REPRESENTACIÓN DE VERSIONES DE MODELOS EN EL PROCESO DE DISEÑO

Silvio Gonnet[a], Horacio Leone[a], Gabriela Henning[b]

[a] GIPSI- UTN / INGAR, Avellaneda 3657, 3000 - Santa Fe, Argentina
[b] INTEC, Güemes 3450, 3000 - Santa Fe, Argentina
{sgonnet,hleone}@ceride.gov.ar, ghenning@intec.unl.edu.ar

1. Introducción

El presente trabajo fue motivado a partir del estudio del proceso de diseño ingenieril; proceso que en general es carente de estructura, y durante el mismo se genera un gran número de modelos de distintos niveles de abstracciones del objeto que se diseña, el cual puede consistir, según el dominio ingenieril, en un sistema de información, un equipo industrial, o una planta de procesos químicos. Los modelos generados difieren en granularidad, complejidad e hipótesis asociadas a los mismos; siendo necesario un mecanismo explícito que administre adecuadamente las distintas versiones de modelos generadas, permitiendo luego el seguimiento del proceso de diseño y la navegabilidad por las distintas versiones que surgieron durante tal proceso. Tal mecanismo es el centro de interés de investigación en el presente trabajo, el cual continúa los avances presentados en [2] y [3]. La principal contribución que se espera es la explicitación de un modelo que permita la captura y representación del proceso de diseño en términos de las operaciones aplicadas a las distintas versiones de los objetos que intervienen en el proceso de diseño. La definición del modelo debe ser lo suficientemente flexible para facilitar la especialización del mismo según un dominio de diseño particular, empleando los conceptos del dominio y las operaciones aplicables en el mismo. A partir de tal especialización se persigue la implementación de herramientas de soporte a las actividades del proceso de diseño, como puede ser una herramienta de soporte a las actividades de solución de problemas de ingeniería de procesos o de ingeniería de software.

Con el objeto de representar las diferentes versiones de modelo que son generadas durante el curso del proyecto de diseño se considera el cálculo situacional propuesto por McCarthy en 1963 [4], como un lenguaje de primer orden para razonar acerca de acciones, y si bien ha recibido muchas críticas acerca de su falta de generalidad para representar problemas del mundo real, recientes trabajos que se están realizando a partir del mismo están revirtiendo estas críticas [5,6]. Tal mecanismo debe extenderse con la tecnología de orientación a objetos [1] para poder representar adecuadamente como un producto del diseño evoluciona durante un proyecto dado y permitir la especialización del mismo para un dominio de diseño ingenieril en particular.

2. Esquema para la Representación de Versiones de Modelos en el Proceso de Diseño

El proceso de diseño puede concebirse como una secuencia de actividades que operan sobre los productos del proceso de diseño, denominados *objetos de diseño*. En este contexto se consideran *objetos de diseño* a los requerimientos, la representación del artefacto de diseño mismo, los argumentos de las decisiones tomadas, etc. En un instante dado, durante la realización del proyecto de diseño, los estados asumidos por el conjunto relevante de *objetos de diseños*, denominado *versión de modelo* (*m*), provee una descripción del estado del proceso de diseño, incluyendo el artefacto que está siendo diseñado.

En la Figura 1 se ilustra el esquema de representación de las versiones de los modelos generados durante el proceso de diseño. El ejemplo presentado se corresponde al dominio de ingeniería de procesos químicos, donde una estructura de entrada salida fue refinada en una sección de reacción y una sección de separación, y los flujos de entrada han sido modificados, especificando las características de los mismos. En la Figura 1 se ilustran tres niveles: *Repositorio*, *Versiones* y *Modelos Inferidos*. Como se puede apreciar, las distintas *versiones de modelos* son *inferidas* a partir de vistas sobre un *repositorio*. El *repositorio* contiene los distintos objetos que surgen durante el proceso, como así también las asociaciones que se establecen entre los mismos. Los objetos que

componen al repositorio se denominan *objetos versionables* (*o*). Sobre el *repositorio* se representa el nivel *versiones*, donde se definen las distintas versiones que se han generado para cada objeto existente en el repositorio, los objetos definidos en este nivel se denominan *versión de objeto* (*v*).

En la presente propuesta se considera que cada *versión de modelo* (*m*) se genera a partir de una *versión de modelo precedente* (m_p), sobre la cual se han aplicado una serie de *operaciones* que en general implican la eliminación, creación, modificación, etc., de las versiones de los objetos de diseño (es decir, de los componentes del modelo: objetos, clases, relaciones, en el caso de ingeniería de software; flujos, separadores, reactores, en el caso de ingeniería de procesos químicos). Resulta entonces, que toda *versión de modelo* puede tener cero o más *versiones de modelos sucesoras* (m_s) y posee una única *versión de modelo precedente*; a excepción de la *versión de modelo inicial* (m_0), la cual carece de *precedente*. En consecuencia, el esquema de representación de versiones tiene una estructura de árbol donde cada versión de modelo es un nodo del mismo y cuya raíz es la versión de modelo m_0.

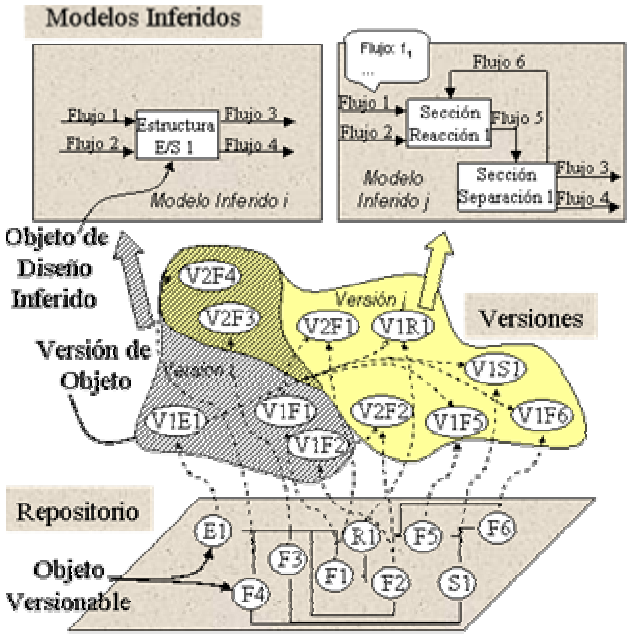

Figura 1. Esquema de Representación de Versiones de Modelos del Diseño

3. Esquema Formal para la Representación de Versiones de Modelo del Proceso de Diseño

Dado que se plantea la evolución de los modelos del diseño como una historia compuesta por situaciones discretas, se toma el cálculo situacional para el modelado del proceso de generación de versiones. La ontología básica del cálculo situacional consiste en situaciones, las cuales corresponden a fotografías del universo estudiado en un instante dado, y acciones, que producen evoluciones de una situación a otra. Se considera que cada nueva situación se genera a partir de una precedente por la aplicación de una acción. En el contexto dado por el proceso de diseño, cada nuevo modelo que se genera se puede asimilar con una situación, y una acción es el conjunto de operaciones que se realizaron sobre el modelo precedente.

Entonces, una nueva *versión de modelo* m_n se genera como consecuencia de la aplicación de una secuencia de operaciones básicas ϕ sobre los componentes de una *versión de modelo m*. Esto se logra ejecutando la siguiente evaluación: $aplicar(\phi, m) = m_n$. La función *aplicar* se define en la expresión 1, donde Φ es el conjunto de todas las secuencias de operaciones posibles, y M es el conjunto de las versiones de modelos posibles.

En la expresión 2 se define en forma recursiva una *secuencia de operaciones*, donde λ es una secuencia vacía, *o* es una operación, y • representa la concatenación entre una operación y una secuencia de operaciones.

$$aplicar: \Phi \times M \rightarrow M \qquad (1) \qquad \phi = \begin{cases} \lambda \\ o \bullet \phi \quad o : operación \end{cases} \qquad (2)$$

A partir de la expresión 2 se puede definir por inducción a la función *aplicar*:

$$aplicar(\lambda, m) = m$$
$$aplicar(o \bullet \lambda, m) = m_i,\ m \neq m_i \qquad (3)$$
$$aplicar(o \bullet \phi, m) = aplicar(\phi, aplicar(o \bullet \lambda, m))$$

Las operaciones primitivas propuestas inicialmente para representar la transformación de versiones de modelos son: *agregar*, *borrar*, y *modificar*. A través de la operación *agregar*(*v*) se puede incorporar a una *versión de modelo* una *versión de objeto* inexistente en la *versión de modelo*

anterior. Inversamente, la operación *borrar*(v) elimina una *versión de objeto* existente en la *versión de modelo anterior*. La operación *modificar*(v_a, v_s) crea una nueva *versión de objeto* v_s a partir de la *versión de objeto* previa v_a, donde v_s es una versión de objeto sucesora de v_a. Entonces, una versión de objeto v pertenece a la versión de modelo resultante de aplicar la secuencia de operaciones ϕ a la versión de modelo m si y solo si: (i) v fue agregada cuando se generó la nueva versión de modelo (*agregar*(v) $\in \phi$ o *modificar*(v_a, v) $\in \phi$); o (ii) v pertenecía a la versión de modelo previa m y no fue eliminada cuando se aplicó ϕ (*borrar*(v) $\notin \phi$) $\wedge$ (*modificar*(v, v_p) $\notin \phi$). Estas definiciones, se especifican formalmente en la expresión 4 utilizando el formato de los axiomas de estado sucesor propuesto por Reiter [5].

$$(\forall\ \phi, v, v_a, v_p, m)\ pertenece(v, aplicar(\phi, m)) \Leftrightarrow$$
$$((pertenece(v, m) \vee (agregar(v) \in \phi) \vee (modificar(v_a, v) \in \phi)) \wedge \quad (\mathbf{4})$$
$$((borrar(v) \notin \phi) \wedge (modificar(v, v_p) \notin \phi))$$

A partir de la expresión 4 es posible determinar que *versiones de objetos pertenecen* a una *versión de modelo* (*pertenece*(v, m)). Entonces, de esta manera se puede reconstruir una *versión de modelo* m_{i+1} aplicando todas las secuencias de operaciones desde la *versión de modelo inicial* m_0. En la expresión 5 se presenta tal reconstrucción, donde $\phi_i \bullet \phi_j$ representa la concatenación de las secuencias ϕ_i y ϕ_j.

$$m_{i+1} = aplicar(\phi_i, m_i);\ m_i = aplicar(\phi_{i-1}, m_{i-1});\ ...;\ m_1 = aplicar(\phi_0, m_0)$$
$$m_{i+1} = aplicar(\phi_i, aplicar(\phi_{i-1}, aplicar(...aplicar(\phi_0, m_0)...))) \quad (\mathbf{5})$$
$$m_{i+1} = aplicar(\phi_0 \bullet ... \bullet \phi_{i-1} \bullet \phi_i, m_0)$$

Una vez definidas que *versiones de objetos* conforman una versión de modelo, es necesario especificar que relaciones existen entre las mismas. Se debe notar, que en la presente propuesta las *versiones de objetos* pertenecientes a una versión de modelo no están explícitamente asociadas entre sí. En cambio, se define por cada objeto de diseño que fue agregado y/o modificado durante un proyecto de diseño un *objeto versionable* (o) que conoce todas sus *versiones de objetos* generadas a través del predicado *versión*(v, o) y que mantiene las asociaciones con los diversos objetos de diseño que surgieron durante el proceso de diseño. Estas asociaciones son capturadas por el predicado *asociación*(a_k, o_i, o_j) que significa que la asociación a_k relaciona al objeto versionable o_i con el objeto o_j. Consecuentemente, las asociaciones existentes entre dos *versiones de objetos* se deben inferir a partir de las asociaciones entre *objetos versionables* (expresión 6).

$$(\forall\ v, v_1, v_2, a, m)\ asociaciónInf(a, v_1, v_2, m) \Leftrightarrow$$
$$(\exists\ o_1, o_2)\ pertenece(v_1, m) \wedge pertenece(v_2, m) \quad (\mathbf{6})$$
$$\wedge\ versión(v_1, o_1) \wedge versión(v_2, o_2) \wedge asociación(a, o_1, o_2)$$

4. Modelo Orientado a Objetos de Versiones de Modelos del Proceso de Diseño

A partir del esquema de Representación de Versiones de Modelos propuesto se trabajó en un modelo orientado a objetos que lo soporte y permita su especialización en distintos dominios de diseño ingenieriles. En la Figura 2 se presentan los distintos paquetes que conforman el modelo del esquema propuesto. El paquete *ModeloDominio*, permite expresar los conceptos modelables, sus características y las relaciones existentes entre los distintos conceptos de un domino en particular; dominio que se empleará en el desarrollo de un proyecto dado.

El esquema propuesto utiliza el cálculo situacional para representar si una versión de objeto pertenece o no a una versión de modelo dada y permitir la reconstrucción de una versión de modelo particular. En este esquema se utiliza la tecnología de orientación a objetos para modelar las relaciones existentes entre las versiones de objetos de distintas versiones de modelos, permitiendo la navegación a través de la historia de las versiones de objetos que conforman una versión de modelo dada.

Las relaciones entre versiones de objetos se representan a través de asociaciones de clase explícitas en el paquete *Nivel Versiones* (Figura 2, especialización de *Historia*). Cada operación de

transformación que se aplica a una versión de un modelo, incorpora información necesaria para mantener rastros de la evolución del mismo. Esencialmente, cada operación, además de ejecutar la acción que la misma involucra, establece una relación entre las versiones de objetos a las que se aplica y las que surgen como resultado de su ejecución (operaciones implementadas en los métodos *ejecutar*, especializados por las distintas operaciones en el paquete *ModeloOperación*).

Una versión de modelo en particular (clase *VersiónModeloTrabajo* del paquete *ModeloInferido*) está conformada por las versiones de objetos y las asociaciones entre estas versiones, que pertenecen a la misma. Las asociaciones entre las *versiones de objetos* se determinan a partir de las asociaciones explícitas en el repositorio entre los objetos que éstas versionan (expresión 6), además, a partir de los objetos versionables del repositorio se determina los tipos de conceptos modelables que representan (*tipoConcepto*). Las características de los distintos objetos inferidos se determinan a partir del paquete *ModeloDominio*, donde se definen las características para cada concepto modelable. Los valores asumidos por las características para una versión de objeto dada se obtienen a partir de la información almacenada en el *Nivel Versiones*.

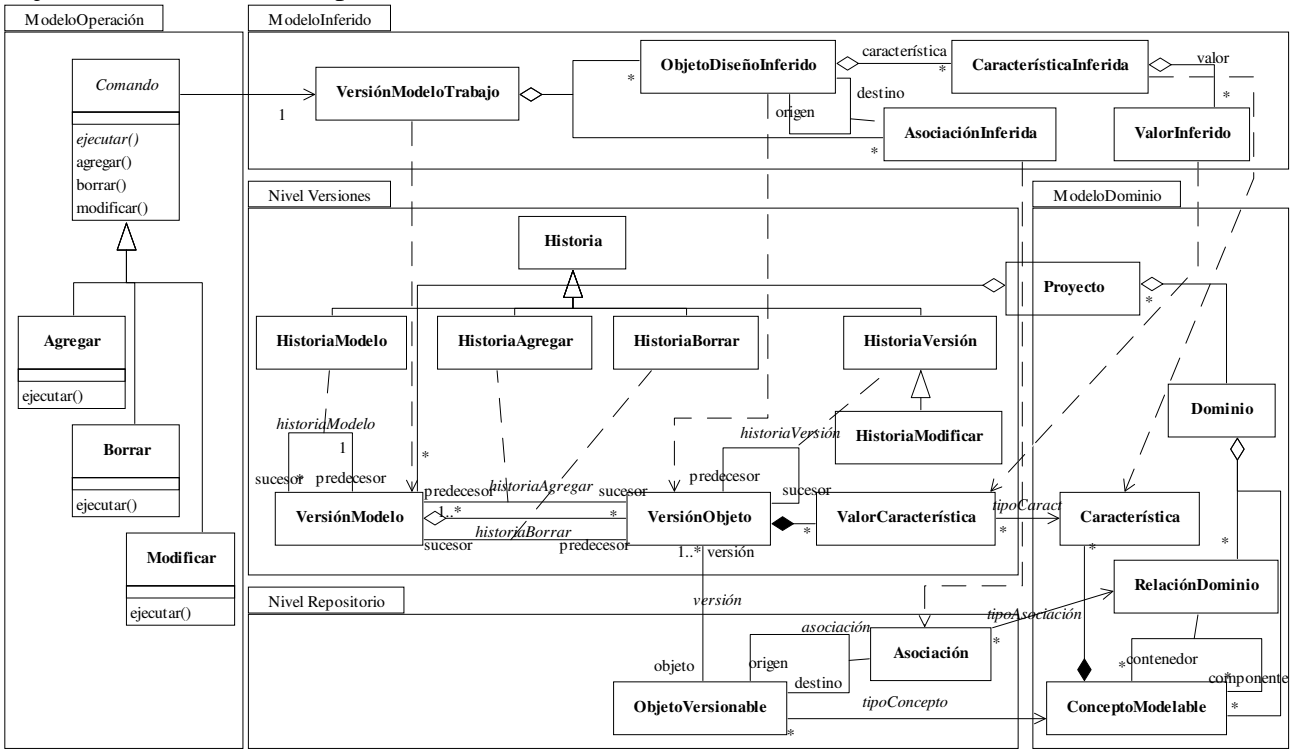

Figura 2. Modelo de Objeto del Esquema de Versionamiento

5. Extensión de las Operaciones Básicas

Existen otras operaciones frecuentes en la generación de modelos que son necesarias representar si se quiere mantener la historia de los cambios realizados en la versión de modelo. Por ejemplo, una operación frecuente en el modelado es la que permite detallar un objeto a partir de su descomposición en una estructura de nuevos objetos. En este contexto tal operación es denominada refinar, y la misma permite que en la nueva versión de modelo aparezcan las versiones de objetos que desarrollan el refinamiento, un ejemplo de tal operación se presentó en la Figura 1. La operación inversa a refinar es simplificar, mediante la cual una estructura de objetos (conjunto de versiones de objetos) se condensa en una versión de objeto.

En principio, la operación *refinar*(v, ψ) se expresa en términos de una serie de operaciones agregar y borrar de acuerdo a la expresión 7. La posibilidad de expresar operaciones compuestas en término de operaciones básicas, como se propuso en la expresión 7, permite mantener el axioma de estado sucesor presentado en la expresión 4, sin tener que actualizarlo con cada nueva operación que se incorpore. Por otro lado, la expresión 7 tiene la falencia de no expresar la relación existente

entre la versión de objeto v y las versiones de objetos v_r que pertenecen a ψ, *refinamiento*(v, ψ). Dicha relación se modela a través del paradigma orientado a objetos. Una versión de objeto se vincula con una o más versiones de objetos antecesoras y una o más versiones de objetos sucesoras. Este vínculo se lo denomina *Historia Versión* (Figura 2 – *NivelVersiones*) y es la asociación a especializar para definir las distintas operaciones. En el caso de la operación refinar, *Historia Versión* se especializa en *Historia Refinar*, donde su antecesor es una versión de objeto y su sucesor es una o más versiones de objetos. La generación de las instancias de *Historia Refinar* es responsabilidad del método *ejecutar()* redefinido en la clase *Refinar*, que especializa *Comando*. Tal método incorpora además las operaciones sobre las versiones de objeto según la expresión 7.

$$\phi = \phi_1 \bullet refinar(v, \psi) \bullet \phi_2 \Rightarrow$$
$$(\forall\ v_r \in \psi,\ agregar(v_r) \in \phi_3 \wedge borrar(v) \in \phi_3) \wedge (\phi = \phi_1 \bullet \phi_3 \bullet \phi_2) \qquad (7)$$

6. Conclusiones

Se ha observado la necesidad en distintas disciplinas ingenieriles de contar con un esquema que permita manejar consistentemente y navegar a través de los modelos generados en el proceso de diseño, para lo cual es necesario especializar el modelo propuesto según el dominio particular con el cual se quiera trabajar. Se pretende a través de esta propuesta documentar la evolución de un modelo resultante de un proceso de diseño, de forma tal de poder rastrear la evolución de los distintos objetos identificados durante el proceso, formando las bases para el aprendizaje y el futuro reuso.

El empleo en conjunción del paradigma de objetos con el cálculo situacional permite la especialización del modelo según los conceptos modelables que conforman a un dominio en particular y el conjunto de operaciones aplicables a una versión de modelo dada. Actualmente se está trabajando en la extensión del modelo para dar soporte al trabajo cooperativo, principalmente en la selección de alternativas desarrolladas por distintos actores y en el trabajo concurrente independiente o débilmente acoplado. Por otro lado, se está desarrollando un prototipo basado en tecnología de base de datos orientada a objetos, para la implementación del mismo se está empleando la base de datos Versant [7] y el lenguaje de programación Java.

Bibliografía

1. Booch, G., Rumbaugh J., Jacobson I., "The unified modelling language user guide", Addison-Wesley (1999).
2. Gonnet, S., H. Leone, "A Framework for Model Version Management in a Design Process", Proceedings of the Thirteenth International Conference on Software Engineering and Knowledge Engineering (SEKE), Knowledge Systems Institute, 260-267 (2001).
3. Gonnet, S., H. Leone, G. Henning, "Representing and Capturing the Experts´ Knowledge in a Design Process", Anales JAIIO – Volumen 31, Subserie ASAI 2002: Argentine Symposium on Artificial Intelligence, 185-196 (2002).
4. McCarthy, J., "Situations, actions, and causal laws", Memo 2, Stanford University Artificial Intelligence Project, Stanford, California (1963).
5. Reiter, R., "Knowledge in Action: Logical Foundation for Describing and Implementing Dynamical Systems", The MIT Press (2001)
6. Scherl, R., H. Levesque, "Knowledge, action, and the frame problem", Artificial Intelligence, 144, 1-39 (2003).
7. Versant Corporation, "Versant Database Administration Manual 5.2" (1998).

Agradecimientos

Este trabajo es financiado en forma conjunta por el Consejo Nacional de Investigaciones Científicas y Técnicas de la República Argentina, la Universidad Tecnológica Nacional y la Universidad Nacional del Litoral. Se agradece el apoyo brindado por estas instituciones.