**Proyecto: iDoctor**

**Plan del proyecto del software**

**(Pressman)**

**Miembros del equipo:**

|  |  |
| --- | --- |
| Samuel Solo de Zaldívar Barbero |  |
| Jesús Martín |  |
| Adrián Agudo García-Heras |  |
| Javier Pino Hernández |  |
| Agustín Jofré Millet |  |
| María Dolores Quilarte |  |
| Huaibo Yang |  |

**Control de cambios**

|  |  |  |  |
| --- | --- | --- | --- |
| **Número de versión** | **Fecha** | **Autores** | **Descripción** |
| Versión 1 | 1/12/16 | iDoctor team | Inicio del documento. |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

# Índice

1. Introducción

1.1 Propósito del plan

1.2 Ámbito del proyecto y objetivos

1.2.1 Declaración del ámbito

1.2.2 Funciones principales

1.2.3 Aspectos de rendimiento

1.2.4 Restricciones y técnicas de gestión

1.3 Modelo de proceso

2. Estimaciones del proyecto

2.1 Datos históricos

2.2 Técnicas de estimación

2.3 Estimaciones de esfuerzo, coste y duración

3. Estrategia de gestión del riesgo

3.1 Introducción: Estudio de los riesgos

3.2 Priorización de riesgos del proyecto

3.3 Plan de gestión del riesgo Reducción, supervisión y gestión del riesgo

3.4 Planificación temporal del Control de Riesgos

3.5 Resumen

4. Planificación temporal

4.1 Estructura de descomposición del trabajo/Planificación temporal

4.2 Gráfico Gantt

4.3 Red de tareas

4.4 Tabla de uso de recursos

5. Recursos del proyecto

5.1 Personal

5.2 Hardware y software

5.2.1 Hardware

5.2.2 Software

5.3 Lista de recursos

6. Organización del personal (Gestión del Equipo)

6.1 Estructura de equipo (si procede)

6.2 Informes de gestión

7. Mecanismos de seguimiento y control

7.1 Garantía de calidad y control (Plan de Calidad)

7.2 Gestión y control de cambios (Plan GCS)

7.2.1 Introducción: Propósito, Alcance, Definiciones, Referencias

7.2.2 Tipos de artefactos a gestionar (los ECSs)

7.2.3 Criterios y protocolos para Nombrar los ECSs

7.2.4 Responsable de los procedimientos de GCS y de la creación de Líneas Base.

7.2.5 Políticas para el Control de Cambios y la Gestión de Versiones

7.2.6 Registros para mantener el rastro de los cambios

8. Apéndices

1. **Introducción**
   1. **Propósito del plan**

El objetivo general del proyecto iDoctor consiste en realizar una aplicación basada en consultas médicas.  Más concretamente, el proyecto va destinado a cualquier persona (potencial paciente o profesional sanitario). En primer lugar, cualquier persona podrá consultar información sobre primeros auxilios, su historial clínico y realizar cualquier pregunta a algún médico. En segundo lugar, un sanitario podrá responder a las preguntas realizadas por los demás usuarios del sistema, pero además podrá editar y agregar patologías que crea conveniente. En ambos casos, tanto el médico como el paciente se tendrán que dar de alta para realizar estos procesos, aunque con diferentes tipos de cuentas, ya que para entrar como personal sanitario se le pedirá su número de colegiado a la hora de registrarse.

* 1. **Ámbito del proyecto y objetivos**
     1. **Declaración del ámbito**

El ámbito de este software serían los pacientes y los médicos que utilicen la aplicación.

La aplicación abarca un sistema de ayuda para los pacientes por parte de una serie de profesionales de la medicina, los cuales ofrecen sus conocimientos y experiencia.

* + 1. **Funciones principales**

Las funciones de la aplicación serán:

-Historial clínico: Cada paciente tiene un listado de todos sus antecedentes médicos, los cuales servirán de ayuda para que el médico que ayude al paciente tenga ciertas referencias a la hora de tratar sus casos.

-Primeros auxilios: Muestra un listado con diversos ejemplos (los más comunes) donde hay que dar primeros auxilios y a su vez cómo tratarlos en caso de que cierta situación lo requiera.

-Diagnósticos: Para la búsqueda de un diagnóstico, el paciente introduce una serie de datos a unas preguntas que le hace el sistema con el fin de ir descartando y asegurando un diagnóstico claro.

* + 1. **Aspectos de rendimiento**
    2. **Restricciones y técnicas de gestión**

El proyecto se desarrolla entre siete componentes, los cuáles nos distribuimos el trabajo para tener las diferentes partes terminadas en el tiempo que nos habíamos propuesto. El lenguaje de programación que vamos a emplear para el desarrollo de la aplicación será Java.

* 1. **Modelo de proceso**

Proceso unificado de desarrollo, que se caracteriza por el uso casos de uso, enfoque a los distintos riesgos, naturaleza iterativa e incremental y centrado en la arquitectura. Consta de 4 fases: inicio, elaboración, construcción y transición.

1. **Estimaciones del proyecto**
   1. **Datos históricos**
   2. **Técnicas de estimación**
   3. **Estimaciones de esfuerzo, coste y duración**
2. **Estrategia de gestión del riesgo**
   1. **Introducción: Estudio de los riesgos**

La gestión de riesgos es un elemento importante para llevar a cabo con éxito el desarrollo del proyecto. Para ello, realizaremos una identificación y análisis de los riesgos que pueden potencialmente interrumpir y producir retraso a lo largo del proceso de desarrollo, incluyendo posibles soluciones a aquellos riesgos que pueden entorpecer el proyecto.

Entendemos por “**riesgo**” cualquier cosa que ponga en peligro el plan de proyecto o provoque efectos negativos en el desarrollo del proyecto.

Dicho riesgo se caracteriza básicamente por dos factores: incertidumbre y cambio.

* 1. **Priorización de riesgos del proyecto**

Se hace necesario ordenar los riesgos en función de su importancia para determinar cuáles se deben solucionar antes y a cuáles hay que asignarle más recursos.

Vamos a proceder a ordenar teniendo en cuenta su exposición al riesgo **(ER)** que se define como el producto entre la probabilidad **(P)** y la consecuencia **(C)**.

|  |
| --- |
| ER = P\*C |

La Probabilidad puede ser:

* **Muy alta ->** (75% < Probabilidad). (P = 1)
* **Alta ->** (50% < Probabilidad < 75%). (P = 0.1)
* **Media ->** (25% < Probabilidad < 50%). (P = 0.01)
* **Baja ->** (10% < Probabilidad < 25%). (P = 0.001)
* **Muy baja ->** (Probabilidad < 10%). (P = 0.0001)

La Consecuencia pueden ser:

* **Insignificante ->** Puede solventarse fácilmente. (C = 1)
* **Tolerable ->** Puede solventarse con pocas consecuencias negativas. (C = 2)
* **Serio ->** Es peligroso para el desarrollo del plan de proyecto. (C = 3)
* **Catastrófico ->** Puede conllevar a la cancelación del proyecto. (C = 4)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Nº** | **ER** | **Riesgo** | **P** | **C** |
| 1 | 0.3 | Tiempo requerido para desarrollar el software está subestimado | Alta | Serio |
| 2 | 0.2 | Fallos de implementación | Alta | Tolerable |
| 3 | 0.03 | Se proponen cambios en los requerimientos que requieren rehacer el diseño | Media | Serio |
| 4 | 0.02 | Falta de tiempo para puesta en Producción y pruebas | Media | Tolerable |
| 5 | 0.02 | Fallos técnicos e indisponibilidad del sistema | Media | Tolerable |
| 6 | 0.02 | El tamaño del SW está subestimado | Media | Tolerable |
| 7 | 0.004 | Pérdida de información | Baja | Catastrófico |
| 8 | 0.003 | Capacitación del personal | Baja | Serio |
| 9 | 0.003 | Problemas inherentes a la Base de Datos | Baja | Serio |
| 10 | 0.002 | Abandono de miembros del equipo | Baja | Tolerable |

* 1. **Plan de gestión del riesgo Reducción, supervisión y gestión del riesgo**

3.1. Riesgo:  Tiempo requerido para desarrollar el software está subestimado

3.3.1.1 Reducción

Ayudarnos de herramientas para la planificación temporal del proyecto tales como diagramas GANTT.

Conocer bien los recursos con que contamos y su disponibilidad tanto presentes como futuras.

3.3.1.2. Supervisión

Controlar los plazos de entrega y fomentar la comunicación entre los integrantes del grupo a fin de evitar duplicidades o dejar temas sin abordar por no tener persona asignada.

3.3.1.3. Plan de Contingencia

Redimensionar la carga de trabajo de manera equitativa entre los integrantes con el fin de cumplir plazos estimados.

Reconfigurar parte del proyecto para poder simplificar o modificar o eliminar algún aspecto del mismo.

3.3.2. Riesgo:  Fallos de implementación

3.3.2.1 Reducción

Al comienzo de cada módulo se releen con atención los requisitos para asegurarnos de que la programación se ajuste a ellos.

Aumentar la frecuencia de las pruebas y revisiones disminuirá el posible impacto que pueda tener un fallo sobre el resto del proyecto.

3.3.2.2. Supervisión

Diseñar una batería de pruebas para poder medir el correcto funcionamiento del sistema.

3.3.2.3. Plan de Contingencia

La persona o personas encargadas se disponen a solucionarlo de inmediato, en especial si se tratara de una parte crítica del sistema o hubiese otros módulos que dependen de la parte errónea.

3.3.3. Riesgo:  Se proponen cambios en los requerimientos que requieren rehacer el diseño

3.3.3.1 Reducción

En la medida de los posible se debe tener a priori una “visibilidad” del proyecto lo más cerrada posible, así como su alcance y posibles vulnerabilidades que minimicen la ocurrencia de nuevas funcionalidades una vez avanzado ya el proyecto.

3.3.3.2. Supervisión

En cada etapa repasar si el desarrollo realizado se ciñe a los requisitos especificados.

Establecer reuniones periódicas entre todos los integrantes que den una idea general de la marcha del proyecto.

3.3.3.3. Plan de Contingencia

Se analiza cuál de los integrantes del grupo puede dedicarse en función de su experiencia y cercanía a las nuevas funcionalidades que se proponen, a la implementación de los nuevos requisitos.

3.3.4. Riesgo:  Falta de tiempo para puesta en Producción y pruebas

3.3.4.1 Reducción

Hay que tener muy en cuenta la necesidad de realizar pruebas sobre de funcionamiento. Tanto a nivel funcionales, como de disponibilidad, tiempos de respuesta, pruebas de estrés a la BBDD, etc.

Incluir dichas pruebas en la estimación de tiempos del proyecto y sobredimensionada en una o dos jornadas.

3.3.4.2. Supervisión

Establecer pruebas no sólo al final del proyecto, sino durante el mismo.

3.3.4.3. Plan de Contingencia

Establecer un sistema de “Gestión de Configuración del Software” y dentro de éste “Gestión del Cambio” que permita de una manera dinámica y fluida comunicar una eventual incidencia.

3.3.5. Riesgo:  Fallos técnicos e indisponibilidad del sistema

3.3.5.1 Reducción

Prestar atención a las indicaciones de esta asignatura y, caso de ser necesario, informarse sobre otras cosas que vayan a ser necesarias para la realización del proyecto (como bases de datos). También favorable trabajar en grupos de más de una persona, para disminuir la probabilidad de que se cometa un fallo. Aumentando la frecuencia de las revisiones podremos detectar antes los fallos para que afecte lo menos posible al resto del proyecto.

3.3.5.2. Supervisión

Debemos dedicar suficiente tiempo a la prueba y a la revisión, preferiblemente por distintos componentes del grupo a los que lo programaron.

3.3.5.3. Plan de Contingencia

Una vez que se ha detectado el fallo hay que solucionarlo inmediatamente, especialmente si hay otros módulos que dependan de él, para minimizar el impacto.

3.3.6. Riesgo:  El tamaño del SW está subestimado

3.3.6.1 Reducción

Realizar a priori un análisis realista sobre el sistema que se quiere desarrollar huyendo de idealizaciones y tratando de evitar nuevas funcionalidades que no hayan sido previamente incluidas.

En este sentido se puede establecer que se trate de una primera versión de un proyecto al que puedan incorporarse posteriormente nuevas versiones o módulos añadidos.

3.3.6.2. Supervisión

Análisis del factor transcurso del proyecto / tiempo para evaluar cualquier desfase entre el análisis de la aplicación y el costo de su programación.

Se deben realizar reuniones oportunas para evaluar dicho aspecto

3.3.6.3. Plan de Contingencia

Si se percata un desfase en cuanto a la carga de trabajo de desarrollo de software y los recursos tanto humanos como temporales dedicados a esta carga se deberán redistribuir las tareas asignadas inicialmente y establecer un sistema de prioridades en las tareas.

3.3.7. Riesgo:  Pérdida de información

3.3.7.1 Reducción

Este factor de riesgo aún siendo uno de los de mayor criticidad se puede controlar con unas medidas básicas de prevención, lo que baja drásticamente la probabilidad de que se produzca.

Con unas medidas básicas de copias de seguridad y buenos sistemas que garanticen el trabajo en grupo de manera concurrente no debería haber mayor problema.

3.3.7.2. Supervisión

Este riesgo se auto-supervisa por todos los elementos del grupo a lo largo de la construcción del proyecto.

3.3.7.3. Plan de Contingencia

Reestablecer la información perdida mediante las oportunas operaciones de “Restore”

3.3.8. Riesgo:  Capacitación del personal

3.3.8.1 Reducción

Los integrantes del grupo son personas muy implicadas con el proyecto, dedicando buena parte de su tiempo al mismo.

Si bien se aprecian deficiencias formativas, se suplen con los apoyos del resto de los compañeros.

3.3.8.2. Supervisión

En este sentido se hace indispensable la buena comunicación en el grupo, poniendo en común las dificultades encontradas y buscando de manera conjunta las soluciones.

También se dispone de las herramientas informáticas oportunas para compartir información, que en nuestro caso son utilidades del estilo  de Google Drive y Google Docs, etc.

3.3.8.3. Plan de Contingencia

Cualquier deficiencia por parte de algún miembro del grupo deberá ser adquirida por otro miembro, y tras ello, ponerla en común para el conocimiento de todos.

3.3.9. Riesgo:  Problemas inherentes a la Base de Datos

3.3.9.1 Reducción

Podemos diferenciar dos aspectos:

* Diseño adecuado de la Base de Datos

Se hace necesario realizar y evaluar un correcto análisis del Modelo de Datos, que cubra todos los aspectos del aplicativo.

* Dimensionar adecuadamente el tamaño de la Base de Datos

Aquí se trata de establecer los correctos parámetros de tamaño de almacenamiento y tiempos de respuesta aceptables

En ambos casos se pueden establecer “pruebas de estrés” para evaluar el correcto funcionamiento de la BBDD

3.3.9.2. Supervisión

El Modelo de Datos debe ser una tarea que debe quedar bastante “cerrada” en las primeras fases de Desarrollo del sistema.

Se debe someter a estudio por parte de todos los miembros del grupo, ya que previsiblemente, la BBDD afectará a todo el sistema.

3.3.9.3. Plan de Contingencia

Revisión de la Base de Datos y puesta a punto de nuevas modificaciones.

3.3.10. Riesgo:  Abandono de miembros del equipo

3.3.10.1 Reducción: cómo evitar que suceda

Este riesgo se autorregula por el propio interés de los integrantes del grupo en el proyecto. Si bien, no se está ni mucho menos exento de sufrir una baja en el equipo.

3.3.10.2. Supervisión: controlar riesgo y supervisar los pasos de reducción

Comunicación. Nada que observar.

3.3.10.3. Plan de Contingencia: Gestión si el Riesgo sucede

Redistribución de las funciones asignadas a la persona saliente entre el resto de integrantes.

* 1. **Planificación temporal del Control de Riesgos**

Lo ideal sería adoptar una estrategia proactiva, es decir, prevenir los riesgos antes de que ocurran, pero no siempre es posible, adoptando una estrategia reactiva, de reacción al riesgo ya acaecido.

* 1. **Resumen**

La Gestión de los Riesgos se revela como una parte esencial del Plan de Proyecto y se erige como una herramienta útil y poderosa, que, sin ella, estaríamos expuestos a situaciones a veces dramáticas que podrían poner en riesgo la viabilidad de todo el proyecto.

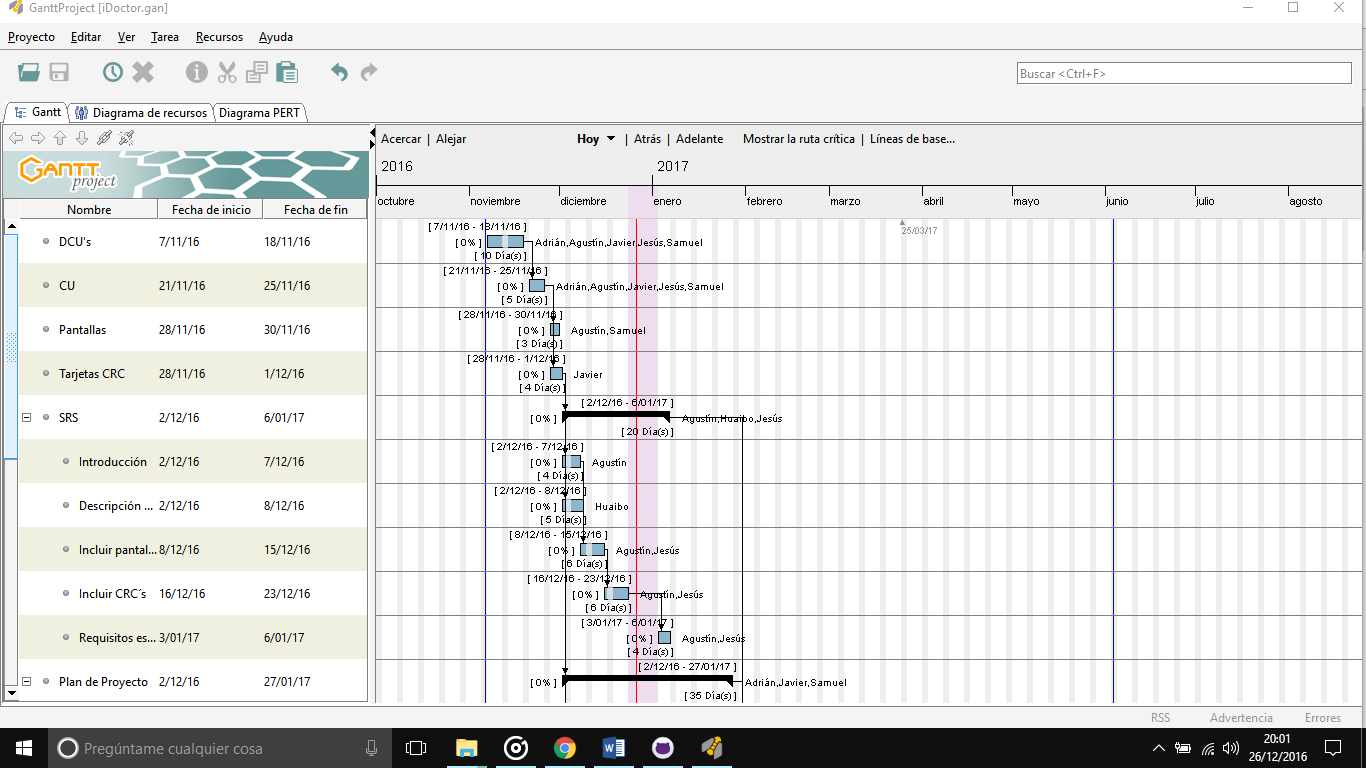
Aun así, es de reconocer, que a veces puede ser complicada esta gestión, debido, en gran parte, al carácter puramente aleatorio de algunos riesgos. Pero en otros muchos casos una buena planificación de la gestión de los riesgos puede ayudar a que si estos se produjeran su impacto sea menor o incluso nulo.

1. **Planificación temporal**
   1. **Estructura de descomposición del trabajo/Planificación temporal**

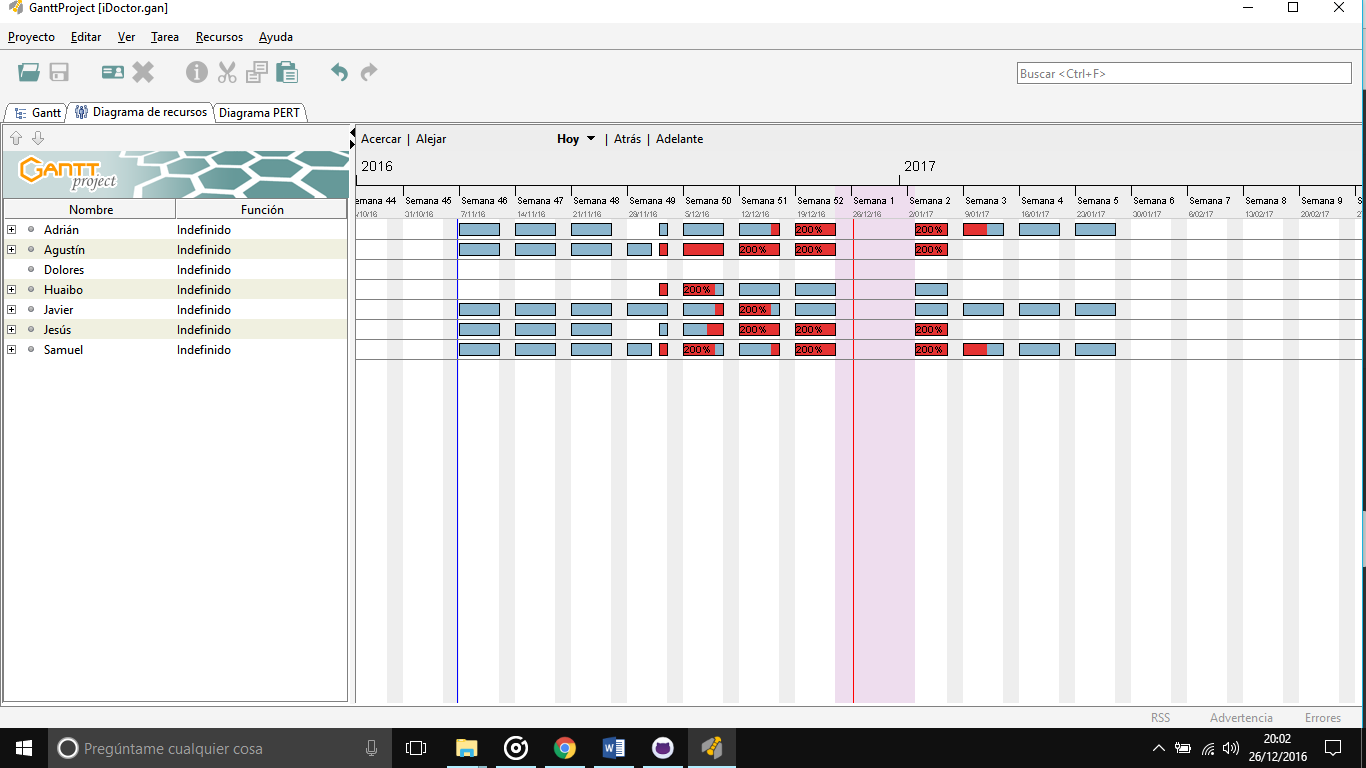
Nuestro proyecto se va a elaborar mediante un proceso unificado de desarrollo, es por eso que vamos a dividir el proyecto en subsistemas lo que nos va a permitir trabajar en varias partes a la vez.

La organización del proyecto será:

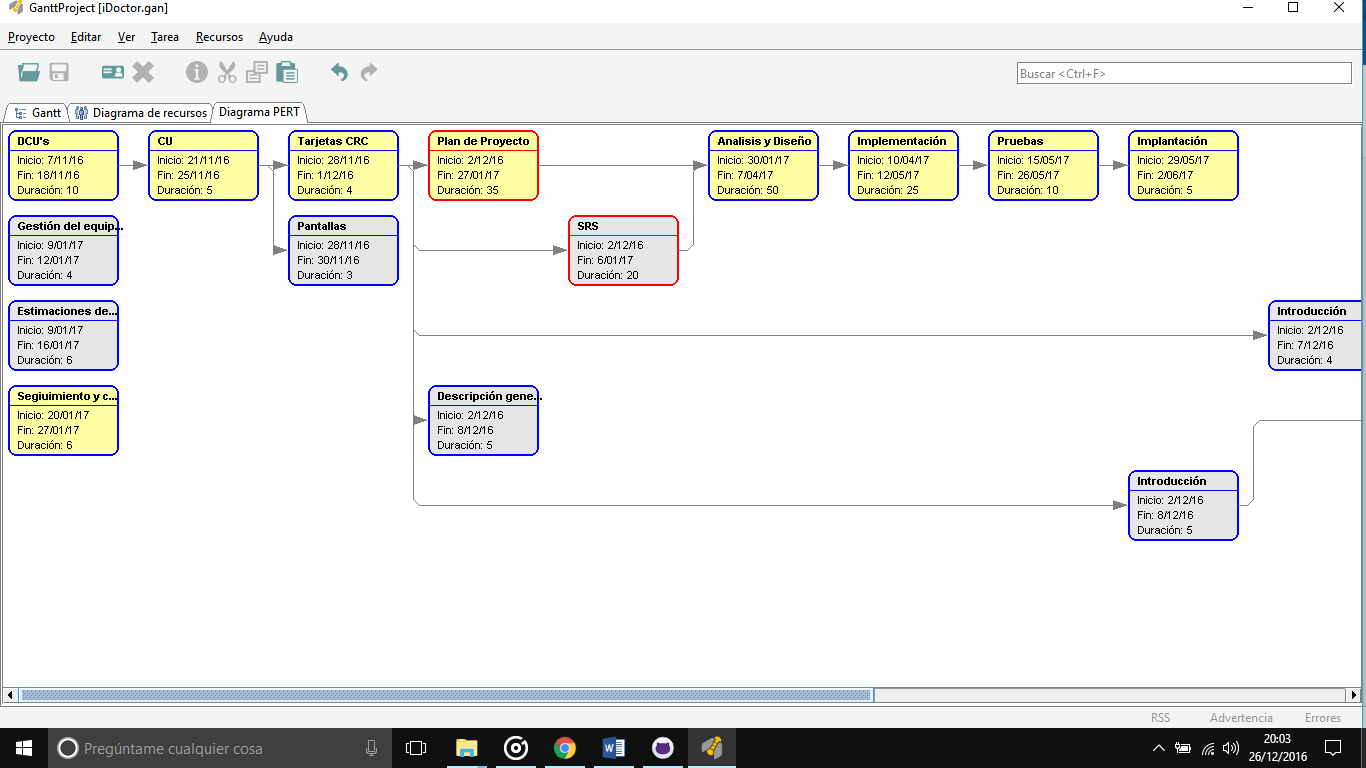
* Análisis: recogemos la información, realizamos los requisitos y definimos los usuarios.
* Diseño: realizamos un diseño del entorno gráfico y definimos los estilos que va a tener el programa.
* Documentación: recopilamos toda la información en el SRS que nos servirá para realizar el diseño del programa.
* Construcción: realizamos la codificación del programa, las pruebas necesarias y el ensamblaje de todas las partes del proyecto.
  1. **Gráfico Gantt**



* 1. **Red de tareas**



* 1. **Tabla de uso de recursos**



1. **Recursos del proyecto**
   1. **Personal**

El grupo iDoctor Team consta de 7 miembros, los cuáles serán repartidos de la siguiente manera:

* 2 revisores de CUs, y su función se limita a asegurar que los distintos casos de uso son coherentes y prácticos.
* 2 diseñadores gráficos, que se encargan de hacer una interfaz amigable y fácil de entender. A su vez se encargarán de la estructura de la programación.
* 3 programadores.
  1. **Hardware y software**

5.2.1 **Hardware**

El hardware empleado para desarrollar la aplicación será sencillamente de ordenador de sobremesa-

5.2.2 **Software**

EL software utilizado para el desarrollo de la aplicación será:

* Eclipse: Es una plataforma de software compuesto por un conjunto de herramientas de programación en Java y C++, compatible con SQL y elementos Web.
* GitHub: Herramienta de control de versiones.
* Balsamiq: Desarrollo gráfico de la aplicación.
  1. **Lista de recursos**

1. **Organización del personal (Gestión del Equipo)**
   1. **Estructura de equipo (si procede)**
   2. **Informes de gestión**
2. **Mecanismos de seguimiento y control**

La finalidad básica del seguimiento es la observación de la persona u objeto que se controla, creando así sistemas de control en el que es fundamental especificar qué cambio se ha realizado en dicha tarea. Para ello haremos uso de un criterio de versionado, así como el establecimiento de pasos para realizar cambios.

La herramienta de versionado que utilizaremos será Git y el servidor GitHub.

* 1. **Garantía de calidad y control (Plan de Calidad)**

Para conseguir crear software de calidad realizaremos revisiones técnicas formales luego de cada tarea finalizada. Se designará a alguien que revise el trabajo hecho por los otros integrantes del grupo y así poder encontrar errores o simples modificaciones para que tenga más sentido la tarea. A partir de aquí, en la siguiente reunión se tendrán que exponer las modificaciones realizadas y sus consecuencias.

* 1. **Gestión y control de cambios (Plan GCS)**
     1. **Introducción: Propósito, Alcance, Definiciones, Referencias**
     2. **Tipos de artefactos a gestionar (los ECSs)**
     3. **Criterios y protocolos para Nombrar los ECSs**
     4. **Responsable de los procedimientos de GCS y de la creación de Líneas Base.**
     5. **Políticas para el Control de Cambios y la Gestión de Versiones**

- Estándares (Plantillas que usas en documentos; Pasos a dar en un cambio; Acuerdos en productos) - Comité de Cambios: Quién hace qué - Herramientas de gestión de versiones – Repositorios - Gestión de Proyectos (si la usas)

* + 1. **Registros para mantener el rastro de los cambios**

(Qué ha pasado, pasos para usar herramientas del punto 7.2.5 )

1. **Apéndices**