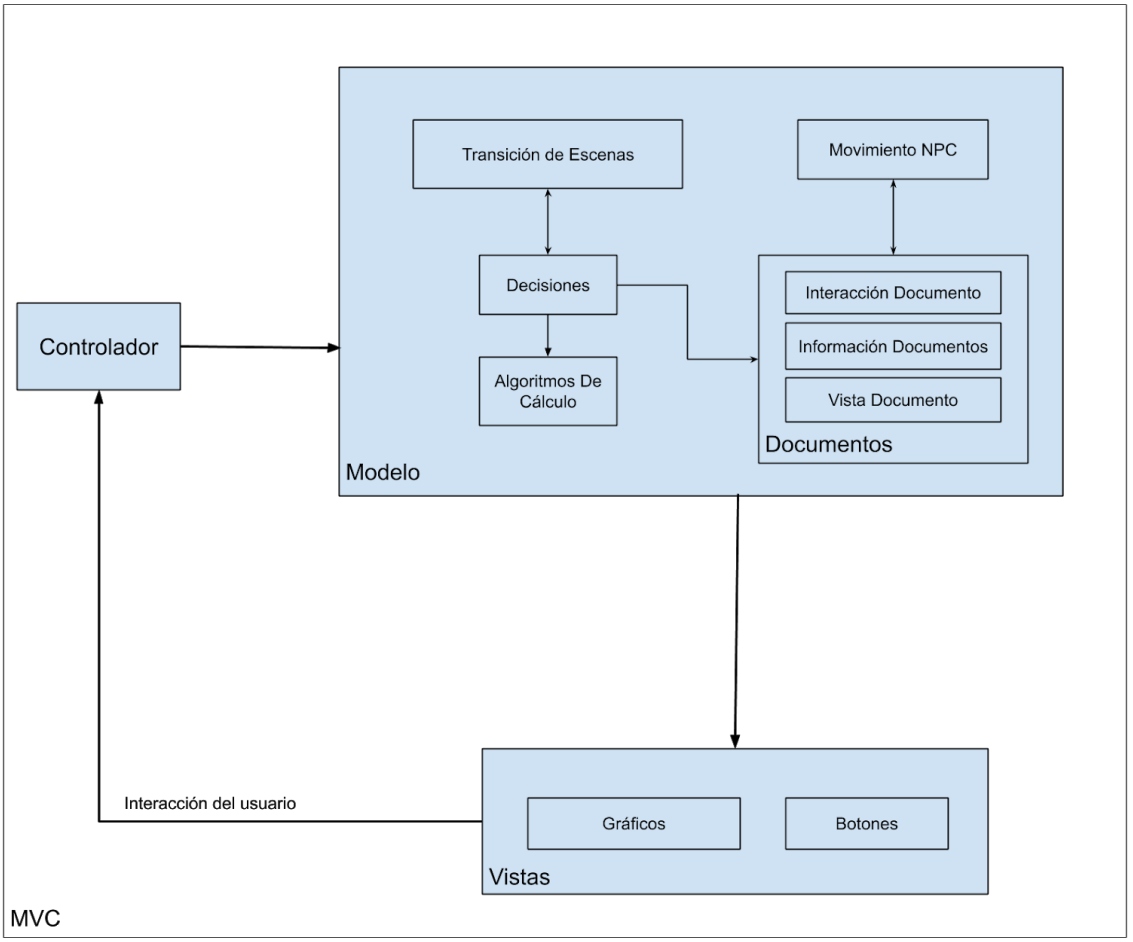
QuarenTEAM

Nombre del grupo: ***QuarenTEAM***
Nombre del proyecto: ***Collateral***
Nombre del documento: ***Documento de arquitectura***
Integrantes:
Don Enzo Martin
Quinteros Tomás
Coschica Francisco Nicolás
Hidalgo Juan Nahuel
Fecha: ***08/06/2020***

Índice:

Arquitectura del Sistema

Diagrama de Arquitectura General:



Explicación MVC

El MVC separa los datos en una aplicación de su representación y el módulo encargado de gestionar los eventos y las comunicaciones. Propone la construcción de tres componentes distintos que son el **modelo**, la **vista** y el **controlador**, es decir, por un lado define componentes para la representación de la información, y por otro lado para la interacción del usuario.

En nuestro caso el componente que hace de modelo es el AdministradorDecisiones el cual se encarga de toda la logica detras del sistema que haciendo uso de la interfaz Algoritmos implementa diferentes algoritmos para manipular sus datos, según el sistema o el usuario lo requiera.

Por otro lado el componente de Controlador es implementado por la clase Controller, la cual utiliza para comunicarse con los demás actores mediante las interfaces Comunicación y Vistas , las cuales son provistas por estos otros actores.

Finalmente el componente de vista se implementa mediante las clases Texto, BarraDesempeño y GraficoTorta. Las cuales implementan la interfaz Vistas que le permiten la comunicación con el Controlador. Se comunican con el modelo por medio de la interfaz Sujeto que implementa el AdministadorDecisiones.

¿Por qué?

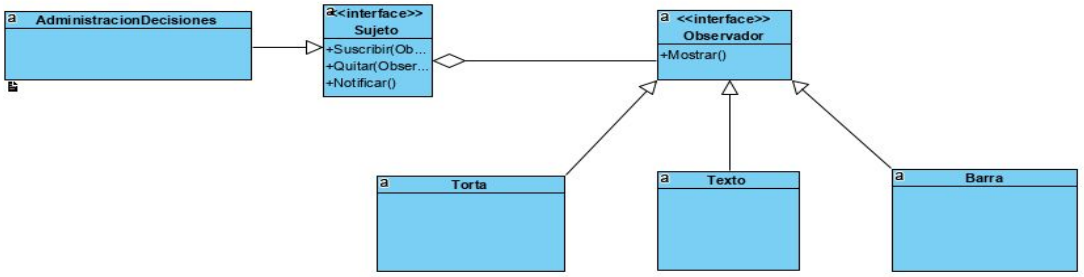
Se utilizó el patrón de MVC que satisface el requerimiento no funcional de Mantenibilidad. El patrón MVC nos permite tener encapsulado los comportamientos y las vistas, desacomplandolas completamente; lo cual permite a nuestro equipo agregar nuevas vistas y comportamientos disminuyendo la cantidad de conflictos en el proceso.

Se debe mostrar también la aplicación de patrones de diseño como el Singleton, Observer, Strategy, etcétera usando diagramas de clases e indicar por qué se utilizan y qué problemas se solucionan con ellos.

Los patrones de diseño que utilizamos en el proyecto fueron el Observer y el strategy.

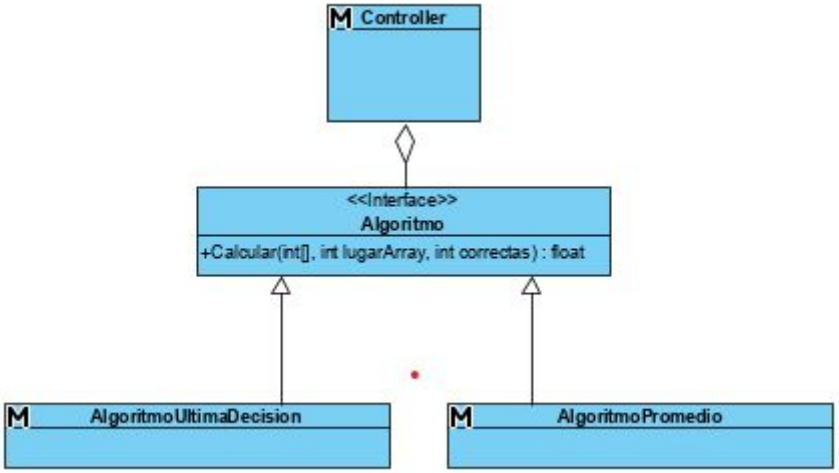
Uso del patrón Observer:

El patrón en nuestro caso se implementa haciendo uso de las clases AdministradorDecisiones, que implementa la interfaz Sujeto, y las clases Texto, Barra y Torta que implementan la interfaz observer. De esta manera las diferentes vistas se suscriben al sujeto y este la notifica de ser necesario. Logrando de esta manera un desacople entre los datos que se quieren mostrar y el medio por el cual se los muestran. Esto nos permite suscribir clases a AdministradorDecisiones sin tener que hacer ninguna modificación en este.



Uso del patrón Strategy:

Se implementa mediante la clase Controller, que cumple el rol de cliente, y las clases AlgoritmoUltimaDecision y AlgoritmoPromedio que son los diferentes algoritmos. De esta manera la clase Controller puede intercambiar dinámicamente según sus necesidades entre estos dos algoritmos. Esto nos permite procesar los datos generados por el usuario mientras juega de diferentes maneras, en tiempo de ejecución.



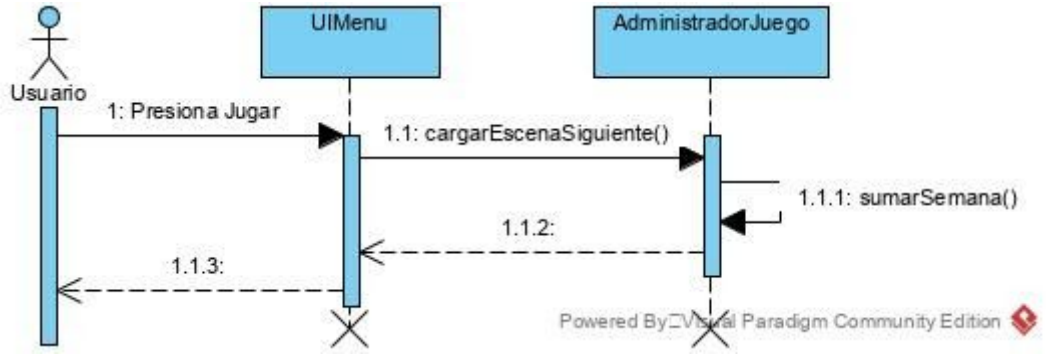
Diseño del Sistema:

A)Diagrama de Clases:

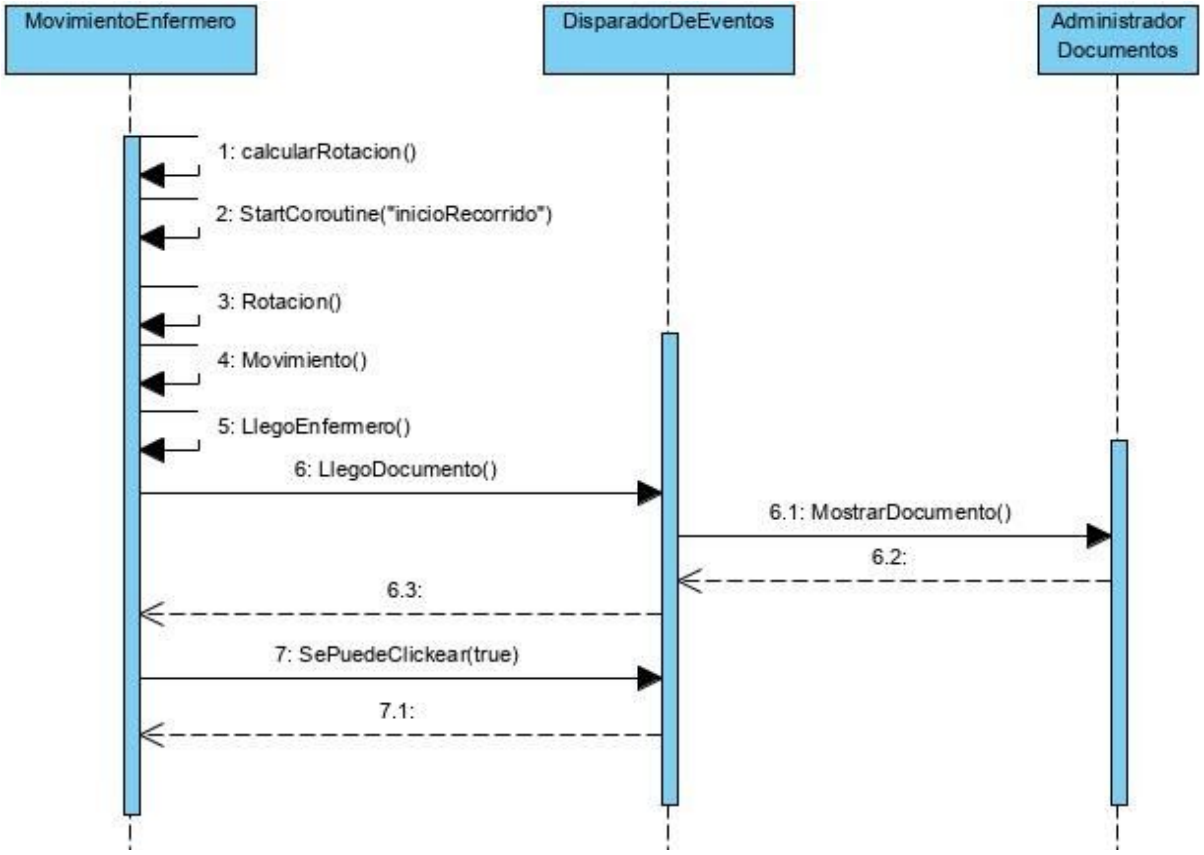
Link al diagrama de clases.

B)Diagramas de Secuencia:

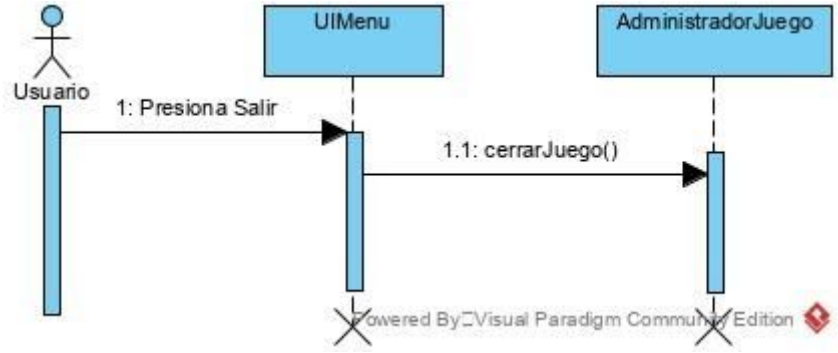
a) Inicio del juego (desde el menú)



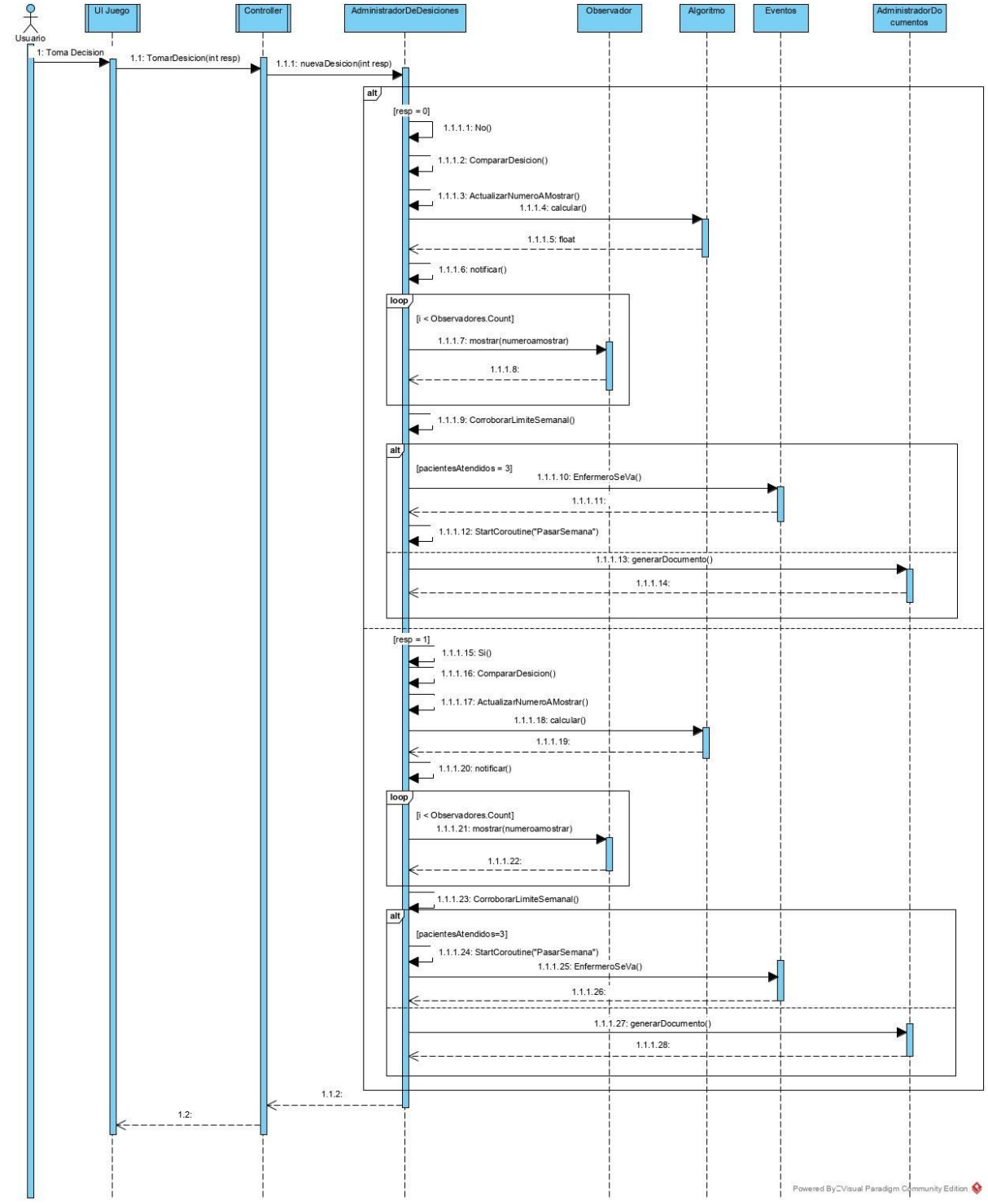
b) Inicio de la semana (explica el movimiento del enfermero)



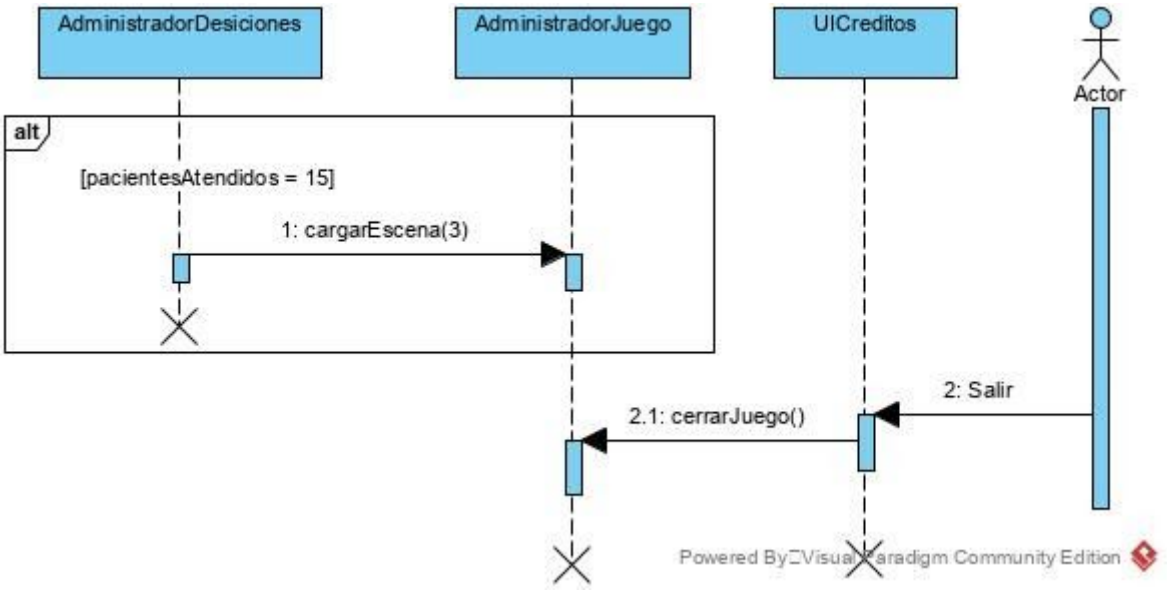
c) Salir del juego



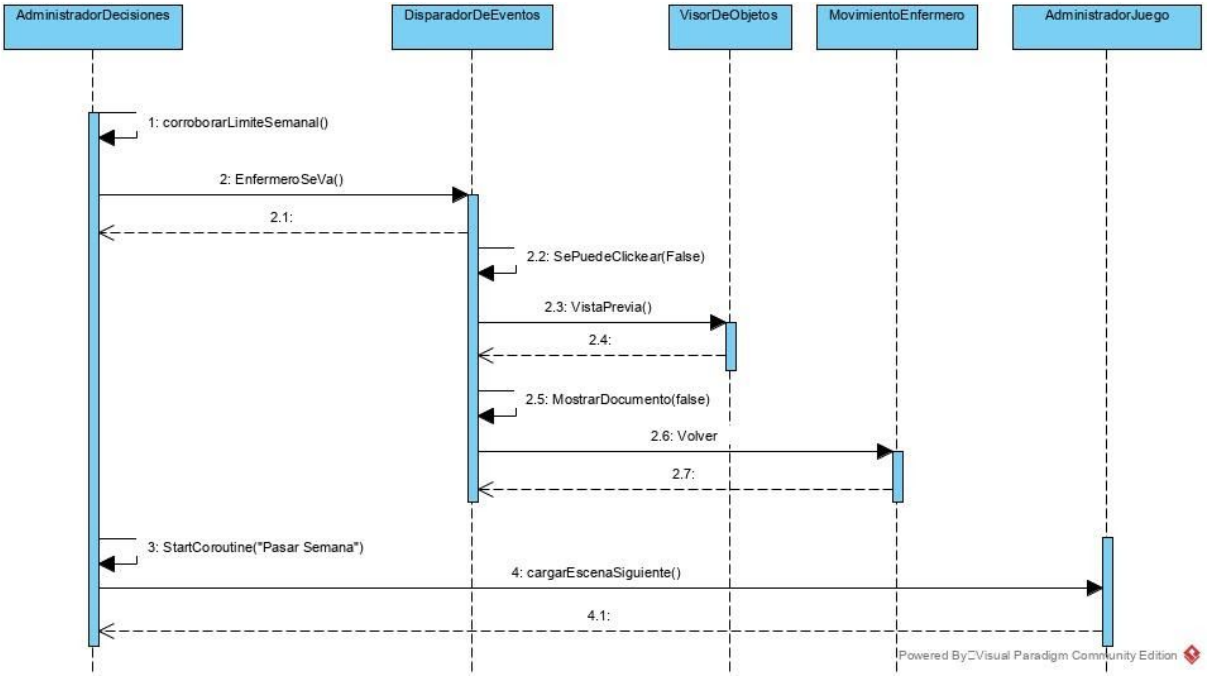
d) Decisión de “Si y No”



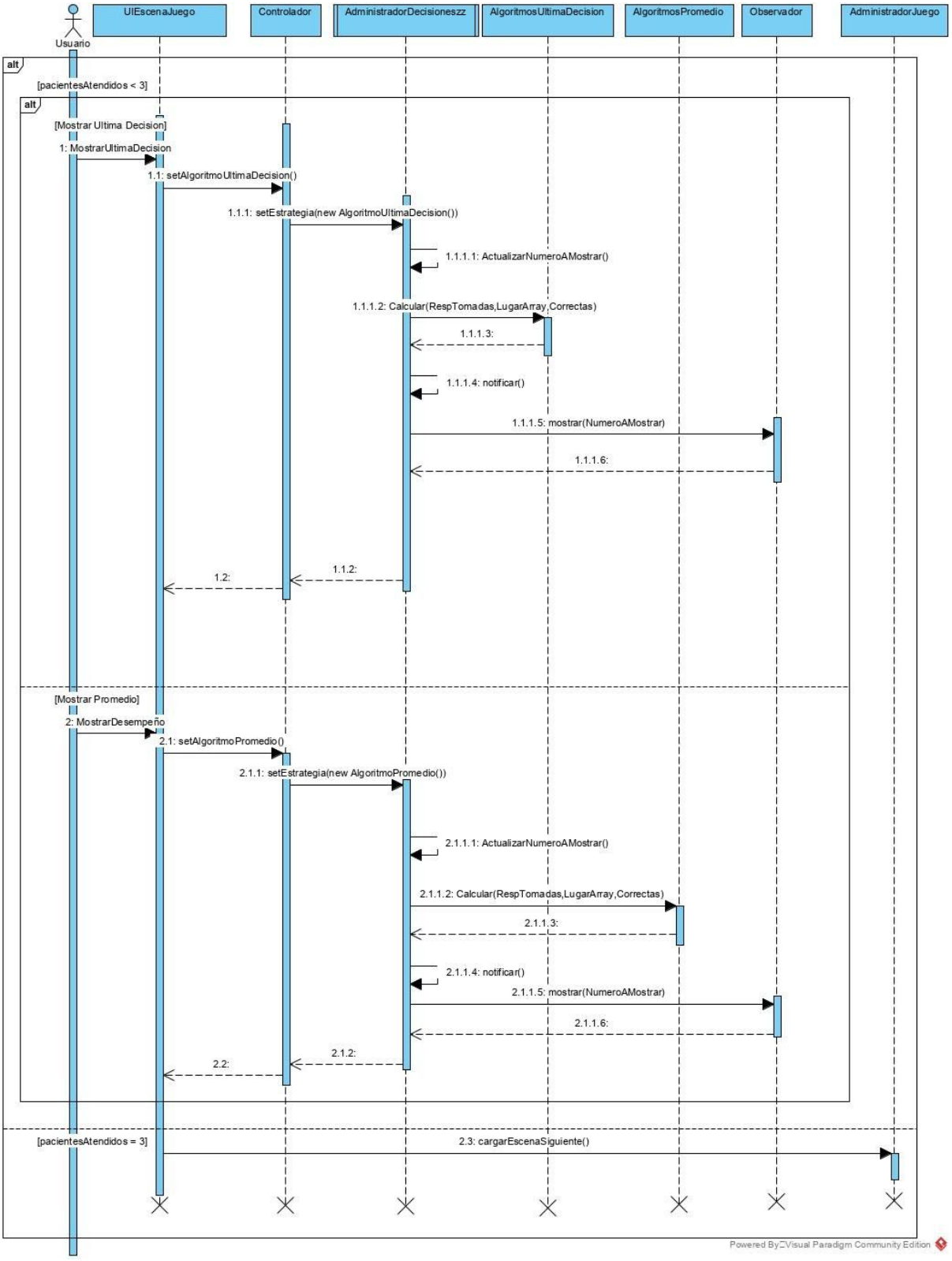
e) Créditos



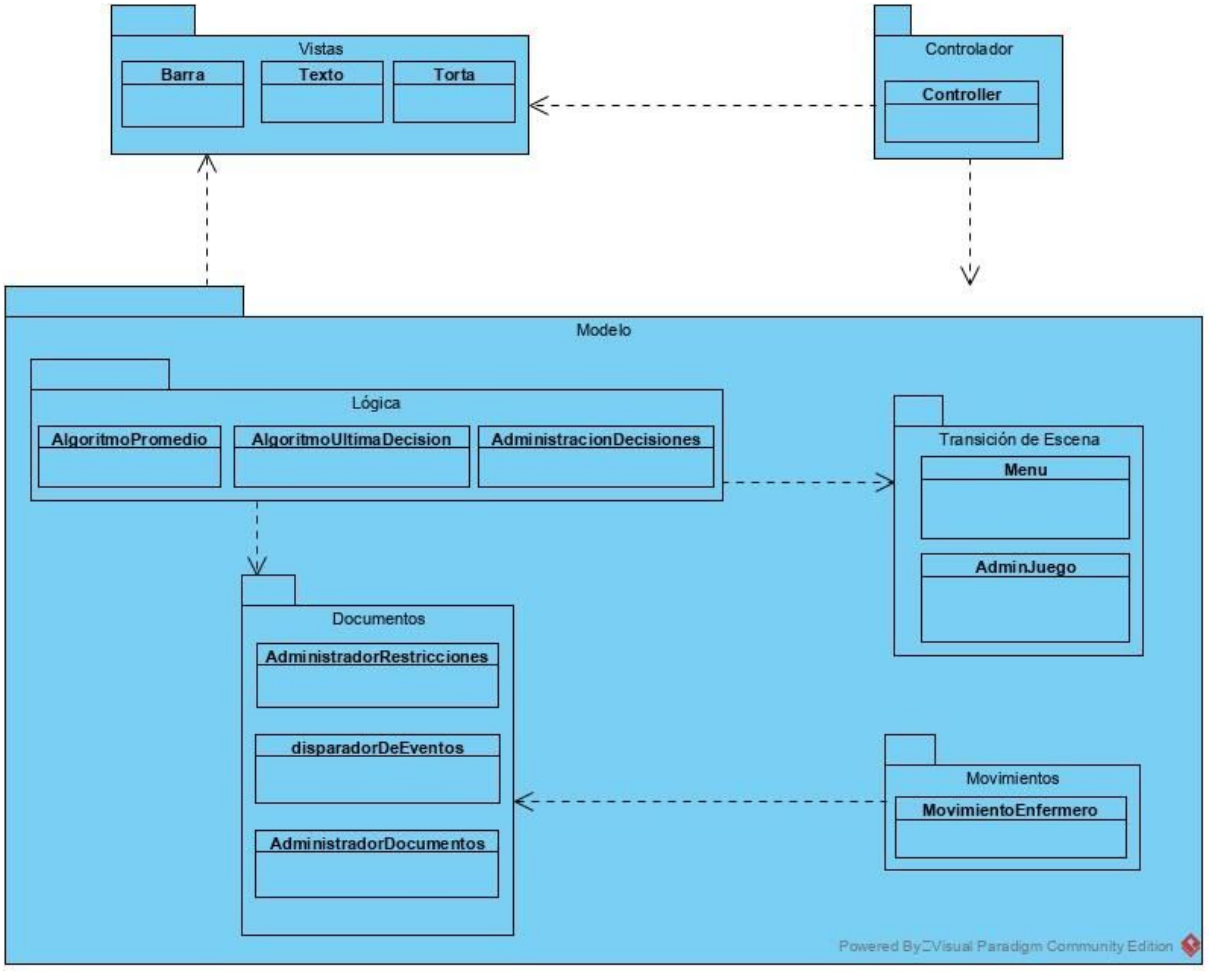
f) Finalizado de escena juego (se irá el enfermero)



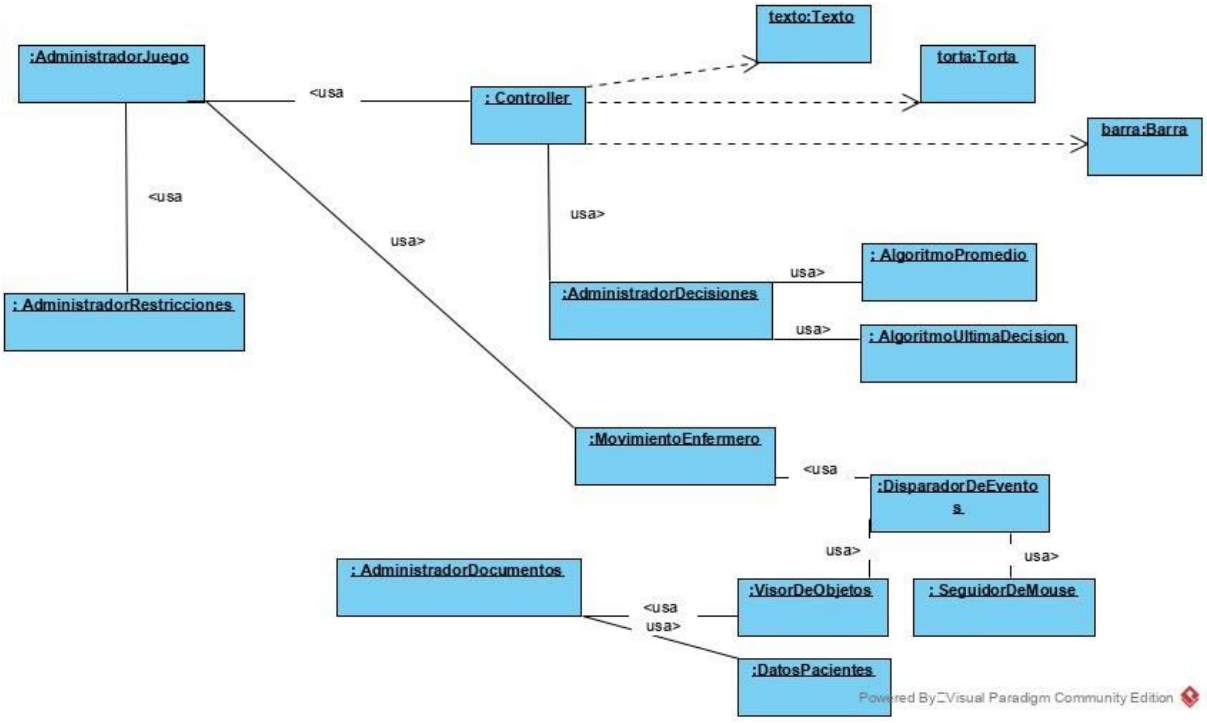
g) Actualizado de la barra y los gráficos



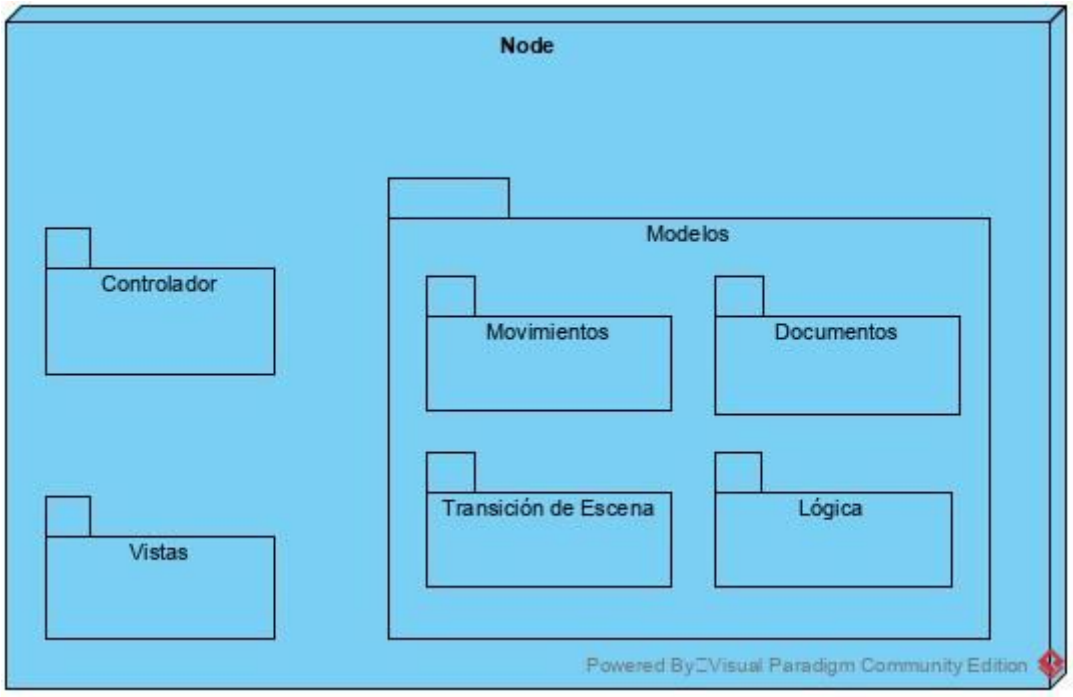
C)Diagrama de Paquetes:



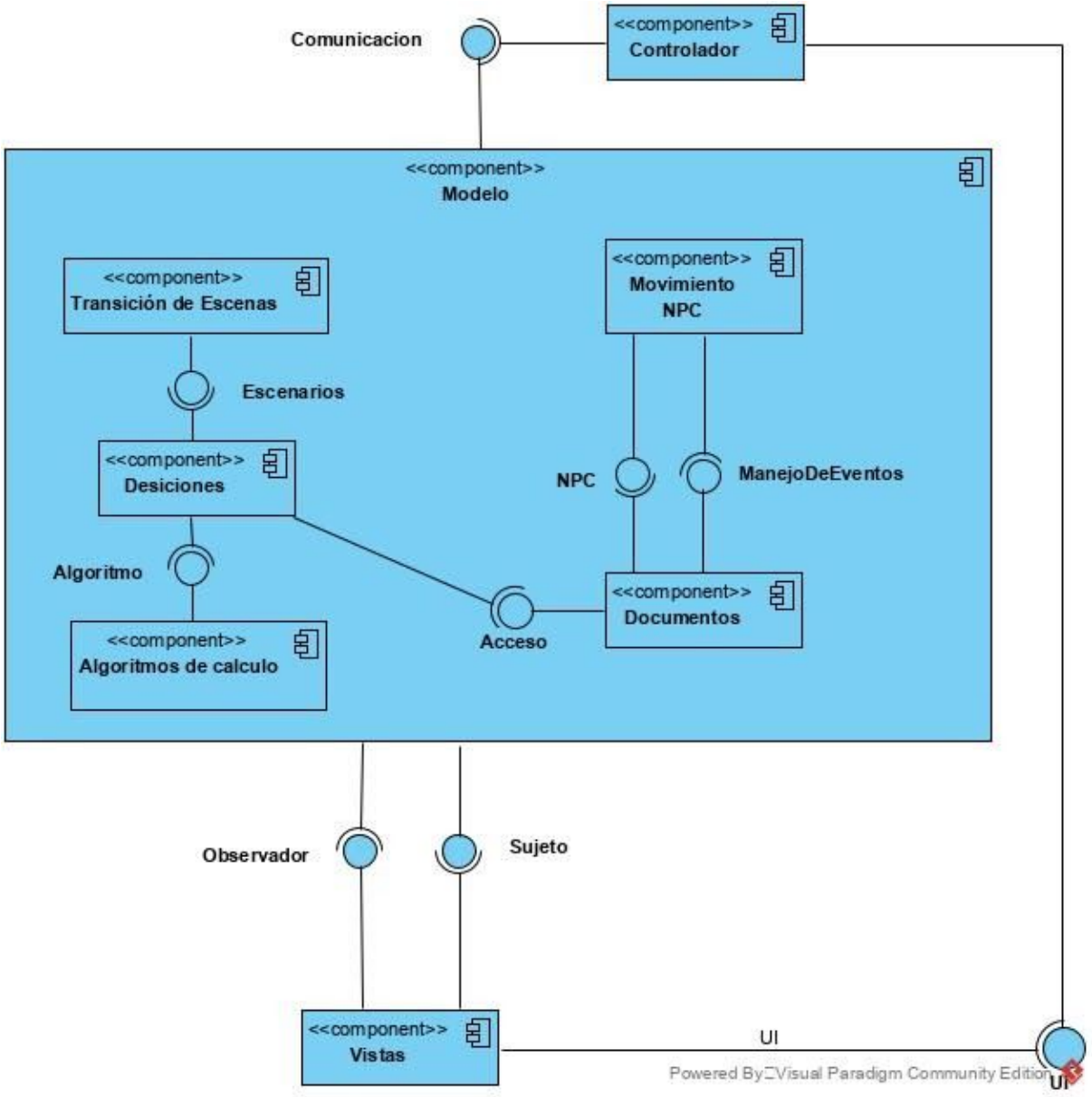
D)Diagrama de Objetos:



E) Diagrama de despliegue



F) Diagrama de componentes



Tests

Dirección al histórico de tests y matrices de trazabilidad (especificando la clasificación de cada test) seleccionados por release:

https://docs.google.com/spreadsheets/d/1aWulhsg7lx4iZAk1HOlFlTii2NRA5kEPKDsnH8B-oNg/edit?usp=sharing

Se adjunta también la carpeta donde se encuentran los test de las métricas de código:
https://github.com/juannahuelhidalgo/QuarenTEAM/tree/master/Documentacion%20Del%20Proyecto/Metricas

Para solicitar correr los test o suscribirse para ser notificado cada vez que se corran todos los test sobre COLLATERAL se deberá mandar un mail especificando nombre completo a:

quarenteam.collateral@gmail.com

Recibirá resultados y confirmación en la brevedad.