Integración de Java y MySQL utilizando MyBatis

MyBatis es un framework para el manejo de la persistencia y permite eliminar casi por completo el código SQL y JDBC de los archivos en java.

Cargar el proyecto.
El primer paso es cargar el proyecto Java_MyBatis el cuál tiene las clases necesarias para la lectura de un archivo y al cuál se le irán agregando las clases y archivos necesarios para trabajar con MyBatis.

Instalando las Bases de Datos
Es necesario tener una base de datos con diferentes tablas, por el momento no se tendrán ninguna restricción sobre los datos para facilitar la inserción. Los *scripts* de creación de la base de datos se muestran a continuación:

Base de Datos

```
CREATE DATABASE IF NOT EXISTS productos_profesor CHARACTER SET utf8 COLLATE utf8_general_ci;
USE productos_profesor;
```

Tabla de Áreas

```
CREATE TABLE area (
	id_area INT AUTO_INCREMENT PRIMARY KEY,
	nombre VARCHAR(45)
) ENGINE=InnoDB;
```

Tabla de Profesores

```
CREATE TABLE profesor (
	numero_economico VARCHAR(10) PRIMARY KEY,
	nombre VARCHAR(45),
	primer_apellido VARCHAR(45),
	segundo_apellido VARCHAR(45),
	grado_academico VARCHAR(45),
	area_id_area INT NOT NULL,
	FOREIGN KEY (area_id_area) REFERENCES area (id_area)
		ON DELETE RESTRICT ON UPDATE RESTRICT) ENGINE=InnoDB;
```

Tabla de Proyectos de Investigación

```
CREATE TABLE proyecto_investigacion (
	id_proyecto_investigacion INT AUTO_INCREMENT PRIMARY KEY,
	nombre VARCHAR (150),
	area_id_area INT NOT NULL,
	FOREIGN KEY (area_id_area) REFERENCES area(id_area)
	ON DELETE RESTRICT ON UPDATE RESTRICT
)ENGINE = InnoDB;
```

Tabla de relación entre Profesores y Proyectos de Investigación

```
CREATE TABLE profesor_proyecto_investigacion(
id_profesor_proyecto_investigacion INT AUTO_INCREMENT PRIMARY KEY,
rol VARCHAR(15),
profesor_numero_economico VARCHAR(10) NOT NULL,
proyecto_investigacion_id_proyecto_investigacion INT NOT NULL,
FOREIGN KEY (profesor_numero_economico)
        REFERENCES profesor(numero_economico)
        ON DELETE CASCADE ON UPDATE CASCADE,
FOREIGN KEY (proyecto_investigacion_id_proyecto_investigacion)
        REFERENCES proyecto_investigacion(id_proyecto_investigacion)
        ON DELETE CASCADE ON UPDATE CASCADE
)ENGINE=InnoDB;
```

Instalando el Entorno de MyBatis

Para comenzar, se debe descargar el archivo jar de MyBatis, una de las direcciones en donde se puede realizar es:

```
http://mvnrepository.com/artifact/org.mybatis/mybatis/3.0.1
```

También es necesario descargar el archivo que sirve como conector entre Java y MySQL, éste se puede descargar de:

```
http://dev.mysql.com/downloads/connector/j/
```

Es conveniente seleccionar la versión que no depende de la plataforma.

Finalmente será necesario descargar el archivo *commons-logging* de:

```
http://commons.apache.org/proper/commons-logging/download_logging.cgi
```

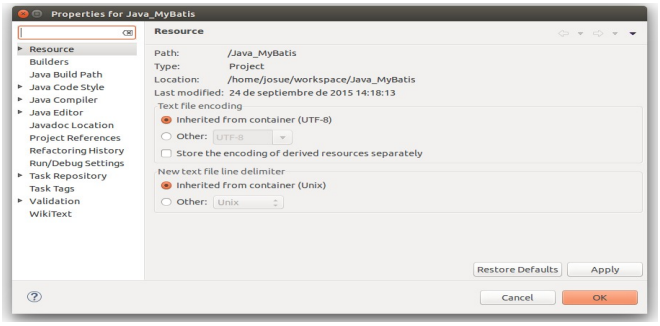
Tanto el conector como *logging* contienen varios archivos y directorios, sin embargo solo será necesario utilizar los archivos JAR.

Se deben agregar al proyecto como *jars* externas, es recomendable crear un directorio llamada *jars* dentro de la estructura de directorios para tenerlas siempre al alcance del proyecto. Aquí se encontrarán los siguientes archivos:

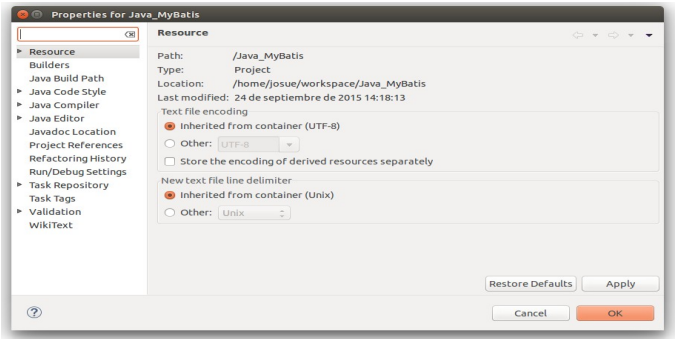
- commons-logging-1.2-javadoc.jar
- mybatis-3.0.1.jar
- mysql-connector-java-5.1.36-bin.jar

Una vez colocadas, es necesario agregarlas al entorno de trabajo, para eso se deben seguir los siguientes pasos:

Dar clic derecho sobre el nombre del proyecto, posteriormente seleccionar la opción *Properties*



Seleccionar la opción *Java Build Path* del menú lateral y dar clic en el botón *Add External JARs...*



Seleccionar los tres Jars que se descargaron del directorio en donde se colocaron y dar clic en *Ok* o *Aceptar*, finalmente dar clic en *OK*.

Configurando MyBatis

El framework MyBatis se basa en archivos de configuración XML, ya sea para la configuración de la conexión o para el mapeo de oepraciones y clases con tablas. Lo primero será crear una clase encargada de crear los objetos necesarios para la creación de la conexión a base de datos. En el paquete **`uam.bases.mybatis.conexion`** se creará la clase **EstablecerConexion**.

EstablecerConexion

```
package uam.bases.mybatis.conexion;

import java.io.IOException;
import java.io.Reader;

import org.apache.ibatis.io.Resources;
import org.apache.ibatis.session.SqlSessionFactory;
import org.apache.ibatis.session.SqlSessionFactoryBuilder;
```

```
public class EstablecerConexion {

	private static SqlSessionFactory sessionFactory = null;
	private static Reader reader;
	private static String ARCHIVO_CONFIGURACION = "sql_configuracion.xml";

	public SqlSessionFactory crearSession(){

			try{
				reader =
Resources.getResourceAsReader(ARCHIVO_CONFIGURACION);
				sessionFactory = new
SqlSessionFactoryBuilder().build(reader);
			}catch (IOException e){
				e.printStackTrace();
			}
		return sessionFactory;

	}


}
```

Uno de los elementos más importantes es el valor de la variable **ARCHIVO_CONFIGURACION**, éste será el archivo en donde se colocará la configuración de la conexión a la base de datos y será un archivo XML que se creará en el directorio *src* del proyecto.

Ésto se logra dando clic derecho sobre el directorio *src*, posteriormente *Other* y seleccionar *XML File*, el nombre debe ser el mismo que se especificó en la variable **ARCHIVO_CONFIGURACION**, en este caso **sql_configuracion.xml**

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE configuration PUBLIC "-//mybatis.org/DTD Config 3.0//EN"
"http://mybatis.org/dtd/mybatis-3-config.dtd">

<configuration>

	<typeAliases>

	</typeAliases>

	<environments default="development">
		<environment id="development">
		<transactionManager type="JDBC"></transactionManager>

		<dataSource type="POOLED">
		<property name="driver" value="com.mysql.jdbc.Driver"></property>
		<property name="url"
value="jdbc:mysql://localhost/productos_profesor"></property>
		<property name="username" value="root"></property>
```

```
            <property name="password" value="root"></property>
            </dataSource>
            </environment>
      </environments>

      <mappers>

      </mappers>

</configuration>
```

Aquí se manejan los valores más comunes para MyBatis, los que deben tomar los valores **adecuados** son aquellos especificados en <property> en donde se especificarán el manejador (debe ser el de JDBC), la url (dirección y nombre de la Base de Datos), usuario (el usuario de la base de datos) y el password (contraseña de la Base de Datos).
También se debe especificar en <transactionManager> el tipo de manejador, en este caso como es Java, el valor es JDBC.

Servicios
MyBatis trabaja a partir de servicios e interfaces, se crearán distintos servicios para manejar las operaciones de cada una de las tablas a controlar, se puede crear un solo servicio general, pero es conveniente crear uno por cada tabla para tener mayor orden en el código. Éstos servicios no son clases, son interfaces y se crearán en el paquete **`uam.bases.mybatis.servicios`**, por el momento solo se manejará el método para insertar.

Para esto, se dará clic derecho sobre el nombre del paquete y se seleccionará una Interface.

ServiciosArea

```
package uam.bases.mybatis.servicios;

import uam.bases.mybatis.clases.Area;

public interface ServiciosArea {

      public void insertar(Area area);
}
```

ServiciosProfesor

```
package uam.bases.mybatis.servicios;

import uam.bases.mybatis.clases.Profesor;

public interface ServiciosProfesor {

      public void insertar( Profesor profesor);
}
```

ServiciosProyectoInvestigacion

```
package uam.bases.mybatis.servicios;

import uam.bases.mybatis.clases.ProyectoInvestigacion;

public interface ServiciosProyectoInvestigacion {

	public void insertar(ProyectoInvestigacion proyectoInvestigacion);

}
```

ServiciosProfesorProyectoInvestigacion

```
package uam.bases.mybatis.servicios;

import uam.bases.mybatis.clases.ProfesorProyectoInvestigacion;

public interface ServiciosProfesorProyectoInvestigacion {

	public void insertar(ProfesorProyectoInvestigacion profesorProyecto);

}
```

Estableciendo la Conexión

Será necesario establecer una conexión cada que se quiere realizar una operación en la Base de Datos, éstas sentencias se colocarán en los métodos que obtienen la información de los archivos. Las siguientes líneas se colocan antes del ***try*** que maneja la lectura del archivo

OperacionesArea

```
SqlSessionFactory sessionFactory;
EstablecerConexion conexion = new EstablecerConexion();

	sessionFactory = conexion.crearSession();
	SqlSession session = sessionFactory.openSession();
	ServiciosArea servicioArea = session.getMapper(ServiciosArea.class);
```

Una vez abierta la sesión, se llamará al método que se encuentra en la Interface creada, ésto se hará cada que se desee insertar un elemento, en este caso después de leer la línea y procesarla, es recomendable que después de cada operación que modifique una tabla, se realice un ***commit***, al finalizar las inserciones, se debe cerrar la conexión.

De esta manera el método ***insertarAreas*** de la clase ***OperacionesArea*** queda:

OperacionesArea

```
public void insertarAreas(){

	String cadenaLeida = "";
	FileReader fr;

	SqlSessionFactory sessionFactory;
	EstablecerConexion conexion = new EstablecerConexion();

	sessionFactory = conexion.crearSession();
	SqlSession session = sessionFactory.openSession();
	ServiciosArea servicioArea = session.getMapper(ServiciosArea.class);



	try {
		fr = new FileReader("areas.txt");
		BufferedReader archivoLectura = new BufferedReader(fr);

		System.out.println("Insertando las Áreas en la Base de Datos");
		cadenaLeida = archivoLectura.readLine();
		while (cadenaLeida != null) {
			Area area = new Area();
			StringTokenizer st = new StringTokenizer(cadenaLeida, ",");

			area.setIdArea(Integer.parseInt(st.nextToken()));
			area.setNombreArea(st.nextToken());

			servicioArea.insertar(area);
			session.commit();

			System.out.println(area.toString());

			cadenaLeida = archivoLectura.readLine();
		}
			archivoLectura.close();
			session.close();
		} catch (FileNotFoundException e) {
			System.out.println("No se pudo encontrar el archivo");
			e.printStackTrace();
		} catch (IOException e) {
			System.out.println("No se pudo leer del archivo");
			e.printStackTrace();
		}

	}
```

Antes de poder insertar elementos, se debe crear un archivo que mapeé las operaciones y cuando sea necesario las clases con sus correspondientes tablas en la base de datos.

Mapeo de Operaciones

Se debe realizar dos acciones para mapear las operaciones, primero indicar que se tiene un archivo de mapeo en el archivo de configuración de MyBatis (**sql_configuracion.xml**), esto se realiza entre las etiquetas **<mappers>** y **</mappers>**. Se puede crear solo un archivo de mapeo para todas las operaciones, o uno para manejar las operaciones de cada clase.

sql_configuracion.xml

```
<mappers>
      <mapper resource="uam/bases/mybatis/mapeos/MapeoAreas.xml"></mapper>
</mappers>
```

El atributo (etiqueta) ***mapper*** indica la ruta del archivo de mapeo, en este caso el archivo se llamará ***MapeoAreas.xml*** como se encuentra en el paquete ***uam.bases.mybatis.mapeos***, se especifica la ruta como directorios finalizando con ***MapeoAreas.xml***

El siguiente paso es crear el archivo con el mapeo y escribir las instrucciones para el manejo de la Base de Datos. Se debe respetar la ruta y nombre del archivo indicado en ***<mapper>***

MapeoAreas.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE mapper PUBLIC "-//mybatis.org/DTD Config 3.0//EN"
"http://mybatis.org/dtd/mybatis-3-mapper.dtd">

<mapper namespace="uam.bases.mybatis.servicios.ServiciosArea">

            <insert id="insertar" parameterType="area">
                  INSERT INTO area (nombre) VALUE (#{nombreArea})
            </insert>

</mapper>
```

Aquí se crea el mapeo para la operación de insertar un área, el nombre debe ser el mismo que el del método en la Interface que da servicio a las operaciones con áreas.

La etiqueta ***<insert>*** recibe como parámetro (***parameterType***) un objeto de la clase ***Area***, el cuál recibe el alias de ***area*** y `(#{nombreArea})` representa el nombre del atributo de la clase que se manejará. Para esto es necesario relacionar la clase ***Area*** con su alias ***area***. Esto se hace en el archivo de configuración de Mybatis, ésto se coloca fuera de las etiquetas de configuración del entorno.

sql_configuracion.xml

```
<typeAliases>
      <typeAlias type="uam.bases.mybatis.clases.Area" alias="area"></typeAlias>
</typeAliases>
```

Aquí se especifica que el alias ***area*** corresponde a la clase ***Area***, es necesario notar que se coloca la ruta completa de la clase (es decir, junto con el paquete). De esta manera el archivo de configuración queda:

sql_configuracion.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE configuration PUBLIC "-//mybatis.org/DTD Config 3.0//EN"
"http://mybatis.org/dtd/mybatis-3-config.dtd">

<configuration>

	<typeAliases>
		<typeAlias type="uam.bases.mybatis.clases.Area"
		alias="area"></typeAlias>
	</typeAliases>

	<environments default="development">
		<environment id="development">
			<transactionManager type="JDBC"></transactionManager>
			<dataSource type="POOLED">
			<property name="driver"
value="com.mysql.jdbc.Driver"></property>
			<property name="url"
value="jdbc:mysql://localhost/productos_profesor"></property>
			<property name="username" value="root"></property>
			<property name="password" value="root"></property>
			</dataSource>
			</environment>
	</environments>

	<mappers>
		<mapper resource="uam/bases/mybatis/mapeos/MapeoAreas.xml"></mapper>
	</mappers>

</configuration>
```

Ya es posible ejecutar la aplicación, solamente el método ***llenarAreas()*** de la clase ***Principal***, los datos del archivo con las áreas se habrán insertado en la tabla correspondiente.

Insertando Profesores

Para insertar un profesor, el funcionamiento es básicamente el miso que para las áreas.

OperacionesProfesor

```
public void insertarProfesores() {

		String cadenaLeida = "";
		FileReader fr;

		SqlSessionFactory sessionFactory;
		EstablecerConexion conexion = new EstablecerConexion();

		sessionFactory = conexion.crearSession();
		SqlSession session = sessionFactory.openSession();
		ServiciosProfesor servicioProfesor =
session.getMapper(ServiciosProfesor.class);


		try {
			fr = new FileReader("profesores.txt");
			BufferedReader archivoLectura = new BufferedReader(fr);


			System.out.println("Insertando los Profesores en la Base de
Datos");
			cadenaLeida = archivoLectura.readLine();
			while (cadenaLeida != null) {

				Profesor profesor = new Profesor();

				StringTokenizer st = new StringTokenizer(cadenaLeida,
",");


				profesor.setNumero_economico(st.nextToken());
				profesor.setNombre(st.nextToken());
				profesor.setPrimer_apellido(st.nextToken());
				profesor.setSegundo_apellido(st.nextToken());
				profesor.setGrado_academico(st.nextToken());

	profesor.setArea_id_area(Integer.parseInt(st.nextToken()));

				servicioProfesor.insertar(profesor);
				session.commit();

				System.out.println(profesor.toString());

				cadenaLeida = archivoLectura.readLine();
			}
			archivoLectura.close();
			session.close();
		} catch (FileNotFoundException e) {
			System.out.println("No se pudo encontrar el archivo");
			e.printStackTrace();
```

```
            } catch (IOException e) {
                  System.out.println("No se pudo leer del archivo");
                  e.printStackTrace();
            }

      }
```

Se debe indicar que habrá un nuevo archivo de mapeo, esto se hace en el archivo de configuración de MyBatis en donde también se realizará el mapeo de un alias llamado ***profesor*** con la clase ***Profesor***.

sql_configuracion.xml

```
<typeAliases>
      <typeAlias type="uam.bases.mybatis.clases.Area" alias="area"></typeAlias>
      <typeAlias type="uam.bases.mybatis.clases.Profesor"
      alias="profesor"></typeAlias>
</typeAliases>

<mappers>
      <mapper resource="uam/bases/mybatis/mapeos/MapeoAreas.xml"></mapper>
      <mapper resource="uam/bases/mybatis/mapeos/MapeoProfesor.xml"></mapper>
</mappers>
```

Mapeo del Profesor

Se creará el archivo para mapear las operaciones del profesor respetando la ruta (paquete) y nombre establecido en el archivo de configuración.

MapeoProfesor.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE mapper PUBLIC "-//mybatis.org/DTD Config 3.0//EN"
"http://mybatis.org/dtd/mybatis-3-mapper.dtd">

<mapper namespace="uam.bases.mybatis.servicios.ServiciosProfesor">

      <insert id="insertar" parameterType="profesor">
      INSERT INTO profesor
(numero_economico,nombre,primer_apellido,segundo_apellido,grado_academico,
area_id_area)      VALUES
(#{numero_economico},#{nombre},#{primer_apellido},#{segundo_apellido},#{grado_aca
demico}, #{area_id_area})
      </insert>
</mapper>
```

Insertando Proyectos de Investigación

Clases necesarias para insertar un Proyecto de Investigación.

OperacionesProyectoInvestigacion

```
	public void insertarProyectosInvestigacion(){

		String cadenaLeida = "";
		FileReader fr;

		SqlSessionFactory sessionFactory;
		EstablecerConexion conexion = new EstablecerConexion();

		sessionFactory = conexion.crearSession();
		SqlSession session = sessionFactory.openSession();
		ServiciosProyectoInvestigacion servicioProyecto =
		session.getMapper(ServiciosProyectoInvestigacion.class);

		try {
			fr = new FileReader("proyectos_investigacion.txt");
			BufferedReader archivoLectura = new BufferedReader(fr);

			System.out.println("Insertando los Proyectos de Investigación en
la Base de Datos");
			cadenaLeida = archivoLectura.readLine();
			while (cadenaLeida != null) {
				ProyectoInvestigacion proyecto = new
ProyectoInvestigacion();
				StringTokenizer st = new StringTokenizer(cadenaLeida,
",");

				proyecto.setNombre(st.nextToken());

	proyecto.setArea_id_area(Integer.parseInt(st.nextToken()));

				servicioProyecto.insertar(proyecto);
				session.commit();

				System.out.println(proyecto.toString());

				cadenaLeida = archivoLectura.readLine();
			}
			archivoLectura.close();
			session.close();

		} catch (FileNotFoundException e) {
			System.out.println("No se pudo encontrar el archivo");
			e.printStackTrace();
		} catch (IOException e) {
			System.out.println("No se pudo leer del archivo");
			e.printStackTrace();
		}

	}
```

Se debe crear su archivo de mapeo y relacionar el alias **proyecto_investigacion** con la clase **ProyectoInvestigacion**.

sql_configuracion.xml

```
<typeAliases>
      <typeAlias type="uam.bases.mybatis.clases.Area" alias="area"></typeAlias>
      <typeAlias type="uam.bases.mybatis.clases.Profesor"
      alias="profesor"></typeAlias>
      <typeAlias type="uam.bases.mybatis.clases.ProyectoInvestigacion"
      alias="proyecto_investigacion"></typeAlias>
</typeAliases>

<mappers>
      <mapper resource="uam/bases/mybatis/mapeos/MapeoAreas.xml"></mapper>
      <mapper resource="uam/bases/mybatis/mapeos/MapeoProfesor.xml"></mapper>
<mapper
resource="uam/bases/mybatis/mapeos/MapeoProyectoInvestigacion.xml"></mapper>
</mappers>
```

Mapeo del Proyecto de Investigación

Se creará el archivo para mapear las operaciones del proyecto de investigación respetando la ruta (paquete) y nombre establecido en el archivo de configuración.

MapeoProyectoInvestigacion.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE mapper PUBLIC "-//mybatis.org/DTD Config 3.0//EN"
"http://mybatis.org/dtd/mybatis-3-mapper.dtd">

<mapper namespace="uam.bases.mybatis.servicios.ServiciosProyectoInvestigacion">

      <insert id="insertar" parameterType="proyecto_investigacion">
      INSERT INTO proyecto_investigacion (nombre,area_id_area)
       VALUES (#{nombre},#{area_id_area})
      </insert>
</mapper>
```

Relacionando Profesores y Proyectos de Investigación

Clases necesarias para relacionar un Profesor con uno o varios Proyectos de Investigación.

OperacionesProfesorProyectoInvestigacion

```
public void relacionarProfesorProyectosInvestigacion(){

	String cadenaLeida = "";
	FileReader fr;

	SqlSessionFactory sessionFactory;
	EstablecerConexion conexion = new EstablecerConexion();

	sessionFactory = conexion.crearSession();
	SqlSession session = sessionFactory.openSession();
	ServiciosProfesorProyectoInvestigacion servicioProfesorProyecto =
	session.getMapper(ServiciosProfesorProyectoInvestigacion.class);

	try {
		fr = new FileReader("profesor_proyecto_investigacion.txt");
		BufferedReader archivoLectura = new BufferedReader(fr);


		System.out.println("Relacionando Profesores con Proyectos de
Investigación en la Base de Datos");
		cadenaLeida = archivoLectura.readLine();
		while (cadenaLeida != null) {
			ProfesorProyectoInvestigacion profesorProyecto = new
ProfesorProyectoInvestigacion();
			StringTokenizer st = new StringTokenizer(cadenaLeida, ",");

			profesorProyecto.setRol(st.nextToken());
			profesorProyecto.setProfesor_numero_economico(st.nextToken());
			profesorProyecto.setProyecto_investigacion_id_proyecto_investiga
			cion(Integer.parseInt(st.nextToken()));

			System.out.println(profesorProyecto.toString());

			servicioProfesorProyecto.insertar(profesorProyecto);
			session.commit();

			cadenaLeida = archivoLectura.readLine();
		}
		archivoLectura.close();
		session.close();
	} catch (FileNotFoundException e) {
		System.out.println("No se pudo encontrar el archivo");
		e.printStackTrace();
	} catch (IOException e) {
		System.out.println("No se pudo leer del archivo");
		e.printStackTrace();
	}

}
```

Se debe crear su archivo de mapeo y relacionar el alias **profesor_proyecto_investigacion** con la clase **ProfesorProyectoInvestigacion**.

sql_configuracion.xml

```
<typeAliases>
	<typeAlias type="uam.bases.mybatis.clases.Area" alias="area"></typeAlias>
	<typeAlias type="uam.bases.mybatis.clases.Profesor"
	alias="profesor"></typeAlias>
	<typeAlias type="uam.bases.mybatis.clases.ProyectoInvestigacion"
	alias="proyecto_investigacion"></typeAlias>
	<typeAlias type="uam.bases.mybatis.clases.ProfesorProyectoInvestigacion"
	alias="profesor_proyecto_investigacion"></typeAlias>
</typeAliases>

<mappers>
	<mapper resource="uam/bases/mybatis/mapeos/MapeoAreas.xml"></mapper>
	<mapper resource="uam/bases/mybatis/mapeos/MapeoProfesor.xml"></mapper>
	<mapper
	resource="uam/bases/mybatis/mapeos/MapeoProyectoInvestigacion.xml"></mapper
	>
	<mapper
	resource="uam/bases/mybatis/mapeos/MapeoProfesorProyectoInvestigacion.xml">
	</mapper>
</mappers>
```

Mapeo de la Relación entre Profesor y Proyecto de Investigación

Se creará el archivo para mapear las operaciones del profesor con el proyecto de investigación respetando la ruta (paquete) y nombre establecido en el archivo de configuración.

MapeoProfesorProyectoInvestigacion.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE mapper PUBLIC "-//mybatis.org/DTD Config 3.0//EN"
"http://mybatis.org/dtd/mybatis-3-mapper.dtd">

<mapper
namespace="uam.bases.mybatis.servicios.ServiciosProfesorProyectoInvestigacion">

	<insert id="insertar" parameterType="profesor">
		INSERT INTO profesor_proyecto_investigacion
(rol,profesor_numero_economico,proyecto_investigacion_id_proyecto_investigacion)
		 VALUES
(#{rol},#{profesor_numero_economico},#{proyecto_investigacion_id_proyecto_investi
gacion})
	</insert>

</mapper>
```