Asignatura: Diseño y Programación Orientada a Objetos

Profesor: Germán Romero

Estudiante: Luis Ernesto Tejón **Código: 202113150**

Estudiante: Pablo Pedreros Díaz **Código: 202112491**

Estudiante: Alejandro Hernández **Código: 202111716**

Proyecto 1 Entrega 2

PRIMERA ITERACIÓN:

Rol(es):

2. El único componente en la primera iteración será la aplicación como tal. Sus **responsabilidades** serán:

- La aplicación tiene la responsabilidad de crear y guardar proyectos.
- La aplicación debe asignar participantes y dueños del proyecto y poder cambiarlos luego si se quiere.
- La aplicación debe dar un reporte de tiempo de trabajo de cada participante.
- La aplicación debe dejar que los participantes registren sus actividades.
- La aplicación se encarga de clasificar las actividades según su tipo.
- La aplicación crea un txt donde se guarda el proyecto y lo actualiza con cada actividad subida.
- La aplicación se encarga de asignar qué participante hizo qué actividad.
- La aplicación se encarga de iniciar, pausar y finalizar un cronómetro que mide cuánto tiempo se demoró en hacer una actividad.

SEGUNDA ITERACIÓN:

Elementos del dominio y roles:

- Los componentes y sus roles serán:

- Iniciar: Maneja y solicita la información al usuario sobre el proyecto, las actividades y los participantes. **(Coordinator).**
- **Proyecto (clase):** Define los atributos básicos del proyecto y delega responsabilidades a otras clases **(Coordinator).**
- **participante(clase):** Se encarga de subir nuevas actividades **(Structurer).**
- **Actividad (clase):** Define los cambios que tendrá el proyecto. **(Structurer).**
- **Reporte (clase):** Guarda información con respecto a los participantes **(Information holder)**
- **Hora (clase):** Guarda información con respecto al tiempo invertido en una actividad **(Information holder).**

Colaboraciones :

- Iniciar recibe información con respecto a la actividad que se va a subir y el participante que la realizó, la cual delega hasta proyecto, por otro lado proyecto delega la responsabilidad de subir una actividad a participante el cual actualiza la información en reporte y hora, por último, proyecto también delega la labor de actualizar el txt a Actividad.
- El txt del proyecto fluye desde la Actividad a Proyecto donde queda guardada.

<u>TERCERA ITERACIÓN:</u>

- En la tercera iteración, ahondamos en la estructura del programa, creando clases que se encargan de manejar la lógica, aumentando los information holders y centralizando el estilo de control. Especificamos las responsabilidades por clase y organizamos cómo se contendrán unas clases dentro de otras.

Elementos del dominio y roles:

- Los componentes y sus roles serán:

- Iniciar : Contiene el método main, que empieza preguntándole al usuario actual su nombre, pidiéndole a la lógica del programa que cargue la base de datos de los proyectos y mostrando el menú de opciones. Ya sabiendo qué usuario estamos usando, desde esta clase se recibirá una instrucción por consola como crear un proyecto, modificar un proyecto (que abrirá un menú con las posibilidades de crear actividad, modificar actividad o modificar datos del proyecto) o mostrar el reporte de un miembro del equipo. Además, tendrá la opción de cambiar el usuario actual **(Coordinator).**
- **ManejadorProyectos:** maneja la mayor parte de la lógica del programa. Recibe instrucciones de la consola y, con base a la instrucción que recibe ejecuta las

instrucciones para cambiar la información en los information holders o devolver información a la consola para que la imprima. Contiene una lista de proyectos que se cargan cuando empieza el programa e información del usuario actual para saber qué proyectos están a su disposición **(Controller).**

- **Proyecto (clase):** Contiene información del nombre, descripción, fecha de inicio y fecha estimada de finalización del proyecto. Además, contiene información de qué miembros de la organización son participantes del proyecto y cuál es el dueño. Por último, contiene una lista de todas las actividades que los participantes han registrado para este proyecto. Debe tener también un método que imprima o actualice toda su información en un archivo .txt que represente ese único proyecto haciendo uso de otros métodos como getters **(Information holder).**
- **Participante (clase):** tendrá la información del nombre y correo del participante, además de una lista de proyectos de los que hace parte, y una lista de actividades que ha realizado que servirán para que calcule y genere un reporte con los datos de sus actividades y de su rendimiento **(Information holder).**
- **Actividad (clase):** contiene la información de una actividad con su nombre, tipo, fecha y hora de inicio y finalización, descripción y participante que la realizó **(Information holder).**
- **Reporte (clase):** contiene la información de las actividades que ha hecho un miembro del equipo, incluye también información de su rendimiento como tiempo total invertido, tiempo promedio por tipo de actividad, tiempo por cada día, etc **(Information holder).**
- **Cronómetro (clase estática):** la clase funciona como un cronómetro que cuenta el tiempo para hacer una actividad, puede pausarse o reanudarse y calculará el tiempo que se tomó haciendo la actividad **(Service provider).**

Colaboraciones:

- La clase iniciar recibe del usuario la información de qué instrucción quiere realizar y le informa a la clase **ManejadorProyectos** para que ejecute la lógica.
- **ManejadorProyectos**, dependiendo de qué instrucción debe realizar, modifica los atributos de las clases **Proyecto, Participante** y **Actividad** que está usando en ese momento. Por ejemplo, si se crea un nuevo proyecto tendrá que agregarse el **Participante** que lo crea como dueño y tendrá que agregarse el proyecto a la lista de proyectos del participante. En otro ejemplo, si se registra una actividad habrá que

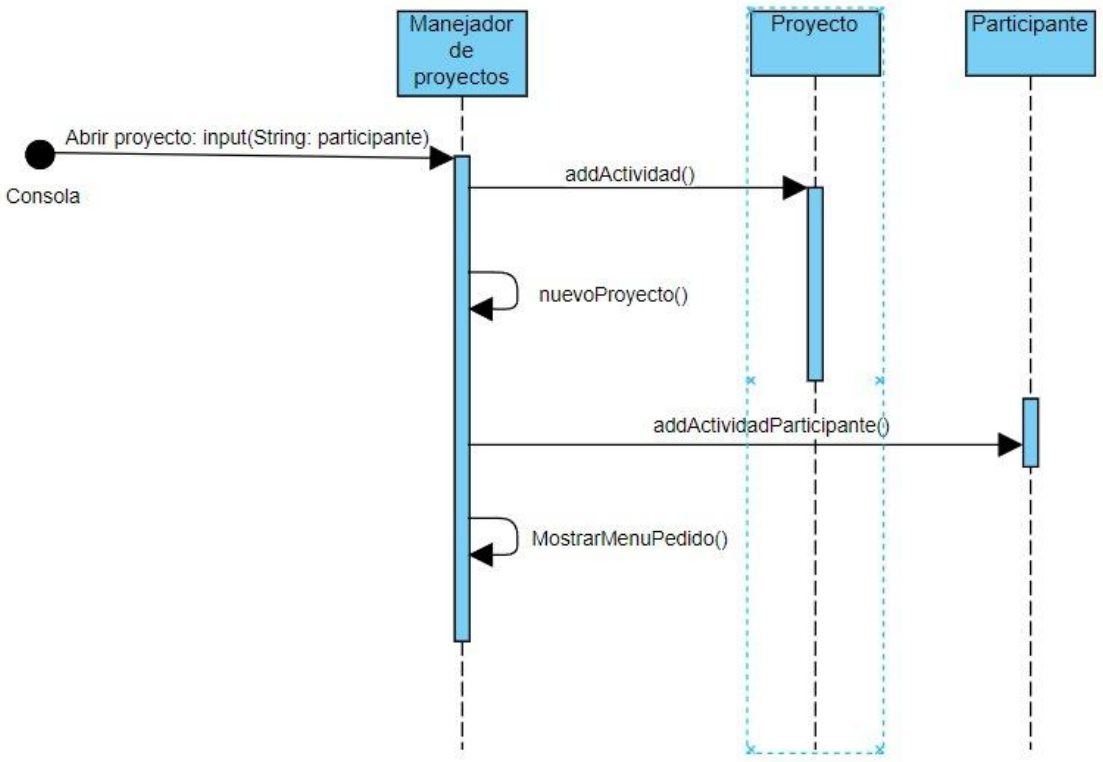
agregarla a la lista de actividades del participante que la realiza y a la lista de actividades del proyecto.

- Al crear un reporte, la clase **Participante** usa los datos de su lista de **Actividades** para generar una instancia de la clase **Reporte** que contenga todos los datos del participante en el momento que genera el reporte.

DIAGRAMA DE SECUENCIA DE LOS ROLES:

Hay dos ocasiones en la que los roles colaboran entre ellos:

- **Cuando alguien publica una nueva actividad.**
- **Cuando se solicita un reporte de un miembro.**



REFLEXIONES Y TRADEOFFS:

1. El diseño que terminamos haciendo es de un estilo de control centralizado, lo que hará mucho más fácil saber en qué clase sucede un posible error, aunque puede hacer un poco más difícil la organización de trabajo para los miembros del equipo, pues la lógica se concentrará en una clase y habrá más espacio a problemas a la hora del merge.
2. El uso de archivos .txt para guardar los datos de los proyectos tendrá un costo de implementación y de memoria, pero facilitará mucho el movimiento de datos del proyecto entre diferentes computadores y permitirá conservar la información entre diferentes ejecuciones del programa.

*Los diagramas están adjuntos en el mismo paquete.