Modelo de Diseño

Vesta Risk Manager

T-Code

Agustín Collareda, Cintia Hernandez, Hugo Frey



El propósito del Modelo de Diseño es empezar a realizar los casos de uso desarrollados durante las etapas anteriores en especial la etapa de captura de requerimientos. Es decir, tomar el Modelo de Casos de Uso y las Especificaciones Suplementarias creadas con anterioridad entre otros insumos y generar un modelo de diseño que pueda ser usado por los desarrolladores durante la etapa de implementación.



Tabla de contenido

[Introducción 5](#_Toc180685813)

[Propósito 5](#_Toc180685814)

[Alcance 5](#_Toc180685815)

[Referencias 5](#_Toc180685816)

[Visión general 6](#_Toc180685817)

[Diseño de Casos de Uso 6](#_Toc180685818)

[Caso de Uso 1: Autentificarse 6](#_Toc180685819)

[Especificación de caso de uso de diseño 6](#_Toc180685820)

[Diagrama de paquetes 6](#_Toc180685821)

[Diagrama de Interacción 7](#_Toc180685822)

[Requerimientos especiales o de implementación 7](#_Toc180685823)

[Caso de Uso 2: Administrar acceso al sistema 7](#_Toc180685824)

[Especificación de caso de uso de diseño 7](#_Toc180685825)

[Diagrama de paquetes 8](#_Toc180685826)

[Diagrama de Interacción 10](#_Toc180685827)

[Caso de Uso 3: Administrar proyectos 12](#_Toc180685828)

[Especificación de caso de uso de diseño 12](#_Toc180685829)

[Diagrama de paquetes 12](#_Toc180685830)

[Diagrama de Interacción 14](#_Toc180685831)

[Caso de Uso 4: Añadir riesgo a la lista 16](#_Toc180685832)

[Especificación de caso de uso de diseño 16](#_Toc180685833)

[Diagrama de paquetes 16](#_Toc180685834)

[Diagrama de Interacción 18](#_Toc180685835)

[Caso de Uso 5: Modificar lista de riesgos 18](#_Toc180685836)

[Especificación de caso de uso de diseño 18](#_Toc180685837)

[Diagrama de paquetes 19](#_Toc180685838)

[Diagrama de Interacción 21](#_Toc180685839)

[Caso de Uso 6: Administrar categorías de riesgos 22](#_Toc180685840)

[Especificación de caso de uso de diseño 22](#_Toc180685841)

[Diagrama de paquetes 22](#_Toc180685842)

[Diagrama de Interacción 23](#_Toc180685843)

[Caso de Uso 7: Realizar evaluación de riesgo 24](#_Toc180685844)

[Especificación de caso de uso de diseño 24](#_Toc180685845)

[Diagrama de paquetes 24](#_Toc180685846)

[Diagrama de Interacción 25](#_Toc180685847)

[Caso de Uso 8: Añadir plan de acción 25](#_Toc180685848)

[Diagrama de paquetes 25](#_Toc180685849)

[Diagrama de Interacción 27](#_Toc180685850)

[Caso de Uso 9: Modificar plan de acción 27](#_Toc180685851)

[Especificación de caso de uso de diseño 27](#_Toc180685852)

[Diagrama de paquetes 28](#_Toc180685853)

[Diagrama de Interacción 29](#_Toc180685854)

[Caso de Uso 10: Realizar informes 30](#_Toc180685855)

[Especificación de caso de uso de diseño 30](#_Toc180685856)

[Diagrama de paquetes 30](#_Toc180685857)

[Diagrama de Interacción 34](#_Toc180685858)

[Caso de Uso 11: Exportar archivos 36](#_Toc180685859)

[Especificación de caso de uso de diseño 36](#_Toc180685860)

[Diagrama de paquetes 36](#_Toc180685861)

[Diagrama de Interacción 37](#_Toc180685862)

[Caso de Uso 12: Realizar análisis de riesgo 37](#_Toc180685863)

[Especificación de caso de uso de diseño 37](#_Toc180685864)

[Diagrama de paquetes 38](#_Toc180685865)

[Estas clases están vinculadas al subsistema de Riesgos.Diagrama de Interacción 40](#_Toc180685866)

[Diagramas 41](#_Toc180685867)

[Diagrama de componentes 41](#_Toc180685868)

[Diagrama de Clases 42](#_Toc180685869)

[Diagramas de Paquetes 43](#_Toc180685870)

Modelo de Diseño

Introducción

El presente documento detalla el modelo de diseño para nuestro proyecto, Vesta Risk Manger. Este documento es una pieza clave en el proceso de desarrollo, ya que proporciona una visión integral de la arquitectura, componentes y estructura general del sistema que estamos construyendo. Además, este documento actúa como una guía invaluable durante la fase de implementación. Proporciona a los desarrolladores una hoja de ruta clara, reduciendo la posibilidad de malentendidos y errores en la construcción del sistema.

Propósito

El propósito principal de este documento es proporcionar una visión clara y completa del diseño de Vesta Risk Manager. Sirve como un punto de referencia para todos los miembros del equipo de desarrollo, diseñadores, y stakeholders involucrados en el proyecto.

Alcance

Este documento de Modelo de Diseño abarca la totalidad de Vesta Risk Manager, incluyendo todos sus componentes, módulos y funcionalidades. El alcance se extiende a la arquitectura general del sistema, el diseño de la interfaz de usuario, la estructura de la base de datos, los componentes del back-end, las integraciones con sistemas externos, los flujos de datos y procesos principales, así como las consideraciones de seguridad y rendimiento.

Referencias

Los documentos a los que se le hace referencia en este documento son:

* Propuesta de desarrollo.
* Especificación de requerimientos.
* Modelo de casos de uso.
* Modelo de datos.
* Arquitectura del sistema.

Estos documentos fueron realizados por el equipo de T-Code.

Visión general

Diseño de Casos de Uso

Caso de Uso 1: Autentificarse

### Especificación de caso de uso de diseño

|  |  |  |
| --- | --- | --- |
| **CASO DE USO N°1** | **Autentificarse** | |
| Actor Principal | Administrador del sistema; Líder del proyecto; Desarrollador | |
| Precondiciones | El actor ingresa a la página index del sistema | |
| Disparador | El actor selecciona “iniciar sesión” en la página de index | |
| DESCRIPCION | Paso | Acción |
|  | 1 | El actor inicia sesión con su cuenta de Google. |
| 2 | La InterfazUsuariosolicita *iniciarSesion(googleUser)* al GestorUsuario. |
| 3 | El GestorUsuario envía el evento *verificarUsuario(email)* a Usuario, Usuario solicita *verificarUsuario(email)* a la BDD, la BDD verifica el usuario (3a). El GestorUsuario envía el evento *redirigir(“inicio”)* a la InterfazUsuario. La InterfazUsuario redirige al actor a la página de inicio del sistema. |
| EXTENSIONES | Paso | Acción de Derivación |
|  | 3a | El actor no se encuentra registrado en la base de datos. El GestorUsuario envía el evento *redirigir(“/”, mensajeError)* a la InterfazUsuario. La interfazUsuario redirige al actor a la página index del sistema y muestra un mensaje de error. |

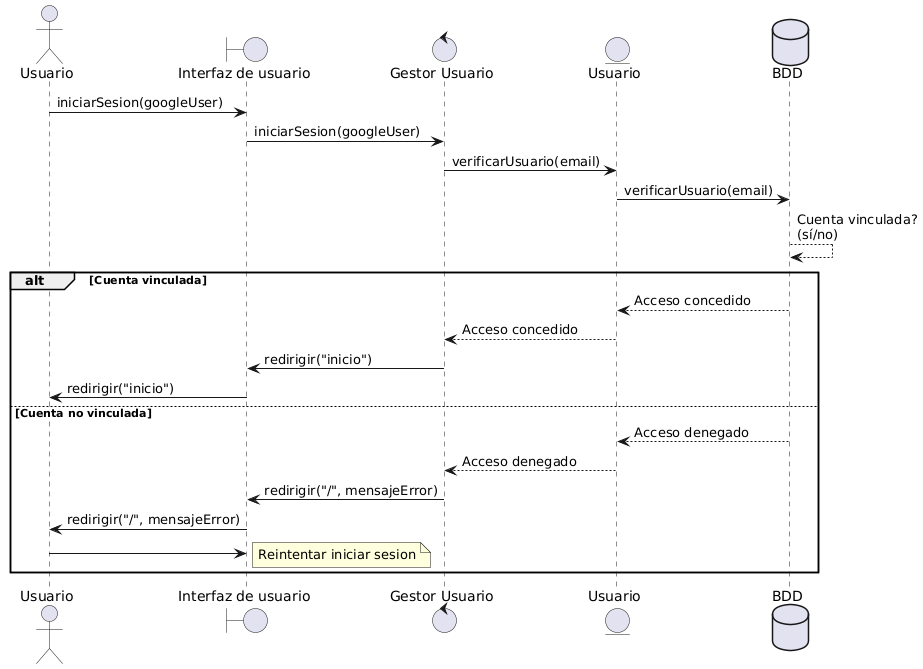
Diagrama de paquetes

Los objetos que fueron identificados en este caso de uso fueron:

* GestorUsuario.
  + Tendrá los métodos: Constructor, retornar un usuario por correo.
  + Comentario: El método obtener usuario tendrá que contar con un método privado que valide que se está pasando un correo.
* Usuario.
  + Tendrá los atributos de nombre, correo y conexión.
  + Tendrá los métodos: Constructor, los métodos get y set de cada atributo excepto para conexión, retornar un usuario por correo.

Estas clases están vinculada al subsistema de UARGflow.

Diagrama de Interacción



**Fig. 1.1**: Diagrama de secuencia CU01.

Requerimientos especiales o de implementación

* **(RNF04):** El sistema deberá ser integrado con UARGflow para el inicio de sesión de los usuarios.

Caso de Uso 2: Administrar acceso al sistema

### Especificación de caso de uso de diseño

|  |  |  |
| --- | --- | --- |
| **CASO DE USO N°2** | **Administrar acceso al sistema** | |
| Actor Principal | Administrador | |
| Precondiciones | El usuario inicio sesión en el sistema | |
| Disparador | El usuario selecciona una de las opciones de administración de acceso | |
| DESCRIPCION | Paso | Acción |
|  | 1 | El actor ingresa a la lista de usuarios. |
|  | 2 | La InterfazUsuario solicita *listarUsuarios* al GestorUsuario. El GestorUsuario envía el evento *listarUsuarios* a Usuario. Usuario solicita *listarUsuarios* a la BDD. |
|  | 3 | El GestorUsuario envía el evento *redirigir(“/usuarios”)* a la InterfazUsuario. La InterfazUsuario redirige al actor a la página /usuarios. |
|  | 4 | El actor selecciona “Registrar usuario” (4a). El actor selecciona “Eliminar usuario” (4b). El actor selecciona “Editar Usuario” (4c). |
| EXTENSIONES | Paso | Acción de Derivación |
|  | 4a | 1. La InterfazUsuario redirige al actor a la página /usuarios/crear. 2. El actor ingresa los datos del usuario a *registrar (nombre, email, roles)* en el formulario de la página y selecciona confirmar. Si selecciona cancelar, la InterfazUsuario redirige al actor a /usuarios. 3. La InterfazUsuario solicita *crearUsuario(datos)* a GestorUsuario. El GestorUsuario envía el evento *crearUsuario(datos)* a Usuario. Usuario solicita *crearUsuario(datos)* a la BDD. La BDD carga el usuario. La InfertazUsuario redirige al actor a la página /Usuarios. |
|  | 4b | 1. El actor selecciona el usuario a eliminar. 2. La InterfazUsuario redirige al actor a la página /usuarios/eliminar con la id del usuario seleccionado y solicita confirmación. 3. El actor confirma la eliminación del usuario. Si cancela la eliminación, la InterfazUsuario redirige al actor a /usuarios. 4. El GestorUsuario envía el evento *eliminarUsuario(idUsuario)* a Usuario. Usuario solicita *eliminarUsuario(idUsuario)* a la BDD. La BDD elimina el usuario. 5. La InterfazUsuario redirige al actor a /usuarios. |
|  | 4c | 1. El actor selecciona el usuario a modificar. 2. La InterfazUsuario redirige al actor a la pagina /usuarios/editar con la id del usuario seleccionado. 3. El actor modifica los datos del usuario seleccionado (nombre, email, roles) en el formulario de la página y selecciona confirmar. Si selecciona cancelar, la InterfazUsuario redirige al actor a /usuarios. 4. La InterfazUsuario solicita *editarUsuario(datos)* a GestorUsuario. El GestorUsuario envia el evento *editarUsuario(datos)* a Usuario. Usuario solicita *editarUsuario(datos)* a la BDD. La BDD modifica el usuario. La InterfazUsuario redirige al actor a /usuarios. |

Diagrama de paquetes

Los objetos que fueron identificados en este caso de uso fueron:

* GestorUsuario.
  + Tendrá los métodos: Constructor, retornar un usuario por correo, retornar todos los usuarios, modificar un usuario, eliminar un usuario, actualizar perfil de usuario, actualizar permiso de perfil, añadir perfil, eliminar perfil.
  + Comentario: El método obtener usuario tendrá que contar con un método privado que valide que se está pasando un correo, Los métodos que se repiten entre usuario y gestión de usuario es para formatear y respetar con la modularización del sistema.
* Usuario.
  + Tendrá los atributos de nombre, correo, conexión y perfil.
  + Tendrá los métodos: Constructor, los métodos get y set de cada atributo excepto para conexión, retornar un usuario por correo, retornar todos los usuarios, modificar un usuario, eliminar un usuario, actualizar perfil de usuario.
* Perfil.
  + Tendrá el atributo nombre, conexión y permisos.
  + Tendrá los métodos: Constructor, método get y set de cada atributo excepto para la conexión, obtener todos los perfiles, obtener un perfil por id, eliminar por id el perfil, actualizar por id el perfil, añadir perfil, actualizar permisos en perfil.
* Permiso.
  + Tendrá el atributo nombre y conexión.
  + Tendrá el método: Constructor, obtener todos los permisos, obtener permisos de usuario.
  + Comentario: Si bien esta clase no aparece en el diagrama de secuencia, esta de manera implícita cuando se habla de administrador, usuario, etc.

Estas clases están vinculadas al subsistema de UARGflow.

Diagrama de Interacción

Diagrama

Descripción generada automáticamente

**Fig. 1.2**: Diagrama de secuencia CU02.

Caso de Uso 3: Administrar proyectos

### Especificación de caso de uso de diseño

|  |  |  |
| --- | --- | --- |
| **CASO DE USO N°3** | **Administrar proyectos** | |
| Actor Principal | Administrador | |
| Precondiciones | El usuario inicio sesión en el sistema | |
| Disparador | El usuario selecciona una de las opciones de administración de proyectos | |
| DESCRIPCION | Paso | Acción |
|  | 1 | El actor ingresa a la InterfazProyectos. |
|  | 2 | La InterfazProyectos solicita *consultarOpciones* al GestorProyectos. |
|  | 3 | El GestorProyectos retorna las opciones disponibles (Crear/Modificar) a la InterfazProyectos. |
|  | 4 | El actor selecciona "Crear proyecto" (4a). El actor selecciona "Modificar proyecto" (4b). |
| EXTENSIONES | Paso | Acción de Derivación |
|  | 4a | 1. La InterfazProyectos solicita el formulario al GestorProyectos. 2. El GestorProyectos genera y despliega el formulario con los campos: nombre, descripción, fecha de inicio, fecha de finalización, estado y participantes. 3. El actor ingresa los datos solicitados en el formulario. 4. La InterfazProyectos envía los datos al GestorProyectos. 5. El GestorProyectos vincula los datos con la entidad Proyecto y Usuario. 6. Las entidades envían los cambios en la BDD y retornan mensajes de confirmación. 7. El actor selecciona "Confirmar". Si selecciona "Cancelar", la InterfazProyectos cierra el mensaje. 8. La BDD vincula los datos con Proyecto, guarda la información y envía mensajes de confirmación. 9. Las entidades notifican al GestorProyectos la creación correcta. 10. La InterfazProyectos muestra al actor el mensaje de creación correcta. |
|  | 4b | 1. La InterfazProyectos solicita la lista de proyectos al GestorProyectos. 2. El GestorProyectos retorna y despliega la lista de proyectos disponibles. 3. El actor selecciona el proyecto a modificar. 4. La InterfazProyectos solicita el formulario al GestorProyectos. 5. El GestorProyectos genera y despliega el formulario con los datos actuales del proyecto. 6. El actor modifica los datos deseados. 7. La InterfazProyectos envía los datos modificados a la BDD. 8. La BDD vincula los datos con Proyecto, guarda los cambios y envía mensajes de confirmación. 9. Las entidades notifican al GestorProyectos la modificación correcta. 10. La InterfazProyectos muestra al actor el mensaje de modificación correcta. 11. Si el actor selecciona "Cancelar" en cualquier momento, la InterfazProyectos cierra el mensaje. |

Diagrama de paquetes

Los objetos que fueron identificados en este caso de uso fueron:

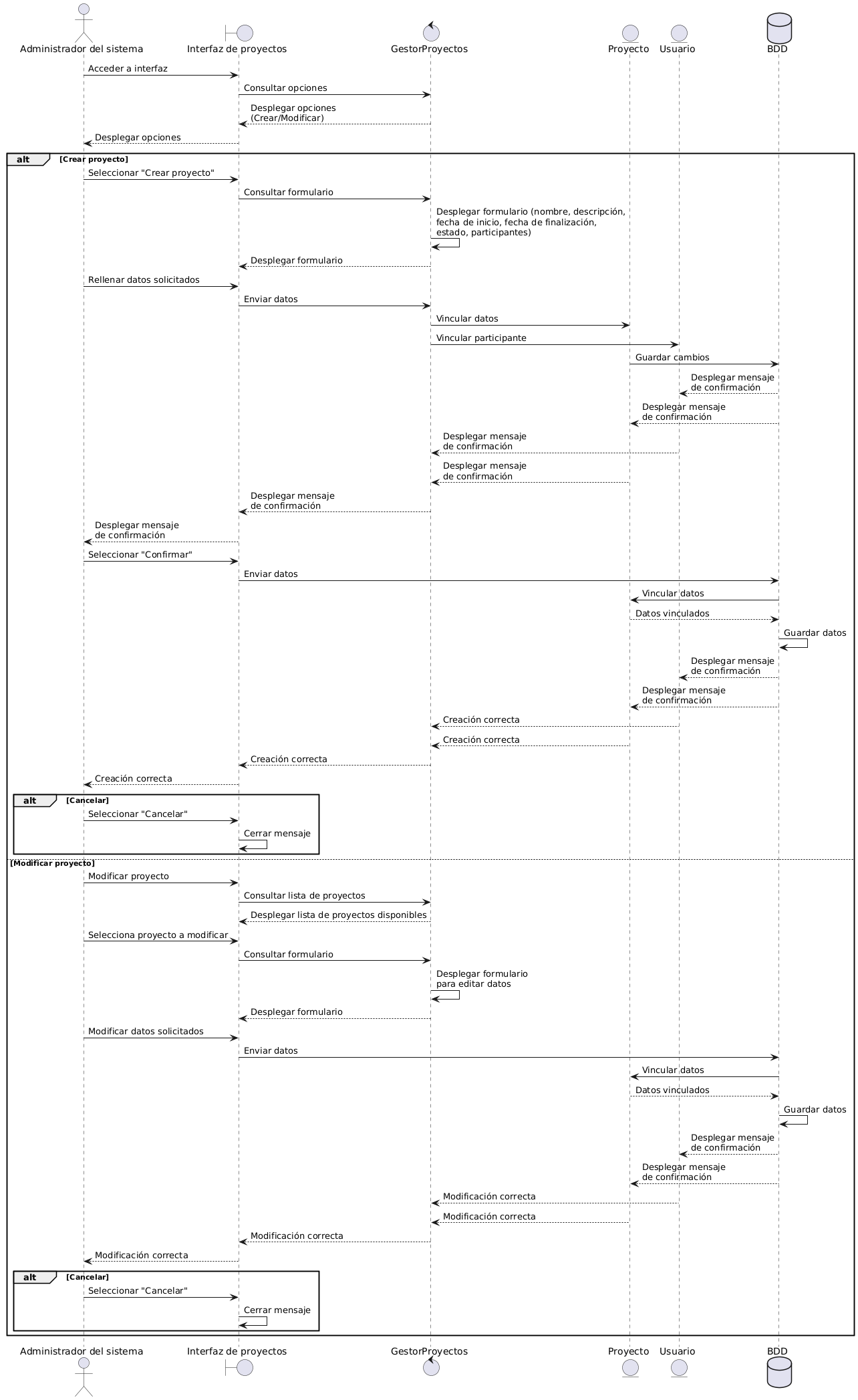
* GestorProyecto.
  + Tendrá los métodos: Constructor, retornar un proyecto por id, retornar proyecto por nombre, crear proyecto, actualizar proyecto, vincular o desvincular iteraciones, obtener todos los proyectos, vincular o desvincular participantes, obtener iteraciones por id de proyecto, actualizar iteración por id, eliminar iteración por id, añadir iteración.
  + Comentario: Cada método deberá comprobar que no se pase ningún campo vacío, ni ninguna inyección sql.
* Proyecto.
  + Tendrá los atributos de nombre, descripción, estado, fecha inicio, fecha fin, iteraciones, participantes y conexión.
  + Tendrá los métodos: Constructor, los métodos get y set de cada atributo excepto para conexión, retornar un proyecto por id, retornar proyecto por nombre, crear proyecto, actualizar proyecto por id, actualizar iteraciones, obtener todos los proyectos, vincular o desvincular participantes.
* Iteracion.
  + Tendrá los atributos fecha inicio, fecha fin y conexión.
  + Tendrá los métodos: Constructor, método get y set de cada atributo excepto para la conexión, obtener iteraciones por id de proyecto, actualizar iteración por id, eliminar iteración por id, añadir iteración.

Estas clases están vinculadas al subsistema de Proyectos.

* GestorUsuario.
  + Tendrá el método: Constructor, obtener todos los usuarios.
* Usuario
  + Tendrá los atributos nombre, correo y conexión.
  + Tendrá los métodos: Constructor, obtener todos los usuarios.

Estas clases están vinculadas al subsistema de UARGflow.

Diagrama de Interacción



**Fig. 1.3**: Diagrama de secuencia CU03.

Caso de Uso 4: Añadir riesgo a la lista

### Especificación de caso de uso de diseño

|  |  |  |
| --- | --- | --- |
| **CASO DE USO N°** | **<Nombre CU>** | |
| Actor Principal | <Actores> | |
| Precondiciones | <Condiciones que se deben cumplir antes de llamar al CU> | |
| Disparador | <Accion que inicia el CU> | |
| DESCRIPCION | Paso | Acción |
|  |  | El actor selecciona “Añadir riesgo”. |
|  |  | La InterfazUsuario redirige al actor a la página /proyecto/riesgo/crear con el id del riesgo seleccionado. |
|  |  | El actor ingresa los datos del riesgo (descripción, responsable, categoría, fechaCreacion) en el formulario de la página. El actor selecciona confirmar. Si selecciona cancelar, la InterfazUsuario redirige al actor a la página /proyecto/riesgo. |
|  |  | La InterfazUsuario solicita *crearRiesgo(datos)* a GestorRiesgo. El GestorRiesgo envia el evento *crearRiesgo(datos)* a Riesgo. Riesgo solicita *crearRiesgo(datos)* a la BDD. La BDD carga el riesgo. La InterfazUsuario redirige al actor a /proyecto/riesgo. |
| EXTENSIONES | Paso | Acción de Derivación |
|  |  |  |

Diagrama de paquetes

Los objetos que fueron identificados en este caso de uso fueron:

* GestorRiesgo.
  + Tendrá los métodos: Constructor, agregar riesgo, obtener todos los riesgos de un proyecto, obtener riesgo por id, actualizar riesgo por id, eliminar riesgo por id, vincular o desvincular responsable, actualizar categoría de un riesgo, vincular o desvincular iteraciones, Agregar categoría, actualizar categoría, eliminar categoría, obtener todas las categorías.
* Riesgo.
  + Tendrá los atributos: descripción, factor riesgo, categoría, iteración y conexión.
  + Tendrá los métodos: Constructor, los métodos get y set de cada atributo excepto para conexión, agregar riesgo, obtener todos los riesgos de un proyecto, obtener riesgo por id, actualizar riesgo por id, eliminar riesgo por id, actualizar categoría de un riesgo, vincular o desvincular proyecto, vincular o desvincular iteraciones.
* Categoria
  + Tendrá los atributos: nombre, descripción y conexión.
  + Tendrá los métodos: Constructor, los métodos get y set de cada atributo excepto para conexión, Agregar categoría, actualizar categoría, eliminar categoría, obtener todas las categorías.
  + Comentario: Esta clase se utiliza de manera implícita en este diagrama de secuencia.

Estas clases están vinculadas al subsistema de Riesgos.

* GestorProyecto.
  + Tendrá los métodos: Constructor, retornar un proyecto por id, retornar proyecto por nombre, crear proyecto, actualizar proyecto, vincular o desvincular iteraciones, obtener todos los proyectos, vincular o desvincular participantes, obtener iteraciones por id de proyecto, actualizar iteración por id, eliminar iteración por id, añadir iteración.
  + Comentario: Cada método deberá comprobar que no se pase ningún campo vacío, ni ninguna inyección sql.
* Proyecto.
  + Tendrá los atributos de nombre, descripción, estado, fecha inicio, fecha fin, iteraciones, participantes y conexión.
  + Tendrá los métodos: Constructor, los métodos get y set de cada atributo excepto para conexión, retornar un proyecto por id, retornar proyecto por nombre, crear proyecto, actualizar proyecto, actualizar iteraciones, obtener todos los proyectos, vincular o desvincular participantes.
* Iteracion.
  + Tendrá los atributos fecha inicio, fecha fin y conexión.
  + Tendrá los métodos: Constructor, método get y set de cada atributo excepto para la conexión, obtener iteraciones por id de proyecto, actualizar iteración por id, eliminar iteración por id, añadir iteración.

Estas clases están vinculadas al subsistema de Proyectos.

* GestorUsuario.
  + Tendrá el método: Constructor, obtener todos los usuarios.
* Usuario
  + Tendrá los atributos nombre, correo y conexión.
  + Tendrá los métodos: Constructor, obtener todos los usuarios.

Estas clases están vinculadas al subsistema de UARGflow.

Diagrama de Interacción

Diagrama, Esquemático

Descripción generada automáticamente

**Fig. 1.4**: Diagrama de secuencia CU04.

Caso de Uso 5: Modificar lista de riesgos

### Especificación de caso de uso de diseño

|  |  |  |
| --- | --- | --- |
| **CASO DE USO N°** | **<Nombre CU>** | |
| Actor Principal | <Actores> | |
| Precondiciones | <Condiciones que se deben cumplir antes de llamar al CU> | |
| Disparador | <Accion que inicia el CU> | |
| DESCRIPCION | Paso | Acción |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
| EXTENSIONES | Paso | Acción de Derivación |
|  |  |  |

Diagrama de paquetes

Los objetos que fueron identificados en este caso de uso fueron:

* GestorRiesgo.
  + Tendrá los métodos: Constructor, agregar riesgo, obtener todos los riesgos de un proyecto, obtener riesgo por id, actualizar riesgo por id, eliminar riesgo por id, vincular o desvincular responsable, actualizar categoría de un riesgo, vincular o desvincular iteraciones, Agregar categoría, actualizar categoría, eliminar categoría, obtener todas las categorías, crear una evaluación, obtener evaluaciones de un riesgo, obtener todas las evaluaciones, actualizar evaluación por id, eliminar evaluación por id.
* Riesgo.
  + Tendrá los atributos: descripción, factor riesgo, categoría, iteración y conexión.
  + Tendrá los métodos: Constructor, los métodos get y set de cada atributo excepto para conexión, agregar riesgo, obtener todos los riesgos de un proyecto, obtener riesgo por id, actualizar riesgo por id, eliminar riesgo por id, actualizar categoría de un riesgo, vincular o desvincular proyecto, vincular o desvincular iteraciones.
* Categoria
  + Tendrá los atributos: nombre, descripción y conexión
  + Tendrá los métodos: Constructor, los métodos get y set de cada atributo excepto para conexión, Agregar categoría, actualizar categoría, eliminar categoría, obtener todas las categorías.
  + Comentario: Esta clase se utiliza de manera implícita en este diagrama de secuencia.
* Evaluacion
  + Tendrá los atributos: impacto, probabilidad y conexión
  + Tendrá los métodos: Constructor, los métodos get y set de cada atributo excepto para conexión, crear una evaluación, obtener evaluaciones de un riesgo, obtener todas las evaluaciones, actualizar evaluación por id, eliminar evaluación por id.

Estas clases están vinculadas al subsistema de Riesgos.

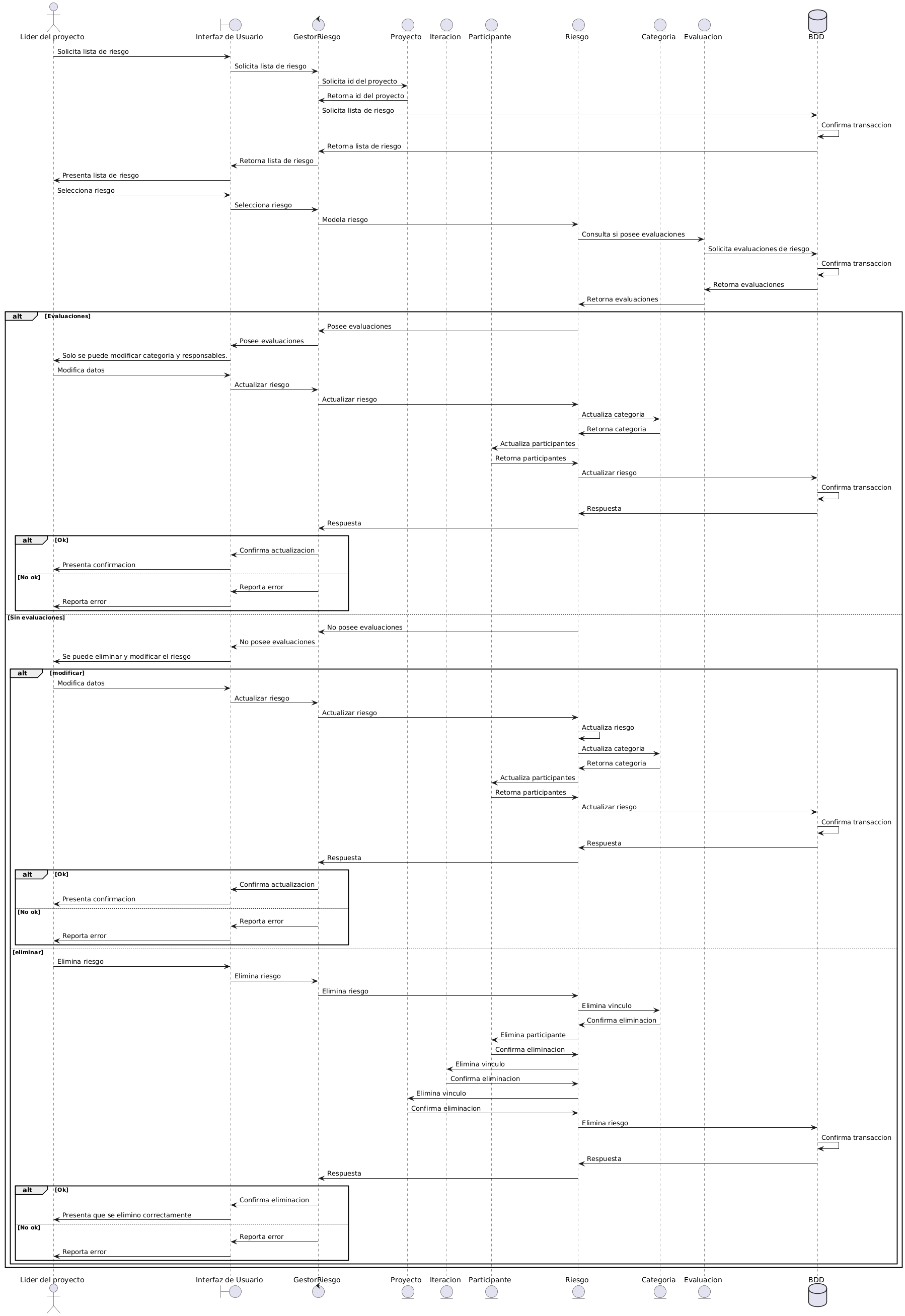
* GestorProyecto.
  + Tendrá los métodos: Constructor, retornar un proyecto por id, retornar proyecto por nombre, crear proyecto, actualizar proyecto, vincular o desvincular iteraciones, obtener todos los proyectos, vincular o desvincular participantes, obtener iteraciones por id de proyecto, actualizar iteración por id, eliminar iteración por id, añadir iteración.
  + Comentario: Cada método deberá comprobar que no se pase ningún campo vacío, ni ninguna inyección sql.
* Proyecto.
  + Tendrá los atributos de nombre, descripción, estado, fecha inicio, fecha fin, iteraciones, participantes y conexión.
  + Tendrá los métodos: Constructor, los métodos get y set de cada atributo excepto para conexión, retornar un proyecto por id, retornar proyecto por nombre, crear proyecto, actualizar proyecto, actualizar iteraciones, obtener todos los proyectos, vincular o desvincular participantes.
* Iteracion.
  + Tendrá los atributos fecha inicio, fecha fin y conexión.
  + Tendrá los métodos: Constructor, método get y set de cada atributo excepto para la conexión, obtener iteraciones por id de proyecto, actualizar iteración por id, eliminar iteración por id, añadir iteración.

Estas clases están vinculadas al subsistema de Proyectos.

* GestorUsuario.
  + Tendrá el método: Constructor, obtener todos los usuarios.
* Usuario
  + Tendrá los atributos nombre, correo y conexión.
  + Tendrá los métodos: Constructor, obtener todos los usuarios.

Estas clases están vinculadas al subsistema de UARGflow.

Diagrama de Interacción



**Fig. 1.5**: Diagrama de secuencia CU05.

Caso de Uso 6: Administrar categorías de riesgos

### Especificación de caso de uso de diseño

|  |  |  |
| --- | --- | --- |
| **CASO DE USO N°** | **<Nombre CU>** | |
| Actor Principal | <Actores> | |
| Precondiciones | <Condiciones que se deben cumplir antes de llamar al CU> | |
| Disparador | <Accion que inicia el CU> | |
| DESCRIPCION | Paso | Acción |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
| EXTENSIONES | Paso | Acción de Derivación |
|  |  |  |

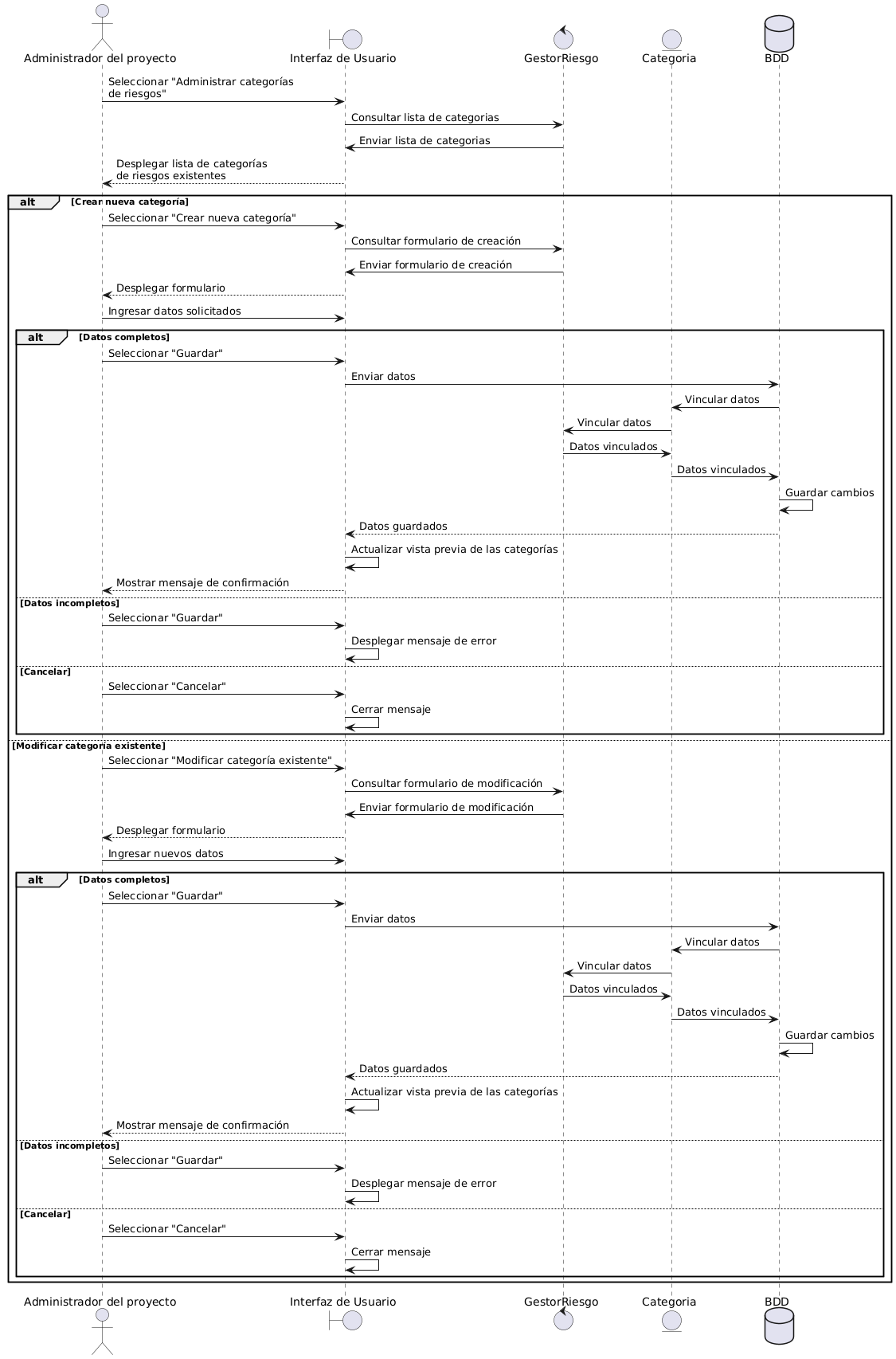
Diagrama de paquetes

Los objetos que fueron identificados en este caso de uso fueron:

* GestorRiesgo.
  + Tendrá los métodos: Constructor, agregar riesgo, obtener todos los riesgos de un proyecto, obtener riesgo por id, actualizar riesgo por id, eliminar riesgo por id, vincular o desvincular responsable, actualizar categoría de un riesgo, vincular o desvincular iteraciones, Agregar categoría, actualizar categoría, eliminar categoría, obtener todas las categorías.
* Riesgo.
  + Tendrá los atributos: descripción, factor riesgo, categoría, iteración y conexión.
  + Tendrá los métodos: Constructor, los métodos get y set de cada atributo excepto para conexión, agregar riesgo, obtener todos los riesgos de un proyecto, obtener riesgo por id, actualizar riesgo por id, eliminar riesgo por id, actualizar categoría de un riesgo, vincular o desvincular proyecto, vincular o desvincular iteraciones.
* Categoria
  + Tendrá los atributos: nombre, descripción y conexión.
  + Tendrá los métodos: Constructor, los métodos get y set de cada atributo excepto para conexión, Agregar categoría, actualizar categoría, eliminar categoría, obtener todas las categorías.

Estas clases están vinculadas al subsistema de Riesgos.

Diagrama de Interacción



**Fig. 1.6** Diagrama de secuencia CU06.

Caso de Uso 7: Realizar evaluación de riesgo

### Especificación de caso de uso de diseño

|  |  |  |
| --- | --- | --- |
| **CASO DE USO N°** | **<Nombre CU>** | |
| Actor Principal | <Actores> | |
| Precondiciones | <Condiciones que se deben cumplir antes de llamar al CU> | |
| Disparador | <Accion que inicia el CU> | |
| DESCRIPCION | Paso | Acción |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
| EXTENSIONES | Paso | Acción de Derivación |
|  |  |  |

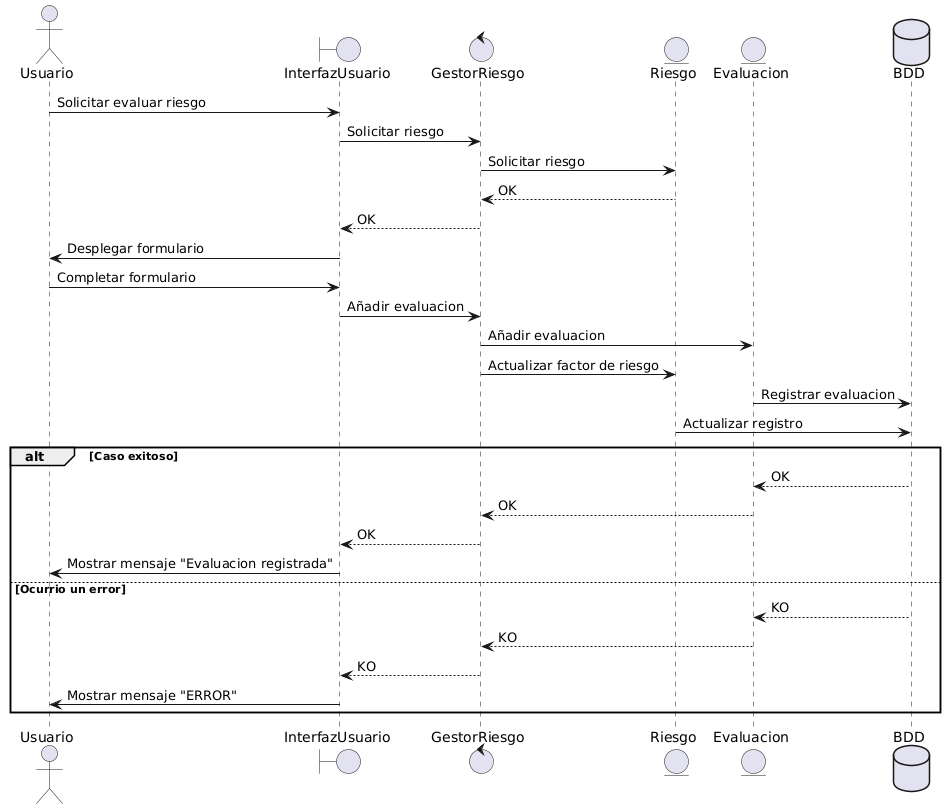
Diagrama de paquetes

Los objetos que fueron identificados en este caso de uso fueron:

* GestorRiesgo.
  + Tendrá los métodos: Constructor, agregar riesgo, obtener todos los riesgos de un proyecto, obtener riesgo por id, actualizar riesgo por id, eliminar riesgo por id, vincular o desvincular responsable, actualizar categoría de un riesgo, vincular o desvincular iteraciones, Agregar categoría, actualizar categoría, eliminar categoría, obtener todas las categorías, crear una evaluación, obtener evaluaciones de un riesgo, obtener todas las evaluaciones, actualizar evaluación por id, eliminar evaluación por id.
* Riesgo.
  + Tendrá los atributos: descripción, factor riesgo, categoría, iteración y conexión.
  + Tendrá los métodos: Constructor, los métodos get y set de cada atributo excepto para conexión, agregar riesgo, obtener todos los riesgos de un proyecto, obtener riesgo por id, actualizar riesgo por id, eliminar riesgo por id, actualizar categoría de un riesgo, vincular o desvincular proyecto, vincular o desvincular iteraciones.
* Categoria
  + Tendrá los atributos: nombre, descripción y conexión.
  + Tendrá los métodos: Constructor, los métodos get y set de cada atributo excepto para conexión, Agregar categoría, actualizar categoría, eliminar categoría, obtener todas las categorías.
  + Comentario: Esta clase se utiliza de manera implícita en este diagrama de secuencia.
* Evaluacion
  + Tendrá los atributos: impacto, probabilidad y conexión
  + Tendrá los métodos: Constructor, los métodos get y set de cada atributo excepto para conexión, crear una evaluación, obtener evaluaciones de un riesgo, obtener todas las evaluaciones, actualizar evaluación por id, eliminar evaluación por id.

Estas clases están vinculadas al subsistema de Riesgos.

Diagrama de Interacción



**Fig. 1.7**: Diagrama de secuencia CU07.

Caso de Uso 8: Añadir plan de acción

Diagrama de paquetes

Los objetos que fueron identificados en este caso de uso fueron:

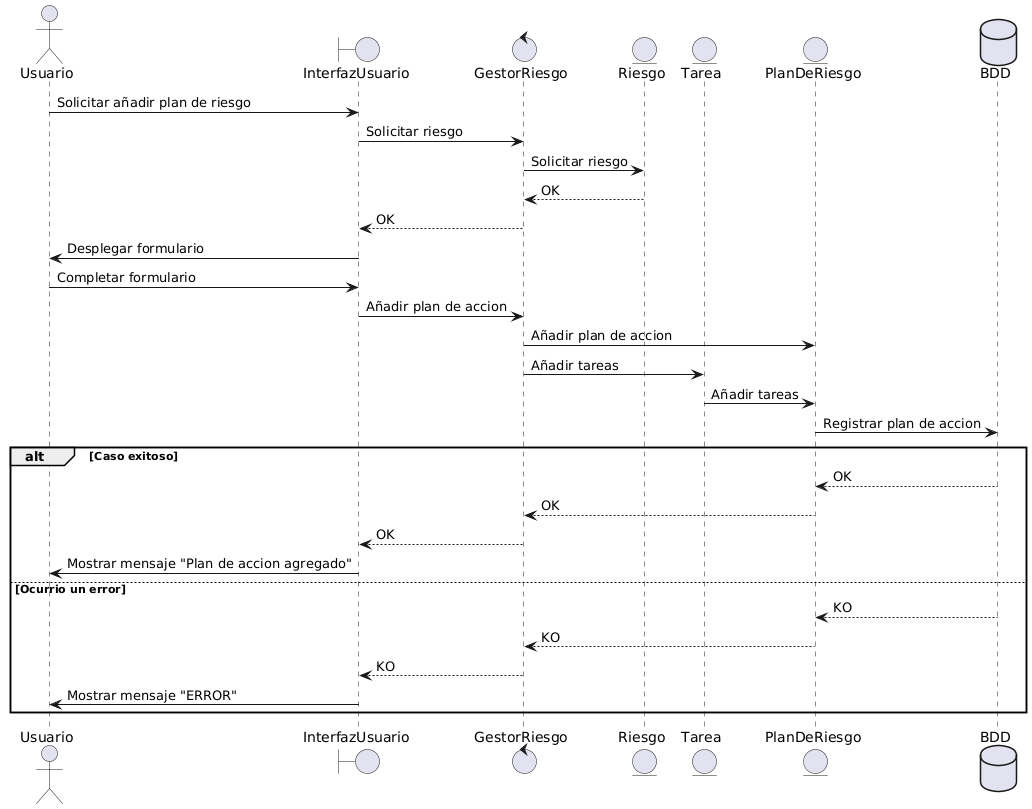
* GestorRiesgo.
  + Tendrá los métodos: Constructor, agregar riesgo, obtener todos los riesgos de un proyecto, obtener riesgo por id, actualizar riesgo por id, eliminar riesgo por id, vincular o desvincular responsable, actualizar categoría de un riesgo, vincular o desvincular iteraciones, Agregar categoría, actualizar categoría, eliminar categoría, obtener todas las categorías, crear una evaluación, obtener evaluaciones de un riesgo, obtener todas las evaluaciones, actualizar evaluación por id, eliminar evaluación por id, Crear tarea, modificar tarea por id, vincular o desvincular participante, eliminar tarea por id, Crear plan de riesgo, actualizar plan de riesgo por id, eliminar plan de riesgo por id, vincular iteración, Vincular o desvincular tareas.
* Riesgo.
  + Tendrá los atributos: descripción, factor riesgo, categoría, iteración y conexión.
  + Tendrá los métodos: Constructor, los métodos get y set de cada atributo excepto para conexión, agregar riesgo, obtener todos los riesgos de un proyecto, obtener riesgo por id, actualizar riesgo por id, eliminar riesgo por id, actualizar categoría de un riesgo, vincular o desvincular proyecto, vincular o desvincular iteraciones.
* Categoria
  + Tendrá los atributos: nombre, descripción y conexión.
  + Tendrá los métodos: Constructor, los métodos get y set de cada atributo excepto para conexión, Agregar categoría, actualizar categoría, eliminar categoría, obtener todas las categorías.
  + Comentario: Esta clase se utiliza de manera implícita en este diagrama de secuencia.
* Evaluacion
  + Tendrá los atributos: impacto, probabilidad y conexión
  + Tendrá los métodos: Constructor, los métodos get y set de cada atributo excepto para conexión, crear una evaluación, obtener evaluaciones de un riesgo, obtener todas las evaluaciones, actualizar evaluación por id, eliminar evaluación por id.
  + Comentario: Esta clase se utiliza de manera implícita en este diagrama de secuencia
* PlanDeRiesgo
  + Tendrá los atributos: nombre, descripción, tipo, tareas y conexión.
  + Tendrá los métodos: Constructor, los métodos get y set de cada atributo excepto para conexión, Crear plan de riesgo, actualizar plan de riesgo por id, eliminar plan de riesgo por id, vincular iteración, Vincular o desvincular tareas.
* Tareas:
  + Tendrá los atributos: nombre, descripción, estado, fecha inicio, fecha fin y conexión.
  + Tendrá los métodos: Constructor, los métodos get y set de cada atributo excepto para conexión, Crear tarea, modificar tarea por id, vincular o desvincular participante, eliminar tarea por id.

Estas clases están vinculadas al subsistema de Riesgos.

* GestorUsuario.
  + Tendrá el método: Constructor, obtener todos los usuarios.
* Usuario
  + Tendrá los atributos nombre, correo y conexión.
  + Tendrá los métodos: Constructor, obtener todos los usuarios.

Estas clases están vinculadas al subsistema de UARGflow.

Diagrama de Interacción



**Fig. 1.8**: Diagrama de secuencia CU08.

Caso de Uso 9: Modificar plan de acción

### Especificación de caso de uso de diseño

|  |  |  |
| --- | --- | --- |
| **CASO DE USO N°** | **<Nombre CU>** | |
| Actor Principal | <Actores> | |
| Precondiciones | <Condiciones que se deben cumplir antes de llamar al CU> | |
| Disparador | <Accion que inicia el CU> | |
| DESCRIPCION | Paso | Acción |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
| EXTENSIONES | Paso | Acción de Derivación |
|  |  |  |

Diagrama de paquetes

Los objetos que fueron identificados en este caso de uso fueron:

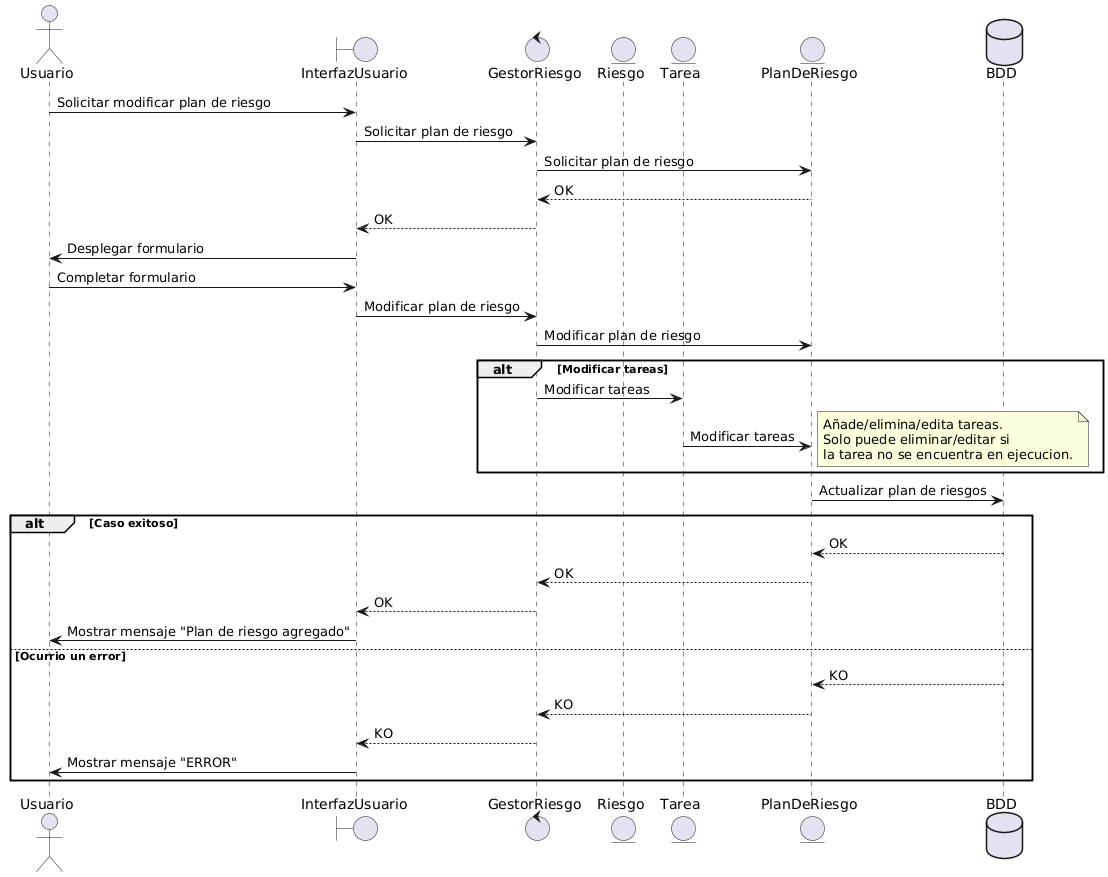
* GestorRiesgo.
  + Tendrá los métodos: Constructor, agregar riesgo, obtener todos los riesgos de un proyecto, obtener riesgo por id, actualizar riesgo por id, eliminar riesgo por id, vincular o desvincular responsable, actualizar categoría de un riesgo, vincular o desvincular iteraciones, Agregar categoría, actualizar categoría, eliminar categoría, obtener todas las categorías, crear una evaluación, obtener evaluaciones de un riesgo, obtener todas las evaluaciones, actualizar evaluación por id, eliminar evaluación por id, Crear tarea, modificar tarea por id, vincular o desvincular participante, eliminar tarea por id, Crear plan de riesgo, actualizar plan de riesgo por id, eliminar plan de riesgo por id, vincular iteración, Vincular o desvincular tareas.
* Riesgo.
  + Tendrá los atributos: descripción, factor riesgo, categoría, iteración y conexión.
  + Tendrá los métodos: Constructor, los métodos get y set de cada atributo excepto para conexión, agregar riesgo, obtener todos los riesgos de un proyecto, obtener riesgo por id, actualizar riesgo por id, eliminar riesgo por id, actualizar categoría de un riesgo, vincular o desvincular proyecto, vincular o desvincular iteraciones.
* Categoria
  + Tendrá los atributos: nombre, descripción y conexión.
  + Tendrá los métodos: Constructor, los métodos get y set de cada atributo excepto para conexión, Agregar categoría, actualizar categoría, eliminar categoría, obtener todas las categorías.
  + Comentario: Esta clase se utiliza de manera implícita en este diagrama de secuencia.
* Evaluacion
  + Tendrá los atributos: impacto, probabilidad y conexión
  + Tendrá los métodos: Constructor, los métodos get y set de cada atributo excepto para conexión, crear una evaluación, obtener evaluaciones de un riesgo, obtener todas las evaluaciones, actualizar evaluación por id, eliminar evaluación por id.
  + Comentario: Esta clase se utiliza de manera implícita en este diagrama de secuencia
* PlanDeRiesgo
  + Tendrá los atributos: nombre, descripción, tipo, tareas y conexión.
  + Tendrá los métodos: Constructor, los métodos get y set de cada atributo excepto para conexión, Crear plan de riesgo, actualizar plan de riesgo por id, eliminar plan de riesgo por id, vincular iteración, Vincular o desvincular tareas.
* Tareas:
  + Tendrá los atributos: nombre, descripción, estado, fecha inicio, fecha fin y conexión.
  + Tendrá los métodos: Constructor, los métodos get y set de cada atributo excepto para conexión, Crear tarea, modificar tarea por id, vincular o desvincular participante, eliminar tarea por id.

Estas clases están vinculadas al subsistema de Riesgos.

* GestorUsuario.
  + Tendrá el método: Constructor, obtener todos los usuarios.
* Usuario
  + Tendrá los atributos nombre, correo y conexión.
  + Tendrá los métodos: Constructor, obtener todos los usuarios.

Estas clases están vinculadas al subsistema de UARGflow.

Diagrama de Interacción



**Fig. 1.9**: Diagrama de secuencia CU09.

Caso de Uso 10: Realizar informes

### Especificación de caso de uso de diseño

|  |  |  |
| --- | --- | --- |
| **CASO DE USO N°** | **<Nombre CU>** | |
| Actor Principal | <Actores> | |
| Precondiciones | <Condiciones que se deben cumplir antes de llamar al CU> | |
| Disparador | <Accion que inicia el CU> | |
| DESCRIPCION | Paso | Acción |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
| EXTENSIONES | Paso | Acción de Derivación |
|  |  |  |

Diagrama de paquetes

Los objetos que fueron identificados en este caso de uso fueron:

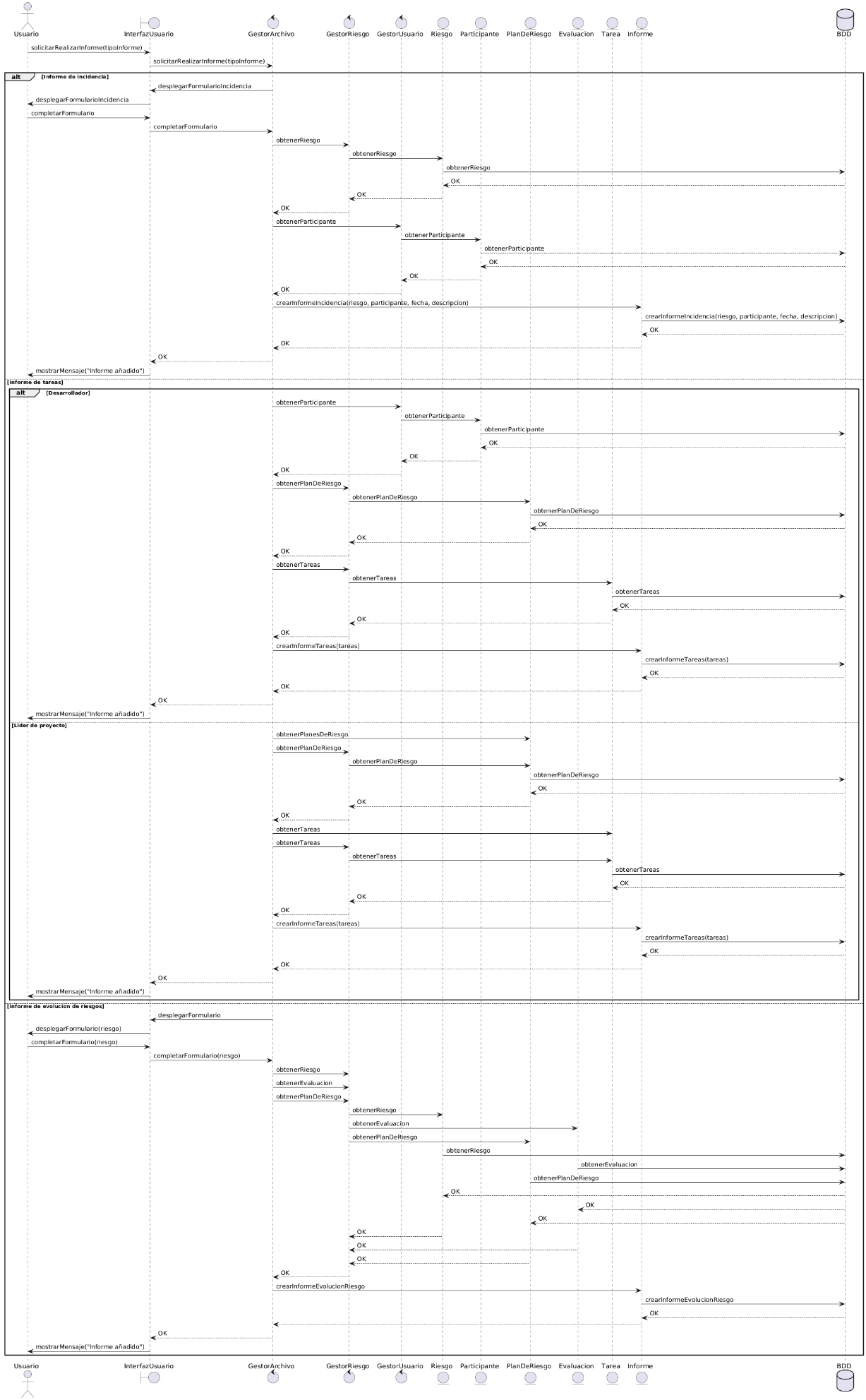
* GestorRiesgo.
  + Tendrá los métodos: Constructor, agregar riesgo, obtener todos los riesgos de un proyecto, obtener riesgo por id, actualizar riesgo por id, eliminar riesgo por id, vincular o desvincular responsable, actualizar categoría de un riesgo, vincular o desvincular iteraciones, Agregar categoría, actualizar categoría, eliminar categoría, obtener todas las categorías, crear una evaluación, obtener evaluaciones de un riesgo, obtener todas las evaluaciones, actualizar evaluación por id, eliminar evaluación por id, Crear tarea, modificar tarea por id, vincular o desvincular participante, eliminar tarea por id, Crear plan de riesgo, actualizar plan de riesgo por id, eliminar plan de riesgo por id, vincular iteración, Vincular o desvincular tareas, crear incidencia, obtener incidencias por id, obtener todas las incidencias.
* Riesgo.
  + Tendrá los atributos: descripción, factor riesgo, categoría, iteración y conexión.
  + Tendrá los métodos: Constructor, los métodos get y set de cada atributo excepto para conexión, agregar riesgo, obtener todos los riesgos de un proyecto, obtener riesgo por id, actualizar riesgo por id, eliminar riesgo por id, actualizar categoría de un riesgo, vincular o desvincular proyecto, vincular o desvincular iteraciones.
* Categoria
  + Tendrá los atributos: nombre, descripción y conexión.
  + Tendrá los métodos: Constructor, los métodos get y set de cada atributo excepto para conexión, Agregar categoría, actualizar categoría, eliminar categoría, obtener todas las categorías.
  + Comentario: Esta clase se utiliza de manera implícita en este diagrama de secuencia.
* Evaluacion
  + Tendrá los atributos: impacto, probabilidad y conexión
  + Tendrá los métodos: Constructor, los métodos get y set de cada atributo excepto para conexión, crear una evaluación, obtener evaluaciones de un riesgo, obtener todas las evaluaciones, actualizar evaluación por id, eliminar evaluación por id.
  + Comentario: Esta clase se utiliza de manera implícita en este diagrama de secuencia
* PlanDeRiesgo
  + Tendrá los atributos: nombre, descripción, tipo, tareas y conexión.
  + Tendrá los métodos: Constructor, los métodos get y set de cada atributo excepto para conexión, Crear plan de riesgo, actualizar plan de riesgo por id, eliminar plan de riesgo por id, vincular iteración, Vincular o desvincular tareas.
* Tareas:
  + Tendrá los atributos: nombre, descripción, estado, fecha inicio, fecha fin y conexión.
  + Tendrá los métodos: Constructor, los métodos get y set de cada atributo excepto para conexión, Crear tarea, modificar tarea por id, vincular o desvincular participante, eliminar tarea por id.
* Incidencia:
  + Tendrá los atributos: descripción y conexión.
  + Tendrá los métodos: Tendrá los métodos: Constructor, los métodos get y set de cada atributo excepto para conexión, crear incidencia, obtener incidencias por id de riesgo, obtener todas las incidencias.
  + Comentario: Es un tipo de informe.

Estas clases están vinculadas al subsistema de Riesgos.

* GestorArchivos.
  + Tendrá los métodos: generar un informe, obtener informe de incidencia por id, descargar archivo como pdf, descargar archivo como xslx.
* Informe
  + Tendrá los atributos: datos.
  + Tendrá los métodos: Constructor, los métodos get y set de cada uno de los atributos, generar un informe, obtener informe de incidencia por id.
* Archivo
  + Tendrá el atributo: informe.
  + Tendrá los métodos: Constructor, método get y set del atributo informe, descargar archivo como pdf, descargar archivo como xslx.

Estas clases están vinculadas al subsistema de Exportar.

Diagrama de Interacción



**Fig. 1.10**: Diagrama de secuencia CU10.

Caso de Uso 11: Exportar archivos

### Especificación de caso de uso de diseño

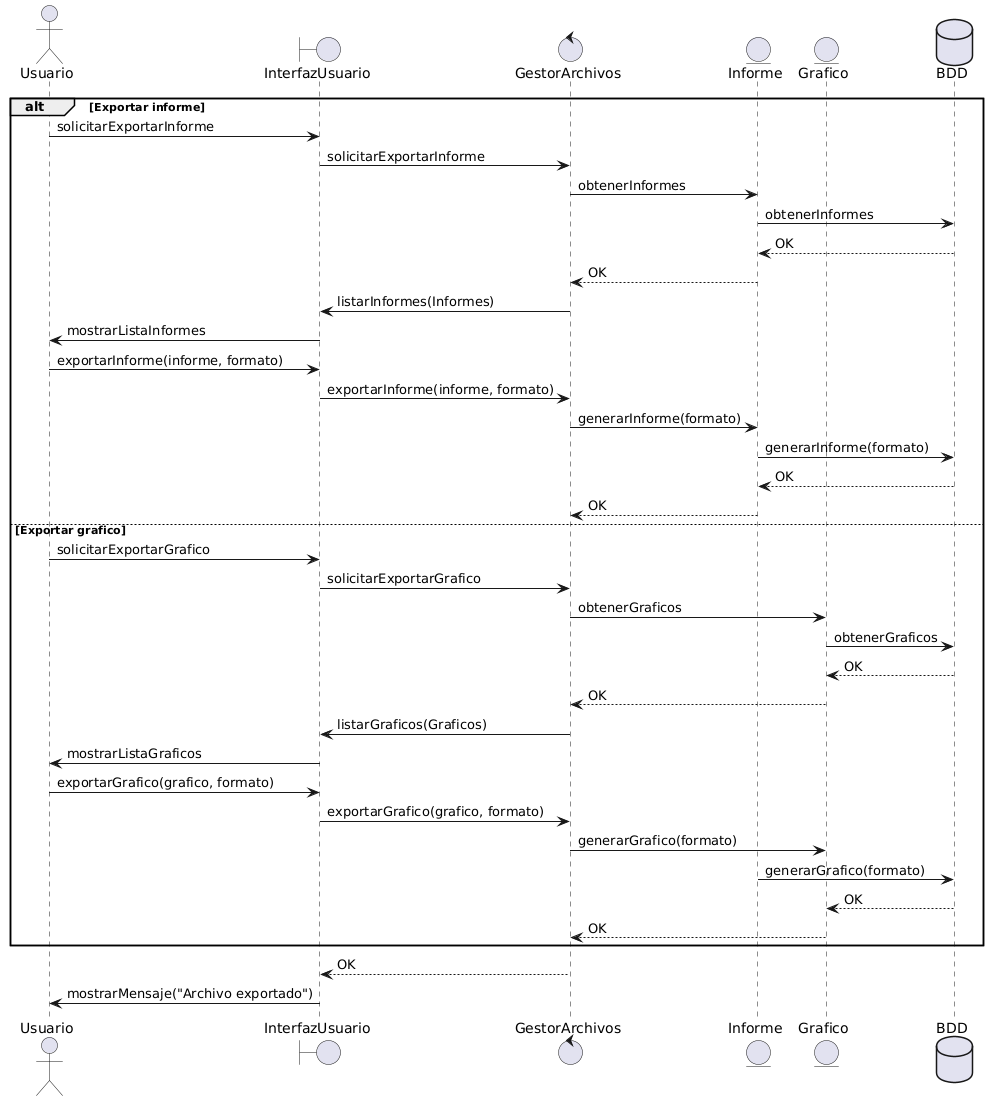
|  |  |  |
| --- | --- | --- |
| **CASO DE USO N°** | **<Nombre CU>** | |
| Actor Principal | <Actores> | |
| Precondiciones | <Condiciones que se deben cumplir antes de llamar al CU> | |
| Disparador | <Accion que inicia el CU> | |
| DESCRIPCION | Paso | Acción |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
| EXTENSIONES | Paso | Acción de Derivación |
|  |  |  |

Diagrama de paquetes

* GestorArchivos.
  + Tendrá los métodos: generar un informe, obtener informe de incidencia por id, descargar archivo como pdf, descargar archivo como xslx.
* Informe
  + Tendrá los atributos: datos.
  + Tendrá los métodos: Constructor, los métodos get y set de cada uno de los atributos, generar un informe, obtener informe de incidencia por id.
* Archivo
  + Tendrá el atributo: informe.
  + Tendrá los métodos: Constructor, método get y set del atributo informe, descargar archivo como pdf, descargar archivo como xslx.

Estas clases están vinculadas al subsistema de Exportar.

Diagrama de Interacción



**Fig. 1.11**: Diagrama de secuencia CU11.

Caso de Uso 12: Realizar análisis de riesgo

### Especificación de caso de uso de diseño

|  |  |  |
| --- | --- | --- |
| **CASO DE USO N°** | **<Nombre CU>** | |
| Actor Principal | <Actores> | |
| Precondiciones | <Condiciones que se deben cumplir antes de llamar al CU> | |
| Disparador | <Accion que inicia el CU> | |
| DESCRIPCION | Paso | Acción |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
| EXTENSIONES | Paso | Acción de Derivación |
|  |  |  |

Diagrama de paquetes

Los objetos que fueron identificados en este caso de uso fueron:

* GestorRiesgo.
  + Tendrá los métodos: Constructor, agregar riesgo, obtener todos los riesgos de un proyecto, obtener riesgo por id, actualizar riesgo por id, eliminar riesgo por id, vincular o desvincular responsable, actualizar categoría de un riesgo, vincular o desvincular iteraciones, Agregar categoría, actualizar categoría, eliminar categoría, obtener todas las categorías, crear una evaluación, obtener evaluaciones de un riesgo, obtener todas las evaluaciones, actualizar evaluación por id, eliminar evaluación por id, Crear tarea, modificar tarea por id, vincular o desvincular participante, eliminar tarea por id, Crear plan de riesgo, actualizar plan de riesgo por id, eliminar plan de riesgo por id, vincular iteración, Vincular o desvincular tareas, crear incidencia, obtener incidencias por id, obtener todas las incidencias, comprobar si riesgo necesita evaluación, comprobar si riesgo necesita un plan, comprobar si riesgo necesita dos planes.
* Riesgo.
  + Tendrá los atributos: descripción, factor riesgo, categoría, iteración y conexión.
  + Tendrá los métodos: Constructor, los métodos get y set de cada atributo excepto para conexión, agregar riesgo, obtener todos los riesgos de un proyecto, obtener riesgo por id, actualizar riesgo por id, eliminar riesgo por id, actualizar categoría de un riesgo, vincular o desvincular proyecto, vincular o desvincular iteraciones.
* Categoria
  + Tendrá los atributos: nombre, descripción y conexión.
  + Tendrá los métodos: Constructor, los métodos get y set de cada atributo excepto para conexión, Agregar categoría, actualizar categoría, eliminar categoría, obtener todas las categorías.
  + Comentario: Esta clase se utiliza de manera implícita en este diagrama de secuencia.
* Evaluacion
  + Tendrá los atributos: impacto, probabilidad y conexión
  + Tendrá los métodos: Constructor, los métodos get y set de cada atributo excepto para conexión, crear una evaluación, obtener evaluaciones de un riesgo, obtener todas las evaluaciones, actualizar evaluación por id, eliminar evaluación por id.
  + Comentario: Esta clase se utiliza de manera implícita en este diagrama de secuencia.
* PlanDeRiesgo
  + Tendrá los atributos: nombre, descripción, tipo, tareas y conexión.
  + Tendrá los métodos: Constructor, los métodos get y set de cada atributo excepto para conexión, Crear plan de riesgo, actualizar plan de riesgo por id, eliminar plan de riesgo por id, vincular iteración, Vincular o desvincular tareas.
* Tareas:
  + Tendrá los atributos: nombre, descripción, estado, fecha inicio, fecha fin y conexión.
  + Tendrá los métodos: Constructor, los métodos get y set de cada atributo excepto para conexión, Crear tarea, modificar tarea por id, vincular o desvincular participante, eliminar tarea por id.
* Incidencia:
  + Tendrá los atributos: descripción y conexión
  + Tendrá los métodos: Tendrá los métodos: Constructor, los métodos get y set de cada atributo excepto para conexión, crear incidencia, obtener incidencias por id de riesgo, obtener todas las incidencias.
  + Comentario: Es un tipo de informe.

Estas clases están vinculadas al subsistema de Riesgos.

Diagrama de Interacción

Tabla

Descripción generada automáticamente

**Fig. 1.12**: Diagrama de secuencia CU12.

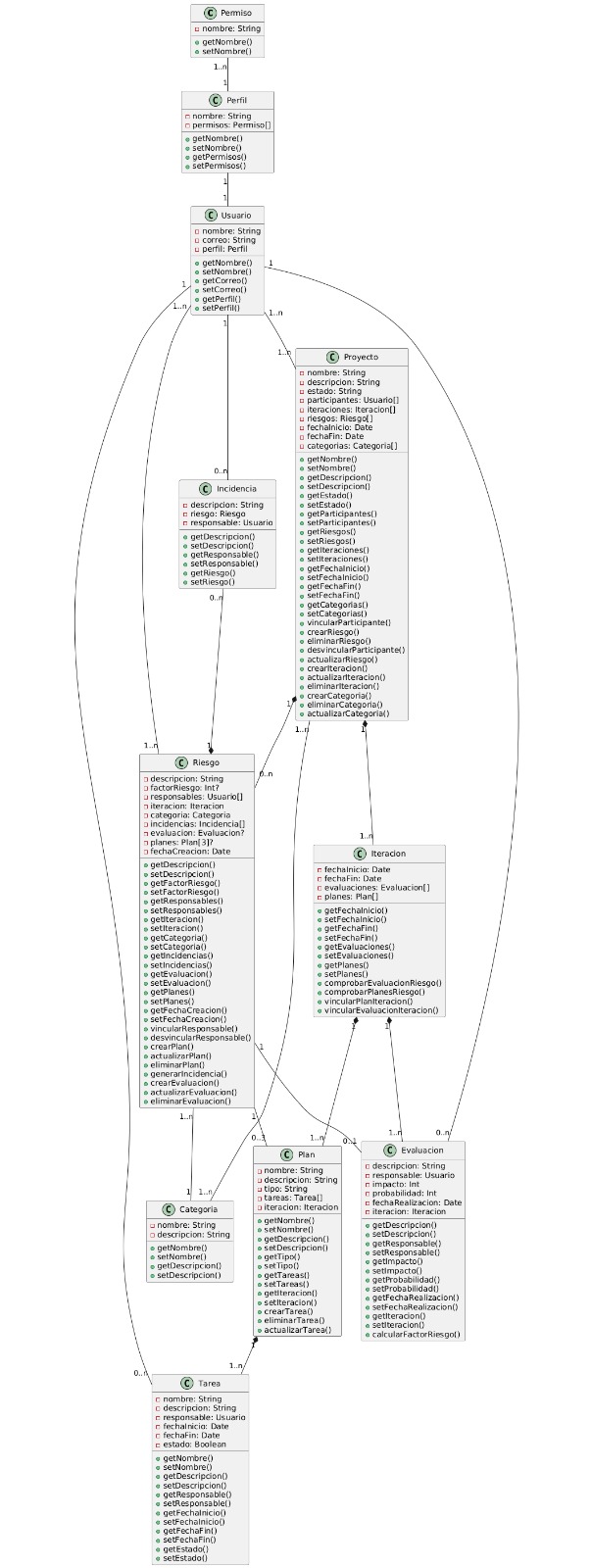
# Diagramas

Diagrama de componentes



**Fig. 2.1**: Modelo de componentes.

Diagrama de Clases



**Fig. 2.2**: Diagrama de Clases.

**Aclaracion**: Este diagrama esta pendiente a modificaciones.

Diagramas de Paquetes

  
**Fig. 2.3**: Diagrama de Paquetes.