Proyecto de Ingeniería de Software II

Acuña Yeomans Eduardo
Contreras Mejía Daniel
Valle Ruiz Francisco Manuel

Lunes 2 de Junio del 2014

1. Presentación

1.1. Sobre Sabrosoftware

Sabrosoftware es la compañía ficticia de desarrollo de software con mayor prestigio en la clase de Ingeniería de Software II impartida por el profesor Adrián en la segunda mitad del año 2014. Engendrada en el 2013 como parte de un proyecto de Ingeniería de Software I, Sabrosoftware ha aprendido de sus errores en el previo trabajo y planea posicionarse como la compañía ficticia de desarrollo de software de toda la región sur-este de la Universidad de Sonora campus Hermosillo.

Figura 1: Logo de Sabrosoftware

1.2. Sobre el equipo

El equipo de trabajo es conformado por tres estudiantes de la licenciatura en Ciencias de la Computación:

Eduardo Acuña Yeomans

Estudiante de sexto semestre, he escrito pequeños programas y trabajando con los otros integrantes de Sabrosoftware dentro y fuera del ámbito escolar.

Tiene experiencia moderada en los lenguajes de programación de C, C++ y Scheme. Tiene poca experiencia en los lenguajes de programación de Java y Python.

Daniel Contreras Mejía

Estudiante del octavo semestre, entusiasta en el área de la computación nivel amateur; cuenta con experiencia media en Ruby, C++, C#, HTML, CSS y es fluido en el lenguaje Inglés a nivel profesional.

Francisco Manuel Valle Ruiz

Estudiante del sexto semestre, se graduó del bachillerato con especialidad técnica en Informática (2008 - 2011) y experiencia media en C++, HTML, CSS

1.3. Flujo y relaciones de trabajo

- **Líder del proyecto:** Eduardo Acuña Yeomans
- **Encargado del diseño interfaz:** Daniel Contreras Mejía
- **Encargado del diseño algorítmico:** Francisco Manuel Valle Ruiz
- **Encargado de documentación:** Eduardo Acuña Yeomans
- **Encargado de pruebas:** Francisco Manuel Valle Ruiz
- **Encargado de mantenimiento técnico:** Daniel Contreras Mejía

El flujo de comunicación y trabajo se basa en desarrollo de documentos y códigos en plataformas colaborativas como Google Docs y GitHub, así como el uso de una lista de correos.

2. Descripción del Proyecto

El problema planteado por el profesor Adrián es el de envío y recepción de oficios por parte de profesores y administrativos de la Universidad de Sonora. Nos mostró que la mayoría de los oficios que se redactan tienen practicamente el mismo formato y que el procedimiento de recepción es el de mandar el oficio al destinatario y que al llegar a la oficina correspondiente el oficio sea sellado de *recibido*.

Se plantea desarrollar un sistema que emule este procedimiento. Las ideas que se mencionaron en clase fueron de generar documentos en formato PDF y manejar una especie de bandeja virtual para clasificar los documentos como recibidos o no recibidos, así como simplificar el proceso de darle formato a los oficios; se entendió que esta parte del sistema funcionara como el "visto" del facebook.

3. Presupuestación

La presupuestación del proyecto se realizó de dos maneras:

3.1. En base a Líneas de Código

Separamos el proyecto en diferentes secciones para poder aproximarnos a una mejor aproximación:

- Interfaz gráfica **LDC: 500**
- Procesamiento de entradas **LDC: 100**
- Generación de documentos **LDC: 500**
- Procesamiento de salidas **LDC: 100**
- Intercambio de documentos entre usuarios **LDC: 200**

Esta asignación numérica resulta con un total de **1400** líneas de código. Asumiendo que el sueldo mensual por persona es de $\$5,000^{00}$ y que la cantidad de líneas de código al mes son 90. Determinamos que el precio por línea de código es de $\$55^{00}$. Por lo tanto, el presupuesto estimado del proyecto, basandonos en líneas de código es de $\$77,000^{00}$.

3.2. En base a Puntos de Función

Al obtener los *puntos de función* totales del proyecto se establecerán una serie de medidas para poder realizar la estimación del presupuesto.

1. **Entradas externas**
   - Datos del oficio a mandar
   - Datos del usuario
2. **Salidas externas**
   - Interfaz gráfica
   - Mensajes de error
   - Visualización de documentos
3. **Consultas externas**
4. **Archivos lógicos internos**
   - Estructura del formato
   - Estructura del contenido del oficio
   - Red de contactos por usuario
5. **Archivos de interfaz externas**
   - Datos de los usuarios

Ya que no se ha realizado un proyecto como este, se estimaron como tareas "promedio" y es así como se ponderan estos dominios para el conteo de los puntos de función. Por lo que el *conteo total ponderado* resulta 60.

Se establecen los factores de ajuste:

1. ¿El sistema requiere respaldo y recuperación confiable? **2**
2. ¿Comunicación de datos especializada es requerida para la transferencia de información? **0**
3. ¿Hay funciones de procesamiento distribuido? **0**
4. ¿Es el rendimiento crítico? **1**
5. ¿El sistema será ejecutado en ambientes altamente utilizados? **2**
6. ¿Requiere entradas de datos en línea? **5**
7. ¿Se requieren múltiples pantallas u operaciones para la transacción de información en línea? **1**
8. ¿Los archivos lógicos internos son actualizados en línea? **3**
9. ¿Hace uso de archivos de entrada/salida complejos? **2**
10. ¿Realiza procesamiento interno complejo? **2**
11. ¿El código está diseñado para ser reutilizado? **2**
12. ¿Se incluye instalación y conversión en el diseño? **0**
13. ¿El sistema está diseñado para múltiples instalaciones en diferentes organizaciones? **0**
14. ¿La aplicación está diseñada para facilitar cambios y fácil uso por el usuario? **0**

Dada esta valoración el *total de factores de ajuste* es **20**; para obtener el valor de *puntos de función (PF)* se usa la fórmula:

$$
\begin{aligned}
PF &= ConteoPonderado \times [0{,}65 + 0{,}01 \times ConteoAjuste] \\
PF &= 60 \times [0{,}65 + 0{,}01 \times 20] \\
PF &= 60 \times [0{,}65 + 0{,}2] \\
PF &= 60 \times 0{,}85 \\
PF &= 51
\end{aligned}
$$

En teoría este valor numérico es útil conociendo información sobre los integrantes del equipo de trabajo como:

- Promedio de líneas de código por PF.

- Promedio de PF al mes.
- Promedio de errores por PF.

Sin embargo, estos datos no se tienen en un primer proyecto, por lo tanto esta estimación será ajustada conforme se avance el proyecto y conforme se desarrollen mas proyectos. Estos primeros valores son relacionados con la estimación de líneas de código, la cual también está basada en una idea muy vaga del proyecto:

- Promedio de líneas de código por PF: **27.45**
- Promedio de PF al mes: **3.28**

4. Ámbito y Descomposición del software

4.1. Ámbito del proyecto

Actualmente en la Universidad de Sonora, varios procedimientos oficiales requieren el envío de oficios. Usualmente estos oficios son documentos con un formato determinado que un empleado redacta para imprimirlo y mandarlo a algún otro empleado u oficina de la Universidad, cuando este documento es recibido por el destinatario es sellado de recibido para tener como constancia que la comunicación se efectuó de manera oficial.

Dado que hay varias partes de este proceso que se pueden automatizar, se plantea elaborar un sistema que pueda emular esta comunicación oficial de manera virtual, comprendiendo la redacción del oficio, la entrega al destinatario y el sellado de recibido.

4.2. Descomposición del proyecto

El proyecto se compone fundamentalmente en tres partes:

- **Captura de información:** Emisor, receptor y contenido del oficio.
- **Procesamiento de información:** Transformar la estructura básica en un oficio con el formato usual.
- **Envío y sellado:** Entrega virtual del documento al destinatario y confirmación de recibido cuando el destinatario consulte los oficios pendientes.

La *captura de información* se encarga de determinar que los datos ingresados sean válidos, se contempla la implementación de una interfaz de usuario gráfica en una página web.

El *procesamiento de información* consistirá de varios algoritmos que manejarán una estructura determinada de los datos para realizar un acomódo adecuado con el formato del oficio establecido. El resultado de este proceso es un archivo en formato PDF.

El *envío y sellado* se encargará de informar a otro usuario que un documento en PDF le fué enviado y al momento de que este usuario revise el PDF se actualizará el documento como recibido.

Se opta por desarrollar una aplicación web para la implementación de este sistema. La ventaja es tanto para el usuario como para los desarrolladores ya que no será necesario descargar un programa para utilizar el sistema y los desarrolladores tendrán un control sobre toda la información debido a que será alojada en un servidor web.

5. Planificación

5.1. Análisis y selección de modelo

Se utiliza el modelo por *prototipos* ya que el cliente solo planteó una idea general del programa y será útil para los desarrolladores ir dejando los detalles sutiles para el final (debido a la poca experiencia que se tiene con este tipo de proyectos).

5.2. Lista de actividades

1 **Analizar el problema: (3 días)**

- 1.1 Reunión con el cliente.
- 1.2 Aplicar un cuestionario de espectativas.
- 1.3 Determinar los puntos clave del sistema.

2 **Diseñar el primer prototipo: (3 semanas)**

- 2.1 Determinar el cuerpo de conocimiento requerido y establecer las técnicas o métodos que el equipo debe aprender.
- 2.2 Establecer la interacción entre los diferentes componentes del software.
- 2.3 Analizar la manera en como el usuario interaccionará con el sistema.
- 2.4 Diseñar la interfaz de usuario.
- 2.5 Elegir o diseñar los algoritmos de generación de formato.
- 2.6 Elegir o diseñar los algoritmos de generación de archivos PDF.
- 2.7 Implementar el sistema.

3 **Pruebas al prototipo: (1 semana)**

- 3.1 Probar el software con usuarios ficticios.
- 3.2 Probar el software con entradas usuales.
- 3.3 Probar el software con entradas erráticas.

4 **Correcciones: (1 semana)**

4.1 Corregir validación de datos.

4.2 Corregir algoritmos de transformación.

4.3 Corregir interfaz de usuario.

5 **Documentación: (4 semanas)**

5.1 Documentación de usuario.

5.2 Documentación de desarrolladores.

5.3 Informe para el cliente.

6 **Reunión con el cliente: (2 días)**

6.1 Presentación del prototipo.

6.2 Anotación de cambios o mejoras.

...

5.3. Red de tareas

En la *Figura 2* se aprecia la dependencia de las actividades.

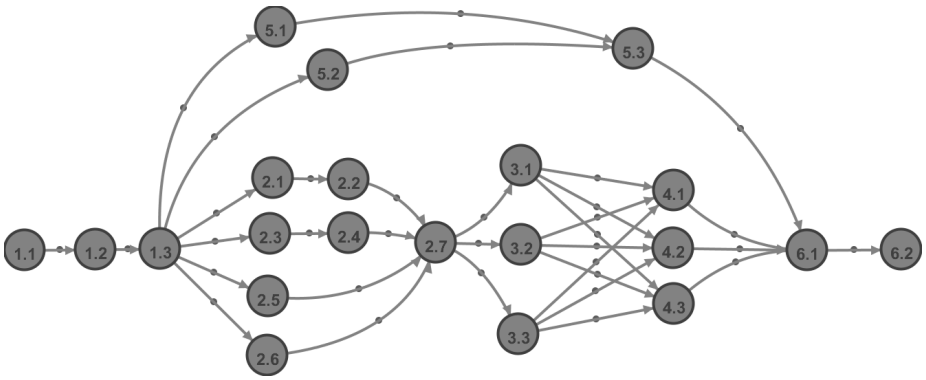

Figura 2: Dependencia de tareas

5.4. Carga de trabajo

- **Acuña Yeomans Eduardo:** 1.1, 1.2, 1.3, 2.7, 5.1, 5.2, 5.3, 6.1, 6.2.
- **Contreras Mejía Daniel:** 1.1, 2.1, 2.2, 2.3, 2.4, 2.7, 4.1, 4.3, 6.1.
- **Valle Ruiz Francisco Manuel:** 1.1, 2.5, 2.6, 2.7, 3.1, 3.2, 4.2, 6.1.

5.5. Diagrama de Gantt

En la *Figura 3* se aprecia la distribución temporal de las actividades.

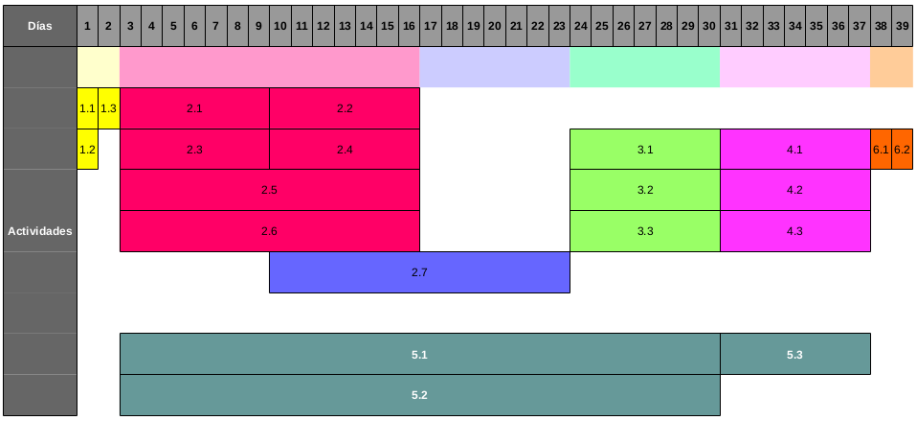

Figura 3: Diagrama de Gantt

6. Análisis de Riesgo

6.1. Posibles problemas

- Pérdida de datos
- Ausencia de integrantes del equipo
- Imposibilidad de comunicación por incompatibilidad de horario

6.2. Plan de contención

Para evitar los potenciales riesgos que se pueden presentar a lo largo del desarrollo del proyecto, se opta por trabajar de manera defensiva y prevenir los sucesos con mas probabilidad de ocurrencia.

- Toda la documentación y código del proyecto será trabajada sobre un repositorio en Github, en donde se podrá observar y obtener métricas de cómo evoluciona el trabajo. De esta manera, cualquier problema de pérdida de datos, será minimizado por el respaldo permanente de cada una de las versiones (con la posibilidad de rastrear cada cambio).
- Si por algún motivo un integrante del equipo no puede realizar la labor asignada, el líder del proyecto se hará cargo de dicha labor de manera personal.
- La comunicación se realizará obligatoriamente vía correo electrónico y presencial de manera opcional.