Estereotipos UML para Aplicar en un Ambiente de Simulación de Procesos Mineros

Andrea Giubergia[(1)], Daniel Riesco[(2)], Marcela Printista[(2)], Verónica Gil Costa[(1)]

[(1)]Dpto. de Minería, UNSL, Chacabuco 917 (5700) San Luis - Argentina
[(2)]Dpto. de Informática, UNSL, Ejercito de los Andes 950 (5700) San Luis - Argentina
{aagiuber; driesco; mprinti; gvcosta}@unsl.edu.ar

Abstract. El diseño de estructuras de un sistema y la descripción de su conducta se puede llevar a cabo eficientemente a través del Lenguaje de Modelado Unificado (UML), el cual permite, a su vez, introducir mecanismos de extensión (estereotipos y perfiles) inherentes al mismo lenguaje, para crear dominios específicos. En este trabajo, se muestra cómo esos mecanismos propuestos por UML para definir un metamodelo se aplican a un dominio concreto de simulación, referido al área de procesos mineros, específicamente a situaciones de carga y transporte de mineral. A partir de allí se proponen un conjunto de estereotipos, cuyas equivalencias con las clases de UML, se obtuvieron por ser éstas las que más se le aproximan semánticamente. Esto permite utilizar a UML como una técnica de pre-simulación para definir el dominio, garantizando un modelo bien formado, que responde a los requisitos sostenidos por la OMG.

Keywords: UML, simulación, estereotipos, diagramas de actividad

1 Introducción

Enfocar un sistema cualquiera desde distintos puntos de vista permite captar claramente la visión de lo que se quiere realizar. Esto sólo es posible a través de una notación sólida que sea comprendida de igual forma tanto por los analistas, desarrolladores y clientes. Tal función la desempeña eficazmente el lenguaje gráfico UML (en inglés, Unified Modeling Languaje) [1], siendo una herramienta que surge a mediados de 1990 como el estándar a seguir que cambió el mundo del software [6]. UML es un lenguaje

que conlleva reglas específicas, poseedor de semánticas y sintaxis propias.

Si bien UML ha logrado establecerse como una herramienta de modelado capaz de capturar el aspecto dinámico de un sistema, el lenguaje no puede, en forma precisa, especificar conceptos de determinados dominios. Es por ello que UML, incorpora el concepto de perfiles UML (UML Profiles), definido por la OMG (Object Management Group) en su especificación [2], como un conjunto predefinido de estereotipos, valores etiquetados, restricciones e iconos de notación que colectivamente especializan y adaptan UML a un dominio específico o proceso. Un conjunto de perfiles que amplían la semántica y la sintaxis de UML han sido estandarizados por la OMG [7] [8].

Acerca del dominio sobre el cual se va a realizar el metamodelo, puede decirse que, por ser una herramienta orientada al proceso, el desarrollo de modelos a través del software de simulación Arena [3], [4], [5] se estructura sobre una base gráfica asociada a la construcción de diagramas de flujo, que describe la serie de pasos que debe seguir una entidad conforme avanza en el sistema.

Como punto de partida, en este trabajo se define el metamodelo que representa el dominio de aplicación, lo cual se muestra en la Sección 2. Una vez definido el metamodelo se definen los estereotipos en la Sección 3. Por último, en la Sección 4 se incluyen las conclusiones.

2 Descripción del Dominio de Aplicación

En general, un software de simulación posibilita la construcción de los modelos a través de una serie de formas o bloques gráficos que permiten desarrollar las descripciones de los procesos que están involucrados en el sistema, también permiten, en algunos casos, acceder al código en el cual el modelo de simulación está construido. La estructura jerárquica se muestra en la Figura 1 donde puede observarse que el alto nivel de modelamiento implica menor flexibilidad porque incluye mayor grado de asistencia gráfica, en cambio, al descender el nivel de modelamiento se aumenta la flexibilidad porque disminuye la asistencia gráfica por la implementación de un código específico en este tema.

Desde esta perspectiva, el modelo se desarrolla utilizando las formas o bloques que forman parte de los procesos básicos, porque son los elementos básicos de construcción de modelos. La dinámica asociada a los procesos se puede describir como nodos de una red donde circulan las entidades causando un cambio en el estado del sistema hasta que finalmente salen del mismo. Las entidades, que tienen atributos y variables que permiten diferenciarlas entre ellas, compiten por los servicios que les brindan los recursos [4].

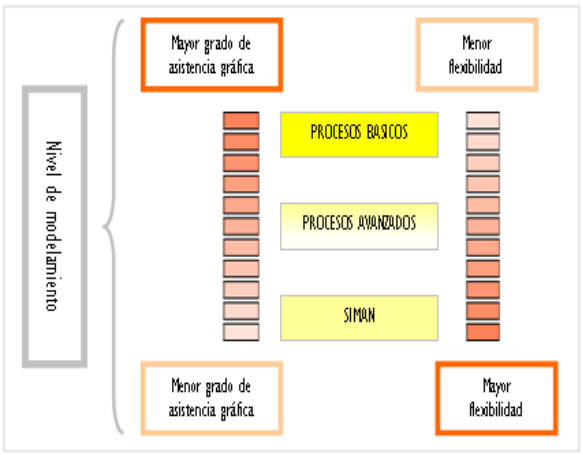

Fig. 1. Estructura jerárquica de un software de simulación

Siempre que una entidad ocupa un recurso lo debe liberar en algún momento en el modelo. En la Figura 2 se muestran algunos de los bloques imprescindibles para representar cualquier situación de simulación y sobre los cuales se van a establecer las equivalencias con las clases de UML.

Los bloques requieren cierto tipo de datos de entrada [3], que son imprescindibles para el buen funcionamiento de la simulación. En el caso que se trate de un punto de partida (Create) de dónde una entidad ingresa al sistema, requiere que se especifique el nombre y tipo de entidad, el tiempo entre llegadas, las unidades (minutos, segundos, horas, días), las unidades por llegada y el número máximo de arribo. Este bloque, a su vez tiene asociado un bloque de datos (Entity), que requiere un nombre, una imagen, datos de costo, entre otros. El bloque que se refiere al uso de un servidor (Process), requiere que se le indique el nombre del proceso, la acción, el tipo y cantidad de recursos, el tipo de demora y las unidades. Este bloque tiene asociado dos bloques de datos, un bloque referido a las colas de espera (Queue) y otro bloque referido al recurso empleado para llevar acabo un proceso determinado (Resource), los cuales tienen sus propios atributos. Una vez liberado el servidor, la entidad puede salir del sistema o regresar a él. El bloque de salida (Dipose) sólo requiere un nombre, sin necesidad de atributos adicionales. La Figura 3 muestra un ejemplo de cómo se relacionan los bloques para formar un modelo de simulación.

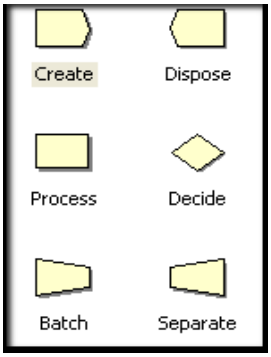

Fig. 2. Bloques básicos

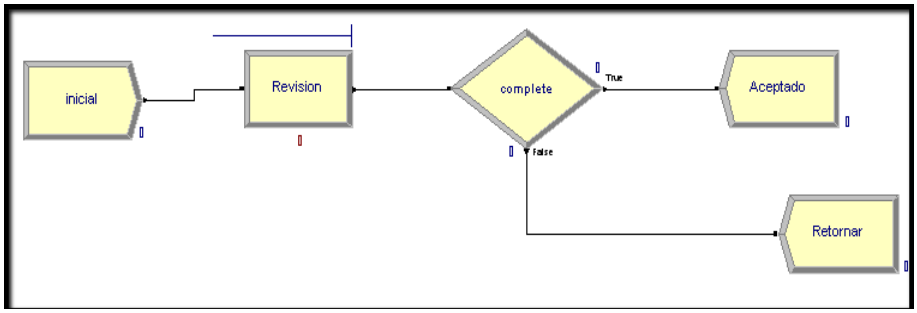

Fig. 3. Ejemplo de modelo de simulación

3 Definición de Estereotipos para el Dominio de Simulación de Procesos Mineros

A menudo sucede que es necesario adaptar UML a un dominio en particular. Para ello, UML se vale de un mecanismo denominado perfil, el cual incluye tres elementos: estereotipos, valores etiquetados y restricciones. Los estereotipos extienden el vocabulario de UML y es posible asociarle valores etiquetados (atributos asociados a los elementos extendidos) y restricciones [9], [13]. El principal constructor del perfil es el estereotipo, se indica como <<estereotipo>>, que le corresponde la misma estructura (atributos, asociaciones, operaciones) del metamodelo de UML, por lo cual no está

permitido modificar la semántica, la estructura y los conceptos originales [10] de UML estándar.

Los estereotipos propuestos para modelar el dominio de procesos básicos del software de simulación se corresponden de manera directa con los conceptos descriptos en el metamodelo. Los bloques a los cuales se aplica un estereotipo son aquellos bloques que resultan imprescindibles y necesarios en cualquier situación de simulación que vaya a crearse a posteriori y que requieren ser referenciados para incorporarlos en el perfil de UML.

Los estereotipos propuestos están relacionados con los componentes del *Diagrama de Actividad* [11], porque por su naturaleza se adapta sin dificultad para caracterizar el modelo que se quiere representar [12]. Este diagrama ya tiene establecido las notaciones y conceptos adecuados que muestra los pasos (actividades), puntos de decisión y bifurcaciones que ocurren en una operación, de tal forma que hace más sencilla su visualización. En la Figura 4 puede verse un ejemplo del Diagrama de Actividad donde se pueden identificar los nodos de contención (nivel donde ocurre las actividades fundamentales), las secuencias de control y flujo de datos entre acciones (nivel donde ocurren las actividades básicas) y la concurrencia de flujos y decisiones (nivel donde ocurren las actividades intermedias). A simple vista pueden apreciarse las similitudes generales con la Figura 3 y la razón por la cual se extienden las clases pertenecientes a Diagrama de Actividad en particular.

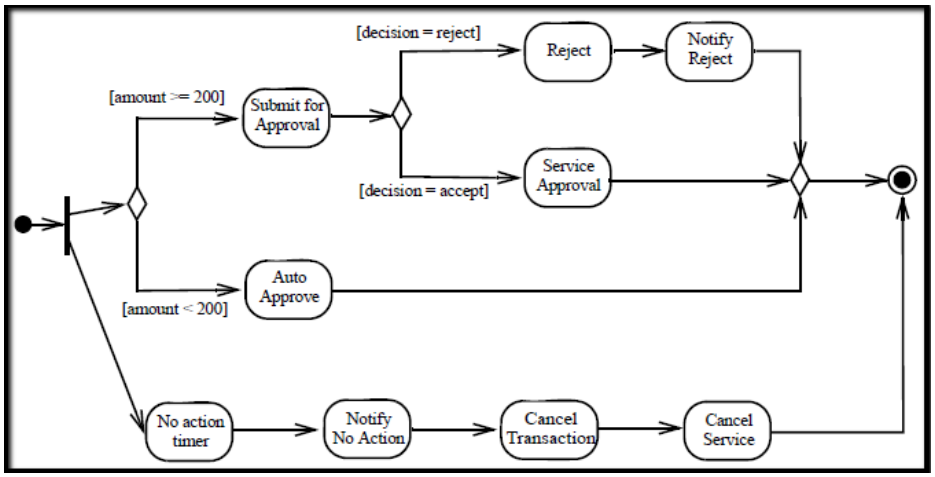

Fig. 4. Ejemplo Diagrama de Actividad de UML

El ejemplo de la Figura 3 se genera a partir de un diagrama de actividad, Figura 5, en un ambiente UML, donde se pone en evidencia la mayor expresividad de este diagrama, aportando un nivel más elevado de claridad en lo que se quiere describir. Por ello se considera que utilizarlo genera mayor nivel de certeza al momento de modelar un sistema de transporte minero en un ambiente de simulación, donde se establece la entrada y salida de camiones de la zona de carga.

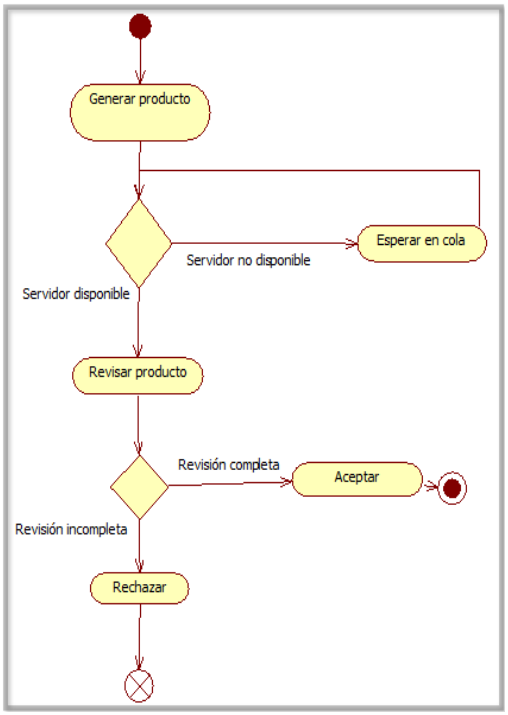

Fig. 5. Diagrama de Actividad Base

3.1 Estereotipos propuestos para las metaclases de UML

Una vez analizada la semántica para realizar las equivalencias correspondientes, se identifican las clases inherentes al Diagrama de Actividad que son necesario extender. Los softwares de simulación poseen un mayor número de bloques, además de los mencionados aquí, algunos de los cuales se identifican con otras clases de UML. En

este trabajo solo se consideran algunas de las clases, las más importantes, que se identifican como parte del Diagrama de Actividad.

El bloque Create se corresponde con la clase InitialNode de UML por su comportamiento similar. Un InitialNode es un punto de inicio para la ejecución de una actividad. En la especificación de UML, Superstructure v2.1.2 [11], se lo define como una generalización de un nodo de control desde donde un objeto comienza a fluir por el sistema cuando una actividad es invocada. En la Figura 6, perteneciente al paquete de Actividades Básicas definido en la sintaxis abstracta del Metamodelo del Kernel de UML de la OMG [11], se muestra esta relación. No tiene atributos adicionales. El bloque Create tiene una serie de atributos que definen los parámetros de entrada de la entidad al momento de iniciar la simulación. Esto hace que sea necesario extender la clase InitialNode a través de un estereotipo denominado <<Create>>.

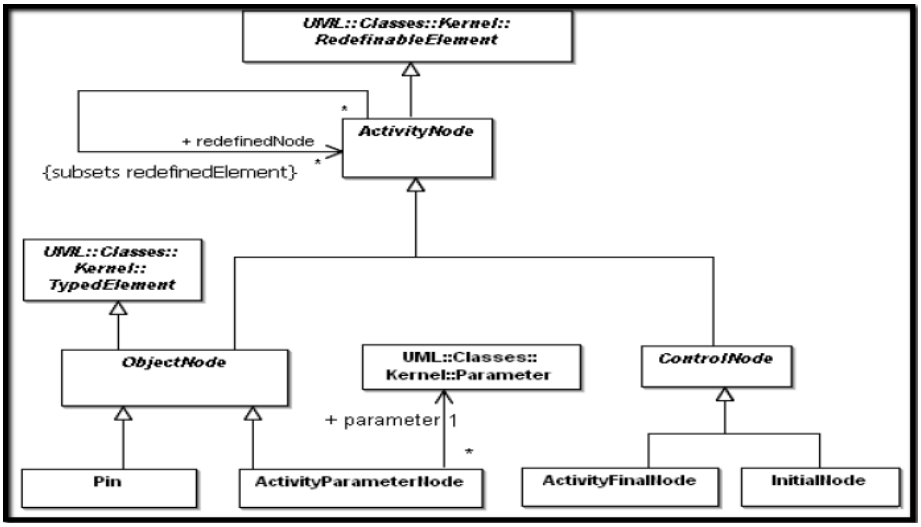

Fig. 6. Nodos: sintaxis abstracta definida en la especificación de UML

En el punto de inicio de un modelo de simulación se considera el tipo de flujo de llegada que se genera (si es al azar, programado, constante) así como también el número de unidades que arriban al sistema en la unidad de tiempo indicada. En el caso concreto de un proceso minero de carga y transporte, el punto de inicio estaría dado por cada camión que ingresa al sistema.

En un ambiente de simulación, los bloques se relacionan unos con otros a través de conectores que indica hacia dónde se dirige la entidad. Este conector aplica a la metaclase ControlFlow del Diagrama de Actividad, la cual no es necesario extender ya que la semántica se corresponde bilateralmente.

El bloque Process se corresponde con la clase **Activity** [11] de UML. Una actividad representa un tipo de proceso que está compuesta por elementos individuales que son acciones. Estas acciones pueden ser iniciadas porque otros procesos en esos modelos terminaron la ejecución, o porque los objetos y datos están disponibles o porque ocurren eventos externos al flujo. Si bien el bloque Process se comporta de manera similar, el bloque Process tiene algunos atributos propios que son necesarios definir. Razón por la cual se extiende la clase Activity, a través de un estereotipo denominado <<Process>> que contempla los atributos imprescindibles para el buen desempeño del modelo.

En el área donde se lleva acabo un proceso en un modelo de simulación se debe especificar la lógica dentro del modelo (si posee una solo modulo donde se lleva la acción o existe una jerarquía de modulos), el tipo de procesamiento que ocurrirá (si habrá demora, si se genera una cola, etc.) y la unidad de tiempo correspondiente. En el caso de procesos mineros referidos a carga y transporte, el bloque Process representaría la operación de cargar, a través de la pala, el camión que ingresa a la zona de carga del mineral.

El bloque Batch se corresponde con la clase **JoinNode** [11] de UML por su comportamiento similar. Un JoinNode es un nodo de control que sincroniza múltiples flujos, teniendo varias entradas y una única salida. Es necesario extender esta clase ya que el bloque Batch posee atributos adicionales que le son propios y que no están comprendidos en la semántica de la clase JoinNode de UML. Se extiende a través de un estereotipo denominado <<Batch>>. En un modelo de simulación se debe explicitar el número de entidades a ser agrupadas como así también cómo serán agrupadas esas entidades (se refiere a si las agrupa al azar o, por el contrario, las entidades deben agruparse según un atributo específico).

En el esquema de la Figura 7 se muestra, en forma resumida, cómo las instancias de los estereotipos deben siempre estar linkeadas a las instancias de las metaclases extendidas. Las metaclases corresponden al Diagrama de Actividad de UML. El esquema está basado en la jerarquía de capas de UML establecida en su Infraestructura [1].

Para comprender su aplicación en un sistema de transporte minero, si los conceptos del modelo del nivel M1 se trasladan a un caso en particular, entonces el estereotipo <<Create>> ya no sería Create 1, sino que se denominaría Camión A. En el caso del estereotipo <<Process>> se denominaría, en lugar de Process 1, Cargar Mineral. Esto puede aplicarse a cualquier yacimiento. En un caso concreto de una yacimiento con una flota de camiones específica, Camión A sería remplazado por la marca del

camión, por ejemplo CAT D25D. Si el yacimiento en explotación fuera caliza, la operación Cargar Mineral, sería Cargar Caliza.

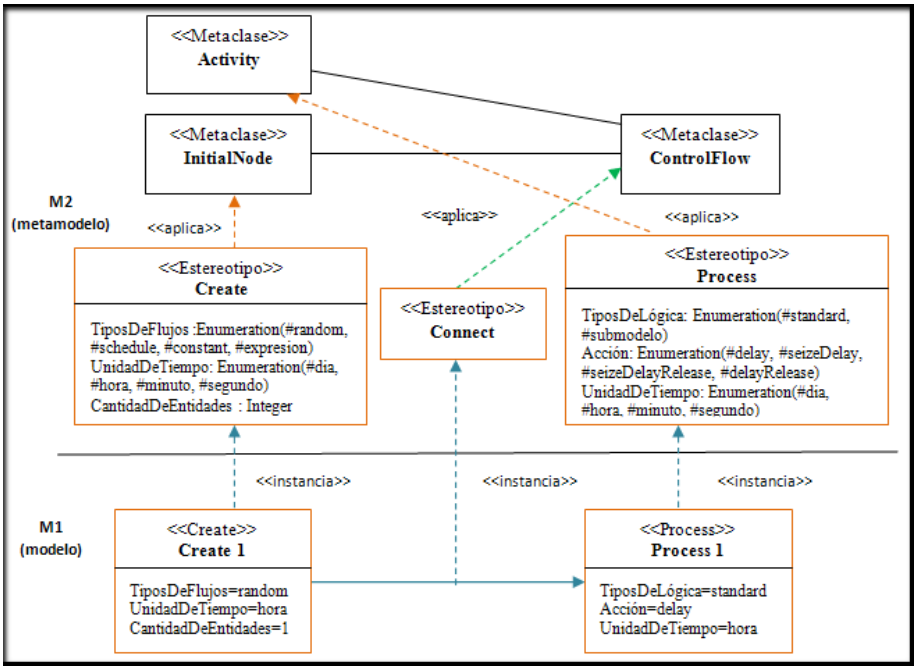

Fig. 7. Esquema representando las equivalencias entre metaclases y estereotipos basado en la jerarquía de UML establecida en la especificación de la OMG

4 Conclusiones

En este trabajo se presentaron una serie de estereotipos que forman parte de un modelo que se desarrolla en un ambiente de simulación. El hecho de utilizar un lenguaje estándar como UML para aplicarlo a este dominio en particular, implica que se obtiene un modelo con estereotipos que responden a reglas bien formadas. Es una ventaja contar con un modelo que no permita tener conceptos ambiguos, es decir, que no presente un sesgo en alguna dirección. Asimismo, permite generar código en distintos lenguajes de simulación como también permite basarse en una especificación precisa utilizando la estructura de la OMG para mostrar la semántica de la extensión

del lenguaje. El hecho de integrar la visión de estos dos lenguajes (UML y simulación) contribuye a construir un modelo consistente al proceso a ser simulado; por ello se utilizó la estructura de los diagramas de actividad, que son los que permiten modelar aspectos dinámicos y ser una abstracción de los programas de simulación.

Referencias

1. OMG Unified Modeling Languaje (OMG UML), Infrastructure, Version2.2, OMG Document Number: formal/2009-02-04 (2009)
2. OMG Unified Modeling Languaje (OMG UML), Infrastructure, Version2.2, OMG Document Number: formal/2009-02-04 http://www.omg.org/spec/UML/2.2/Infrastructure (2009)
3. Ataeepour M., Baafi E.Y.: Application of Arena simulation system to compare truck-shovel operation in dispatching and non-dispatching modes, Mine Planning and Equipment Selection (1998)
4. Teilans A., Kleins A., Merkuryev Y., Grinbergs A.: Design of UML models and their simulation using ARENA, WSEAS TRANSACTIONS ON COMPUTER RESEARCH, Issue 1, Volume 3, January 2008 (2008)
5. Arena User's Guide, Rockwell Software Inc. USA, October 2005, Rockwell Automation, http://www.rockwellsoftware.com (2005)
6. Watson A.: Visual Modeling: past, present and future, White paper UML Resource Page (2008)
7. OMG http://www.omg.org/technology/documents/profile_catalog.htm#UML_for_EDOC
8. UML Home Page, Object Management Group (OMG), UML Resource Page, http://www.uml.org
9. Debnath N.C., Garis A., Riesco D., Montejano G.: Defining Patterns Using UML Profiles, IEEE Conference Proceeding (2006).
10. Abdullah M.S., Paige R., Kimble C., Benest I.: A UML Profile for Knowledge-Based Systems Modelling, Fifth International Conference on Software Engineering Research, Management and Applications. IEEE (2007)
11. OMG Unified Modeling Languaje (OMG UML), Superestructure, Version2.1.2, OMG Document Number: formal/2007-11-02, Standard Document, http://www.omg.org/spec/UML/2.1.2/Superstructure/PDF (2007)
12. Bartolini C., Bertolino A., De Angelis G., Lipari G.: A UML Profile and a Methodology for Real-Time Systems Design, Proceedings of the 32nd EUHOMICHO Conference on Software Engineering and Advanced Applications. IEEE (2006)
13. Debnath N.C., Baigorria L., Riesco D., Montejano G.: Metrics applied to Aspect Oriented Design using UML profiles, Computers and Communications 2008, Symposium on Digital Object Identifier, pp. 654--657. IEEE (2008)