Universidad de los Andes

Diseño y Programación O.O.

Proyecto 3: Entrega 1

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

El siguiente documento tiene como objetivo ilustrar y documentar el proceso de identificar, planear y diseñar el manejo de errores dentro de la aplicación. Esto con el fin de mostrar como la aplicación afrontara determinadas posibles situaciones de error las cuales puedan surgir durante la ejecución de esta. Situaciones que pueden surgir por diferentes razones, ya sea una digitación errónea, lectura errónea de algún documento, demoras en las respuestas, respuestas incorrectas, respuestas nulas, entre otros. Para alcanzar este derrotero en primer lugar se identificarán las posibles fuentes de error junto a las causas que podrían generar diversos errores. En segundo lugar, se planeará el manejo de errores evaluando variables como: la probabilidad de ocurrencia, posibilidad de prevenir el problema, el impacto del problema y la posibilidad de corregir el problema. Finalmente, se diseñarán los mecanismos para detectar los errores y las reacciones a los errores.

POSIBLES FUENTES DE ERROR

Usuarios: Usualmente usuarios nuevos con la aplicación podrían digitar ya sea información invalida, información en un formato equivoco (como lo podría ser el formato de una fecha) O en su defecto no digitar información crucial para que la aplicación pueda ejecutarse correctamente. En general pueden tener comportamientos erróneos que podrían afectar el correcto funcionamiento de la aplicación. Usualmente, estos errores provienen más que todo de la interacción usuario-interfaz debido a una posible mala interpretación de lo que se le está solicitando, al seleccionar opciones de las cuales no está seguro de que hacen o clics aleatorios que puedan seleccionar o accionar botones que aún no puedan ser ejecutados sin determinados parámetros o información.

Es así como, los requerimientos funcionales que el usuario podría comprometer en situaciones indeseables serian:

- El usuario puede no digitar toda la información o en su defecto información inequívoca a la hora de crear un proyecto
- El usuario puede cerrar la ventana del cronómetro e intentar finalizar la actividad sin haber detenido el tiempo
- El usuario puede intentar realizar una actividad y digitar una hora o fecha de finalización que sea previa la inicial o viceversa
- El usuario puede querer aportar a una nueva tarea realizando una actividad, sin embargo, no especifica a que tarea hace parte esa actividad
- El usuario puede querer eliminar una tarea que ya posea actividades
- El usuario puede no digitar toda la información o en su defecto información inequívoca a la hora de crear una actividad dentro de una tarea

Datos: Los datos en la aplicación juegan un rol muy importante al ser el método para almacenar información y poder guardar los avances que cada participante realice dentro del proyecto. Sin embargo, es un mecanismo el cual requiere que lo que se quiera almacenar

tenga propiedades específicas para ser serializado y así mismo ser guardado en binario en un archivo *.dat*. Es así como, si algún objeto del proyecto no es serializable o posee dicha propiedad, existiría una pérdida de la información a la hora de querer cargar el proyecto una vez ya creado

Así las cosas, los requerimientos funcionales que los datos podrían comprometer en situaciones indeseables serian:

- Los datos al no guardarse en su totalidad debido a no tener la propiedad de serializable haría que algunos de los cambios realizados por el ultimo participante no queden almacenados en el archivo y por ene no se encuentre completa la información.

PROBABILIDAD DE OCURRENCIA DE ERRORES

Usuarios: Podemos considerar la probabilidad de que el usuario llegue a cometer/generar errores bastante alta. Esto debido a que, como cuenta con la responsabilidad de ingresar la información requerida por el sistema, tiene un gran número de oportunidades de equivocación. Es posible que un usuario olvide llenar un campo relevante, digite un dato incorrectamente, o presione sin intención botones que no pensaba o debía oprimir aún, lo que podría afectar la ejecución del programa.

Ahora bien, se puede tener en cuenta una distinción entre usuarios que sean un poco más experimentados con el uso de la aplicación y usuarios que sean nuevos al sistema. Aquellos que cuenten con experiencia tendrían una menor posibilidad de caer en errores conforme más conozcan la interfaz con la que están trabajando. Por otro lado, podemos asumir que la posibilidad de ocurrencia de usuarios que estén haciendo uso de la herramienta por primera vez se encuentra en su punto máximo, puesto que no conocen la estructura de la aplicación lo que conlleva a que realicen acciones que contradigan el normal funcionamiento de esta.

Datos: La principal fuente de error, en cuanto a los datos, surgiría a la hora de serializar. En este caso consideramos la posibilidad de que alguna de las clases no se encuentre bien programada para ser serializada, lo que llevaría a perdidas de datos entre utilización y utilización de la plataforma. No obstante, teniendo en cuenta que poseemos el conocimiento de qué clases deben poder serializarse, tomamos esta probabilidad de incidencia como baja.

IMPACTO DE ERRORES

Usuarios: Los errores provenientes de los usuarios en su gran mayoría tienen un alto impacto puesto que ellos son los que suministran la información necesaria para el correcto uso y aprovechamiento de la aplicación. Lo anterior nos llevaría a que, al ellos no digitar un dato que es importante para la funcionalidad de la aplicación, se vean comprometidos muchos requerimientos funcionales.

Datos: Los datos, al ser el medio mediante el cual se podrá seguir trabajando en un proyecto a través del tiempo, tienen también un alto nivel de impacto. Si estos datos están incompletos o, en su defecto, no se realiza correctamente la actualización de los datos, el progreso dentro de la aplicación se vería comprometido puesto que tendría que hacerse todo el proyecto en una sola iteración para que la herramienta tenga un correcto funcionamiento.

POSIBILIDAD DE PREVENCIÓN/CORRECCIÓN DE ERRORES

Usuario: A pesar de ser posiblemente la fuente más grande de errores, el usuario reacciona a cómo la aplicación esta presentada y va adquiriendo experiencia con el tiempo. Así las cosas, hay una alta posibilidad de corregir el problema si la interfaz es más intuitiva con el usuario en función a los requerimientos que esta necesita cumplir. Adicionalmente, en caso de que un diseño más intuitivo no sea suficiente, la aplicación pueda ser capaz de reaccionar a las acciones del usuario e informarle que está realizando una acción que podría comprometer el correcto funcionamiento de la aplicación, ya sea digitar información erróneamente, no digitar completamente la información o presionar algún botón por error en un momento que no debía.

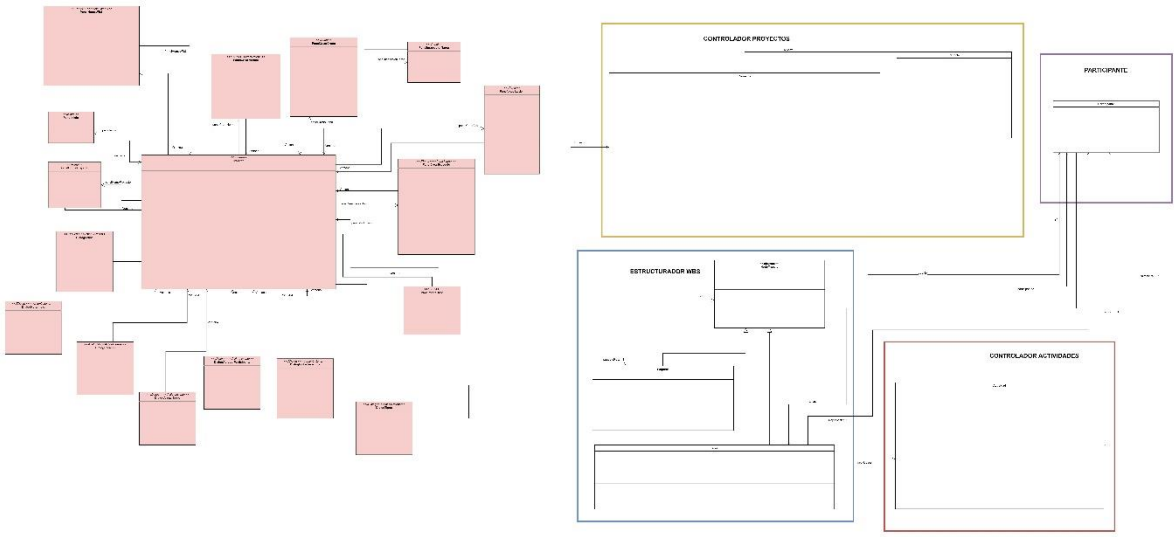
MANEJO DE ERRORES

Teniendo en cuenta las fuentes de error mencionadas anteriormente, planeamos hacer uso de las siguientes estrategias para manejarlos:

- Para cando el usuario ingrese mal la información al crear el proyecto, contaremos con un manejo de error en el método constructor de la clase *Proyecto*. En este caso, retornaremos al usuario un mensaje de excepción informando que han ingresado mal los datos para la ceración del proyecto y le permitiremos intentarlo de nuevo.
- Para cuando el usuario cierre la ventana del cronómetro e intentar finalizar la actividad sin haber detenido el tiempo, contaremos con un manejo de error en la clase *PanelCrearCrono.* Aquí, le preguntaremos al usuario por segunda vez si en verdad desea cerrar la herramienta y le informamos que perderá los datos si decide hacerlo. Esta pregunta la llevaremos a cabo mediante un OptionPane al recibir el mensaje de excepción.
- Para cuando el usuario digite una hora o fecha de finalización de una tarea que sea cronológicamente anterior a la inicial o viceversa, contaremos con un manejo de error en método constructor de la clase *Actividad.* Se evaluará la congruencia cronológica y, en caso de que se presente alguna anomalía, se le informará al usuario que las fechas son incongruentes y no se pudieron guardar los datos.
- Para cuando el usuario quiera aportar una actividad y no especifique a qué tarea corresponde esa actividad, contaremos con un manejo de error en el método *getSeleccioando()* de la clase *PanelHomeWBS*. En este caso, le informaremos al usuario que no ha seleccionado ninguna tarea y debido a esto no es posible agregar la actividad.
- Para cuando el usuario quiere eliminar una tarea que ya posee actividades, contaremos con un manejo de error en el método *eliminarNodo()* en la clase *Paquete*. Básicamente, se le dirá el usuario que no puede llevar a cabo esta acción mediante un mensaje que también explique que esto no se puede realizar si una tarea ya posee actividades.
- Para cuando el usuario no digite toda la información o en su defecto información equívoca a la hora de crear una actividad dentro de una tarea, contaremos con un manejo de error en el método *crearActividad()* de la clase *Interfaz*. En esta ocasión, se le informará al usuario que no fue posible crear la información debido a datos faltantes. Esta situación también puede presentarse en distintos requerimiento –al crear un proyecto, al añadir un participante y al insertar un tipo de tarea– por lo que contaríamos con el mismo tipo de excepción en

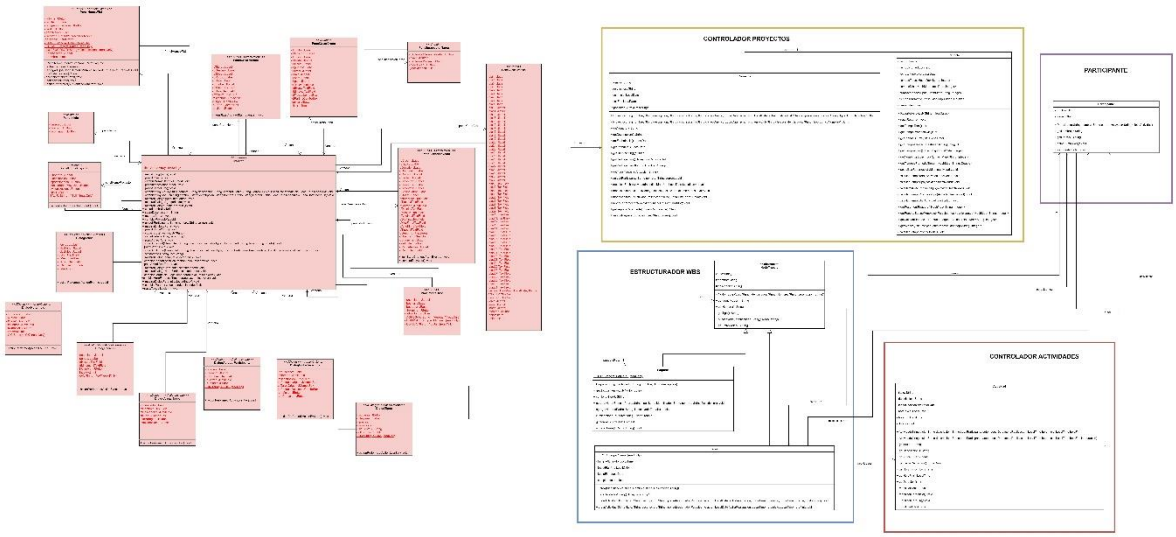
Es importante resaltar que, debido a la naturaleza del problema, la mayoría de estos errores se manejarán por medio de excepciones evaluadas en un try catch. De igual manera, se trabajará una interfaz más intuitiva la cuál minimice las probabilidades de error.

DIAGRAMA DEL MODELO DE NEGOCIO – ALTO NIVEL



Para una mejor visualización de la imagen recomendamos seguir el siguiente link https://acortar.link/Hrq5rb

DIAGRAMA DEL MODELO DE NEGOCIO- BAJO NIVEL



Para una mejor visualización de la imagen recomendamos seguir el siguiente link https://acortar.link/8y9xVh