Revista Cubana de Ciencias Informáticas
Vol. 14, No. 4, Octubre-Diciembre, 2020
ISSN: 2227-1899 | RNPS: 2301
http://rcci.uci.cu
Pág. 210-225

Tipo de artículo: Artículo original
Temática: Ingeniería y Gestión de Software
Recibido: 01/09/2020 | Aceptado: 20/11/2020

CASE jMDA de Arquitectura Dirigida por Modelos para Sistemas de Información

CASE jMDA for Model Driven Architecture for Information Systems

Rosendo Moreno-Rodríguez [1]* https://orcid.org/0000-0002-2918-0207

[1] Universidad Central "Marta Abreu" de Las Villas. Carretera a Camajuaní km. 5 ½. rosendo@uclv.edu.cu

*Autor para la correspondencia. (rosendojmr@gmail.com)

RESUMEN

El desarrollo de un software es un proceso complejo y difícil de gestionar en el que intervienen múltiples elementos. El ciclo de vida de este producto se ve permeado de disímiles problemáticas que lo afectan, hasta no responder a las necesidades identificadas. El Object Management Group, ha prestado especial atención al problema de interoperabilidad e integración de software, definiendo numerosas especificaciones y estándares, y en el 2001, establece el marco de trabajo "Model Driven Architecture", como arquitectura para el desarrollo de aplicaciones. En este paradigma de desarrollo, los modelos guían todo el proceso. Varias herramientas se han desarrollado a partir de esta idea internacionalmente, pero muchas son de autor o privativas, y además no cubren el modelo completo. En la Universidad Central de Las Villas se ha estado desarrollando una herramienta jMDA que pretende cumplir con las tres transformaciones establecidas teóricamente, entre las cuatro fases, a partir de la creación de tres módulos independientes pero que tienen comunicación entre todos, lo que permite cumplir con un ciclo completo del MDA, constituyendo de hecho

en una herramienta CASE totalmente soberana y basada en software libre. El primer módulo cuenta con 2 versiones, el segundo ya tiene 5, y el tercero 4, todas desarrolladas a partir de investigaciones del autor con la colaboración de estudiantes terminales de Ciencia de la Computación e Ingeniería Informática.

Palabras clave: Arquitectura Dirigida por Modelos (MDA); Ingeniería del Software; Herramienta CASE; Sistemas de Información; Lenguaje Unificado de Modelado (UML).

ABSTRACT

The development of software is a complex and difficult process in which multiple elements intervene. The software life cycle is permeated with dissimilar problems that affect it, until it does not respond to the identified needs. The Object Management Group has paid special attention to software interoperability and integration problem, defining numerous specifications and standards, and in 2001, established the framework "Model Driven Architecture", as an architecture for the development of applications. In this development paradigm, models guide the entire process. Several tools have been developed from this idea internationally, but many are proprietary, and also do not cover the complete model. At the Central University of Las Villas, a jMDA tool has been developed that aims to comply with the three theoretically established transformations, among the four phases, from the creation of three independent modules that have communication between all of them, which allows to comply with a complete MDA cycle, in fact constituting a fully sovereign CASE tool based on free software. The first module has 2 versions, the second already has 5, and the third 4, all developed from the author's research with the collaboration of terminal students of Computer Science and Computer Engineering.

Keywords: Model Driven Architecture; Software Engineering; CASE Tool; Information System; Unified Modeling Language.

Revista Cubana de Ciencias Informáticas
Vol. 14, No. 4, Octubre-Diciembre, 2020
ISSN: 2227-1899 | RNPS: 2301
http://rcci.uci.cu
Pág. 210-225

Introducción

La interoperabilidad y la integración son elementos cada vez más necesarios en los sistemas de software que se construyen (Booch, 2001). Los cambios constantes en el negocio y en las tecnologías de implementación exigen de esfuerzos constantes en la búsqueda de mejores modos de diseñar las aplicaciones, de integrarlas y adaptarlas de manera continua a los nuevos requisitos que surjan.

El Object Management Group (OMG) en búsqueda de una alternativa para resolver estos problemas en el año 2001 establece el framework Arquitectura Dirigida por Modelos (MDA), nuevo paradigma en el que los modelos constituyen la guía de todo el proceso de desarrollo de software (Kleppe y otros, 2005). Con este framework OMG demuestra que el enfoque de modelado es una forma potente de especificar sistemas, permitiendo que se obtengan beneficios importantes en aspectos fundamentales como son la productividad, la portabilidad, la interoperabilidad y el mantenimiento.

A pesar de las ventajas de esta nueva filosofía de trabajo resulta imprescindible la existencia de herramientas que le den soporte. En este contexto el Centro de Investigaciones de Informática de la UCLV desde el año 2010 se propone desarrollar una herramienta que implemente el paradigma MDA por sus tres módulos de conversión de manera independiente. En este trabajo se hace referencia a los módulos correspondientes a las transformaciones Modelo Independiente de la Computación (CIM) al Modelo Independiente de la Plataforma (PIM), PIM al Modelo Específico de la Plataforma (PSM), y PSM-Código que da posibilidades de crear código en Java, Python y C#. Cada una de estos módulos se han desarrollado en versiones cada una de las cuales logra implementar nuevas opciones y mejoras en el ambiente gráfico, los algoritmos de transformación y reglas, que permiten establecer que ya se tiene una herramienta CASE totalmente aplicable en empresas de desarrollo de software, así como en la docencia universitaria.

La industria de software nacional puede contar entonces con una herramienta profesional para software del tipo Sistema de Información, que supera las limitantes con que cuentan las existentes en la actualidad, ya sea el pago de una licencia en el caso de las propietarias o limitantes de orden técnico en las de código abierto. Por lo que significaría un aporte invaluable al proceso de desarrollo de software en Cuba la implementación de una herramienta con estas características.

Revista Cubana de Ciencias Informáticas
Vol. 14, No. 4, Octubre-Diciembre, 2020
ISSN: 2227-1899 | RNPS: 2301
http://rcci.uci.cu
Pág. 210-225

Metodología Computacional

Materiales y Métodos

El presente trabajo es una investigación aplicada. Se usaron métodos y técnicas de la Ingeniería del Software como el modelado UML y la programación en Java. Para el desarrollo de la herramienta jMDA se utilizó el entorno de desarrollo integrado libre *NetBeans*. Para la codificación se utilizaron bibliotecas contenidas en el JDK versión 8 tales como: *swing y awt*, para manipular todos los componentes visuales de la aplicación, así como la biblioteca *MxGraph*, sobre la cual se desarrolló el ambiente gráfico que permite crear y modificar continuamente los diagramas UML tanto en el modelo PIM como en el PSM. Además, con esa biblioteca se logró la importante opción de guardar en un formato específico que posibilita la conectividad entre los diferentes módulos.

Análisis del proceso de desarrollo de software MDA

El desarrollo de sistemas de software es una labor compleja en la que intervienen múltiples elementos a lo largo del proceso (Sommerville, 2011). La diversidad de lenguajes y técnicas de programación, así como los disímiles entornos integrados de desarrollo crean una indecisión en el momento de su selección. En la actualidad la selección de una tecnología o una combinación de estas es una tarea compleja y permeada de incertidumbre. Diversas investigaciones han demostrado que es muy frecuente el fracaso de los proyectos de software (Pressman, 2010) debido a los continuos cambios que sufren los requerimientos del usuario hasta la obtención del producto final. Ocasionando que se entreguen productos que no satisfacen las necesidades del cliente o que los sistemas deban implementarse una y otra vez.

En los enfoques tradicionales de software, incluso las nuevas tendencias de enfoques ágiles, en cada etapa del ciclo de vida del producto participan diferentes actores que asumen distintos roles, lo que provoca que en cada fase se hagan ajustes o interpretaciones de acuerdo a las expectativas de su ejecutor, lo que a su vez conlleva a que no se obtenga lo que realmente se quería. Esto ha impulsado la búsqueda de soluciones que permitan, de manera asistida por computadoras, realizar conversiones invisibles a los usuarios para

Revista Cubana de Ciencias Informáticas
Vol. 14, No. 4, Octubre-Diciembre, 2020
ISSN: 2227-1899 | RNPS: 2301
http://rcci.uci.cu
Pág. 210-225

transformar modelos independientes de una plataforma en su equivalente en un modelo específico de una determinada plataforma de implementación (Quintero y otros, 2005). OMG no propone un modelo estándar de desarrollo de software para MDA. Sin embargo, tal modelo es necesario ya que puede permitir organizar los pasos necesarios para el desarrollo del software de una forma definida, sistemática, y repetible (Quintero-Anaya, 2007).

En este contexto surge MDA para brindar una alternativa que se centra en los modelos los cuales se encargan de guiar todo el proceso de desarrollo, permitiendo la generación de código de manera automatizada, a partir de modelos y las reglas de transformación establecidas (Durán Muñoz y otros, 2013). En la figura 1 se presenta el proceso de desarrollo de software siguiendo el paradigma MDA que incluye 4 modelos y 3 transformaciones. La investigación y desarrollo llevado a cabo incluye tres módulos, cada uno correspondiente a una de esas transformaciones entre dos modelos concretos.

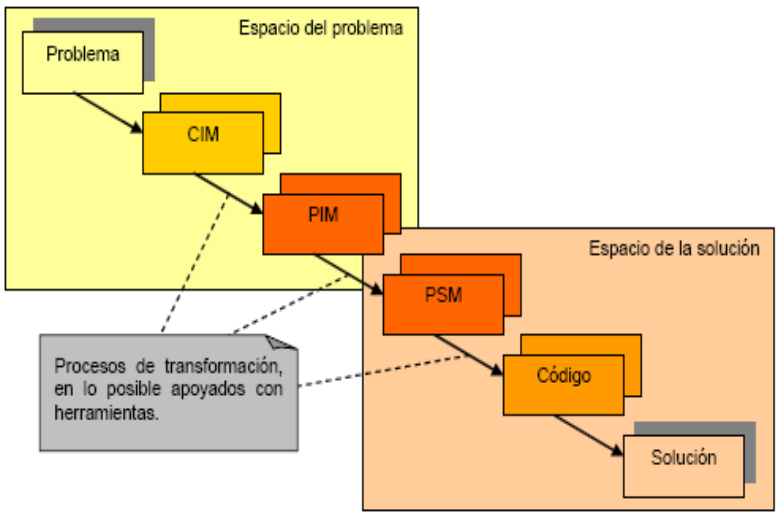

Fig. 1- Proceso de desarrollo con MDA.

(Peña-Moreno, 2011)

Revista Cubana de Ciencias Informáticas
Vol. 14, No. 4, Octubre-Diciembre, 2020
ISSN: 2227-1899 | RNPS: 2301
http://rcci.uci.cu
Pág. 210-225

Resultados y discusión

Módulo CIM-PIM

Es evidente que un módulo CIM es muy difícil y diverso de lograr. La estrategia seguida fue la de hacer un módulo CIM para un tipo específico de software, en este caso Sistemas de Gestión de Información. Independientemente de esto, en la literatura se encuentran varios enfoques de desarrollo de módulos CIM como:

1. Creación de un editor de texto en lenguaje natural (pseudocódigo realmente) donde el cliente o experto del problema, lo describa de manera semiestructurada, y a partir de esta descripción (historia de usuario), mediante programa de reconocimiento de ese lenguaje natural (con varias reglas y conocimiento incluido) se logre transformar esa descripción en los elementos necesarios de los diagramas UML de un módulo PIM.
2. Modelado del negocio del problema en cuestión en base a algún lenguaje gráfico que lo logre describir con exactitud (generalmente BPMN) y luego su transformación a los diagramas UML esenciales de un PIM. Esta variante tiene la dificultad del desconocimiento del lenguaje BPMN y las herramientas asociadas por parte de los expertos del problema.
3. Desarrollo de interfaces para específicos tipos de software, de preguntas-respuestas que posibiliten determinar automáticamente los elementos de los diagramas UML deseados. Esta fue la variante seleccionada para nuestro trabajo.

Debido a que el Modelo Independiente de la Plataforma (PIM) representa modelos que describen una solución de software que no contiene detalles de la plataforma concreta de implementación de la solución, se determinó que las transformaciones deseadas en esta versión darían lugar a diagramas de Casas de Uso, de Clases y de Actividades. La versión actual de este módulo es la 2 (Moreno-Ferrer, 2018).

Revista Cubana de Ciencias Informáticas
Vol. 14, No. 4, Octubre-Diciembre, 2020
ISSN: 2227-1899 | RNPS: 2301
http://rcci.uci.cu
Pág. 210-225

Interfaz de preguntas-respuestas

Como quiera que decidió utilizar la variante de preguntas-respuestas se realizó un estudio sobre las cuestiones básicas de análisis de requisitos para sistemas de gestión de información, entre otras las siguientes:

1. ¿Cuántas entidades usted desea controlar en su SI? Descríbalas.
2. ¿Qué características tienen esas entidades? Descríbalas.
3. ¿Cuáles son las funcionalidades deseadas para su SI? Nómbrelas.
4. ¿Quiénes serían los encargados de interactuar con esas funcionalidades? Defínalos.

En otra parte de la interfaz pueden preguntarse otros aspectos relativos a características propias del software, como si se desea una interfaz basada en menús y formularios o por Web, etc.

Breve descripción del módulo

Supongamos que se desea modelar un sistema para controlar la información asociada a los estudiantes y profesores pertenecientes a la Facultad de Matemática-Física-Computación; para ello se describirá paso a paso cómo se logra el resultado de la anterior problemática utilizando la herramienta jMDA módulo CIM-PIM.

Paso 1: Crear el nuevo formulario donde primeramente el analista debe seleccionar el nodo en el árbol del proyecto correspondiente al diagrama que desea modelar, luego hacer doble click sobre este e introducir el nombre del nuevo formulario que desea crear.

Paso 2: Responder las preguntas del formulario. Las Figuras 2a y 2b muestran el ambiente de preguntas-respuestas que se diseñó para lograr tanto el diagrama de casos de uso, como el diagrama de clases del modelo CIM.

Revista Cubana de Ciencias Informáticas
Vol. 14, No. 4, Octubre-Diciembre, 2020
ISSN: 2227-1899 | RNPS: 2301
http://rcci.uci.cu
Pág. 210-225

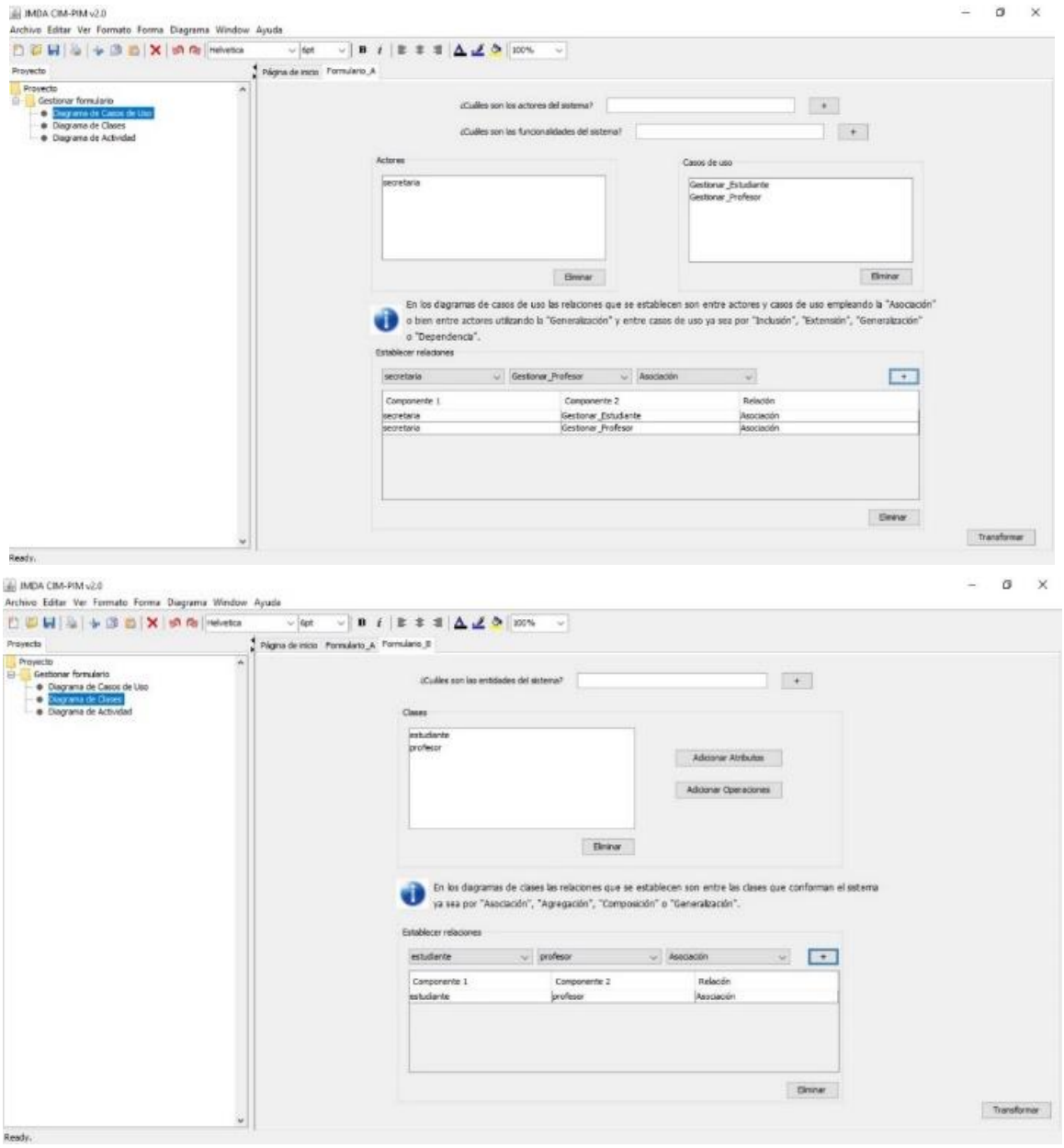

Fig. 2 - Ambiente de preguntas-respuestas para los elementos de Diagrama de Casos de Uso y de Clases.

De la misma manera existe otro formulario para determinar elementos de un Diagrama de Actividades que describa algún caso de uso o funcionalidad.

Revista Cubana de Ciencias Informáticas
Vol. 14, No. 4, Octubre-Diciembre, 2020
ISSN: 2227-1899 | RNPS: 2301
http://rcci.uci.cu
Pág. 210-225

Paso 3: Transformar al modelo PIM. Una vez que el formulario sea respondido el analista puede pasar a seleccionar el botón "Transformar" el cual realiza la transformación del modelo CIM al PIM. En las Figuras 3a y 3b se muestran los diagramas de casos de uso y de clases obtenidos al realizar la transformación correspondiente.

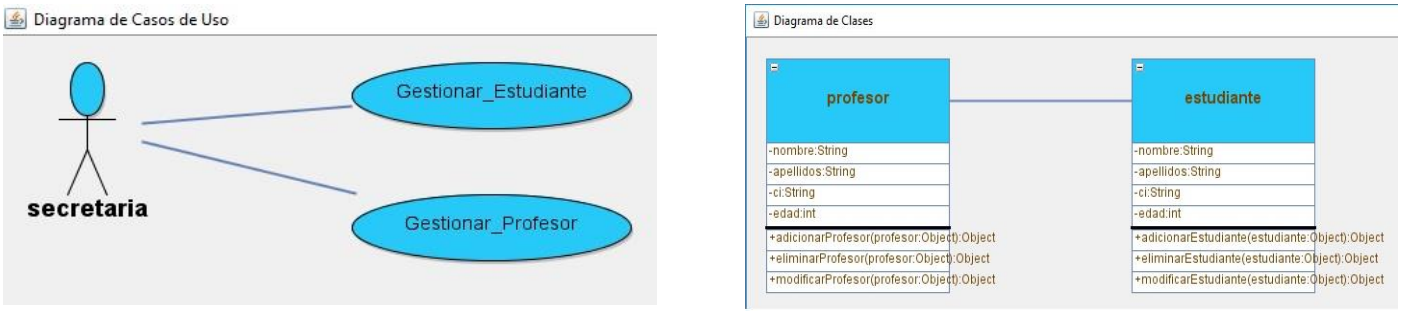

Fig. 3 - Diagramas de Casos de Uso y de Clases del modelo PIM.

Módulo PIM-PSM

Este módulo puede obtener los diagramas definidos en el módulo anterior o funcionar independientemente para crear los diagramas de casos de uso, clases, actividades, secuencia, y estados, para un problema típico de desarrollo de un Sistema de Información con Bases de Datos relacionales.

Posteriormente es posible generar automáticamente estos diagramas en ambiente PSM, lo que implica adicionar diferentes elementos o detalles propios del lenguaje que se usará para la programación, lo cual es elegible en una de las opciones del menú. Para lograr esto se concibieron y programaron varias reglas de transformación de los artefactos encontrado en un modelo a otro, incluyéndoles características como por ejemplo los métodos *set* y *get* correspondientes a los atributos que tiene y en función de la plataforma

elegida. En base a esas reglas y a un algoritmo de transformación elaborado se logran generar automáticamente los diagramas en PSM.

El módulo PIM-PSM en su versión 5.0 (Moreno- Orozco, 2018), al iniciarse presenta su ventana principal con un menú principal con las opciones estándares. De acuerdo a la opción seleccionada aparecerán otras opciones correspondientes. Debajo en la parte izquierda está el panel de los Diagramas UML contemplados en esta versión (Casos de Uso, Actividades, Clases, Estados y Secuencias) para el PIM. Cuando se selecciona un diagrama entonces se amplía el panel con los componentes correspondientes. El área de trabajo de la aplicación incluye tres áreas principales que serían el árbol del proyecto que es donde se crean los nuevos diagramas y se pueden abrir otros que se hayan creado, también está la paleta de componentes que son todos los componentes a utilizar para la elaboración de los diagramas, y la otra es el área de trabajo donde se puede trabajar con los diagramas que estén en elaboración.

Revista Cubana de Ciencias Informáticas
Vol. 14, No. 4, Octubre-Diciembre, 2020
ISSN: 2227-1899 | RNPS: 2301
http://rcci.uci.cu
Pág. 210-225

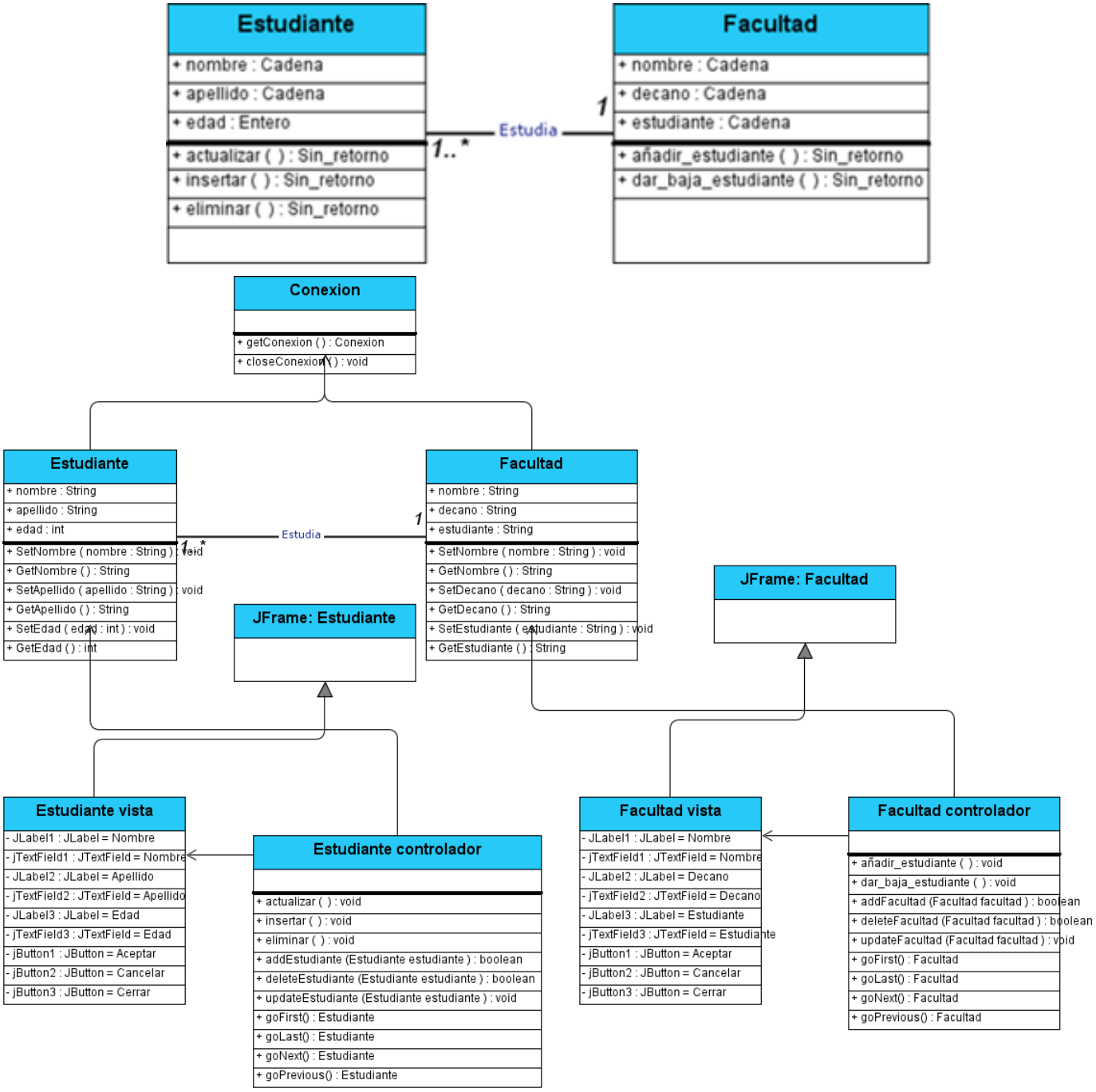

Fig. 4 - Diagrama de clases en PIM y su transformado a PSM.

En la Figuras 4a y 4b se muestra un ejemplo de diagrama de clases reducido de un sistema para controlar la información asociada a los estudiantes, profesores y facultades de una universidad, en modelo PIM y el transformado al PSM usando el modelo vista-controlador.

Módulo PSM-Código

El módulo jMDA PSM-Código versión 4.0 (Moreno-Becerra, 2020) también fue creado a partir de una serie de reglas de transformación definidas, para los lenguajes Java, Phyton y C#. Como quiera que esta herramienta ha sido ideada inicialmente para Sistemas de Información en Bases de Datos, las clases en el modelo vista pueden ser transformadas a un SQL estándar que pueda ser captado por cualquier SGBDR conocido.

En este módulo se implementó también un algoritmo de transformación que incluye 4 platillas de programas (para Java, Python, C# y SQL) y 13 reglas con subreglas, tomando en cuenta que algunas de las reglas dependen del lenguaje seleccionado para generar el prototipo de esqueleto en código fuente correspondiente. El Panel de creación de diagramas se puede observar en la siguiente figura.

Revista Cubana de Ciencias Informáticas
Vol. 14, No. 4, Octubre-Diciembre, 2020
ISSN: 2227-1899 | RNPS: 2301
http://rcci.uci.cu
Pág. 210-225

Fig. 5 - Panel de Diagramas y destaque del Panel de Componentes para Diagrama de Clases.

La opción de exportación o creación de código tiene la siguiente forma:

Revista Cubana de Ciencias Informáticas
Vol. 14, No. 4, Octubre-Diciembre, 2020
ISSN: 2227-1899 | RNPS: 2301
http://rcci.uci.cu
Pág. 210-225

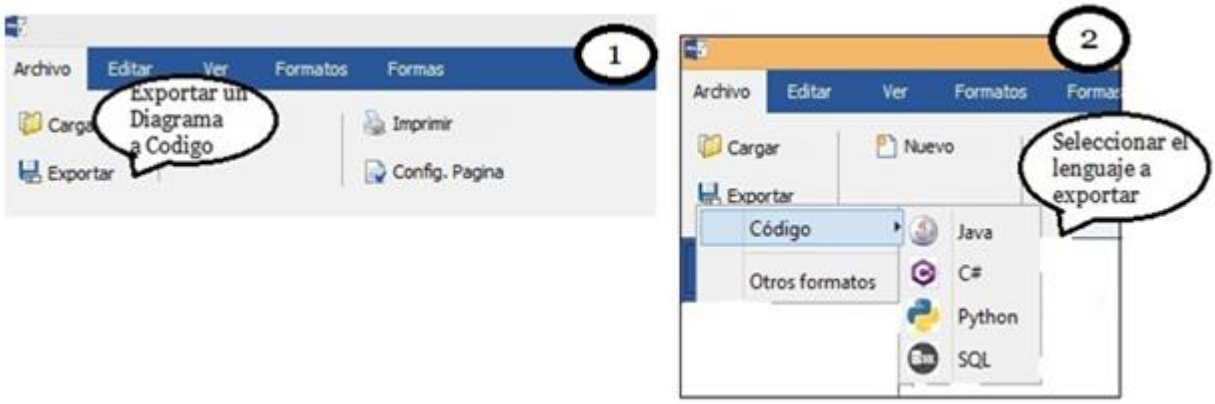

Fig. 6 - Opción de exportación con destaque de los lenguajes implementados.

Para comprobar la aplicabilidad de este módulo PSM-Código en su versión 4 se comparó el diseño de un diagrama de clases sencillo y su transformación en código de los lenguajes previstos utilizando el CASE Visual Paradigm y la herramienta presentada en este trabajo. Se demostró de esta forma que el algoritmo de transformación concebido es suficientemente correcto.

Conclusiones

1. La herramienta está conformada por tres módulos que pueden funcionar independientemente o usarse consecutivamente siguiendo las tres transformaciones correspondientes. Es destacable que esta herramienta incluye un módulo CIM-PIM que no está muy desarrollado en el mundo, sobre todo en herramientas CASE conocidas. Por otra parte, es de factura nacional.
2. El módulo gráfico utilizado en la herramienta permite la edición directa o a través de la importación entre los diferentes módulos, de los diagramas UML esenciales para la generación automática de código fuente, con las funcionalidades adecuadas y normales en este tipo de herramientas, al que se le incorporaron los formatos de salida (JPG, PNG, MXE) logrando exportar los distintos diagramas creados con una excelente calidad. Además, contiene de forma adicional otros formatos de salida propios de la biblioteca utilizada para la confección de los diagramas para su conveniente uso.

3. La herramienta en sus diferentes módulos está registrada en CENDA y puede ser utilizada por casas de software nacional, que pudieran ampliarla mediante la implementación de nuevas reglas de transformación acorde a sus necesidades de desarrollo.
4. La herramienta está concebida de manera modular, por lo que se le pueden seguir incluyendo nuevas posibilidades, como un ambiente CIM para otros tipos de software, u otros lenguajes destinos en su módulo de generación de código, agregándole nuevas plantillas y sus correspondientes reglas de transformación.

Referencias

Booch, G. Developing the Future. Software Solutions. Communications of ACM, 2001. vol. 44 (3), pp. 119-121.

Kleppe A.; Warmer J.; BAST W. MDA Explained: The Model Driven Architecture: Practice and Promise. 2005.Addison-Wesley. New York. 348 pag.

Sommerville, I. Ingeniería de Software. 9a edición. 2011. Pearson Educación de México, S.A. de C.V. México DF. 774 pag.

Pressman, R. Ingeniería de Software. Un enfoque práctico. 8a edición. 2010. McGraw Hill. Interamericana editores, S.A. de C.V. México DF. 778 pag.

Quintero, J. B.; Anaya, R.; Marin, C., Bilbao, A. Un estudio comparativo de herramientas para el modelado con UML. En: Revista Universidad EAFIT. 2005. Vol. 41, (137) pag 60-73

Quintero, J. B.; ANAYA, R. MDA y el papel de los modelos en el proceso de desarrollo de software. Revista EIA, 2007. (8), 131–146.

Durán, M. et al. Desarrollo de software dirigido por modelos. Journal of Chemical Information and Modeling. 2013. 53(9), 1689–1699.

Ferrer, J.; Moreno, R. Herramienta CASE para Arquitectura Dirigida por Modelos jMDA. Módulo CIM-PIM versión 2.0. Tesis de grado. UCLV. Santa Clara. 2018.

Orozco, E.A.; Moreno, R. Desarrollo del módulo PIM-PSM versión 5.0 de la herramienta jMDA. Tesis de grado. UCLV. Santa Clara. 2018.

Becerra, D.E.; Moreno, R. Desarrollo del módulo PSM-CODIGO versión 4.0 de la herramienta JMDA. Tesis de grado. UCLV. Santa Clara. 2020.

Conflicto de interés

El autor autoriza la distribución y uso de su artículo.

Financiación

El presente trabajo no requirió financiación. El mismo forma parte de dos proyectos nacionales de investigación.