Proyecto Final

Estefanía García González, Sebastián Mora Sabogal

5 de mayo de 2019

Índice general

Índice de figuras

Parte I

PROYECTO

Capítulo 1

Caso de Estudio

1.1. Introducción

Desde que el ser humano cuenta con raciocinio , ha buscado organizarse, desarrollar metodologías y nuevas tecnologías que faciliten su diario vivir. Ha habido un recorrido histórico en el cual las necesidades humanas de optimización de tiempo y recursos han ido en aumento, así mismo las soluciones a éstas. En los últimos años se ha podido apreciar una constante migración al uso de tecnologías de la información que permiten realizar a cabo tareas en todos los ámbitos de forma óptima. Uno de los actores que más se han visto inmersos en la revolución digital son los estudiantes, pero en su contexto universitario, hace falta desarrollar estrategias que le permitan mejorar la gestión de tiempo de sus actividades académicas; por lo cual se buscará una solución tecnológica que se adapte a las necesidades de los universitarios.

1.2. Objetivo General

Desarrollar un software que gestione actividades y tiempos de las asignaciones académicas a estudiantes universitarios, utilizando los modelos y metodologías de ingeniería de software para mejorar la productividad del universitario.

1.3. Objetivos Específicos

1. Analizar el problema teniendo en cuenta la observación de las necesidades del estudiante, para así enfocarse en estos elementos primordiales a la hora de desarrollar el software.

2. Presentar una solución a nivel de software a partir del previo análisis del problema para finalmente implementarlo.

1.4. Descripción del problema

La vida universitaria y académica suele ser difícil de manejar debido a la cantidad de trabajos que se deben entregar diariamente, a la prioridad que cada una es para el usuario y a la gestión de tiempo para poder realizarlos. Tareas, trabajos, talleres y grandes proyectos son algunas de las actividades que un estudiante realiza durante su semestre; además de que cada uno tiene complejidad y tiempo de realización diferentes estimados por el estudiante. Una solución factible es la utilización de un software gestor de tareas orientado a la organización y optimización de actividades académicas.

1.5. Alcance

Este software tendrá la capacidad de gestionar los horarios de los estudiantes, añadir recordatorios de trabajos próximos a presentar y ofrecer el servicio de organizar en horarios la realización de las tareas pendientes. Esto se llevará a cabo de acuerdo a la complejidad de la actividad a realizar, en la cual se tomará en cuenta el nivel de dificultad, si se puede desarrollar en diferentes etapas y la fecha de entrega.

El estudiante estará en la capacidad de añadir actividades, determinar la complejidad de éstas y asignarles un horario de realización que puede ser repartido en varios bloques cuando la tarea requiere de mucho tiempo. Adicionalmente, las actividades podrán personalizarse añadiendoles objetivos a cumplir o subactividades.

Capítulo 2

Metodología

2.1. Introducción

contenido ...

2.2. Proceso de Software

Parte importante de un proyecto de software es definir el, o los ciclos de vida que se manejarán dentro del proyecto, ya que estos determinarán estrategias para planificar,desarrollar y mantener el software. Por esta razón, se definirá el modelo de procesos a utilizar, tomando en cuenta los siguientes criterios:

- Es necesaria una metodología que sea pertinente para un proyecto de software pequeño con pocos desarrolladores.
- Se considera importante la verificación en cada fase del ciclo de vida, ya que permite sentar buenas bases dentro del proyecto y reducir el riesgo.
- Además de la verificación, es necesaria una retroalimentación constante, ya que es posible ver con mayor claridad las falencias y carencias del proyecto.
- Como último criterio fundamental, se contempla la necesidad de desarrollar algunas partes de software de forma rápida, ya que esto facilitaría la retroalimentación del sistema.

Para cumplir con las pautas anteriormente mencionadas, los ciclos de vida que se elegirán son prototipo y V. Cada uno de estos modelos obedece solo a algunas de las especificaciones, pero juntos se complementan de la siguiente manera:

- El modelo V es perfecto para equipos de trabajo pequeños, ya que es sencillo, de fácil aprendizaje, robusto e incluye pruebas en cada fase, lo que facilita el trabajo cuando hay pocas personas.
- Gracias a los dos ciclos de vida, es posible hacer una verificación y retroalimentación de forma efectiva, ya que con el modelo en V se hacen pruebas en cada fase y con el prototipo es posible obtener resultados a corto plazo que se pueden ir revisando y evaluando.
- El modelo de prototipo brinda la posibilidad de construir partes del proyecto de forma prematura, por lo que es posible realizar pruebas y verificar qué cosas es necesario cambiar o añadir.

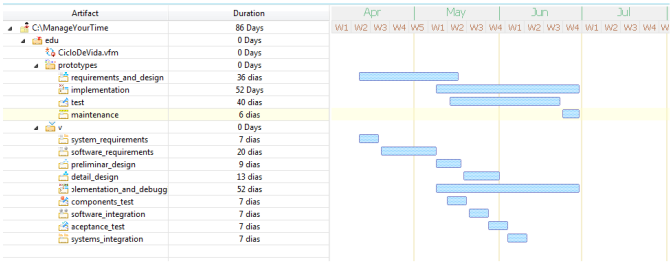


Figura 2.1:

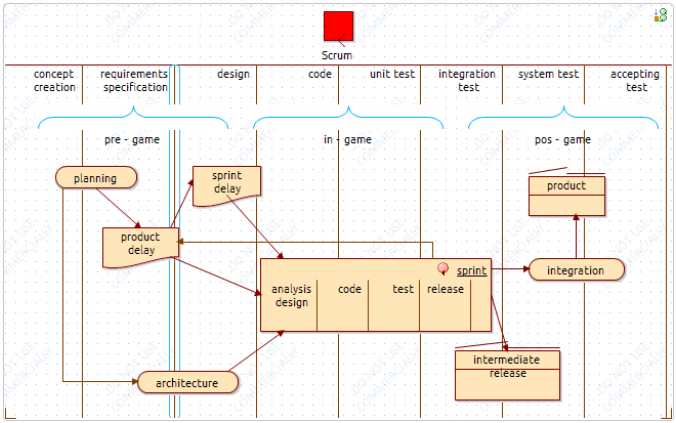
2.2.1. Metodología de implementación

Los criterios que se establecieron al momento de justificar la elección los procesos de software prototipo y V, cuentan con la misma validez para determinar la metodología de implementación, debido a que para esta etapa también es necesario tener un plan de acción que beneficie la gestión de tiempos del proyecto, complemente los procesos de software, cumpliendo un proceder de forma organizada.

2.3. Open Source

Desde que las personas empezaron a desarrollar software, han empezado a indagar en diferentes formas de realizar las cosas, a fin de obtener la solución computacional que solucione su necesidad. Con el tiempo estos pensamientos han devenido en ideologías que orientan la variedad de metodologías disponibles para desarrollar software.

El pensamiento o filosofía que entra en cuestión, es la del software libre, donde uno de sus principios, consiste en la reutilización del conocimiento, en este caso, el código. Es aquí donde entra el Open Source, que se relaciona con el código abierto, y con su revisión por parte de una comunidad de desarrolladores externos. Siguiendo el principio de filosofía libre, se pretende utilizar el concepto O.S con la intención de obtener una ayuda en momentos donde la implementación se torne complicada, llegandose a extrapolar a diversos casos en los que se necesite la apreciación del problema que se está trabajando por parte de un externo el cual ya lo haya desarrollado.


Scrum
concept creation
requirements specification
design
code
unit test
integration test
system test
accepting test
pre - game
in - game
pos - game
planning
sprint delay
product delay
product
sprint
analysis design
code
test
release
integration
intermediate release
architecture

Parte II

DISEÑO

Capítulo 3

Requerimientos

3.1. Introducción

Como cualquier proyecto de cualquier indole, un punto fundamental, consiste en conocer cual es la necesidad o el problema que se desea resolver, lo cual conlleva a saber qué es lo que se debe realizar, para solventar tales necesidades. Esta es la idea básica de los requerimientos, definir cuales son los aspectos que se deben realizar para complacer las necesidades de un cliente.

3.2. Requerimientos del Cliente

Se entienden como el qué es lo que el cliente esperará encontrar, cuando interactúe con la aplicación. Bajo la anterior premisa, se procede a definir los siguientes requerimientos de cliente:

1. Mostrar todas las tareas pendientes.
2. Mostrar las tareas pendientes por categoría.
3. Mostrar las tareas pendientes por su tipo.
4. Mostrar las tareas pendientes para una fecha.
5. Mostrar las tareas pendientes por dificultad.
6. Mostrar los horarios asignados para las tareas pendientes.
7. Alertar de la próxima entrega de una tarea.
8. Sugerir horarios de clase del usuario.
9. Sugerir cuánto tiempo podría tomar una tarea.
10. Sugerir tiempos de pausas activas durante la realización de una tarea.
11. Advertir si se debe sacrificar algún espacio destinado a descanso.
12. Advertir cuando no se alcanzará a terminar una tarea para una fecha especificada.

Capítulo 4

Interacción

4.1. Introducción

cntenido...

Capítulo 5

Clases

5.1. Introducción

cntenido...

Capítulo 6

Patrones

6.1. Introducción

cntenido...

6.2. Prototipo

Capítulo 7

Estados

7.1. Introducción

cntenido...

Capítulo 8

Componentes

8.1. Introducción

cntenido...

Capítulo 9

Nodos

9.1. Introducción

cntenido...

Capítulo 10

Actividades

10.1. Introducción

cntenido...

Parte III

REFLEXIONES

Capítulo 11

Conclusiones

11.1. Introducción

cntenido...

Bibliografía