

INGENIERÍA DEL SOFTWARE SEGUNDO CUATRIMESTRE - PRIMERA PRÁCTICA

DESARROLLO DE UN AGENTE PARA EL ACCESO A BASE DE DATOS

Patrón Agente

Dado que el segundo cuatrimestre se centra en el desarrollo de sistemas empleando el paradigma orientado a objetos, se plantea como primera práctica el desarrollo de una clase *Agente* que permita una gestión centralizada del acceso a la base de datos.

Dicha clase *Agente* se corresponde con el patrón *Agente* de base de datos o *Broker*. El *Agente* permite a las clases del sistema (que tengan que manejar aspectos de persistencia) el acceso a la base de datos de una forma centralizada: toda la lógica de acceso a la base de datos reside en el *Agente*.

Para el desarrollo del *Agente* se propone el siguiente esqueleto:

```
import java.sql.*;
import java.util.Vector;
public class Agente {
    protected static Agente mInstancia=null;
    protected static Connection mBD;
    //Constructor
    public Agente()throws Exception { /*TO DO*/}

    //Implementación del patrón Singleton
    public static Agente getAgente() throws Exception{
        if (mInstancia==null){
            mInstancia=new Agente();
        }
        return mInstancia;
    }

    private static void conectar() throws Exception {
        String url="jdbc:odbc:Nombre_Cadena_Conexion";
        String driver="sun.jdbc.odbc.JdbcOdbcDriver";
        Class.forName(driver);
        mBD=DriverManager.getConnection(url, "login", "password");
    }

    public static void desconectar() throws Exception{mBD.close();}

    public int insert(String SQL) throws SQLException, Exception{ /*TO DO*/}

    public Vector select(String SQL) throws SQLException,Exception{ /*TO DO*/}

    public int delete(String SQL) throws SQLException,Exception{ /*TO DO*/}

    public int update(String SQL) throws SQLException,Exception{ /*TO DO*/}
}
```

Donde los métodos *insert*, *update*, *delete* y *select* se corresponden con las operaciones CRUD u operaciones para la gestión de la persistencia. El desarrollo de un buen *Agente* implica el que este sea genérico y útil para todas las clases del sistema, por lo que se recomienda la revisión de las siguientes clases de la API de Java:

- *java.sql.ResultSet*
- *java.sql.ResultSetMetaData*
- *java.sql.Types*

Práctica con Agentes

Una vez desarrollado el *Agente* genérico, este se probará con una específica, que realizará consultas contra la base de datos, y mostrará la información por pantalla. En el caso de las operaciones *insert*, *update*, y *delete*, no se mostrará nada, sin embargo, sí que será necesario mostrar el resultado de la consulta cuando sea de tipo *select*.

La clase de prueba es la siguiente:

```
import java.sql.SQLException;
import java.util.Vector;
public class PruebaAgente {
    public static void main(String[] args) {
        Vector resultado = new Vector();
        String SQL_insert_1 = "INSERT INTO Propietario VALUES ('12235625G','Perico','Palotes');";
        String SQL_insert_2 = "INSERT INTO Propietario VALUES ('85749623T','Enjuto','Mojamuto');";
        String SQL_insert_3 = "INSERT INTO Coche VALUES ('8975-CCR','12235625G','Citroen Xsara');";
        String SQL_insert_4 = "INSERT INTO Coche VALUES ('8775-LNH','12235625G','Audi');";
        String SQL_insert_5 = "INSERT INTO Coche VALUES ('4529-JUT','85749623T','Talbot');";
        String SQL_update = "UPDATE Propietario SET apellido='Palustres' WHERE nif='12235625G';";
        String SQL_delete = "DELETE FROM Coche WHERE propietario='85749623T';";
        String SQL_select = "SELECT nombre,apellido,matricula,modelo FROM Propietario,Coche"+
            "WHERE propietario=nif AND nif='12235625G';";

        try
        {
            Agente.getAgente().insert(SQL_insert_1);
            Agente.getAgente().insert(SQL_insert_2);
            Agente.getAgente().insert(SQL_insert_3);
            Agente.getAgente().insert(SQL_insert_4);
            Agente.getAgente().insert(SQL_insert_5);
            Agente.getAgente().update(SQL_update);
            Agente.getAgente().delete(SQL_delete);
            resultado = Agente.getAgente().select(SQL_select);
            //Código para mostrar el resultado.
        }
        catch(SQLException sqle) {
            //Gestión de excepciones
        }catch(Exception e) {
            //Gestión de excepciones
        }
    }
}
```

Como puede observarse, las consultas ya están construidas, y se dispone además de la base de datos, con lo que las tareas a realizar son: (1) implementar la clase *Agente*, (2) implementar en la clase *PruebaAgente* el código para mostrar el resultado de la consulta *select*, y (3) si fuera necesario, el código para la gestión de las excepciones.