Semántica formal de asociaciones y agregados para su implementación con reglas activas

Octavio Martín, Jesús Torres, Amador Durán,
José A. Pérez y Miguel Toro

Dpto. de Lenguajes y Sistemas Informáticos
Facultad de Informática y Estadística, Universidad de Sevilla
41.012 Sevilla, España (Spain)
{octavio,jtorres,amador,jperez,mtoro}@lsi.us.es

1 Interrelaciones entre objetos

Las metodologías orientadas a objetos consideran al modelo de objetos como la estructura estática de un sistema, mostrando habitualmente algunas clases de objetos, sus propiedades (atributos y métodos), y sus interrelaciones. Éstas últimas podemos clasificarlas en varios tipos importantes: las **asociaciones**, que definen un conjunto de enlaces entre objetos; las **agregaciones**, una clase especial de asociación, que definen una interrelación todo–parte; y las **generalizaciones**, que definen una interrelación superclase–subclase.

Podemos añadir a estas interrelaciones algún tipo de semántica, entre otras: la **multiplicidad** (restricción sobre el número de objetos participantes en cada papel) y la **semántica de eliminación** (en *cascada* si al borrar un objeto son eliminados aquellos que lo referencian, *restringida* si no se puede borrar un objeto referenciado mediante algún enlace de este tipo, y otros). La agregación añade la **dependencia existencial** (cuando el agregado se elimina, se eliminan sus componentes) y la **exclusividad** (un objeto sólo pertenece a un agregado). En [5, 4] podemos encontrar algunos ejemplos de definición formal de asociaciones mediante restricciones. [2] presenta una semántica para interrelaciones binarias para ODMG–93.

Veamos un ejemplo: *Sea un modelo donde tenemos objetos representando edificios, oficinas y aparcamientos. Un edificio tiene un conjunto de oficinas de manera obligatoria, y varios aparcamientos de manera opcional. Pero cada oficina pertenece a un único edificio, mientras que los aparcamientos podrían no pertenecer a ningún edificio. Si un edificio fuera demolido, todas sus oficinas deberían ser destruidas también, pero no así los aparcamientos a su alrededor.* Tendríamos que definir una agregación $1, 1..n$ entre las clases *edificio* y *oficina* (con una dependencia existencial exclusiva), y una agregación $0..n, 0..n$ entre las clases *edificio* y *aparcamiento* (sin más restricciones).

2 Implementación con bases de datos activas

Normalmente, las asociaciones binarias entre objetos se han implementado en las bases de datos mediante la integridad referencial. Sin embargo, no es suficiente

JISBD'99
I.S.B.N.: 84-699-0956-08

cuando la complejidad se incrementa: si se crea un objeto *edificio*, deben crearse también todos los objetos *oficina*; si no es así, no debería permitirse la creación de tal objeto. Este comportamiento debe ser realizado por una aplicación externa. Podremos mejorar el rendimiento si diseñamos un conjunto de reglas que automáticamente realice tal comportamiento dentro de una base de datos activa: en una transacción, se crean los objetos *edificio* y *oficina*, y se establecen los enlaces; al final de la transacción, una regla se activará por la creación de un objeto de la clase *edificio*, y comprobará que todas las restricciones se satisfacen, y si no fuera así, la transacción se deshace. Otras operaciones, como la eliminación de objetos, tienen un comportamiento más complicado.

Las bases de datos activas [3,6] constituyen un nuevo área de desarrollo que incrementa la funcionalidad de las bases de datos tradicionales con reglas activas (*triggers*). Normalmente, una regla activa se compone por un evento, una condición y una acción, así que cada acción será automáticamente ejecutada en respuesta a la ocurrencia de su evento si su condición se cumple. Por lo tanto, el diseño de reglas activas nos permite definir procedimentalmente acciones que reparen las rupturas de la integridad, aparte de proveer un mecanismo uniforme y eficiente para desarrollar otras tareas sobre una base de datos, disminuyendo el volumen de las comunicaciones entre la base de datos activa y las aplicaciones, aunque perdiendo la ventaja de la naturaleza declarativa de la especificación de restricciones.

Actualmente, hemos resuelto algunos problemas que se derivan de la falta de potencia de las actuales bases de datos activas, y trabajamos sobre el comportamiento colectivo de las reglas, fuente de errores debido a las interacciones e influencias mutuas entre ellas [1].

Referencias bibliográficas

1. A. Aiken, J. Widom, and J.M. Hellerstein. Behaviour of database production rules: Termination, confluence, and observable determinism. In M. Stonebraker, editor, *Proceedings ACM SIGMOD Int. Conf. on Management of Data*, pages 59–68, San Diego, CA, May 1.992.
2. E. Bertino and G. Guerrini. Extending the ODMG object model with composite objects. In *Proceedings of the ACM OOPSLA '98 Conference on Object-Oriented Programming: Systems, Languages, and Applications*, pages 259–270, Vancouver, Canada, October 1.998.
3. S. Ceri and P. Fraternali. *Designing Database Applications with Object Rules.* Addison–Wesley, 1.997.
4. D. Costal, A. Olivé, and M. Ribera. Temporal features of class populations and attributes in conceptual models. In *Proceedings of ER '97, 16th International Conference on Conceptual Modeling*, pages 57–70, Los Angeles, CA, November 1.997.
5. J. Torres, J.A. Troyano, and M. Toro. Definiendo asociaciones entre clases de objetos mediante restricciones. In *Jornadas Iberoamericanas de Ingeniería de Requisitos y Ambientes Software*, pages 169–180, Torres (Brasil), 1.998.
6. J. Widom and S. Ceri. *Active Database Systems: Triggers and Rules for Advanced Dababase Processing.* Morgan Kaufmann Publishers Inc., San Francisco, California, 1.996.