Herramientas CARE (Computer Aided Requirements Engineering) **para Lenguaje Específico de Dominio en ámbitos de Salud**

Tesista de Posgrado: **María Cecilia ARISTE** [1] , **Leandro ROCCA** [1]

Directores: **Roxana GIANDINI**[1,2] **, Leopoldo NAHUEL**[1]

[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

{mariste83, leonahuel, roxanagiandini}@gmail.com

Resumen—Este artículo es un reporte de experiencia de parte del camino recorrido por investigadores del Grupo GIDAS de UTN FRLP en el Desarrollo de un Lenguaje Específico del Dominio (DSL) para Salud, al que llamamos DSL_SALUD, y el diseño de una herramienta CASE que implemente dicho lenguaje. Esta herramienta 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, por ejemplo: diagrama de clases 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, es decir, de esta forma ponen en práctica Ingeniería de Requerimientos Dirigida por Modelos IRDM en el contexto del Desarrollo Dirigido por Modelos MDD. Este aporte metodológico es independiente del dominio de Salud porque de la misma forma podría implementarse en otro dominio, siempre con el objeto de mejorar aspectos claves del Proceso de Desarrollo de Software como la Agilidad del proceso mismo, la Trazabilidad de requerimientos, la Usabilidad del producto software final, entre otros.

Palabras Claves—Ingeniería de Requerimientos Dirigida por Modelos IRDM, Lenguaje Específico del Dominio DSL, Metalenguajes, Desarrollo Dirigido por Modelos MDD, Herramienta CASE, Proceso de Desarrollo de Software, Agilidad, Trazabilidad de Requerimientos, Usabilidad del Software.

1 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, disociando negocio-tecnología y sustituyendo al código de lenguajes de programación por modelos con distintos niveles de abstracción.

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 independientes de la computación CIM hacia modelos independientes de la plataforma PIM escritos en UML [13] , por ejemplo: diagrama 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 optimizando así la gestión de recursos del proyecto de desarrollo.

2 MARCO TEÓRICO

2.1 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. Al parecer, el principal motivo del avance de esta disciplina, tiene que ver con el hecho de que requerimientos bien especificados y rastreables, entre los artefactos o productos de desarrollo, ayudan a una eficiente gestión del cambio, evolución y mantenimiento del producto software, llegando así a productos de mejor calidad y escalables.

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.

2.2 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 (por sus siglas en inglés: Model Driven Software Development) [1, 2, 3]. Este paradigma ofrece mejorar los procesos de construcción de software. La idea troncal de MDD, es obtener mediante transformaciones automáticas, modelos más específicos o concretos, a partir de otros más abstractos.

Los postulados básicos de MDD son los siguientes:
los modelos asumen un rol protagónico en el proceso de desarrollo del software;
los modelos pasan de ser entidades contemplativas para convertirse en entidades productivas a partir de las cuales se deriva la implementación en forma automática.

La iniciativa MDD promueve:

_ el uso de un mayor nivel de abstracción tanto en la especificación del problema a resolver como de la solución correspondiente, en relación con los métodos tradicionales de desarrollo de software.

_ el aumento de confianza en la automatización asistida por computadora para soportar el análisis, el diseño y la

ejecución.

_ el uso de estándares industriales como medio para facilitar las comunicaciones, la interacción entre diferentes aplicaciones y productos, y la especialización tecnológica.

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 o la arquitectura de implementación.

MDD utiliza cuatro tipos de modelos (de mayor a menor nivel de abstracción): 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 (según la tecnología de implementación) y de un PSM, obtener el código fuente en una tecnología específica.

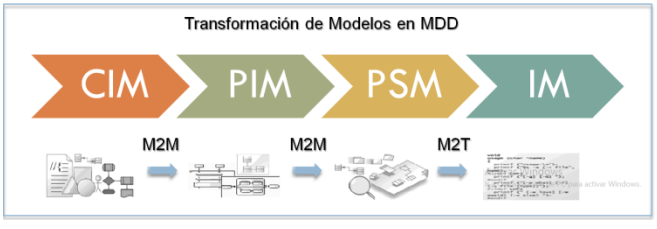

Figura 1: Transformación de modelos en MDD.

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. Por ejemplo, en el metamodelo DSL_SALUD encontraremos a "Paciente", "Encuentro", "Profesional de Salud", etc. que luego aparecerán instanciadas en un modelo UML y también en un modelo relacional.

2.3 INGENIERÍA DE REQUERIMIENTOS DIRIGIDA POR MODELOS (MDRE)

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, y es por esto, que MDRE toma fuerte relevancia en el campo de la ingeniería de software.

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.

MDRE es una disciplina que se posiciona en un lugar muy importante en los proyectos de software, especialmente en aquellos proyectos robustos, con alto nivel de complejidad y de construcción a largo plazo, debido a una serie de aspectos como:

- Trazabilidad de requerimientos entre modelos.

- Practicidad de versionado de los modelos para validación con los usuarios.

- Reuso de artefactos de modelado en la creación de patrones funcionales del dominio.

- Navegación desde los requerimientos hacia sus correspondientes artefactos de implementación (clases, tablas, decisiones de diseño, interfaz de usuario, entre otros.) y a la inversa.

- Gestión de mantenimiento y registro de cambios de los artefactos de modelado en un proyecto que evoluciona a través de sus distintas iteraciones del proceso.

2.4 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".

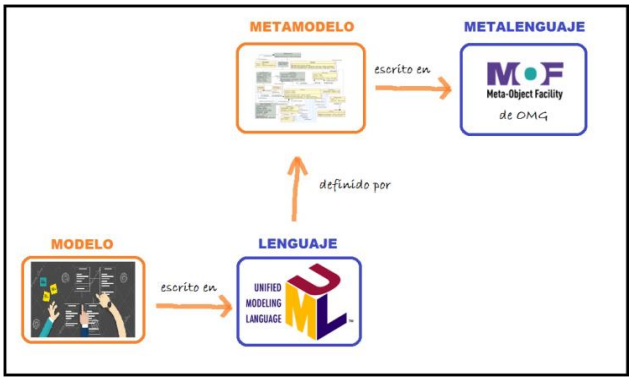

Figura 2: Modelo, Lenguaje, Metamodelos y Metalenguajes.

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].

El metamodelo describe la sintaxis abstracta del lenguaje. Esta sintaxis es la base para el procesamiento automatizado (basado en herramientas) de los modelos, de lo que destacamos como relevante, las transformación automáticas entre modelos. Por otra parte, la sintaxis concreta es definida mediante otros mecanismos, es la interfaz para el modelador e influye fuertemente en el grado de legibilidad de los modelos.

La arquitectura de cuatro capas de modelado de OMG cuenta con estos niveles:

_ Nivel M0: Instancias. Se trata de todas las instancias reales del sistema, es decir, los objetos de la aplicación.

_ Nivel M1: Modelo del sistema. Representa el modelo de un sistema de software. Los conceptos del nivel M1 representan categorías de las instancias de M0.

_ Nivel M2: Metamodelo. En este nivel aparecen conceptos claves como Clase, Atributo y Operación. Por ejemplo: metamodelo de UML.

_ Nivel M3: Meta-metamodelo. Es el nivel más abstracto, que permite definir metamodelos concretos. Dentro del OMG, MOF es el lenguaje estándar de la capa M3.

Para MDD el uso del metamodelado es importantísimo por estas razones:

_ es necesario contar con un mecanismo para definir lenguajes de modelado sin ambigüedades

_ es necesario que una herramienta de transformación de modelos pueda leer, escribir y entender los modelos.

_ las reglas de transformación que constituyen una definición de una transformación describen como un modelo en un lenguaje fuente puede ser transformado a un modelo en un lenguaje destino. Estas reglas usan los metamodelos de los lenguajes fuente y destino para definir la transformación.

_ la sintaxis de los lenguajes en los cuales se expresan las reglas de transformación también debe estar formalmente definida para permitir su automatización. En este caso también se utilizará la técnica de metamodelado para especificar dicha sintaxis.

2.5 MODELOS DE INFORMACIÓN PARA HISTORIA DE SALUD ELECTRÓNICA

Existen estándares internacionales que ofrecen modelos de información para Historia de Salud Electrónica que tienen una gran divulgación y muchos casos de éxito en desarrollos reales para la industria de software específica para Salud. Los más destacados son HL7 y OpenEHR. Estas organizaciones 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.

3 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, considerando aspectos estáticos y dinámicos del dominio.

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 las 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.

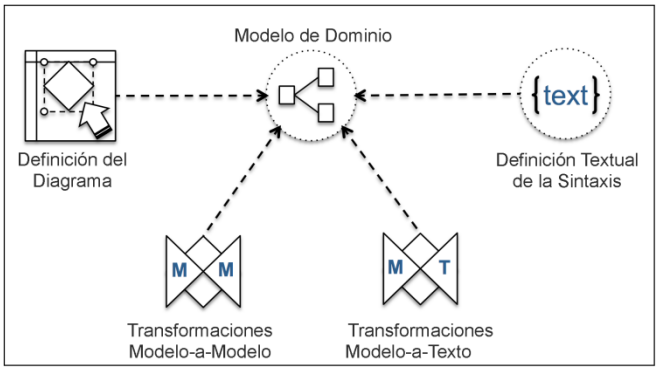

Figura 3: Metamodelo del Dominio Salud, DSL_SALUD.

4 DISEÑO DE HERRAMIENTA CASE

Cuando ya se cuente con el DSL_SALUD construído 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 construída para que asista al ingeniero de software en las siguientes cuestiones:

_ construcción de modelos escritos en DSL_SALUD (que a su vez está basado en estándares internacionales de modelado de información para SALUD) mediante interfaz gráfica para modelado.

_ ejecutar transformaciones automáticas desde estos modelos escritos en lenguaje DSL_SALUD hacia modelos escritos otros lenguajes, es decir, automáticamente transformar estos modelos independientes de la computación CIM hacia modelos independientes de la plataforma PIM escritos en UML, por ejemplo: diagrama de clases UML, y también transformar estos a modelos PSM como el modelos 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.

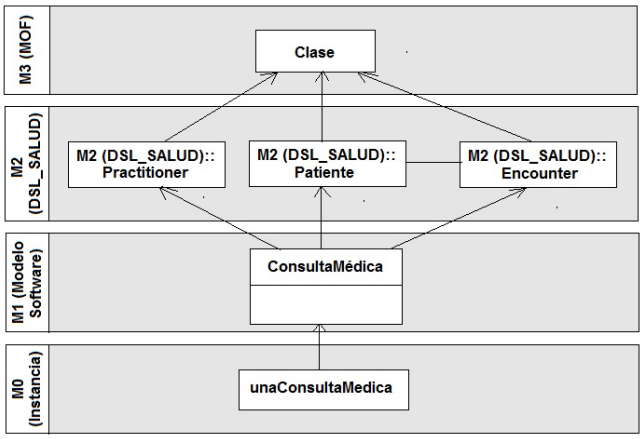

Figura 4: Niveles de modelado y metamodelado para DSL_SALUD

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.

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.

5 MARCO METODOLÓGICO

Crear 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.

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 como la Agilidad del proceso mismo, la Trazabilidad de requerimientos a lo largo de todos los productos del desarrollo, la Usabilidad del producto software final, entre otros.

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, con estos sub-objetivos:

_ 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 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 optimizando así la gestión de recursos del proyecto de desarrollo.

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.

6 CONCLUSIONES Y TRABAJO FUTURO

Esta publicación surge del trabajo en conjunto del equipo de investigadores del PI&D:"Herramientas de soporte a la Ingeniería de Requisitos Dirigida por Modelos: desde las necesidades de negocio hacia los requisitos de software", proyecto parte del Grupo GIDAS del Departamento de Sistemas de Información de la Facultad Regional La Plata - UTN. El PID cuenta con 9 miembros entre coordinadores, investigadores graduados y alumnos, y está homologado por Rectorado UTN.

Las líneas de Investigación del PID son: Ingeniería de Requisitos Dirigida por Modelos, Procesos de Negocio, Lenguajes Específicos de Dominio, Herramientas CASE.
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, 21] 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 que permite construir modelos a partir de nuestro metamodelo DSL_SALUD, extendido del modelo OpenEHR/FHIR, disponer de la funcionalidad de transformaciones automáticas M2M y M2T (escritas en QVT y Xpand respectivamente).

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 Diagrama 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 BIBLIOGRÁFICAS

[1] Pons, C., Giandini, R. y Pérez, G. "Desarrollo de Software Dirigido por Modelos: conceptos teóricos y su aplicación práctica". 1°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, Roca, 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.

[21] L. Nahuel, E. Santanera, L. Rocca, C. Ariste, R. Giandini. Aportes de las Tecnologías para Gestión de Procesos de Negocio al Desarrollo de Software Dirigido por Modelos. HCITISI 2013, (ISBN 978.88.96.471.25.8).