

**Universidad Tecnológica Nacional**  
**Tecnicatura en Programación a Distancia**  
**Programación II**  
**Grupo 161**

**Informe Trabajo Práctico Integrador**  
**Sistema de Gestión de Vehículos y Seguros**

**1. Integrantes y Roles del Equipo**

Integrante	Responsabilidades
<b>Valentín Rodríguez Arce</b>	Desarrollo SeguroVehicular, TipoCobertura, DAOs y Services correspondientes
<b>Raul Alberto Robino</b>	Desarrollo Vehiculo, VehiculoDAO, VehiculoImpl, integración transaccional
<b>Mariano Rodríguez Arce</b>	Desarrollo DatabaseConnection, TransactionManager, modelo de datos, menú, Base de datos

**2. Elección del Dominio y Justificación**

**Dominio: Gestión de Vehículos y Seguros**

**Justificación:**

1. Conocimiento a fin: Se está familiarizado con los datos vehiculares pertinentes y se entiende el funcionamiento en el mundo real.
2. Casos reales: Refleja situaciones que requieren atomicidad (vehículo no puede existir sin seguro)
3. Escalabilidad: Permite extensiones futuras (siniestros, conductores, renovaciones)

**3. Diseño del Sistema**

**3.1 Decisión Clave: Relación 1:1 Vehículo-Seguro**

**Opción elegida:**

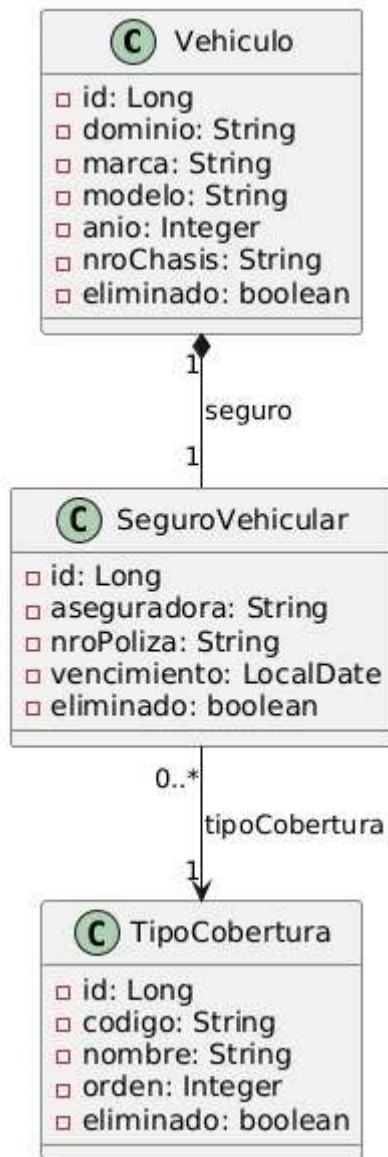
Foreign Key única en Vehiculo hacia SeguroVehicular

Vehiculo.seguro\_id → SeguroVehicular.id (FK UNIQUE)

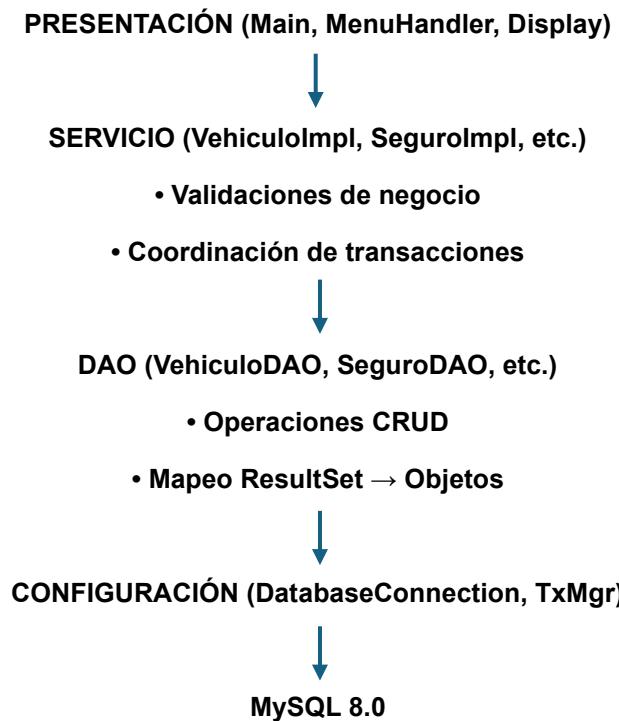
Alternativa	Ventajas	Desventajas	Decisión
<b>FK única en Vehiculo</b>	Clara semántica, facilita queries, flexible	Requiere transacción	<b>ELEGIDA</b>
PK compartida	Garantía 1:1 en BD	Dificulta inserción, inflexible	Rechazada

**Justificación:** La FK única refleja mejor "el vehículo posee un seguro", permite crear el seguro primero en una transacción, y facilita consultas partiendo del vehículo (caso más común).

### 3.2 Diagrama UML Simplificado



#### 4. Arquitectura por Capas



#### Responsabilidades por paquete:

Paquete	Responsabilidad	Características
Models	Entidades del dominio	POJOs, herencia de Base, getters/setters
DAO	Persistencia	PreparedStatement, try-with-resources, métodos *Tx()
Service	Lógica de negocio	Validaciones, orquestación de transacciones
Config	Infraestructura	Factory de conexiones, gestión transaccional
Main	Presentación	MVC simplificado, interacción usuario

#### 5. Persistencia y Transacciones

##### 5.1 Gestión Transaccional

**TransactionManager** implementa AutoCloseable para gestión automática:

```
public class TransactionManager implements AutoCloseable {
```

```

private final Connection conn;
private boolean completed = false;

public TransactionManager() throws SQLException {
    this.conn = DatabaseConnection.getConnection();
    this.conn.setAutoCommit(false);
}

public void commit() throws SQLException {
    conn.commit();
    completed = true;
}

@Override
public void close() {
    if (!completed) rollback();
    conn.setAutoCommit(true);
    conn.close();
}
}

```

## 5.2 Ejemplo de Uso: Crear Vehículo con Seguro

```

public Long crearVehiculoConSeguro(Vehiculo v) {
    validarObligatorios(v);

    try (TransactionManager tx = new TransactionManager()) {
        // 1. Insertar seguro
        Long idSeguro = seguroDAO.insertTx(v.getSeguro(), tx.getConnection());
        v.getSeguro().setId(idSeguro);

        // 2. Insertar vehículo
        Long idVehiculo = vehiculoDAO.insertTx(v, tx.getConnection());

        tx.commit(); // Si todo OK
        return idVehiculo;
    } // Rollback automático si falla
}

```

**Flujo:**

1. Validaciones de negocio
2. Iniciar transacción (autocommit=false)
3. INSERT SeguroVehicular → obtener ID
4. INSERT Vehiculo con seguro\_id
5. COMMIT (o ROLLBACK automático)

## 6. Validaciones y Reglas de Negocio

### 6.1 Matriz de Validaciones

Entidad	Regla	Capa	Implementación
Vehiculo	Dominio único	Service + BD	existsByDominio() + UNIQUE
Vehiculo	Marca/modelo obligatorios	Service	validarObligatorios()
Vehiculo	Debe tener seguro	Service	if (seguro == null) throw...
SeguroVehicular	Póliza única	Service + BD	existsByNroPoliza() + UNIQUE
SeguroVehicular	Vencimiento futuro	Service	if (vencimiento.isBefore(now()))
SeguroVehicular	Cobertura válida	Service	coberturaDAO.findById()
TipoCobertura	Código único	Service + BD	existsByCodigo() + UNIQUE

### 6.2 Ejemplo de Validación

```
private void validarObligatorios(Vehiculo v) {
    if (v == null)
        throw new RuntimeException("Falta objeto vehículo");
    if (isBlank(v.getDominio()))
        throw new RuntimeException("Dominio obligatorio");
    if (isBlank(v.getMarca()))
        throw new RuntimeException("Marca obligatoria");
    if (isBlank(v.getModelo()))
        throw new RuntimeException("Modelo obligatorio");
}
```

### 6.3 Reglas de Negocio Clave

ID	Regla	Tipo
RN01	Un vehículo requiere exactamente un seguro	Estructural
RN02	Dominio y póliza son únicos	Negocio
RN03	Vencimiento no puede estar vencido	Negocio
RN04	Soft delete (no eliminación física)	Operacional
RN05	Al eliminar vehículo se elimina su seguro	Negocio

## 7. Pruebas Realizadas

### 7.1 Casos de Prueba

Crear vehículo válido Dominio: ABC123, Marca: Toyota  Creado con ID  PASS

```
----- MSH# VEHICULOS -----
1) Crear Vehículo + Seguro
2) Buscar por dominio
3) Listar todos
4) Eliminar (soft) Vehículo + Seguro
0) Salir
Opción: 1

--- NUEVO VEHICULO ---
Dominio: ABC123
Marca: Toyota
Modelo: India
A#o (enter si no aplica): 2000
N# chasis (enter si no aplica): 1111111111111111

--- SEGURO VEHICULAR ---
Aseguradora: La City
N# póliza (#nica): 123456789

--- TIPOS DE COBERTURA ---
1) BC - Responsabilidad Civil
2) TERCEROS - Terceros
3) TODO RIESGO - Todo Riesgo
ID de cobertura: 1
Vencimiento (AAAA-MM-DD): 2020-11-28
* Vehículo creado con id: 16387
```

Dominio duplicado Dominio: ABC123 (existe)  Error "Ya existe"  PASS

```
----- MSH# VEHICULOS -----
1) Crear Vehículo + Seguro
2) Buscar por dominio
3) Listar todos
4) Eliminar (soft) Vehículo + Seguro
0) Salir
Opción: 1

--- NUEVO VEHICULO ---
Dominio: ABC123
Marca: Ford
Modelo: Ranger
A#o (enter si no aplica): 2000
N# chasis (enter si no aplica): 123456789

--- SEGURO VEHICULAR ---
Aseguradora: La City
N# póliza (#nica): 123456789

--- TIPOS DE COBERTURA ---
1) BC - Responsabilidad Civil
2) TERCEROS - Terceros
3) TODO RIESGO - Todo Riesgo
ID de cobertura: 1
Vencimiento (AAAA-MM-DD): 2020-11-28
* ERROR: Ya existe un vehículo con ese dominio.
```

Póliza duplicada Póliza: POL-001 (existe)  Error "Póliza existe"  PASS

```
----- MENÚ VEHICULOS -----
1) Crear Vehículo + Seguro
2) Buscar por dominio
3) Listar todos
4) Eliminar (soft) Vehículo + Seguro
0) Volver
Opción: 1

---- NUEVO VEHICULO ----
Dominio: KIA3311
Marca: Ford
Modelo: Ranger
Año (enter si no aplica): 
Nº chasis (enter si no aplica): 

---- SEGURO VEHICULAR ----
Aseguradora: La Caja
Nº póliza (#nica): POL-001

---- TIPOS DE COBERTURA ----
1) RC - Responsabilidad Civil
2) TERCEROS - Terceros
3) TODO_RIESGO - Todo Riesgo
ID de cobertura: 1
Vencimiento (AAAA-MM-DD): 2020-11-25
? ERROR: La póliza ya existe: POL-001
```

Vencimiento pasado	Vencimiento: 2020-01- 01	<input checked="" type="checkbox"/> Error "Fecha vencida"	<input checked="" type="checkbox"/> PASS
-----------------------	-----------------------------	--------------------------------------------------------------	------------------------------------------

```
----- MENÚ VEHICULOS -----
1) Crear Vehículo + Seguro
2) Buscar por dominio
3) Listar todos
4) Eliminar (soft) Vehículo + Seguro
0) Volver
Opción: 1

---- NUEVO VEHICULO ----
Dominio: KIA3311
Marca: Ford
Modelo: Ranger
Año (enter si no aplica): 2024
Nº chasis (enter si no aplica): KSD03123445644

---- SEGURO VEHICULAR ----
Aseguradora: La Caja
Nº póliza (#nica): 1231230987

---- TIPOS DE COBERTURA ----
1) RC - Responsabilidad Civil
2) TERCEROS - Terceros
3) TODO_RIESGO - Todo Riesgo
ID de cobertura: 3
Vencimiento (AAAA-MM-DD): 2024-11-10
? ERROR: La fecha de vencimiento no puede estar vencida.
```

Buscar por dominio Dominio: ABC123  Datos completos  PASS

```
----- MEN  VEH CULOS -----
1) Crear Veh culo + Seguro
2) Buscar por dominio
3) Listar todos
4) Eliminar (soft) Veh culo + Seguro
0) Volver
Opci n: 2

Dominio a buscar: ABC123

--- Resultado ---
ID: 16387
Veh culo: Toyota Yaris (ABC123)
Chasis: 123123ASDASD
P liza: 1231231
Aseguradora: La Caja
Cobertura: RC - Responsabilidad Civil
Vencimiento: 2030-11-23
```

Eliminar vehículo ID: 1  Soft delete (veh + seg)  PASS

```
----- MEN  VEH CULOS -----
1) Crear Veh culo + Seguro
2) Buscar por dominio
3) Listar todos
4) Eliminar (soft) Veh culo + Seguro
0) Volver
Opci n: 4

ID de veh culo a eliminar (soft): 1
? Eliminado con  xito (veh culo + seguro).
```

## 7.2 Consultas SQL de Verificación

```
-- Verificar relaci n 1:1 (no debe haber seguros compartidos)

SELECT seguro_id, COUNT(*)
FROM Vehiculo WHERE eliminado=0
GROUP BY seguro_id HAVING COUNT(*) > 1;

-- Resultado esperado: 0 filas

-- Verificar soft delete

SELECT 'Vehiculos' AS tabla,
       COUNT(*) AS total,
       SUM(CASE WHEN eliminado=0 THEN 1 ELSE 0 END) AS activos
```

```

FROM Vehiculo;

-- Consulta completa con JOINs

SELECT v.dominio, v.marca, v.modelo, s.nro_poliza,
       s.aseguradora, tc.codigo, tc.nombre
  FROM Vehiculo v
  JOIN SeguroVehicular s ON v.seguro_id = s.id
  JOIN TipoCobertura tc ON s.tipo_cobertura = tc.id
 WHERE v.eliminado = 0;

```

## 8. Conclusiones

### 8.1 Objetivos Alcanzados

- CRUD completo para tres entidades con relaciones complejas
- Arquitectura en capas robusta con separación de responsabilidades
- Validaciones en múltiples niveles (Service + BD)
- Soft delete implementado para preservar historial

### 8.2 Mejoras Futuras

#### Técnicas

- Tests unitarios automatizados con JUnit 5

#### Funcionales

- Gestión de siniestros y reclamos
- Múltiples conductores por vehículo
- Notificaciones de vencimiento

#### Arquitectura

- API REST para exponer funcionalidades
- Interfaz gráfica (JavaFX o web con React)

## 10. Referencias

#### Documentación:

- Oracle Java SE 17 Documentation: <https://docs.oracle.com/en/java/javase/17/>
- MySQL 8.0 Reference Manual: <https://dev.mysql.com/doc/refman/8.0/en/>

#### Herramientas:

- JDK: OpenJDK 17 | IDE: IntelliJ IDEA / Eclipse
- SGBD: MySQL 8.0 | Cliente: MySQL Workbench
- Driver: MySQL Connector/J 8.0.33

- Control de versiones: Git

**Inteligencia Artificial:** Claude AI (Anthropic - Sonnet 4) fue utilizada como herramienta de apoyo para:

- Revisión de código y detección de bugs
- Sugerencias de optimización
- Generación de documentación