EDUCAQuest

Trabajo de Fin de Grado

Ingeniería Informática

VNiVERSiDAD
D SALAMANCA

Junio de 2025

Autor:
Iker Botana Vázquez

Tutores:
María José Polo Martín
Jaime Rodríguez Moro
Enrique Prieto Conde

Anexo II: Análisis y Diseño del sistema

ÍNDICE

INTRODUCCIÓN **5**
MODELO DEL DOMINIO **6**
ARQUITECTURA LÓGICA DEL SISTEMA **9**
Gestión de actividades 11
Gestión de Roles y Usuarios 11
Gestión de Seguimiento y Evaluación 12
DIAGRAMAS DE SECUENCIA **13**
Registrarse 13
Iniciar sesión 14
Recuperar contraseña 15
Cerrar sesión 16
Editar perfil 17
Asignar rol a usuario 18
Crear actividad 19
Visualizar actividades asignadas 20
Enviar actividad completada 21
Corregir actividad 22
Corregir actividad de casa 23
Visualizar ranking de alumnos 24
Recibir aviso de tareas pendientes 25
Ver corrección de actividad 26
Filtrar actividades por asignatura 27
Ordenar actividades 28
Cambiar vista de actividades (lista/cuadrícula) 29
Retroceder a pantalla anterior 30
Asignar curso a usuario 31
Asignar asignatura a usuario 32
PROPUESTA DE ARQUITECTURA **33**

LISTA DE FIGURAS

Figura 1. Modelo de dominio 7
Figura 2. Gestión de Usuarios 10
Figura 3. Gestión de Actividades 11
Figura 4. Gestión de Roles y Usuarios 11
Figura 5. Gestión de Seguimiento y Evaluación 12
Figura 6. Diagrama de secuencia 1 (Registrarse) 13
Figura 7. Diagrama de secuencia 2 (Iniciar Sesión) 14
Figura 8. Diagrama de secuencia 3 (Recuperar contraseña) 15
Figura 9. Diagrama de secuencia 4 (Cerrar Sesión) 16
Figura 10. Diagrama de secuencia 5 (Editar perfil) 17
Figura 11. Diagrama de secuencia 6 (Asignar rol a usuario) 18
Figura 12. Diagrama de secuencia 7 (Crear actividad) 19
Figura 13. Diagrama de secuencia 8 (Visualizar actividades asignadas) 20
Figura 14. Diagrama de secuencia 9 (Enviar actividad completada) 21
Figura 15. Diagrama de secuencia 10 (Corregir actividad) 22
Figura 16. Diagrama de secuencia 11 (Corregir actividad de casa) 23
Figura 17. Diagrama de secuencia 12 (Visualizar ranking de alumnos) 24
Figura 18. Diagrama de secuencia 13 (Recibir aviso tareas pendientes) 25
Figura 19. Diagrama de secuencia 14 (Ver corrección actividad) 26
Figura 20. Diagrama de secuencia 15 (Registrarse) 27
Figura 21. Diagrama de secuencia 16 (Ordenar actividades) 28
Figura 22. Diagrama de secuencia 17 (Cambiar vista de actividades) 29
Figura 23. Diagrama de secuencia 18 (Retroceder a pantalla anterior) 30
Figura 24. Diagrama de secuencia 19 (Asignar curso a usuario) 31
Figura 25. Diagrama de secuencia 20 (Asignar asignatura a usuario) 32
Figura 26. Propuesta de Arquitectura 33

INTRODUCCIÓN

En este Anexo se detallarán los requisitos del sistema de software para EDUCAQuest, una aplicación educativa creada para facilitar la labor de la “Fundación Montemadrid”, cuyo objetivo es motivar a los niños en edad escolar a mantenerse al día con sus tareas mediante técnicas de gamificación.

El sistema contará con un registro de usuarios con diferentes perfiles, permitiendo un acceso personalizado según el rol. Se distinguen varios tipos de usuarios, como estudiantes, educadores y administradores. Los estudiantes podrán visualizar sus tareas representadas como misiones, recibir notificaciones y obtener recompensas. Además, podrán personalizar su propio avatar para mejorar su experiencia dentro de la plataforma.

Para la recopilación de requisitos, se ha seguido la metodología propuesta por Durán y Bernárdez, que contempla la identificación de participantes, objetivos y requisitos, clasificados en distintas categorías que se detallarán a continuación.

MODELO DEL DOMINIO

El sistema EDUCAQuest se concibe como una plataforma educativa gamificada que permita gestionar tareas escolares de forma dinámica y participativa. Está dirigida principalmente a niños y niñas en edad escolar que forman parte de los programas de la Fundación Montemadrid en entornos de vulnerabilidad social.

Los actores principales identificados en el sistema son:

- Estudiantes: usuarios principales que gestionan tareas, completan misiones, personalizan su perfil y participan en dinámicas sociales.

- Educadores: pueden crear misiones, revisar el progreso del alumnado y moderar la plataforma.

- Padres/Tutores legales: tienen acceso a los informes de actividad y pueden configurar ciertos parámetros del sistema.

- Administradores: usuarios con privilegios sobre la configuración general, integración con plataformas externas y gestión de la seguridad.

En la *Figura 1* se presenta el modelo de dominio desarrollado para este proyecto, el cual representa una abstracción conceptual de los principales objetos del sistema y sus relaciones. Este modelo facilita la comprensión de la estructura lógica del software, permitiendo identificar las entidades clave y cómo se conectan entre sí, tomando como referencia su comportamiento en el mundo real.

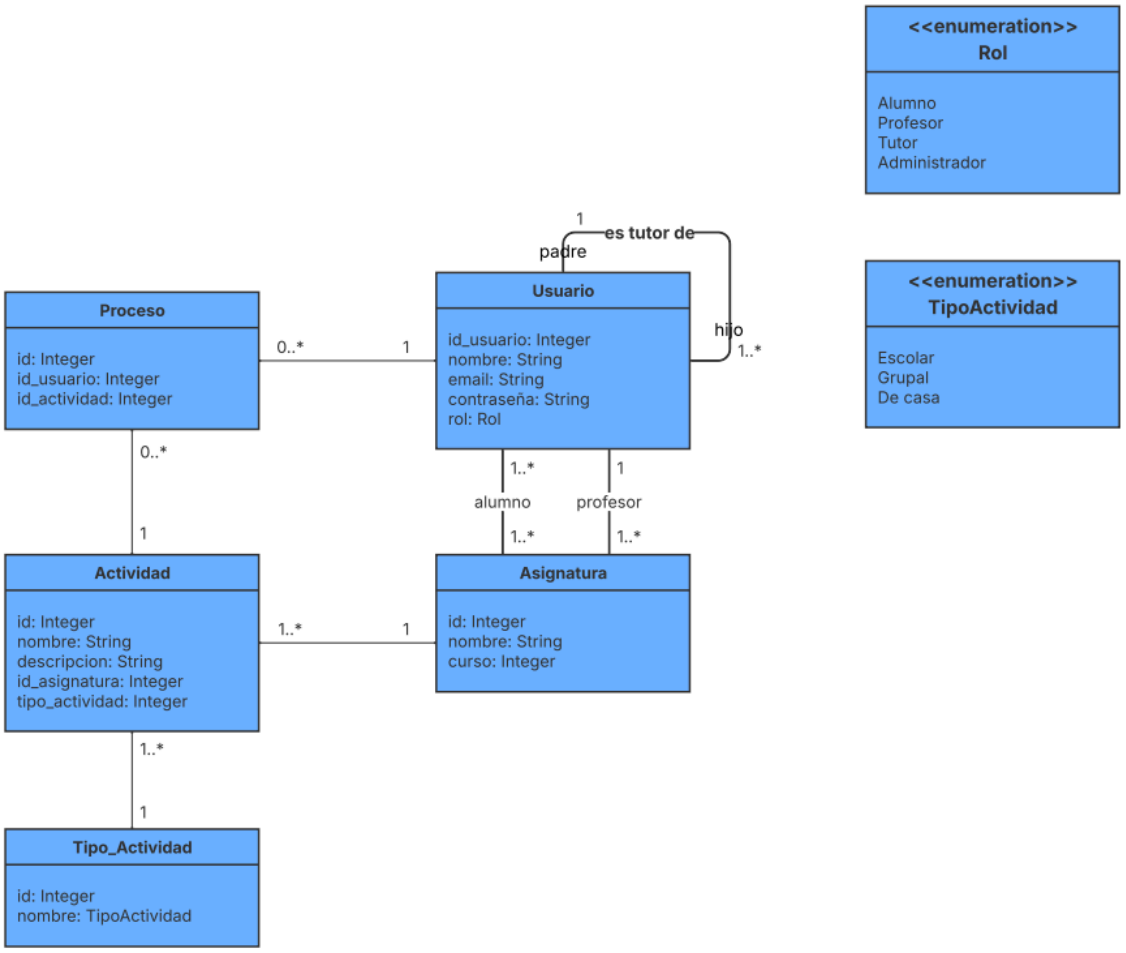

Figura 1. Modelo de dominio

En el anterior diagrama aparecen representadas las siguientes entidades y relaciones:

- Usuario: Representa a cualquier persona registrada en el sistema, pudiendo tener uno de los roles definidos en la enumeración Rol (Alumno, Profesor, Tutor o Administrador). Puede estar vinculado a otros usuarios como tutor legal (padre/madre) de un alumno.

- Asignatura: Representa una materia académica. Puede estar vinculada a varios usuarios con rol de profesor y sirve como categoría para las actividades educativas.

- Actividad: Representa una tarea educativa dentro de una asignatura, pudiendo clasificarse mediante la enumeración TipoActividad (Escolar, Grupal, De casa). Cada actividad puede ser asignada a múltiples usuarios.

- Proceso: Representa el vínculo entre un usuario y una actividad. Puede incluir información como el resultado de la entrega o su estado, y sirve como base para registrar y evaluar la participación del alumno.

- TipoActividad: Es una enumeración que define el tipo de actividad según su contexto o modalidad (Escolar, Grupal, De casa).

- Rol: Es una enumeración que identifica el tipo de usuario registrado en el sistema, determinando su nivel de acceso y funcionalidades disponibles.

- Relación tutor-hijo (autorrelación en Usuario): Define la relación familiar entre un usuario con rol Tutor y un usuario con rol Alumno, estableciendo quién es responsable de quién dentro del sistema.

ARQUITECTURA LÓGICA DEL SISTEMA

El proyecto ha sido desarrollado siguiendo el patrón de arquitectura Modelo-Vista-Controlador (MVC), adaptado a una arquitectura distribuida entre cliente y servidor. Esta elección permite separar de forma clara la lógica de negocio, la presentación de la información y la gestión de los datos, favoreciendo el mantenimiento, la escalabilidad y el desarrollo modular de la aplicación.

- Vista (View)

La vista corresponde al cliente web desarrollado en Angular. Está compuesta por interfaces dinámicas implementadas en HTML, CSS y TypeScript, encargadas de representar los datos al usuario y gestionar la interacción con el sistema.

Las vistas se comunican exclusivamente con el controlador mediante peticiones HTTP (REST), utilizando servicios definidos en Angular. No tienen acceso directo a los datos, sino que muestran la información que el backend les proporciona y recogen la introducida por el usuario a través de formularios, botones u otros elementos interactivos.

Cada tipo de usuario (administrador, alumno, profesor o tutor) tiene acceso a interfaces específicas en función de sus permisos, y estas vistas se organizan por funcionalidades como gestión de actividades, asignaturas, usuarios y estadísticas.

- Controlador (Controller)

El controlador está implementado en el servidor backend con Node.js y Express. Su función principal es recibir las peticiones del cliente, procesar la lógica correspondiente y comunicarse con el modelo para acceder o modificar los datos.

A través de una API REST, se definen endpoints que permiten realizar operaciones como la creación de usuarios, asignación de actividades, recuperación de estadísticas, autenticación, etc. Cada endpoint ejecuta funciones específicas que validan los datos, aplican reglas de negocio y devuelven la respuesta adecuada al cliente.

Esta capa actúa como intermediaria entre la vista y el modelo, garantizando la seguridad y controlando el flujo de información según el rol del usuario.

- Modelo (Model)

El modelo está compuesto por una base de datos relacional MySQL y por el conjunto de funciones y scripts que permiten acceder a ella. Representa la estructura lógica de la información y es responsable del almacenamiento, recuperación y modificación de los datos.

Las tablas del modelo incluyen entidades como usuarios, actividades, asignaturas, respuestas, entre otras, que reflejan directamente el modelo de dominio conceptual del sistema.

La conexión entre el backend y la base de datos se realiza mediante módulos específicos que encapsulan las consultas SQL necesarias. Esto permite mantener una separación clara entre la lógica de acceso a datos y el resto del controlador.

Gestión de usuarios

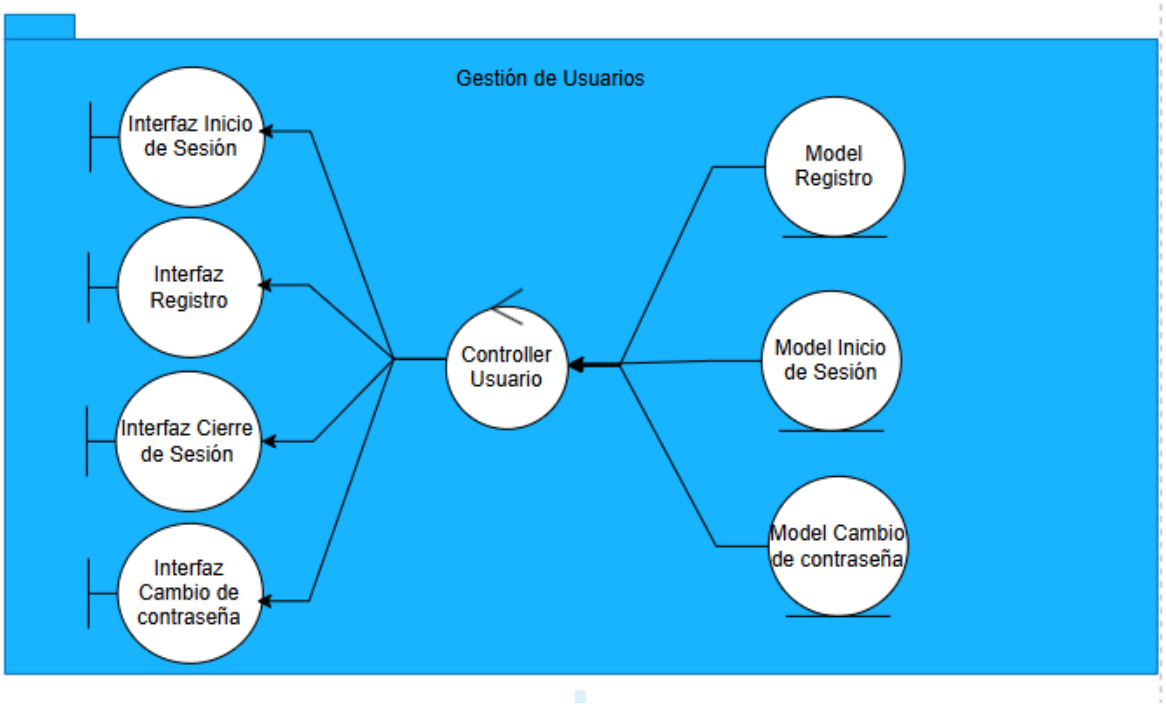

Figura 2. Gestión de Usuarios

Gestión de actividades

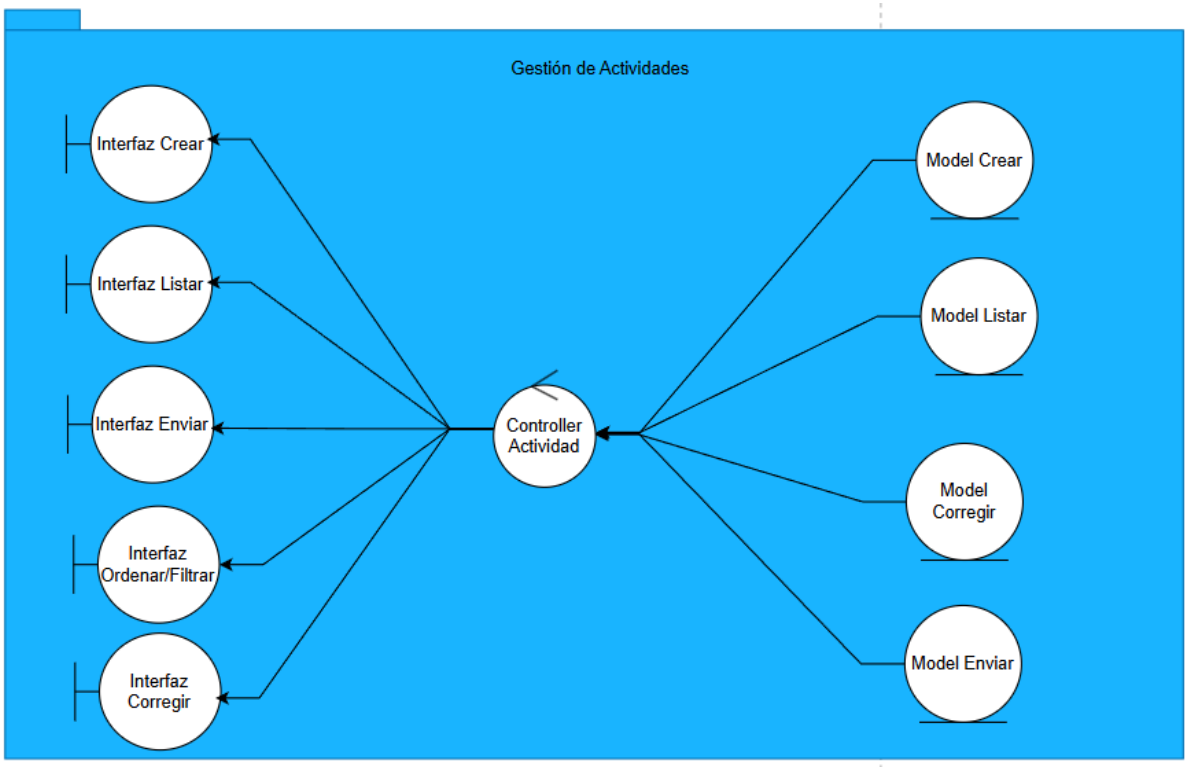

Figura 3. Gestión de Actividades

Gestión de Roles y Usuarios

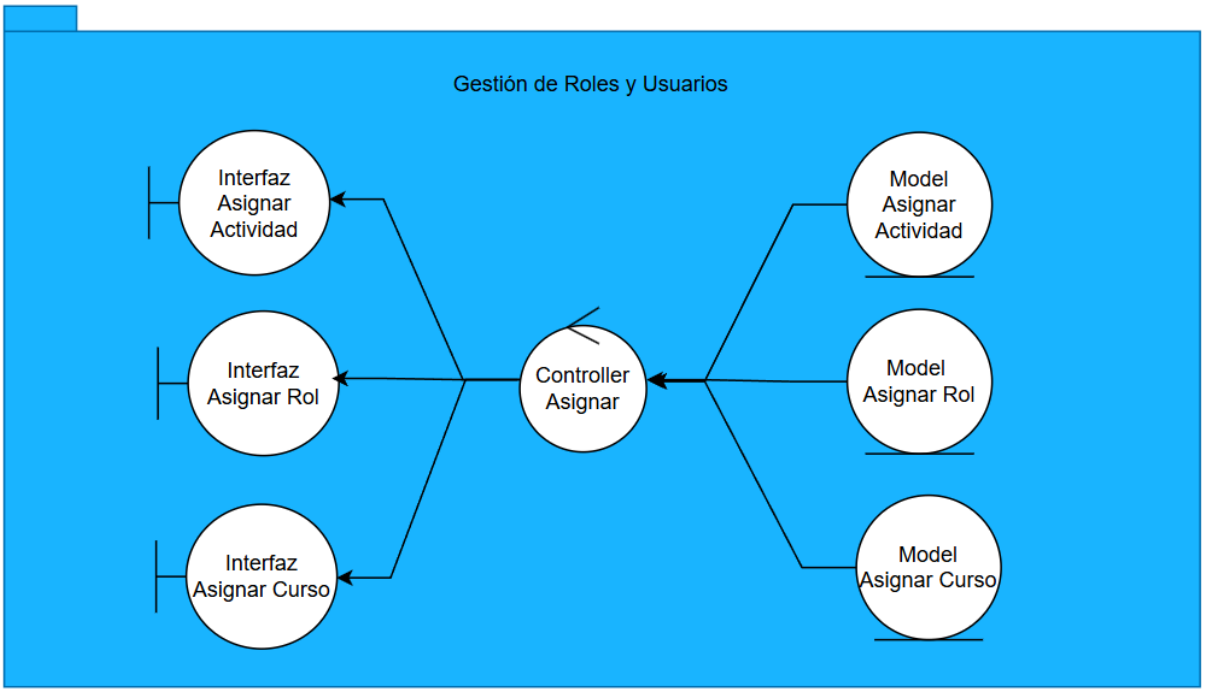

Figura 4. Gestión de Roles y Usuarios

Gestión de Seguimiento y Evaluación

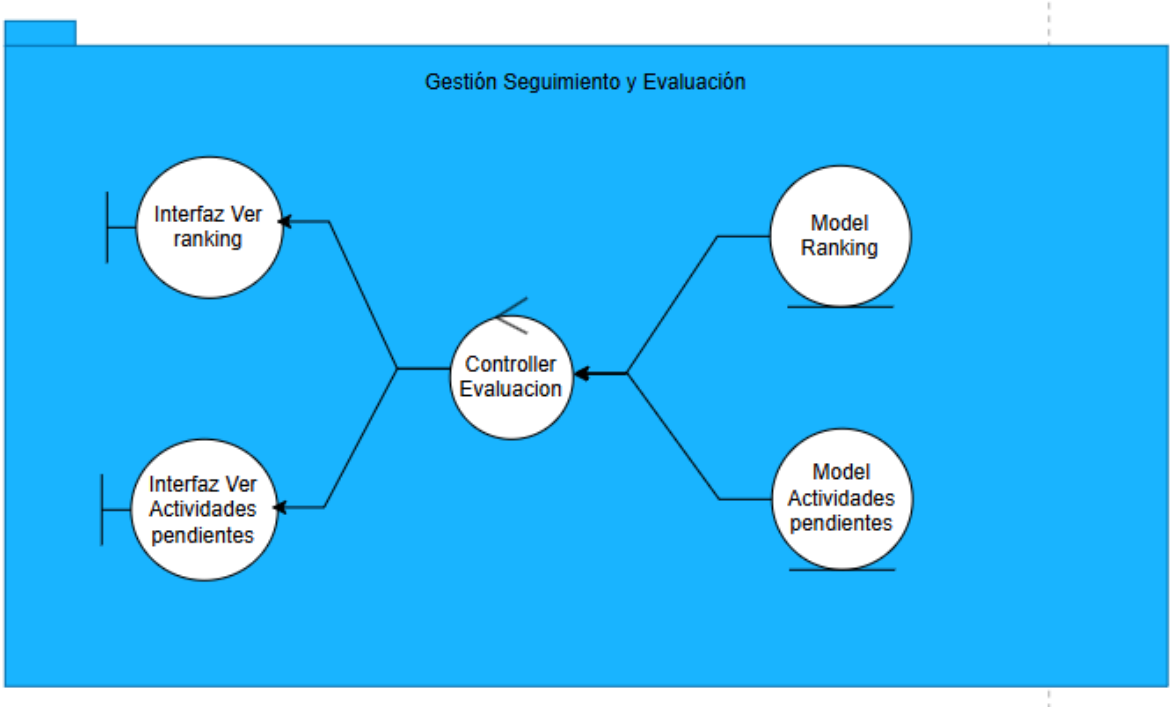

Figura 5. Gestión de Seguimiento y Evaluación

DIAGRAMAS DE SECUENCIA

Para complementar la descripción funcional del sistema, se han elaborado diagramas de secuencia que reflejan la interacción entre los distintos elementos de la arquitectura MVC durante la ejecución de funcionalidades clave. Estos diagramas muestran el flujo de mensajes entre la vista (Angular), el controlador (Node.js) y el modelo (MySQL) en casos como el registro de usuarios o la gestión de actividades.

A continuación, se muestran los diagramas correspondientes a los principales procesos del sistema.

Registrarse

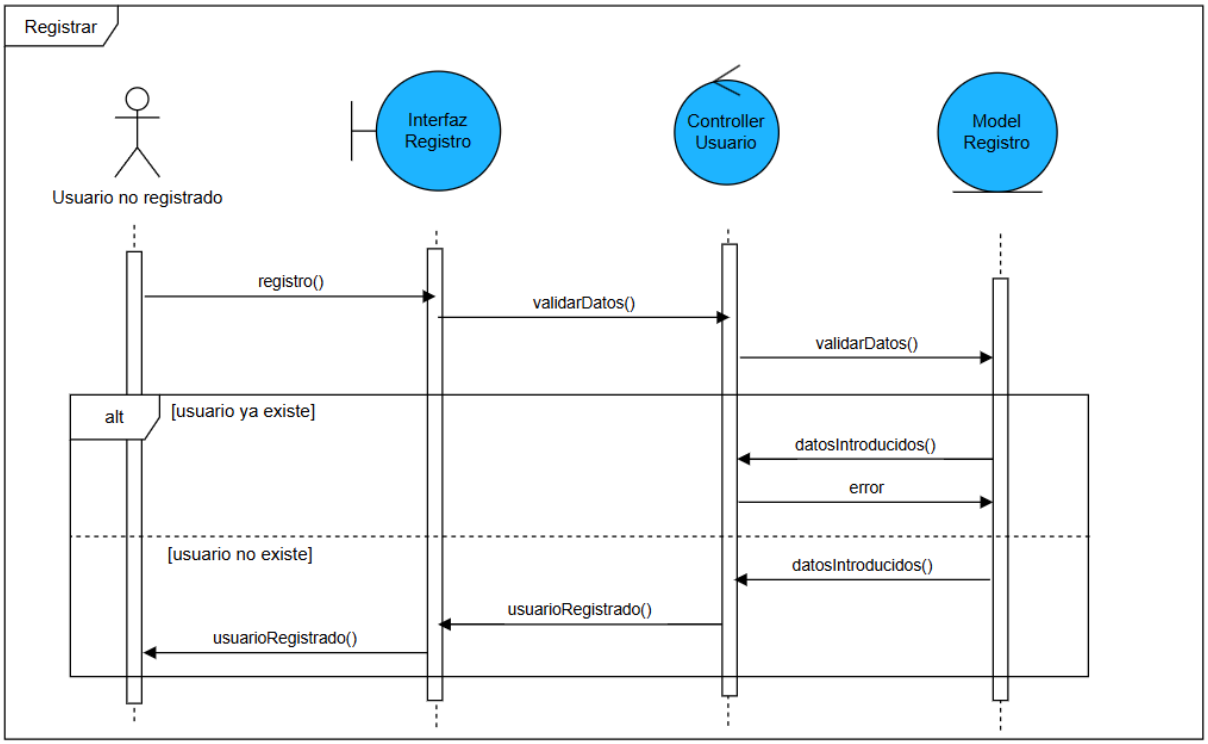

Figura 6. Diagrama de secuencia 1 (Registrarse)

Iniciar sesión

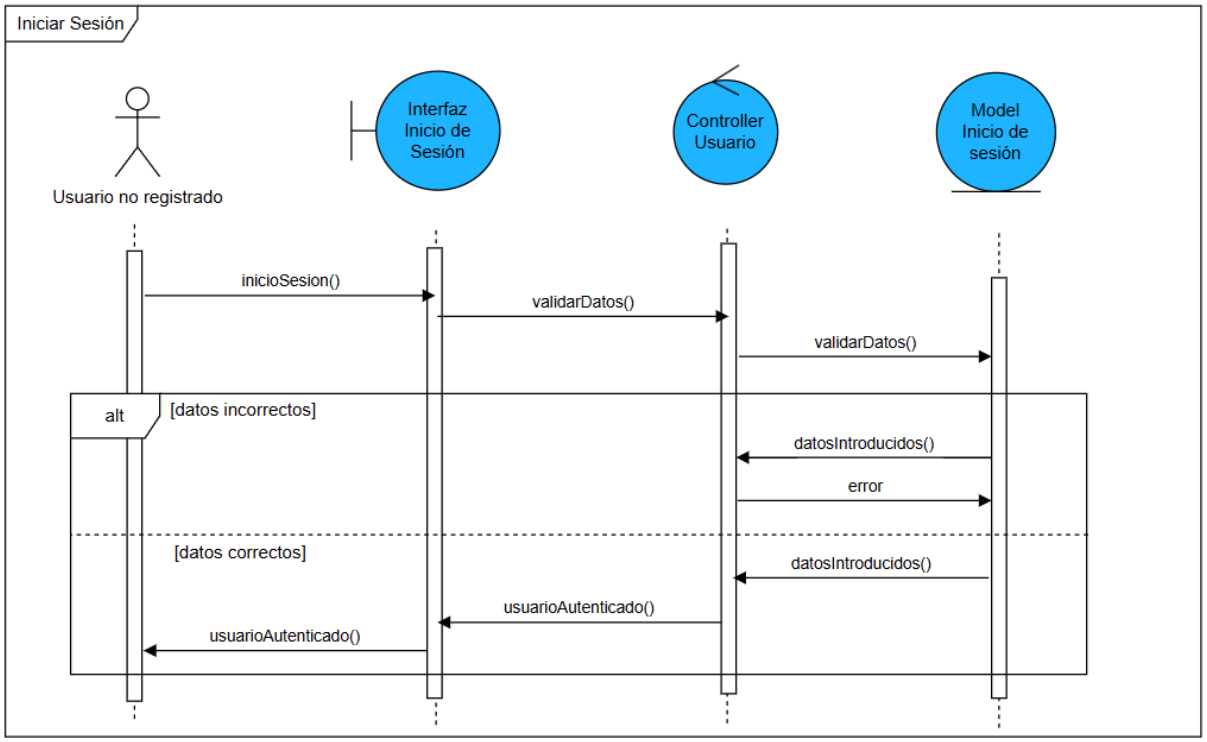

Figura 7. Diagrama de secuencia 2 (Iniciar Sesión)

Recuperar contraseña

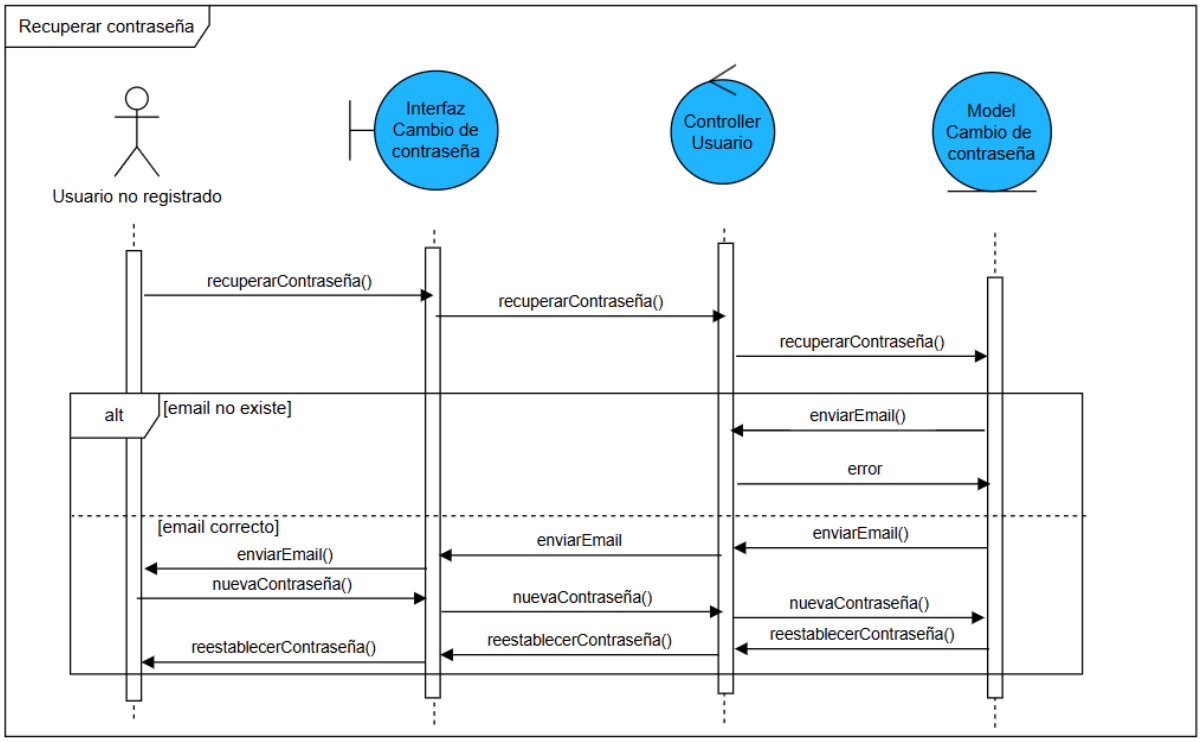

Figura 8. Diagrama de secuencia 3 (Recuperar contraseña)

Cerrar sesión

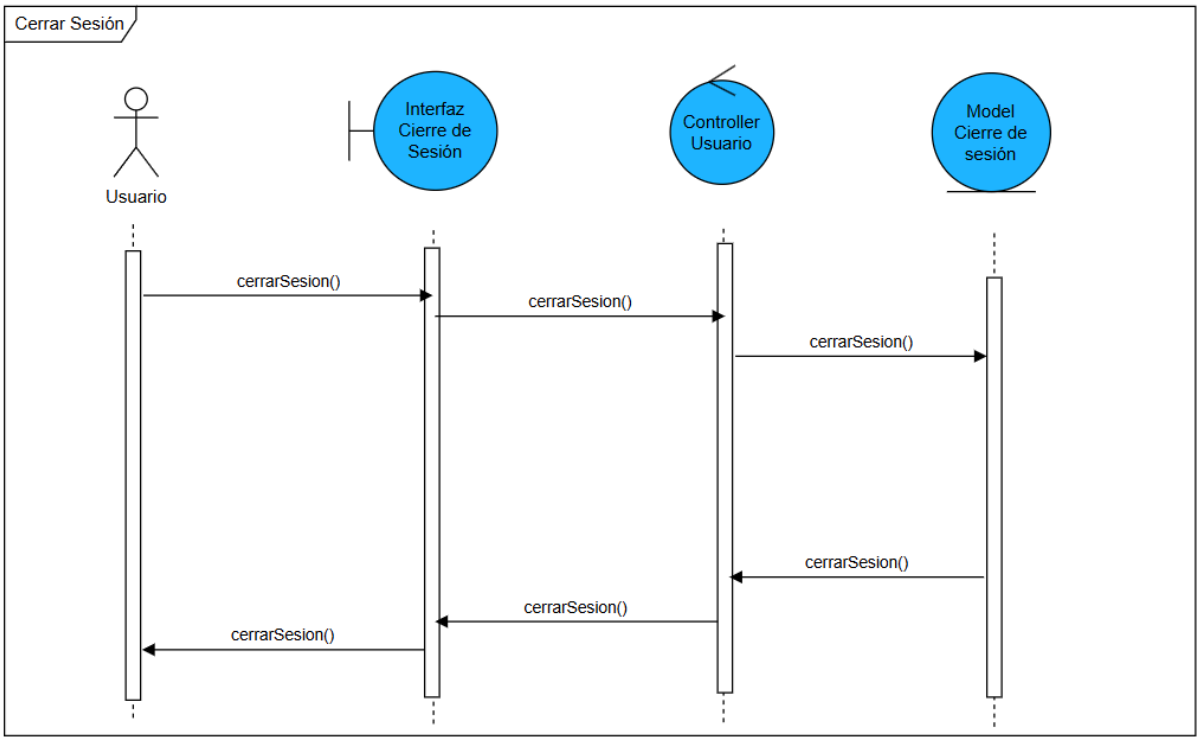

Figura 9. Diagrama de secuencia 4 (Cerrar Sesión)

Editar perfil

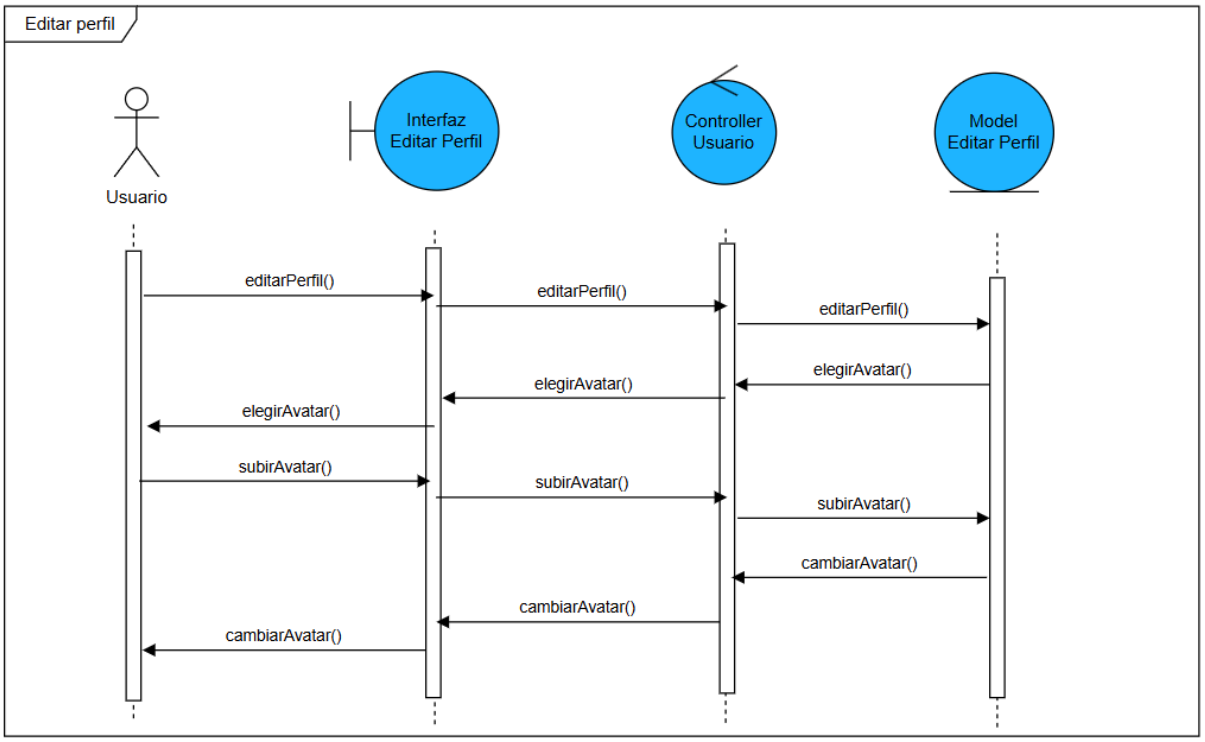

Figura 10. Diagrama de secuencia 5 (Editar perfil)

Asignar rol a usuario

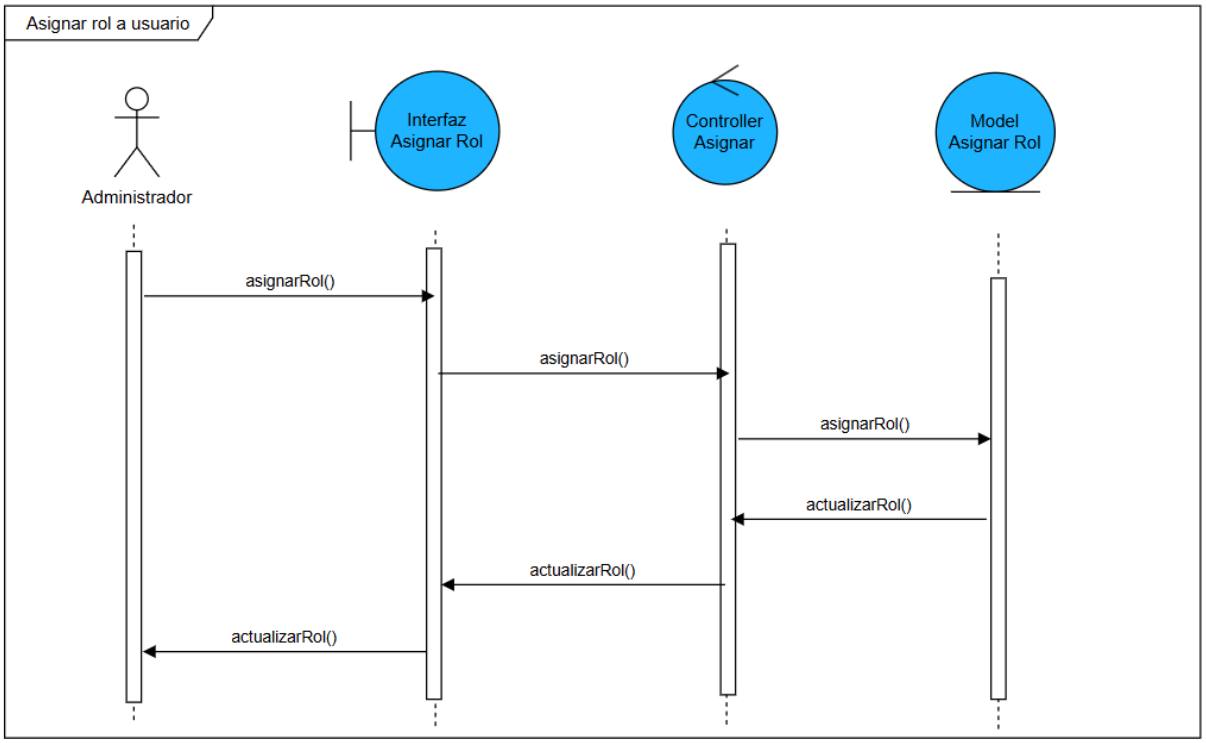

Figura 11. Diagrama de secuencia 6 (Asignar rol a usuario)

Crear actividad

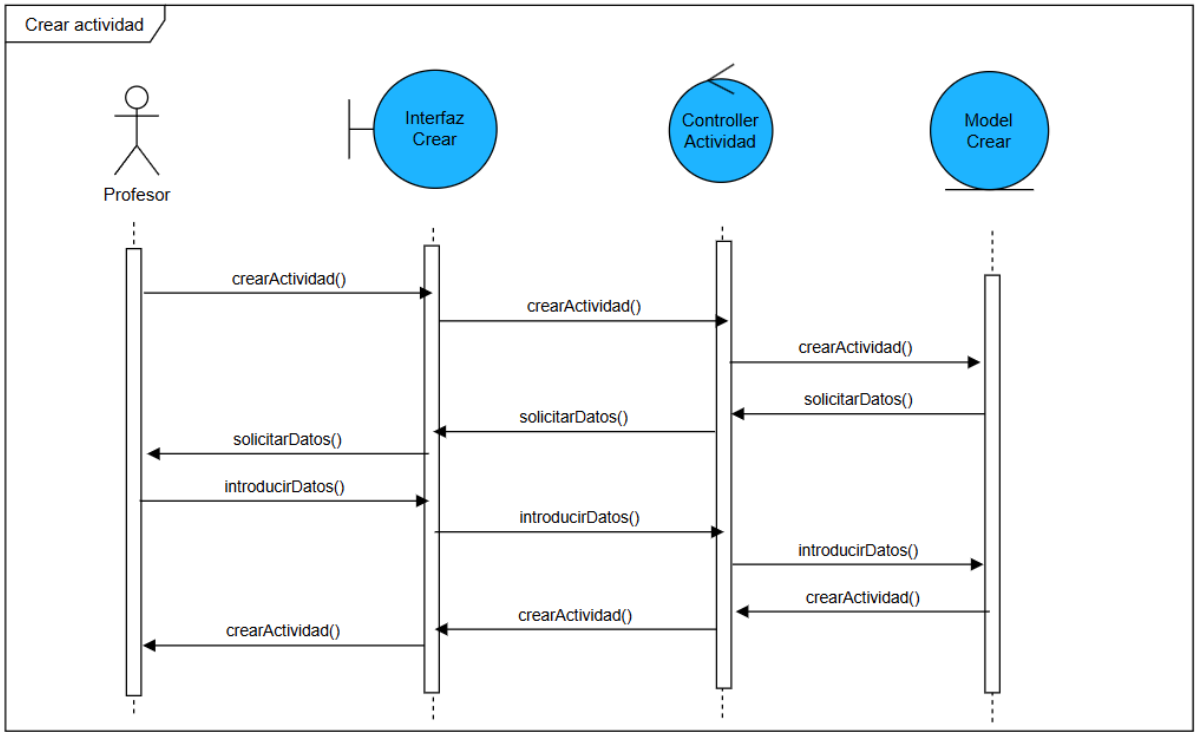

Figura 12. Diagrama de secuencia 7 (Crear actividad)

Visualizar actividades asignadas

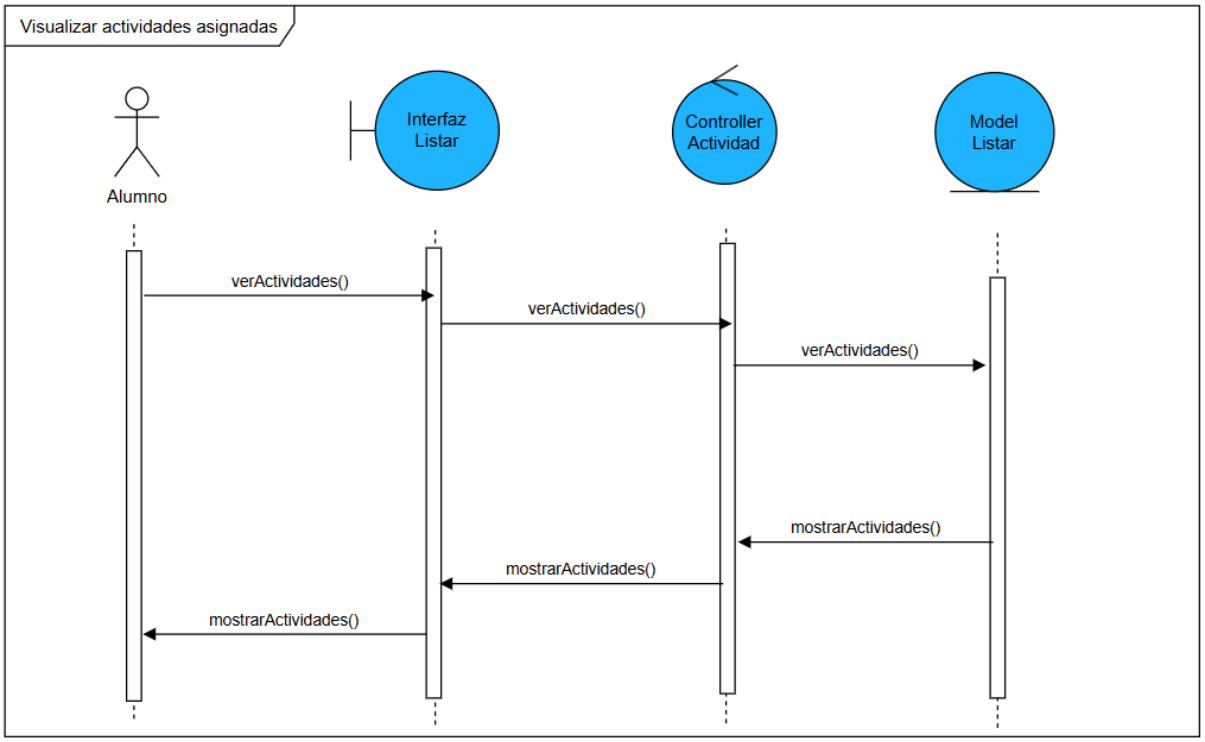

Figura 13. Diagrama de secuencia 8 (Visualizar actividades asignadas)

Enviar actividad completada

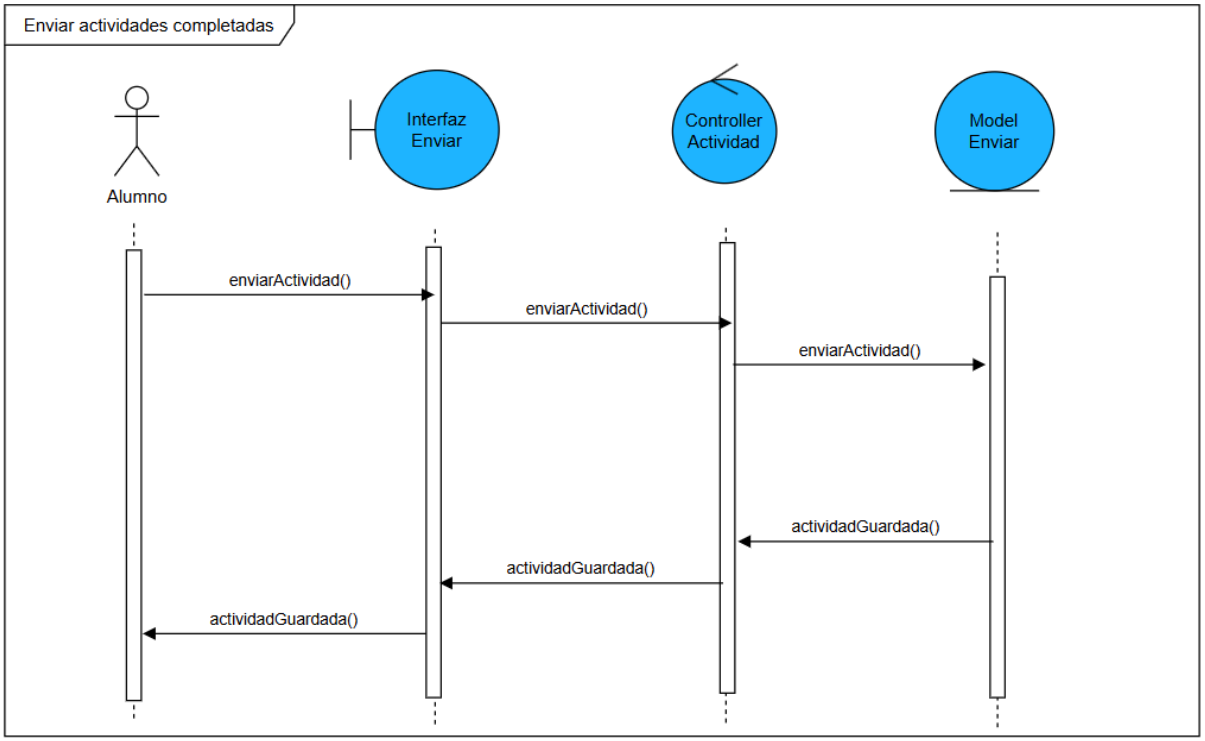

Figura 14. Diagrama de secuencia 9 (Enviar actividad completada)

Corregir actividad

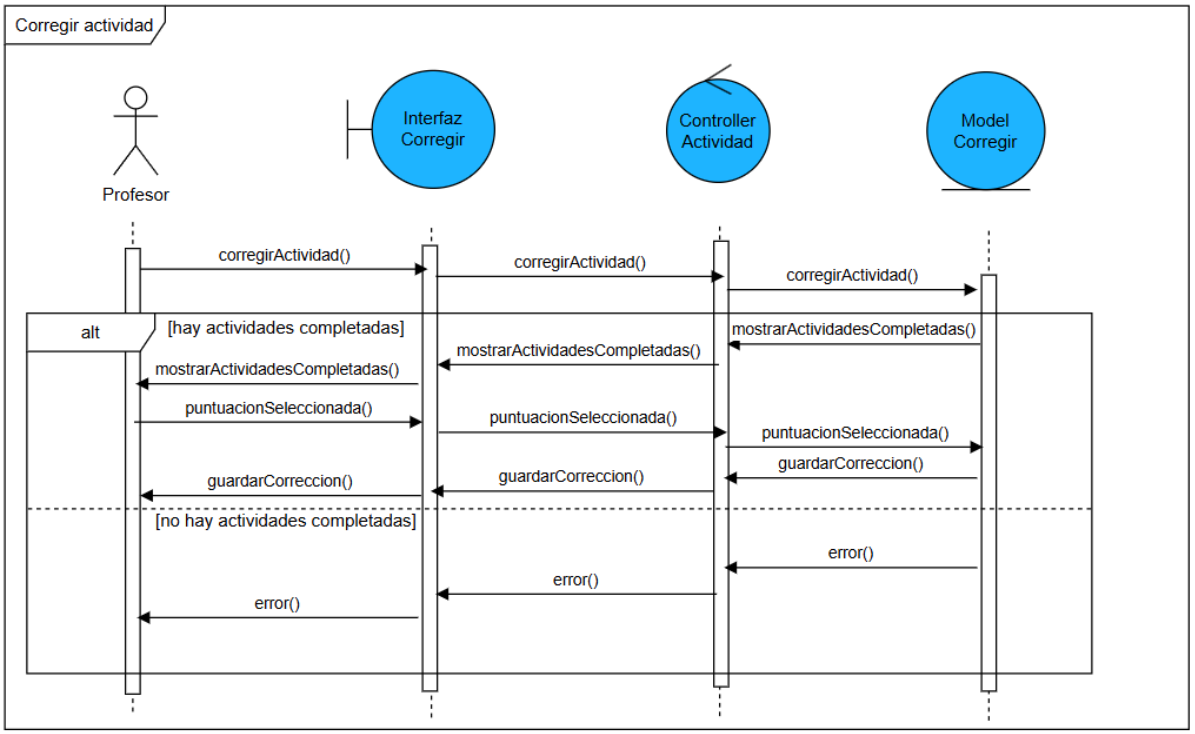

Figura 15. Diagrama de secuencia 10 (Corregir actividad)

Corregir actividad de casa

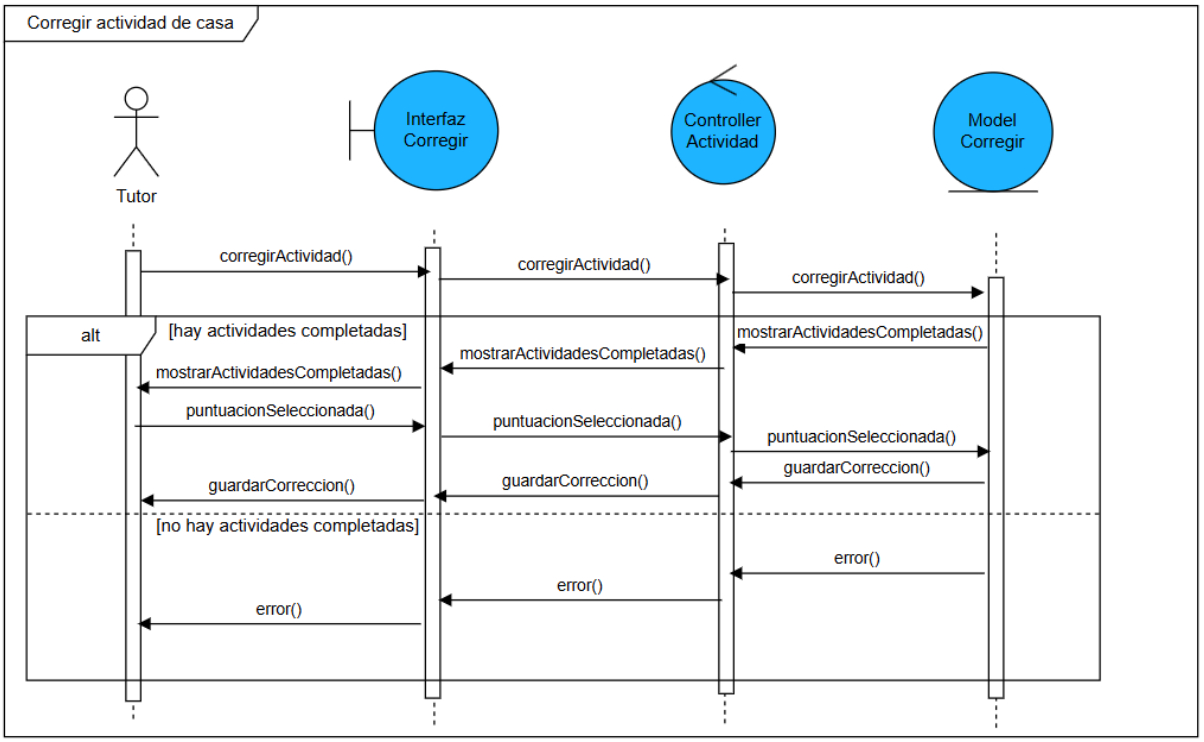

Figura 16. Diagrama de secuencia 11 (Corregir actividad de casa)

Visualizar ranking de alumnos

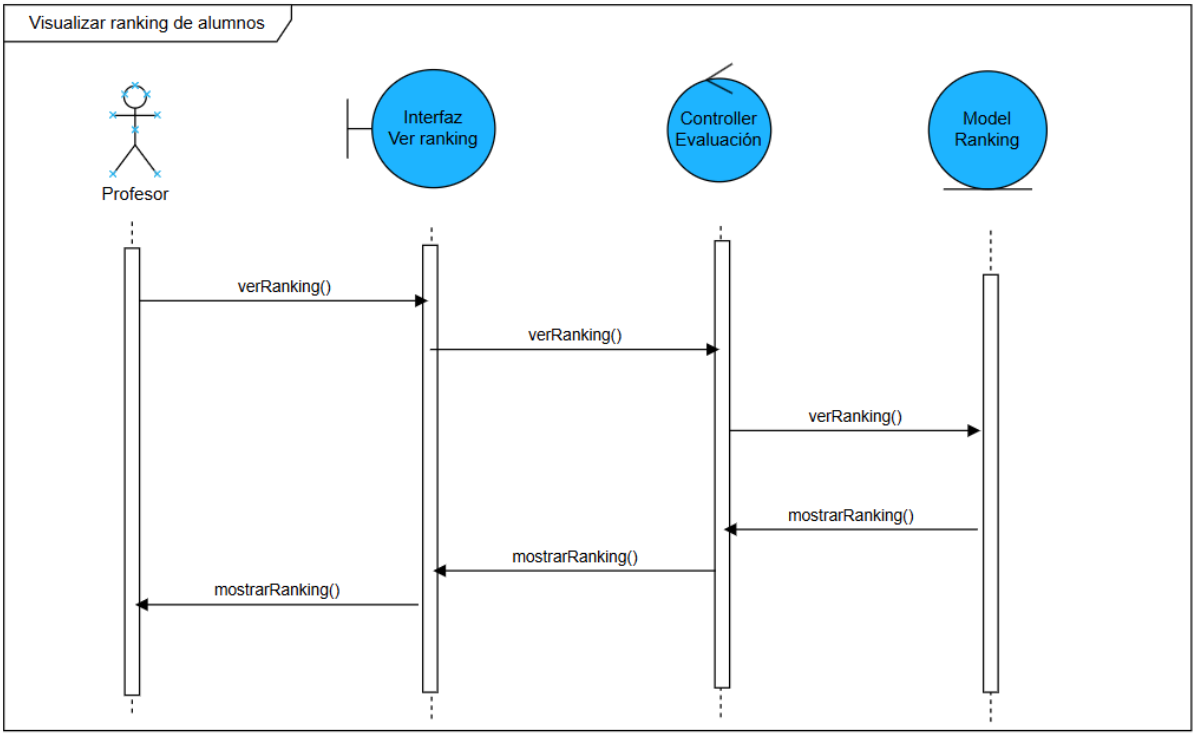

Figura 17. Diagrama de secuencia 12 (Visualizar ranking de alumnos)

Recibir aviso de tareas pendientes

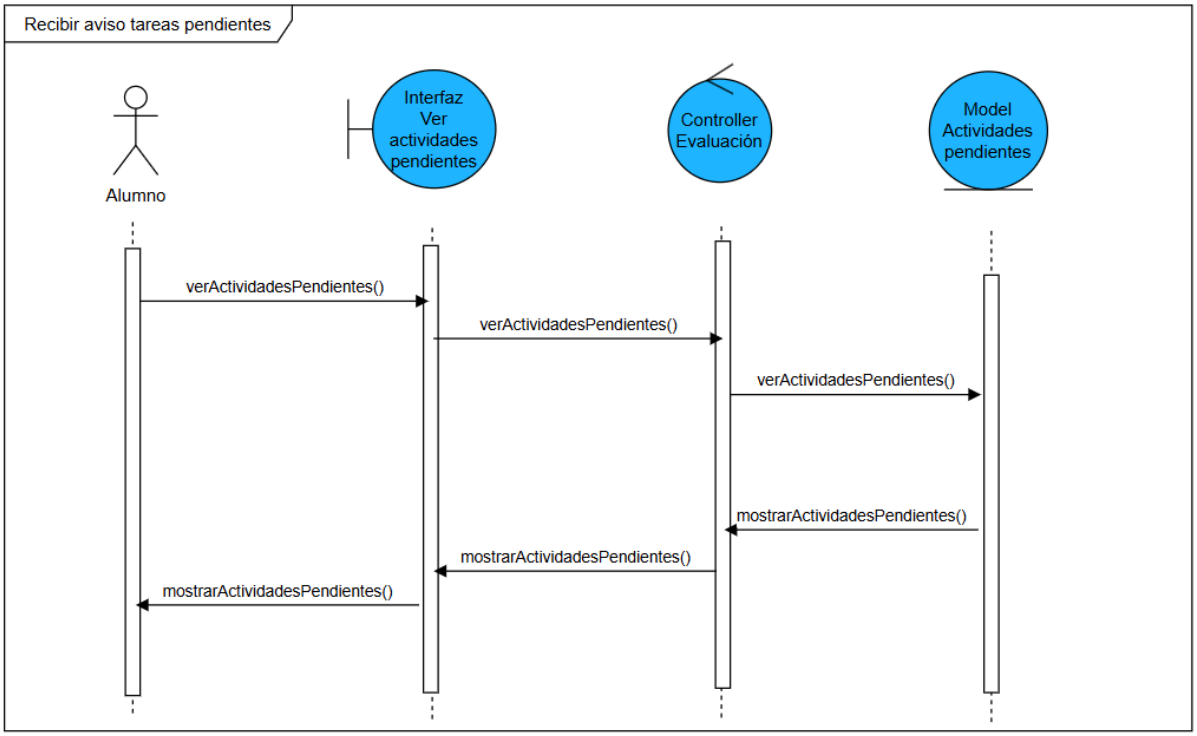

Figura 18. Diagrama de secuencia 13 (Recibir aviso tareas pendientes)

Ver corrección de actividad

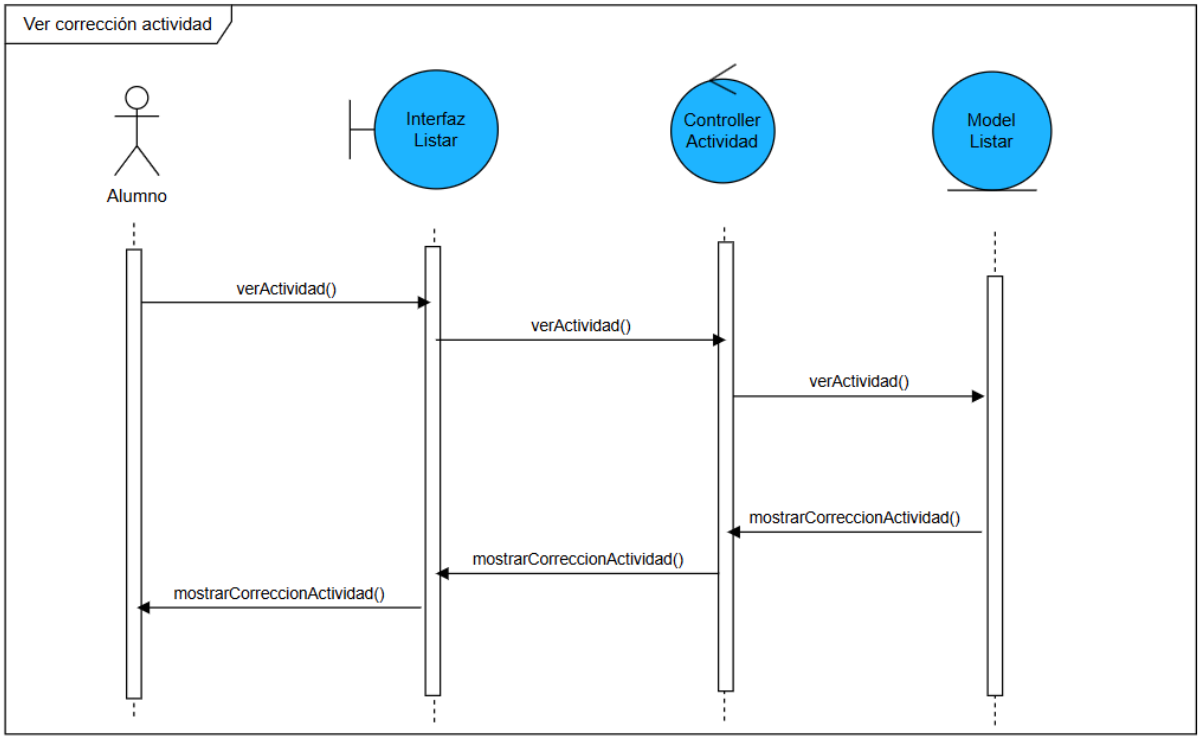

Figura 19. Diagrama de secuencia 14 (Ver corrección actividad)

Filtrar actividades por asignatura

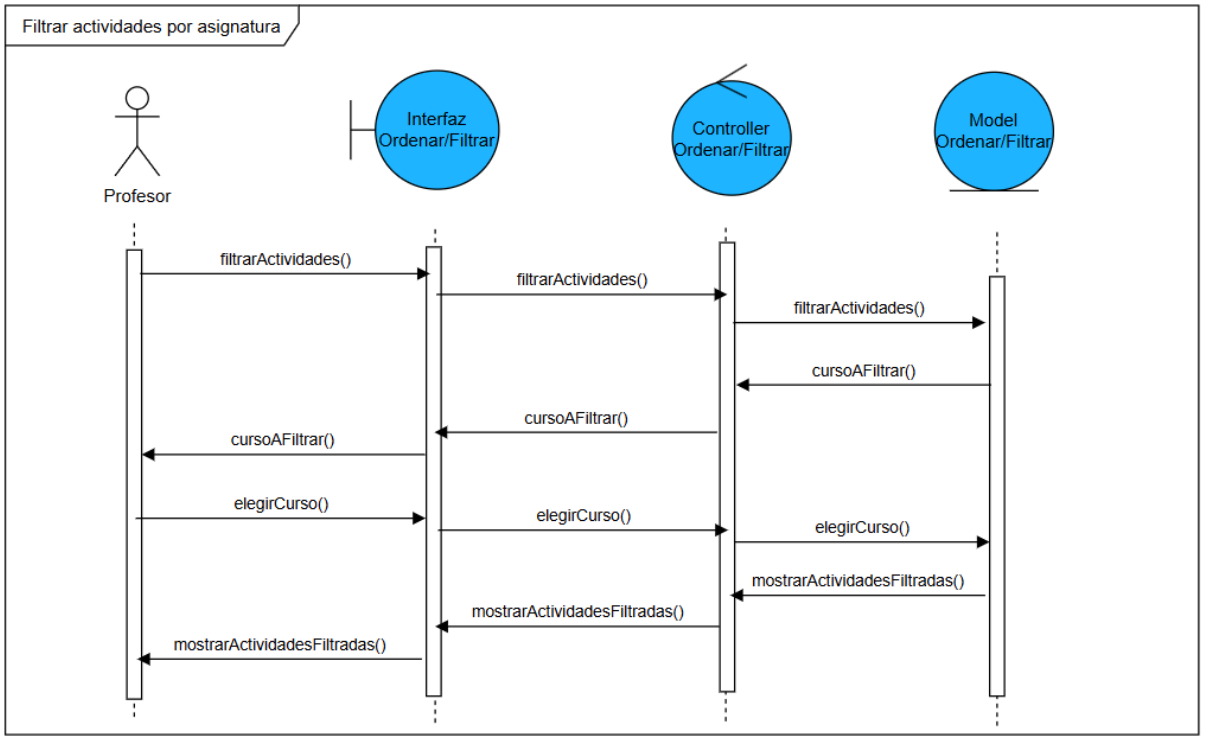

Figura 20. Diagrama de secuencia 15 (Registrarse)

Ordenar actividades

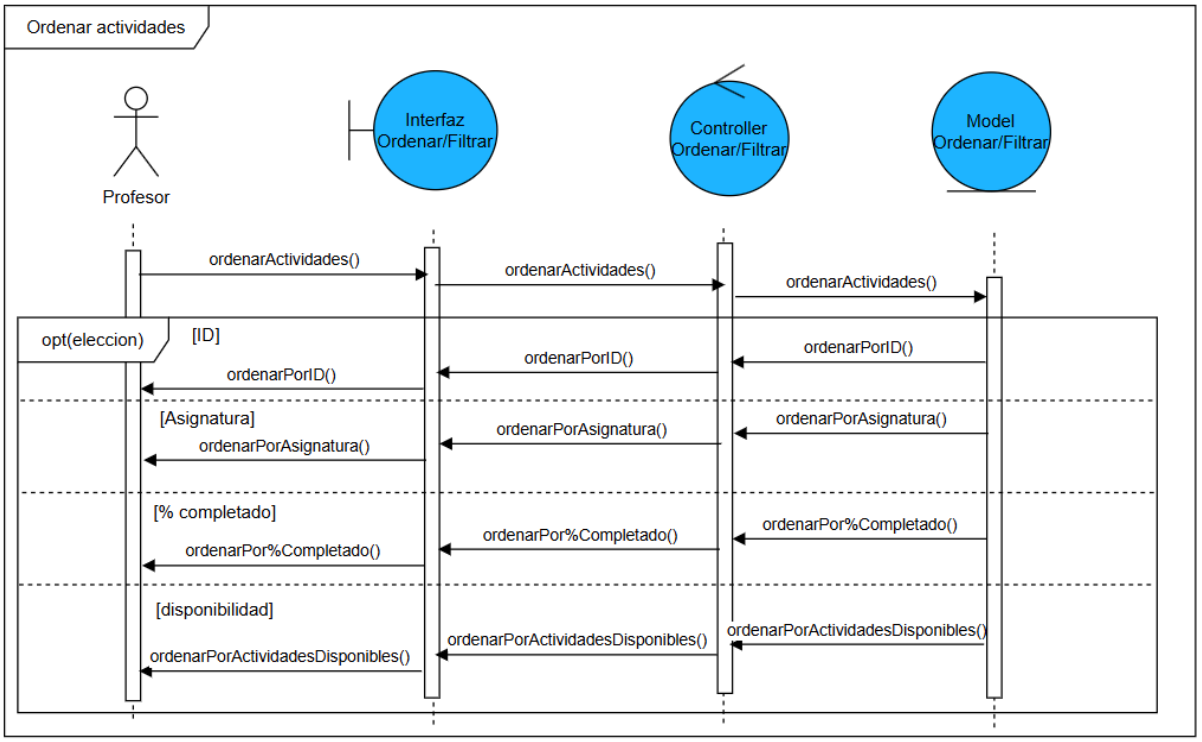

Figura 21. Diagrama de secuencia 16 (Ordenar actividades)

Cambiar vista de actividades (lista/cuadrícula)

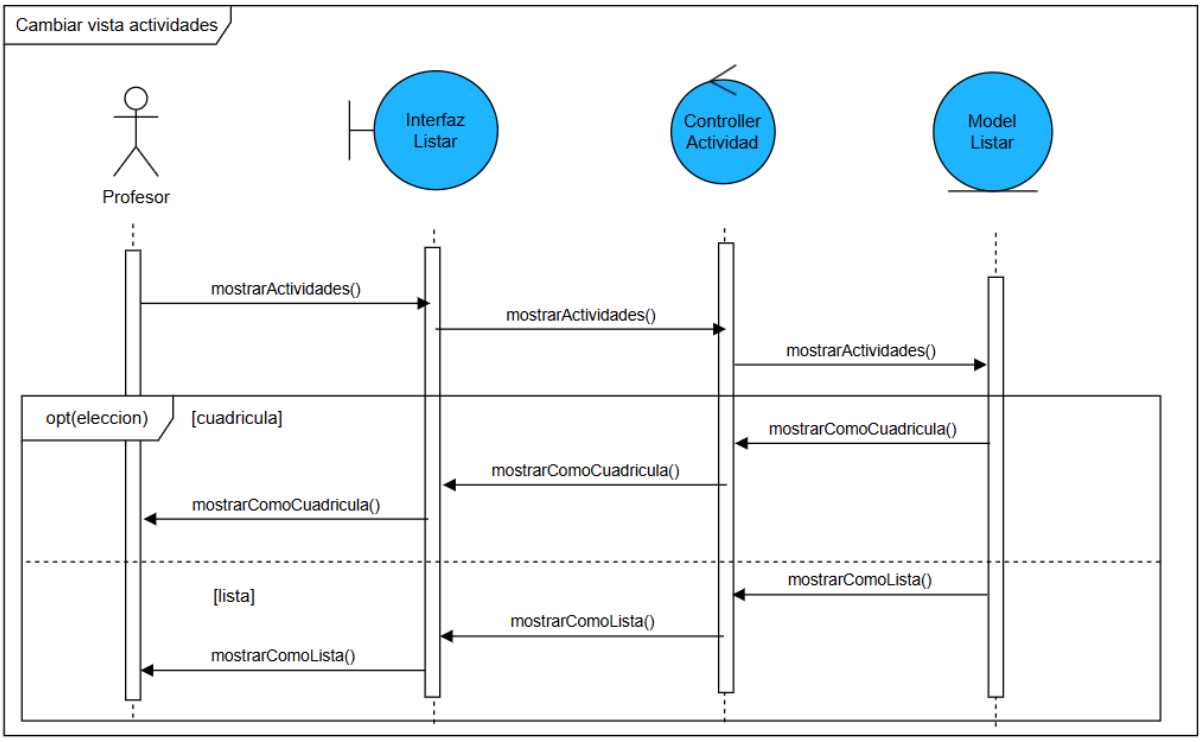

Figura 22. Diagrama de secuencia 17 (Cambiar vista de actividades)

Retroceder a pantalla anterior

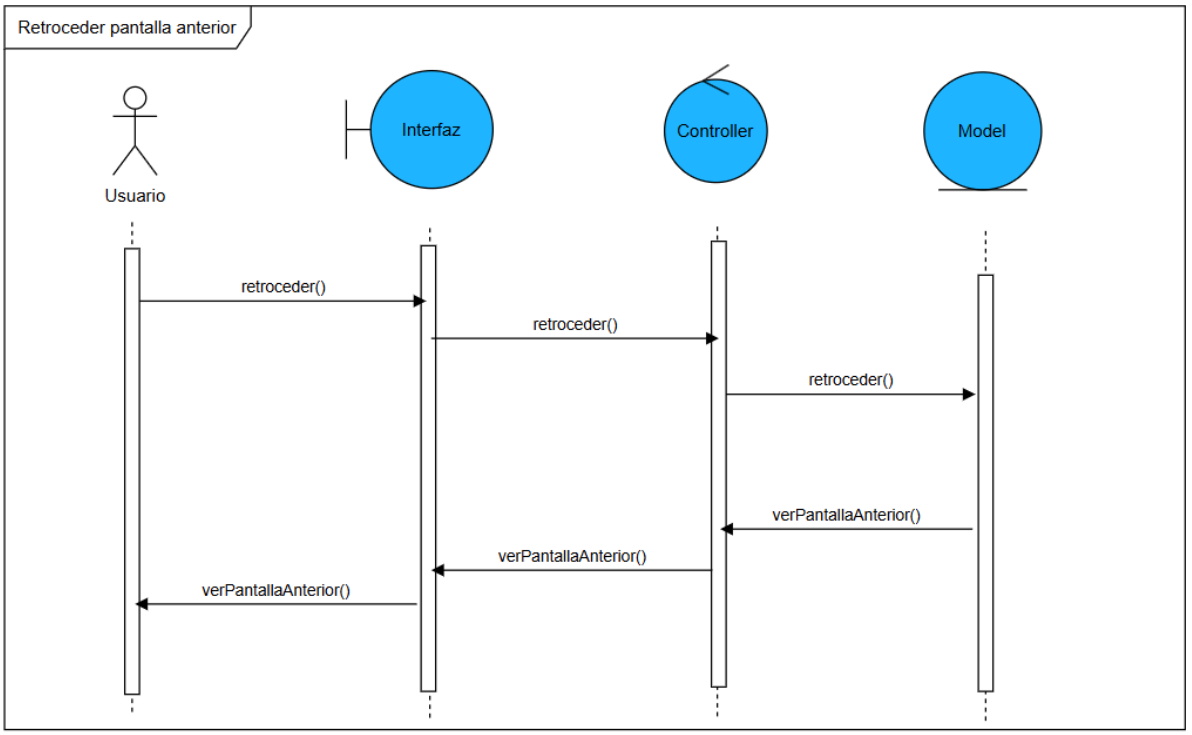

Figura 23. Diagrama de secuencia 18 (Retroceder a pantalla anterior)

Asignar curso a usuario

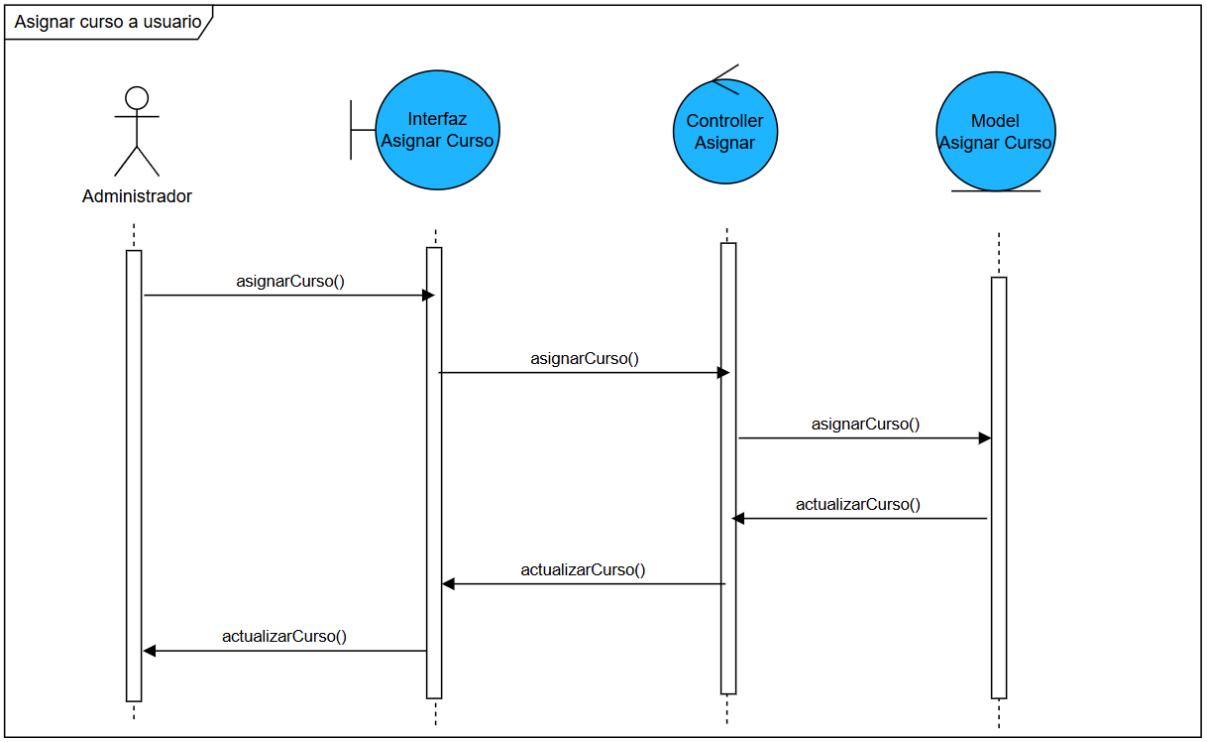

Figura 24. Diagrama de secuencia 19 (Asignar curso a usuario)

Asignar asignatura a usuario

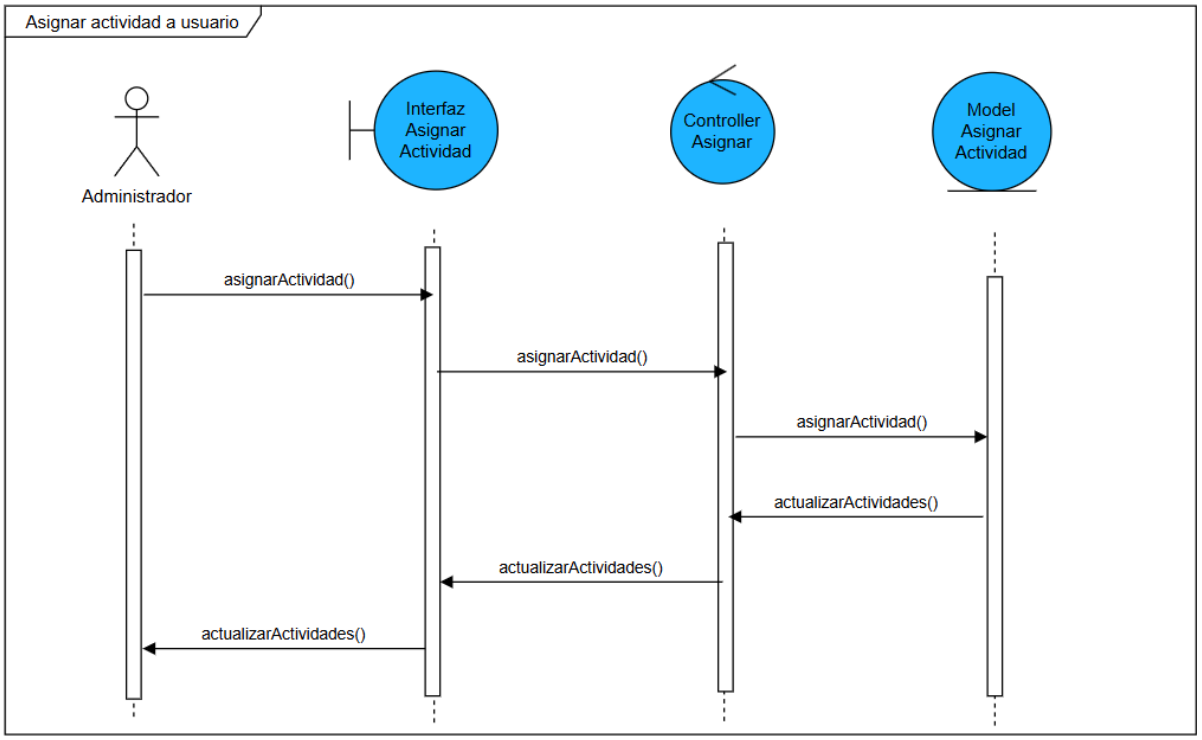

Figura 25. Diagrama de secuencia 20 (Asignar asignatura a usuario)

PROPUESTA DE ARQUITECTURA

En este proyecto, se ha optado por una arquitectura basada en el patrón Modelo-Vista-Controlador (MVC), distribuyendo las responsabilidades entre el cliente (Angular), el servidor (Node.js/Express) y la base de datos (MySQL). Esta estructura permite una separación clara entre la lógica de presentación, el procesamiento de las peticiones y el acceso a los datos.

A continuación, se presenta una propuesta de arquitectura general del sistema, detallando cómo se organizan sus componentes principales.

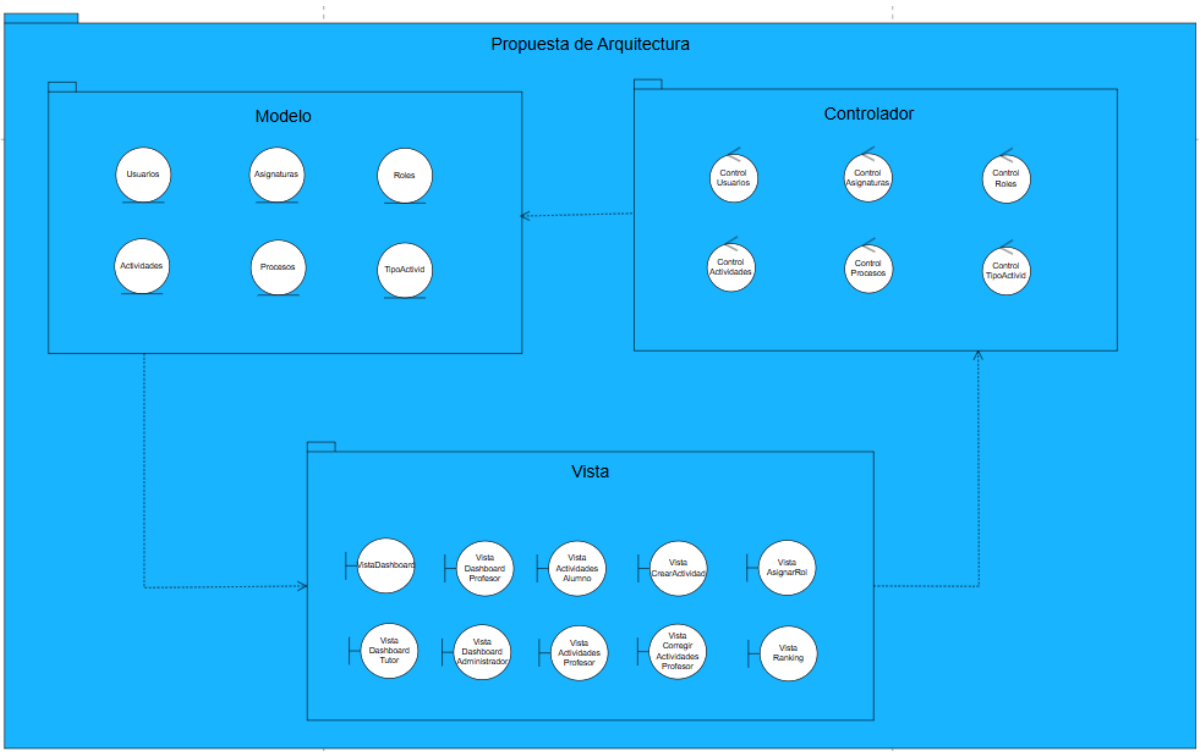

Figura 26. Propuesta de Arquitectura