preliminares también han hecho patente la necesidad de parametrizar dichas transformaciones con información de los modelos de más alto nivel de abstracción.

En este trabajo hemos motivado la problemática, introducido el concepto de parametrización de transformaciones horizontales y presentado una tipología preliminar: explícita e implícita. Para establecer esta tipología nos hemos apoyado en casos de estudio presentados tanto en este trabajo como en trabajos relacionados. Nuestro objetivo a medio plazo es formalizar y definir teoricamente este tipo de parametrización, y proponer métodos y técnicas para incluir transformaciones horizontales en arquitecturas generativas aplicadas a los distintos escenarios de reingeniería de software.

Referencias

1. Kazman, R., Woods, S.S., Carrière, S.J.: Requirements for integrating software architecture and reengineering models: Corum ii. In: WCRE. (1998) 154–163
2. Ulrich, W.M., Newcomb, P.: Information Systems Transformation: Architecture-Driven Modernization Case Studies. Morgan Kaufmann (2011)
3. Kshusidman, V.: Soa enabled workflow modernization. BP Trends (2006)
4. Yu, Y., Wang, Y., Mylopoulos, J., Liaskos, S., Lapouchnian, A., do Prado Leite, J.C.S.: Reverse engineering goal models from legacy code. In: RE. (2005) 363–372
5. Kshusidman, V.: Adm transformation. ADM Task Force. White Paper (2008)
6. Sánchez Ramón, O., Sánchez Cuadrado, J., García Molina, J.: Model-driven reverse engineering of legacy graphical user interfaces. In: ASE '10. (2010) 147–150
7. Sánchez Ramón, O., Sánchez Cuadrado, J., García Molina, J.: Reverse engineering of event handlers of rad-based applications. WCRE (2011) 293–302