La documentación de los programas es un aspecto sumamente importante, tanto en el desarrollo de la aplicación como en el mantenimiento de la misma. Mucha gente no hace este parte del desarrollo y no se da cuenta de que pierde la posibilidad de la reutilización de parte del programa en otras aplicaciones, sin necesidad de conocerse el código al dedillo.   
  
La documentación de un programa empieza a la vez que la construcción del mismo y finaliza justo antes de la entrega del programa o aplicación al cliente. Así mismo, la documentación que se entrega al cliente tendrá que coincidir con la versión final de los programas que componen la aplicación.   
  
La documentación se puede dividir claramente en dos categorías, interna y externa:

* Interna: Es aquella que se crea en el mismo código, ya puede ser en forma de comentarios o de archivos de información dentro de la aplicación.
* Externa: Es aquella que se escribe en cuadernos o libros, totalmente ajena a la aplicación en sí.

Los documentos sobre los proyectos son susceptibles de variar a lo largo del tiempo conforme van apareciendo contingencias y el proyecto evoluciona. Es importante encontrar la manera de organizar las versiones de los documentos para que no se pierda ningún dato importante y estén ordenados.

Para llevar un control de versiones usaremos la herramienta GitHub. Una herramienta basada en Git que es un sistema de control de versiones distribuido, es decir, en un servidor se guarda todas las versiones del proyecto y los desarrolladores que estén trabajando en la aplicación pueden descargarse las versiones que quieran.

Usaremos Git debido a que Git modela sus datos como un conjunto de instantáneas de un mini sistema de archivos. Cada vez que confirmas un cambio, o guardas el estado de tu proyecto en Git, él básicamente hace una foto del aspecto de todos tus archivos en ese momento, y guarda una referencia a esa instantánea. Para ser eficiente, si los archivos no se han modificado, Git no almacena el archivo de nuevo, sólo un enlace al archivo anterior idéntico que ya tiene almacenado.

En cuanto al proyecto, podemos almacenarlos en ficheros y controlándolo por versiones por ejemplo EclipseV1.1. EclipseV1.2… Dentro de cada fichero incluiríamos el código fuente que actuaría como documentación interna y como documentación externa podríamos usar especificaciones de requerimientos de software como es el IEEE 830. Este estándar estaría formado de la siguiente manera:

**1. Introducción (Se definen los fines y los objetivos del software)**

A. Referencia del sistema

B. Descripción general

C. Restricciones del proyecto

**2. Descripción de la información (Descripción detallada del problema,**

**incluyendo el HW y SW necesario)**

A. Representación del flujo de la información.

B. Representación del contenido de la información.

C. Descripción de la interfaz del sistema.

**3. Descripción funcional (Descripción de cada función requerida, incluyendo**

**diagramas)**

A. Partición funcional

B. Descripción funcional

1. Narrativa de procesamiento

2. Restricciones/Limitaciones.

3. Requisitos de rendimiento.

4. Restricciones de diseño

5. Diagramas de soporte

C. Descripción del control

1. Especificación del control

2. Restricciones de diseño

**4. Descripción del comportamiento (comportamiento del SW ante sucesos**

**externos y controles internos)**

A. Estados del sistema

B. Sucesos y acciones

**5. Criterios de validación.**

A. Límites de rendimiento

B. Clases de pruebas

C. Respuesta esperada del SW

D. Consideraciones especiales

**6. Bibliografía**

**7. Apéndice.**

Toda esta documentación la llevaríamos en una carpeta aparte distinguiendo entre código y documentación. En la documentación estarían estos puntos guardados en un archivo de Google Docs. debido a que nos aportan aún más control de versiones