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](#_Toc180152862)

[Propósito 5](#_Toc180152863)

[Alcance 5](#_Toc180152864)

[Referencias 5](#_Toc180152865)

[Visión general 6](#_Toc180152866)

[Diseño de Casos de Uso 6](#_Toc180152867)

[Caso de Uso 1: Autentificarse 6](#_Toc180152868)

[Diagrama de paquetes 6](#_Toc180152869)

[Diagrama de Interacción 6](#_Toc180152870)

[Requerimientos especiales o de implementación 7](#_Toc180152871)

[Caso de Uso 2: Administrar acceso al sistema 7](#_Toc180152872)

[Diagrama de paquetes 7](#_Toc180152873)

[Diagrama de Interacción 8](#_Toc180152874)

[Caso de Uso 3: Administrar proyectos 9](#_Toc180152875)

[Diagrama de paquetes 9](#_Toc180152876)

[Diagrama de Interacción 10](#_Toc180152877)

[Caso de Uso 4: Añadir riesgo a la lista 11](#_Toc180152878)

[Diagrama de paquetes 11](#_Toc180152879)

[Diagrama de Interacción 12](#_Toc180152880)

[Caso de Uso 5: Modificar lista de riesgos 12](#_Toc180152881)

[Diagrama de paquetes 12](#_Toc180152882)

[Diagrama de Interacción 15](#_Toc180152883)

[Caso de Uso 6: Administrar categorías de riesgos 16](#_Toc180152884)

[Diagrama de paquetes 16](#_Toc180152885)

[Diagrama de Interacción 17](#_Toc180152886)

[Caso de Uso 7: Realizar evaluación de riesgo 18](#_Toc180152887)

[Diagrama de paquetes 18](#_Toc180152888)

[Diagrama de Interacción 19](#_Toc180152889)

[Caso de Uso 8: Añadir plan de acción 19](#_Toc180152890)

[Diagrama de paquetes 19](#_Toc180152891)

[Diagrama de Interacción 21](#_Toc180152892)

[Caso de Uso 9: Modificar plan de acción 21](#_Toc180152893)

[Diagrama de paquetes 21](#_Toc180152894)

[Diagrama de Interacción 22](#_Toc180152895)

[Caso de Uso 10: Realizar informes 22](#_Toc180152896)

[Diagrama de paquetes 22](#_Toc180152897)

[Diagrama de Interacción 23](#_Toc180152898)

[Caso de Uso 11: Exportar archivos 23](#_Toc180152899)

[Diagrama de paquetes 23](#_Toc180152900)

[Diagrama de Interacción 24](#_Toc180152901)

[Caso de Uso 12: Realizar análisis de riesgo 24](#_Toc180152902)

[Diagrama de paquetes 24](#_Toc180152903)

[Diagrama de Interacción 25](#_Toc180152904)

[Diseño de Objetos 25](#_Toc180152905)

[[Objeto 1] 25](#_Toc180152906)

[[Objeto 2] 26](#_Toc180152907)

[Diseño de Subsistemas 26](#_Toc180152908)

[Subsistemas Específicos 26](#_Toc180152909)

[[Nombre del Subsistema Específico 1] 26](#_Toc180152910)

[Propósito 26](#_Toc180152911)

[Función 26](#_Toc180152912)

[Subordinados 26](#_Toc180152913)

[Dependencias 26](#_Toc180152914)

[Recursos 27](#_Toc180152915)

[Diagramas 28](#_Toc180152916)

[Diagrama de componentes 28](#_Toc180152917)

[Diagrama de Clases 28](#_Toc180152918)

[Diagramas de Paquetes 29](#_Toc180152919)

[Diagrama de Colaboración 29](#_Toc180152920)

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

[Esta sección debe proporcionar una lista completa de todos los documentos a los que se hace referencia en el documento Modelo de Diseño. Cada documento debe identificarse por el título, número del informe (si se aplica), fecha, y organización que lo publica. Especifique las fuentes de las que pueden obtenerse las referencias. Esta información puede proporcionarse por la referencia a un apéndice o a otro documento.]

Visión general

Diseño de Casos de Uso

Caso de Uso 1: Autentificarse

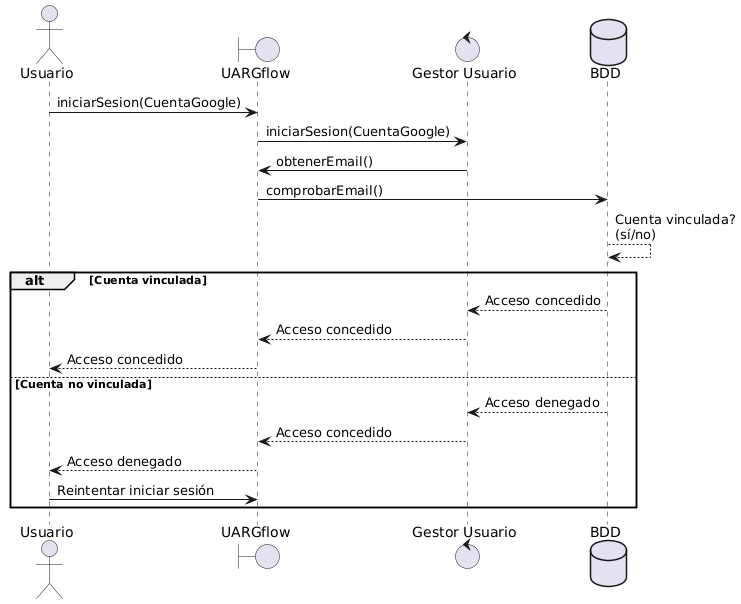
Diagrama de paquetes

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

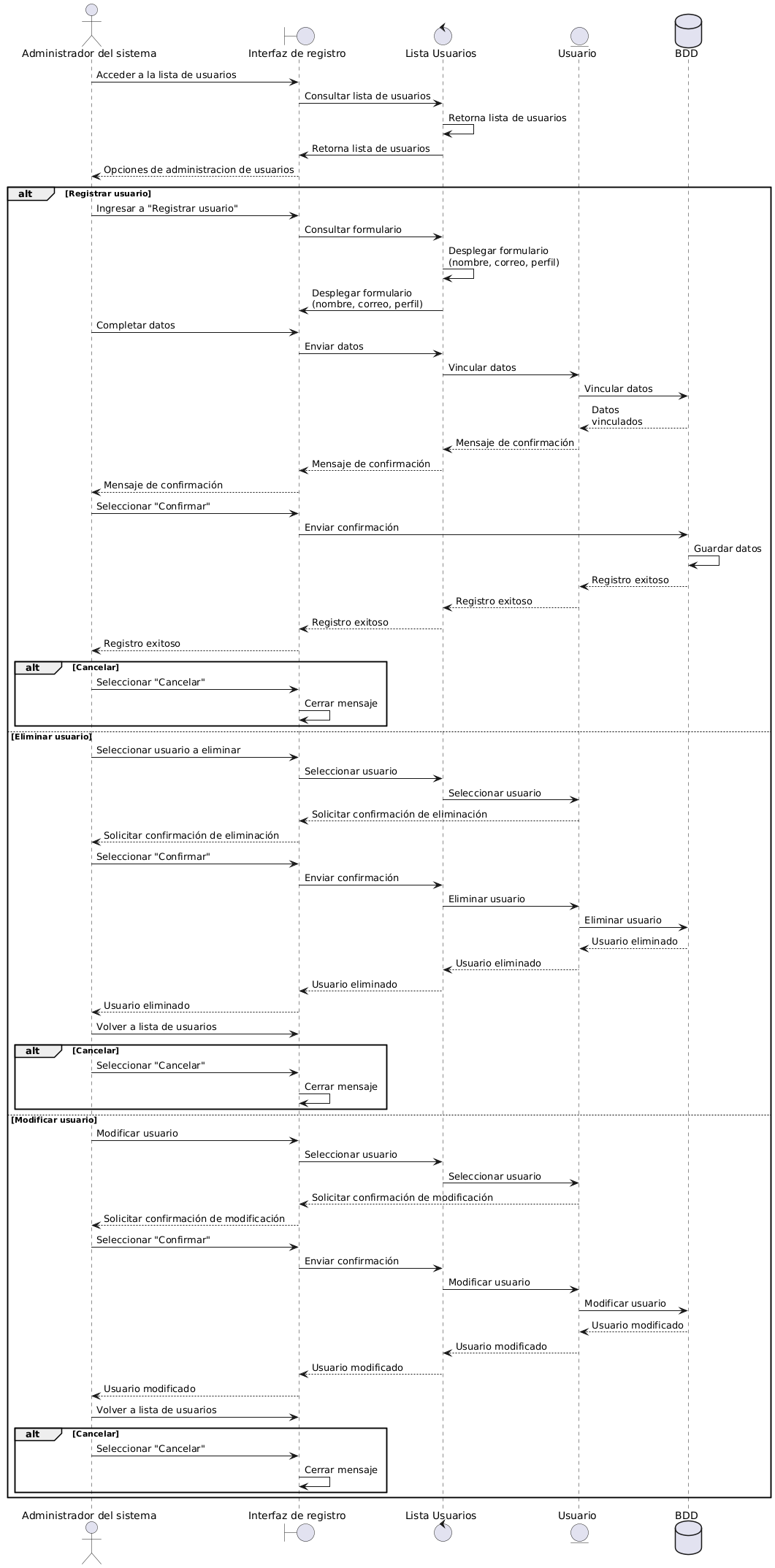
Diagrama de paquetes

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



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

Caso de Uso 3: Administrar proyectos

Diagrama de paquetes

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

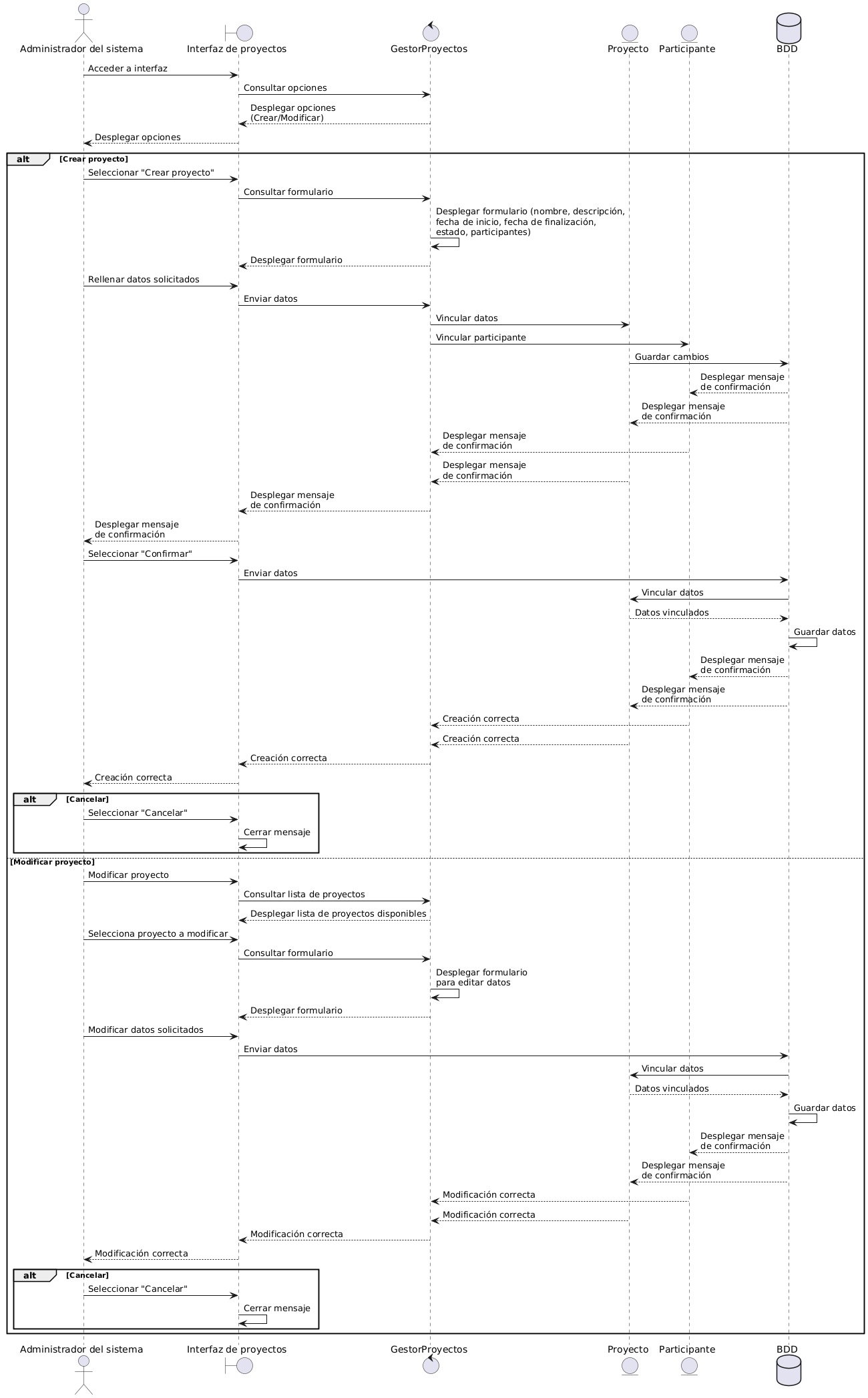
* GestorProyecto.
  + Tendrá los métodos: Constructor, retornar un proyecto por id, retornar proyecto por nombre, crear proyecto, actualizar proyecto, vincular o desvincular iteraciones, obtener todos los proyectos, vincular o desvincular participantes, obtener iteraciones por id de proyecto, actualizar iteracion por id, eliminar iteracion por id, añadir iteracion.
  + Comentario: Cada método deberá comprobar que no se pase ningún campo vacio, ni ninguna inyección sql.
* Proyecto.
  + Tendrá los atributos de nombre, descripcion, 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 iteracion por id, eliminar iteracion por id, añadir iteracion.

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

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: descripcion, factor\_riesgo, categoría, iteracion 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, descripcion 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.

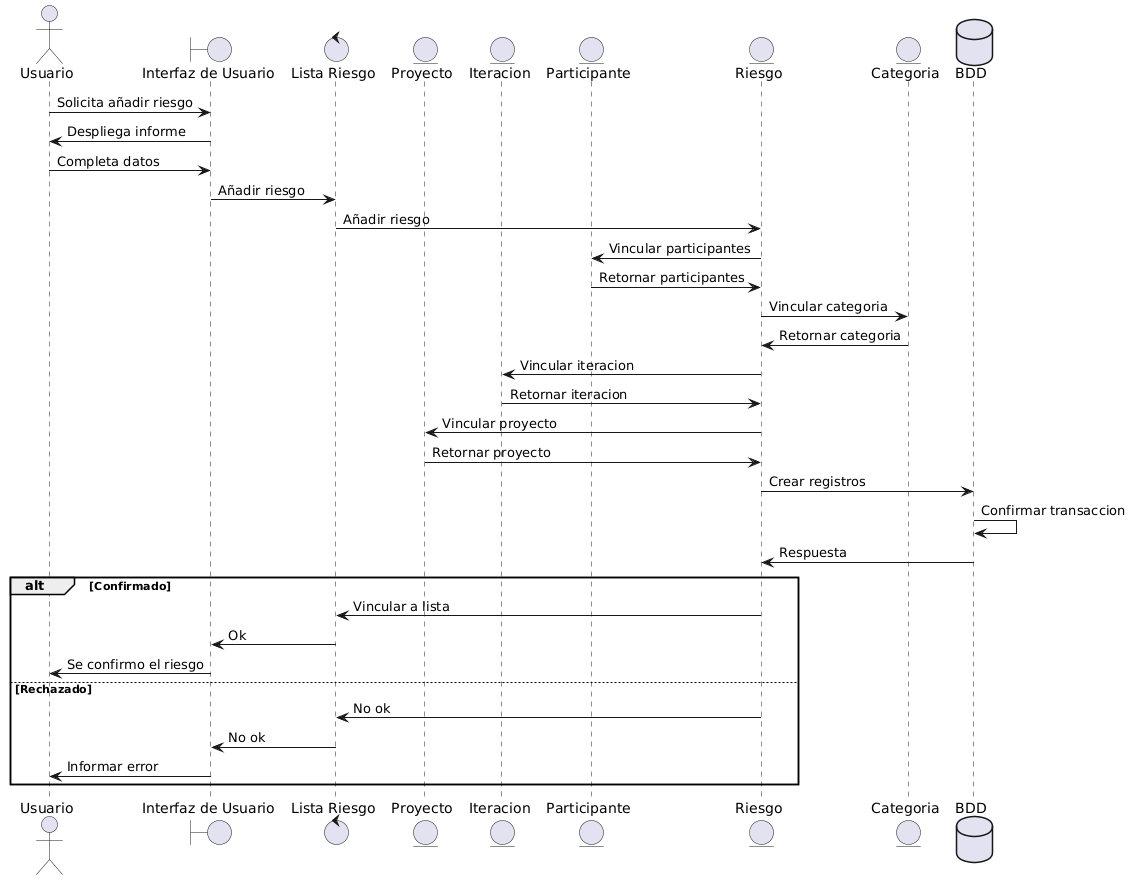
* GestorProyecto.
  + Tendrá los métodos: Constructor, retornar un proyecto por id, retornar proyecto por nombre, crear proyecto, actualizar proyecto, vincular o desvincular iteraciones, obtener todos los proyectos, vincular o desvincular participantes, obtener iteraciones por id de proyecto, actualizar iteracion por id, eliminar iteracion por id, añadir iteracion.
  + Comentario: Cada método deberá comprobar que no se pase ningún campo vacio, ni ninguna inyección sql.
* Proyecto.
  + Tendrá los atributos de nombre, descripcion, 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 iteracion por id, eliminar iteracion por id, añadir iteracion.

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. 1.4**: Diagrama de secuencia CU04.

Caso de Uso 5: Modificar lista de riesgos

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 evaluacion, obtener evaluaciones de un riesgo, obtener todas las evaluaciones, actualizar evaluacion por id, eliminar evaluacion por id.
* Riesgo.
  + Tendrá los atributos: descripcion, factor\_riesgo, categoría, iteracion 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, descripcion 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 evaluacion, obtener evaluaciones de un riesgo, obtener todas las evaluaciones, actualizar evaluacion por id, eliminar evaluacion por id.

Estas clases están vinculadas al subsistema de Riesgos.

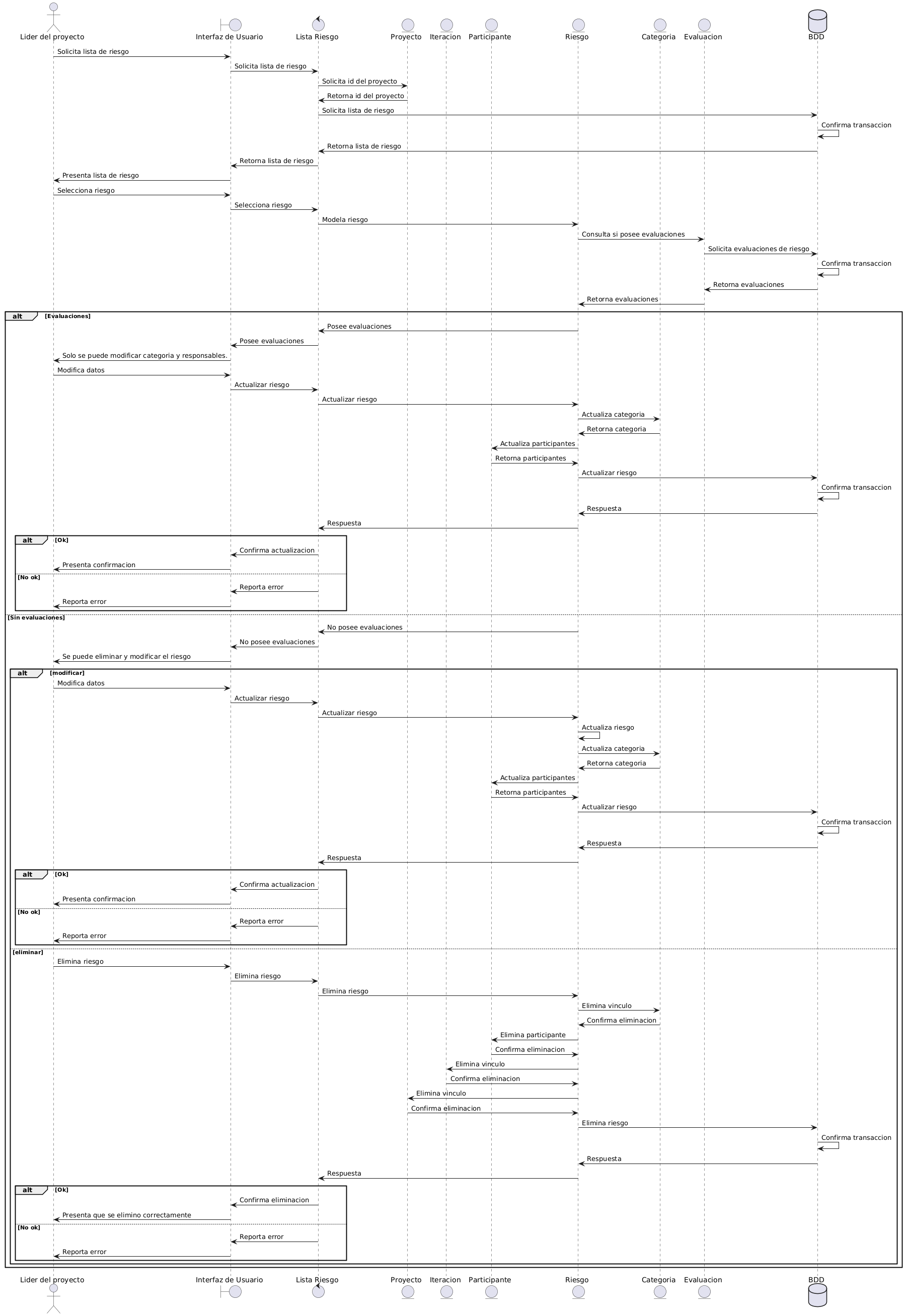
* GestorProyecto.
  + Tendrá los métodos: Constructor, retornar un proyecto por id, retornar proyecto por nombre, crear proyecto, actualizar proyecto, vincular o desvincular iteraciones, obtener todos los proyectos, vincular o desvincular participantes, obtener iteraciones por id de proyecto, actualizar iteracion por id, eliminar iteracion por id, añadir iteracion.
  + Comentario: Cada método deberá comprobar que no se pase ningún campo vacio, ni ninguna inyección sql.
* Proyecto.
  + Tendrá los atributos de nombre, descripcion, 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 iteracion por id, eliminar iteracion por id, añadir iteracion.

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

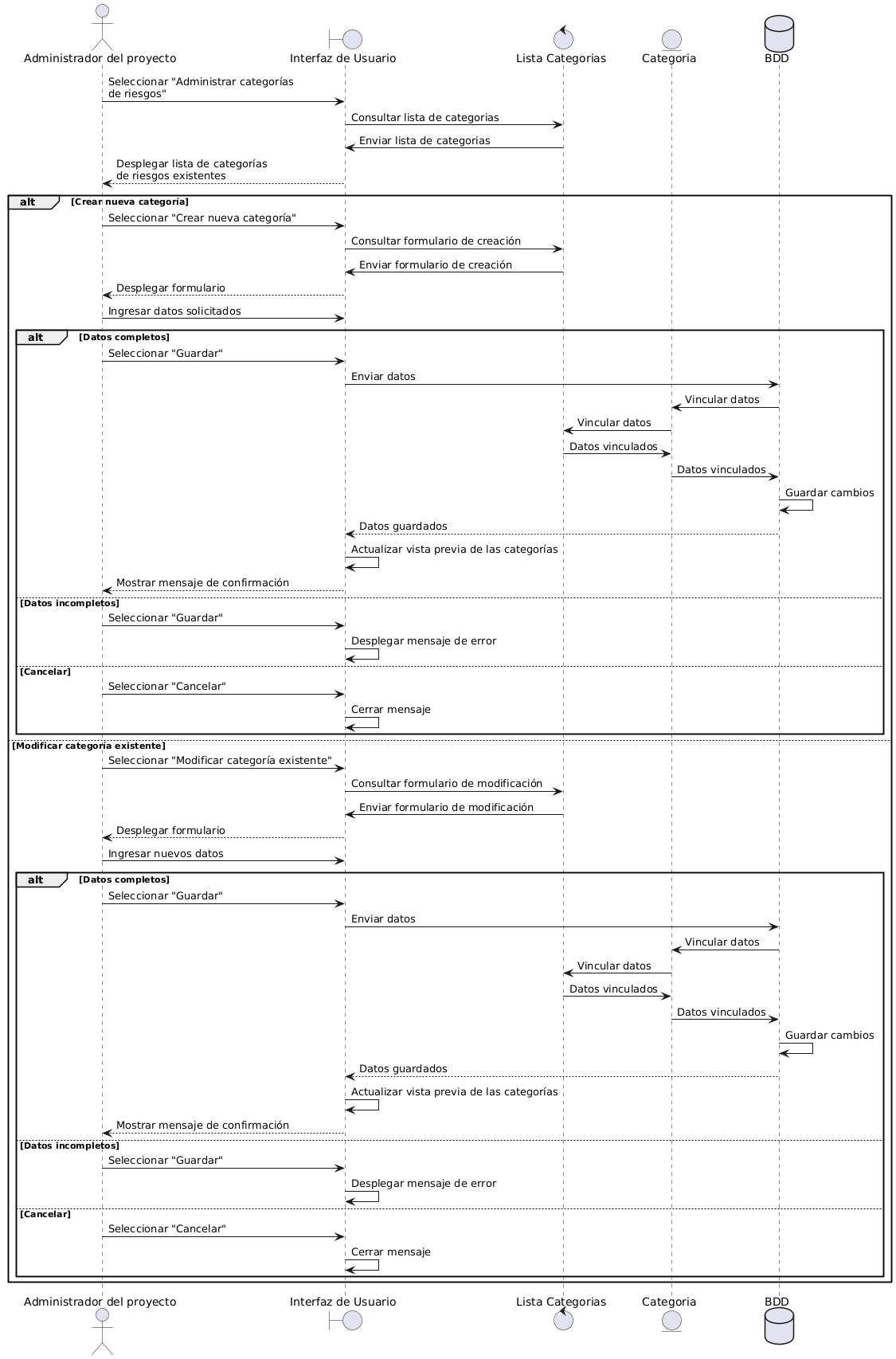
Diagrama de paquetes

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: descripcion, factor\_riesgo, categoría, iteracion 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, descripcion 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

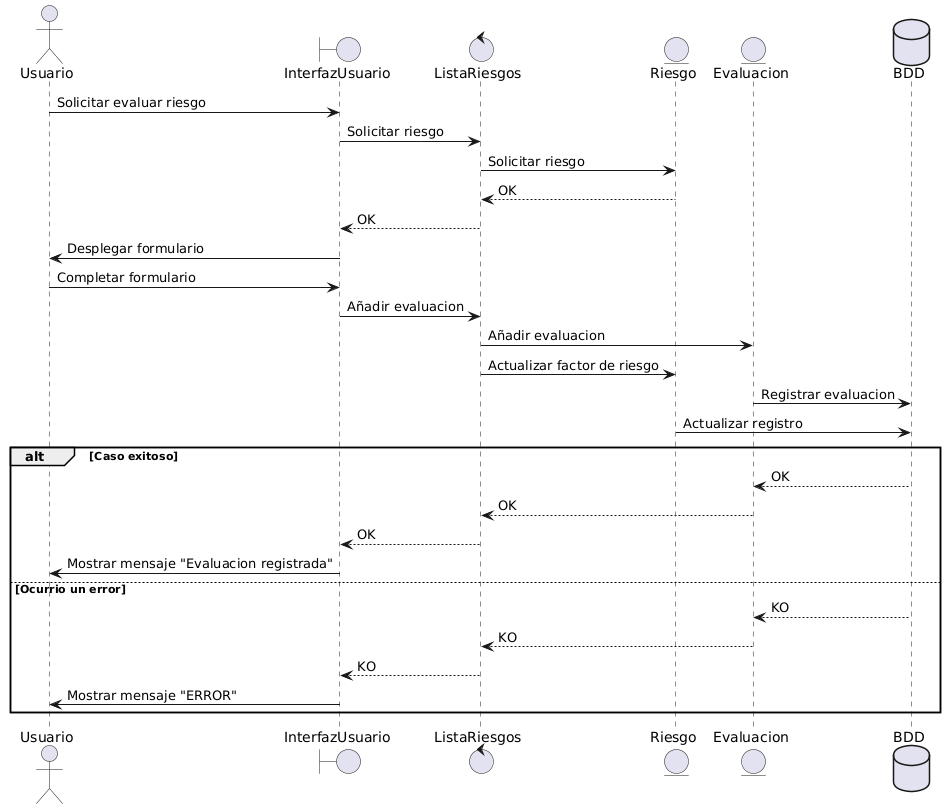
Diagrama de paquetes

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 evaluacion, obtener evaluaciones de un riesgo, obtener todas las evaluaciones, actualizar evaluacion por id, eliminar evaluacion por id.
* Riesgo.
  + Tendrá los atributos: descripcion, factor\_riesgo, categoría, iteracion 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, descripcion 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 evaluacion, obtener evaluaciones de un riesgo, obtener todas las evaluaciones, actualizar evaluacion por id, eliminar evaluacion 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

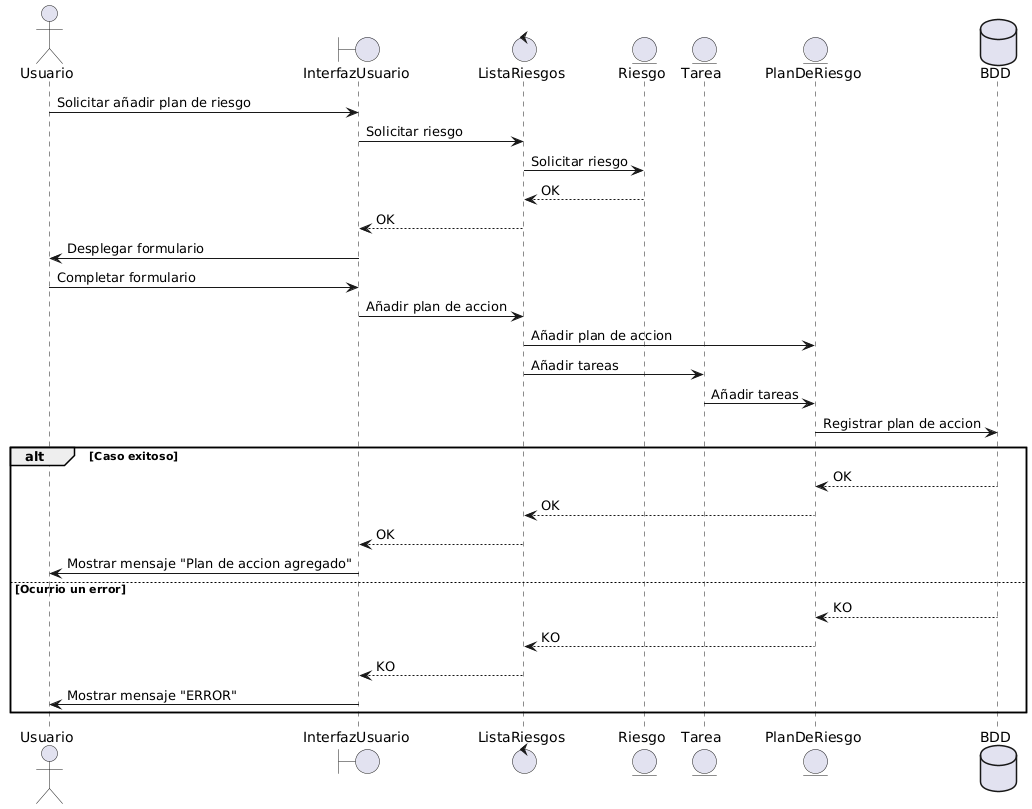
Diagrama de paquetes

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 evaluacion, obtener evaluaciones de un riesgo, obtener todas las evaluaciones, actualizar evaluacion por id, eliminar evaluacion 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: descripcion, factor\_riesgo, categoría, iteracion 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, descripcion 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 evaluacion, obtener evaluaciones de un riesgo, obtener todas las evaluaciones, actualizar evaluacion por id, eliminar evaluacion por id.
  + Comentario: Esta clase se utiliza de manera implícita en este diagrama de secuencia
* PlanDeRiesgo
  + Tendrá los atributos: nombre, descripcion, 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, descripcion, 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.

Diagrama de Interacción



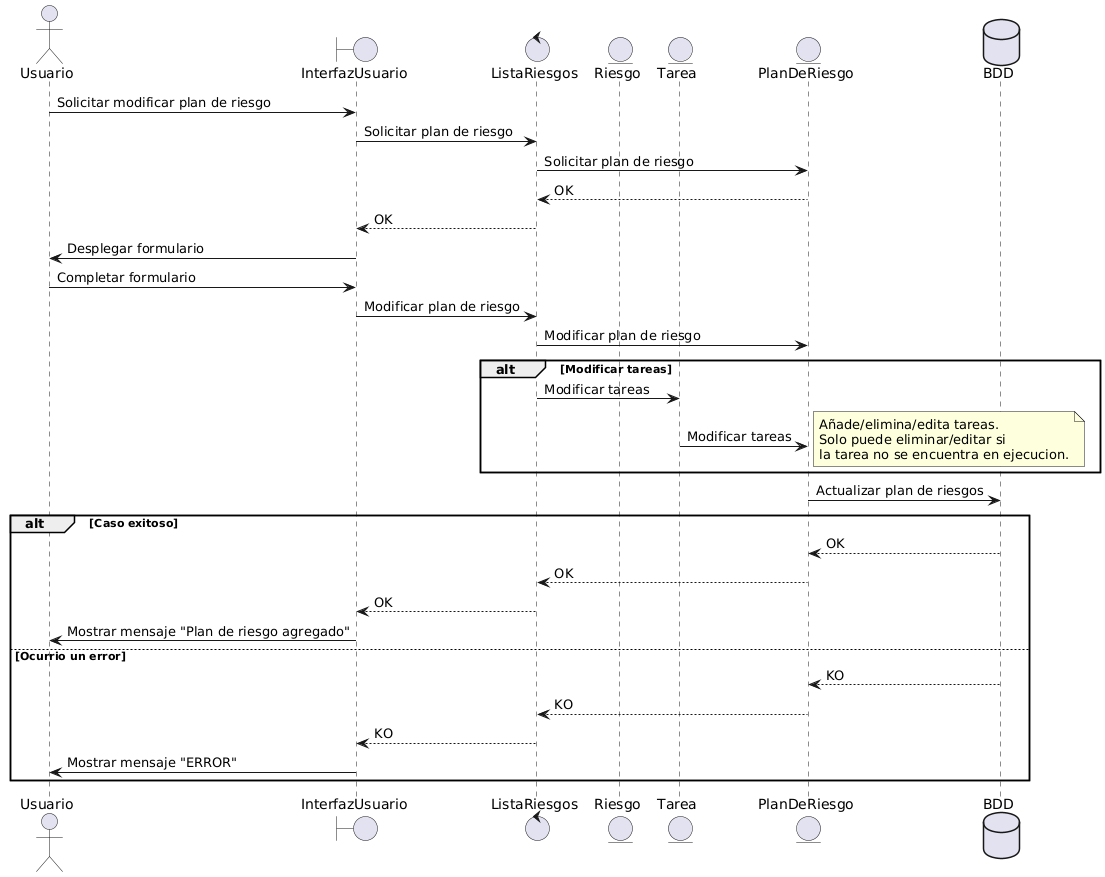
**Fig. 1.8**: Diagrama de secuencia CU08.

Caso de Uso 9: Modificar plan de acción

Diagrama de paquetes

Será definido en la siguiente entrega.

Diagrama de Interacción



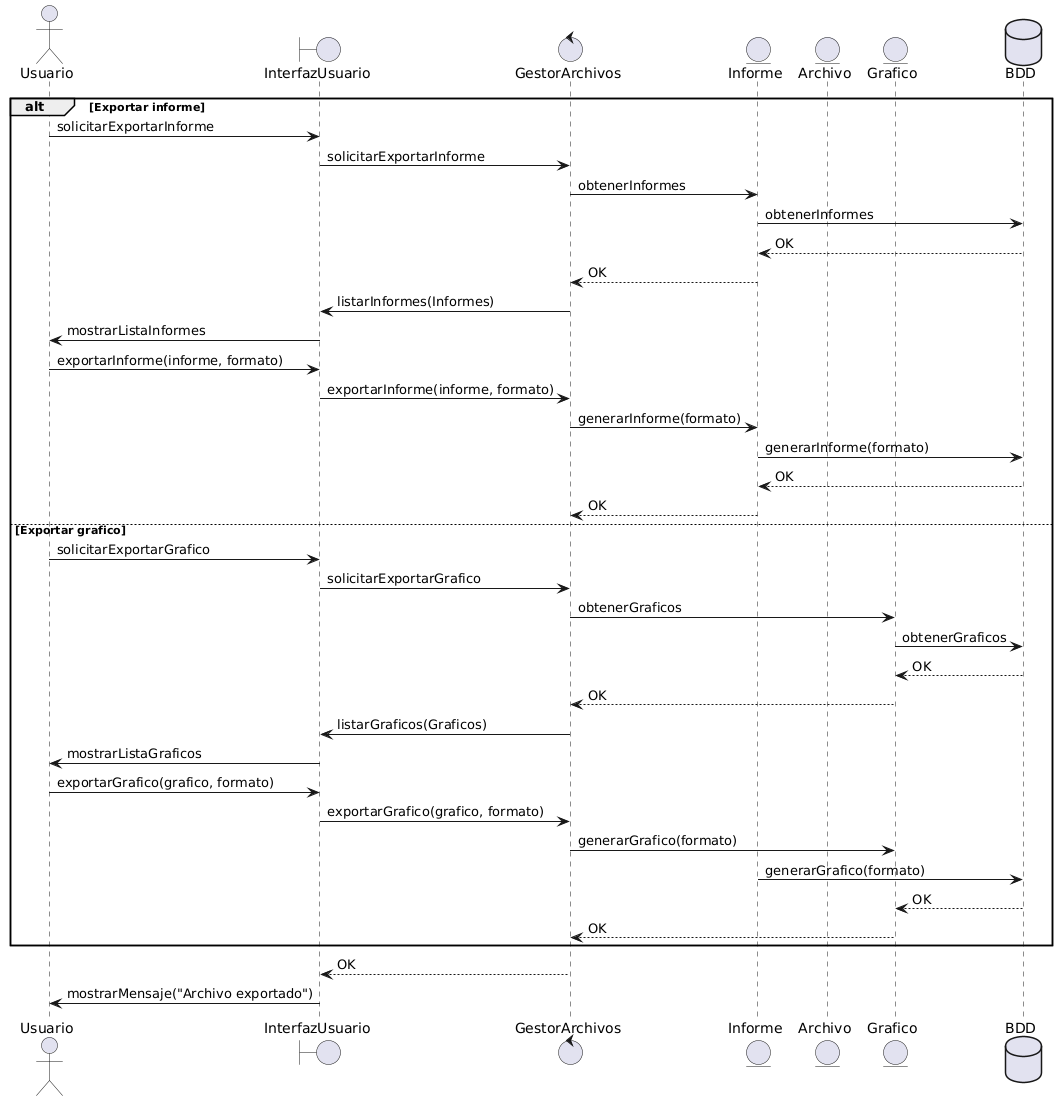
**Fig. 1.9**: Diagrama de secuencia CU09.

Caso de Uso 10: Realizar informes

Diagrama de paquetes

Será definido en la siguiente entrega.

Diagrama de Interacción



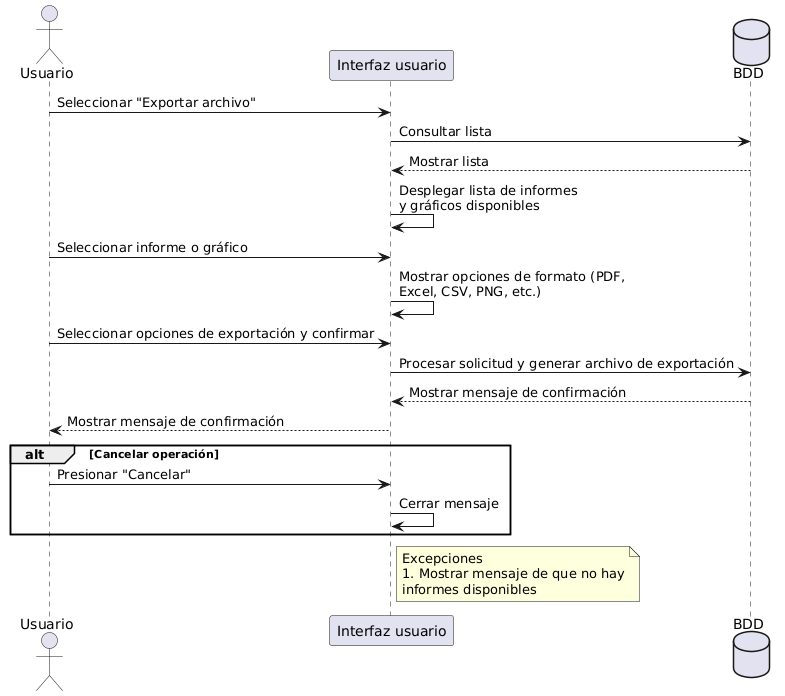
**Fig. 1.10**: Diagrama de secuencia CU10.

Caso de Uso 11: Exportar archivos

Diagrama de paquetes

Será definido en la siguiente entrega.

Diagrama de Interacción



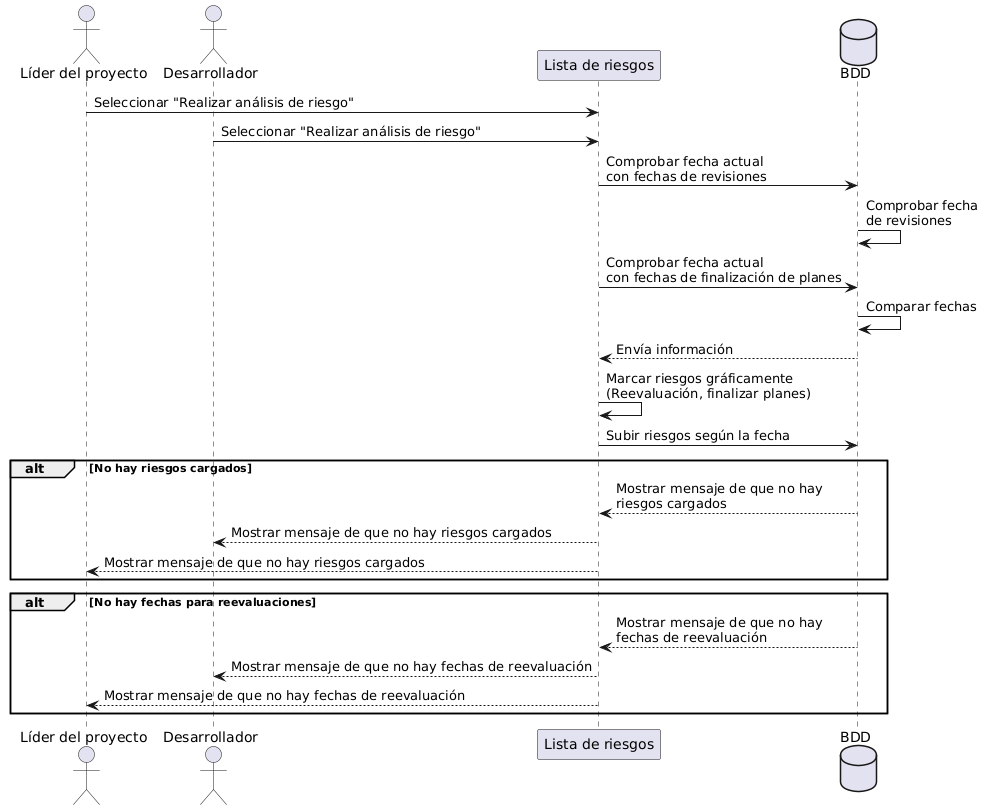
**Fig. 1.11**: Diagrama de secuencia CU11.

Caso de Uso 12: Realizar análisis de riesgo

Diagrama de paquetes

Será definido en la siguiente entrega.

Diagrama de Interacción



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

Diseño de Objetos

A Será definido en la siguiente entrega.

[Objeto 1]

[Se especifican los parámetros, reglas, condiciones usando la misma sintaxis del lenguaje y el código y métodos son especificados en pseudocódigo.]

* Descripción:
* Objetivo:
* Atributos:
* Métodos
* Interacciones:
* Diagrama de estado
* Interfaces que implementa

[Objeto 2]

...

Diseño de Subsistemas

Será definido en la siguiente entrega.

Subsistemas Específicos

[Nombre del Subsistema Específico 1]

Propósito

[Descripción de por qué el subsistema existe. Este atributo debe dar la razón de la creación del subsistema. Como ser la funcionalidad específica y los requerimientos de performance por los cuales fue creado. También describe requerimientos especiales que se deben lograr con él que no están incluidos en la especificación de requerimientos del software.]

Función

[Expresa qué realiza el subsistema. Establece la transformación aplicada a las entradas del subsistema para producir la salida deseada.]

Subordinados

[Se identifican los objetos de diseño y subsistemas de diseño que componen el subsistema que se describe. Se propone representar esta información con un diagrama de paquetes.]

Dependencias

[Descripción de la relación de este subsistema con otros subsistemas de diseño. Describir la naturaleza de cada interacción incluyendo características como tiempo y condiciones de la interacción. Estas, pueden involucrar la iniciación, orden de ejecución, datos compartidos, creación, duplicación, uso o almacenamiento.

Se propone representar esta información con una tabla de dependencias.]

|  |  |  |
| --- | --- | --- |
| Subsistema del que depende | Naturaleza de interacción | Características |
| [Identificación del subsistema del que depende] | [Condiciones para que se realice la interacción] | [Características de la interacción, como ser, pasaje de parámetros, mensajes, datos compartidos, etc.] |

Recursos

[Se deben Identificar y describir todos los recursos externos al diseño que necesita el subsistema para realizar su función. Especificar las reglas de interacción y métodos para usar el recurso.

Este atributo brinda información sobre elementos como dispositivos físicos (impresoras, particiones de disco, bancos de memoria), servicios de software (librerías, servicios del sistema operativo), y recursos de procesamiento (ciclos de CPU, ubicación de memoria, buffers).

Se deben describir características de uso como el tiempo de proceso al cual se debe adquirir el recurso e incluir la cantidad de tiempo de uso. Debe incluir también la identificación de capacidad potencial y facilidades de manejo del recurso.]

# Diagramas

Diagrama de componentes

Este diagrama se realizará una vez definida la arquitectura del sistema.

Diagrama de Clases



**Fig. 2.1**: Diagrama de Clases.

Diagramas de Paquetes



Diagrama de Colaboración

Este diagrama se realizará una vez definida la arquitectura del sistema.