iSIL Instituto San Ignacio
Aprende haciendo

Desarrollo de Software I

Patrones – Parte I

Ing. Eric Gustavo Coronel Castillo
gcoronelc@gmail.com
gcoronelc.blogspot.com

® Derechos reservados

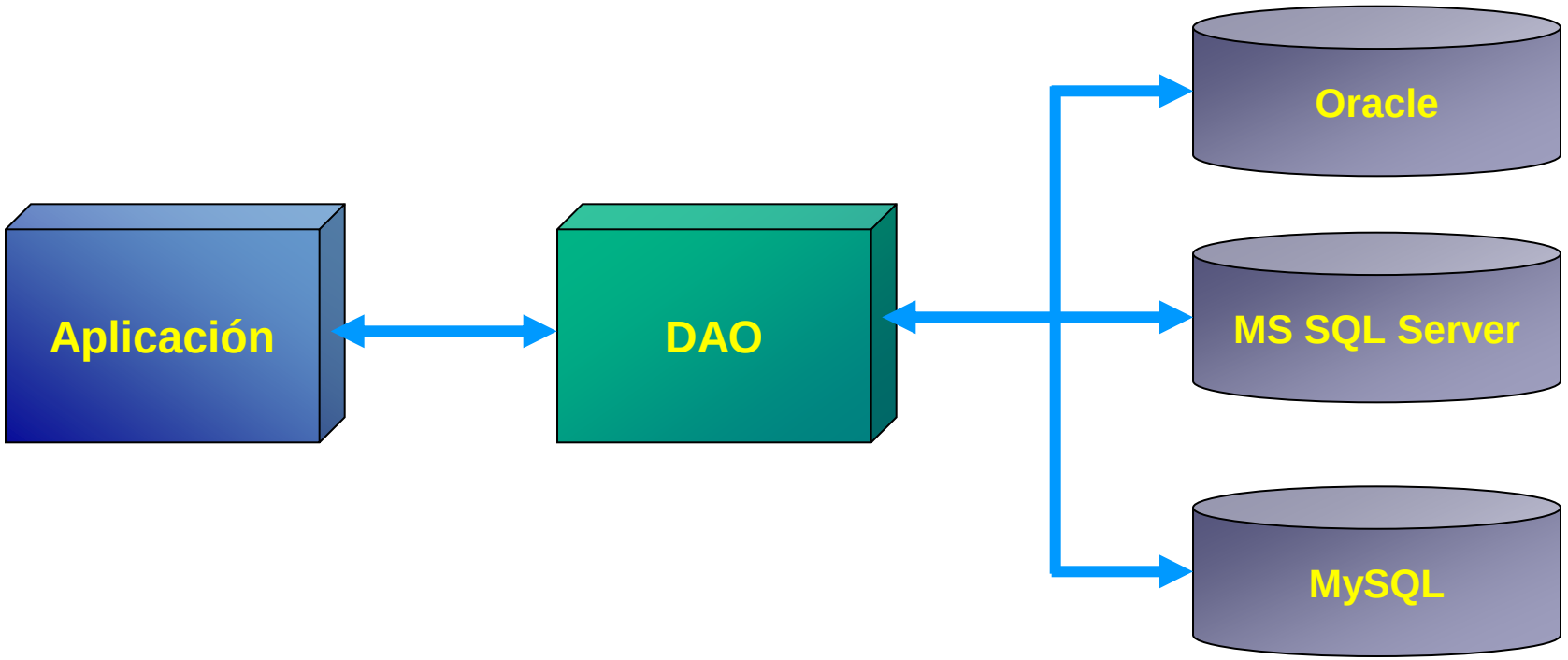
Índice

- Objetivo
- Desarrollo Clásico
- Patrón de Diseño Singleton
- Patrón de Diseño Data Transfer Object
- Patrón de Diseño DAO

® Derechos reservados

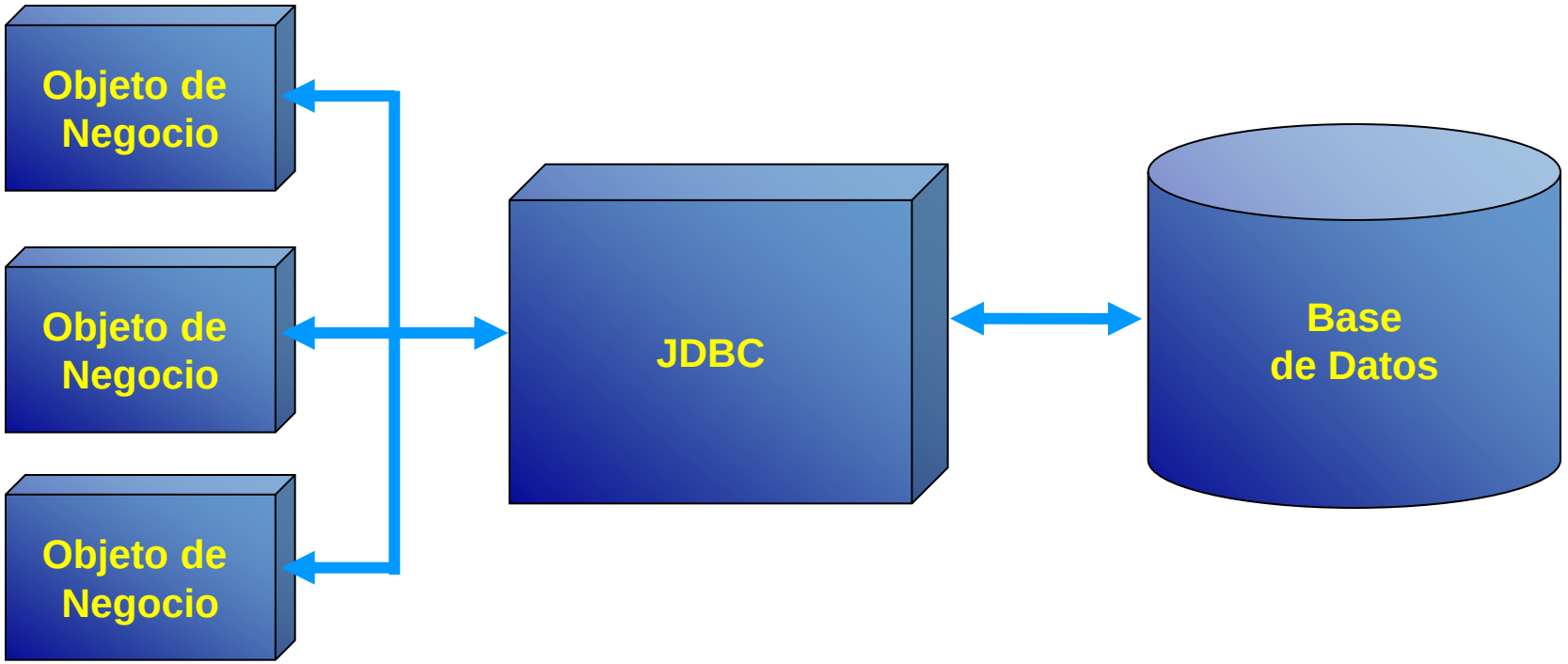
Objetivo

Estandarizar el acceso a fuentes de datos utilizando el patrón de diseño DAO.



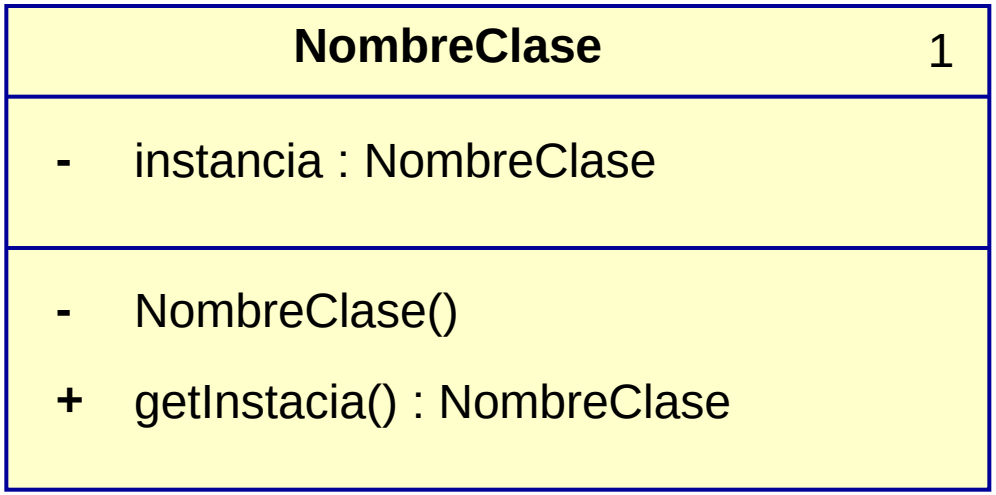
® Derechos reservados

Desarrollo Clásico



® Derechos reservados

Patrón de Diseño Singleton



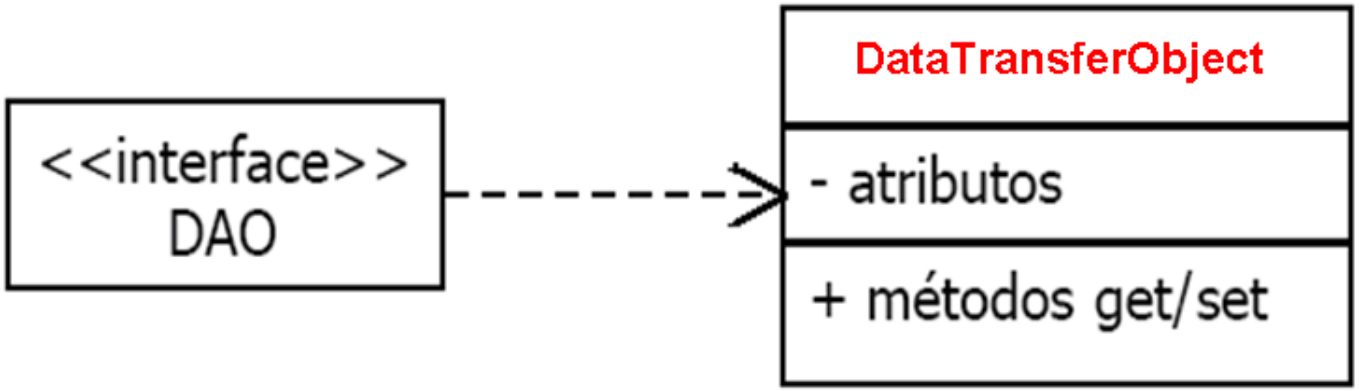
® Derechos reservados

Patrón de Diseño Singleton

```
public class Demo{

        private static Demo instancia = null;

        public static Demo getInstancia(){
                if( instancia == null ){
                        instancia = new Demo();
                }
                return instancia;
        }

        private Demo() {
        }

        . . .
        . . .

}
```


® Derechos reservados

Patrón de Diseño Data Transfer Object



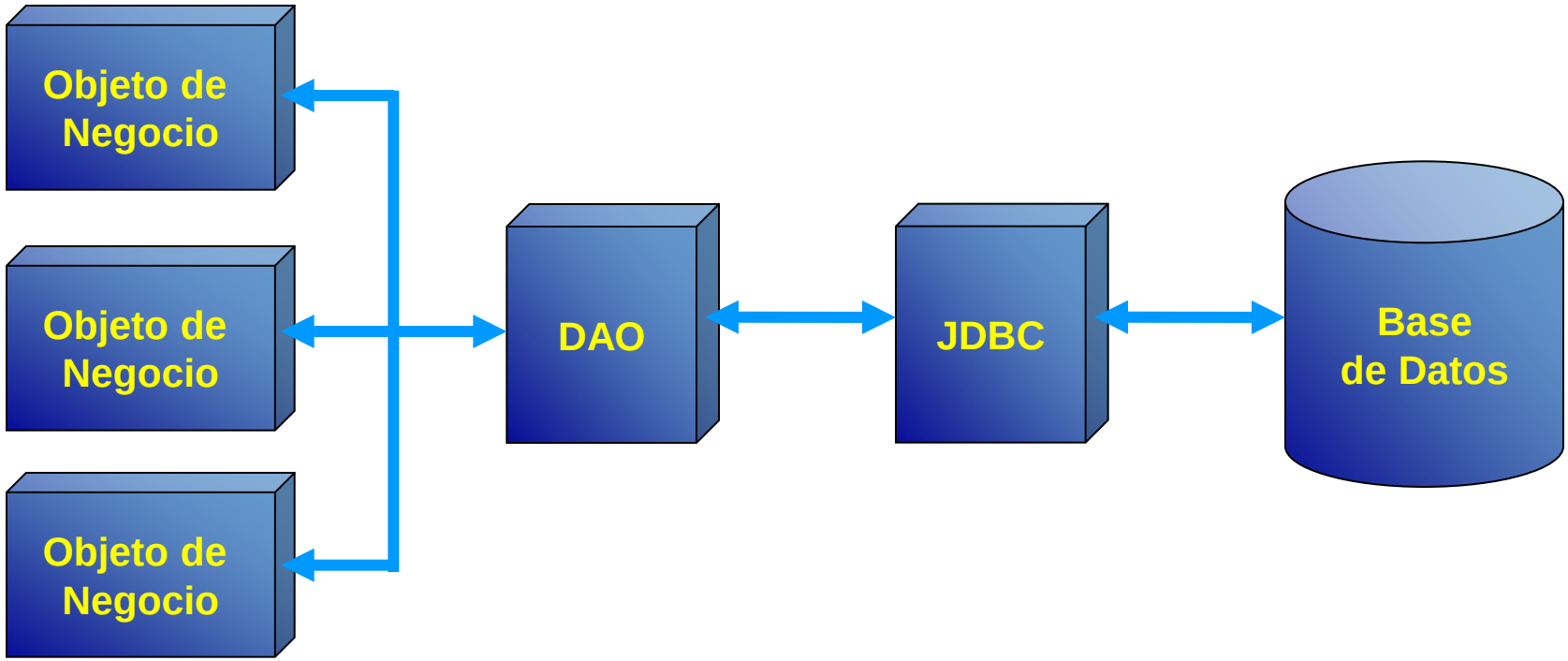
® Derechos reservados

Patrón de Diseño Data Transfer Object

```
public class EmpleadoDTO {

    private Integer id;
    private String nombre;
    private Double sueldo;

    public EmpleadoDTO() { }

    public Integer getId() { return id; }

    public void setId(Integer id) { this.id = id; }

    public String getNombre() { return nombre; }

    public void setNombre(String nombre) { this.nombre = nombre; }

    public Double getSueldo() { return sueldo; }

    public void setSueldo(Double sueldo) { this.sueldo = sueldo; }
}
```

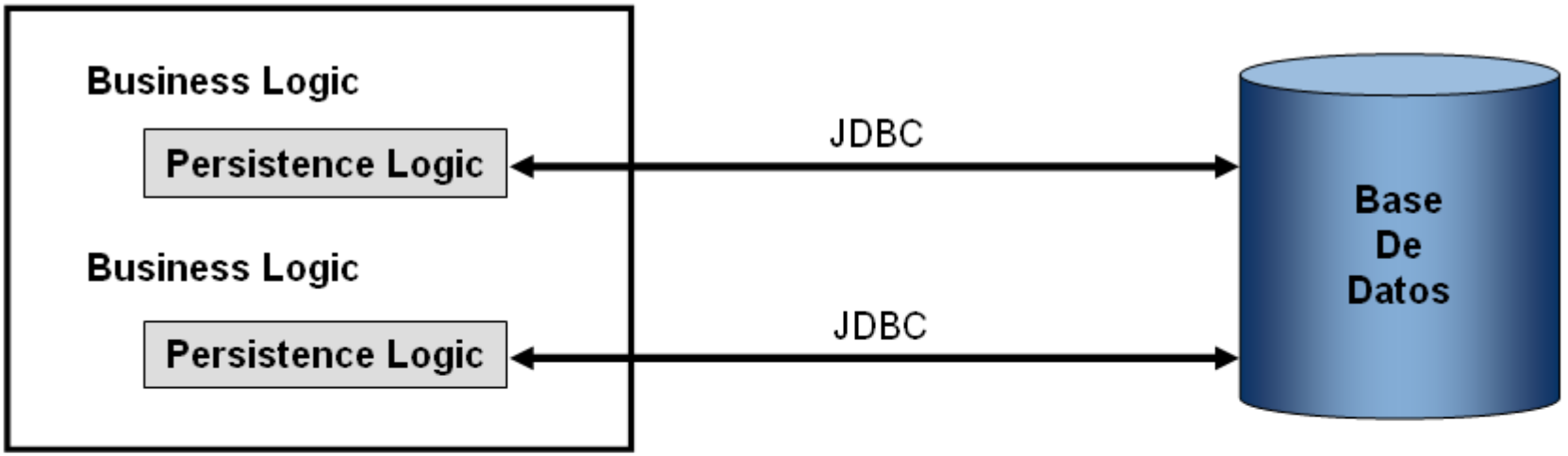

® Derechos reservados

Patrón de Diseño DAO

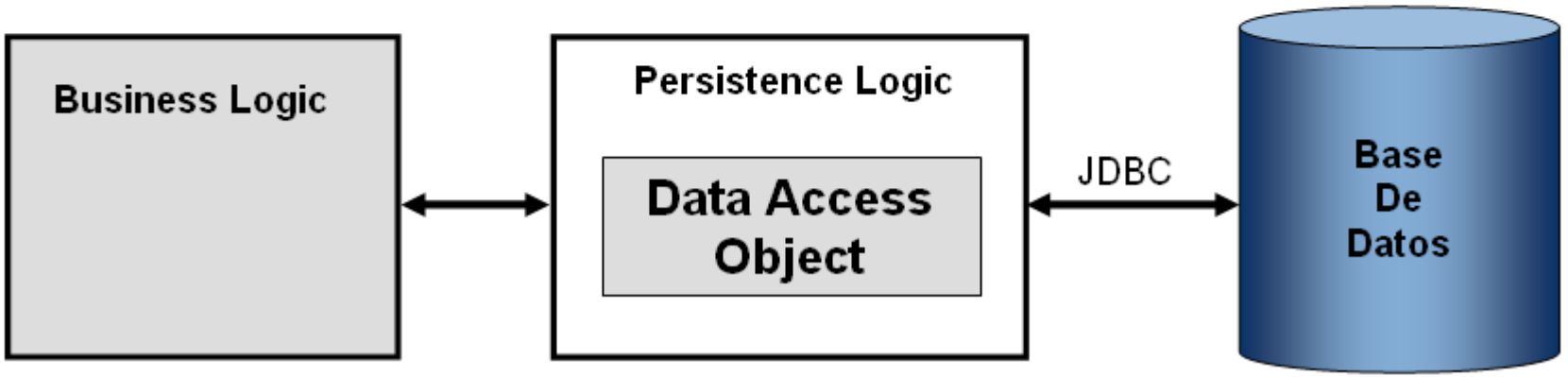


® Derechos reservados

Patrón de Diseño DAO



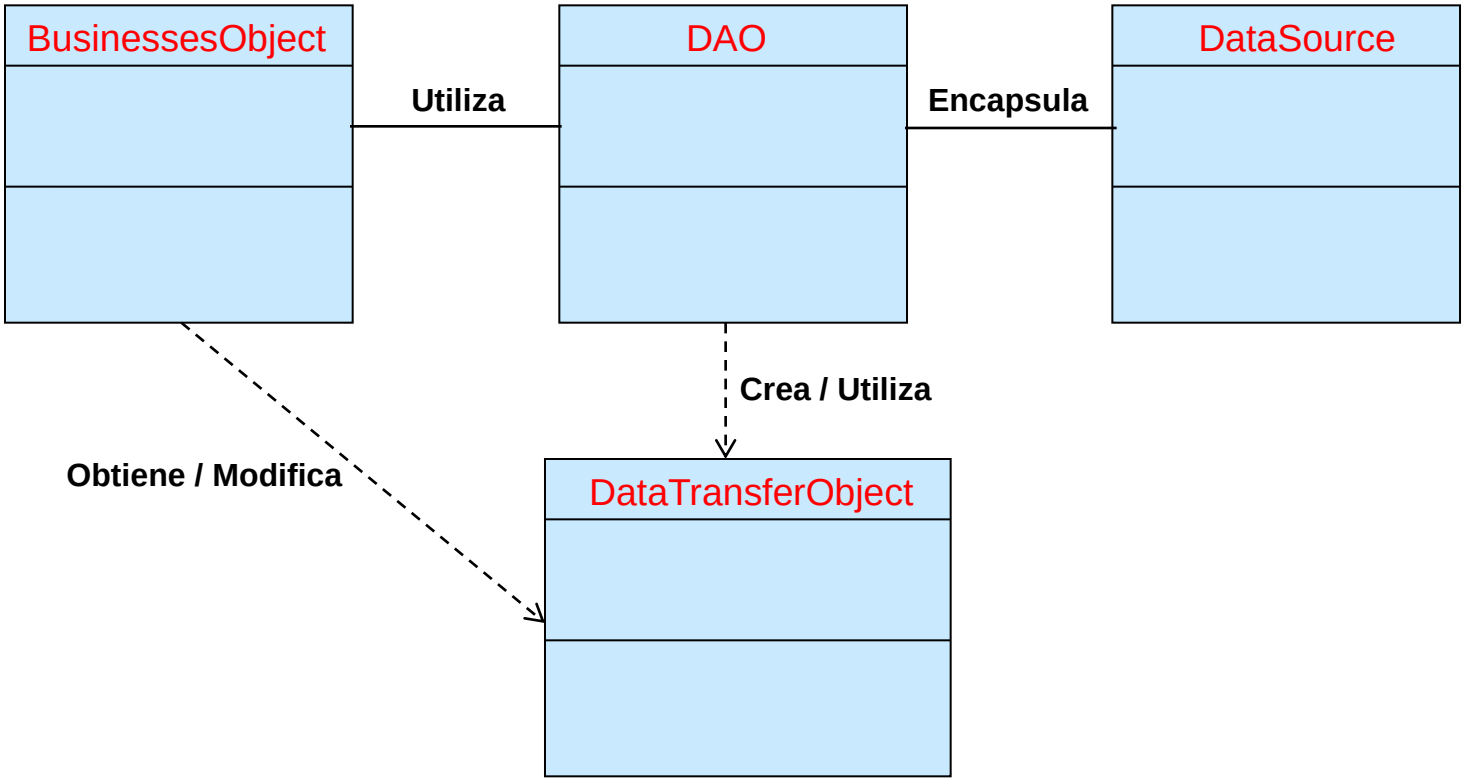
Antes de DAO: código persistente esparcido dentro del la lógica de negocio



Después de DAO: nueva capa para encapsular la interacción con el almacén de persistencia

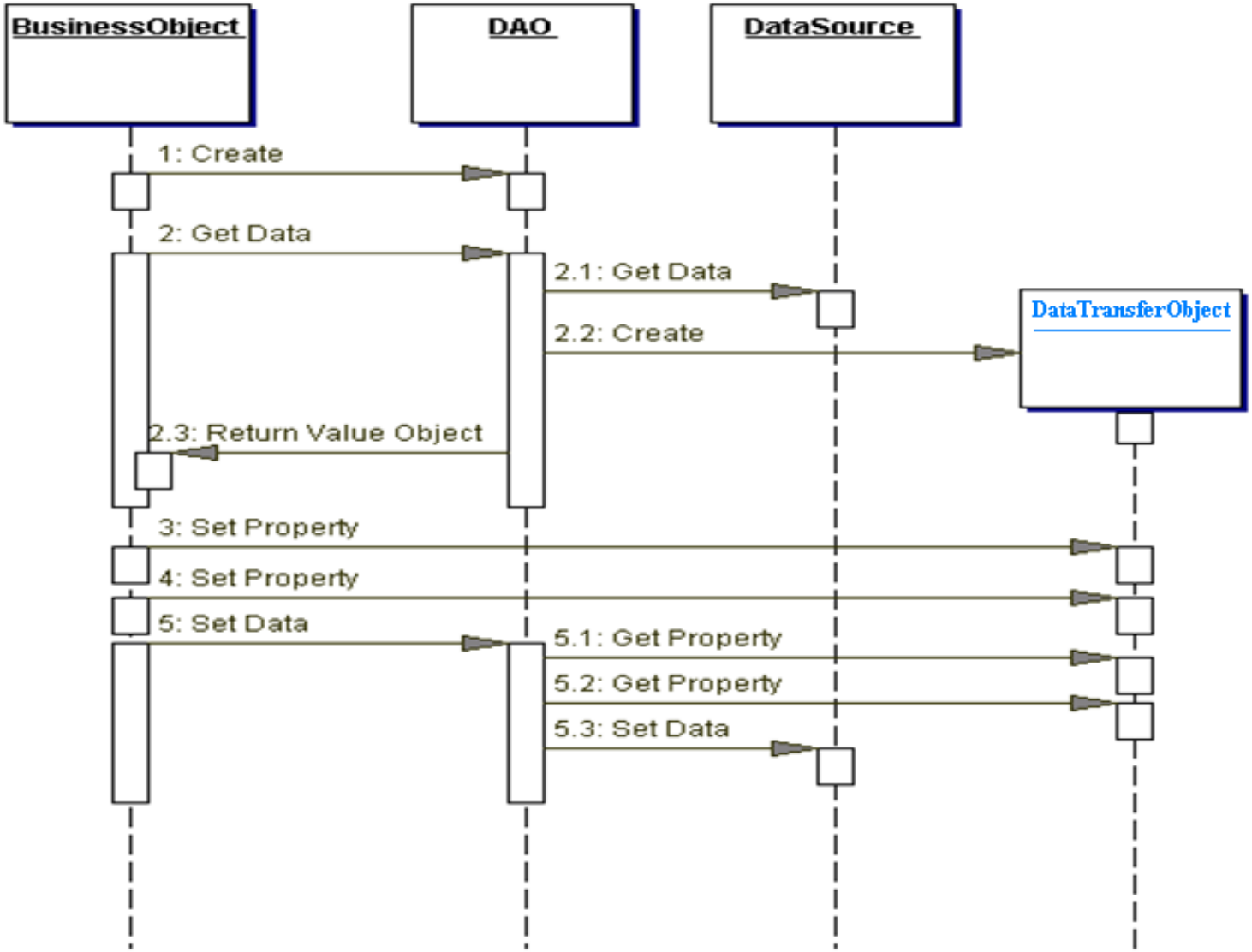
® Derechos reservados

Patrón de Diseño DAO



® Derechos reservados

Patrón de Diseño DAO



® Derechos reservados

Conclusiones

- El patrón de diseño Singleton nos permite tener una sola instancia de una clase.
- El patrón de diseño DAO se usa para separar las operaciones de bajo nivel de acceso a datos (low-level data access), de las operaciones de alto nivel de la lógica del negocio (high-level business logic).
- El uso de DAO no solo estandariza el desarrollo de aplicaciones, sino, también las simplifica; además permite que varios equipos participen en diferentes componentes dependiendo de la especialidad de cada persona.
- Implementando el patrón de diseño Singleton en los componentes DAO aseguramos tener sólo una instancia de cada componente DAO.

® Derechos reservados

Bibliografía

- Desarrollando Soluciones con Java y MySQL Server
  Eric Gustavo Coronel Castillo
- Piensa en Java
  Bruce Eckel
- Como Programar en Java
  Deitel y Deitel
- Java 2
  Steven Holzner
- La Biblia de Java 2 v5.0
  Herbert Schildt
- Acceso a Bases de Datos con Java-JDBC
  Ángel Esteban

® Derechos reservados

Enlaces

- http://java.sun.com/blueprints/corej2eepatterns/Patterns/DataAccessObject.html
- http://www.programacion.net/java/tutorial/patrones2/8/
- http://weblogs.javahispano.org/hip/category/Java
- http://sherekan.com.ar/2008/04/03/data-access-object-i/
- http://sherekan.com.ar/2008/05/09/data-access-object-ii-dao-singleton/

® Derechos reservados