HERRAMIENTA PARA LA TRANSFORMACIÓN DE MODELOS ORIENTADOS A AGENTES A MODELOS ORIENTADOS A SÍNTESIS DE REDES DE PROCESOS

OSCAR ANDRÉS FAJARDO F.

UNIVERSIDAD AUTÓNOMA DE BUCARAMANGA
FACULTAD DE INGENIERÍA DE SISTEMAS
GRUPO PRISMA
GESTIÓN DE CONOCIMIENTO
BUCARAMANGA
2011

HERRAMIENTA PARA LA TRANSFORMACIÓN DE MODELOS ORIENTADOS A AGENTES A MODELOS ORIENTADOS A SÍNTESIS DE REDES DE PROCESOS

OSCAR ANDRÉS FAJARDO F.

Trabajo De Grado Presentado Para Optar El Título De
INGENIERO DE SISTEMAS

Director
Juan Carlos García-Ojeda, MSC.

UNIVERSIDAD AUTÓNOMA DE BUCARAMANGA
FACULTAD DE INGENIERÍA DE SISTEMAS
GRUPO PRISMA
GESTIÓN DE CONOCIMIENTO
BUCARAMANGA
2011

NOTA DE ACEPTACIÓN

_____

_____

_____

_____

_____

_____

_____

Firma del Presidente del Jurado

_____

Firma del Jurado

_____

Firma del Jurado

Bucaramanga 4 de Noviembre de 2011

DEDICATORIA

A mis padres.

AGRADECIMIENTOS

Quisiera agradecer a todas aquellas personas que me acompañaron durante este proceso. Todos y cada uno aportó su granito de arena para construir el castillo que para mí es hoy este trabajo de grado.

Muchísimas gracias por su paciencia, apoyo y participación.

Familia Fajardo Fajardo.
Wilson Briceño Pineda, M.Sc.
Juan Carlos García-Ojeda, M.Sc.
Joan Fons Cors, Ph.D.
Javier Gonzales Huerta, M.Sc.

TABLA DE CONTENIDO

pág.

LISTA DE TABLAS

pág.

LISTA DE FIGURAS

pág.

LISTA DE ANEXOS

pág.

GLOSARIO

DSDM: Desarrollo de Software Dirigido por Modelos, ver MDSD.

DSL: Domain Specific Language.

ECORE: Es el meta-metamodelo sobre el que se definen los lenguajes de dominio específico.

EMF: Eclipse Modeling Framework.

GMF: Graphical Modeling Framework.

GMP: Graphical Modeling Project.

LDE: Lenguaje de Dominio Especifico, ver DSL.

M2M: Model To Model.

M2T: Model To Text.

MAS: Multi-Agent System.

MDA: Model Driven Architecture.

MDE: Model Driven Engineering.

MDSD: Model Driven Software Development.

OCL: Object Constraint Language.

OMACS: Organization Model for Adaptative Complex Systems.

O-MaSE: Organization-based Muliagent System Engineering Process Framework.

OMG: Object Management Group.

PNS: Process Network Synthesis.

QVT: Query-View-Transformation.

QVTc: Query-View-Transformation Core.

QVTo: Query-View-Transformation Operational.

QVTr: Query-View-Transformation Relations.

SMA: Sistema Multi-Agente, ver MAS.

SRP: Síntesis de Redes de Procesos, ver PNS.

UML: Unified Modeling Language

.

RESUMEN

Este documento muestra los resultados alcanzados durante el desarrollo de una herramienta que permite realizar transformaciones de modelos orientados a agentes (MAS) a modelos de síntesis de redes de procesos (PNS), cuyo análisis, diseño e implementación se basa en el paradigma de desarrollo de software basado por modelos (DSDM).

El DSDM plantea como producto final los modelos, estos modelos definen DSLs, los cuales permiten representar las particularidades de sistemas, como en el caso de los MAS y los PNS. Los DSLs están definidos en cuatro partes: sintaxis abstracta, sintaxis concreta, semántica estática y semántica dinámica. Para la definición de éstos existe una variedad de herramientas que soportan el DSDM, sin embargo, en su mayoría están en etapas de incubación, investigación y desarrollo, las herramientas de desarrollo de este proyecto están basadas sobre EMF, GMF, OCL y QVTo componentes lo suficientemente maduros para este trabajo.

Finalmente la herramienta producto de este trabajo tiene como objetivo principal transformar modelos MAS en modelos PNS, sirviendo de plataforma tecnológica para una metodología de evaluación temprana de MAS, derivando en la prevención de ciertos comportamientos no deseados de éstos.

Gestión de conocimiento

Palabras Claves: OMACS, PNS, DSDM, M2M, QVT, OCL

INTRODUCCIÓN

Actualmente los MAS están comenzando a tener una amplia participación en el mercado actual, aplicaciones de manufactura, sistemas industriales y de control de procesos, aplicaciones críticas de control de tráfico aéreo, etc.[1] Estas aplicaciones requieren de un meticuloso y efectivo proceso de análisis y diseño que permita a los ingenieros prever los comportamientos y estados de los agentes en un entorno dinámico, con el fin de evitar comportamientos no deseados y emergentes, parte de la naturaleza de los MAS, de tal modo que estos sistemas no culminen en la no obtención de los objetivos para los que fueron diseñados.[2]

Es por esto que es imperativa una metodología de evaluación temprana de estos sistemas, actualmente se han desarrollado un conjunto de métodos[3] ,sin embargo este trabajo de investigación se centra en la obtención de un modelo orientado a PNS a partir de un modelo MAS que posteriormente se utilizara en una de estos nuevos métodos.[4]

Esta herramienta se desarrollará mediante la utilización de técnicas y tecnologías de punta, basados en la metodología de desarrollo dirigido por modelos MDA y herramientas como EMF, GMF, y QVT. De este modo se crearán DSLs que permitan abarcar las particularidades de los MAS y los PNS, permitiendo a su vez

[1] ERRECALDE Marcelo Luis Agentes y Sistemas Multiagente 2009 [En línea]. Universidad Nacional de San Luis [Citado 6 de Agosto de 2010]. Disponible en: www.dirinfo.unsl.edu.ar/~sma/Teorias/teo5ag4.pdf

[2] DELOACH Scott A., Kolesnikov Valeriy A. and Robby Using Design Metrics for Predicting System Flexibility [Article]. Springer-Verlag Berlin Heidelberg. Febrero 04, 2006. p. 184 - 196.

[3] DELOACH Scott A., Oyenan Walamitien and Matson. Eric T. A Capabilities Based Model for Adaptive Organizations [Journal]. Journal of Autonomous Agents and Multiagent Systems. Febrero 2008. Vol. 16. p. 13-56.

[4] GARCIA-OJEDA Juan Carlos [y otros] A Preliminary Study of the Application of the P-Graph Methodology in the Assesment of Organizational-Based Multiagent Systems Desing. - Bucaramanga : [s.n.], 01 de 04 de 2011

transformaciones M2M que culminarán con la obtención del objetivo principal, la creación de un conjunto de plug-ins para Eclipse que permitan la creación de modelos basados en MAS y PNS y transformaciones M2M desde MAS a PNS.

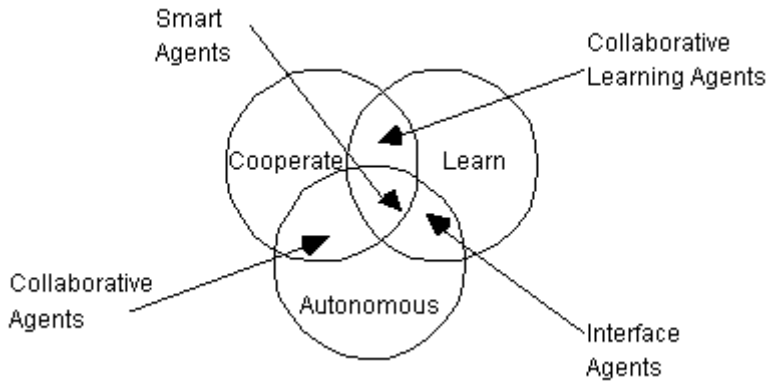
1. AGENTES Y SISTEMAS MULTIAGENTE

Los agentes son entidades humanas o artificiales que sin importar su clasificación comparten características básicas como autonomía, reactividad, iniciativa y sociabilidad. A parte de éstas, los agentes dependiendo del uso que se les vaya a dar, cuentan con otras características que lo complementan, entre ellas se encuentran: movilidad, veracidad, benevolencia, inteligencia (racional, coherente, y adaptable).[5] Cada una de estas características le permite al paradigma de agentes diferenciarse de otros paradigmas,[6] incluso esta definición de agente permite incluir a los seres humanos dentro de este conjunto.

En vista que la definición de agente puede ser bastante genérica. Los agentes pueden ser clasificados por medio de varios criterios. El primer criterio que se tiene en cuenta para clasificar un agente es discriminándolo por su nivel de movilidad, estos agentes pueden ser móviles o estáticos; el segundo por su nivel de interacción con el medio, ya pueden ser deliberativos o reactivos; y el tercero dependiendo de sus habilidades cooperativas, cognitivas y autonómicas, ya sean agentes colaborativos, de interface, de aprendizaje colaborativo o inteligentes.[7]

[5] ROMERO TERNERO Maria del Carmen. Sistemas MultiAgente [En línea]. [Citado 06 de Septiembre de 2010]. Disponible en: http://www.dte.us.es/personal/mcromero/masredes/docs/SMARD.0910.mas.pdf.
[6] ODELL James Agents and Objects [En línea]. James Odell [Citado 08 de Septiembre de 2010] Disponible en: http://www.jamesodell.com/Agents_and_Objects.pdf
[7] NWANA Hyacinth S. Software Agents: An Overview [En línea]. UMBC Agent Web [Citado 09 de Septiembre de 2010]. Disponible en: http://agents.umbc.edu/introduction/ao/.

Figura 1. Clasificación de Agentes



Fuente: (Nwana, 1996)

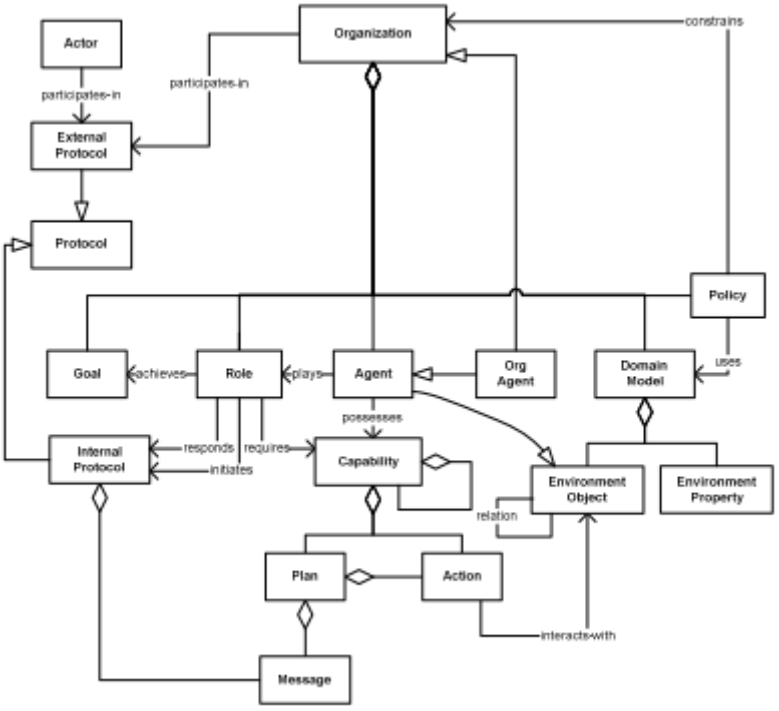
La gran capacidad de los agentes de interpretar su entorno y tomar decisiones concernientes a los objetivos para los que fue diseñado desencadenó en los MAS, que en pocas palabras son un conjunto de agentes interactuando entre ellos para obtener un objetivo. Las características que definen a un sistema MAS son: (1) cada agente tiene información y capacidades limitadas y por ende tiene un punto de vista limitado, (2) no hay un controlador global del sistema, (3) la información es descentralizada y (4) la comunicación es asíncrona.[8]

Los MAS permiten explotar los beneficios del paralelismo, la robustez, y la escalabilidad en dominios impropios para aplicaciones monolíticas. En otras palabras los sistemas MAS son pertinentes a la hora de enfrentar problemas en los que se necesite consolidación de información de diversas fuentes, resolución de conflictos, procesamiento de grandes cantidades de información en tiempo real, etc.[9]

[8] SYCARA Katia P. Multiagent Systems [En línea].American Association for Artificial Intelligence [Citado 08 de Septiembre de 2010]. Disponible en: http://www.aaai.org/AITopics/assets/PDF/AIMag19-02-2-article.pdf
[9] Weiß Gerard Adaptation and Learning in Multiagent Systems> Some Remarks and a Bibliography [En línea].CiteSeerX [Citado 09 de Septiembre de 2010]. Disponible en: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.45.1884&rep=rep1&type=pdf.

Es así como poco a poco los Sistemas Multiagente están empezando a tener mayor protagonismo en sistemas de información de alto rendimiento y precisión[10], los Sistemas Multiagente actualmente cuentan con un contado número de metodologías que permiten su correcto análisis y diseño entre estas se encuentran AAII, MAS-CommonKADS. GAIA, MaSE, MESSAGE e INGENIAS.[11] De estas anteriores existe un descendiente denominado O-MaSE, que presenta un marco de procesos orientados a organizaciones, políticas y dominios para la representación eficiente y posterior desarrollo de MAS.[12]

Figura 2. Metamodelo O-MaSE



Fuente: (Garcia-Ojeda, y otros, 2007)

[10] ERRECALDE Marcelo Luis Agentes y Sistemas Multiagente 2009 [En línea]. Universidad Nacional de San Luis [Citado 6 de Agosto de 2010]. Disponible en: www.dirinfo.unsl.edu.ar/~sma/Teorias/teo5ag4.pdf
[11] ROMERO TERNERO Maria del Carmen Sistemas MultiAgente [En línea]. [Citado 06 de Septiembre de 2010]. Disponible en: http://www.dte.us.es/personal/mcromero/masredes/docs/SMARD.0910.mas.pdf.
[12]GARCIA-OJEDA Juan Carlos [y otros] O-MaSE: A Customizable Approach to Developing Multiagent Development Processes.16 de Marzo de 2007

Sin embargo en los Sistemas Multiagente existe una característica que sobresale entre las demás, los MAS tienden a experimentar comportamientos inesperados que afectan la obtención de los objetivos para los cuales fueron diseñados debido a entornos de aplicación imprevistos durante la fase de diseño. Para sobreponerse a estos sucesos se desarrolló un modelo pensado en la adaptabilidad de los sistemas computacionales, OMACS, permitiendo a los sistemas reorganizarse en tiempo de ejecución para la obtención de los objetivos.[13]

1.1 FRAMEWORK OMACS

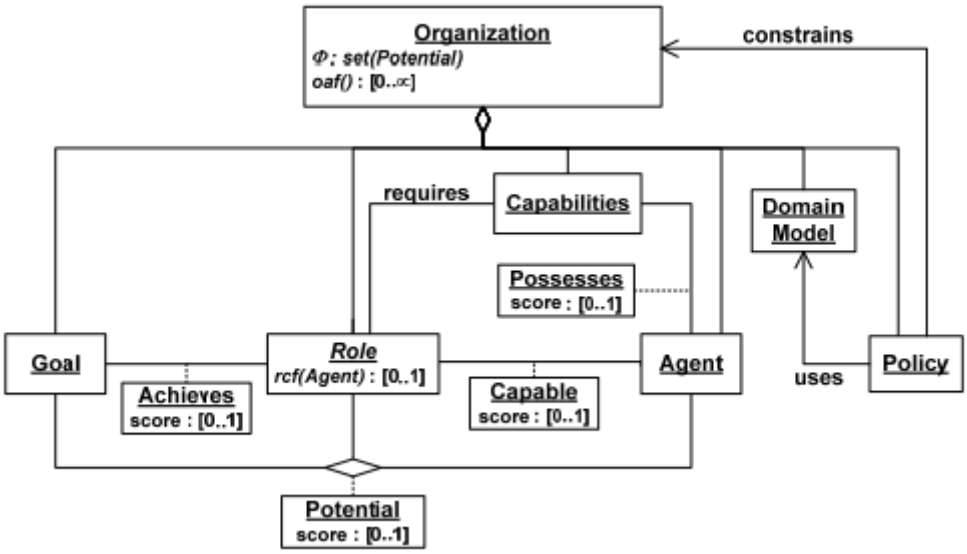
El Framework OMACS surge como una solución para sobrellevar los problemas debidos a la indeterminación de los comportamientos de los agentes en ambientes inciertos o no previstos por el diseñador. Permitiendo desarrollar aplicaciones complejas, sistemas distribuidos con la capacidad de adaptarse a su ambiente cambiante[14]. Un componente clave de OMACS es el metamodelo que define los conocimientos requeridos de la estructura organizacional de un sistema y las capacidades que le permitan reorganizar en tiempo de ejecución y alcanzar sus objetivos eficientemente frente a un entorno y capacidades cambiantes de los agentes. De esta manera se busca explotar todas las capacidades inherentes de los Sistemas Multiagente heterogéneos.[15]

[13]DELOACH Scott A., OYENAN Walamitien and MATSON. Eric T. A Capabilities Based Model for Adaptive Organizations [Journal]. Journal of Autonomous Agents and Multiagent Systems. Febrero 2008. Vol. 16. p. 13-56

[14]MANGHAT Jaidev Simulation of power distribution management system using OMACS metamodel [En línea]. K-State Research Exchange [Citado 09 de Septiembre de 2010]. Disponible en: http://krex.k-state.edu/dspace/bitstream/2097/944/1/JaidevManghat2008.pdf

[15]SERRANO Ana García y OSSOWSKI Sascha Inteligencia Artificial Distribuida y Sistema Multiagente [En línea]. Universidad Politécnica de Madrid [Citado 09 de Septiembre de 2010]. Disponible en: http://www.dia.fi.upm.es/~agarcia/publications/archivos/REV3.pdf

Figura 3. Modelo Organizacional OMACS



Fuente: (DeLoach, et al., 2008)

La organización está compuesta de cuatro entidades: Meta (Goal), Rol (Role), Agente (Agent) y Capacidad (Capabilities). Una Meta define la funcionalidad genérica de la organización y un Rol define una posición en la organización de quien se espera alcance una meta o un conjunto de metas. Un Agente es una entidad humana o artificial, ya bien sea hardware o software, que percibe su ambiente y puede realizar acciones a partir de sus percepciones. Para que un agente puede percibir e incluso actuar, el agente posee Capacidades, que definen las percepciones/acciones los agentes tienen a su alcance, las capacidades pueden ser tangibles (relacionadas al hardware) e intangibles (relacionadas al software) al igual que los tipos de agentes mencionados anteriormente.

Como se puede observar las relaciones definidas por OMACS, Requires (Requiere), Possesses (Posee), Achieves (Alcanza), Capable (Interpreta), permiten definir la manera en la que una organización se distribuye para obtener un objetivo. Las funciones rfc, potencial, oaf, y Φ, describen las capacidades que

un agente necesita para representar un rol, la calidad de capacidades que los agentes poseen, que tan bien un rol alcanza una meta, que tan bien un agente interpreta un rol, que tan bien un agente puede interpretar un rol para alcanzar una meta y las asignaciones a un agente a su respectivo rol y meta, en otras palabras, cómo se comportará el agente y que va a realizar.[16]

Actualmente se pueden ver aplicaciones de OMACS en amplias áreas de aplicación distribuidas en varios laboratorios pertenecientes a la Universidad de Kansas.

- K-State Intelligent Power Distribution Group (IPDS).
- Multiagent and Cooperative Robotics (MACR) Laboratory.
- Expeditionary Capabilities Consortium (ECC).
- Pervasive Sensor Network Laboratory (PerSNL).
- Center for Information and Systems Assurance (CISA).
- Specification, Analysis, and Transformation of Software (SAnToS) Laboratory.

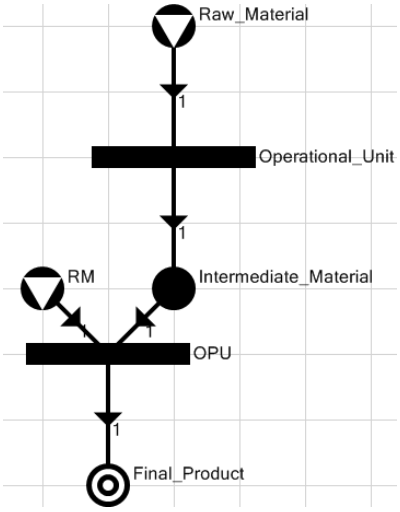
Y patrocinadas por entidades como la Oficina de la Fuerza Aérea de Investigación Científica (AFOSR), la Fundación Nacional de Ciencias (NSF), el Comando de Sistemas de los Marines, M2 Technologies, y Stanfield Systems, Inc.

[16] GARCÍA-OJEDA Juan Carlos OMACS Overview, 02 de Septiembre de 2010.

2. SÍNTESIS DE REDES DE PROCESOS

Las Síntesis de Redes de Procesos es una técnica que permite la optimización basada en un modelo mixto tanto heurístico (grafos) como matemático (combinatorias), en el que se elige la red de procesos más óptima para obtener un producto determinado. [17]

Figura 4. Ejemplo de una PNS



Fuente: Autor del proyecto

Representando a la red de procesos por medio de una grafo bipartita, denominada P-Graph, simboliza la estructura de un proceso que captura la sintaxis y semántica contenida en el sistema de procesos. Entre los elementos que conforman una gráfica de procesos se pueden encontrar: Materiales (Materials), Unidades

[17] FRIEDLER F. [y otros] Graph-Theoretic Approach to Process Synthesis Polynomial Algorithm for Maximal Structure Generation [En línea]. Department of Computer Science and Systems Technology, University of Pannonia, Hungary [Citado 23 de 09 de 2010]. Disponible en: http://dcs.vein.hu/cikkek/Grap-Theor_Appr_to_Proc_Synth_Polyn_Alg_for_Max_Struc_Gen.pdf.

Operativas (Operating Units) y arcos. Los materiales se subdividen en: Materia Prima (Raw Material), Materiales Intermedios (Intermediate Materials), o Productos Terminados (Products). Las unidades operativas son definidas en función del material de entrada/salida. En la Figura, se pueden observar diferentes tipos de elementos materia prima, materia intermedia, producto final, unidades operacionales y relaciones entre éstos.[18]

Figura 5. Materia Prima

Fuente: Autor del proyecto

Figura 6. Material Intermedio

Fuente: Autor del proyecto

Figura 7. Producto Terminado

Fuente: Autor del proyecto

[18] VARGA Virag [y otros] PNS Solutions: a P-Graph Based Programming Framework for Process Network Synthesis [En línea]. Department of Computer Science and Systems Technology, University of Pannonia, Hungary [Citado 23 de Septiembre de 2010]. Disponible en: http://www.dcs.vein.hu/cikkek/PNS_solutions_a_P-graph-based.pdf.

Figura 8. Unidad Operativa

Fuente: Autor del proyecto

Figura 9. Arco o relación

Fuente: Autor del proyecto

Una estructura de la solución de un problema de combinatorias PNS se considera viable si se satisfacen 5 axiomas. (1) Todas las productos deben figurar en la estructura, (2) un material es un recurso si y solo si no es producto de cualquier unidad operacional, (3) cada unidad operacional en la estructura está definida en la síntesis del problema, (4) cualquier unidad operacional tiene al menos un arco que lo una a un producto y, (5) si un material pertenece a la estructura debe está a la entrada o salida de una unidad operacional.[19]

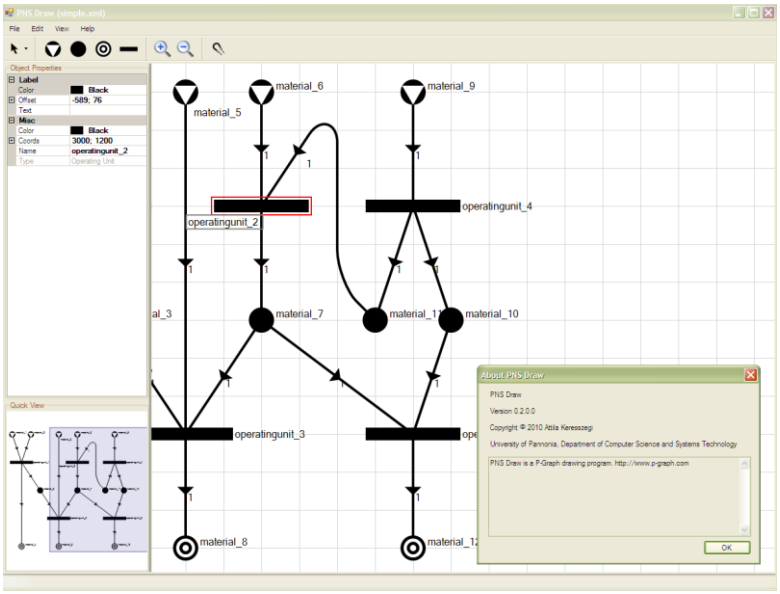
2.1 HERRAMIENTAS DE DISEÑO PNSDRAW Y ANÁLISIS PNSSTUDIO

Las herramientas PNS-Draw y PNS-Studio son dos herramientas que permiten la creación y análisis de síntesis de redes de procesos. Cada una de ellas tiene un propósito determinado, PNS-Draw como el nombre lo dice es un editor gráfico que

[19]FRIEDLER F. [y otros] Graph theoretic approach to process synthesis axioms and theorems [En línea]. Department of Computer Science and Systems Technology, University of Pannonia, Hungary [Citado 23 de Septiembre de 2010]. Disponible en: http://dcs.vein.hu/cikkek/Grap-Theor_Appr_to_Proc_Synth_Ax_and_Theor.pdf

permite dibujar materiales, unidades operacionales y conexiones, a su vez que se pueden definir textos personalizados y tasas de flujo.

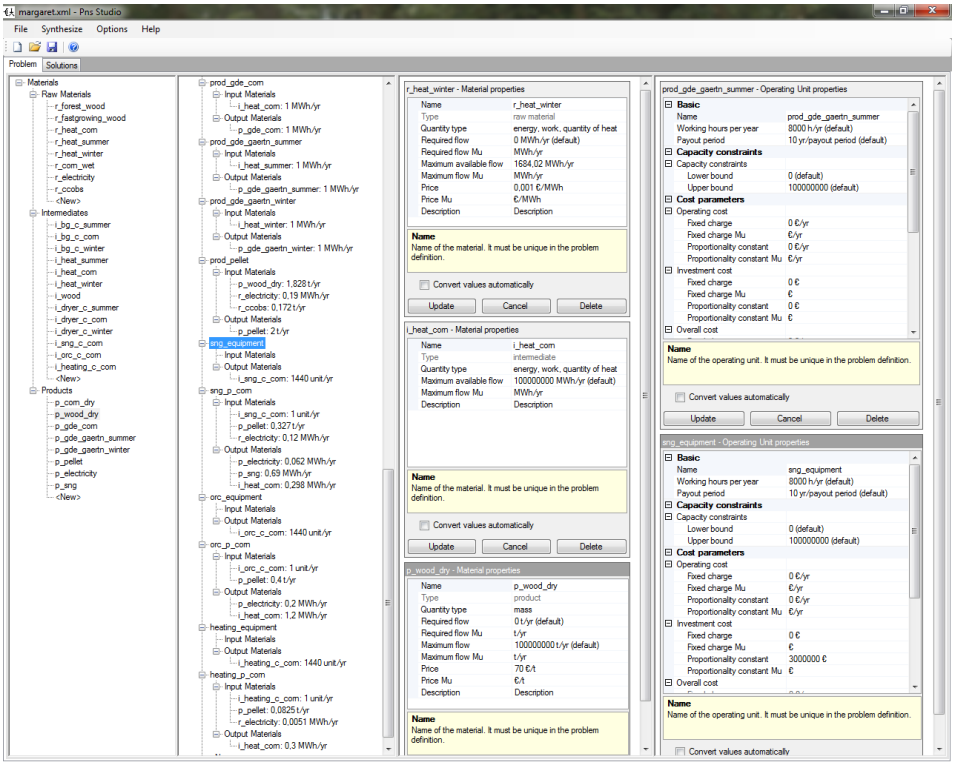
Figura 10. Ambiente PNS Draw



Fuente: (Keresszegi)

PNS-Studio es un aplicativo que se encarga de analizar y resolver situaciones representadas por medio de síntesis de redes de procesos creadas por medio de la herramienta PNS-Draw. PNS-Studio puede facilitar la definición del problema por medio de vista en árboles, edición en paralelo, asignación de materiales de entrada y salida a unidades operacionales por medio de arrastrar-y-soltar. Además cuenta con solucionadores embebidos que permite encontrar soluciones de alta complejidad, y con un intérprete que permite la representación gráfica de dichas soluciones, y por último permite la exportación de reportes a Excel.

Figura 11. Ambiente PNS Studio



Fuente: (Bertok)

3. MODEL DRIVEN ENGINEERING

La ingeniería dirigida por modelos (MDE por sus siglas en inglés) es un área del conocimiento relativamente reciente, sin embargo los problemas a los que se enfrenta son los mismos a los que se ha enfrentado la industria del software desde sus inicios.[20] Los más grandes fracasos han sido siempre marcados por la dificultad de expresión de los requerimientos de un cliente, formulados en terminología de un dominio en específico, en algoritmos para lenguajes de programación.[21] Otro de los problemas que se quiere atacar es la problemática inherente de los sistemas heredados, en los que muchas veces una actualización de un pequeño detalle puede desembocar en inconvenientes técnicos que resultan en el fracaso de la actualización del sistema, y por último la capacidad de crear sistemas que permitan interoperabilidad entre plataformas, que permitan una evolución permanente del software protegiendo la lógica del sistema y convirtiéndola en un elemento independiente de la tecnología.[22].

El paradigma propuesto por MDE permite visualizar cualquier elemento que se pueda abstraer como un modelo, una vez establecido esto, cualquier producto del proceso de desarrollo de software es un modelo. Convirtiendo en los modelos no solo en los productos iniciales del desarrollo, los cuales poca actualización

[20]WOLTERINK Tjerk The Future of Software Engineering: Model Driven Engineering [En línea].Tjerk's Tech Blog [Citado 21 de Septiembre de 2010]. Disponible en: http://tjerktech.wordpress.com/2010/04/19/the-future-of-software-engineering-model-driven-engineering/.

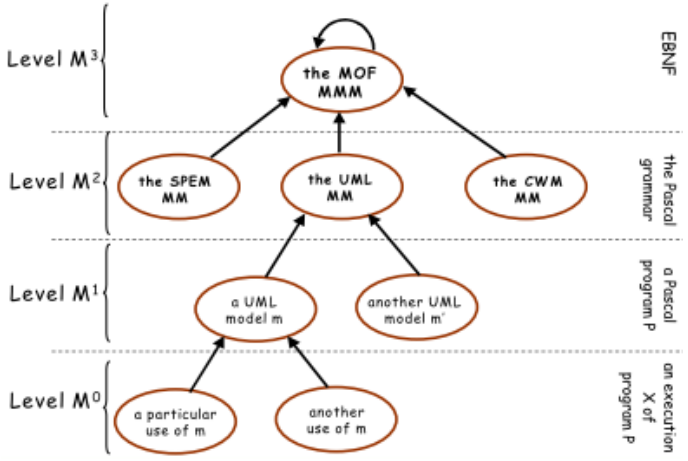
[21]NEIMAT Taimour Al Why Projects Fail [En línea].Project Perfect [Citado 21 de Septiembre de 2010]. Disponible en: http://www.projectperfect.com.au/downloads/Info/info_it_projects_fail.pdf.

[22]ZOUFALY Federico Issues and Challenges Facing Legacy Systems [En línea]. [Citado 21 de Octubre de 2010]. Disponible en: http://www.developer.com/mgmt/article.php/1492531/Issues-and-Challenges-Facing-Legacy-Systems.htm.

reciben, sino en el principal elemento de trabajo durante todo el desarrollo del sistema.[23]

Para poder explotar las capacidades de los modelos se debe mencionar el papel de los Metamodelos, los Metamodelos son las estructuras que definen las relaciones entre conceptos en un dominio determinado y también la semántica y las restricciones propias del dominio.[24] Actualmente existen cuatro niveles en la jerarquía de modelos, siendo el M3, ver Figura, el más complejo ya que éste define a todos los demás Metamodelos (Metamodelo UML, Metamodelo BPMN, Metamodelo SPEM, entre otros) y a él mismo. Entre más alto es el nivel de jerarquía, más abstracto y complejo es el trabajo de definición del dominio. Sin embargo este sistema de jerarquías ha permitido alcanzar niveles de interoperabilidad que anteriormente no se pensaba.[25]

Figura 12. Niveles de Jerarquías de Modelos



Fuente (Vallecillo, 2008)

[23]WOLTERINK Tjerk The Future of Software Engineering: Model Driven Engineering [En línea].Tjerk's Tech Blog [Citado 21 de Septiembre de 2010]. Disponible en: http://tjerktech.wordpress.com/2010/04/19/the-future-of-software-engineering-model-driven-engineering/.

[24] SCHMIDT Douglas C. Model-Driven Engineering [En línea]. Washington University in St. Louis [Citado 21 de Octubre de 2010]. Disponible en: http://www.cs.wustl.edu/~schmidt/GEI.pdf.

[25] INSFRÁN Emilio Tema 3 IntroDSL v8. - Valencia : [s.n.], 09 de Noviembre de 2009.

Las aplicaciones que tienen modelos y Metamodelos actualmente permiten resaltar una serie de ventajas:

- La representación directa de la solución en términos del dominio del problema.

- Comprensibilidad, un modelo es mucho más comprensible que código.

- Documentación, un modelo representa cómo debe ser construido un sistema.

- Validación, Un modelo puede ser validado dependiendo de las reglas o criterios de diseño que se le quieran aplicar al sistema.

- Simulación, un modelo puede ser utilizado para simular diferentes situaciones y hallar la mejor solución antes de construir el sistema.

- Trazabilidad, un modelo puede referencia a otros modelos e incluso código, por lo tanto la información o los cambios en un modelo se pueden ver reflejados en otros modelos.

- Automatización, la implementación del código puede ser generada a partir de los modelos, de tal modo que incremente la velocidad, reduce la cantidad de errores y facilita la corrección de estos.[26]

[26] INSFRÁN Emilio Tema 3 IntroDSL v8. - Valencia : [s.n.], 09 de Noviembre de 2009.

3.1 MODEL DRIVEN ARCHITECTURE

Model Driven Architecture (MDA) es la iniciativa de Object Management Group (OMG) que más difusión ha tenido siguiendo la propuesta metodológica de Model Driven Engineering (MDE). Esta propuesta utiliza estándares tales como MOF, UML, OCL, XMI y QVT, de los cuales MOF y UML permite la definición de nuevas familias de lenguajes.[27].

Anunciada en el año 2000 como campo de investigación, planificó un plazo de maduración de 10 años, y una vez alcanzado este tiempo pronostica un uso de esta tecnología por aproximadamente 20 años más.

La característica más importante y sobresaliente de MDA es la independencia con la plataforma, una solución que no requiera ser diseñada en función del lenguaje, la arquitectura, etc., es una solución que permite una fácil y rápida implementación y actualización. Los tipos de modelos que se pueden manejar en un DSDM (Desarrollo de Software Dirigido por Modelos) por lo general son PIM (Platform Independent Model) y PSM (Platform Specific Model).[28]

Por lo general cuando se habla de MDA, damos por entendido que es generación de código a partir de un modelo realizado en una herramienta. Sin embargo MDA va mucho más allá de este enunciado, ésta permite el desarrollo de DSLs.[29] Un DSL permite entre otras cosas la adaptabilidad de un sistema a problemas endémicos de un modelo de negocio particular. La construcción de un DSL está conformada por cuatro partes esenciales:

---

[27] PELENCHANO Vicente Tema 1 TCP -2008 UPV. Valencia: [s.n.],17 de Octubre de 2007
[28] INSFRÁN Emilio Tema 3 IntroDSL v8. - Valencia : [s.n.], 09 de Noviembre de 2009.
[29] INSFRÁN Emilio Tema 3 IntroDSL v8. - Valencia : [s.n.], 09 de Noviembre de 2009.

- Sintaxis abstracta: conceptos del lenguaje y sus relaciones (diagramas de clases del Metamodelo).

- Sintaxis concreta: visualización de los conceptos de la sintaxis abstracta.

- Semántica estática: restricciones del sistema, en ocasiones escrito en lenguaje OCL.

- Semántica dinámica: transformaciones de Modelo a Modelo (M2M) o generación de código.

Para la definición de la sintaxis abstracta se puede emplear EMF de Eclipse, la definición de un Metamodelo en EMF puede representarse en tres formas distintas, XML schemas, diagramas de clase UML y Java anotado, EMF permite la migración entre cualquiera de estas representaciones de manera transparente, potencializando la labor del diseñador y permitiendo un sinnúmero de posibilidades.

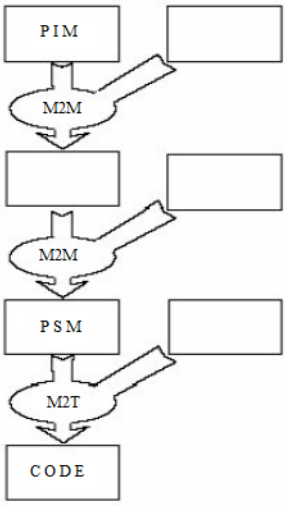
La sintaxis concreta se puede definir por medio de GMF de Eclipse, una herramienta bastante potente que permite al diseñador crear un conjunto de archivos que desencadenarán en la generación de un editor gráfico específico para el SDL desarrollado previamente, con integración de restricción.

Para la implementación de la semántica estática se emplean restricciones escritas en OCL tanto a nivel de Metamodelo (EMF) como en la definición de la sintaxis concreta (GMF). OCL hace parte de la iniciativa de la OMG siendo la especificación de éste un documento relacionado en la especificación tanto de MDA como de UML. La plataforma de Eclipse en su entorno de desarrollo orientado a modelos permite la integración de restricciones OCL a nivel de Metamodelo, permitiendo al diseñador del sistema incrustar restricciones que no

se pueden representar por medio de la definición de diagramas de clases, schemas XML, o java anotado (las tres formas existentes de definir un Metamodelo en EMF). El componente que permite estas integraciones tan útiles se denomina OCLInEcore Editor, lo que hace es presentar el archivo .Ecore en otra perspectiva que permite la definición de restricciones OCL de manera natural e intuitiva.

La semántica dinámica puede comprenderse como la capacidad de un modelo de transformarse, ya sea en otros modelos o en código. Las transformaciones entre modelos se referencias M2M y las transformaciones de generación de código como M2T.[30]

Figura 13. Transformaciones de modelos



Fuente: (Pelenchano, 2007)

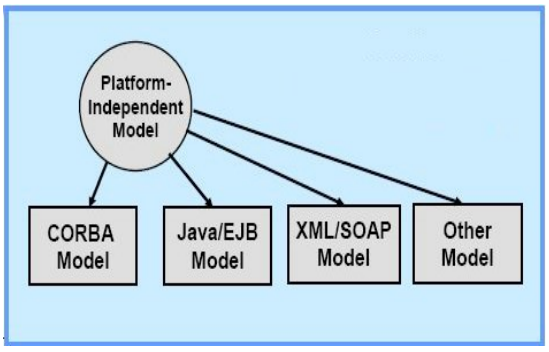
[30] STEINBERG Dave [y otros] EMF: Eclipse Modeling Framework [Libro].[s.l.]: Addison-Wesley Professional, 2008

Para generar una aplicación utilizando MDA se debe comenzar definiendo la lógica del negocio y la funcionalidad de un sistema, a esto se le denomina PIM. Un Modelo Independiente de la Plataforma permite la definición de las condiciones (previas y posteriores) y su comportamiento sin tener que incluir ningún tipo de detalle con respecto a la plataforma en la que se va a desarrollar. Una vez ya se define el PIM es indispensable definir la plataforma en la que se desarrollara la aplicación, es así que por medio de una transformación de modelo a modelo, ya sea utilizando QVT (Query View Transformation) o cualquier otra herramienta de transformación de modelos se obtiene el PSM.

QVT, definido por la OMG como lenguaje de transformación de modelos, que está compuesto por tres partes QVTcore, QVTrelations y QVTOperative. De estas tres partes, el QVTc y QVTr son basados en una estructura declarativa, el QVTo está basado en una estructura imperativa. QVTc y QVTr hacen parte del paquete QVTdeclarative (QVTd) de Eclipse, aún en desarrollo e incubación, QVTo es parte de un proyecto aparte totalmente dedicado a implementar en su totalidad las capacidades de este componente del lenguaje, aunque aún está en desarrollo se encuentra en una etapa más avanzada que los componentes mencionados anteriormente.

Todos los componentes de QVT están a su vez potencializados por las capacidades del lenguaje OCL permitiendo al desarrollador/modelador explorar los Metamodelos a medida que va creando las transformaciones, y validar que los objetos que se están manipulando en las transformaciones cumplan con las especificaciones de los Metamodelos origen/destino.

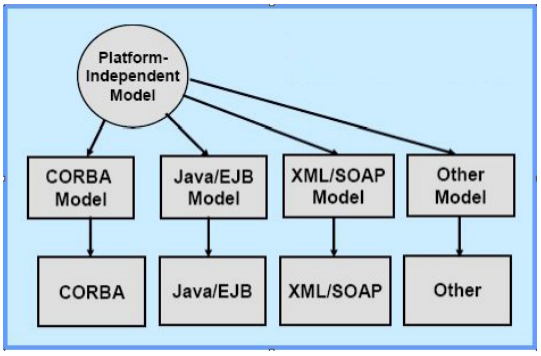
Figura 14. Transformación PIM – PSM



Fuente: (Pelenchano, 2007)

También existe la posibilidad de utilizar una o más transformaciones dependiendo del nivel de detalle que se requiera o incluso en las plataformas que se quiera. Una vez se obtiene el PSM es posible agregar detalles de implementación y por medio de una transformación de modelo a código se obtiene el producto terminado o casi terminado.

Figura 15. Transformación PSM – Código



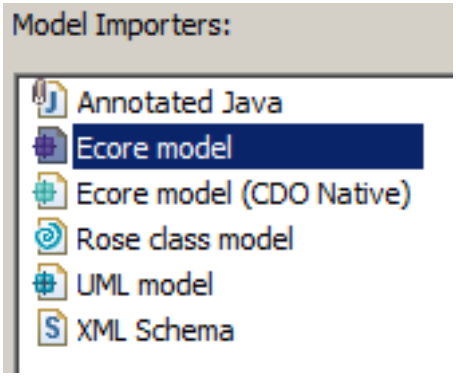
Fuente: (Pelenchano, 2007)

4. ECLIPSE MODELING PROJECT

4.1 ECLIPSE MODELING FRAMEWORK

Eclipse Modeling Framework (EMF) es un framework de modelado y generador de código para construir herramientas y otras aplicaciones basadas en un modelo de datos estructurado. A partir de una especificación del modelo descrito en XMI, EMF provee herramientas y soporte en tiempo de ejecución para obtener un conjunto de clases Java, que permiten la edición del modelo por medio de visualizaciones o por medio de comandos, y un editor gráfico básico. Lo más importante de todo, EMF suministra las bases para la interoperabilidad con otras herramientas y aplicaciones basadas en EMF, los Modelos pueden ser especificados usando Java Anotado, documentos XML, o herramientas de modelado como Rational Rose, y después ser importados a EMF.[31]

Figura 16. Importación diferentes especificaciones de Metamodelos.



Fuente: Autor del proyecto

[31] SKRYPUCH Neil Eclipse Modeling Framework Project (EMF) [En línea]. Eclipse.org [Citado 16 de Septiembre de 2010]. Disponible en: http://www.eclipse.org/modeling/emf/.

EMF define el modelo del dominio. Sin embargo EMF realmente está basado en dos Metamodelos, el Metamodelo Ecore y el Metamodelo Genmodel.

- El Metamodelo Ecore contiene información acerca de las clases definidas.

- El Metamodelo Genmodel contiene información adicional y parámetros para la generación de código.

Los modelos de Ecore permiten definir diferentes elementos, entre ellos los principales son:

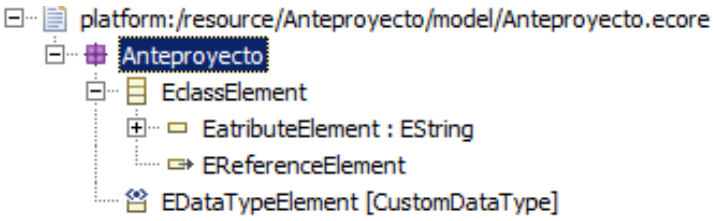
- EClass, representa una clase con cero o más atributos y con cero o más referencias;

- EAtribute, representa un atributo que tiene un nombre y un tipo de dato;

- EReference, describe una asociación entre dos clases. Tiene un indicador que representa contención y una referencia a la clase que lo contiene;

- EDataType, representa el tipo de dato de un atributo.

Figura 17. Elementos Ecore



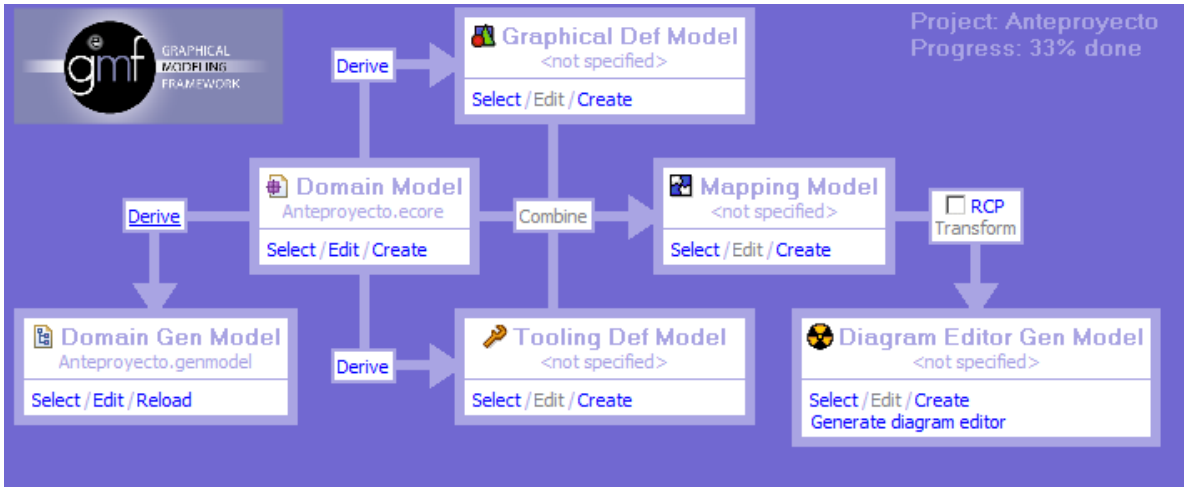
Fuente: Autor del proyecto

4.2 GRAPHICAL MODELING FRAMEWORK

Eclipse GMF (Graphical Modeling Framework) es un framework DSL utilizado para crear editores gráficos basados en EMF (Eclipse Modeling Framework) y GEF (Graphical Editing Framework). Este framework cuenta con una serie de validaciones de modelos, transformaciones de modelo a modelo y modelo a texto que permiten la creación de un entorno grafico totalmente funcional a partir de un Metamodelo en Ecore. Como se puede observar en la Figura, GMF cuenta con un flujo de trabajo en el que se muestra el nivel de avance del proyecto de creación de editores gráficos. Una vez se cumplan todos los requisitos de cada nodo se obtiene un editor gráfico que posteriormente puede ser publicado como un plug-in completamente funcional.[32]

[32] RICHLEY Jeff GMF: Beyond the Wizards [En línea]. O'Reilly Media, Inc [Citado 22 de Octubre de 2010]. Diponible en: http://onjava.com/pub/a/onjava/2007/07/11/gmf-beyond-the-wizards.html

Figura 18. Flujo de Trabajo GMF



Fuente: (Gronback, 2009)

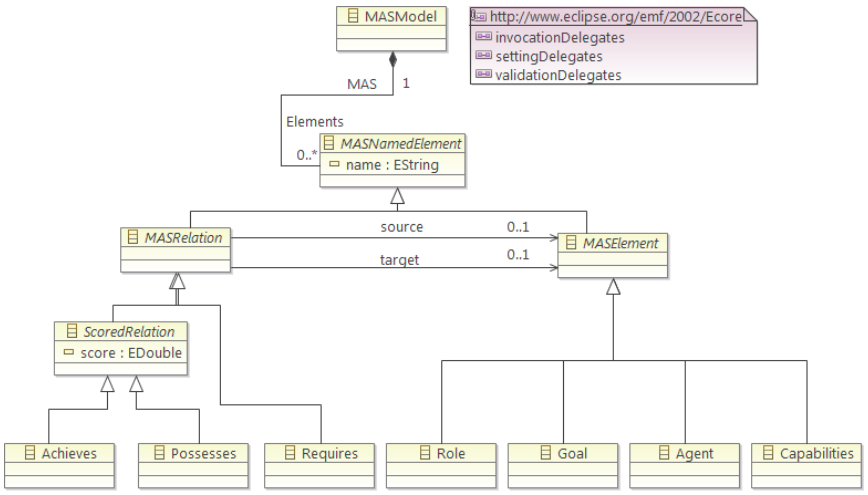
5. DESARROLLO DE METAMODELOS

Para la creación de los Metamodelos origen/destino es necesario tener instalados los plug-ins EMF que son los que permiten la creación de éstos. La creación del Metamodelo MAS se basó en un Metamodelo existente, se requirió la modificación de éste debido a que EMF no permite relaciones terciarias. La creación del Metamodelo PNS se realizó desde cero, éste fue basado en un análisis de la herramienta PNS Graph y PNS Studio.

5.1 DESARROLLO METAMODELO MAS

El Metamodelo origen está basado en el Metamodelo OMACS sin embargo para este trabajo sólo se tendrán en cuenta los elementos Role, Capability, Agent y Goal. Y las relaciones Achieves, Requires y Possesses. Después de un análisis previo se determina el siguiente Metamodelo base.

Figura 19. Metamodelo MAS



Fuente: Autor del proyecto

En este Metamodelo como se puede observar se crea una clase MASModel que contendrá MASNamedElement. Esta clase MASModel será la clase padre o en otras palabras será el archivo que contenga los elementos mencionados anteriormente. La clase MASNamedElement Es una clase abstracta que definirá el nombre de todos los elementos del modelo, facilitando así una posterior implementación de una restricción que evite la ocurrencia de nombres repetidos. Esta clase a su vez es padre de las clases abstractas MASRelation y MASElement. Como se puede observar los MASElement se relacionan con MASRelation por medio de las relaciones source y target es así como un MASElement se relacionan consigo mismo a través de las MASRelation en el modelo. Los MASElement a su vez son especializados en: Role, Goal, Agent y Capability

La MASRelation se especializa en dos clases; una abstracta, denominada ScoredRelation que permitirá la creación de clases especializadas con un atributo Score, y una clase Requires que identifica cuándo un Role requiere de una Capability en específico. La clase ScoredRelation se especializa en las clases

Achieves y Possesses. La clase Achieves permite identificar cuándo un Role alcanza un Goal en específico, y la clase Possesses describe cuándo un Agent posee cierta Capability

Este Metamodelo es funcional, sin embargo es demasiado genérico y permite crear relaciones indebidas según el Metamodelo OMACS, es por esto que se deben integrar restricciones externas al Metamodelo, la solución más práctica es incrustar restricciones OCL en el Metamodelo de tal forma que las restricciones no hagan parte de un archivo aparte sino por el contrario hagan parte del mismo archivo .Ecore. La herramienta utilizada para incrustar las restricciones OCL en el Metamodelo se denomina OCLinEcore Editor. Esta herramienta permite abrir el archivo .Ecore en una perspectiva de solo texto en la cual se pueden escribir las restricciones OCL tal cual como el estándar OCL por OMG lo indica, haciendo de este proceso una actividad transparente para el desarrollador.

Figura 20. Ejemplo Restricciones OCL

```
class Requires extends MASRelation
{
    invariant RequiresDestinationType: self.target->forAll(a|a.oclIsKindOf(Capabilities));
    invariant RequiresSourceType: self.source->forAll(a|a.oclIsKindOf(Role));
}
class Achieves extends ScoredRelation
{
    invariant AchievesDestinationType: self.target->forAll(a|a.oclIsKindOf(Goal));
    invariant AchievesSourceType: self.source->forAll(a|a.oclIsKindOf(Role));
}
class Possesses extends ScoredRelation
{
    invariant PossessesDestinationType: self.target->forAll(a|a.oclIsKindOf(Capabilities));
    invariant PossessesSourceType: self.source->forAll(a|a.oclIsKindOf(Agent));
}
```

Fuente: Autor del proyecto

Como se puede observar en la imagen, las restricciones son incrustadas como solo texto, sin embargo al generar código a partir del Metamodelo y al crear

instancias dinámicas de éste, permite la validación de las restricciones y por ende la validez de los elementos y las relaciones en el modelo. Las restricciones definidas para este Metamodelo constan de:

- Restricciones para la relación Requires (sólo las Capability pueden ser target, y solo los Role pueden ser Source).

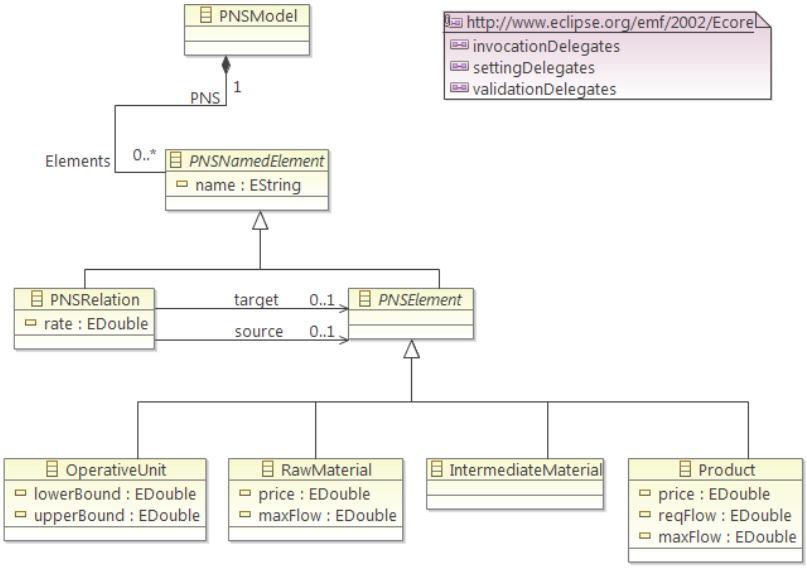
- Para la relación Achieves (sólo los Goal pueden ser target y solo los Role pueden ser Source).

- Por último para la relación Possesses (sólo los Capability pueden ser target y solo los Agent pueden ser Source).

5.2 DESARROLLO METAMODELO PNS

El Metamodelo destino no está basado en ningún Metamodelo definido previamente, sino que es abstraído a partir de la sintaxis concreta y semántica estática presente en el editor PNSGraph. El Metamodelo base se muestra a continuación.

Figura 21. Metamodelo base PNS



Fuente: Autor del proyecto

En este Metamodelo como se puede observar es bastante similar al Metamodelo origen, con el fin de facilitar la transformación. En este Metamodelo se crea una clase PNSModel que contendrá PNSNamedElement. Esta clase PNSModel será el elemento que contenga todos los demás elementos del modelo. La clase PNSNamedElement es una clase abstracta que definirá el nombre de todos los elementos del modelo, facilitando así una posterior implementación de una restricción que evite la ocurrencia de nombres repetidos.

Al igual que en el Metamodelo anterior existen las clases PNSRelation y PNSElement, se puede ver en la Figura que PNSRelation no se especializa en ninguna otra clase, siendo esta la principal diferencia entre los dos Metamodelos. Finalmente la clase abstracta PNSElement se especializa en OperativeUnit, RawMaterial, IntermediateMaterial y Product.

Similar al Metamodelo MAS, este Metamodelo también requiere de restricciones. Al ser un Metamodelo más sencillo a nivel de PNSRelation, puesto que ésta no se especializa, las restricciones se aplican solo a un elemento del Metamodelo.

Figura 22. Restricciones OCL en Metamodelo PNS

```
class PNSRelation extends PNSNamedElement
{
    invariant DestinationType:
    if self.source.oclIsKindOf(RawMaterial)
        or self.source.oclIsKindOf(IntermediateMaterial)then
            self.target-> forAll(a|a.oclIsKindOf(OperativeUnit))
    else
        if self.source.oclIsKindOf(Product)then
            self.target->forAll(target.oclIsUndefined())
        else
            self.target -> forAll(a|a.oclIsKindOf(IntermediateMaterial))
            or self.target -> forAll(a|a.oclIsKindOf(Product))
        endif
    endif;
```

Fuente: Autor del proyecto

Al igual que en el caso anterior las restricciones definidas para este Metamodelo están incrustadas y constan de:

- Restricciones para la relación PNSRelation(Un RawMaterial o un IntermediateMaterial solo se pueden relacionar con un OperativeUnit, Un Product no se puede relacionar con ningún elemento del modelo y un OperativeUnit solo se puede relacionar con IntermediateMaterial y Product)

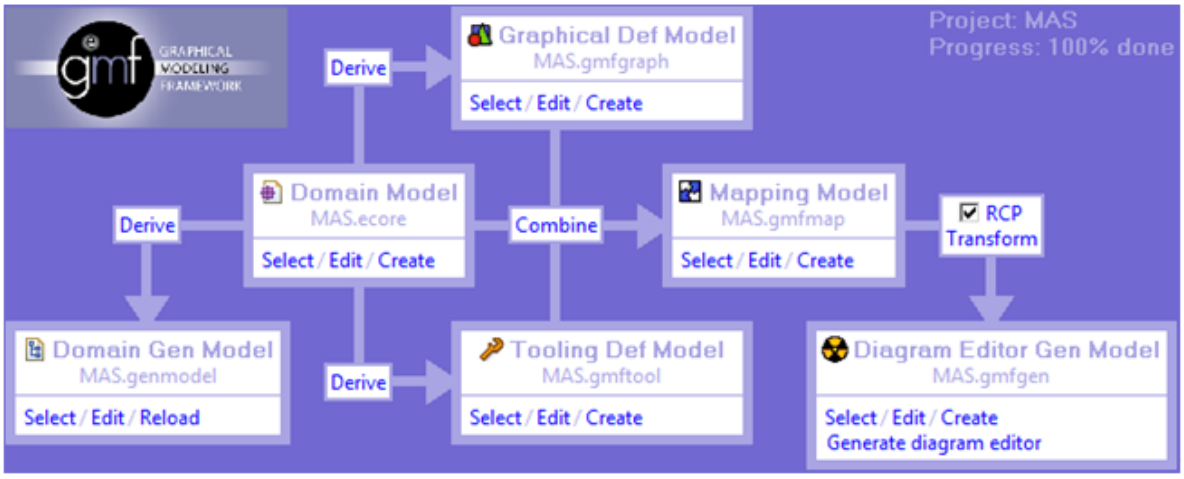
Es así como concluye la definición de los Metamodelos origen/destino. Posterior a esto se debe crear el archivo .Genmodel, a partir del archivo .Ecore que se acaba de crear, que permitirá la generación de código que permitirá la creación de

instancias dinámicas del Metamodelo y que servirá como insumo principal para el desarrollo de editores gráficos.

6. DESARROLLO DE EDITORES GRÁFICOS

La creación de editores grafico para las instancias dinámicas de los Metamodelos (modelos) se basa en la siguiente figura. Como se puede observar es necesario haber definido previamente el archivo .Ecore y generado el archivo .Genmodel. Los pasos subsiguientes se explicaran en esta sección.

Figura 23. Guía de Creación Editores Gráficos



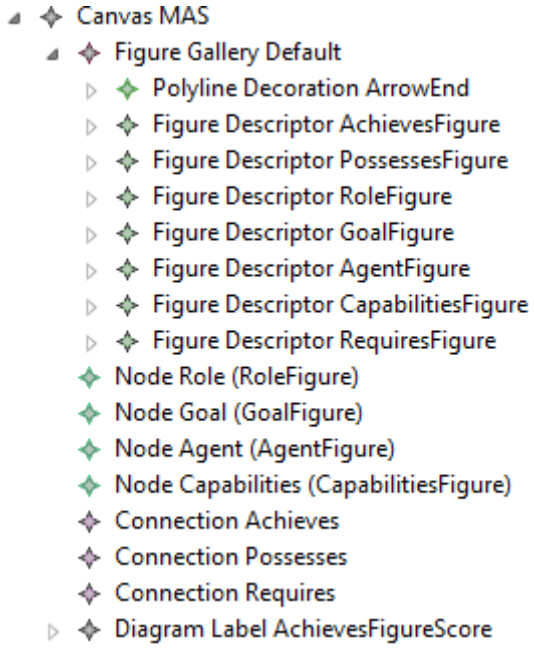
Fuente: Autor del proyecto

Como se puede observar se requiere de tres archivos antes de obtener el archivo generador de editores gráficos (.GMFGen), estos tres archivos son Modelo de definición grafica (.GMFGraph), modelo de definición de herramientas graficas (.GMFTool) y modelo de mapeo de modelos (.GMFMap).

6.1 CREACIÓN GMFGRAPH.

La creación del modelo de definición grafica permite al desarrollador escoger la sintaxis concreta del Metamodelo, relacionando los elementos y conexiones de cada Metamodelo con una representación gráfica o figura. A su vez se puede definir si la figura contiene labels, la propiedad que describen y la manera en cómo se accede.

Figura 24. Archivo GMFGraph

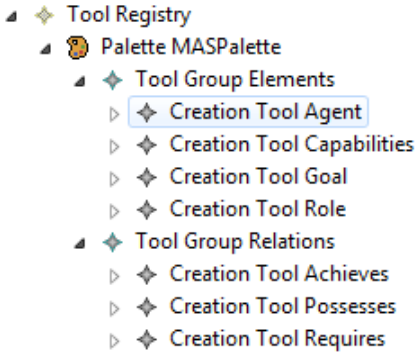


Fuente: Autor del proyecto.

6.2 CREACIÓN GMFTOOL.

La creación del modelo de definición de herramientas graficas permite al desarrollador definir las herramientas que permiten la representación gráfica de los elementos y relaciones en el editor gráfico. A su vez también se puede definir la imagen con la que se representará dicho elemento, la forma en que se agrupan y presentan dichas herramientas, selección de la herramienta por defecto en cada grupo de herramientas y definición de opciones de minimización, maximización de los grupos de herramientas.

Figura 25. Archivo GMFGraph



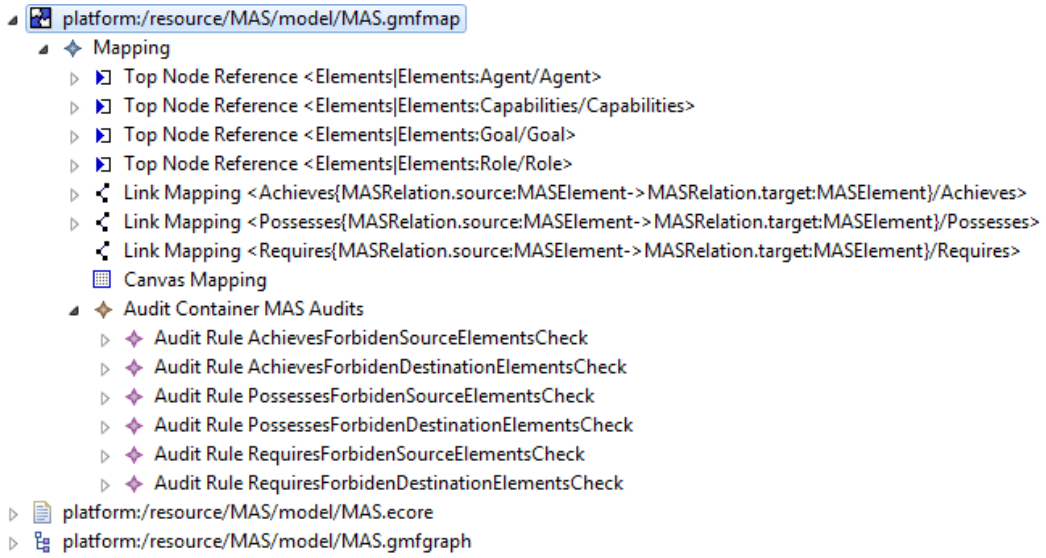
Fuente: Autor del proyecto.

6.3 CREACIÓN GMFMAP.

La creación del modelo de mapeo de modelos permite al desarrollador la creación de relaciones entre los modelos .Genmodel, GMFGraph y GMFTool. De este modo se asigna una representación gráfica a los elementos del Metamodelo y una

herramienta de dibujo. También se pueden crear auditorías al editor gráfico para que valide en tiempo real las restricciones incrustadas en el Metamodelo.

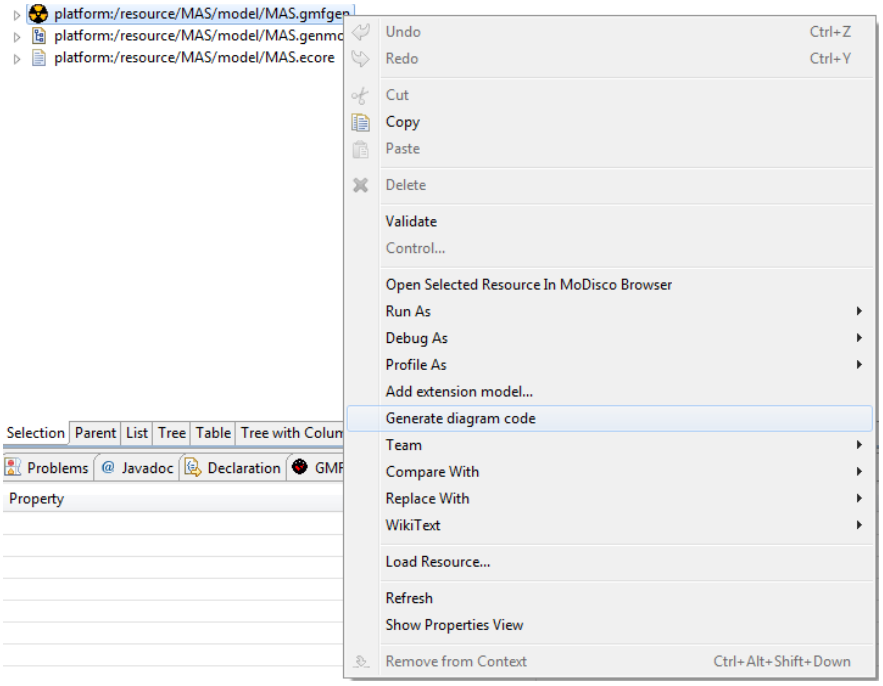
Figura 26. Archivo GMFGraph



Fuente: Autor del proyecto.

Finalmente una vez todos los modelos son relacionados en el GMFMap se procede a crear el archivo GMFGen que permitirá la creación del editor gráfico. El GMFGen, permite al desarrollador escoger opciones que se verán reflejadas e el editor gráfico, entre estas opciones se encuentra la posibilidad de elegir entre la creación de dos archivos separados un para la definición textual del modelo y otro para la definición gráfica del modelo, o crear un solo archivo para describir las dos perspectivas, también puede elegir las extensiones de archivo de cada uno de los archivos mencionados anteriormente, la compatibilidad del código generado en java y los directorios en donde se almacenara el código generado entre otras cosas.

Figura 27. Creación del editor gráfico.



Fuente: Autor del proyecto.

7. CREACIÓN SCRIPT DE TRANSFORMACIÓN M2M EN QVTO

Para la creación del script de transformación se requiere tener los Metamodelos terminados de tal modo que se puedan publicar e instalar en el entorno de desarrollo Eclipse o puedan ser referenciados en el encabezado del archivo.

El lenguaje de transformación QVTo permite al desarrollador crear reglas de transformación de una manera similar a como se crean métodos en las clases y realizar invocaciones de estos métodos como se realiza en el paradigma orientado a objetos.

Las reglas de transformación para obtener un modelo PNS a partir de un modelo MAS no son relaciones uno a uno, es decir, no existe una correspondencia entre cada uno de los elementos del Metamodelo origen al Metamodelo destino.

El algoritmo que define la transformación fue tomado del trabajo de investigación (Garcia-Ojeda, y otros, 2011). En éste se define cómo se deben asociar los elementos y relaciones del modelo MAS hacia el modelo PNS.

Figura 28. Algoritmo de transformación MAS2PNS

input: set M, pair OMD
comment: M = ∅, tuples Elements = $OMD_{elements}$ and, Relationships = $OMD_{relationships}$
output: set of relationships O
update: set of material M
begin

$st1$: $\forall g \in Elements_{goals}$
$M = M \cup \{\langle g \cdot oaf \rangle\}$
$O = O \cup \{\{\{\{\langle g, 1, ? \rangle\}, \{\langle g \cdot oaf, 1, 1 \rangle\}\}\}, \infty\}$

$st2$: $\forall a \in Elements_{agents}$
$C_{agent} = \varnothing$
$Pos_{agent} = \varnothing$

$ln4$: $\forall \langle a', c, v' \rangle \in Relationships_{pos}$
$if\ a = a'$
$C_{agent} = C_{agent} \cup \{c\}$
$Pos_{agent} = Pos_{agent} \cup \langle a', c, v' \rangle$

$ln8$: $\forall \langle r_1, \wp(c) \rangle \in Relationships_{req}$
$if\ \wp(c) \subseteq C_{agent}$
$t = \varnothing$

$ln12$: $\forall \langle r_2, g, v'' \rangle \in Relationships_{ach}$
$if\ r_1 = r_2$

$ln14$: $M = M \cup \{\langle a \cdot r_1 \cdot g \rangle\}$

$ln15$: $z = z \cup \left\{ \langle a \cdot r_1 \cdot g, \frac{\sum_{c' \in \wp(c)} compPos(Pos_{agent}, c')}{|\wp(c)|}, ? \rangle \right\}$

$ln16$: $t = t \cup \{\langle\{\{\{\langle a \cdot r_1 \cdot g, 1, ? \rangle\}, \{\langle g, v'' \rangle\}\}\}, \infty\rangle\}$

$ln17$: $O = O \cup \{\langle\{\{\{\langle a, 1, 0 \rangle\}, \{z\}\}\}, 1\rangle\} \cup t$

end

Fuente: (Garcia-Ojeda, y otros, 2011)

Cómo se puede observar en la Figura se está manejando teoría de conjuntos para la transformación de modelo a modelo, para este caso en específico es una situación favorable debido a que el lenguaje OCL permite el manejo de conjuntos en la realización de consultas, similares a las que se realizan en lenguajes de bases de datos como SQL. Posterior a un análisis detallado se definen dos etapas de transformación. En la primera etapa se logra abarcar la primera sentencia del algoritmo, la segunda etapa abarca la segunda sentencia.

Figura 29. Primera Sentencia del Algoritmo de Transformación

$$st1:\ \forall g \in Elements_{goals}$$
$$M = M \cup \{\langle g \cdot oaf \rangle\}$$
$$O = O \cup \{\langle\{\langle\{\langle g, 1, ? \rangle\}, \{\langle g \cdot oaf, 1, 1 \rangle\}\rangle\}, \infty \rangle\}$$

Fuente: (Garcia-Ojeda, y otros, 2011)

Para la primera sentencia se realizó la transformación en dos lenguajes diferentes QVTr y QVTo, finalmente se descartó la primera opción debido a que la segunda presenta mejores capacidades en cuestión de trazabilidad de la transformación y despliegue automático de un plug-in que permita la implantación de la transformación en el Eclipse sin necesidad de adicionar código ni parámetros sensibles a errores y siguiendo la metodología MDA.

Figura 30. Ejemplo de transformación en lenguaje QVTo

```
abstract mapping MAS::Goal::Goal2PNSElements(p: PNS::PNSModel)
{
    var im : IntermediateMaterial;
    var ou : OperativeUnit;
    var pd: Product;

    im := self.map Goal2IntermediateMaterial(p);
    ou := self.map Goal2OperationalUnit(p);
    pd := self.map Goal2Product(p);
    map newRelation1(p,im,ou);
    map newRelation2(p,ou,pd);

    allAgentsAtOnce();
}
```

Fuente: Autor del proyecto.

Figura 31. Segunda Sentencia del Algoritmo de Transformación

$$
\begin{aligned}
&st2{:}\ \forall a \in Elements_{agents} \\
&\quad C_{agent} = \varnothing \\
&\quad Pos_{agent} = \varnothing \\
&ln4{:}\ \forall \langle a', c, v' \rangle \in Relationships_{pos} \\
&\qquad if\ a = a' \\
&\qquad\quad C_{agent} = C_{agent} \cup \{c\} \\
&\qquad\quad Pos_{agent} = Pos_{agent} \cup \langle a', c, v' \rangle \\
&ln8{:}\ \forall \langle r_1, \wp(c) \rangle \in Relationships_{req} \\
&\qquad if\ \wp(c) \subseteq C_{agent} \\
&\qquad\quad t = \varnothing \\
&ln12{:}\quad \forall \langle r_2, g, v'' \rangle \in Relationships_{ach} \\
&\qquad\quad if\ r_1 = r_2 \\
&ln14{:}\quad\quad M = M \cup \{\langle a \cdot r_1 \cdot g \rangle\} \\
&ln15{:}\quad\quad z = z \cup \left\{ \langle a \cdot r_1 \cdot g, \frac{\sum_{c' \in \wp(c)} compPos(Pos_{agent}, c')}{|\wp(c)|}, ? \rangle \right\} \\
&ln16{:}\quad\quad t = t \cup \{\langle\{\{\{\langle a \cdot r_1 \cdot g, 1, ? \rangle\}, \{\langle g, v'' \rangle\}\}\}, \infty \rangle\} \\
&ln17{:}\quad O = O \cup \{\langle\{\{\{\langle a, 1, 0 \rangle\}, \{z\}\}\}, 1 \rangle\} \cup t
\end{aligned}
$$

Fuente: (Garcia-Ojeda, y otros, 2011)

En la segunda sentencia de la transformación, se implementaron consultas y objetos temporales para la transformación, puesto que posee una complejidad más alta y requiere de una secuencia de agrupaciones y cálculos. Se realiza una analogía lo más fiel posible al pseudocódigo que presenta el algoritmo previamente presentado obteniendo los resultados esperados.

8. CONCLUSIONES

La creación de los metamodelos que definen los modelos MAS y PNS se realizaron con la herramienta EMF se les incrustaron restricciones escritas en OCL por medio del OCLinEcore Editor que se basa en los plug-ins de texto extendido Xtext. Estos metamodelos se crearon lo más similares posibles de tal modo que permitieran una asociación entre ellos de una manera similar y sus transformaciones fueran lo más transparentes posibles.

Debido a lo novedosas que son estas tecnologías presentan varios inconvenientes, entre estos se puede mencionar la implementación incompleta de algunas características fundamentales para este trabajo; ej. La implementación de restricciones OCL en clases abstractas; la falta de documentación con respecto a las características de los lenguajes tanto de definición como de transformación de modelos; también se referenciaron algunas características inesperadas (bugs) en los editores de definición[33].

Para la creación de los editores gráficos es prerrequisito tener los metamodelos terminados, el código de implementación del modelo, y de los editores textuales generados, ya que el editor gráfico se basa en éstos. Fue necesaria la implementación de las restricciones OCL en el editor gráfico de tal manera que expresara en tiempo real las advertencias y validaciones incrustadas en el Metamodelo.

[33]WILLINK Edward Bugzilla [En línea]. [Citado 06 de Febrero de 2011]. Disponible en: https://bugs.eclipse.org/bugs/show_bug.cgi?id=329389.

La transformación de M2M se desarrolló inicialmente en el leguaje QVTr que permitía una visualización más transparente de las relaciones entre modelos, sin embargo fue descartado y reemplazado por el lenguaje QVTo que permite una trazabilidad de la transformación mucho más confiable que la ofrecida por QVTr además de la generación y liberación de un plug-in al finalizar el proceso de desarrollo de la transformación.

La liberación de plug-ins es una labor sencilla para los proyectos creados en EMF, GMF y QVTo ya que desde el inicio del desarrollo éste es creado y al finalizar simplemente se realiza un despliegue de los plug-ins que contienen toda la implementación realizada durante el proyecto.

9. TRABAJOS FUTUROS

Como se mencionó anteriormente existen características que no se pudieron utilizar durante este desarrollo, trabajos futuros relacionados podrían incluir la implementación de más y mejores restricciones a nivel de Metamodelo y editores gráficos.

A su vez se resalta la importancia de ajustarse lo más ceñidamente posible a la metodología MDA de tal modo que las modificaciones necesarias puedan aplicarse desde los modelos y no directamente en el código como se realizó en algunos casos especiales; ej. La definición de los “node plates” para geometrías no definidas en EMF.[34]

La creación de ayudas (Cheat Sheets) y update sites para los plug-ins generados durante este trabajo, serían de gran ayuda para los usuarios de estos productos. Puesto que como todo desarrollo es vulnerable a cambios y mejoras constantes y su uso a pesar de ser intuitivo, no siempre la intuición es el mejor manual de usuario.

[34] BRAZEAU Jean-François GMF Samples and Tutorials [En línea]. [Citado 10 de Diciembre de 2010]. Disponible en: http://gmfsamples.tuxfamily.org/wiki/doku.php?id=gmf_tutorial4#comment_22669eace5741a67599aa2b3e2df920b

BIBLIOGRAFIA

ANACLETO Valerio Adrián Epidata Consulting [En línea]. Epidata Consulting [Citado 2 de Septiembre de 2010]. Disponible en: http://www.epidataconsulting.com/site/files/60-64%20code%2031%20whitepaper.pdf.

BARENDRECHT P.J. Modeling transformations using QVT Operational Mappings [En línea]. Eindhoven University of Technology [Citado 23 de Febrero de 2011]. Disponible en: http://redpanda.nl/BEP_P.J.Barendrecht.pdf.

BERTOK Botond P-graph - PNS Studio [En línea]. P-graph [Citado 16 de 09 de 2010]. Disponible en: http://www.p-graph.com/pnsstudio/index.html.

BOLLATI Verónica A. [y otros] Análisis de QVT Operational Mappings: un caso de estudio [En línea] [Citado 11 de Febrero de 2011]. Disponible en: http://www.sistedes.es/TJISBD/Vol-3/No-2/articles/DSDM-09-bollati-aqom.pdf.

BOYKO Sergey, DVORAK Radomil y IGDALOV Alexander The Art of Model Transformation with Operational QVT [En línea].Borland Software Corporation. [Citado 23 de Febrero de 2011]. Disponible en: www.eclipse.org/m2m/qvto/doc/EclipseCon_2009.ppt.

BRAZEAU Jean-François GMF Samples and Tutorials [En línea]. [Citado 10 de Diciembre de 2010]. Disponible en: http://gmfsamples.tuxfamily.org/wiki/doku.php?id=gmf_tutorial4#comment_22669eace5741a67599aa2b3e2df920b.

CORREDERA DE COLSA Luis Enrique Corredera.net [En línea]. Corredera.net [Citado 02 de Septiembre de 2010]. Disponible en: http://www.corredera.net/mda_j2me.pdf.

DELOACH Scott A., KOLESNIKOV Valeriy A. and Robby Using Design Metrics for Predicting System Flexibility [Article]. Springer-Verlag Berlin Heidelberg. Febrero 04, 2006. p. 184 - 196.

DELOACH Scott A., OYENAN Walamitien and Matson. Eric T. A Capabilities Based Model for Adaptive Organizations [Journal]. Journal of Autonomous Agents and Multiagent Systems. Febrero 2008. Vol. 16. p. 13-56.

DVORAK Radomil Model Transformation with Operational QVT [En línea]. Borland Software Corporation.[Citado 12 de Febrero de 2011]. Disponible en: http://www.eclipse.org/m2m/qvto/doc/M2M-QVTO.pdf.

ERRECALDE Marcelo Luis Agentes y Sistemas Multiagente 2009 [En línea]. Universidad Nacional de San Luis [Citado 6 de Agosto de 2010]. Disponible en: www.dirinfo.unsl.edu.ar/~sma/Teorias/teo5ag4.pdf.

FRIEDLER F. [y otros] Graph-Theoretic Approach to Process Synthesis Polynomial Algorithm for Maximal Structure Generation [En línea]. Department of Computer Science and Systems Technology, University of Pannonia, Hungary [Citado 23 de 09 de 2010]. Disponible en: http://dcs.vein.hu/cikkek/Grap-Theor_Appr_to_Proc_Synth_Polyn_Alg_for_Max_Struc_Gen.pdf.

FRIEDLER F. [y otros] Graph theoretic approach to process synthesis axioms and theorems [En línea]. Department of Computer Science and Systems Technology, University of Pannonia, Hungary [Citado 23 de Septiembre de 2010]. Disponible en: http://dcs.vein.hu/cikkek/Grap-Theor_Appr_to_Proc_Synth_Ax_and_Theor.pdf.

FRIEDLER F. [y otros] Graph-Theoretic Approach to Process Synthesis Polynomial Algorithm for Maximal Structure Generation [En línea].Department of Computer Science and Systems Technology, University of Pannonia, Hungary [Citado - 23 de Septiembre de 2010]. Disponible en: http://dcs.vein.hu/cikkek/Grap-Theor_Appr_to_Proc_Synth_Polyn_Alg_for_Max_Struc_Gen.pdf.

GARAVITO Oscar L., OCAMPO Juan D. y TORRES Miguel E. Metodología de Pruebas para Sistemas MultiAgentes (SMA) integrada a AOPOA [En línea].Universiad Nacional de Colombia Sede Medellín. [Citado 07 de Septiembre de 2010].Diponible en: http://pisis.unalmed.edu.co/3CCC/pdf/52.pdf.

GARCIA-OJEDA Juan Carlos [y otros] A Preliminary Study of the Application of the P-Graph Methodology in the Assesment of Organizational-Based Multiagent Systems Desing. - Bucaramanga : [s.n.], 01 de 04 de 2011.

GARCIA-OJEDA Juan Carlos [y otros] O-MaSE: A Customizable Approach to Developing Multiagent Development Processes.16 de Marzo de 2007.

GARCÍA-OJEDA Juan Carlos OMACS Overview, 02 de Septiembre de 2010.

GRONBACK Richard C. Eclipse modeling project : a domain-specific language (DSL) toolkit [Libro]. Massachusetts. Pearson Education, Inc., 2009.

INSFRÁN Emilio Tema 3 IntroDSL v8. - Valencia : [s.n.], 09 de Noviembre de 2009.

KAMINKA Gal A. Robots are Agents, Too! [En línea]. Agentlink [Citado 22 de Septiembre de 2010]. Disponible en: http://u.cs.biu.ac.il/~galk/Publications/Papers/agentlink04.pdf.

KERESSZEGI Attila P-graph. P-graph drawer (PNS Draw) [En línea]. P-graph. [Citado 16 de 09 de 2010]. Disponible en: http://www.p-graph.com/pnsdraw/screenshot.png.

KÖNEMANN Patrick A QVT model transformation for creating Model-Independent Diffs [En línea]. Technical University of Denmark [Citado 12 de Febrero de 2011]. Disponible en: http://modeldiff.imm.dtu.dk/modeldiff/images/docs/emfdiff2indepdiff.pdf.

MANGHAT Jaidev Simulation of power distribution management system using OMACS metamodel [En línea]. K-State Research Exchange [Citado 09 de Septiembre de 2010]. Disponible en: http://krex.k-state.edu/dspace/bitstream/2097/944/1/JaidevManghat2008.pdf.

MELLOULI Sehl FATMAS: A Methodology to Design Fault-tolerant Multi-agent Systems [En línea]. Networked Digital Library of Theses and Dissertation [Citado 14 de Septiembre de 2010]. Disponible en: http://www.theses.ulaval.ca/2005/22674/22674.pdf.

NEIMAT Taimour Al Why Projects Fail [En línea].Project Perfect [Citado 21 de Septiembre de 2010]. Disponible en: http://www.projectperfect.com.au/downloads/Info/info_it_projects_fail.pdf.

NWANA Hyacinth S. Software Agents: An Overview [En línea]. UMBC Agent Web [Citado 09 de Septiembre de 2010]. Disponible en: http://agents.umbc.edu/introduction/ao/.

ODELL James Agents and Objects [En línea]. James Odell [Citado 08 de Septiembre de 2010] Disponible en: http://www.jamesodell.com/Agents_and_Objects.pdf.

OMG Meta Object Facility (MOF) 2.0 Query/View/Transformation Specification [En línea]. OMG[Citado 22 de Febrero de 2011]. Disponible en: http://www.omg.org/spec/QVT/1.1/PDF/.

OMG Meta Object Facility (MOF) Core Specification [En línea]. OMG[Citado 22 de Febrero de 2011]. Disponible en: http://www.omg.org/spec/MOF/2.0/PDF/.

OMG Object Constraint Language [En línea]. OMG [Citado 22 de Febrero de 2011]. Disponible en: http://www.omg.org/spec/OCL/2.2/PDF.

PELENCHANO Vicente Tema 1 TCP -2008 UPV. Valencia: [s.n.],17 de Octubre de 2007.

PELENCHANO Vicente Tema 1 TCP -2008 UPV. Valencia: [s.n.], 17 de 20 de 2007.

QUINTERO Paulo C. Comparación de metodologías y arquitecturas de sistemas multiagente encontradas en 10 aplicaciones utilizadas en medicina y servicios de e-salud [En línea]. Scribd.com [Citado 07 de Noviembre de 2010]. Disponible en: http://www.scribd.com/doc/23897006/Comparacion-de-Metodologias-y-Arquitecturas-de-Sistemas-Multiagente-as-en-10-Aplicaciones-Utilizadas-en-Medicina-y-Servicios-de-E-salud.

RICHLEY Jeff GMF: Beyond the Wizards [En línea]. O'Reilly Media, Inc [Citado 22 de Octubre de 2010]. Diponible en: http://onjava.com/pub/a/onjava/2007/07/11/gmf-beyond-the-wizards.html.

ROMERO TERNERO Maria del Carmen Sistemas MultiAgente [En línea]. [Citado 06 de Septiembre de 2010]. Disponible en: http://www.dte.us.es/personal/mcromero/masredes/docs/SMARD.0910.mas.pdf.

SANZ Jorge J. Gómez Metodologías para el desarrollo de sistemas multi-agente [En línea].Universidad Nacional de Educación a Distancia [Citado 08 de Septiembre de 2010]. Disponible en: http://cabrillo.lsi.uned.es:8080/aepia/Uploads/18/38.pdf.

SCHMIDT Douglas C. Model-Driven Engineering [En línea]. Washington University in St. Louis [Citado 21 de Octubre de 2010]. Disponible en: http://www.cs.wustl.edu/~schmidt/GEI.pdf.

SERRANO Ana García y OSSOWSKI Sascha Inteligencia Artificial Distribuida y Sistema Multiagente [En línea]. Universidad Politécnica de Madrid [Citado 09 de Septiembre de 2010]. Disponible en: http://www.dia.fi.upm.es/~agarcia/publications/archivos/REV3.pdf.

SKRYPUCH Neil Eclipse Modeling Framework Project (EMF) [En línea]. Eclipse.org [Citado 16 de Septiembre de 2010]. Disponible en: http://www.eclipse.org/modeling/emf/.

STEINBERG Dave [y otros] EMF: Eclipse Modeling Framework [Libro].[s.l.]: Addison-Wesley Professional, 2008.

SYCARA Katia P. Multiagent Systems [En línea].American Association for Artificial Intelligence [Citado 08 de Septiembre de 2010]. Disponible en: http://www.aaai.org/AITopics/assets/PDF/AIMag19-02-2-article.pdf.

TOLVANEN Juha-Pekka DSL in Practice [En línea]. Institut für Softwaretechnik und Theoretische Informatik [Citado 16 de Septiembre de 2010]. Disponible en: http://tfs.cs.tu-berlin.de/gtvmt08/Program/DSMinPractice_Tolvanen_29March2008.pdf.

VALLECILLO Antonio Model Driven Development [En línea]. Departamento Lenguajes y Ciencias de la Computación Universidad de Málaga [Citado - 22 de Septiembre de 2010]. Disponible en: http://www.lcc.uma.es/~canal/sabc/MDA-doctorado.pdf.

VARGA Virag [y otros] PNS Solutions: a P-Graph Based Programming Framework for Process Network Synthesis [En línea]. Department of Computer Science and Systems Technology, University of Pannonia, Hungary [Citado 23 de Septiembre de 2010]. Disponible en: http://www.dcs.vein.hu/cikkek/PNS_solutions_a_P-graph-based.pdf.

WEIß Gerard Adaptation and Learning in Multiagent Systems> Some Remarks and a Bibliography [En línea].CiteSeerX [Citado 09 de Septiembre de 2010]. Disponible en: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.45.1884&rep=rep1&type=pdf.

WILLINK Edward Bugzilla [En línea]. [Citado 06 de Febrero de 2011]. Disponible en: https://bugs.eclipse.org/bugs/show_bug.cgi?id=329389.

WOLTERINK Tjerk The Future of Software Engineering: Model Driven Engineering [En línea].Tjerk's Tech Blog [Citado 21 de Septiembre de 2010]. Disponible en: http://tjerktech.wordpress.com/2010/04/19/the-future-of-software-engineering-model-driven-engineering/.

ZOUFALY Federico Issues and Challenges Facing Legacy Systems [En línea]. [Citado 21 de Octubre de 2010]. Disponible en: http://www.developer.com/mgmt/article.php/1492531/Issues-and-Challenges-Facing-Legacy-Systems.htm.

BIBLIOGRAFÍA COMPLEMENTARIA

ANACLETO Valerio Adrián Epidata Consulting [En línea]. Epidata Consulting [Citado 2 de Septiembre de 2010]. Disponible en: http://www.epidataconsulting.com/site/files/60-64%20code%2031%20whitepaper.pdf.

BARENDRECHT P.J. Modeling transformations using QVT Operational Mappings [En línea]. Eindhoven University of Technology [Citado 23 de Febrero de 2011]. Disponible en: http://redpanda.nl/BEP_P.J.Barendrecht.pdf.

BERTOK Botond P-graph - PNS Studio [En línea]. P-graph [Citado 16 de 09 de 2010]. Disponible en: http://www.p-graph.com/pnsstudio/index.html.

BOLLATI Verónica A. [y otros] Análisis de QVT Operational Mappings: un caso de estudio [En línea] [Citado 11 de Febrero de 2011]. Disponible en: http://www.sistedes.es/TJISBD/Vol-3/No-2/articles/DSDM-09-bollati-aqom.pdf.

BOYKO Sergey, DVORAK Radomil y IGDALOV Alexander The Art of Model Transformation with Operational QVT [En línea].Borland Software Corporation. [Citado 23 de Febrero de 2011]. Disponible en: www.eclipse.org/m2m/qvto/doc/EclipseCon_2009.ppt.

BRAZEAU Jean-François GMF Samples and Tutorials [En línea]. [Citado 10 de Diciembre de 2010]. Disponible en: http://gmfsamples.tuxfamily.org/wiki/doku.php?id=gmf_tutorial4#comment_22669eace5741a67599aa2b3e2df920b.

CORREDERA de Colsa Luis Enrique Corredera.net [En línea]. Corredera.net [Citado 02 de Septiembre de 2010]. Disponible en: http://www.corredera.net/mda_j2me.pdf.

DELOACH Scott A., KOLESNIKOV Valeriy A. and Robby Using Design Metrics for Predicting System Flexibility [Article]. Springer-Verlag Berlin Heidelberg. Febrero 04, 2006. p. 184 - 196.

DELOACH Scott A., OYENAN Walamitien and MATSON. Eric T. A Capabilities Based Model for Adaptive Organizations [Journal]. Journal of Autonomous Agents and Multiagent Systems. Febrero 2008. Vol. 16. p. 13-56.

DVORAK Radomil Model Transformation with Operational QVT [En línea]. Borland Software Corporation.[Citado 12 de Febrero de 2011]. Disponible en: http://www.eclipse.org/m2m/qvto/doc/M2M-QVTO.pdf.

ERRECALDE Marcelo Luis Agentes y Sistemas Multiagente 2009 [En línea]. Universidad Nacional de San Luis [Citado 6 de Agosto de 2010]. Disponible en: www.dirinfo.unsl.edu.ar/~sma/Teorias/teo5ag4.pdf.

FRIEDLER F. [y otros] Graph-Theoretic Approach to Process Synthesis Polynomial Algorithm for Maximal Structure Generation [En línea]. Department of Computer Science and Systems Technology, University of Pannonia, Hungary [Citado 23 de 09 de 2010]. Disponible en: http://dcs.vein.hu/cikkek/Grap-Theor_Appr_to_Proc_Synth_Polyn_Alg_for_Max_Struc_Gen.pdf.

FRIEDLER F. [y otros] Graph theoretic approach to process synthesis axioms and theorems [En línea]. Department of Computer Science and Systems Technology, University of Pannonia, Hungary [Citado 23 de Septiembre de 2010]. Disponible en: http://dcs.vein.hu/cikkek/Grap-Theor_Appr_to_Proc_Synth_Ax_and_Theor.pdf.

FRIEDLER F. [y otros] Graph-Theoretic Approach to Process Synthesis Polynomial Algorithm for Maximal Structure Generation [En línea].Department of Computer Science and Systems Technology, University of Pannonia, Hungary [Citado - 23 de Septiembre de 2010]. Disponible en: http://dcs.vein.hu/cikkek/Grap-Theor_Appr_to_Proc_Synth_Polyn_Alg_for_Max_Struc_Gen.pdf.

GARAVITO Oscar L., OCAMPO Juan D. y TORRES Miguel E. Metodología de Pruebas para Sistemas MultiAgentes (SMA) integrada a AOPOA [En línea].Universiad Nacional de Colombia Sede Medellín. [Citado 07 de Septiembre de 2010].Diponible en: http://pisis.unalmed.edu.co/3CCC/pdf/52.pdf.

GARCIA-OJEDA Juan Carlos [y otros] A Preliminary Study of the Application of the P-Graph Methodology in the Assesment of Organizational-Based Multiagent Systems Desing. - Bucaramanga : [s.n.], 01 de 04 de 2011.

GARCIA-OJEDA Juan Carlos [y otros] O-MaSE: A Customizable Approach to Developing Multiagent Development Processes.16 de Marzo de 2007.

GARCÍA-OJEDA Juan Carlos OMACS Overview, 02 de Septiembre de 2010.

GRONBACK Richard C. Eclipse modeling project : a domain-specific language (DSL) toolkit [Libro]. Massachusetts. Pearson Education, Inc., 2009.

INSFRÁN Emilio Tema 3 IntroDSL v8. - Valencia : [s.n.], 09 de Noviembre de 2009.

KAMINKA Gal A. Robots are Agents, Too! [En línea]. Agentlink [Citado 22 de Septiembre de 2010]. Disponible en: http://u.cs.biu.ac.il/~galk/Publications/Papers/agentlink04.pdf.

KERESSZEGI Attila P-graph. P-graph drawer (PNS Draw) [En línea]. P-graph. [Citado 16 de 09 de 2010]. Disponible en: http://www.p-graph.com/pnsdraw/screenshot.png.

KÖNEMANN Patrick A QVT model transformation for creating Model-Independent Diffs [En línea]. Technical University of Denmark [Citado 12 de Febrero de 2011]. Disponible en: http://modeldiff.imm.dtu.dk/modeldiff/images/docs/emfdiff2indepdiff.pdf.

MANGHAT Jaidev Simulation of power distribution management system using OMACS metamodel [En línea]. K-State Research Exchange [Citado 09 de Septiembre de 2010]. Disponible en: http://krex.k-state.edu/dspace/bitstream/2097/944/1/JaidevManghat2008.pdf.

MELLOULI Sehl FATMAS: A Methodology to Design Fault-tolerant Multi-agent Systems [En línea]. Networked Digital Library of Theses and Dissertation [Citado 14 de Septiembre de 2010]. Disponible en: http://www.theses.ulaval.ca/2005/22674/22674.pdf.

NEIMAT Taimour Al Why Projects Fail [En línea].Project Perfect [Citado 21 de Septiembre de 2010]. Disponible en: http://www.projectperfect.com.au/downloads/Info/info_it_projects_fail.pdf.

NWANA Hyacinth S. Software Agents: An Overview [En línea]. UMBC Agent Web [Citado 09 de Septiembre de 2010]. Disponible en: http://agents.umbc.edu/introduction/ao/.

ODELL James Agents and Objects [En línea]. James Odell [Citado 08 de Septiembre de 2010] Disponible en: http://www.jamesodell.com/Agents_and_Objects.pdf.

OMG Meta Object Facility (MOF) 2.0 Query/View/Transformation Specification [En línea]. OMG[Citado 22 de Febrero de 2011]. Disponible en: http://www.omg.org/spec/QVT/1.1/PDF/.

OMG Meta Object Facility (MOF) Core Specification [En línea]. OMG[Citado 22 de Febrero de 2011]. Disponible en: http://www.omg.org/spec/MOF/2.0/PDF/.

OMG Object Constraint Language [En línea]. OMG [Citado 22 de Febrero de 2011]. Disponible en: http://www.omg.org/spec/OCL/2.2/PDF.

PELENCHANO Vicente Tema 1 TCP -2008 UPV. Valencia: [s.n.],17 de Octubre de 2007.

PELENCHANO Vicente Tema 1 TCP -2008 UPV. Valencia: [s.n.], 17 de 20 de 2007.

QUINTERO Paulo C. Comparación de metodologías y arquitecturas de sistemas multiagente encontradas en 10 aplicaciones utilizadas en medicina y servicios de e-salud [En línea]. Scribd.com [Citado 07 de Noviembre de 2010]. Disponible en: http://www.scribd.com/doc/23897006/Comparacion-de-Metodologias-y-Arquitecturas-de-Sistemas-Multiagente-as-en-10-Aplicaciones-Utilizadas-en-Medicina-y-Servicios-de-E-salud.

RICHLEY Jeff GMF: Beyond the Wizards [En línea]. O'Reilly Media, Inc [Citado 22 de Octubre de 2010]. Diponible en: http://onjava.com/pub/a/onjava/2007/07/11/gmf-beyond-the-wizards.html.

ROMERO TERNERO Maria del Carmen Sistemas MultiAgente [En línea]. [Citado 06 de Septiembre de 2010]. Disponible en: http://www.dte.us.es/personal/mcromero/masredes/docs/SMARD.0910.mas.pdf.

SANZ Jorge J. GÓMEZ Metodologías para el desarrollo de sistemas multi-agente [En línea].Universidad Nacional de Educación a Distancia [Citado 08 de Septiembre de 2010]. Disponible en: http://cabrillo.lsi.uned.es:8080/aepia/Uploads/18/38.pdf.

SCHMIDT Douglas C. Model-Driven Engineering [En línea]. Washington University in St. Louis [Citado 21 de Octubre de 2010]. Disponible en: http://www.cs.wustl.edu/~schmidt/GEI.pdf.

SERRANO Ana García y OSSOWSKI Sascha Inteligencia Artificial Distribuida y Sistema Multiagente [En línea]. Universidad Politécnica de Madrid [Citado 09 de Septiembre de 2010]. Disponible en: http://www.dia.fi.upm.es/~agarcia/publications/archivos/REV3.pdf.

Skrypuch Neil Eclipse Modeling Framework Project (EMF) [En línea]. Eclipse.org [Citado 16 de Septiembre de 2010]. Disponible en: http://www.eclipse.org/modeling/emf/.

STEINBERG Dave [y otros] EMF: Eclipse Modeling Framework [Libro].[s.l.]: Addison-Wesley Professional, 2008.

SYCARA Katia P. Multiagent Systems [En línea].American Association for Artificial Intelligence [Citado 08 de Septiembre de 2010]. Disponible en: http://www.aaai.org/AITopics/assets/PDF/AIMag19-02-2-article.pdf.

TOLVANEN Juha-Pekka DSL in Practice [En línea]. Institut für Softwaretechnik und Theoretische Informatik [Citado 16 de Septiembre de 2010]. Disponible en: http://tfs.cs.tu-berlin.de/gtvmt08/Program/DSMinPractice_Tolvanen_29March2008.pdf.

VALLECILLO Antonio Model Driven Development [En línea]. Departamento Lenguajes y Ciencias de la Computación Universidad de Málaga [Citado - 22 de Septiembre de 2010]. Disponible en: http://www.lcc.uma.es/~canal/sabc/MDA-doctorado.pdf.

VARGA Virag [y otros] PNS Solutions: a P-Graph Based Programming Framework for Process Network Synthesis [En línea]. Department of Computer Science and Systems Technology, University of Pannonia, Hungary [Citado 23 de Septiembre de 2010]. Disponible en: http://www.dcs.vein.hu/cikkek/PNS_solutions_a_P-graph-based.pdf.

Weiß Gerard Adaptation and Learning in Multiagent Systems> Some Remarks and a Bibliography [En línea].CiteSeerX [Citado 09 de Septiembre de 2010]. Disponible en: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.45.1884&rep=rep1&type=pdf.

WILLINK Edward Bugzilla [En línea]. [Citado 06 de Febrero de 2011]. Disponible en: https://bugs.eclipse.org/bugs/show_bug.cgi?id=329389.

WOLTERINK Tjerk The Future of Software Engineering: Model Driven Engineering [En línea].Tjerk's Tech Blog [Citado 21 de Septiembre de 2010]. Disponible en: http://tjerktech.wordpress.com/2010/04/19/the-future-of-software-engineering-model-driven-engineering/.

ZOUFALY Federico Issues and Challenges Facing Legacy Systems [En línea]. [Citado 21 de Octubre de 2010]. Disponible en: http://www.developer.com/mgmt/article.php/1492531/Issues-and-Challenges-Facing-Legacy-Systems.htm.

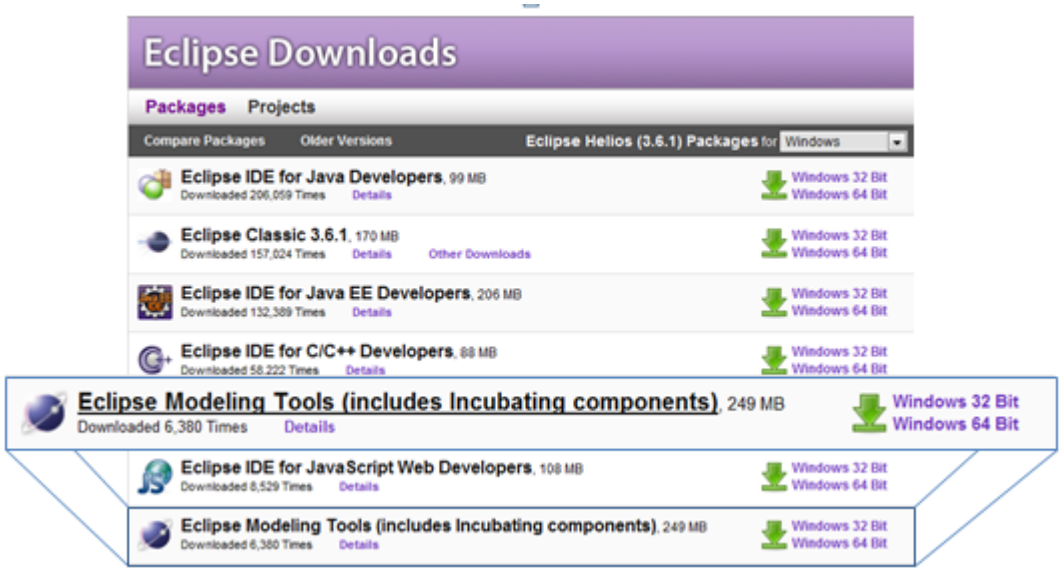
ANEXOS

ANEXO A. GENERALIDADES DE ECLIPSE

A.1 DESCARGA E INSTALACIÓN DE ECLIPSE.

Para el desarrollo del proyecto se requiere Eclipse IDE más específicamente el Eclipse Modeling Tools que permitirá el desarrollo de metamodelos, scripts de transformaciones de modelo a modelo, generadores de código, editores gráficos y creación de plug-ins sin necesidad de descargar paquetes innecesarios que corresponden a otros ámbitos. El Eclipse Modeling Tools se puede obtener directamente de la página de descargas de Eclipse.[35]

Figura 32. Página de Descargas Eclipse



Fuente: Autor del Proyecto

Una vez seleccionado el sistema operativo y la versión se descarga del servidor más cercano.

[35] http://www.eclipse.org/downloads/

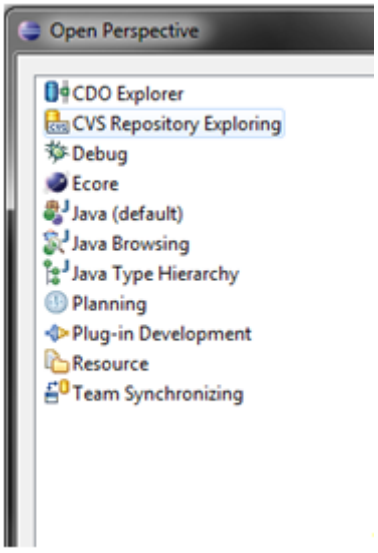
Figura 33. Selección Tipo de Descarga

Fuente: Autor del Proyecto

Una vez descargado el archivo se descomprime y se ejecuta el archivo dentro de la carpeta eclipse, es decir, eclipse no se instala en el ordenador ni modifica el archivo de registro siendo una herramienta totalmente portable, y la funcionalidad ofrecida depende de los plug-ins y los features incluidos.

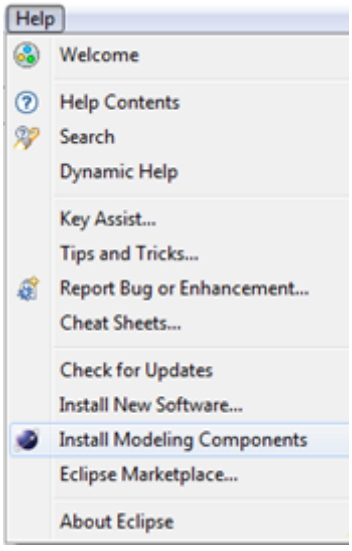
Una vez se esté ejecutando el eclipse Modeling Tools, éste carga las herramientas básicas para desarrollo en java y otras especializadas para trabajo con metamodelos.

Figura 34. Perspectivas Disponibles por Defecto



Fuente: Autor del Proyecto

Figura 35. Instalar Componentes de Modelado

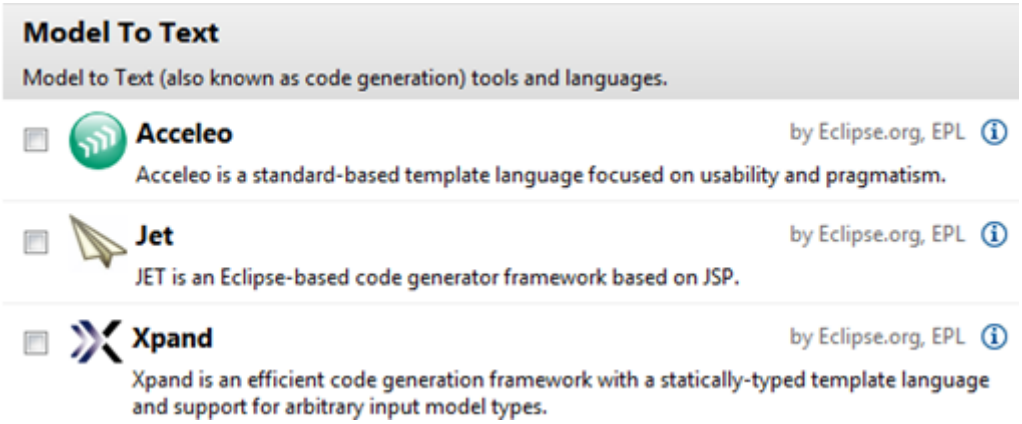


Fuente: Autor del Proyecto

Sin embargo existe la posibilidad de descargar más paquetes para facilitar el trabajo en cuanto a definición grafica de metamodelos, generación de editores

gráficos, transformaciones de modelos, etc. que se pueden obtener por medio del menú de Help. Una vez se selecciona Install Modeling Components aparecerá un dialogo que separa por propósito las herramientas de modelado disponibles.

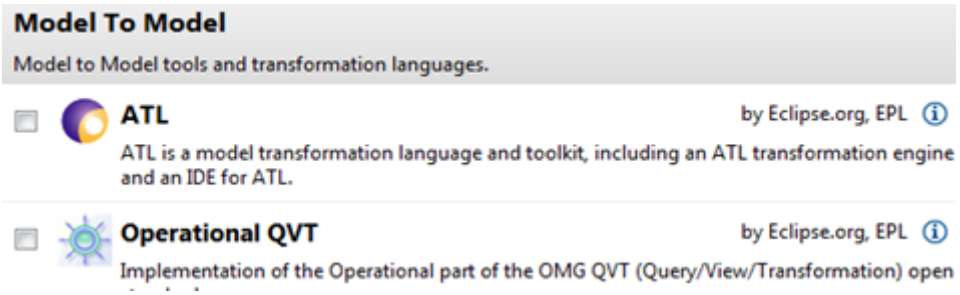
Figura 36. Plug-ins para Transformación M2T



Fuente: Autor del Proyecto

Las herramientas de transformación de modelo a texto permiten obtener código a partir de modelos basados en los metamodelos definidos por el usuario. Cada una de estas herramientas trabaja a base de scripts y templates de transformación de transformación distintos y deben ser seleccionadas cuidadosamente, ya que cada uno tiene ciertos alcances y limitaciones que se deben tener en cuenta al momento del desarrollo del proyecto.

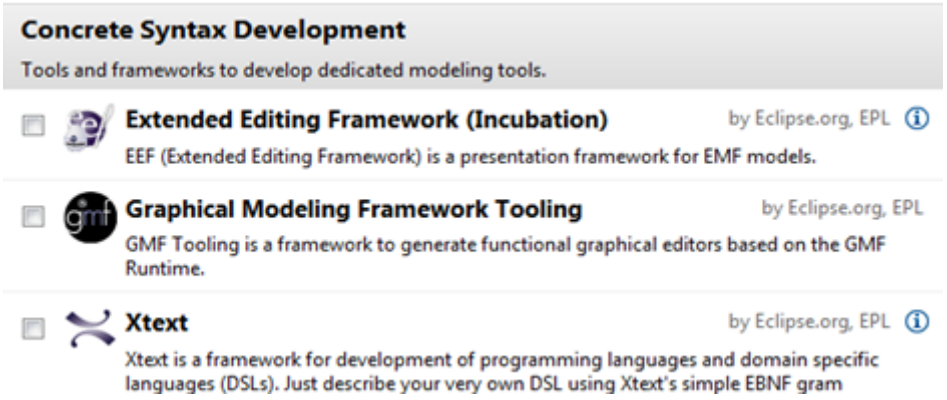
Figura 37. Plug-ins para M2M



Fuente: Autor del Proyecto

Similar a los lenguajes de transformaciones de modelo a texto, las herramientas de transformación de modelo a modelo trabajan a base de scripts.

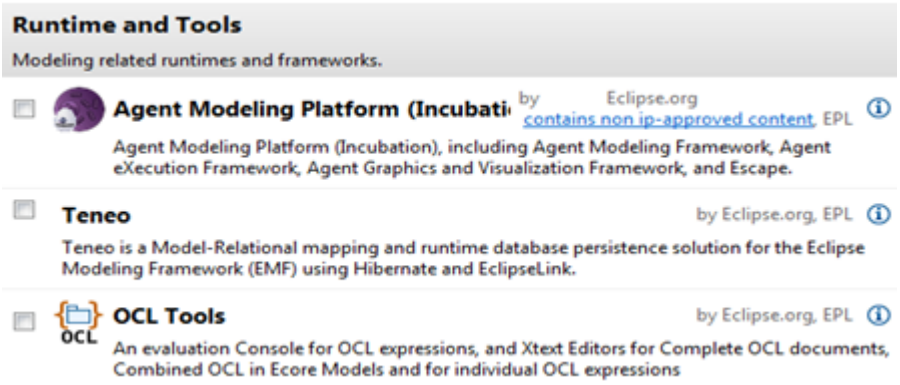
Figura 38. Plug-ins Definición Sintaxis Concreta



Fuente: Autor del Proyecto

Las herramientas de Definición de Sintaxis Concreta permiten la creación de editores gráficos correspondientes al Metamodelo

Figura 39. Plug-ins Relacionadas al Modelado



Fuente: Autor del Proyecto

Estas herramientas complementarias permiten expandir las posibilidades de modelado, por ejemplo las OCL Tools permiten un aprovechamiento real del lenguaje OCL en metamodelos ecore. A su vez el Agent Modeling Platform ofrece una alternativa de definición de MAS a la planteada anteriormente AgentTool III.

A.2 PAQUETES A INSTALAR PARA EL DESARROLLO DEL TRABAJO FINAL

Para este trabajo se utilizó el framework de definición de modelos EMF, para el diseño e implementación de los metamodelos genéricos OMACS y PNS. Una vez las instancias de los metamodelos se ajustaron a las especificaciones que requieren tanto OMACS como PNS Graph y PNS Studio se procedió a la creación de los scripts de transformación basados en los metamodelos. Como herramienta de apoyo de EMF se utilizaron los paquetes de GMF para simplificar la definición de modelos (MAS - PNS) por medio de herramientas de edición gráfica.

Para la elaboración de los scripts de transformación de modelo a modelo se eligió QVToperational ofrecido dentro de los paquetes de modelado de Eclipse IDE. Una vez obtenidos los scripts de transformación se planea publicar el trabajo como un plug-in para Eclipse IDE que tomará como entrada un modelo MAS y obtendrá como salida un modelo PNS.

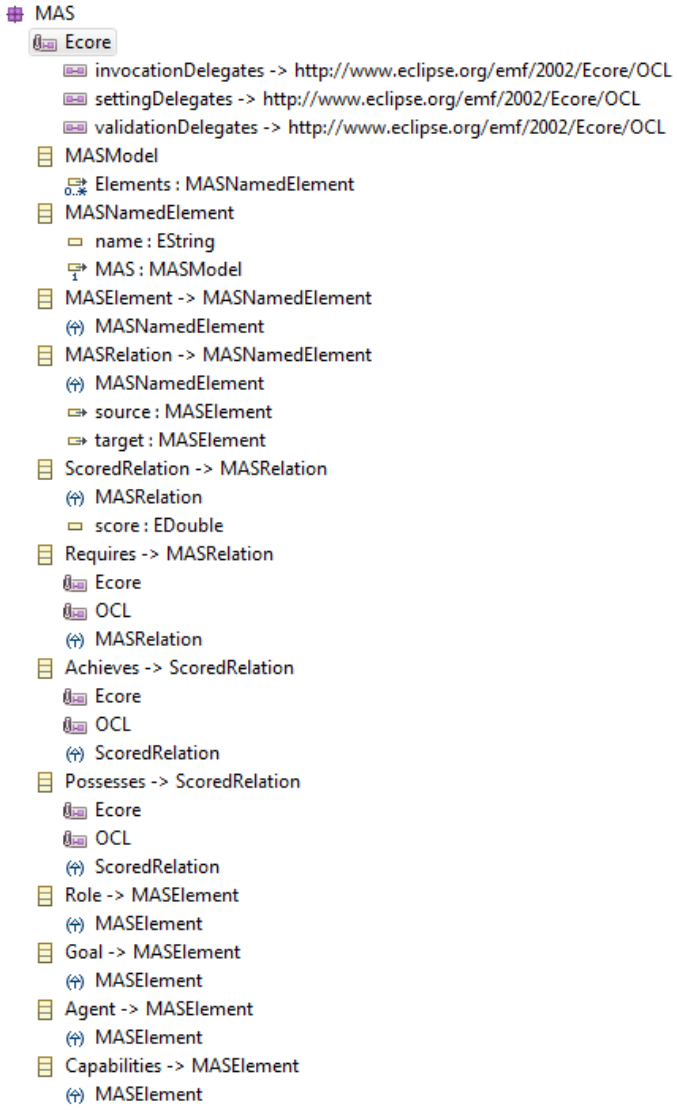
Todos los elementos mencionados vienen incluidos en la distribución por defecto del Eclipse Modeling Tools, si no se cuenta con esta versión existe varias maneras de instalarlo desde el Update Site de Eclipse.

ANEXO B. DESARROLLO DE LA APLICACIÓN

B.1 METAMODELO MAS

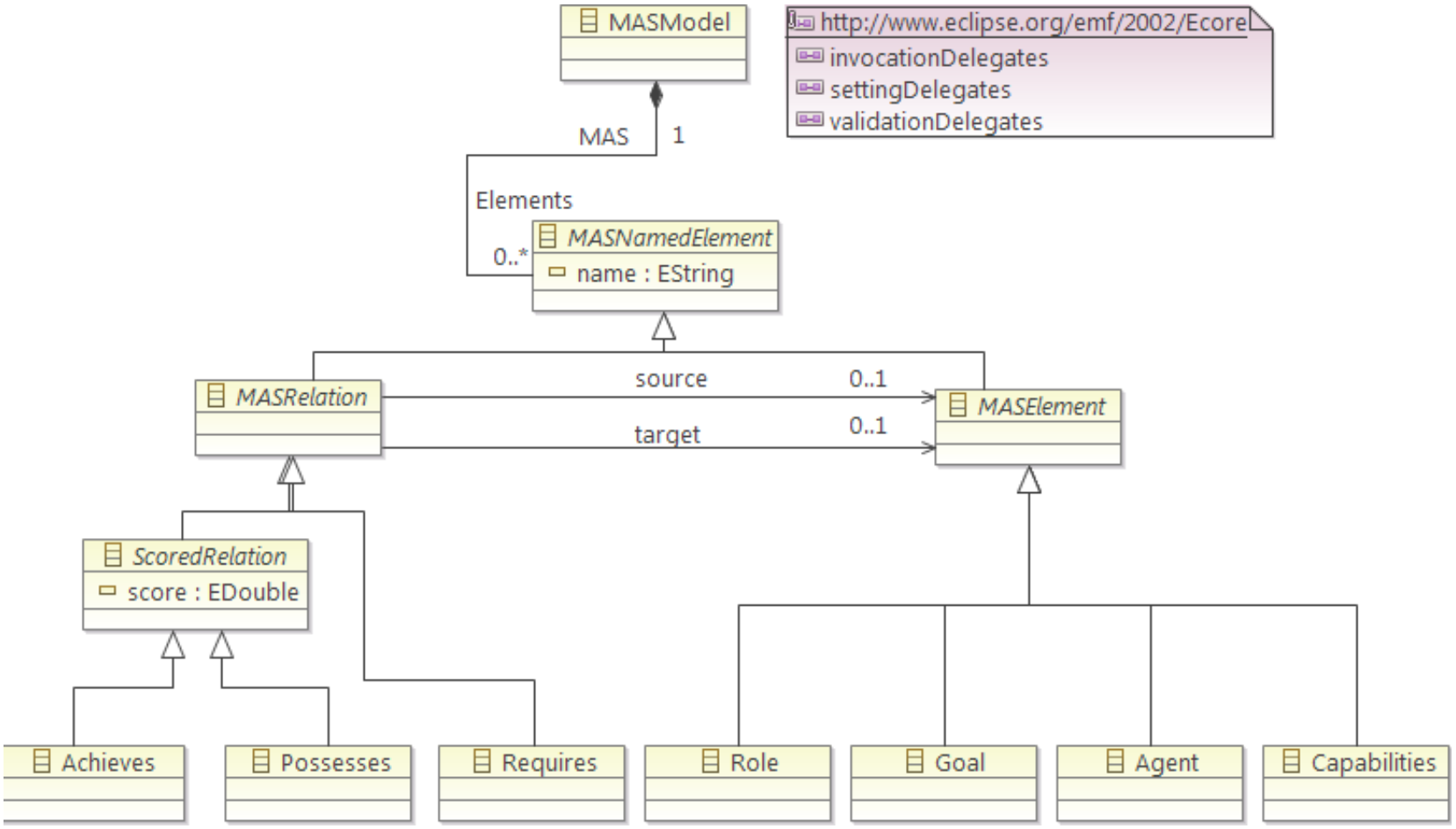
El código fuente del Metamodelo MAS así como algunas de sus diferentes representaciones serán expuestos en este apartado.

Figura 40. Metamodelo MAS Representación Tipo Árbol



Fuente: Autor del Proyecto

Figura 41. Metamodelo MAS Representación Gráfica



Fuente: Autor del proyecto

Tabla 1. Código Metamodelo MAS OCLinEcore

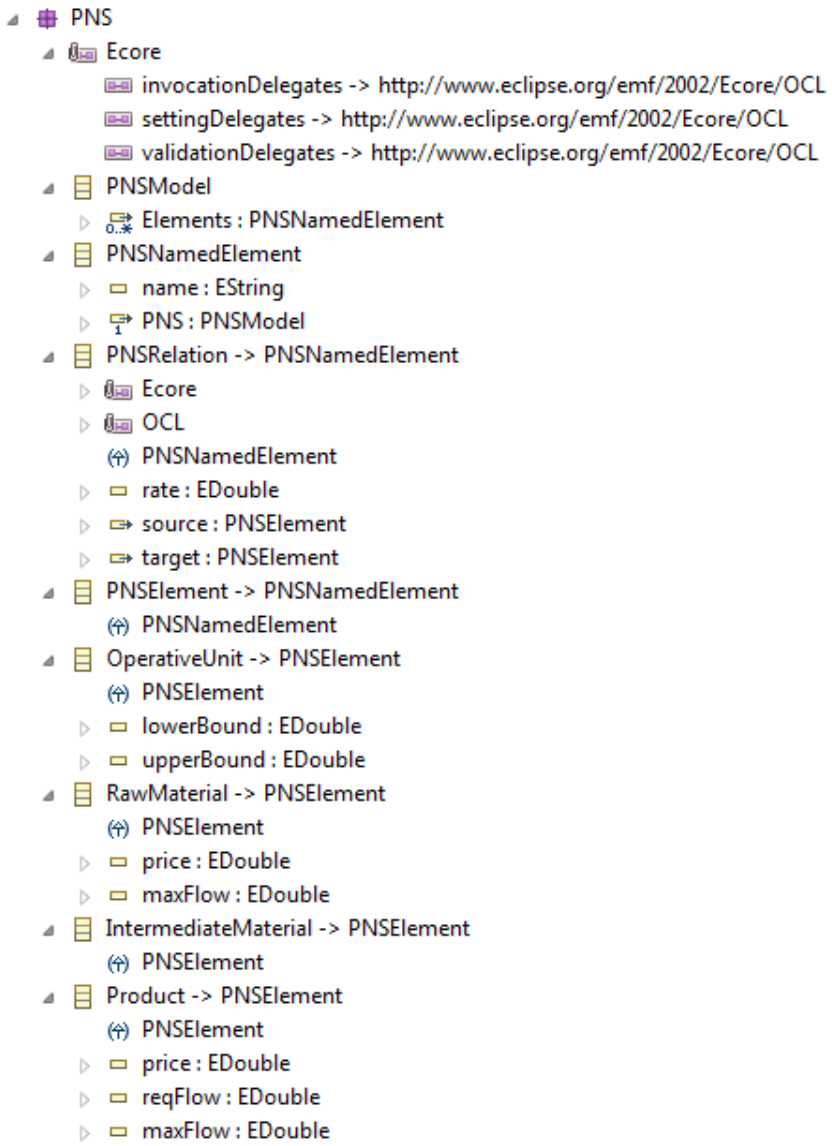
```
import ecore_0 : 'http://www.eclipse.org/emf/2002/Ecore#/';

package MAS : MAS = 'http://MAS/1.0'
{
	class MASModel
	{
		property Elements#MAS : MASNamedElement[*] { composes };
	}
	class MASNamedElement { abstract }
	{
		attribute name : String[?];
		property MAS#Elements : MASModel[1];
	}
	class MASElement extends MASNamedElement { abstract };
	class MASRelation extends MASNamedElement { abstract }
	{
		property source : MASElement[?];
		property target : MASElement[?];
	}
	class ScoredRelation extends MASRelation { abstract }
	{
		attribute score : ecore_0::EDouble[?] { !ordered,!unique };
	}
	class Requires extends MASRelation
	{
		invariant RequiresSourceType:
			self self.source->forAll(a|a.oclIsKindOf(Role));
		invariant RequiresTargetType:
			self.target->forAll(a|a.oclIsKindOf(Capabilities));
	}
	class Achieves extends ScoredRelation
	{
		invariant AchievesSourceType:
			self self.source->forAll(a|a.oclIsKindOf(Role));
		invariant AchievesTargetType:
			self self.target->forAll(a|a.oclIsKindOf(Goal));
	}
	class Possesses extends ScoredRelation
	{
		invariant PossessesSourceType:
			self self.source->forAll(a|a.oclIsKindOf(Agent));
		invariant PossessesTargetType:
			self self.target->forAll(a|a.oclIsKindOf(Capabilities));
	}
	class Role extends MASElement;
	class Goal extends MASElement;
	class Agent extends MASElement;
	class Capabilities extends MASElement;
}
```

Fuente: Autor del Proyecto

B.2 METAMODELO PNS

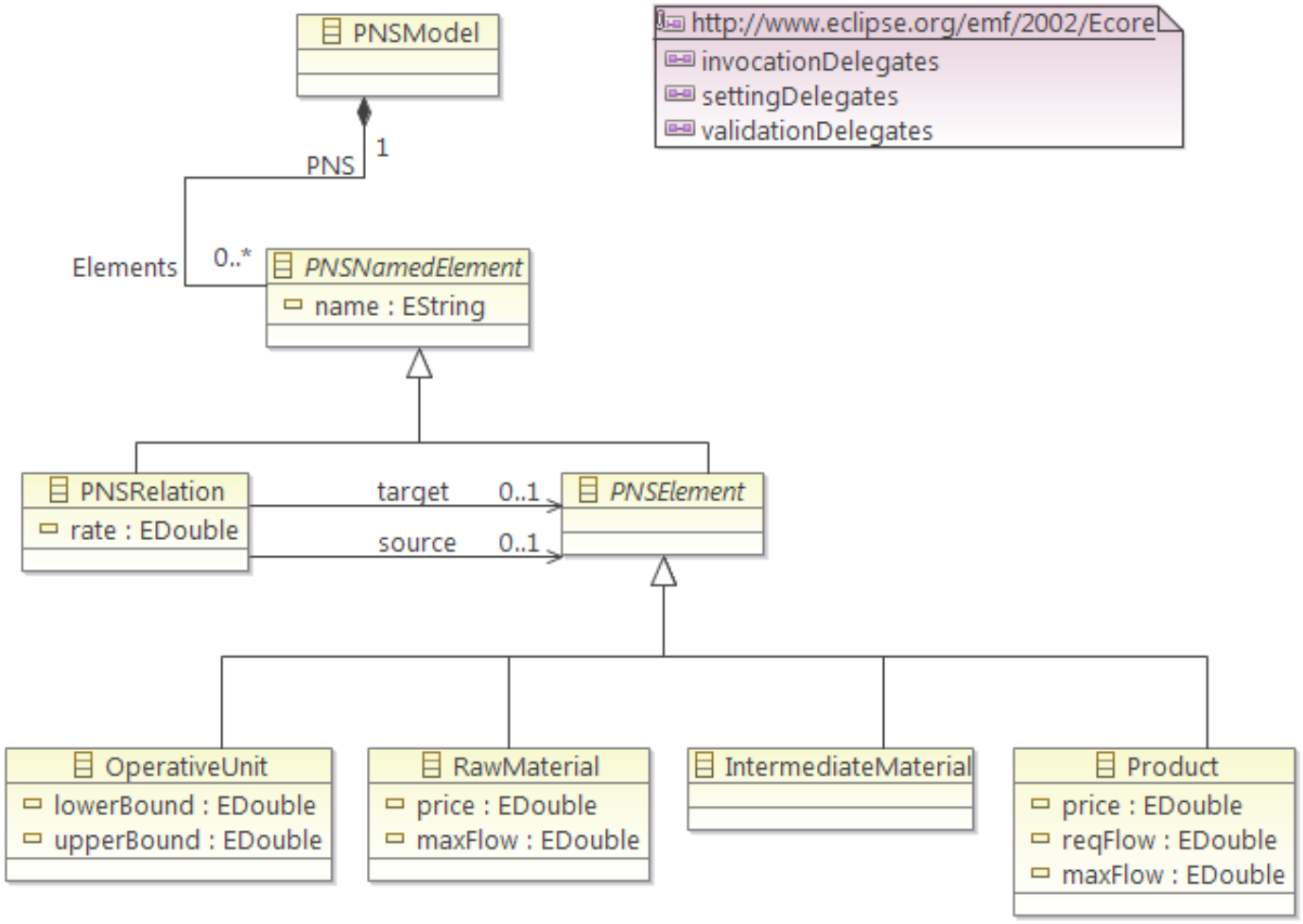
El código fuente del Metamodelo PNS así como algunas de sus diferentes representaciones serán expuestos en este apartado.

Figura 42. Metamodelo PNS Representación Tipo Árbol



Fuente: Autor del Proyecto

Figura 43. Metamodelo PNS Representación Gráfica



Fuente: Autor del Proyecto

Tabla 2. Código Metamodelo PNS OCLinEcore

```
import ecore_0 : 'http://www.eclipse.org/emf/2002/Ecore#/';

package PNS : PNS = 'http://PNS/1.0'
{
	class PNSModel
	{
		property Elements#PNS : PNSNamedElement[*] { composes };
	}
	class PNSNamedElement { abstract }
	{
		attribute name : String[?];
		property PNS#Elements : PNSModel[1];
	}
	class PNSRelation extends PNSNamedElement
	{
		invariant DestinationType:
		if	self.source.oclIsKindOf(RawMaterial) or
			self.source.oclIsKindOf(IntermediateMaterial)
		then
			self.target-> forAll(a|a.oclIsKindOf(OperativeUnit))
		else
			if self.source.oclIsKindOf(Product)
			then
				self.target->forAll(target.oclIsUndefined())
			else
				self.target ->
				forAll(a|a.oclIsKindOf(IntermediateMaterial)) or
				self.target -> forAll(a|a.oclIsKindOf(Product))
			endif
		endif;
		attribute rate : ecore_0::EDouble[?] { !ordered,!unique };
		property source : PNSElement[?];
		property target : PNSElement[?];
	}
	class PNSElement extends PNSNamedElement { abstract };
	class OperativeUnit extends PNSElement
	{
		attribute lowerBound : ecore_0::EDouble[?];
		attribute upperBound : ecore_0::EDouble[?];
	}
	class RawMaterial extends PNSElement
	{
		attribute price : ecore_0::EDouble[?];
		attribute maxFlow : ecore_0::EDouble[?];
	}
	class IntermediateMaterial extends PNSElement;
	class Product extends PNSElement
	{
		attribute price : ecore_0::EDouble[?];
		attribute reqFlow : ecore_0::EDouble[?];
		attribute maxFlow : ecore_0::EDouble[?];
	}
}
```

Fuente: Autor del Proyecto

B.3 TRANSFORMACIÓN DE MODELOS

A continuación se expondrá el código fuente de la transformación M2M que permite obtener un modelo PNS a partir de un modelo MAS.

Tabla 3. Código Fuente Transformación MAS->PNS

```
modeltype MAS uses 'http://MAS/1.0';
modeltype PNS uses 'http://PNS/1.0';

transformation MAS2PNS(in mas:MAS,out pns:PNS);

main()
{
	log("Incio Transformacion: " );
	mas.rootObjects()[MAS::MASModel]->map MASmodel2PNSmodel();
	log("Fin Transformacion: ");
}

mapping MAS::MASModel::MASmodel2PNSmodel():PNS::PNSModel
{
	result.Elements+=self.Elements.map MASNE2PNSNE(result,self);
}

mapping MAS::MASNamedElement::MASNE2PNSNE(p:PNS::PNSModel,m:MAS::MASModel)
disjuncts MASElement::MASElement2PNSElement
{
}

mapping MAS::MASElement::MASElement2PNSElement(p:PNS::PNSModel,m:MAS::MASModel)
disjuncts MAS::Agent::Agent2PNSElements
{
}

mapping MAS::Agent::Agent2PNSElements(p:PNS::PNSModel,m:MAS::MASModel)
when{self.oclIsTypeOf(Agent)}
{
--------------------------------------------------------------------------
	var CapabilitiesOfAgent:OrderedSet(MAS::Capabilities);
--Obtiene todas las Capabilities un Agente Posee
	var PossessesOfAgent:OrderedSet(MAS::Possesses);
--Obtiene todos los Possesses correspondientes a un Agente
	var RequiresOfCapabilities:OrderedSet(MAS::Requires);
--Obtiene todos los Requires correspondientes a una Capability
	var RoleOfRequires:OrderedSet(MAS::Role);
--Obtiene todos los Roles correspondientes a un Requires
	var AchievesOfRole:OrderedSet(MAS::Achieves);
--Obtiene todos los Achieves correspondientes a un Role
	var GoalOfAchieves:OrderedSet(MAS::Goal);
--Obtiene todos los Goals de un Achieves
```

```
        var Roles: OrderedSet(MAS::Role);
--Obtiene todos los Roles del modelo
        var RequiresOfRole:OrderedSet(MAS::Requires);
--Obtiene todos los Requires de un Role
        var CapabilitiesOfRole:OrderedSet(MAS::Capabilities);
--Obtiene todos los Capabilities de un Role
        var CoRvsCoA : OrderedSet(MAS::Capabilities);
--Interseccion entre CapabilitiesOfRole vs CapabilitiesOfAgent
        var PossessesOfRole : OrderedSet(MAS::Possesses);
--Obtiene todos los Possesses correspondientes a un Role
        var CapabilityCounter:Real;
 --Cuenta cuantas Capabilities pertenecen a un Agent
        var rm :RawMaterial;
--Objeto temporal de tipo RawMaterial permitira concatenar elementos
        var ou1:OperativeUnit;
--Objeto temporal de tipo OperativeUnit permitira concatenar elementos
        var im1:IntermediateMaterial;
--Objeto temporal de tipo IntermediateMaterial permitira concatenar elementos
        var ou2:OperativeUnit;
--Objeto temporal de tipo OperativeUnit permitira concatenar elementos
        var a1:Achieves;
--Objeto temporal de tipo Achieves permitira concatenar elementos
-------------------------------------------------------------------------
-----------------------------------------
--      Creo Raw Material L0
        rm := self.map Agent2RawMaterialL0(p,m);
-----------------------------------------
--Obtengo todas las relaciones Possesses del Agente que se esta procesando actualmente.
        PossessesOfAgent:=(m.Elements->select(b|b.oclIsTypeOf(Possesses) and
b.oclAsType(Possesses).source=self)).oclAsType(Possesses)->asOrderedSet();
        PossessesOfAgent->forEach(PA)
        {
--Obtengo todas las Capabilities que se relacionan con el Agente por medio de la relacion Possesses.
                CapabilitiesOfAgent+=(PA.oclAsType(Possesses).target.oclAsType(Capabilities));
        };
        CapabilitiesOfAgent->forEach(CA)
        {
--Obtengo los Requires que pertenecen a cada Capabilities
                RequiresOfCapabilities+=(m.Elements->select(c|c.oclIsTypeOf(Requires) and
c.oclAsType(Requires).target=CA)).oclAsType(Requires)->asOrderedSet();
        };
--Obtengo los Roles del modelo
        Roles := (m.Elements->select(b|b.oclIsTypeOf(Role))).oclAsType(Role)->asOrderedSet();
        Roles -> forEach(R)
        {
--Obtengo los requires de cada Role
                RequiresOfRole := (m.Elements-> select(d|d.oclIsTypeOf(Requires) and
d.oclAsType(Requires).source = R)).oclAsType(Requires)->asOrderedSet();
                CapabilitiesOfRole := OrderedSet{};
                CapabilityCounter :=0;
                RequiresOfRole -> forEach(RoR)
                {
--Obtengo las capabilities asociadas a cada Role
                        CapabilitiesOfRole +=
                                m.Elements ->select(e|e.oclIsKindOf(Capabilities)
                                and RoR.target = e).oclAsType(Capabilities)->asOrderedSet();
                        CapabilityCounter := CapabilityCounter +1;
                };
```

```
--Comparo las Capabilities requeridas por el Role y las Capabilities que posee el Agent
                CoRvsCoA :=
                 (CapabilitiesOfRole ->intersection(CapabilitiesOfAgent))->asOrderedSet();
--Determina si el Agente a es capaz de cumplir el Role R
                if CoRvsCoA = CapabilitiesOfRole then
                {
                        PossessesOfRole:= OrderedSet{};
                        CoRvsCoA ->forEach(CraC)
                        {
                                ------------------------------------------------
                                --  Creo Operative Unit L1
                                        ou1:= map newOperativeUnitL1(p,rm,R);
                                ------------------------------------------------
                                PossessesOfRole+=
                                 (m.Elements->select(f|f.oclIsTypeOf(Possesses)
                                        and f.oclAsType(Possesses).target=CraC
                                        and f.oclAsType(Possesses).source = self)
                                 ).oclAsType(Possesses)->asOrderedSet();
                        };
--El Agent a es capaz de cumplir con el Role R
                        log("El Agente "+self.name + " es capaz de cumplir con el Rol "+R.name);
--Obtengo todos los Achieves de cada Role R
                        AchievesOfRole+=(m.Elements->select(g|g.oclIsTypeOf(Achieves)
                                                and g.oclAsType(Achieves).source=R)
                                        ).oclAsType(Achieves)->asOrderedSet();
                        AchievesOfRole->forEach(ACH)
                        {
--Obtengo todos los Goal que alcanza cada Role
        GoalOfAchieves+=ACH.oclAsType(Achieves).target.oclAsType(Goal)->asOrderedSet();
                        };
                        GoalOfAchieves->forEach(G)
                        {
--Creo un intermediate class para representar los objetos que se usaran mas adelante.
                                var mx:=object MaterialX
                                {
                                        name:= self.name + R.name + G.name;
                                        score:=(PossessesOfRole.score->sum()
                                                        /CapabilityCounter);
                                };
                                var im:IntermediateMaterial;
                                var ou:OperativeUnit;
                                var pd:Product;

                        ------------------------------------------------
                        --  Creo Relation L0-1
                                map newRelationL01(p, rm, ou1);
                        ------------------------------------------------
                        --  Creo Intermediate Material L2
                                im1:= map newMaterialIntermediateL2(p,mx);
                        ------------------------------------------------
                        --  Creo Relation L1-2
                                map newRelationL12(p,ou1,im1, mx);
                        ------------------------------------------------
                        --  Creo Operative Unit L3
                                ou2 := map newOperativeUnitL3(p,im1);
                        ------------------------------------------------
                        --  Creo Relation L2-3
                                map newRelationL23(p,im1,ou2);
```

```
                        ------------------------------------------
                        --  Creo Intrermediate Material L4
                                im:=G.map Goal2IntermediateMaterialL4(p);
                        ------------------------------------------
                        a1 := AchievesOfRole ->
                                select(w|w.source = R
                                        and w.target = G
                                        ).oclAsType(Achieves)->asOrderedSet()->first();
                        ------------------------------------------------
                        --  Creo Relation L3-4
                                map newRelationL34(p, ou2, im, a1);
                        ------------------------------------------------
                        --  Creo Operative Unit L5
                                ou:=G.map Goal2OperationalUnitL5(p);
                        ------------------------------------------------
                        --  Creo una Relation L4-5
                                map newRelationL45(p,im,ou);
                        ------------------------------------------
                        --  Creo Product L6
                                pd:=G.map Goal2ProductL6(p);
                        ------------------------------------------
                        --  Creo Relation L5-6
                                map newRelationL56(p,ou,pd);
                        ------------------------------------------
                        };
                }
                else
                {
--El Agent a no es capaz de cumplir con el Role R
                log("El Agente "+self.name + " NO es capaz de cumplir el Rol "+R.name);
                }
                endif;
        };
}

intermediate class MaterialX
{
        name: String;
        score: Real;
}

mapping MAS::Agent::Agent2RawMaterialL0(p:PNS::PNSModel,m:MAS::MASModel):PNS::RawMaterial
when{self.oclIsTypeOf(Agent)}
{
        result.name:='L0_'+self.name;
        result.PNS:=p;
        result.price:=1.0;
        result.maxFlow:=10000.0;
}
mapping newRelationL01(p:PNS::PNSModel,rm:PNS::RawMaterial, ou:PNS::OperativeUnit):PNS::PNSRelation
{
        result.PNS := p;
        result.name :='R_'+rm.name +"->"+ou.name;
        result.source :=rm;
        result.target :=ou;
        result.rate := 1.0;
}
```

```
mapping newOperativeUnitL1(p:PNS::PNSModel, rm:PNS::RawMaterial, R:MAS::Role):PNS::OperativeUnit
{
        result.name:='L1_op_'+rm.name+'_'+R.name;
        result.PNS:=p;
        result.lowerBound:=10000.0;
        result.upperBound:=10000.0;
}

mapping newRelationL12(p:PNS::PNSModel,ou:PNS::OperativeUnit,im:PNS::IntermediateMaterial,
k:MaterialX):PNS::PNSRelation
{
        result.name :='R_'+ou.name +"->"+im.name;
        result.PNS := p;
        result.source :=ou;
        result.target :=im;
        result.rate := k.score;
}

mapping newMaterialIntermediateL2(p:PNS::PNSModel, k:MaterialX):PNS::IntermediateMaterial
{
        result.name := 'L2_'+k.name;
        result.PNS :=p;
}

mapping
newRelationL23(p:PNS::PNSModel,im:PNS::IntermediateMaterial,ou:PNS::OperativeUnit):PNS::PNSRelation
{
        result.name:='R_'+im.name+'->'+ou.name;
        result.PNS:=p;
        result.source:=im;
        result.target:=ou;
        result.rate:=1.0;
}
mapping newOperativeUnitL3(p:PNS::PNSModel, im:PNS::IntermediateMaterial):PNS::OperativeUnit
{
        result.name:='L3_op_'+im.name;
        result.PNS:=p;
        result.lowerBound:=10000.0;
        result.upperBound:=10000.0;
}

mapping newRelationL34 (p:PNS::PNSModel, ou:PNS::OperativeUnit,im:PNS::IntermediateMaterial,
A:MAS::Achieves):PNS::PNSRelation
{
        result.name :='R_'+ou.name +"->"+im.name;
        result.PNS := p;
        result.source :=ou;
        result.target :=im;
        result.rate := A.score;
}

mapping MAS::Goal::Goal2IntermediateMaterialL4(p:PNS::PNSModel):PNS::IntermediateMaterial
{
        result.name:='L4_'+self.name;
        result.PNS:=p;
}

mapping
```

```
newRelationL45(p:PNS::PNSModel,im:PNS::IntermediateMaterial,ou:PNS::OperativeUnit):PNS::PNSRelation
{
	result.name:='R_'+im.name+'->'+ou.name;
	result.PNS:=p;
	result.source:=im;
	result.target:=ou;
	result.rate:=1.0;
}

mapping MAS::Goal::Goal2OperationalUnitL5(p:PNS::PNSModel):PNS::OperativeUnit
{
	result.name:='L5_op_'+self.name;
	result.PNS:=p;
	result.lowerBound:=10000.0;
	result.upperBound:=10000.0;
}

mapping newRelationL56(p:PNS::PNSModel,ou:PNS::OperativeUnit,pd:PNS::Product):PNS::PNSRelation
{
	result.name:='R_'+ou.name+'->'+pd.name;
	result.PNS:=p;
	result.source:=ou;
	result.target:=pd;
	result.rate:=1.0;
}

mapping MAS::Goal::Goal2ProductL6(p:PNS::PNSModel):PNS::Product
{
	result.name:='L6_'+self.name+'_oaf';
	result.PNS:=p;
	result.price:=1.0;
	result.reqFlow:=0.0;
	result.maxFlow:=10000.0;
}
```

Fuente: Autor del Proyecto

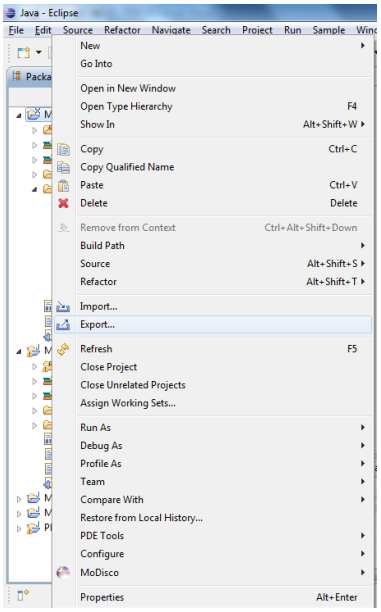
ANEXO C. INSTALACIÓN DE LA APLICACIÓN

C.1 LIBERACIÓN E INSTALACIÓN DE LOS PLUG-INS

Para la liberación de los plug-ins que se han creado durante el desarrollo de este trabajo Eclipse proporciona facilidades que permiten al desarrollador liberarlos cuando el trabajo ya esté concluido.

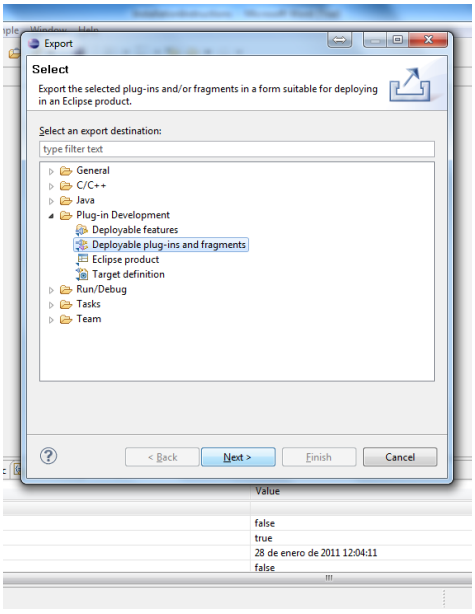
El primer paso para la liberación de los plug-ins es ubicar en el menú File la opción Export al dar clic sobre esta opción un dialogo aparece y permite seleccionar el origen de la exportación. Para este caso se ha de seleccionar Plug-in Development y seleccionar la opción Deployable Plug-ins and fragments.

Figura 44. Menú Exportar



Fuente: Autor del proyecto

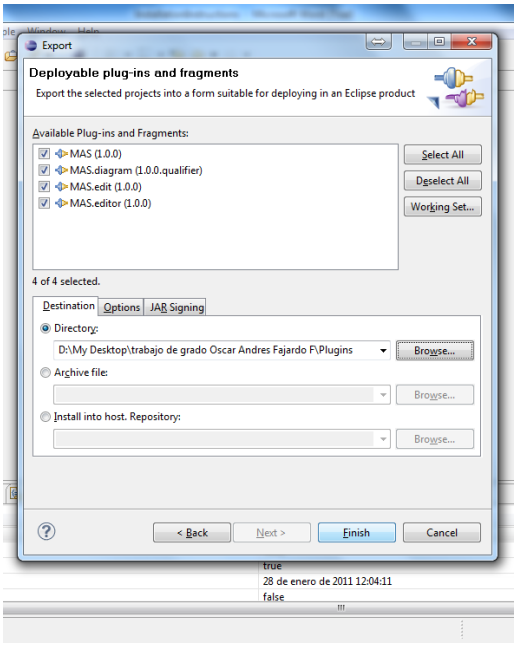
Figura 45. Exportar plug-ins



Fuente: Autor del proyecto

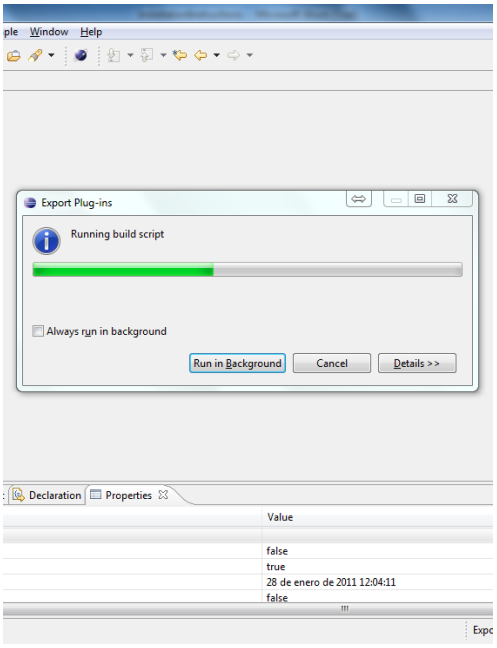
A continuación el dialogo presenta una lista de plug-ins disponibles para publicación, entre éstos debe encontrarse los plug-ins desarrollados en este trabajo. En la parte de debajo de este dialogo se puede elegir entre tres opciones de liberación. A un directorio (Directory), esta opción ubica los plug-ins en una carpeta especificada por el desarrollador. A un archivo (Archive File), ubica los plug-ins en un archivo comprimido ZIP y por ultimo al host (Install into Host) que realiza una instalación de los plug-ins en el entorno de desarrollo que en el que se está trabajando actualmente.

Figura 46. Selección plug-ins a exportar



Fuente: Autor del proyecto

Figura 47. Exportación e instalación plug-ins



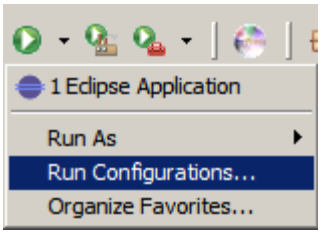
Fuente: Autor del proyecto

ANEXO D. MANUAL DE LA APLICACIÓN

D.1 CONFIGURACIÓN Y TRANSFORMACIÓN DE MODELOS

Una vez los plug-ins han sido instalados. El script de transformación está listo para ejecutarse y realizar la transformación MAS2PNS. Lo principal en este momento es configurar adecuadamente el intérprete de transformaciones QVT, para lograr esto se requiere acceder la siguiente ruta del menú.

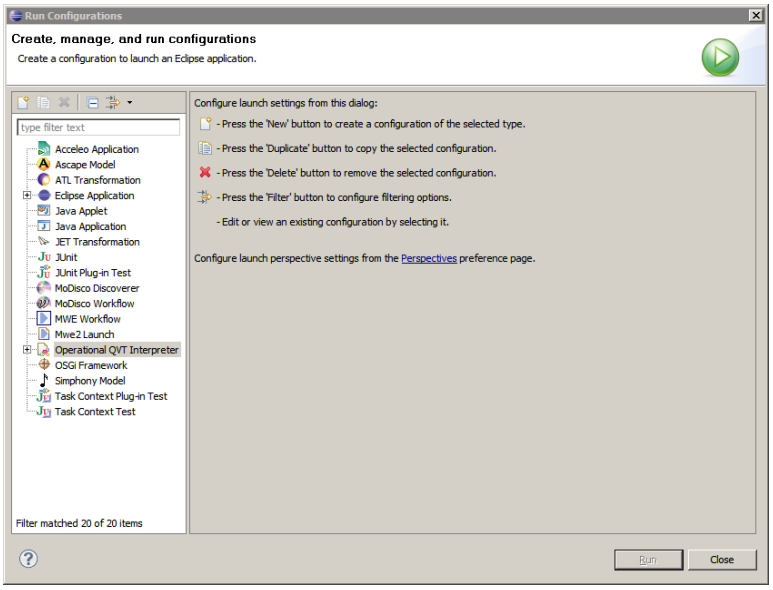
Figura 48. Menú de Configuración de Ejecuciones



Fuente: Autor del proyecto.

Una vez en la pantalla de configuración se puede observar una lista de configuraciones disponibles, sin embargo el único elemento de importancia en este momento es el Intérprete de QVT Operacional.

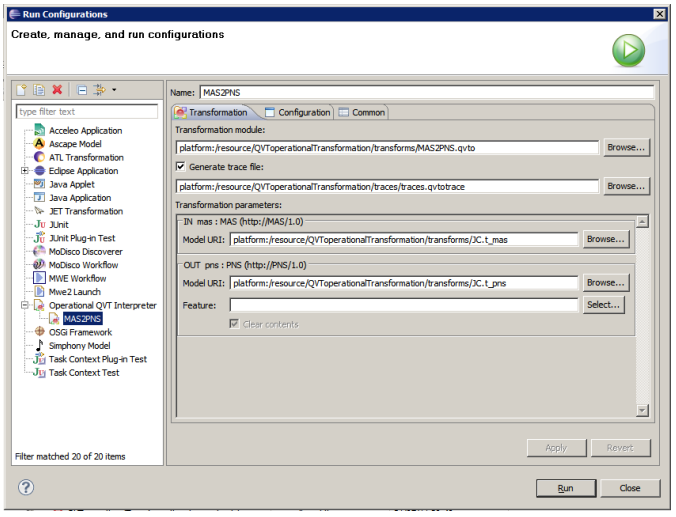
Figura 49. Pantalla de Configuraciones de Ejecución



Fuente: Autor del proyecto.

Una vez sobre el elemento se siguen las instrucciones y se crea un nuevo perfil de ejecución para el intérprete de QVTo.

Figura 50. Pantalla de Configuraciones de Ejecución



Fuente: Autor del proyecto.

Una vez en la pantalla de configuración se debe seleccionar la transformación QVT. El ‘Transformation Module’ nos solicita la ruta en la que se encuentran las fuentes de la transformación QVTo. Las fuentes de transformacion de este proyecto se denominan ‘MAS2PNS.qvto’.

La segunda opción determina si se quiere o no crear archivos de trazado. Estos archivos permiten identificar si una transformación fue llevada a cabo correctamente, al mejor estilo de prueba de escritorio, en el que se muestra paso a paso como se ejecutó la transformación y que elementos fueron transformados.

El bloque inferior ‘Transformation Parameters’, como lo indica su nombre son los parámetros de entrada/salida del algoritmo de transformación. Como la transformación es en una sola vía se debe especificar como modelo de entrada un modelo MAS compatible, y en el modelo de salida se debe especificar al menos una ruta destino, en el que se obtendrá un modelo PNS compatible.

Una vez finalizada la configuración se da clic sobre el botón de aplicar para salvar la configuración y poderla ejecutar inmediatamente dando clic sobre Ejecutar o luego por medio del menú de ejecución.