**3.2 Suposiciones, dependencias y restricciones**

Que las herramientas proporcionadas por los profesores funcionan, como así hemos comprobado ejecutando el código del Buscaminas el cuál tiene una funcionalidad mínima que iremos ampliando, o el Microsoft Office que nos ayudará a realizar algunos diagramas y organizar tareas entre otras cosas.

También que el ordenador que se nos proporciona para la presentación funcione correctamente, así como los ordenadores de clase para poder trabajar desde ellos en las horas de clase que se nos dejan para realizar la práctica.

Además de los ordenadores del aula también hemos necesitado acceder a myApps para utilizar algunas herramientas que se nos proporcionan en este entorno como puede ser Microsoft Project.

También que estas herramientas son las adecuadas, como los estándares a seguir para realizar este proyecto.

En cuanto a los acontecimientos externos de los cuáles el proyecto depende podríamos considerar:

-La estabilidad de GitHub, herramienta indispensable la cuál usamos para poner las cosas en común.

-Ya que es un proyecto académico, que las herramientas sean gratuitas y no necesitemos presupuesto para conseguirlas

-Que el personal esté en condiciones para realizar su parte

Restricciones bajo las cuales el proyecto va a ser guiado:

-No menos de 30 páginas de extensión

-El grupo debe estar formado por 3-5 personas

-Fecha límite para entregar el proyecto es el 25 de abril a las 23:55h

-Que el diagrama de Gant tenga entre 15 y 20 actividades

-Que el diagrama de Pert tenga un mínimo de 20 actividades

-Relacionado con el código que se nos pide mejorar podemos considerar las siguientes restricciones:

• Añadir una opción para poder reiniciar el juego en cualquier momento y sin tener que reiniciar el programa.

• Mostrar en todo momento la cantidad de minas que quedan por descubrir, así como el tiempo que llevamos jugando en la partida actual.

• Crear diferentes niveles de dificultad:

- Principiante (tablero 10x10 y 10 minas)

-Intermedio (tablero 16x16 y 40 minas)

- Experto (tablero 32x16 y 99 minas)

-Personalizado (permite definir el tamaño del tablero y la cantidad de minas)

• Guardar los 10 mejores tiempos en cada dificultad (sin contar Personalizado) • En el mensaje al ganar, mostrar junto con el tiempo un campo para añadir un nombre de jugador y dar la opción de guardar el tiempo o no.

• Permitir guardar la partida actual (estado del tablero y tiempo) en un fichero, así como poder recuperar una partida guardada a partir de un fichero.

• Crear un menú que nos permita acceder a las diferentes opciones del juego.

**3.3 Gestión de riesgos**

Podemos identificar varios riesgos:

-Al modificar código, como cada vez que programamos, corremos el riesgo de perder funcionalidad.

-Que al subir cada uno nuestra parte a GitHub haya conflictos.

-Que al ir avanzando en el proyecto nos demos cuenta de nuevas actividades y haya que introducir en los diferentes diagramas con los cambios que ello conlleva.

-Al ser un proyecto académico no corremos riegos con el dinero, pero si con el tiempo empleado en cada actividad por eso hay que planificarlo bien.

-Que en la planificación y distribución de tareas hayamos dedicado poco tiempo a una actividad determinada y no nos de tiempo a realizarla y haya que dedicarle más tiempo. Al revés también, si dedicamos demasiado tiempo a una actividad que no necesita tanto tiempo, corremos el riesgo de no realizar actividades posteriores por falta de tiempo.

Para intentar evitar estos riesgos hay que tener una planificación adecuada y que la comunicación del grupo sea fluida para atajar los problemas a tiempo. Así como el innegociable esfuerzo de cada uno de los miembros para entregar todo a tiempo según lo hemos planificado.

A la hora de distribuir el trabajo es importante que nos ayudemos unos a otros en el caso de que uno acabe su tarea, ya que así avanzaremos más rápidamente y no corremos el riesgo de ir justos con los plazos de entrega.

**3.4 Mecanismos de supervisión y control**

-Las herramientas principales que hemos utilizado han sido Word y Microsoft Project. Con el primero creamos los informes y los documentos finales y con el Microsoft Project generamos documentos con la gestión de tareas y diagramas. Cada uno lo genera en su propio formato que nosotros sabemos gestionar sin problemas.

-En GitHub los archivos que subimos van a ser tanto .java como documentos Word y de Microsoft Project.

-Una vez hemos terminado la parte de código, el encargado de esta parte mediante la herramienta Eclipse, que es un entorno de desarrollo para programar en diferentes lenguajes, genera el ejecutable entregable cuya extensión es .jar.

**3.5 Plan de personal**

Para llevar a cabo este proyecto los profesores de la asignatura nos recomendaron grupos de entre 3 y 5 personas, y nos ayudaron a completar el equipo de trabajo el cuál está formado finalmente por cuatro personas

En cuanto al nivel de habilidad, se requiere un mínimo de conocimiento de Java por parte de los cuatro componentes del grupo para entender lo que se pide e interpretar código. También saber manejar el Microsoft Project para realizar los diagramas y gestionar las tareas. Así cómo saber como funciona GitHub para que cada miembro del grupo suba su trabajo y éste sea visible para el resto del grupo. Para trabajar con el código proporcionado por los profesores de la asignatura, tras consensuarlo entre los componentes del grupo, hemos decidido utilizar Eclipse que tiene una interfaz bastante amigable para los alumnos y así trabajar de forma eficiente con el código. Se requiere un mínimo de habilidad con esta herramienta ya que debemos generar finalmente un ejecutable con el código definitivo.

¿Tiempos de comienzo? ¿Duración de las necesidades?

**4. Proceso técnico**

**4.1 Metodología, herramientas y técnicas**

Los únicos sistemas de computación que hemos utilizado han sido nuestros ordenadores personales, y en ocasiones los ordenadores de la universidad.

Para el desarrollo del proyecto hemos establecido varias responsabilidades de acuerdo con la carga de trabajo que suponía cada tarea. Hemos establecido dos departamentos: el departamento encargado de la realización del Plan de Proyecto, y el departamento encargado de la Gestión de la Configuración del Software. Este último departamento lo hemos dividido a su vez en dos subdepartamentos, el departamento de desarrollo y el departamento de documentación, cuya coordinación entre ambos ha sido fundamental para la elaboración de este documento.

El lenguaje de programación utilizado es JAVA.

Como ya hemos comentado vamos a utilizar la herramienta GitHub para subir el trabajo individual y ponerlo en común. También el Microsoft Office para definir tareas y hacer los diagramas.

También hemos complementado los conocimientos adquiridos con el paso de la asignatura y sus respectivas diapositivas con videotutoriales de Youtube cuyos links se indican en apartados anteriores y en los que nos hemos apoyado para entender mejor determinadas cosas que teníamos que realizar.

Otra herramienta fundamental que hemos utilizado ha sido la aplicación de mensajería móvil *WhatsApp* para una comunicación fluida y más rápida.

Además, hemos utilizado Skype para hacer una especie de reunión y discutir algunos temas que teníamos pendientes, esta herramienta también ha sido importante ya que muchas veces debido a la carga de trabajo que teníamos nos era imposible quedar en persona por el tiempo que teníamos que dedicarle.

**4.2 Documentación software**

Esta subsección del PGPS debería contener o referenciar el plan de documentación del proyecto software. El plan del documento debería especificar los requisitos de documentación y los hitos, documentos base, revisiones y despedidas de la documentación del software. El plan de documentación también puede contener una guía de estilo, convenciones en la nomenclatura y formatos del documento. El plan de documentación podría incluir un resumen de la agenda y los recursos necesarios para el esfuerzo de la documentación. El Estándar ANSI/IEEE 829-1983 “Standard for Software Test Documentation” ofrece el estándar para la documentación de pruebas del software.

**4.3 Funciones de apoyo al proyecto**

Esta subsección del PGPS debería contener directamente o por referencia los planes para las funciones de soporte para el proyecto software. Estas funciones pueden incluir, pero no están limitadas a, gestión de la configuración, aseguramiento de la calidad del software, y verificación y validación. Los planes para las funciones de soporte al proyecto deberán ser desarrollados con un nivel de detalles consistente con las otras secciones del PGPS. En particular, se deben especificar las responsabilidades, requerimientos de recursos, agendas y herramientas para cada una de las funciones de soporte al proyecto. La naturaleza y tipo de las funciones de soporte al proyecto variarán de unos a otros proyectos.; de todos modos, la ausencia del aseguramiento de calidad software, gestión de la configuración o plan de verificación y validación debe ser explícitamente justificado en los planes de los proyectos que no los incluyan.

**5. Plan de desarrollo**

**5.1 Paquetes de trabajo**

Esta subsección del PGPS debería contener directamente o por referencia los planes para las funciones de soporte para el proyecto software. Estas funciones pueden incluir, pero no están limitadas a, gestión de la configuración, aseguramiento de la calidad del software, y verificación y validación. Los planes para las funciones de soporte al proyecto deberán ser desarrollados con un nivel de detalles consistente con las otras secciones del PGPS. En particular, se deben especificar las responsabilidades, requerimientos de recursos, agendas y herramientas para cada una de las funciones de soporte al proyecto. La naturaleza y tipo de las funciones de soporte al proyecto variarán de unos a otros proyectos.; de todos modos, la ausencia del aseguramiento de calidad software, gestión de la configuración o plan de verificación y validación debe ser explícitamente justificado en los planes de los proyectos que no los incluyan.

**5.2 Dependencias**

Esta subsección del PGPS debería especificar las relaciones de orden entre los distintos paquetes de trabajo para reflejar de alguna forma las interdependencias entre ellos y la dependencia de acontecimientos externos al proyecto. Se pueden usar técnicas tales como las listas de dependencia, redes de actividades, y método de camino crítico para describir las dependencias entre los distintos paquetes de trabajo

**5.3 Recursos**

Somos cuatro estudiantes de la Ingeniería Informática los que realizamos este proyecto y hemos utilizado distintos tipos de recursos para completarlo.

Hemos trabajado principalmente con ordenadores donde hemos desarrollado el código nuevo que se nos pedía mediante Eclipse en este caso.

A su vez, hemos usado el Word como herramienta principal para generar los informes pertinentes a cada parte.

El Microsoft Project, recomendado por el profesor, nos ha sido de gran ayuda para generar los diagramas de Pert y de Gant, los cuáles sirven posteriormente para completar el PGPS. También para hacernos un diagrama relacionando las tareas y el tiempo invertido en cada una.

GitHub nos ha permitido subir cada uno nuestro trabajo a la plataforma y así hacerlo común.

También mediante el Eclipse hemos generado el .jar que es el ejecutable que entregaremos a los profesores para que sea evaluado.

¿Tiempo de computación?

¿Requerimientos de mantenimiento?

**5.4 Presupuesto y distribución de recursos**

Todas las herramientas y recursos que hemos necesitado han sido gratuitos y en esta práctica no era necesario ningún presupuesto. A su vez, todas las herramientas las hemos obtenido o bien de internet, como es el caso de GitHub, Microsoft Project y Eclipse, o porque ya los teníamos previamente instalados en nuestros ordenadores como el Word.

**5.5 Calendario**

Esta subsección del PGPS debería ofrecer un calendario para que las distintas funciones, actividades y tareas del proyecto fuesen desarrollados en un determinado orden y se consiguiesen los plazos previstos. Las agendas pueden ser expresadas bien en forma de calendario relativo, bien en incrementos con respecto a fechas de hitos claves del proyecto.