Universidad de los Andes

Diseño y Programación O.O.

Proyecto 3: Entrega 2

Daniel Arango – 202110646
Laura Daniela Arias – 202020621
Diego Alejandro González – 202110240

1. Contexto del problema:

Para el caso de estudio de este documento de diseño se plantea una aplicación que tenga como principal objetivo el seguimiento de proyectos realizado por una o varias personas siguiendo una estructura WBS. Así las cosas, y antes de iniciar con la labor de diseño fino de cada una de las componentes que deberían estar presentes en la elaboración de esta aplicación, se plantea un modelo sencillo que permita a través de una pequeña pieza visual la definición de las funcionalidades de alto nivel que deben poderse ejecutar o llevar a cabo al tener interacción con la interfaz gráfica. Lo anterior se hace con el objetivo de definir las capacidades tanto de la interfaz como de la aplicación en general, entendiendo que ambos elementos componen esta aplicación. A continuación, se presenta la piza visual mencionada con los componentes base que definen el problema de manera global:

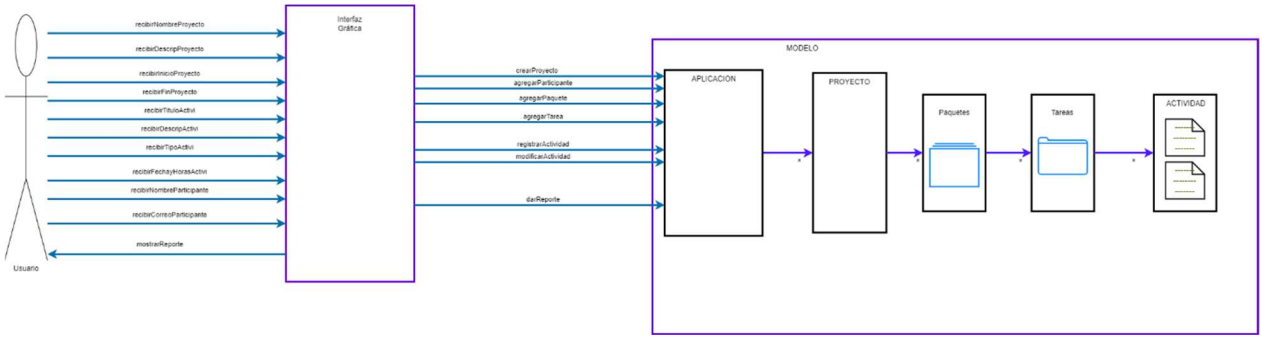


Figura: Para mejor visualización, se recomienda la apertura de la imagen en el siguiente enlace: https://acortar.link/h1U85W

Antes de comenzar a hablar de dicha interacción del usuario con la interfaz gráfica, es necesario aclarar que es el modelo WBS (Work BreakDown Structure), la cual es una herramienta de gestión que permite describir todas las tareas que se deben realizar dentro de un proyecto, planear el tiempo que tardarán, y hacerle seguimiento a su avance. Todo esto mediante una estructura jerárquica que permita por medio de algún criterio separar los objetos y ordenarlos según dicho criterio, puede ser por importancia, por tipo de tarea, entre muchos otros. Para este caso, el WBS de la aplicación esta estructurado de manera tal que el nivel más alto en la jerarquía son los paquetes quienes dentro de si pueden contener otros paquetes o tareas, y a su vez las tareas pueden contener ninguna o muchas actividades las cuales determinaran el progreso de dicha tarea y a medida que se cumplan tareas se avanzara en el progreso de los paquetes y por ende el progreso del proyecto en general. Esto permite que el avance dentro del proyecto pueda analizarse desde sus pequeñas responsabilidades hasta el como va en general todo el conjunto de responsabilidades.

Para el caso de la interacción del usuario con la interfaz gráfica, este debe proporcionar toda la información para poder generar los registros de cada uno de los componentes del proyecto. Así

pues, se reciben para el proyecto: nombre, descripción, fecha de inicio, fecha de finalización y elementos esenciales para la creación del paquete raíz del proyecto como el identificador de este. Es así como,
tanto para la creación de paquetes como de tareas se recibe, en cual paquete se desea almacenar, un identificador único del paquete o tarea a crear, un nombre y descripción. Además, para la actividad se recibe del usuario: Título, descripción, fecha, hora de inicio, hora de finalización, identificador y un indicador que le pregunta al usuario si con esa actividad finaliza la tarea. Finalmente, para el participante se recibe del usuario: nombre y correo.
Por otra parte, para la interacción entre interfaz y modelo, se deben tener en cuenta los procesos macro que debe realizar la aplicación, con sus respectivos parámetros. En este sentido, los procesos identificados fueron:

- La creación del proyecto
- La agregación de un participante a un proyecto
- La creación de un paquete
- La creación de una tarea
- La modificación de un paquete
- La modificación de una tarea
- La creación de una actividad
- La modificación de una actividad
- La generación de un reporte.

1.1. Responsabilidades:

De forma inconsciente, a la hora de considerar los nueve componentes candidatos que se mencionaron anteriormente, se pensó también en algunas de las responsabilidades de las cuales se harían cargo. La siguiente tabla presenta en detalle cada una de las responsabilidades y el componente asociado que debe asumirlas**.**

#	Responsabilidad	Componente
1	Iniciar Aplicación	Interfaz
2	Crear Actividad	Controlador Actividades
3	Modificar Fecha Actividad	
4	Modificar Participante Actividad	
5	Modificar Hora Inicio Actividad	
6	Modificar Hora Fin Actividad	
7	Agregar Participante	Controlador Proyecto
8	Crear Proyecto	
9	Calcular tiempo actividad	Cronómetro
10	Calcular tiempos pausas	
11	Calcular Puntos	Participante
12	Dar reporte participante	
13	Calcular tiempo total invertido	
14	Calcular tiempo promedio en tipo	
15	Calcular tiempo por día	
16	Crear Participante	
17	Añadir paquete	Estructurador WBS
18	Añadir tarea	
19	Modificar paquete	
20	Modificar tarea	
21	Reporte avance	

De la tabla cabe aclarar que la interfaz gráfica al formar parte activa de la recolección de la información base para la conformación de cada uno de los componentes, va a estar presente en muchos más procesos de los mencionados anteriormente. Así mismo, para el caso del estructurador WBS, este "contiene" componentes como las actividades y los participantes, por lo que también participará en varios procesos asignados a estos 2, como se verá más adelante.

2. Nivel 2

2.1 . Interfaz gráfica:

Definimos un *interfacer* en *"Interfaz Gráfica"*, el cual, como lo indica su estereotipo, se encarga de transformar información y peticiones entre diferentes partes del sistema. Puede que este elemento, al no llevar a cabo acciones más allá de interactuar con el resto del programa para que los demás componentes puedan hacer lo que deben hacer, presente características que se le pueden atribuir a un *controller*. Sin embargo, se dice que la labor de *interfacer* supera a esta otra, no solo por el nombre del componente, sino porque efectivamente este transforma la información ingresada por el usuario al sistema al igual que las peticiones, pasándolas entre los componentes de diálogo y panel, y las clases que conforman el modelo del problema.

3.1.1. Componentes y estereotipos

Este elemento del diseño se encuentra compuesto por varios paneles y diálogos emergentes los cuales facilitan la interacción con el usuario a la hora de solicitarle información y delegarla a componentes propios del modelo. Para lograr esto, la interfaz cuenta con un JFrame en el cual están contenidos todos los paneles con los cuales el usuario podrá interactuar. Es así como, se planteó una funcionalidad particular la cual activa o desactiva la visibilidad de un panel en función a lo que el usuario desee ver (esto emulando lo que podría ser una aplicación móvil). Esto quiere decir que el usuario únicamente estará viendo un panel a la vez, y lo que determinará que panel visualizará serán las interacciones que el usuario tenga con la interfaz al presionar botones para navegar por los paneles. Es por ello que, la interfaz estará compuesta por varias clases, siendo la principal la clase interfaz (Un JFrame) que se conecta a otras clases (paneles o diálogos) como: Panel inicio, Panel crear proyecto, Panel home, entre otros. De esta forma, el usuario podrá interactuar con la los diferentes paneles y sus componentes para direccionar la información o peticiones suministradas al modelo de la aplicación.

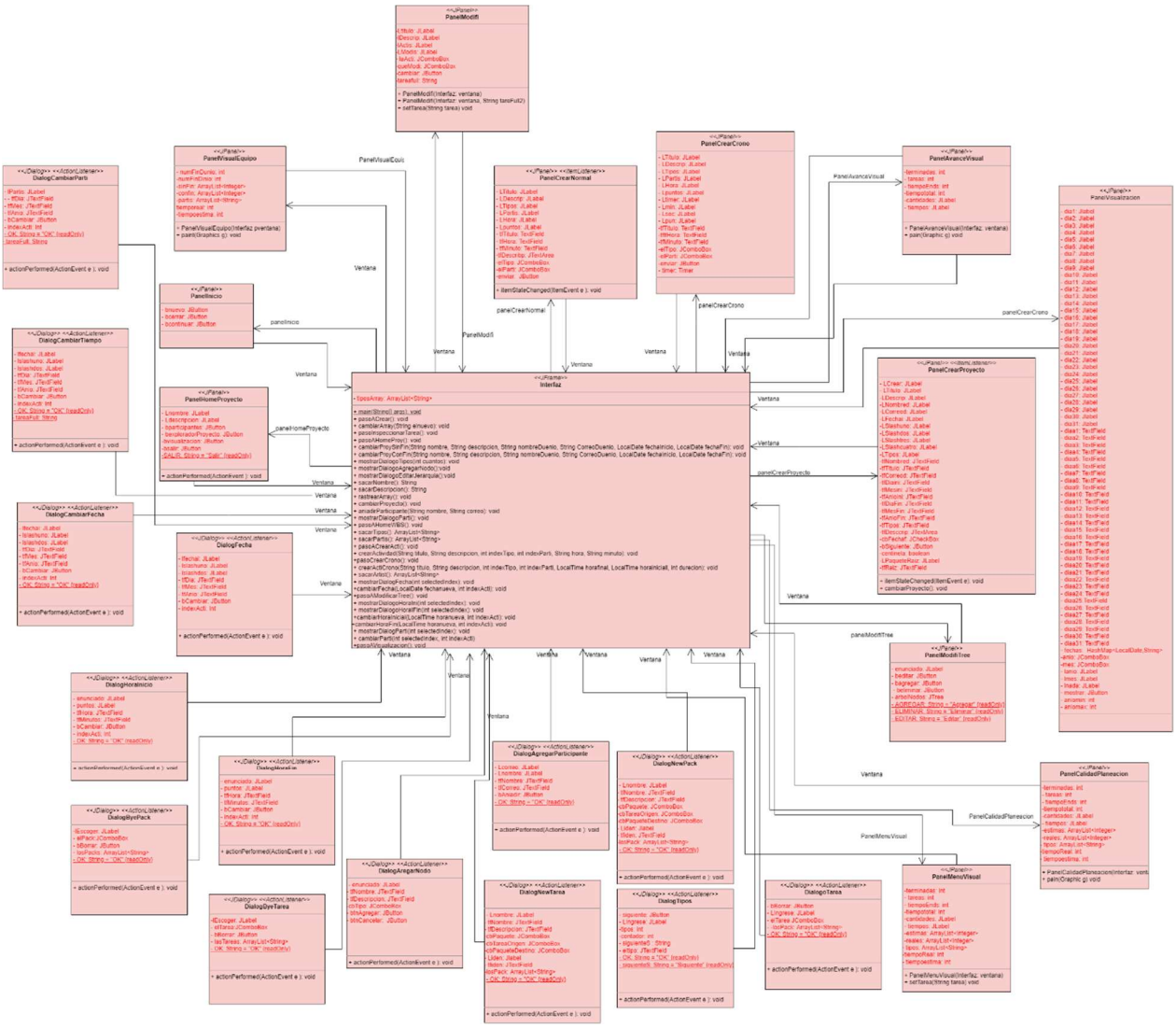


Figura: Para mejor visualización, se recomienda la apertura de la imagen en el siguiente enlace: https://acortar.link/dTfiyt

3.1.2. Responsabilidades y colaboraciones

La interfaz gráfica implementada se encuentra compuesta por un elemento JFrame, a los cuales se conectan diversos JDialogs y JPanels para formar la parte visual de la aplicación. Estas relaciones son las que conforman las colaboraciones dentro de este componente de la aplicación. El JFrame Interfaz, el cual como tal está conectado al modelo del problema, transfiere toda la información necesaria a los páneles y diálogos relacionados a cierta actividad a realizar para que luego este vuelva a pasar la información a los elementos del modelo y muestre al usuario la información requerida.

2.2Controlador actividades:

De la misma manera, se necesita un componente que maneje todas las acciones relacionadas con las actividades que se realizan dentro del proyecto. Debido a la magnitud de estas, también se le define su propio *controlador*, el cual llamaremos *"Controlador Actividades"*.

3.2.1. Componentes y estereotipos

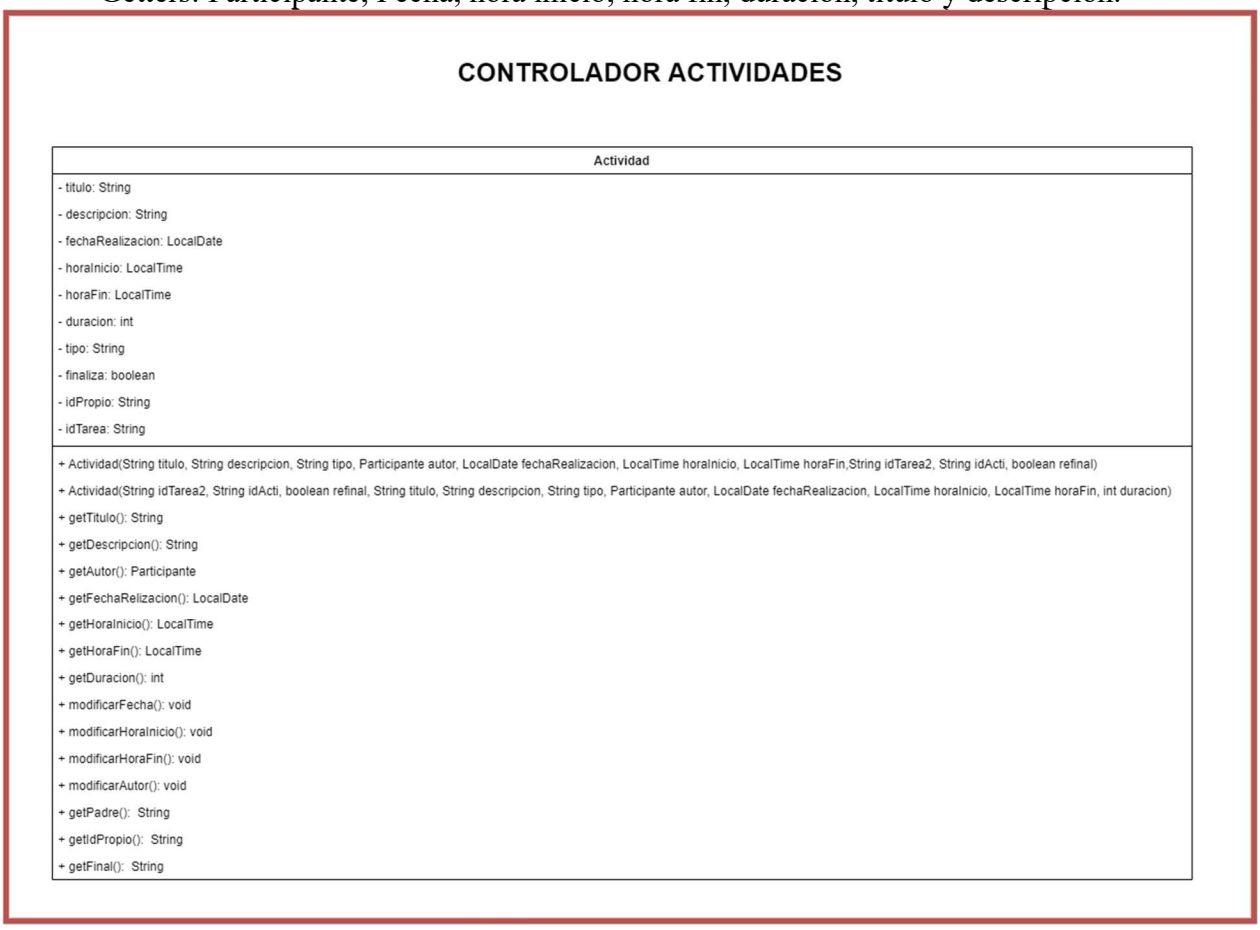
Para el caso del controlador de actividades, se hizo de especial importancia la consideración de elementos clave del enunciado como lo son todas las características que componían cada una de las actividades. Además, se debió tener en cuenta para los métodos todas las posibles opciones que le permitían al usuario interactuar directamente con cada una de las actividades.

Así las cosas, en lo que se refiere a los atributos de esta clase, se tienen los siguientes:

- Datos base dados por el usuario: Título, descripción, id propio y id de tarea a la que pertenece.
- Datos modificables: Participante, Fecha, Hora de inicio, hora de fin y identificador de finalización.
- Datos calculados: Duración

Finalmente, en lo que se refiere a los métodos de esta clase, se tienen los siguientes:

- Métodos base: constructores
- Modificaciones: Participante, Fecha, Hora de inicio y hora de fin
- Getters: Participante, Fecha, hora inicio, hora fin, duración, título y descripción.



3.2.2. Responsabilidades y colaboraciones

Contrario al caso anterior, en esta oportunidad se sabe que se condensó toda la actividad e información de este componente en una sola clase, lo que cohíbe enseguida todas las posibles colaboraciones internas que puedan llegar a existir. Sin embargo, y como se ha mencionado en las ocasiones anteriores, las colaboraciones con otros componentes siguen intactas.

2.3Controlador proyectos:

En el programa, se requiere un componente encargado de controlar todos los eventos relacionados a lo que ocurre dentro del proyecto, por lo que se define uno específicamente encargado de esto. Este componente lo denominamos "*Controlador Proyectos"*.

3.3.1. Componentes y estereotipos

Para el caso del controlador, se debe tener en cuenta elementos clave que salen directamente del enunciado del ejercicio, y es que solo se puede hacer seguimiento a un proyecto a la vez, lo que simplifica el problema del controlador de proyectos. Sin embargo, como este elemento se compone de otros como la clase paquete, sus actividades van a estar bastante relacionadas con el control de los elementos y la asignación de labores a cada uno de ellos. Así las cosas, para el caso de los métodos, encontramos lo siguiente:

- Constructores: Básicos para la creación del proyecto sobre el que se va a hacer seguimiento.
- Relacionado con paquetes: control de la creación de paquetes y sus parámetros de ingreso.
- Relacionado con los participantes: Manejo y control de los mismos, además de la creación de nuevas instancias en caso de ser necesario
- Getters: Permitirán imprimir atributos y resultados en la interfaz

Para el caso de los atributos, se hace uso explícito del enunciado del problema, el cual habla de lo siguiente: nombre del proyecto, descripción de este, fecha de creación y finalización, y miembro generador o dueño del proyecto, paquete raíz.

3.3.2. Responsabilidades y colaboraciones

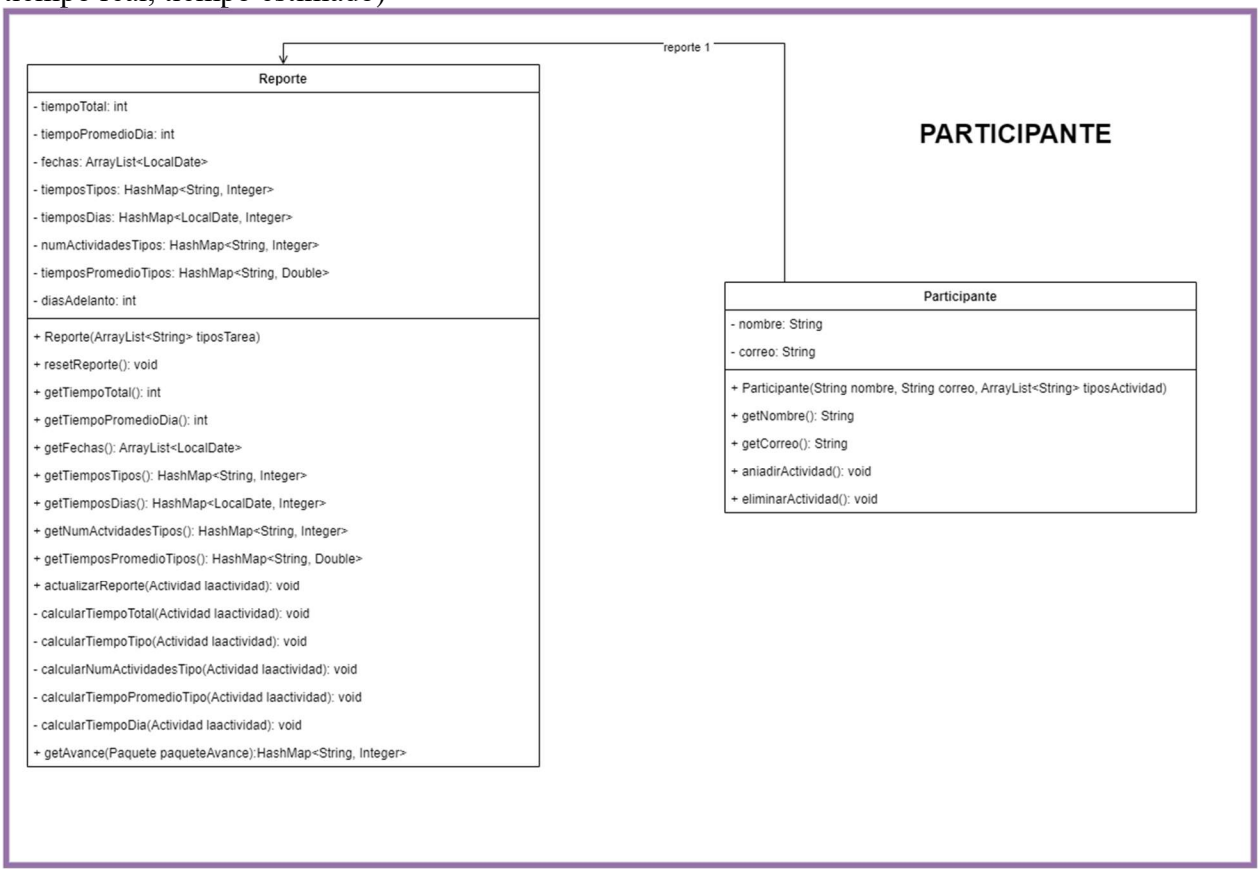
Una vez más, como se pudo observar tanto en la descripción anterior como en la gráfica planteada, se sabe que se condensó toda la actividad e información de este componente en una sola clase, lo que cohíbe enseguida todas las posibles colaboraciones internas que puedan llegar a existir. Sin embargo, y como se ha mencionado en las ocasiones anteriores, las colaboraciones con otros componentes siguen intactas.

2.4Participante:

Adicionalmente, contamos con *"Participante"* como *information holder*. Este componente surge de tener que manejar información sobre los participantes del proyecto y actualizar esta información conforme hay avances.

3.5.1. Componentes y estereotipos

Para el caso del participante, se tuvo en especial cuidado el enunciado que menciona la necesidad de la generación de unos reportes que correspondan a la labor y aporte de cada uno de los integrantes del proyecto en la construcción del mismo. Así las cosas, se hizo necesario la creación de 2 clases: Una que conservara los datos esenciales del usuario y otra que tuviera un reporte listo para el participante en cuestión con cada uno de los elementos mencionados de cuidado en los reportes (tiempo total invertido, tiempo promedio por tipo de actividad, tiempo diario gastado, tiempo real, tiempo estimado)



3.5.2. Responsabilidades y colaboraciones

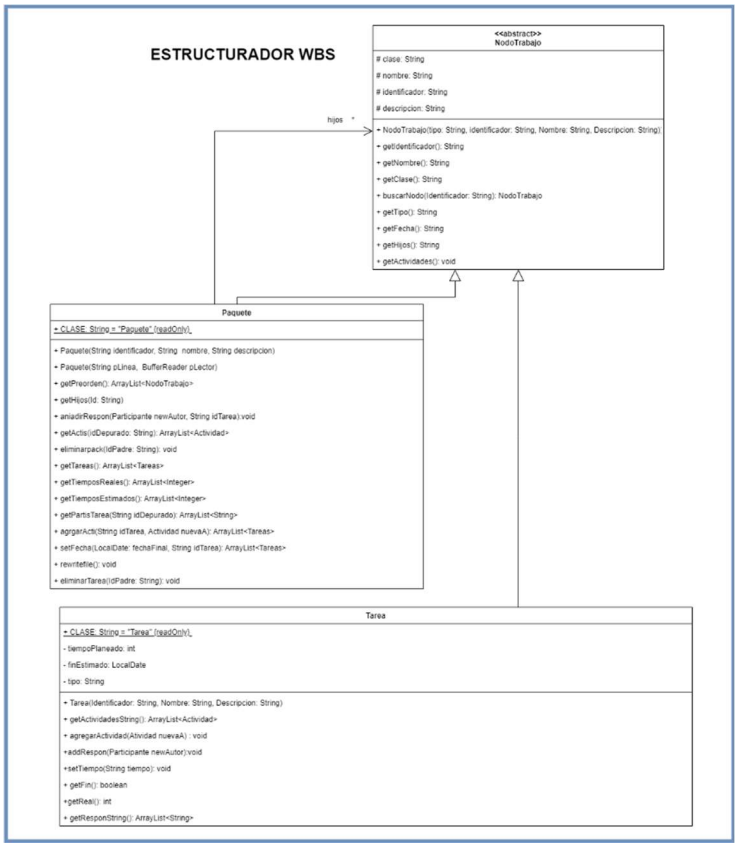
En este caso puntual, al haberse generado 2 clases asociadas entre ellas si se pueden considerar las colaboraciones entre sí. En este sentido, cuando se crea un participante nuevo, se le asigna un reporte negativo donde inicializa todos sus componentes en 0´s, mientras que, en el caso de la solicitud de un reporte específico, se hace necesaria la contribución de la clase participante con su información base.

2.5Estructurador WBS

Finalmente, contamos con "*Estructurador WBS"* como *Structurer.* Este componente surge de buscar cumplir con la jerarquía WBS previamente descrita para que a su vez dichas relaciones permitan dar información de cómo va el proyecto.

3.6.1Componentes y estereotipos

Para el caso del estructurador WBS, fue necesario idear cuidadosamente una manera de representar la jerarquía entre componentes de manera tal que fuesen compatibles entre si y a la vez pudiesen ser organizados teniendo en cuenta algún criterio en específico. De esta forma, se pensó en el modelo de un árbol n-ario en el cual cada uno de los nodos cumpliría con ciertas características en común, pero se diferenciarían por sus responsabilidades dentro del proyecto. De allí se obtuvo la idea de dividir este estructurador en 3 clases. Una clase capaz de representar esa abstracción de nodo dentro del árbol n-ario a construir que contara con las características y funcionalidades mínimas que cualquier nodo dentro del árbol debería cumplir. De esta abstracción surgen tanto la clase paquete como la clase tarea, las cuales se encargan de administrar la gestión de participantes y actividades dentro del proyecto para poder llevar un registro del avance alcanzado hasta el momento.



3.6.2 Responsabilidades y colaboraciones

En este caso puntual, al haberse generado 3 clases asociadas entre ellas si se pueden considerar las colaboraciones entre sí. En este sentido, tanto la clase paquete como la clase tarea, son abstracciones de la clase NodoTrabajo la cual es una clase abstracta que plantea el comportamiento esperado de cualquier nodo dentro de la estructura WBS. De esta forma, tanto la clase paquete como la clase tarea son nodos de trabajos los cuales construyen un árbol n ario, donde por medio de identificadores se construyen relaciones de jerarquía entre los componentes, para que de esta forma se puedan hacer recorridos y los hijos conozcan de sus padres al igual que los padres la información de sus nodos hijos.

3. NIVEL 3: DISEÑO FINAL

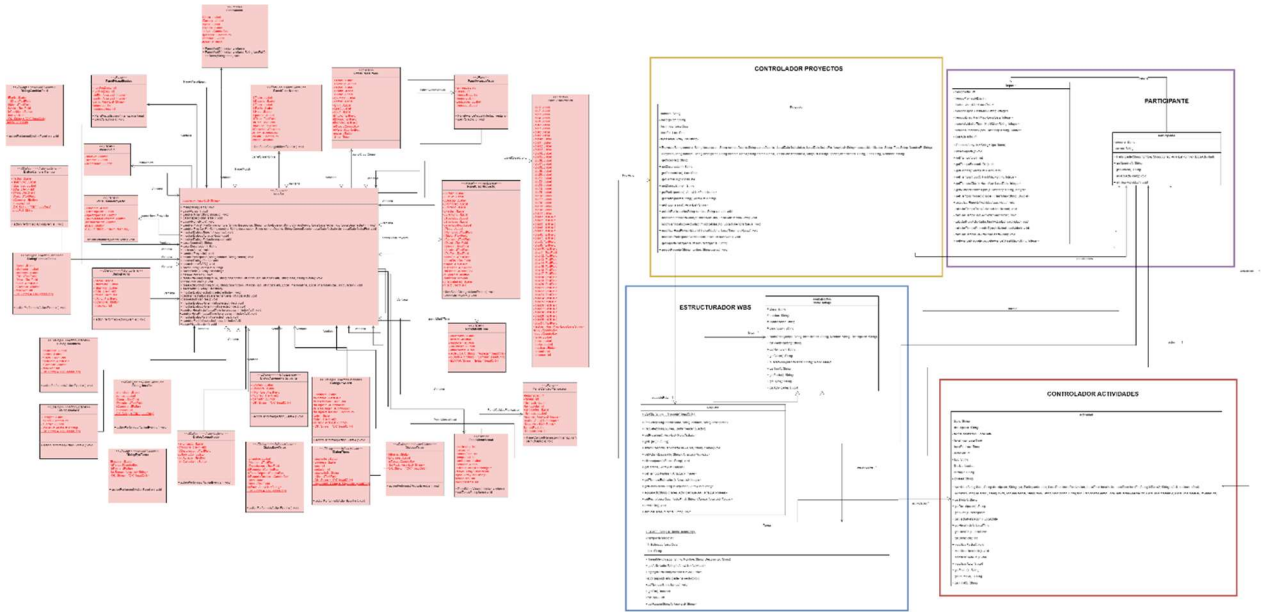


Figura: Para mejor visualización, se recomienda la apertura de la imagen en el siguiente enlace: https://acortar.link/Zt13Ta

DIAGRAMA DE ALTO NIVEL:

Figura: Para mejor visualización, se recomienda la apertura de la imagen en el siguiente enlace: https://acortar.link/e8exFK