



TECNOLOGICO NACIONAL DE MEXICO INSTITUTO TECNOLOGICO DE OAXACA

Carrera:

Ingeniería en Sistemas Computacionales

Materia:

Diseño e Implementación de software con patrones

Documentación de patrón Proxy

Docente:

Espinoza Pérez Jacob

Equipo:

Ordaz Pacheco Ruudvan

Santos Manuel Julia Marlenny

Vera Acevedo Héctor Aramís

Grupo:

7SB

Fecha de entrega:

07/04/2025

Documentación del Patrón Proxy en el Sistema de Ventas

Introducción

El patrón Proxy es un patrón de diseño estructural que **actúa como un intermediario** para controlar el acceso a otro objeto. En nuestro sistema de gestión de categorías, implementamos este patrón para añadir capas de control y validación a las operaciones CRUD de categorías, sin modificar la lógica principal del sistema.

Interfaz ICategoriaDAO (Sujeto)

Define el contrato que deben implementar tanto el objeto real como el proxy:

Responsabilidades:

- Establecer los métodos fundamentales para gestión de categorías
- Servir como contrato común para implementaciones reales y proxies
- Permitir la sustitución transparente entre implementaciones

```
public interface ICategoriaDAO {
   boolean guardar(Categoria categoria);
   boolean existeCategoria(String descripcion);
   boolean actualizar(Categoria categoria, int idCategoria);
   boolean eliminar(int idCategoria);
}
```

CategoriaDAOReal (Sujeto Real)

public class CategoriaDAOReal implements ICategoriaDAO {

Implementa las operaciones directas con la base de datos:

Características:

- Contiene la lógica pura de acceso a datos
- No incluye validaciones complejas
- Hereda directamente del controlador original

```
goverride
public boolean guardar(Categoria objeto) {
   boolean respuesta = false;
   Connection on = Conexion.getConexion();
   try {

        PreparedStatement consulta = cn.prepareStatement("insert into tb_categoria values(7,7,7)");
        consulta.setInt(1,0);
        consulta.setString(2, objeto.getDescripcion());
        consulta.setInt(3, objeto.getEstado());

        if (consulta.executeUpdate() > 0) {
            respuesta = true;
        }

        Conexion.cerrarConexion();

        } catch (SQLException e) {
            System.out.println("Error al guardar cartegoria: " + e);
        }

        return respuesta;
}

goverride
public boolean existeCategoria(String categoria) {
        boolean respuesta = false;
        String sql = "select descripcion from tb_categoria where descripcion = '" + categoria + "';";
        Statement st;

        try {
            Connection on = Conexion.getConexion();
            st = on.createStatement();
            ResultSet rs = st.executeQuery(sql);
            while (rs.next()) {
                 respuesta = true;
            }
        } catch (SQLException e) {
                  System.out.println("Error al consultar cartegoria: " + e);
        }
        return respuesta;
}
```

```
public boolean actualizar(Categoria objeto, int idCategoria) {
   boolean respuesta = false;
   Connection cn = Conexion.getConexion();
                                          PreparedStatement consulta = cn.prepareStatement("update tb_categoria set descripcion=? where idCategoria = "" + idCategoria + """);
                                           consulta.setString(l, objeto.getDescripcion());
                                          if (consulta.executeUpdate() > 0) {
                                        respuesta = true;
                              Conexion.cerrarConexion();
                             , occor (SQLException e) {
    System.out.println("Error al actualizar cartegoria: " + e);
}
                            return respuesta;
               @Override
               public boolean eliminar(int idCategoria) {
                            boolean respuesta = false;
Connection cn = Conexion.getConexion();
                                          PreparedStatement consulta = cn.prepareStatement(
    "delete from tb_categoria where idCategoria ='" + idCategoria + "'");
                                           consulta.executeUpdate();
                                          if (consulta.executeUpdate() > 0) {
    respuesta = true;
}
                                        Conexion.cerrarConexion();
                             , Galling Could be seen and the second of th
return respuesta;
```

CategoriaDAOProxy (Proxy)

Implementa las validaciones y controles adicionales:

Responsabilidades:

- Validar datos antes de operaciones críticas
- Implementar logging para auditoría
- Controlar acceso a operaciones sensibles
- Gestionar caché si es necesario
- Delegar al objeto real después de validaciones

```
public class CategoriaDAOProxy implements ICategoriaDAO {
    private CategoriaDAOReal categoriaDAOReal;
    private Logger logger = Logger.getLogger(CategoriaDAOProxy.class.getName());
    public CategoriaDAOProxv() {
        this.categoriaDAOReal = new CategoriaDAOReal();
    @Override
    public boolean quardar (Categoria categoria) {
        if(validarDescripcion(categoria.getDescripcion())) {
            logger.log(Level.INFO, "Intentando guardar categoría: {0}", categoría.getDescripcion());
            boolean resultado = categoriaDAOReal.guardar(categoria);
            if(resultado) {
               logger.log(Level.INFO, "Categoría guardada exitosamente");
           return resultado;
        return false;
    @Override
    public boolean existeCategoria(String descripcion) {
       logger.log(Level.INFO, "Verificando existencia de categoría: {0}", descripcion);
        return categoriaDAOReal.existeCategoria(descripcion);
    public boolean actualizar(Categoria categoria, int idCategoria) {
       if(validarDescripcion(categoria.getDescripcion())) {
            logger.log(Level.INFO, "Actualizando categoría ID: {0}", idCategoria);
           return categoriaDAOReal.actualizar(categoria, idCategoria);
        return false;
```

Adaptación del Ctrl_Categoria

El controlador principal se modifica para usar el proxy:

Cambios realizados:

- Sustitución directa del DAO real por el proxy
- Transparencia completa para las interfaces
- Mismos métodos, pero con funcionalidad extendida

```
public class Ctrl_Categoria {
    private ICategoriaDAO categoriaDAO;

public Ctrl_Categoria() {
        this.categoriaDAO = new CategoriaDAOProxy();
}

public boolean guardar(Categoria objeto) {
        return categoriaDAO.guardar(objeto);
}

public boolean existeCategoria(String categoria) {
        return categoriaDAO.existeCategoria(categoria);
}

public boolean actualizar(Categoria objeto, int idCategoria) {
        return categoriaDAO.actualizar(objeto, idCategoria);
}

public boolean eliminar(int idCategoria) {
        return categoriaDAO.eliminar(idCategoria);
}
```

Flujo de Operación con Proxy

- 1. **Interfaz gráfica** realiza solicitud al controlador
- 2. **Controlador** delega la operación al proxy
- 3. **Proxy** ejecuta validaciones preliminares
- 4. Si validaciones son exitosas, **proxy** delega a DAOReal
- 5. **DAOReal** ejecuta la operación en base de datos
- 6. Resultados fluyen de vuelta a través de la misma cadena

Ventajas de la Implementación

- 1. **Control de acceso:** Validación estricta de datos de entrada
- 2. **Seguridad:** Protección contra eliminación de categorías con productos
- 3. **Registro de actividades:** Logging para auditoría y diagnóstico
- 4. Extensibilidad: Fácil añadir nuevas validaciones
- 5. **Mantenibilidad:** Separación clara de responsabilidades
- 6. **Transparencia:** Misma interfaz para componentes existentes

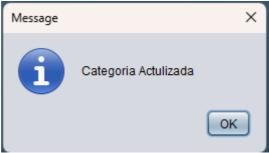
Resultados

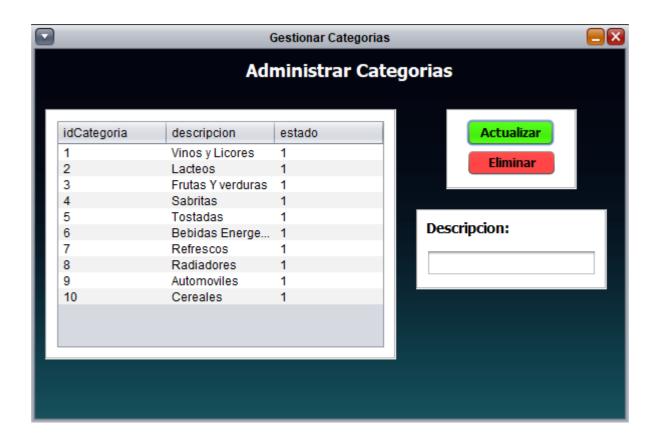












Conclusión

La implementación del patrón Proxy en el sistema de gestión de categorías ha demostrado ser una solución efectiva para:

- 1. Añadir controles sin modificar la lógica principal
- 2. Mejorar la robustez del sistema
- 3. Centralizar validaciones complejas
- 4. **Preparar el sistema** para futuras extensiones
- 5. Mantener limpia la separación de responsabilidades

Este enfoque permite que el sistema de categorías evolucione de manera controlada, facilitando la incorporación de nuevas reglas de negocio o requisitos de seguridad sin afectar los componentes principales del sistema.