MAPEO OBJETO RELACIONAL

Andia Zeballos, Alonso Andre (2016054945)

Escuela Profesional de Ingeniería de Sistemas

Universidad Privada de Tacna

Tacna, Perú

Abstract

Object Relational Mapping (also known as ORM or OR mapping) is a process that involves the transformation between object and relational models and between the systems that support these methodologies. In order to carry out this transformation it is necessary to have a broad knowledge of object-oriented and relational technologies, as well as their similarities and differences.

1. Resumen

El mapeo objeto/relacional (también conocido como ORM o OR mapping) es un proceso que consiste en la transformación entre modelos de objetos y relacional y entre los sistemas que soportan estas metodologías. Para poder realizar esta transformación es necesario tener un amplio conocimiento de las tecnologías orientada a objetos y relacional, así como también de sus similitudes y diferencias.

2. Introducción

Varias áreas de aplicaciones para los sistemas de bases de datos se hallan limitadas por las restricciones del modelo de datos relacional. El modelo relacional orientado a objetos combina caracteristicas del modelo relacional y del modelo orientado a objetos. Este modelo proporciona un sistema de tipos de datos variado. Aplica la herencia a las relaciones, no sólo a los tipos. El modelo de datos relacional orientado a objetos permite una migración fácil desde las bases de datos relacionales. El término base de datos orientada a

objetos se utiliza para describir los sistemas de bases de datos que soportan el acceso directo a los datos desde lenguajes de programación orientados a objetos, sin necesidad de lenguajes de consultas relacionales como interfaz de las bases de datos.

3. Marco Teórico

3.1. ***Concepto del ORM***

Bauer y King señalan que el mapeo Objeto/Relacional es "la persistencia automatizada y transparente de las tablas en una Base de Datos relacional, usando metadatos que definen el mapeo entre los objetos y la Base de Datos" [1]

Un ORM es un modelo de programación que permite mapear las estructuras de una base de datos relacional (SQL Server, Oracle, MySQL, etc.), en adelante RDBMS (Relational Database Management System), sobre una estructura lógica de entidades con el objeto de simplificar y acelerar el desarrollo de nuestras aplicaciones.[7]

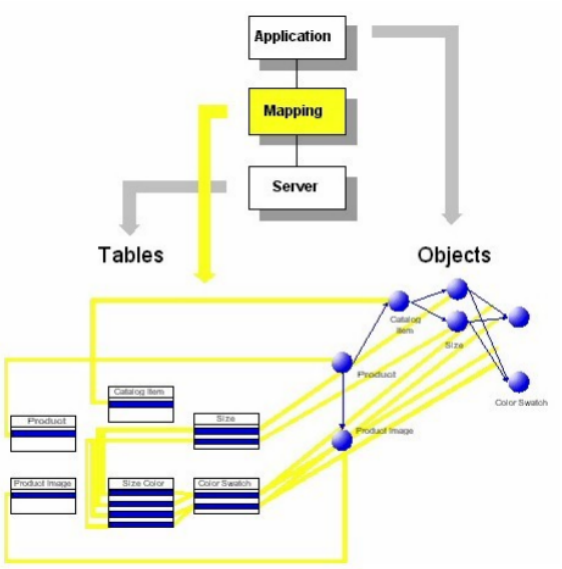

Figura 1: Mapeo Objeto/Relacional

3.2. ***Ventajas y Desventajas del ORM***

3.2.1. ***Ventajas***

Algunas de las ventajas del ORM son: [2] [9]

- Rapidez de desarrollo
- Abstracción de la base de datos
- Lenguaje propio para consultas a la base de datos
- Encapsulación
- Portabilidad
- Reutilización
- Seguridad
- Mantenimiento del código

3.2.2. ***Desventajas***

Algunas de las desventajas del ORM son: [2] [9]

- Tiempo utilizado en el aprendizaje
- Aplicaciones algo más lentas

3.3. ***ORMs en el mercado actual***

- **Microsoft Entity Framework 6.0:**
  Microsoft Entity Framework se publicó por primera vez en 2008, como parte de .NET Framework 3.5 SP1 y Visual Studio 2008 SP1. A partir de la versión 4.1, se ha distribuido como paquete NuGet independiente convirtiéndose en uno de los más populares en NuGet.org. Sólo para plataforma Windows. Muy estable. Open Source. Proporciona servicios avanzados de modelado y seguimiento de estados de entidades, persistencia automática de cambios, caching, gestión de transacciones, etc.

- **Microsoft Entity Framework Core 2.0:**
  Nueva implementación de Entity Framework en versión Core. Más ligera, extensible y multiplataforma (Windows, Linux, Mac). Ofrece un rendimiento más potente en comparación con la versión desarrollada específicamente para .Net Framework. En continua evolución. Open Source. Puede ejecutarse sobre .Net Framework 4.6.1, .Net Core 2.0 o posteriores.

- **Microsoft Entity Framework Core 2.1:**
  Versión más reciente de Entity Framework Core lanzada a finales de mayo de 2018. Incluye mejoras funcionales y de rendimiento, así como correcciones sobre la versión anterior. No obstante, no toda la funcionalidad de Entity Framework 6.0 ha sido migrada todavía a esta última versión de Entity Framework Core.

- **NHibernate:**
  Conversión de Hibernate en Java para lenguaje C# compatible con plataforma .Net. Estable. Open Source.

- **Dapper:**
  Micro-ORM que proporciona métodos de extensión a las clases de .Net Framework para mapear resultados y persistir datos, previa inyección de código SQL personalizado. No nos libera pues de la implementación de nuestro código SQL, actuando como simple mapeador, pero a cambio ofrece un buen rendimiento. Es un ORM creado y usado en producción por el conocido portal web Stack Overflow. [7]

3.4. Modelo de datos basado en objetos

[10] El modelo de datos orientado a objetos se basa en el paradigma de los lenguajes de programación orientados a objetos, que actualmente se usa en gran medida. La herencia, la identidad de los objetos y la encapsulación , con métodos para ofrecer una interfaz para los objetos, están entre los conceptos principales de la programación orientada a objetos que han encontrado aplicación en el modelado de datos.

El modelo orientado a objetos puede considerarse una extensión del modelo E-R con los conceptos de encapsulación, métodos e identidad de los objetos.

3.5. ***Bases de datos basadas en objetos***

El primer obstáculo al que se enfrentan los programadores que usan el modelo relacional de datos es el limitado sistema de tipos soportado por el modelo relacional. El segundo obstáculo es la dificultad de acceso a los datos de la base de datos desde los programas escritos en lenguajes de programación como C++ o Java. La mera extensión del sistema de tipos soportado por las bases de datos no resulta suficiente para resolver completamente este problema. El término lenguajes de programación persistentes hace referencia a las extensiones de los lenguajes de programación existentes que añaden persistencia y otras características de las bases de datos usando el sistema de tipos nativo del lenguaje de programación. El término sistemas de bases de datos orientadas a objetos se usa para hacer referencia a los sistemas de bases de datos que soportan sistemas de tipos orientados a objetos y permiten el acceso directo a los datos desde los lenguajes de programación orientados a objetos usando el sistema de tipos nativo del lenguaje. [10]

3.5.1. ***Lenguaje de programación persistentes***

Los lenguajes de las bases de datos se diferencian de los lenguajes de programación tradicionales en que trabajan directamente con datos que son persistentes; es decir, los datos siguen existiendo una vez que el programa que los creó haya concluido. Las relaciones de las bases de datos y las tuplas de las relaciones son ejemplos de datos persistentes. Los lenguajes de programación persistentes son lenguajes de programación extendidos con estructuras para el tratamiento de los datos persistentes. Los lenguajes de programación persistentes pueden distinguirse de los lenguajes con SQL incorporado, al menos, de dos maneras:

1. En los lenguajes incorporados el sistema de tipos del lenguaje anfitrión suele ser diferente del sistema de tipos del lenguaje para el tratamiento de los datos. Los programadores son responsables de las conversiones de tipos entre el lenguaje anfitrion y SQL. Hacer que los programadores lleven a cabo esta tarea presenta varios inconvenientes :

- El código para la conversión entre objetos y tuplas opera fuera del sistema de tipos orientado a objetos y , por tanto tiene más posibilidades de presentar errores no detectados.

- La conversión en la base de datos entre el formato orientado a objetos y el formato relacional de las tuplas necesita gran cantidad de código.

2. Los programadores que usan lenguajes de consultas incorporados son responsables de la escritura de código explícito para la búsqueda en la memoria de los datos de la base de datos. Si se realizan actualizaciones , los programadores deben escribir explicitamente código para volver a guardar los datos actualizados en la base de datos.[10]

3.5.2. ***Sistemas orientados a objetos y sistemas relacionales orientados a objetos***

Los sistemas relacionales orientados a objetos se dirigen a simplificar la realización de los modelos de datos y de las consultas mediante el uso de tipos de datos complejos. Entre las aplicaciones habituales están el almacenamiento y la consulta de datos complejos, incluidos los datos multimedia. Los lenguajes declarativos como SQL, sin embargo, imponen una reducción significativa del rendimiento a ciertos tipos de aplicaciones que se ejecutan principalmente en la memoria principal y realizan gran número de accesos a la base de datos. Los lenguajes de programación persistentes se dirigen a las aplicaciones de este tipo que tienen necesidad de un rendimiento elevado. Proporcionan acceso a los datos persistentes con poca sobrecarga y eliminan la necesidad de traducir los datos si hay que tratarlos con un lenguaje de programación. Sin embargo, son más susceptibles de deteriorar los datos debido a los errores de programación y no suelen disponer de gran capacidad de consulta. Entre las aplicaciones habituales están las bases de datos de CAD.

Los puntos fuertes de los diversos tipos de sistemas de bases de datos pueden resumirse de la manera siguiente:

- Sistemas relacionales: tipos de datos sencillos, lenguajes de consultas potentes, protección elevada.
- Bases de datos orientadas a objetos basadas en lenguajes de programación persistentes: tipos de datos complejos, integración con los lenguajes de programación, elevado rendimiento.
- Sistemas relacionales orientados a objetos: tipos de datos complejos, lenguajes de consultas potentes, protección elevada.[10]

3.6. ***Herramienta de realización de Mapeos para .Net***

- **NHibernate:** según su website es: "NHibernate es un maduro , de código abierto mapeador objeto-relacional para el marco .NET. Está desarrollado de forma activa y se utiliza en miles de proyectos exitosos."[8]

Hibernate es bastante antiguo y es utilizado por miles de aplicaciones de productos, NHibernate no solo ha heredado la mayoría de las buenas características de su antecesor de Java, sino que también se ha enriquecido con características de .NET, como las consultas LINQ. Las consultas LINQ (consulta integrado al lenguaje) ofrece consultas de datos y sistanxis actualizada, la cual es similar a SQL. NHibernate tiene muchas de estas características que harán que escribir código de interacción de base de datos sea muy fácil. Es un proyecto de código abierto que prospera enteramente en las contribuciones de la comunidad..[3]
Caracterísiticas:
- Comunidad establecida
- Ciclo de desarrollo rápido
- Toneladas de complementos y herramientas de búsqueda de texto completo.
- Fácil para Visual Studio Asigne

- **Entity Framework:**es el puente que une naturalmente ambos lados. Entity Framework ofrece un entorno estable para mapear una base de datos relacional de manera bastante eficiente. Titulary Los marcos de trabajo cargan los datos, tablas y relaciones automáticamente desde la base de datos real, sin necesidad de declarar objetos y variables previamente .[6]

- **DataObjects.NET:** es un marco de mapeo de persistencia y relacional de objetos para Microsoft .NET. Permite a los desarrolladores definir objetos persistentes y lógica de negocios directamente en C#, Visual Basic. Los objetos persistentes pueden ser recuperados por consultas LINQ. Los datos persistentes se pueden almacenar en servidores SQL. A diferencia de muchos otros marcos ORM, el modelo de base de datos se genera y mantiene automáticamente.[5]

4. Análisis

4.1. ***Herramientas ORM***

Los ORM proporcionan una abstraccion de alto nivel en una base de datos relacional que permite a un desarrollador escribir codigo Python en lugar

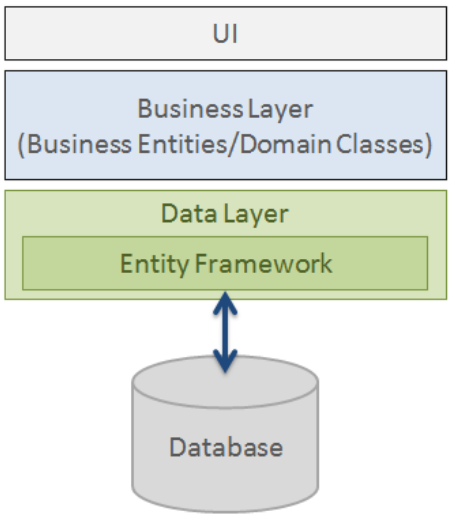

Figura 2: Entity Framework

de SQL para crear, leer, actualizar y eliminar datos y esquemas en su base de datos. Los desarrolladores pueden usar el lenguaje de programacion con el que se sienten comodos para trabajar con una base de datos en lugar de escribir sentencias de SQL o procedimientos almacenados.

4.2. ***Como ejemplo tenemos la herramienta Dapper***

Dapper es un simple mapeador de objetos para .NET y posee el título de King of Micro ORM en términos de velocidad y es virtualmente tan rápido como usar un lector de datos en bruto ADO.NET. Un ORM es un Asignador Relacional de Objetos, que es responsable de la asignación entre la base de datos y el lenguaje de programación. Dapper extiende la IDbConnection proporcionando métodos de extensión útiles para consultar su base de datos. [4]

4.3. ***¿Como funciona?***

- Crear un objeto IDbConnection.
- Escribe una consulta para realizar operaciones de CRUD.
- Pasar la consulta como un parámetro en el método de ejecución.

```
string sqlOrderDetails = "SELECT TOP 5 * FROM OrderDetails;";
string sqlOrderDetail = "SELECT * FROM OrderDetails WHERE OrderDetailID = @OrderDetailID;";
string sqlCustomerInsert = "INSERT INTO Customers (CustomerName) Values (@CustomerName);";

using (var connection = new SqlConnection(FiddleHelper.GetConnectionStringSqlServerW3Schools()))
{
        var orderDetails = connection.Query<OrderDetail>(sqlOrderDetails).ToList();
        var orderDetail = connection.QueryFirstOrDefault<OrderDetail>(sqlOrderDetail, new {OrderDetailID = 1});
        var affectedRows = connection.Execute(sqlCustomerInsert,  new {CustomerName = "Mark"});

        Console.WriteLine(orderDetails.Count);
        Console.WriteLine(affectedRows);

        FiddleHelper.WriteTable(orderDetails);
        FiddleHelper.WriteTable(new List<OrderDetail>() { orderDetail });
}
```

4.4. ***Metodos***

- Ejecutar
- Consulta
- QueryFirst
- QuerySingle
- QueryMultiple

5. Conclusiones

- Conclusion 1 : El mapeo objeto-relacional (ORM) es una técnica para mapear sistemas orientados a objetos a bases de datos relacionales.Utilizar las diferentes herramientas de ORM, da mas rapidez y eficiencia a la construcción de base de datos.

- Conclusion 2 :
  La utilización de mapeadores objeto/relacional brinda muchos beneficios para las aplicaciones que manejan un elevado volumen de datos y tienen una lógica de negocio compleja. Se debe tener en cuenta aspectos referentes a la performance de la aplicación durante las primeras etapas de diseño y en la definición de la arquitectura, dejando de lado los aspectos puros de diseño de objetos y relacional para rescatar lo positivo de cada uno y combinarlo para lograr mejores resultados.

Referencias

[1] Bauer, C. y King, G. (2005). *Hibernate in Action*. Manning Publications.

[2] Borja, A. (2013). Mapeo de objeto relacional (orm). *Universidad Autónoma Latinoamericana*. Accedido el 15-09-2019.

[3] Chatekar, S. (2015). *Learning NHibernate 4*. Packt Publishing.

[4] Dapper (ne). Dapper tutorial. Recuperado de https://dapper-tutorial.net/es/tutorial/1000167/apuesto. Accedido el 16-09-2019.

[5] DataObjects.NET (ne). Dataobjects.net. Recuperado de https://dataobjects.net/. Accessed: 2019-04-07.

[6] Isaj, S. (2015). *Entity Framework*.

[7] Muro, J. (ne). ¿qué es un orm? Recuperado de https://n9.cl/o53g. Accedido el 13-09-2019.

[8] NHibernate (ne). Home. Recuperado de https://nhibernate.info/. Accessed: 2019-04-07.

[9] Polo, L. (2013). Estudio comparativo de sistemas de mapeo objeto relacional desarrolladosen plataformas open source. *Revista Citecsa*. Accedido el 15-09-2019.

[10] Silberschatz, A. y Sudarshan, S. (2006). *Fundamentos de Bases de Datos*. McGraw-Hill.