

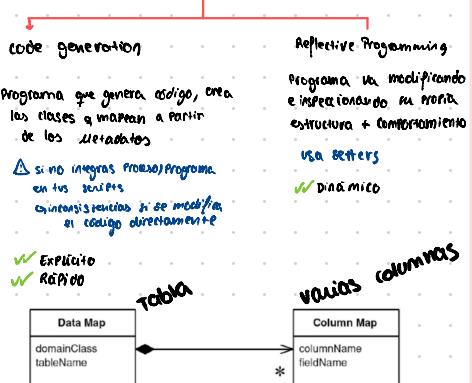


T7 : PATRONES III

Object-Relational Metadata Mapping

Metadata Mapping

Asociar info. de una fuente/sistema a otra
trabajando desde un punto de vista, evitando inconsistencias



Estructura: la clase Data Map (que representa una tabla) está compuesta por varias Column Maps (columnas de atributos).

Relaciones con otros Patrones

- ④ Domain Object
 - ④ Unit of Work
 - ④ Layer Supertype
 - ④ Query Object

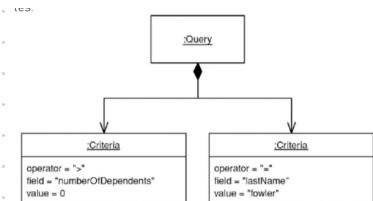
Query Object

Representar una consulta a una base de datos NO incluido en SQL
(No tienes que considerar las PARTICULARIDADES de cada motor BD o BD)

MySQL, PostgreSQL, MongoDB, ... \Rightarrow iDa signal !!

El obj Cliente → genera Obj Query

L₂ define cláusulas/criterios en P(x) de los objetos



```

private Criteria...  
  

    private String sqlOperator;  

    protected String field;  

    protected Object value;  

    protected Interval valueInterval;  

    public static Criteria greaterThan(String fieldName, int value) {  

        return Criteria.greaterThan(fieldName, value);  

    }  

    public static Criteria greaterThan(String fieldName, Object value) {  

        return new Criteria("gt", fieldName, value);  

    }  

    private Criteria(String sql, String field, Object value) {  

        this.sqlOperator = sql;  

        this.field = field;  

        this.value = value;  

    }
}

```

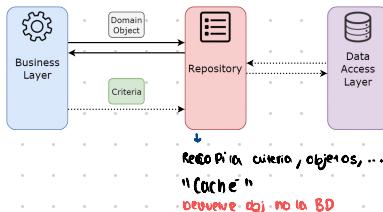
```
QueryObject query = new QueryObject(Person.class);
query.addCriteria(Criteria.greaterThan("numberOfDependents", 0));
```

Patrones Relacionales

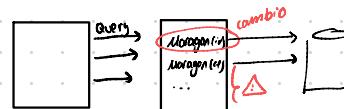
- ④ Interpreter: every object es caso concreto de interpreter
 - ④ Metadato Mapping: necesita relación entre objeto y la BD
 - ④ Identity Map:
 - ④ Domain Model:
 - ④ Data Mapper:

• Repository

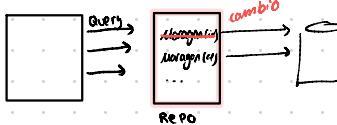
Actua de intermediario o interfaz entre el dominio y el mapeo de datos
v/a) Pruebas con objetos ficticios, evita duplicados e inconsistencias



Si utilizamos memoria



pero si por el respo:



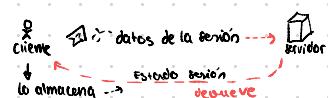
• **Query Object** → Puede hacer consultas directamente sobre el objecto que ya tiene el repository
(si \exists repos → lo hace en them)

T7 : PATRONES III

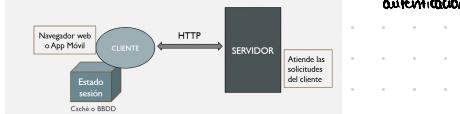
Client Session State

Gestiona el estado de la sesión del cliente

El proxy solicita que el servidor puede identificar de dónde viene la solicitud



Para mantener estado sesión suele utilizar cookies o tokens de autenticación



¿Cuando usarlo?

- ↳ Gestionar autenticación / autorización
- ↳ Control de compras
- ↳ Personalización → config. Preferencias,...
- ↳ Rastrear acciones

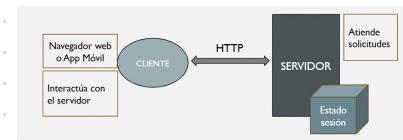
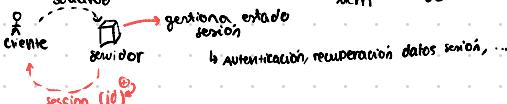
Session State Patterns

Server Session State Pattern

Gestiona estado de la sesión en el servidor

Servidor asigna un identif único ⇒ cada sesión de cliente

+ almacena temporalmente datos
en memoria

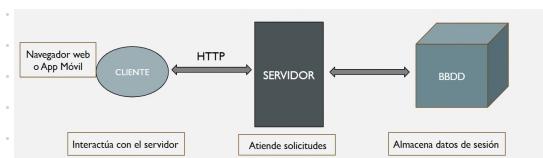


¿Cuando usarlo?

- ① control y info sesión
- ② facilita escalabilidad (Almacena num. usuarios)
- ③ distrib. carga más sencilla
- ④ Asegura datos sesión sean consistentes

Database Session Pattern

Almacenar el estado de la sesión del cliente en una BD



¿Cuando usarlo?

- ↳ Mayor escalabilidad y durabilidad
- ↳ Varios servidores
- ↳ Mayor flexibilidad y control

Característica	Client Session State Pattern	Server Session State Pattern	Database Session State Pattern
Dónde se almacena el estado de la sesión	En el cliente	En el servidor	En una base de datos externa
Control de seguridad	Limitado control de seguridad debido a que los datos de sesión residen en el cliente.	Mayor control de seguridad ya que los datos de sesión residen en el servidor.	Mayor control de seguridad, pero requiere gestión adecuada de la base de datos.
Escalabilidad	Puede requerir menos recursos del servidor, pero puede ser más vulnerable a ataques desde el cliente.	Escalable, pero puede requerir más recursos del servidor al almacenar la información de la sesión.	Escalabilidad mejorada, especialmente cuando se distribuye en múltiples servidores.
Recuperación de sesiones perdidas	Limitada recuperación en caso de pérdida de sesión.	Limitada recuperación en caso de pérdida de sesión.	Mayor capacidad de recuperación.
Persistencia a largo plazo	Generalmente no	Generalmente no	Puede utilizarse para mantener datos de sesión a largo plazo, si es necesario.

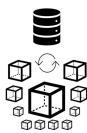
T7 : PATRONES III

Unity of Work

Mantiene una lista de objetos afectados por una TRANSACCIÓN y COORDINA la ejecución de cambios y la resolución de problemas de CONCURRENCIA.

¿Qué Problema resuelve?

- Gestión de transacciones: Facilita la gestión de transacciones agrupando varias operaciones en una única transacción.
- Eficiencia de acceso a datos: Se reduce la necesidad de acceso a la base de datos, mejorando la eficiencia.



Registro Transacc.

- Consistencia de datos: Garantiza la consistencia organizando las operaciones de lectura y escritura.
- Seguimiento de datos: Permite un seguimiento de los datos, registrando y aplicando de manera coherente en la BBDD.



¿Cómo Funciona?

- Atomicidad: Si se hace commit de todo, se hace roll back de todo.
- Consistencia: Estado consistente y no corrupto justo antes del inicio de la transacción e inmediatamente después de su finalización.
- Aislamiento: Los cambios no son visibles a ninguna otra transacción hasta que esta hace commit exitosamente.
- Durabilidad: Después del commit, los cambios son persistentes.



¿Para qué queremos?

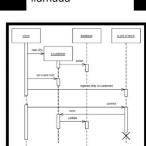
- Cuando tenemos
varias BDs



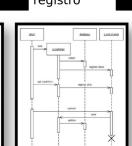
Incluso puede haber obj q no
sean IA BDs

implementación
típica de una sesión

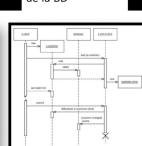
Registro por llamada



Auto-registro



UoW controlador de la BD

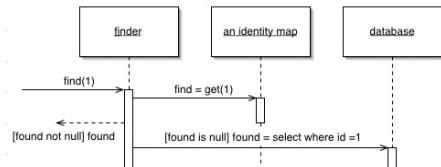


Object - relational Behavioral Patterns

Identity y Map ↛ Repository

Garantiza que cada objeto se cargue solo 1 VEZ

Manteniendo cada obj cargado en un mapa



Elección clave Primaria → Importante

¿Qué Problema resuelve?

- Relacionado con esto hay un claro problema de rendimiento.
- Genera un coste elevado en llamadas remotas.
- Tiempo escribiendo los cambios en la base de datos.
- No ayuda a mantener la integridad de los datos.



¿Cuando hay q usarlo?

No se necesita para objetos inmutables



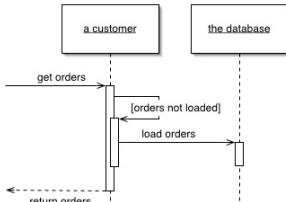
Actúa como un caché para las lecturas de la base de datos.

Ayuda a evitar conflictos de actualización dentro de una sola sesión, pero no hace nada para manejar conflictos que cruzan sesiones.

En general se utiliza un Identity Map para gestionar cualquier objeto traído de una base de datos y...

Lazy Load

Optimiza la carga de aplicaciones del usuario



* Caso Web o lento Fortnite más claro *

¿Qué Problema Resuelve?

- Mejora el rendimiento.
- Optimiza la experiencia de usuario.
- Ahorra ancho de banda.
- Reduce el tiempo de carga.

¿Cuando hay q usarlo?

Gestión de grandes conjuntos de datos

Grandes tiempos de carga iniciales

Optimización para equipos poco potentes

Mejora en eficiencia y en experiencia de usuario

