UNIVERSIDAD DE BURGOS
ESCUELA POLITÉCNICA SUPERIOR
Grado en Ingeniería Informática

TFG del Grado en Ingeniería Informática

Evaluación de simulaciones de proyectos en GitHub orientadas a docencia. Documentación Técnica

Presentado por Licinio Fernández Maurelo
en Universidad de Burgos — 10 de junio
de 2023
Tutor: Dr. Carlos López Nozal

Índice general

Índice general i

Índice de figuras iii

Índice de tablas iv

Apéndice A Plan de Proyecto Software 1
A.1. Introducción 1
A.2. Planificación temporal 1
A.3. Estudio de viabilidad 6

Apéndice B Especificación de Requisitos 9
B.1. Introducción 9
B.2. Objetivos generales 9
B.3. Catalogo de requisitos 10
B.4. Especificación de requisitos 10

Apéndice C Especificación de diseño 13
C.1. Introducción 13
C.2. Diseño de modelo de dominio 13
C.3. Diseño procedimental 15
C.4. Diseño arquitectónico 17

Apéndice D Documentación técnica de programación 19
D.1. Introducción 19
D.2. Estructura de directorios 19
D.3. Manual del programador 19

D.4. Compilación, instalación y ejecución del proyecto 19
D.5. Pruebas del sistema 19

Apéndice E Documentación de usuario **21**
E.1. Introducción 21
E.2. Requisitos de usuarios 21
E.3. Instalación 21
E.4. Manual del usuario 21

Bibliografía **23**

Índice de figuras

A.1. Detalle parcial de sprints cerrados en este trabajo en GitHub . . 2
A.2. Tareas sprint 6 en GitHub 3
A.3. Java Developer Hourly Rate Guide 7
A.4. Inclusión de licencia MIT del trabajo en GitHub 8

B.1. Diagrama de casos de uso 10

C.1. Modelo de datos para la gestión de proyectos ágiles 14
C.2. Modelo de datos para el control de versiones 15
C.3. Diseño procedimental de la evaluación del laboratorio virtual de gestión de proyectos ágiles 16
C.4. Diseño procedimental de la evaluación del laboratorio virtual de gestión de proyectos ágiles 17
C.5. Arquitectura hexagonal de avrela 18

Índice de tablas

B.1. UC1 APM Case Study Simulation Evaluation. 11
B.2. UC2 SCM Case Study Simulation Evaluation. 12

Apéndice A

Plan de Proyecto Software

A.1. Introducción

En todo proyecto sofware es fundamental considerar los aspectos relactivos a:

- Planificación temporal
- Estudio de viabilidad ecónomica y legal

A.2. Planificación temporal

La planificación temporal es un componente vital de la gestión de proyectos, que contribuye a la finalización exitosa del mismo. La gestión temporal eficaz de un proyecto implica:

- Creación de un cronograma de proyecto realista
- Identificación de hitos clave
- Definición de entregables

En este trabajo se ha hecho uso de metodología scrum, dividiendo las tareas del proyecto en sprints de duración de dos semanas como norma general si bien en algunos casos estos sprints se han ampliado en duración.

El detalle de los sprints del proyecto puede consultarse en **GitHub**[1] como puede apreciarse en A.1:

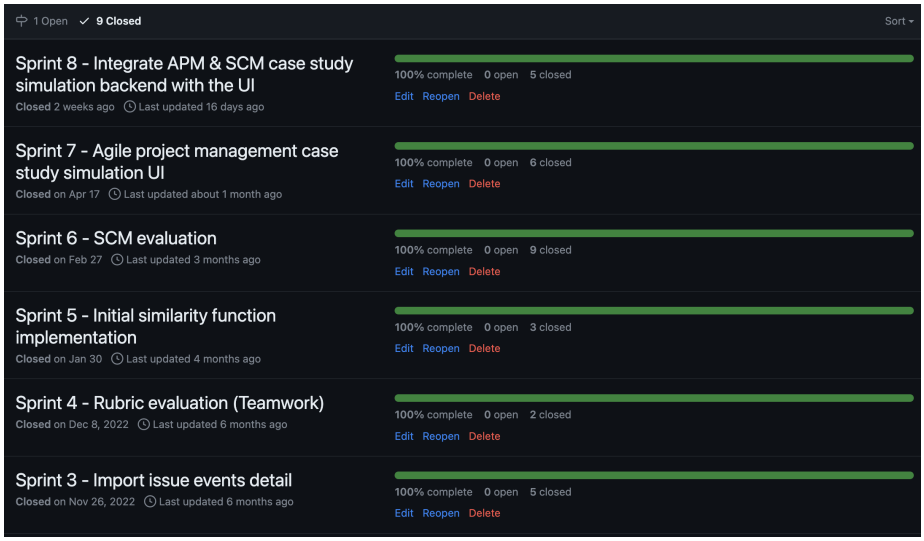

Figura A.1: Detalle parcial de sprints cerrados en este trabajo en GitHub

Cada sprint contiene las tareas que lo componen como puede apreciarse en la captura del sprint 6 A.2.

[1] *GitHub, sprints finalizados:* https://github.com/lfx1001/avrela/milestones?state=closed

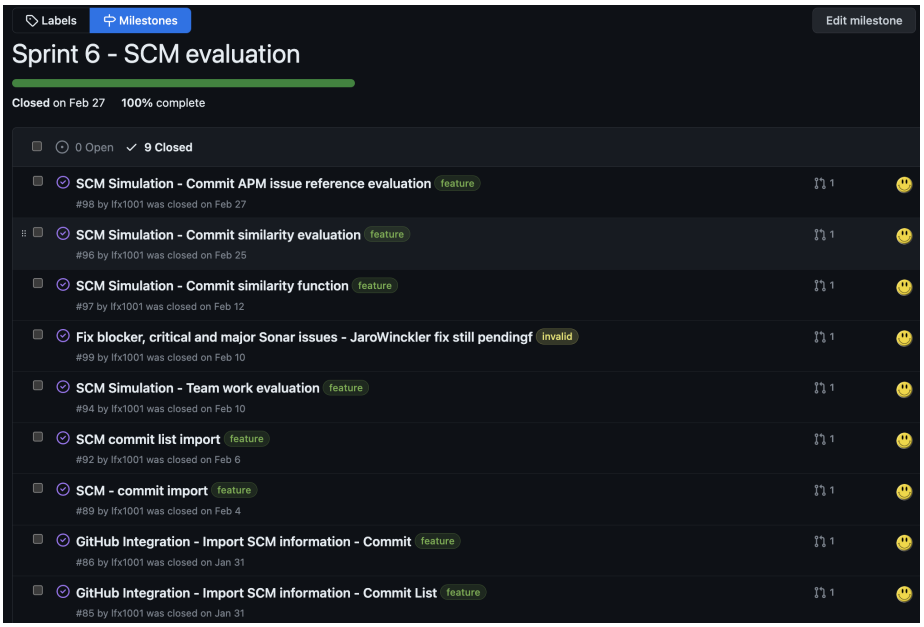

Figura A.2: Tareas sprint 6 en GitHub

Se enumeran a continuación los sprints que se han realizado en el proyecto junto con una descripción breve de los mismos.

Sprint 0 - Kick-off initial project documentation. En este primer sprint

- revisamos la temática de este trabajo y comenzamos la redacción de los apartados de introducción y objetivos.
- se creó el repositorio GitHub como soporte de los activos documentales.
- se realizaron unas primeras pruebas de trabajo con las plantillas LaTeXcorrespondiente a la memoria y anexos de este trabajo.

Sprint 1 - Initial domain model, GitHub integration . En este segundo sprint:

- se creó el proyecto Java asociado.
- se codificaron los primeros elementos del modelo.
- experimentamos con el API REST de GitHub y realizamos pruebas creando un cliente del API REST personalizado utilizando Open Feign.

- se configuró la integración continua para poder tener un feedback actualizado de métricas fundamentales en la calidad de software (cobertura de tests, duplicados, code smells).

Sprint 2 - Import agile project management data. En este sprint:

- se implementa la ingesta de información de la gestión ágil de proyectos.
- se incluyen tests funcionales utilizando Cucumber.
- se analiza la problemática de la integración de la información de los puntos de historia de usuario desde ZenHub.

Sprint 3 - Import issue events detail. En este sprint:

- se complementó la información relativa a la gestión de proyectos ágiles añadiendo e integrando la parte de eventos de las tareas.
- se llevaron a cabo tareas de documentación de algunas de las herramientas y técnicas utilizadas hasta el momento.

Sprint 4 - Rubric evaluation (Teamwork). En este sprint:

- se trabaja el aspecto de evaluación de las simulaciones.
- se implementa la evaluación de trabajo el equipo en la gestión de proyectos ágiles utilizando Cucumber.

Sprint 5 - Initial similarity function implementation. En este sprint:

- se implementa la función de semejanza para la comparación de tareas utilizando el índice de Jaccard.
- se evalúa el criterio de semejanza de descripción de tareas.

Sprint 6 - SCM evaluation. En este sprint:

- se implementa la importación de commits dado un intervalo de tiempo.

- se implementa la función de semejanza de commits. Se utiliza la distancia de Jaro-Wincker junto con el índice de Jaccard.
- se evalúa el criterio de trazabilidad de la simulación de control de versiones con la herramienta de gestión de proyectos.
- se evalúa el criterio de trabajo en equipo de la simulación de control de versiones.
- se evalúa el criterio de semejanza de la simulación de control de versiones.

Sprint 7 - Agile project management case study simulation UI. En este sprint:

- se realizan pruebas de implementación de interfaces de usuario con Thymeleaf.
- se implementa una versión inicial de pantallas para la evaluación de proyectos ágiles.

Sprint 8 - Integrate APM SCM case study simulation backend with the UI. En este sprint:

- se implementa una versión inicial de pantallas para la evaluación de control de versiones.
- se integran las pantallas de evaluación de gestión ágil de proyectos con la lógica de negocio.
- se integran las pantallas de evaluación de control de versiones con la lógica de negocio.

Sprint 9 - Documentation Web application improvements. En este sprint:

- se corrigen incidencias de las evaluación.
- se realizan mejoras de la interfaz visual de la evaluación de simulaciones de control de versiones.
- se trabaja en la documentación de anexos y memoría.

Sprint 10 - Documentation Web application improvements. En este sprint:

- se corrigen incidencias de las evaluación.
- se realizan mejoras de la interfaz visual de la evaluación de simulaciones de gestión de proyectos ágiles.
- se trabaja en la documentación de anexos y memoría.

A.3. Estudio de viabilidad

Viabilidad económica

Los costes de este proyecto software se basan en las siguientes líneas:

- Coste de desarrollo.
- Coste de recursos asociados.

En este caso vamos a optar por un cálculo simplificado que incluya los propios recursos asociados en el coste de desarrollo bajo el supuesto de que este proyecto se gestionase "llave en mano". De esta forma, el cliente (en esta caso, la universidad) encargaría a un tercero (el alumno) la realización del proyecto como un proveedor externo.

Este proveedor externo gestionaría internamente los gastos correspondientes recursos humanos y materiales. Proporcionaría al cliente una cifra final de costes. Este coste está determinado por la expresión:

$$\text{Coste} = \text{Jornadas} \times \text{Tarifa jornada}$$

Jornadas representa el número de jornadas necesarias para acometer el proyecto. Se estima en 40 jornadas (320 horas).

Tarifa jornada representa el coste unitario de la jornada. Este coste está relacionado con el perfil del recurso que vaya a acometer el proyecto. Para el cálculo se considera un perfil de desarrollador Java Senior. La tarifa presenta una gran variabilidad en función de la localización. Aprovecharemos el estudio realizado por **flexiple**[2] cuya captura se recoge en A.3. Este estudio estima en 55,43 Euros (60 Dólares) la tarifa hora, lo que supone un coste de 443,7 Euros por jornada.

[2] *https://flexiple.com/java/hourly-rate/*

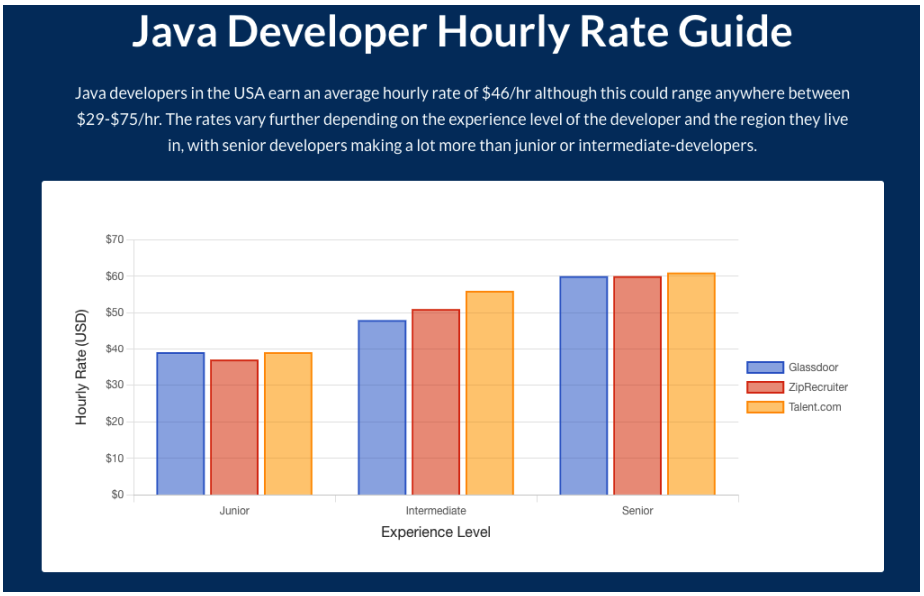

Figura A.3: Java Developer Hourly Rate Guide

En base a los supuestos anteriores, una estimación posible del coste del proyecto, impuestos no incluidos, es:

$$40 \text{ Jornadas} \times 443{,}7 \text{ Euros Tarifa jornada} = 17.748 \text{ Euros}$$

Viabilidad legal

La viabilidad legal abarca el cumplimiento de las leyes, reglamentos y derechos de propiedad intelectual pertinentes. Implica consideraciones tales como licencias, privacidad de datos, seguridad y cualquier obligación legal asociada con el proyecto de software. Al abordar de manera proactiva los aspectos legales, mitigamos los riesgos potenciales y nos aseguramos de que el proyecto opere dentro de los límites de las leyes aplicables.

Se listan a continuación las librerías y agrupaciones de librerías (indicadas con la terminación .*) con su licencia correspondiente:

- **org.springframework.boot.* :** licencia Apache-2.0
- **org.springframework.cloud.* :** licencia Apache-2.0
- **org.webjars:bootstrap :** licencia MIT

Las licencias anteriores nos permitirían el uso de las mismas con fines comerciales. Este trabajo se distribuye bajo licencia **MIT**[3]. Para facilitar conocer la licencia de este trabajo se ha añadido la misma en GitHub A.4.

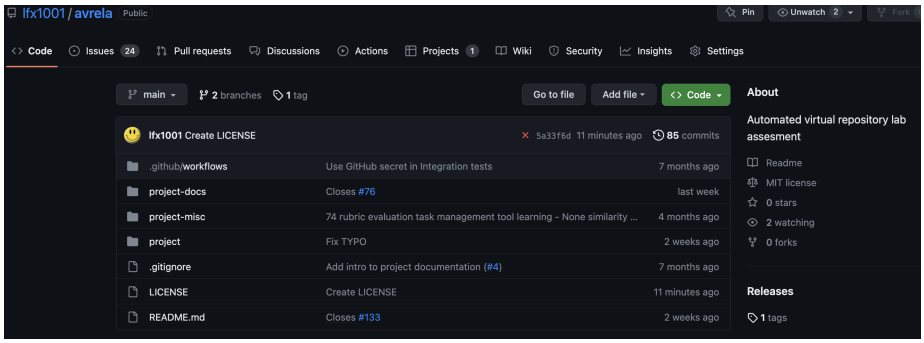

Figura A.4: Inclusión de licencia MIT del trabajo en GitHub

[3] *Licencia MIT:* https://choosealicense.com/licenses/mit/

Apéndice B

Especificación de Requisitos

B.1. Introducción

En este apéndice se detalla lo que software debe hacer y cómo debe comportarse con un doble objetivo:

- aclarar las expectativas y necesidades de los usuarios o partes interesada.
- servir como referencia para el equipo de desarrollo.

B.2. Objetivos generales

Se enumeran a continuación los objetivos generales del proyecto:

- Aplicación de rúbrica de evaluación sobre:
  - La información de gestión de tareas y del control de versiones de un repositorio.
  - La comparación de la información de gestión de tareas y de control de versiones de dos repositorios, el original y el resultante de la simulación.
- Proporcionar retroalimentación de la comparación de la información de gestión de tareas y de control de versiones de dos repositorios.

B.3. Catalogo de requisitos

La lista de requisitos funcionales es:

- REQ1: Evaluación del laboratorio virtual de gestión de proyectos ágil y retroalimentación.
- REQ2: Evaluación del laboratorio virtual de control de versiones y retroalimentación.

B.4. Especificación de requisitos

A partir de los requisitos funcionales se plantea el diagrama de casos de uso de la Figura B.1:

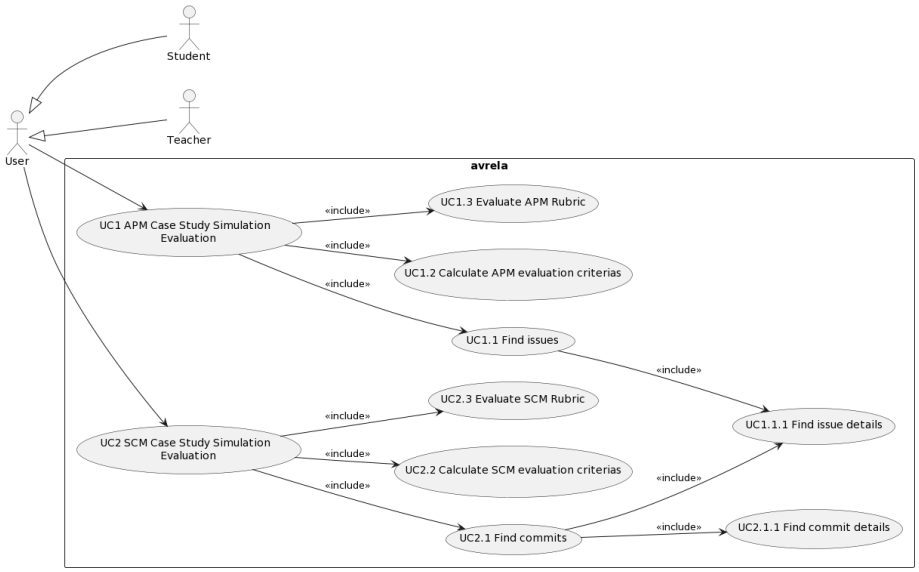

Figura B.1: Diagrama de casos de uso

UC1	**APM Case Study Simulation Evaluation**
Versión	1.0
Autor	Alumno
Requisitos asociados	REQ1
Descripción	Evaluación y retroalimentación del laboratorio virtual de gestión de proyectos ágil
Precondiciones	1. Existe el caso de estudio 2. Existe la simulación
Acciones	1. Recuperar datos del caso de estudio 2. Recuperar datos de la simulación 3. Calcular criterios de evaluación 4. Aplicar rúbrica 5. Mostrar información recuperada del caso de estudio y de la simulación 6. Mostrar rúbrica y nota final 7. Mostrar retroalimentación
Postcondición	
Excepciones	
Importancia	Alta

Tabla B.1: UC1 APM Case Study Simulation Evaluation.

UC2	**SCM Case Study Simulation Evaluation**
Versión	1.0
Autor	Alumno
Requisitos asociados	REQ2
Descripción	Evaluación y retroalimentación del laboratorio virtual de gestión de control de versiones
Precondiciones	1. Existe el caso de estudio 2. Existe la simulación
Acciones	1. Recuperar datos del caso de estudio 2. Recuperar datos de la simulación 3. Calcular criterios de evaluación 4. Aplicar rúbrica 5. Mostrar información recuperada del caso de estudio y de la simulación 6. Mostrar rúbrica y nota final 7. Mostrar retroalimentación
Postcondición	
Excepciones	
Importancia	Alta

Tabla B.2: UC2 SCM Case Study Simulation Evaluation.

Apéndice C

Especificación de diseño

C.1. Introducción

C.2. Diseño de modelo de dominio

En este proyecto se trabaja con varios dominios:

- Gestión de proyectos ágiles.
- Control de versiones.

El dominio de la gestión de proyectos ágiles se refleja en el diagrama de clases de la Figura C.1.

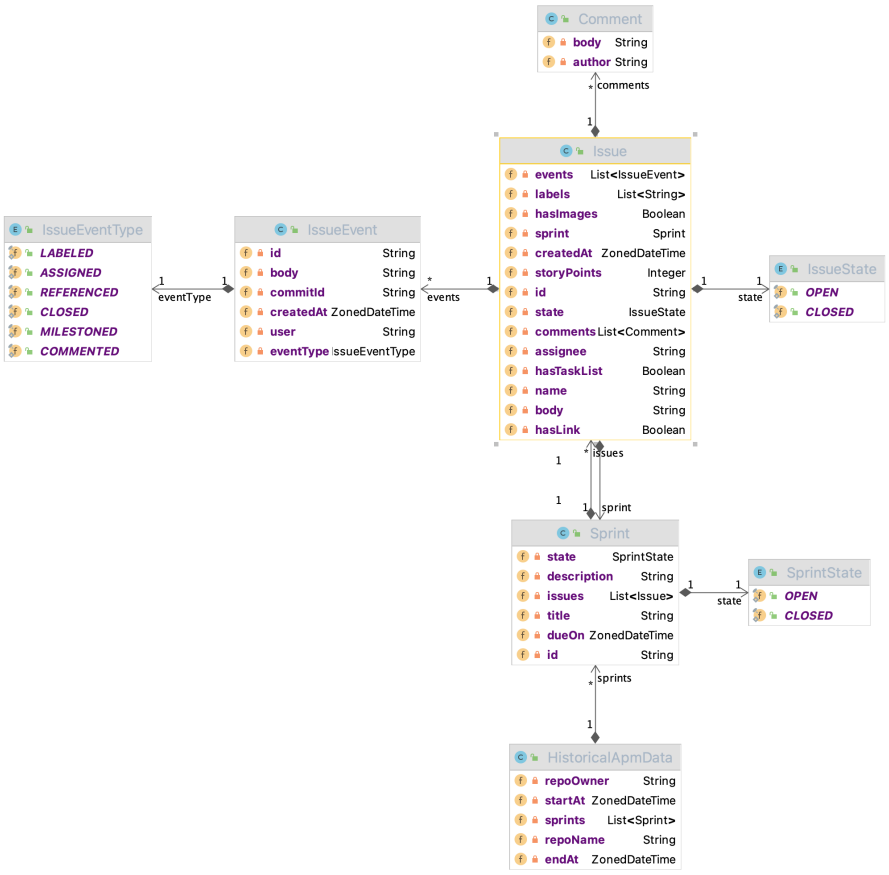

Figura C.1: Modelo de datos para la gestión de proyectos ágiles

El modelo de dominio control de versiones obedece al diagrama de clases de la Figura C.2.

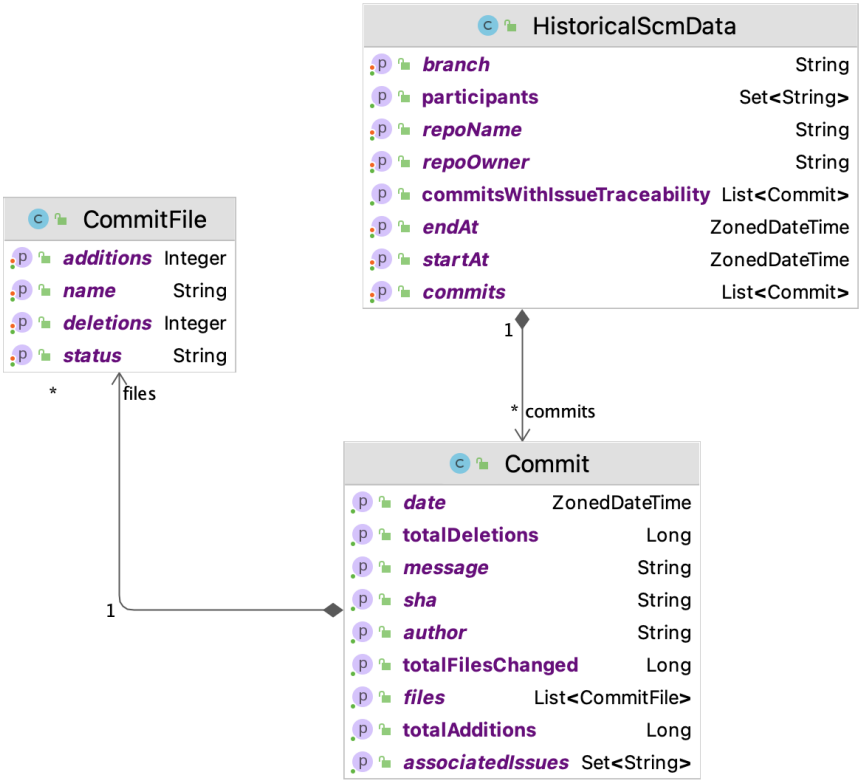

Figura C.2: Modelo de datos para el control de versiones

C.3. Diseño procedimental

Para modelar este diseño se utilizarán diagramas UML de secuencia. En ellos se han creado grupos para facilitar la identificación de la correspondencia entre los casos de uso y el diseño procedimental.

El diseño procedimental de la evaluación del laboratorio virtual de gestión de proyectos ágiles corresponde al diagrama C.3:

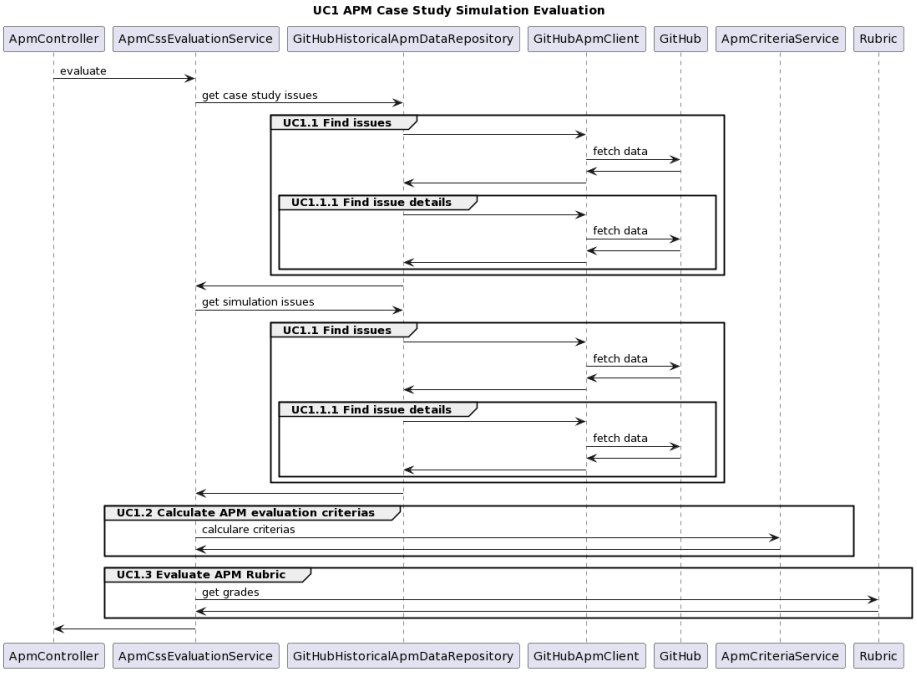

Figura C.3: Diseño procedimental de la evaluación del laboratorio virtual de gestión de proyectos ágiles

El diseño procedimental de la evaluación del laboratorio virtual de control de versiones corresponde al diagrama C.4:

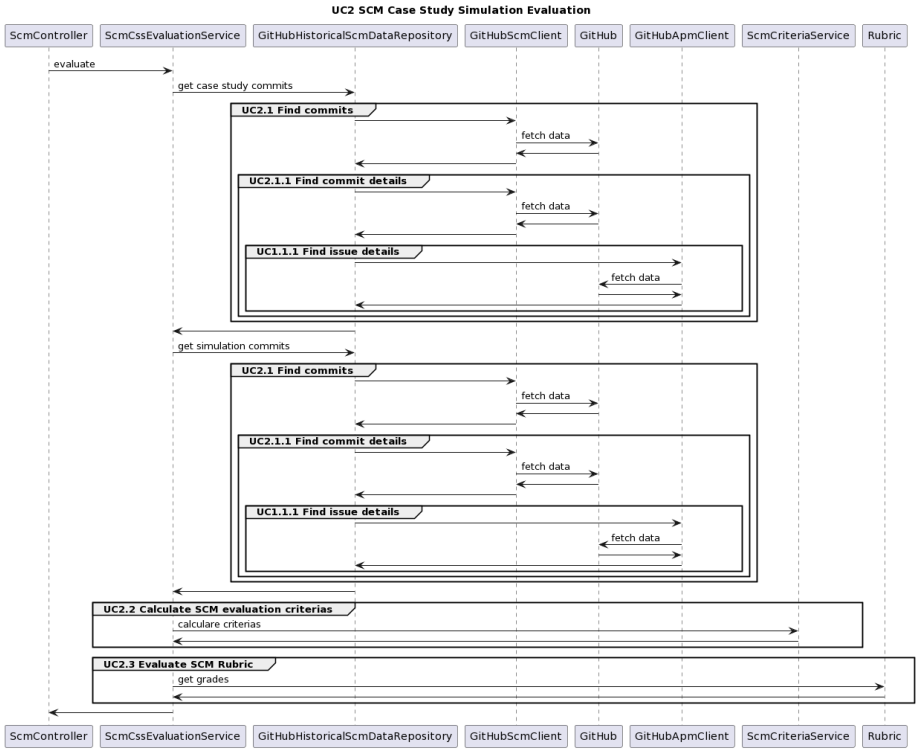

Figura C.4: Diseño procedimental de la evaluación del laboratorio virtual de gestión de proyectos ágiles

C.4. Diseño arquitectónico

La aplicación se ha desarrollado siguiendo una arquitectura hexagonal C.5:

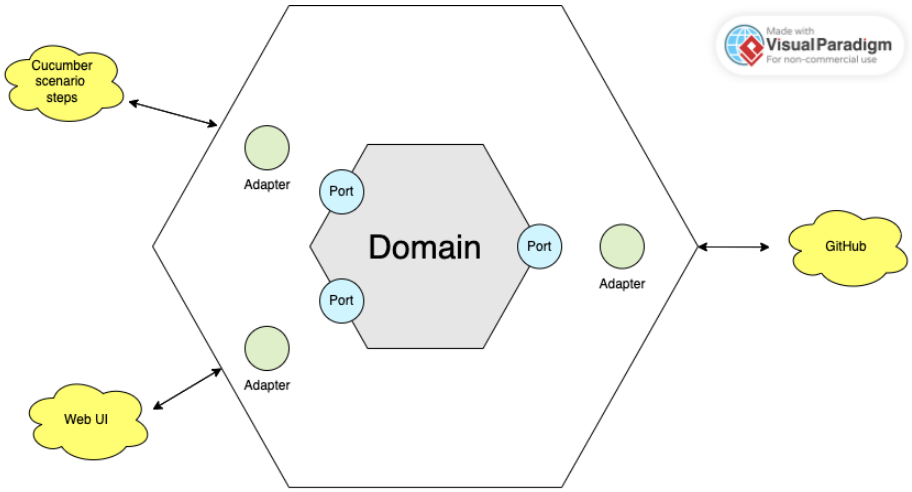

Figura C.5: Arquitectura hexagonal de avrela

Apéndice D

Documentación técnica de programación

D.1. Introducción

D.2. Estructura de directorios

D.3. Manual del programador

D.4. Compilación, instalación y ejecución del proyecto

D.5. Pruebas del sistema

Apéndice E

Documentación de usuario

E.1. Introducción

E.2. Requisitos de usuarios

E.3. Instalación

E.4. Manual del usuario

Bibliografía