

Fundamentos de las Bases de Datos Orientadas a Aspectos

Ing. Leonardo M. Moreno Villalba*

Abstract

The improved modularization and separation of concerns offered by AOP techniques opens up to develop databases and applications that are customizable, extensible, evolvable, and maintainable in the face of changing requirements. This improved can be afforded at both the vendor end to provide pre-fabricated customizations of fundamentals components in the form of a product line of database, or at the customer end where developer may customize specific features to meet the organizational work practices.

Introducción

Las técnicas convencionales de desarrollo y los modelos de bases de datos, son actualmente insuficientes para abordar la personalización, extensibilidad, evolutibilidad y mantenimiento de las bases de datos y sus aplicaciones. Hoy día, son construidas y distribuidas como una simple solución que se acomoda a todos los escenarios y que nosotros debemos aceptar como la única forma de manejar nuestros problemas de datos. Esta falta de capacidad para llevar a cabo los procesos antes mencionados, resulta en una gran inversión humana y financiera para poder adaptar las aplicaciones y las bases de datos a los cambiantes requerimientos de las empresas.



Acerca del autor...

* Profesor de la Licenciatura en Informática, Tecnológico de Estudios Superiores de Ecatepec

1. Comparación

El término bases de datos orientadas a aspectos, se refiere a aquellas que permiten direccionar una o más de las tres facetas llave de una relación entre una base de datos y la programación orientada a aspectos (Figura 1).



Fig.1

Como se muestra en la Figura 1, las bases de datos orientadas a aspectos tienen las siguientes características:

- ◆ Utilizan la orientación a aspectos para su implementación y operación.
- ◆ Soportan las técnicas de orientación a aspectos para proveer el almacenamiento y manipulación de los aspectos.
- ◆ Utilizan las técnicas de orientación a aspectos para separar el acceso a la base de datos y la interacción de las aplicaciones.

Idealmente, este tipo de base de datos debería cubrir cada una de esas tres características. La Tabla 1 muestra una comparativa general de cómo una base de datos orientada a aspectos se puede distinguir de una convencional.

Base Convencional	BDOA
Utilizan técnicas de implementación convencionales (O.O.)	Emplean la orientación a aspectos para complementar las técnicas convencionales y arquitecturas.
Tienen una limitada capacidad de personalización, extensibilidad, evolutibilidad y mantenimiento.	Tiene un alto grado de personalización, extensibilidad, evolutibilidad y mantenimiento.
Los modelos de datos están muy por debajo de la composición y operación día a día.	El modelo orientado a aspectos complementa el modelo de datos empleado por los sistemas para facilitar la implementación y separación de características.
No proveen soporte para la persistencia de aspectos.	Proveen un soporte para el manejo de la persistencia como un aspecto.
Proveen un limitado o parcial soporte para la separación de la persistencia para la lógica de negocios o para aplicaciones.	Permiten ofrecer un marco para el manejo y modularización de la persistencia.



2. Uso de la OA en las bases de datos

Los sistemas de bases de datos evolutivos semiautónomos (SADES), es decir, la siguiente fase de la orientación a aspectos de las bases de datos orientadas a objetos, y los sistemas genéricos de bases de datos orientados a objetos (GOODS), son dos sistemas que emplean parte de la orientación a aspectos en su implementación y operación. Pero no lo hacen totalmente, por lo que no ofrecen un alto grado de personalización de componentes.

3. Modelo de la POA en bases de datos

En la Figura 2, los aspectos pueden ser usados para separar las características personalizables de *crusscutting* (reglas de entrelazado) de componentes de la base de datos que no son personalizables y tienen poca funcionalidad. El término componente es usado en el sentido más genérico y por lo tanto incorpora bases de datos monolíticas.

La base de datos o los componentes de un sistema manejador de bases, así como los granos gruesos y granos finos de los componentes de éste, también incorporan entidades (en bases de datos orientadas a objetos: objetos, metaobjetos, metaclasses; y en bases de datos relacionales: instancias, relaciones y metadatos) que son almacenadas y residen en la propia base de datos.

Los cambios a estas características encapsulados por aspectos localizados, permiten incrementar en gran medida el grado de personalización. Tanto los cambios menores como mayores son propagados durante el *weaving* (tejido) o *reweaving* (en tiempo de ejecución o tiempo de compilación). También es posible separar las características propias de los granos gruesos y propagarlas en los granos finos como un aspecto que puede servir a uno o más componentes.

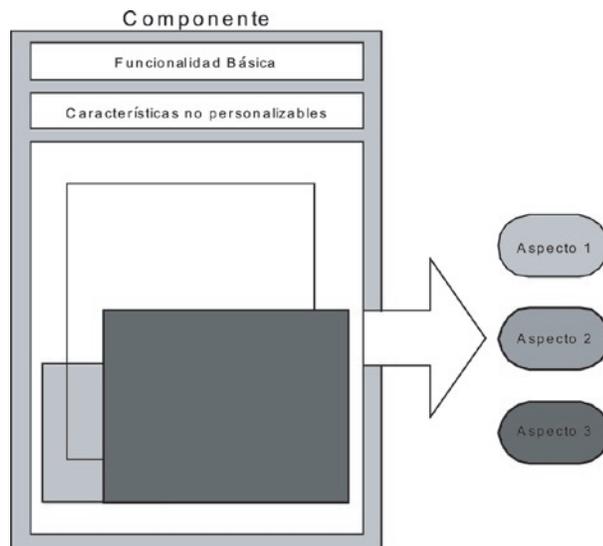


Fig. 2

Como se puede observar en la Figura 3, un acercamiento a la orientación a aspectos, puede hacer posible la personalización tanto en tiempo de ejecución como en tiempo de compilación. Por ejemplo un lenguaje de aspectos es asociado a un *weaver* (tejedor de aspectos), soportando tanto aspectos dinámicos como estáticos.

Como se muestra en la Figura 3, los aspectos 1, 2, y 3, pueden ser personalizados o permutados por cambios localizados antes del tiempo de ejecución, el *weaving* se encargará de tejer esos aspectos en cada uno de los demás componentes. Las características encapsuladas en aspectos que necesitan ser personalizados en tiempo de compilación, sólo son mezcladas con el resto de los componentes durante esta fase y no existen en tiempo de ejecución. Ello reduce la saturación del manejo de un gran número de aspectos, así como su *weaving/reweaving* en tiempo de ejecución.

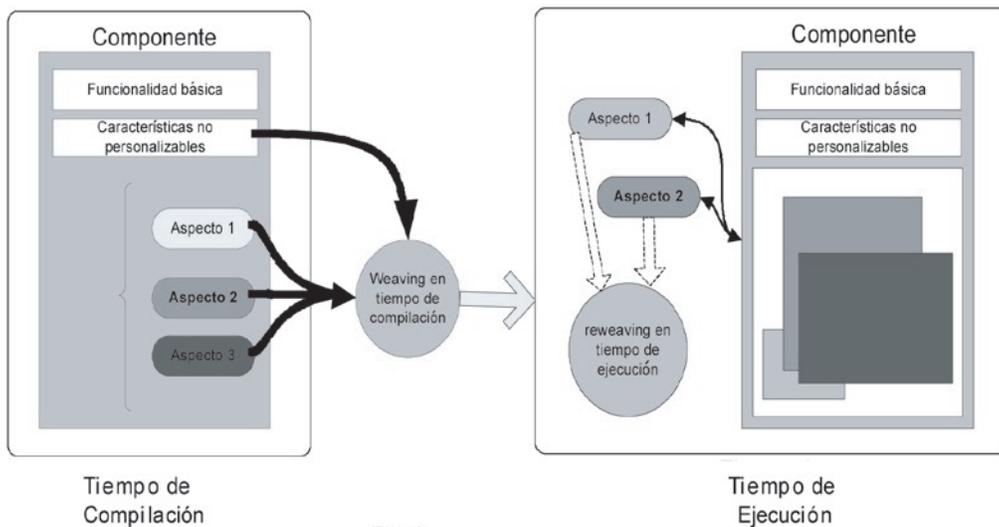


Fig. 3

Conclusión

Este artículo nos ha proporcionado una breve introducción de algunos sistemas de bases de datos, sus conceptos y relevancias, así como una breve comparación entre los sistemas convencionales y los orientados a aspectos. El término “base de datos orientado a aspectos”, abarca ciertas características que lo hacen único y distinguible de los demás, proporcionándole personalización, extensibilidad, evolutibilidad, y mantenimiento, lo cual se detalla en el punto 3, donde es posible apreciar las características encapsuladas en aspectos pueden tener un *weaving* tanto en tiempo de ejecución como de compilación, o ambos.

Referencias...

Gianpaolo Cugola, Carlo Ghezzi, Mattia Monga, “*Language Support for Evolvable Software: An Initial Assessment of Aspect-Oriented Programming*”, in Proceedings of the International Workshop on the Principles of Software Evolution, IWPSE99, julio 1999.

K. Mehner, A. Wagner, “*An Assessment of Aspect Language Design*”, Position Paper in Young Researchers Workshop, GCSE '99, 1999.

Ken Anderson, “*Thoughts on Aspect-Oriented Programming*”, summarizing Cristina Lopes' work, Northeastern University, 1996.

Claire Tristram, “*Untangling Code*”, in the January/February 2001 issue of the *Technology Review*.

Junichi Suzuki, Yoshikazu Yamamoto, “*Extending UML with Aspects: Aspect Support in the Design Phase*”, 3er Aspect-Oriented Programming (AOP) Workshop at ECOOP '99.

Pascal Fradet, Mario Südholt, “*An Aspect Language for robust programming*”, in Proceedings of the European Conference on Object-Oriented Programming (ECOOP) Workshops 1999.

Gianpaolo Cugola, Carlo Ghezzi, attia Monga, “*Coding Different Design Paradigms for Distributed Applications with Aspect-Oriented Programming*”, in the Workshop su Sistemi Distribuiti: Algoritmi, Architetture e Linguaggi (WSDAAL), September 1999.

Gregor Kickzales, John Lamping, Anurag Mendhekar, Chris Maeda, Cristina Videira Lopes, Jean-Marc Loingtier, John Irwin, “*Aspect-Oriented Programming*”, in Proceedings of the European Conference on Object-Oriented Programming (ECOOP), Finland. Springer-Verlag.

LNCS 1241. Junio 1997. Gregor Kickzales, Erik Hilsdale, Jim Hugunin, Mik Kersten, Jeffrey Palm, William G. Grisnold, “*Getting Started with AspectJ*”, in Communications of the ACM (CACM), Vol. 44, Nº 10, October 2001.