

Universidad de Sevilla

Escuela Técnica Superior de Ingeniería Informática

Historia del Proyecto




Grado en Ingeniería Informática – Ingeniería del Software

Proceso de Software y Gestión 2

Curso 2022 – 2023


Fecha	Versión
26/02/2023	1.0

Grupo de prácticas: G4-43		
Autores por orden alfabético	Rol	Descripción del rol
Alcobendas Santos, Jose Javier - 53986326J	Developer	Miembro del equipo de desarrollo
Campos Garrido, Juan Jesús - 47547107N	Scrum Master	Encargado de facilitar la labor de Scrum
Nunes Ruiz, Javier - 29517615J	Developer	Miembro del equipo de desarrollo
Pizarro López, Eduardo - 77933507P	Developer	Miembro del equipo de desarrollo
Reyes Alés, David - 29504757N	Developer	Miembro del equipo de desarrollo

	Proceso de Software y Gestión 2 Documentación del Sprint 2
	Control de Versiones


Control de Versiones

Fecha	Versión	Descripción
15/02/2023	0.1	Creación del documento
22/02/2023	1.0	Historia del proyecto terminada

	<div>Proceso de Software y Gestión 2</div>
---	--

Índice de contenido

1. Introducción	2
2. Objetivo	2
3. Contenido	3
3.1. Diagrama de commits	3
3.2. Resolución de conflictos	4
4. Conclusiones	4
5. Referencias	5


	<div>Proceso de Software y Gestión 2</div>
---	--

1. Introducción

Tras haber trabajado en las primeras tareas de implementación sin utilizar ninguna política de gestión de ramas y trabajando directamente desde main, hemos sacado las siguientes conclusiones y a continuación se detalla el proceso.

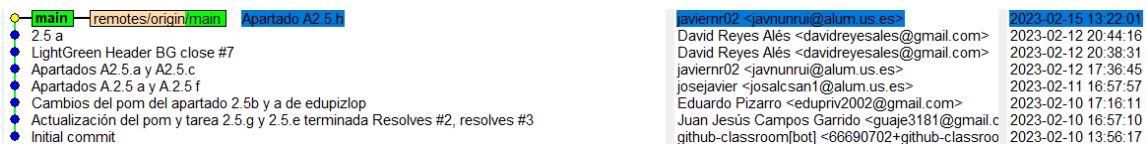
2. Objetivo

El objetivo de este documento es analizar los problemas que podemos tener al no utilizar una buena política de gestión de ramas para poder facilitar así el desarrollo concurrente y poder ser más eficientes.

	<div>Proceso de Software y Gestión 2</div>
---	--

3. Contenido

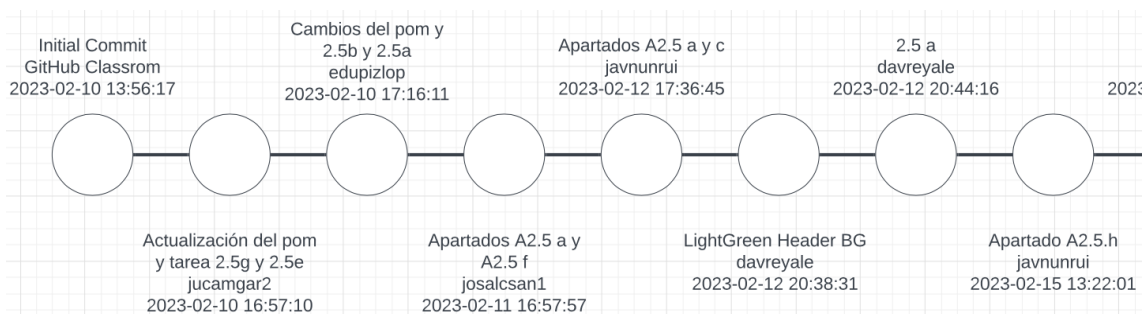
3.1. Diagrama de commits



Información generada mediante comando “gitk –all”. Este comando proporciona un diagrama en forma de árbol que representa el historial de commits del repositorio. Cada nodo del árbol representa un commit, ordenados de abajo hacia arriba, donde los que están más arriba son los que han sido realizados más recientemente. Por otro lado, las ramas del árbol representan las ramas del repositorio (de momento solo hay una rama main). De cada commit aparece información sobre el mensaje del commit, autor junto con su correo electrónico y fecha y hora.

La información que aparece en el diagrama corresponden a las tareas del apartado A2.5 del Product Backlog que consistían en implementar pequeños cambios al proyecto base Petclinic. El primer commit que se puede visualizar en el diagrama, el que está más abajo, corresponde al commit inicial que automáticamente se realiza al inicializar el repositorio y que fue realizado el 10/02/2023. Por otro lado, se muestran el resto de commits realizados por los integrantes del grupo, donde el apartado A2.5 a consistía en añadir la información personal al archivo pom.xml, apartado que fue realizado por todos. Además, aparece el apartado A2.5 b que consistía en incluir en el archivo pom.xml información sobre el grupo, el apartado A2.5 c que consistía en cambiar el color de los botones del encabezado a rojo cuando se seleccionan, el apartado A2.5 d cambiar el color del fondo del encabezado a verde claro, el apartado A2.5 e cambiar el mensaje en la página de Welcome, el apartado A2.5 f cambiar la imagen de la página Welcome con otra mascota, el apartado A2.5 g cambiar el color de fondo del encabezado de la tabla cuando se busque a propietarios a gris claro y el apartado A2.5 h cambiar la información incluida en el archivo info.yml para reflejar la información del grupo.

En este apartado nos obligaban a trabajar a todos en la misma rama (main), de ahí lo que se ha comentado anteriormente de que en el diagrama solo aparece una rama. Por ello, nos dimos cuenta que sí no queríamos tener conflictos deberíamos seguir un orden a la hora de subir los diferentes commits.



Por otro lado, realizamos el diagrama manualmente con la página Lucidchart donde representamos la misma información, pero de forma más visual.


3.2. Resolución de conflictos

Mientras desarrollamos las tareas correspondientes al apartado A2.5 del Product Backlog, al subirlas todas a la rama master, no encontramos ningún conflicto ya que por cuestiones de gestión temporal, cada uno trabajó en un momento del día en sus tareas, evitando tener que posteriormente solucionar conflictos.

Como hemos dicho, una vez asignamos las tareas durante la clase del viernes 10/02/2023, cada uno en su casa se organizó y cuando pudo, realizó su tarea de implementación lo que hizo que en ningún momento dos integrantes del grupo modificasen algún archivo al mismo tiempo y por eso no encontramos ningún conflicto.

4. Conclusiones

En resumen, al no haber modificado el mismo archivo al mismo tiempo por dos personas distintas no hemos encontrado conflictos, pero a pesar de no haber encontrado conflictos, pensamos que esta forma de gestionar el trabajo concurrente es muy ineficiente ya que no

	<div>Proceso de Software y Gestión 2</div>
---	--

permite que dos personas modifiquen el mismo archivo al mismo tiempo lo que implicaría que alguna persona tendría que esperar a que otra acabase para evitar problemas.

5. Referencias

- Product Backlog de la asignatura Proceso Software y Gestión II, Universidad de Sevilla.
- Lucidchart, Software de diagramas online, <https://www.lucidchart.com/pages/es>