Diseño de un DSL para el Dominio Salud aplicando Ingeniería de Requerimientos Dirigida por Modelos

Ariste María Cecilia 1, Rocca Leandro 1, Caputti Matías 1,
Zugnoni Iván 1, Vicente Diego 1, Nahuel Leopoldo 1,2, Giandini Roxana 1,2,3

1 GIDAS – Grupo de Investigación del Departamento de Sistemas - Facultad Regional La Plata - Universidad Tecnológica Nacional, La Plata, Buenos Aires, Argentina
2 LIFIA - Facultad de Informática - Universidad Nacional de La Plata, La Plata, Buenos Aires, Argentina
3 CIC - Comisión de Investigaciones Científicas de la Provincia de Buenos Aires.

{mariste83, leorocca, matias.caputti, zugnoni.ivan,dvicentea, leonahuel, roxanagiandini}@gmail.com

***Resumen*—Con la intención de poner en práctica Ingeniería de Requerimientos Dirigida por Modelos IRDM en el contexto del Desarrollo Dirigido por Modelos MDD y de brindar un aporte metodológico nos enfocamos en el dominio de Salud y nos basamos en estándares abiertos de modelado de recursos de información e interoperabilidad para Salud FHIR de HL7 y OpenEHR. Este trabajo muestra el avance de un proyecto para el desarrollo de un Lenguaje Específico del Dominio (DSL) para Salud, al que llamamos DSL_SALUD, además del diseño de una herramienta CASE que implemente dicho lenguaje que permitirá crear modelos escritos en el lenguaje DSL_SALUD, y generar código automáticamente, además de dar la facilidad de transformarlos automáticamente a otros modelos escritos en lenguaje UML. Todo esto se hace con la intención de que los Ingenieros de Software, especializados en soluciones IT para Salud, puedan hacer uso tanto del metalenguaje DSL_SALUD como de la herramienta CASE para llevar a cabo la construcción de modelos que representen los requerimientos propios para sus desarrollos de aplicaciones software para Salud.**

***Palabras clave*— Lenguaje Específico del Dominio DSL, Metalenguajes, Desarrollo Dirigido por Modelos MDD, Ingeniería de Requerimientos Dirigida por Modelos IRDM, Herramienta CASE, Interoperabilidad para Sistemas de Salud, FHIR de HL7, OpenEHR**

I. Introducción

La construcción de modelos es una de las claves para desarrollar nuevas tecnologías de información en diversos campos de aplicación. En este contexto, el Desarrollo Dirigido por Modelos (MDD, Model Driven Development) [1, 2, 3, 4] es actualmente un desafiante paradigma del campo de Ingeniería de Software [5], situando a los modelos como artefactos principales durante todo el proceso de construcción de software.

Se propone aquí formalizar un marco de trabajo y herramientas de asistencia a la Ingeniería de Requerimientos Dirigida por Modelos (IRDM) [6, 7, 8], con el propósito de agilizar etapas iniciales de la línea de producción de software. Así se espera aportar nuevos mecanismos para la construcción de modelos CIM (Computational Independent Model) y la transformación automática a modelos PIM (Platform Independent Model) del paradigma MDD, creando un nuevo enfoque para el tratamiento de requerimientos en el proceso de IRDM.

Lo primero es desarrollar un lenguaje específico del dominio [9, 10] Salud (DSL_SALUD) a partir de estándares de Interoperabilidad y Modelado de Información para Historia Clínica Electrónica: FHIR de HL7 [11] y OpenEHR [12], considerando aspectos estáticos y dinámicos del dominio.

Luego se sigue con el desarrollo de una herramienta CASE que permita construir modelos a partir del lenguaje específico para Salud (DSL_SALUD), y que además permita automáticamente transformar estos modelos CIM hacia modelos PIM escritos en UML [13] , por ejemplo: diagramas de clases UML.

Seguir con la creación de un marco de trabajo conceptual y metodológico en el que se lleve a cabo Ingeniería de Requerimientos Dirigida por Modelos IRDM en el contexto del Desarrollo de Software Dirigido por Modelos MDD.

Finalmente desarrollar un estudio experimental en un Proyecto de Desarrollo de Software puntual, que permita evaluar la aplicación del marco conceptual y metodológico, junto con el uso de la herramienta CASE, con el objetivo de evaluar estos puntos:

- Analizar cómo la incorporación de Transformación de Modelos al Proceso Ingeniería de Requerimientos aporta específicamente a la trazabilidad de requerimientos y brinda agilidad al Proceso Desarrollo de Software.
- Analizar la incidencia de la incorporación de Transformación de Modelos al Proceso Ingeniería de Requerimientos en aspectos comunicacionales y aceptación del producto software (Usabilidad).
- Conocer el grado en que la incorporación de Transformación de Modelos al Proceso Ingeniería de Requerimientos adelanta las solicitudes de cambios de requerimiento a etapas tempranas del Proceso Desarrollo de Software.

II. Marco Teórico

A. Ingeniería de Requerimientos

La Ingeniería de Requerimientos (IR) es considerada

hoy un tema clave y fundamental de la Ingeniería de Software, esto se evidencia en el hecho de contar con su propio campo de trabajo y su avance continuo como disciplina.

El proceso de IR es un proceso iterativo que consta a su vez de tres grandes subprocesos: elicitación, especificación y validación de requerimientos. La IR logró expandirse para no sólo abordarse en etapas tempranas del ciclo de vida del software, sino en prácticamente todas las etapas.

La especificación funcional de requerimientos sistémicos puede hacerse tanto con lenguaje natural, utilizando reglas, formatos y plantillas de documentación (por ejemplo Especificación de Requerimientos de Software ERS de IEEE 830, y especificación de Casos de Uso), y también a través de modelos gráficos construidos con lenguajes de modelado como UML y SysML.

B. *Desarrollo de Software Dirigido por Modelos y Lenguaje Específico del Dominio (DSL)*

La Ingeniería de Software [4] tiene la problemática de construir software de calidad, que pueda ser capaz de sobrevivir a la evolución de sus requisitos funcionales, y que sea flexible a los cambios en la tecnología que lo sustenta, es actualmente contemplado en los principios del paradigma de desarrollo de software conocido como MDD [1, 2, 3]. La idea troncal de MDD, es obtener mediante transformaciones automáticas, modelos más específicos o concretos, a partir de otros más abstractos.

Uno de los beneficios en el desarrollo de software que conlleva aplicar MDD es la adaptación a los cambios tecnológicos ya que los modelos de alto nivel están libres de detalles de la implementación, lo cual facilita la adaptación a los cambios que pueda sufrir la plataforma tecnológica subyacente MDD utiliza cuatro tipos de modelos: CIM (Computation Independent Model), PIM (Platform Independent Model), PSM (Platform Specific Model) y CODE (código fuente de la aplicación).

Como se expresa la Figura 1, la idea del paradigma MDD, es obtener mediante transformaciones automáticas, modelos más específicos a partir de otros más abstractos; es decir, de un PIM obtener uno o varios PSM y de un PSM, obtener el código fuente en una tecnología específica.

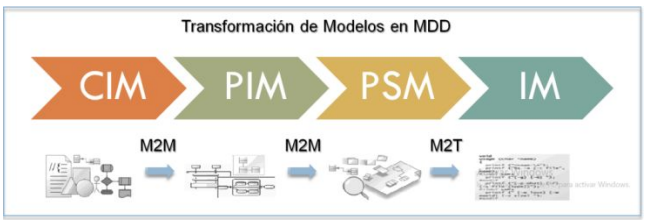

Fig. 1: Transformación de modelos en MDD.

Una propuesta concreta utilizada en el ámbito MDD es la idea de crear modelos para un dominio específico a través de lenguajes DSLs (por su nombre en inglés: Domain-Specific Language), focalizado y especializado para dicho dominio. Estos lenguajes permiten especificar la solución usando directamente conceptos del dominio del problema. Los productos finales son luego generados automáticamente desde estas especificaciones de alto nivel. La técnica más usada de para especificar un DSL es el metamodelado donde se define qué elementos pueden existir en el modelo.

C. *Ing. de Requerimientos Dirigida por Modelos*

La Ingeniería de Requerimientos Dirigida por Modelos (MDRE) [13, 14, 15] integra un conjunto de conocimientos que se aplican durante la primera etapa del desarrollo de software, apuntando a obtener un relevamiento del dominio y el problema en sí mediante modelos, lo más acertado posible, de manera de entender qué es lo que se requiere hacer. Esta etapa es indispensable si se pretende lograr una buena solución tecnológica acorde a las necesidades de los usuarios.

MDRE es parte de la Ingeniería de Software Basada en Modelos, ya que también apunta a la conceptualización y especificación de los requerimientos exclusivamente con modelos. La descripción de requerimientos con modelos más utilizada en la industria del software, es la especificación de escenarios de los casos de uso como Diagramas de Interacción de UML, más exactamente, Diagramas de secuencia UML y Diagramas de actividades UML.

D. *Metamodelos y Metalenguajes*

La definición de un lenguaje de modelado establece qué elementos pueden existir en un modelo. Usando un lenguaje de modelado, podemos crear modelos que especifican qué elementos pueden existir en un sistema.

Se puede describir un lenguaje por medio de un modelo, llamado "metamodelo" del lenguaje, que describe qué elementos pueden ser usados en el lenguaje y cómo deben ser conectados.

Un metamodelo es también un modelo y por lo tanto debe estar escrito en un lenguaje bien definido que llamamos "metalenguaje".

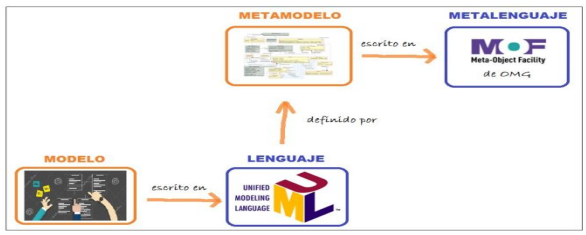

Fig. 2: Modelo, Lenguaje, Metamodelos y Metalenguajes

Siguiendo la Figura 2, para el caso del lenguaje de modelado UML, tenemos por un lado modelos escritos en UML construidos en el contexto de un desarrollo de software. Por otro lado, tenemos el lenguaje UML que está definido por su metamodelo, el metamodelo de UML está escrito en el metalenguaje Meta-Object Facility MOF. El metamodelo de UML, y sus sucesivas versiones, son publicados como estándar por OMG [16].

E. *Modelos de Información para Historia de Salud Electrónica*

Existen estándares internacionales que ofrecen modelos de información para Historia de Salud Electrónica. Los más destacados son los estándares abiertos FHIR de

HL7 y OpenEHR. Estos estándares, surgieron por la gran necesidad de intercambio de información entre sistemas informáticos que existe en el área de salud, esto es, interoperabilidad entre sistemas.

HL7 Internacional es una organización fundada en 1987 que se dedica al desarrollo de estándares para el ámbito de salud. El objetivo es minimizar las incompatibilidades entre sistemas de información en salud, permitiendo el intercambio de datos entre aplicaciones heterogéneas, independientemente de su plataforma tecnológica, y fomentando la interacción entre las mismas.

OpenEHR es un estándar abierto que describe la administración y almacenamiento de información sanitaria en forma de informes de historia clínica electrónica (HCE), que cuenta con una comunidad virtual que de esta forma aporta a la interoperabilidad entre sistemas de información de salud.

III. Desarrollo del Lenguaje Específico del Dominio DSL_SALUD

Comenzamos con el Desarrollo de un Lenguaje Específico del Dominio (DSL) para Salud, al que llamaremos DSL_SALUD, a partir de estándares de Interoperabilidad y Modelado de Información para Historia Clínica Electrónica: FHIR de HL7 y OpenEHR.

Desde el punto de vista del estudio profundo de los conceptos propios del dominio, se analizaron los modelos de información que brindan los estándares FHIR de HL7 y OpenEHR. Se hizo un recorte del modelo de información de FHIR considerando recursos de información fundamentales: Resource, DomainResource, Patient, Practitioner, Encounter y EpisodeOfCare. Con estas clases se podrá armar un DSL_SALUD incial que luego iremos ampliando en futuras iteraciones del proceso de construcción del DSL.

Desde el punto de vista tecnológico de implementación del DSL, se evaluaron distintos entornos de trabajo para la construcción de DSLs, entre los cuales destacamos: Domain-Specific Language Tools de Microsoft, Sirius y DSL Toolkit, siendo los dos últimas parte del proyecto Eclipse Modeling Project. Elegimos como entorno a la herramienta DSL Toolkit por ser más versátil en cuanto a las posibilidades de modelado gráfico y textual, y porque cuenta con la posibilidad de utilizar los lenguajes de transformación de modelos ATL y QVT en forma integrada en el mismo entorno.

El paso siguiente será llevar a cabo el propio desarrollo tecnológico del metalenguaje DSL_SALUD utilizando Eclipse Modeling Framework, EMF, dentro de DSL Toolkit. EMF nos permitirá modelar y desarrollar el núcleo de nuestro DSL_SALUD. Utilizando EMF desarrollaremos la sintaxis abstracta de nuestro DSL_SALUD. Además permitirá aplicar al mismo distintas restricciones OCL y validar los modelos. En este paso se construirá el metamodelo DSL_SALUD como una estructura ECORE.

Como puede verse en la Figura 3, nuestro DSL_SALUD o Metamodelo de Dominio Salud resultante estará compuesto por:

- La Definición del Diagrama con las interacciones entre las clases que componen nuestro dominio.
- Las Reglas de Transformación M2M.
- Las Reglas de Transformación M2T.
- Las Definiciones Textuales de la Sintaxis.

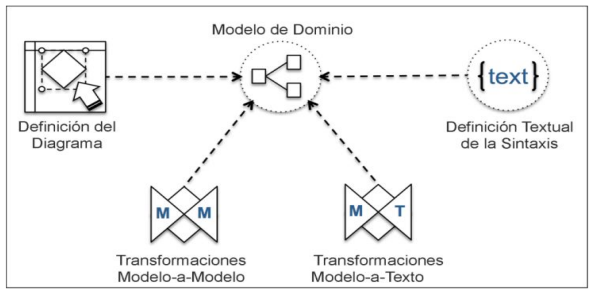

Fig. 3: Metamodelo del Dominio Salud: DSL_SALUD

IV. Diseño de Herramienta CASE

Cuando ya se cuente con el DSL_SALUD construido y disponible para utilizar, se continuará con el desarrollo tecnológico de una herramienta CASE.

Utilizando las posibilidades de desarrollo que el paquete de librerías y plugins de DSL Toolkit ofrece, nos centraremos y utilizaremos las siguientes:

- GMF - Graphical Modeling Framework: será utilizado para generar la herramienta de modelado gráfico de nuestro DSL_SALUD, ya que ofrece librerías y un entorno de ejecución para asistir con este desarrollo.
- TMF - Textual Modeling Framework: entorno de desarrollo para modelar lenguajes textualmente.
- M2M - Model-to-Model Transformations: se utilizará la librería QVT para escribir las transformaciones desde nuestro modelos hacía otros modelos, como por ejemplo: Diagrama de Actividades UML.
- M2T - Model-to-Text Transformations: se utilizará la librería Xpand para generación de código a partir de los modelos de entrada.
- Amalgation: librería para empaquetar todos nuestros archivos finales y generar un plugin que pueda ser integrado y usado en otros entornos.

La herramienta CASE será construida para que asista al Ingeniero de Software en las siguientes cuestiones:

- construcción de modelos escritos en DSL_SALUD) mediante interfaz gráfica para modelado;
- ejecutar transformaciones automáticas desde estos modelos escritos en lenguaje DSL_SALUD hacia modelos escritos en otros lenguajes, por ejemplo: diagramas de clases UML, y también transformar éstos últimos a modelos PSM, como el modelo relacional.

Todo esto se hace con la intención de que los Ingenieros de Software, especializados en soluciones IT para Salud, puedan hacer uso tanto del metalenguaje DSL_SALUD como de la herramienta CASE para llevar a cabo la construcción de modelos que representen los requerimientos propios para sus desarrollos de aplicaciones software para Salud, es decir, de esta forma poner en práctica Ingeniería de Requerimientos Dirigida por Modelos IRDM en el contexto del Desarrollo Dirigido por Modelos MDD.

Como puede verse en la Figura 4, el Ingeniero de Software diseñará un modelo propio para una aplicación software que estará en el nivel M1 y que será instancia del metamodelo DSL_SALUD del nivel M2.

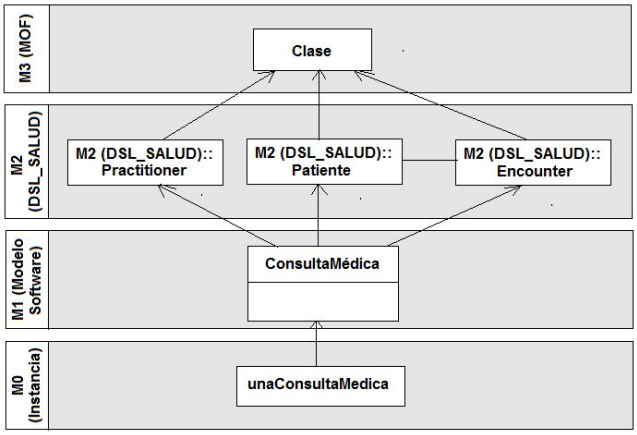

Fig. 4: Niveles de modelado y metamodelado para DSL_SALUD

Finalmente se realizará un estudio experimental del uso de esta herramienta CASE en el contexto de un proyecto de desarrollo de software real en el ámbito de la Salud.

V. Marco Metodológico

El aporte metodológico no es exclusivo para el dominio de la Salud, dado que, de la misma forma podría implementarse en otro dominio. Transfiriendo el conocimiento, la metodología de trabajo y la herramienta CASE a equipos de desarrollo reales, se podrá analizar el aporte realizado por la construcción y uso del metalenguaje a través de la herramienta CASE, teniendo en cuenta aspectos claves del Proceso de Desarrollo de Software.

Llevar a cabo un estudio experimental en un Proyecto de Desarrollo de Software puntual, que permita evaluar la aplicación del marco conceptual y metodológico, junto con el uso de la herramienta CASE.

Finalmente se realizará un estudio experimental del uso de esta herramienta CASE en el contexto de un proyecto de desarrollo de software real en el ámbito de la Salud. Así se podrán analizar las consecuencias del uso de esta herramienta. Para llevar a cabo este estudio experimental, por un lado, en una primera etapa, se asistirá en la implementación y uso de la herramienta CASE, se llevarán a cabo entrevistas no estructuradas con preguntas abiertas a los stakeholders y a los miembros del equipo de desarrollo, y por otro lado en una segunda etapa, se realizarán cuestionarios estructurados para completar el estudio. Luego se buscará cotejar ambos resultados para llevar a cabo el análisis de los datos obtenidos. Se redactarán informes de experiencia e informes técnicos, como así también publicaciones en congresos especializados a nivel nacional.

VI. Conclusiones y Trabajo Futuro

Este proyecto es la continuidad de un PID anterior llamado “Modelado Ágil para la Producción de Software MAPS” por el que se realizaron publicaciones [17, 18, 19, 20] siempre con la temática principal: Desarrollo Dirigido por Modelos.

En lo que va de proyecto ya se ha demostrado que es factible su realización desde todo punto de vista, por lo que se comenzará a desarrollar el DSL_SALUD y la herramienta CASE, que van a ser distribuidos como un plugin para Eclipse y que permitirán construir modelos a partir de nuestro metamodelo DSL_SALUD. Dicho plugin permitirá llevar a cabo una correcta modelización y transformación automática desde un modelo construido e instanciado visualmente a partir de nuestro DSL, hacia distintos diagramas UML como por ejemplo Diagramas de Actividades, contribuyendo así a la generación de un modelo PIM necesario para la etapa de inicio del proceso de desarrollo de sistemas orientados a objetos.

Referencias

[1] Pons, C., Giandini, R. y Pérez, G. “Desarrollo de Software Dirigido por Modelos: conceptos teóricos y su aplicación práctica”. 1ª Edición 2010. EDULP & McGraw-Hill.

[2] J. García, F. O. García, V. Pelechano, A. Vallecillo, J.M. Vara, C. Vicente-Chicote. “Desarrollo de Software Dirigido por Modelos”. ISBN 978-84-9964-215-4 (2013).

[3] F. Durán Muñoz, J. Troya Castilla, A. Vallecillo Moreno. “Desarrollo de software dirigido por modelos”. Universitat Oberta de Catalunya (2013).

[4] MDA, “Model Driven Architecture Guide” (OMG). V.2.0, 2014. Disponible en http://www.omg.org/cgi-bin/doc?ormsc/14-06-01

[5] I. Sommerville, “Ingeniería de Software”, 7ma. edición, Pearson, 2005. ISBN: 84-7829-074-5.

[6] Wiegers, K. E. (2003) Software Requirements, Microsoft Press, Redmond.

[7] Young, R., (2001) Effective Requirements Practices, Addison-Wesley.

[8] Macaulay, L. A. (2001) Requirements Engineering, Springer Verlag.

[9] “Domain-Specific Modeling,, Enabling Full Code Generation”, STEVEN KELLY, JUHA-PEKKA TOLVANEN, IEEE COMPUTER SOCIETY - Ed. 2008

[10] A Domain-Specific Language Toolkit, Richard C. Gronback - Adison Wesley - Ed. 2009.

[11] Especificación de FHIR de HL7, Realease 3. Disponible en: https://www.hl7.org/fhir

[12] OpenEHR Information Model, 2017. Disponible en: http://www.openehr.org/releases/RM/latest/docs/ehr/ehr.html

[13] Línea de Investigación del CEIS - Centro de Estudios de Ingeniería de Software http://www.ceisufro.cl/index.php?id=45

[14] A Systematic Review of the Use of Requirements Engineering Techniques in Model-Driven Development. MODELS 2010: Model Driven Engineering Languages and Systems pp 213-227. Grzegorz Loniewski, Emilio Insfran, Silvia Abrahão https://link.springer.com/chapter/10.1007%2F978-3-642-16129-2_16?LI=true

[15] Neil A. M. Maiden, Sara V. Jones, Sharon Manning, John Greenwood, L. Renou. Model-Driven Requirements Engineering: Synchronising Models in an Air Traffic Management Case Study. CAiSE 2004: Advanced Information Systems Engineering pp 368-383 https://link.springer.com/chapter/10.1007/978-3-540-25975-6_27

[16] UML, “Unified Modeling Language Infrastructure” (OMG). Versión 2.4.1, 2011. Disponible en http://www.omg.org/spec/UML/2.4/

[17] Ariste Cecilia, Ponisio Julieta, Nahuel Leopoldo, Giandini Roxana. JAIIO - ASSE (2015). “Diseñando Transformaciones de Modelos CIM / PIM: desde un enfoque de negocio hacia un enfoque de sistema”.

[18] Informe técnico “Especificación de la Transformación de Proceso BPD en BPMN a Diagrama de Actividades UML” PID MAPS 2015. Disponible en: http://maps.frlp.utn.edu.ar/

[19] Giandini, Roxana, Nahuel, Leopoldo, Rocca, Leandro, Caputi, Matías, Zugnoni, Ivan - CoNaIISI (2014). “Implementando Transformación de Modelos utilizando MOSKitt Tool en adhesión al Paradigma MDD”. Congreso Nacional de Ingeniería en Informática/Sistemas de Información.

[20] L. Nahuel, E. Santanera, M. C. Ariste, L. Rocca, R. Giandini. Integración Metodológica para el Desarrollo de Tecnologías Software Dirigidas por Modelos y Basadas en Procesos de Negocio. CIINDET 2014.