Trabajo final IS

Escuela Politécnica Superior
Ingeniería del Software
GRUPO 37.

23 de diciembre del 2018

Agenda Alumnos

Índice

1. Visión general (descripción del problema)

El profesorado de la asignatura de Ingeniería del Software quiere informatizar los datos de contacto de los alumnos en un programa informático que guarde esta información. Además el profesorado solicita un tipo de usuario que pueda administrar a los alumnos y, además tenga el privilegio de crear copias de seguridad. Se accederá a la aplicación con un determinado usuario y contraseña que tendrá permisos de administrador o profesor.

2. Extracción de requisitos

Partes interesadas del Proyecto

1. Profesor
2. Alumnos

Datos que se van a almacenar en la aplicación

1. DNI
2. Username (profesor)
3. Password (profesor)
4. Nombre del alumno
5. Apellidos del alumno
6. Teléfono
7. Fecha de nacimiento
8. Dirección
9. Email de la UCO
10. Curso más alto en el que el alumno esté matriculado
11. Grupo al que pertenece dentro de la Asignatura
12. Id indicando si el alumno es líder o no de un grupo
13. Id indicando si el profesor es coordinador o ayudante

Requisitos Funcionales

- Login Profesor. Prioridad: 0
- Cargar datos de un fichero. Prioridad: 1
- Añadir alumno. Prioridad: 1
- Guardar datos en fichero. Prioridad: 2
- Mostrar 1 alumno. Prioridad: 3
- Mostrar todos los alumnos. Prioridad: 3
- Mostrar un grupo de alumnos. Prioridad: 3
- Modificar alumno. Prioridad: 4
- Borrar alumno. Prioridad: 5
- Crear copia de seguridad. Prioridad: 6
- Cargar copia de seguridad. Prioridad: 6

Requisitos No Funcionales

- Lenguaje de Programación será C++
- Funcional obligatoriamente en GNU/Linux
- Número máximo de Alumnos = 150
- Lenguaje de documentación será Markdown
- La visualización de los datos será obligatoriamente por línea de comandos y HTML o Markdown a elegir uno de los dos
- El formato de los ficheros de los datos será binario

3. Historias de usuario

HU-00 Login

“Como profesor quiero poder acceder al sistema.”

Prioridad: 0

OBJETIVO:

- Se debe comprobar que los datos introducidos por el profesor son correctos.

HU-01 Cargar datos de un fichero

“Como profesor quiero cargar unos alumnos guardados anteriormente.”

Prioridad: 1

OBJETIVO:

- Quiero poder cargar un fichero
- Debe de existir el fichero para cargar

HU-02 Añadir alumno

Como profesor quiero poder añadir un alumno en mi curso

Prioridad: 1

OBJETIVO:

- Quiero poder insertar un alumno en la base de datos
- No puede insertarse ningún alumno repetido
- Todos los campos son obligatorios excepto el grupo al que pertenece y si el alumno es líder o no

HU-03 Guardar datos en fichero

Como profesor quiero poder guardar los cambios que he realizado

Prioridad: 2

OBJETIVO:

- Quiero poder guardar las modificaciones que he realizado
- Debe de haber modificaciones previas en la base de datos
- Si no existe el fichero , se crea

HU-04 Mostrar un alumno

“Como profesor quiero poder mostrar un alumno por pantalla.”

Prioridad: 3

OBJETIVO:

- Necesito visualizar todos los datos de un alumno.
- Para poder visualizar un alumno necesito buscarlo por apellidos o por DNI.
- En caso de que haya dos alumnos con mismos apellidos, se dará la opción de buscar por DNI.

HU-05 Mostrar Todos los alumnos

“Como profesor quiero poder mostrar todo el conjunto de alumnos”

Prioridad: 3

OBJETIVO:

- Necesito poder mostrar todos los datos de todos los alumnos de la clase por pantalla.

- A la hora de visualizar el conjunto de alumnos podré ordenarlos alfabéticamente por apellidos y nombre, por dni, por curso más alto en el que esté matriculado y por grupo al que pertenecen; tanto ascendente como descendentemente.
- Se deberá identificar de alguna manera al Líder del grupo.

HU-06 Mostrar Alumnos de un grupo concreto

"Como profesor quiero poder mostrar por pantalla los alumnos que estén en cierto grupo de clase."

Prioridad: 3

OBJETIVO:

- Necesito poder visualizar los alumnos que están en un grupo concreto de la clase de prácticas
- Tiene que existir el grupo indicado.
- No tiene número máximo de alumnos, pero deberá tener un mínimo de un alumno.
- En caso de que el grupo tenga líder, se deberá remarcar de alguna manera dicho alumno.

HU-07 Modificar Alumno

"Como profesor quiero poder modificar los datos de un alumno."

Prioridad: 4

OBJETIVO:

- Se debe identificar al alumno a modificar por su Dni o por su apellido.
- En caso de haber dos alumnos con el mismo apellido de solicitará el Dni del alumno.
- Poder elegir los datos a modificar de un alumno.

HU-08 Borrar Alumno

“Como profesor quiero poder eliminar un alumno.”

Prioridad: 5

OBJETIVO:

- Se debe solicitar el alumno que se desea eliminar a través de sus apellidos o Dni.
- Se debe comprobar que el alumno a eliminar existe en la base de datos de los alumnos.
- Se eliminan todos los datos del alumno seleccionado.

HU-09 Crear Copia Seguridad

“Como profesor administrador quiero poder crear una copia de seguridad de mi fichero de alumnos.”

Prioridad: 6

OBJETIVO:

- Se debe comprobar que el profesor tiene permisos de administrador.
- Se debe solicitar el nombre del fichero del cual se va a realizar la copia de seguridad.

HU-10 Cargar Copia Seguridad

“Como profesor administrador quiero poder cargar los datos de alumnos de una copia de seguridad existente.”

Prioridad: 6

OBJETIVO:

- Se debe comprobar que el profesor tiene permisos de administrador.
- Se debe solicitar el nombre del fichero que tenga la copia de seguridad.

4. Casos de uso

CU-00 Login

Breve descripción: El sistema solicita un login para acceder al programa

Actores principales: Profesor

Precondiciones:

- Debe existir en el sistema un fichero con la información de los profesores que se loguean.

Flujo principal:

1. El caso de uso empieza cuando el Profesor decide acceder al sistema.
2. Se comprueba que los datos de login sean correctos.

Postcondiciones:

- Se muestra un Menú correspondiente con los permisos que tenga el profesor que acaba de acceder al sistema.

Flujo alternativo:

2.a) En caso de que los datos del profesor no coincidan, se da un mensaje de error con la salida del sistema.

CU-01 Cargar datos de un fichero

Breve descripción: El sistema permite cargar datos de un fichero ya creado

Actores principales: Profesor Actores secundarios: Alumno

Precondiciones:

- Deben existir un fichero binario para poder cargarlo

Flujo principal:

1. El caso de uso empieza cuando el profesor decide cargar unos datos que ya tiene en el sistema.
2. Se indica el archivo que se quiera cargar
3. El sistema cargar el archivo

Postcondiciones:

- El sistema debe de haber cargado el fichero correctamente

Flujo alternativo:

2.a) En el caso de que no exista el archivo se muestra un mensaje de error

3.a) En el caso de que no se haya abierto correctamente el archivo muestra un mensaje de error.

CU-02 Añadir Alumno

Breve descripción: El sistema permite añadir un alumno.

Actores principales: Profesor Actores secundarios: Alumno.

Precondiciones:

- No debe existir un alumno con el mismo DNI que se quiera insertar
- Debe introducir los campos obligatorios pedidos por el sistema

Flujo principal:

1. El caso de uso empieza cuando el profesor desea añadir un alumno al sistema
2. Busca alumno en la base de datos
3. Inserta un alumno a los datos cargados actualmente

Postcondiciones:

- Se comprobará que se haya insertado correctamente el alumno

- Se comprobará que estén todos los datos obligatorios rellenos

Flujo alternativo:

2.a) Si encuentra un alumno con el mismo DNI , muestra un mensaje de error.

2.b) Pide de nuevo el DNI del alumno.

2.c) Si el número de alumnos en nuestra base de datos es de 150 , muestra un mensaje de error.

2.d) Si encuentra un alumno con el mismo email , muestra un mensaje de error

2.e) Pide de nuevo el email del alumno.

3.a) Si no se ha insertado correctamente el alumno en los datos , muestra un mensaje de error.

CU-03 Guardar los datos en fichero

Breve descripción: El sistema permite guardar los cambios que ha realizado

Actores principales: Profesor Actores secundarios: Alumno

Precondiciones:

- Debe de haber cambios en los datos

Flujo principal:

1. El caso de uso empieza cuando el profesor desea gurdar los datos que ha modificado
2. Se indica el archivo donde queramos guardar los datos
3. Se guardan los nuevos cambios

Postcondiciones:

- Se comprobará que se haya guardado correctamente el fichero binario

Flujo alternativo:

2.a) En caso de que no se indique un archivo correctamente , muestra un mensaje de error.

2.b) Si existe otro fichero con el mismo nombre , se sobreescriben los datos.

3.a) Si no se ha guardado correctamente , muestra un mensaje de error.

CU-04 Mostrar UN alumno

Breve descripción: El sistema permite mostrar todos los datos de un alumno en concreto.

Actores principales: Profesor Actores secundarios: Alumno

Precondiciones:

- El alumno debe existir en el sistema

Flujo principal:

1. El caso de uso empieza cuando el profesor decide ver los datos de un alumno concreto de manera visual.
2. Se solicitará por pantalla a la hora de buscar, el DNI o los apellidos del alumno.
3. En caso de que el alumno exista se mostrarán por pantalla los datos de dicho alumno.

Postcondiciones:

- Se comprobará que se hayan mostrado todos los datos del alumno

Flujo alternativo:

2.a) En caso de que el alumno no exista en la base de datos se dará un aviso por pantalla.

2.b) En caso de que el profesor quiera buscar al alumno por sus apellidos y exista otro alumno con los mismos, se solicitará el DNI del alumno que quiere visualizar.

CU-05 Mostrar TODOS los alumnos

Breve descripción: El sistema permite mostrar todos los datos de todos los alumnos.

Actores principales: Profesor Actores secundarios: Alumno

Precondiciones:

- Deben existir al menos un alumno en la base de datos.

Flujo principal:

1. El caso de uso empieza cuando el profesor decide ver los datos de todos los alumnos de manera visual.
2. El sistema recoge los datos de los alumnos.
3. A la hora de visualizar todos los alumnos, se preguntará si se desea ver en un fichero HTML o directamente por pantalla.
4. En caso de que solicite verlo en pantalla se imprimirá directamente.

Postcondiciones:

- Se mostrarán todos los datos de los alumnos
- Se mostrará ordenadamente de la manera indicada por el profesor
- Se mostrará al líder del grupo de una manera especial

Flujo alternativo:

2.a) En caso de que no exista la base de datos se mostrará un mensaje de error por pantalla.

2.b) En caso de que no exista nadie en la base de datos de igual modo se enviará por pantalla un mensaje de error.

CU-06 Mostrar los alumnos de un grupo

Breve descripción: El sistema permite mostrar todos los datos de los alumnos que pertenecen a cierto grupo.

Actores principales: Profesor Actores secundarios: Alumno

Precondiciones:

- Deben existir al menos un alumno dentro de ese grupo en la base de datos.

Flujo principal:

1. El caso de uso empieza cuando el profesor decide ver los datos de todos los alumnos que pertenecen a cierto grupo de manera visual.
2. El sistema busca a los alumnos que pertenecen a ese grupo
3. El sistema recoge los datos de los alumnos de dicho grupo
4. Se deberá marcar de alguna manera al líder del grupo.

Postcondiciones:

- Se mostrarán todos los datos de los alumnos
- Se mostrarán los alumnos ordenados de la manera indicada por el profesor
- Se mostrará al líder del grupo de una manera especial

Flujo alternativo:

2.a) En caso de que no exista la base de datos se mostrará un mensaje de error por pantalla.

2.b) En caso de que no exista nadie en la base de datos de igual modo se enviará por pantalla un mensaje de error.

CU-07 Modificar Alumno

Breve descripción: El sistema permite modificar los datos de un alumno.

Actores principales: Profesor Actores secundarios: Alumno

Precondiciones:

- El alumno debe existir en el sistema.

Flujo principal:

1. El caso de uso empieza cuando el Profesor decide modificar algún dato de un alumno.
2. Se solicitará la identificación de dicho alumno tanto por su Dni como por sus apellidos.
3. Una vez encontrado el alumno se deben mostrar las opciones de los datos que se van a modificar.
4. Se deberá tener en cuenta el atributo líder.

Postcondiciones:

- Se deben visualizar los nuevos datos del alumno.
- No existe un grupo con más de un líder.

Flujo alternativo:

2.a) En caso de identificar al alumno por sus apellidos y coincidir con otro alumno del sistema, se solicitará el Dni para diferenciarlos.

2.b) En caso de no existir ese alumno en el sistema se debe dar un mensaje de error.

3.a) En caso de que el grupo del alumno ya tenga un líder, muestra un mensaje de error y no lo asigna como líder.

3.b) En caso de existir un líder en el grupo, se pone al alumno modificado como no líder y el sistema debe dar un mensaje de error.

CU-08 Borrar Alumno

Breve descripción: El sistema permite borrar al completo los datos de un alumno.

Actores principales: Profesor Actores secundarios: Alumno

Precondiciones:

- El alumno debe existir en el sistema.

Flujo principal:

1. El caso de uso comienza cuando el Profesor decide eliminar a un alumno.
2. Se solicitará la identificación de dicho alumno tanto por su Dni como por sus apellidos.
3. Se eliminan todos los datos de dicho alumno.

Postcondiciones:

- El alumno no debe existir en el sistema.

Flujo alternativo:

2.a) En caso de identificar al alumno por sus apellidos y coincidir con otro alumno del sistema, se solicitará el Dni para diferenciarlos.

2.b) En caso de no existir el alumno en el sistema se informa con un mensaje de error.

CU-09 Crear Copia de Seguridad

Breve descripción: El sistema permite crear una copia de seguridad del fichero almacenado

Actores principales: Profesor Administrador

Precondiciones:

- Debe existir en el sistema un fichero con la información de los alumnos.

Flujo principal:

1. El caso de uso empieza cuando el Profesor Administrador decide crear una copia de seguridad del fichero de alumnos almacenados hasta el momento.

2. Se comprueba que el profesor tiene los permisos para realizar copia de seguridad (ser coordinador).
3. Se solicita el nombre del fichero al cual se desea hacer copia de seguridad.

Postcondiciones:

- Se habrá creado un fichero con el nombre <backup_nombreFichero> con la información del fichero en cuestión.

Flujo alternativo:

2.a) En caso de que el profesor no tenga permisos correspondientes, el sistema dará un mensaje de error.

3.a) En caso de que el fichero indicado no exista, el sistema dará un mensaje de error.

CU-10 Cargar Copia de Seguridad

Breve descripción: El sistema permite cargar una copia de seguridad existente.

Actores principales: Profesor Administrador

Precondiciones:

- Debe existir en el sistema el fichero que contenga la copia de seguridad.

Flujo principal:

1. El caso de uso empieza cuando el Profesor Administrador decide cargar una copia de seguridad que contiene la información de los alumnos.
2. Se comprueba que el profesor tiene los permisos para cargar la copia de seguridad en el sistema (ser administrador).
3. Se solicita el nombre del fichero copia de seguridad.

Postcondiciones:

- Se habrán cargado en el sistema los datos de los alumnos pertenecientes a la copia de seguridad en cuestión.

Flujo alternativo:

2.a) En caso de que el profesor no tenga permisos correspondientes, el sistema dará un mensaje de error.

3.a) En caso de que el fichero indicado no exista, el sistema dará un mensaje de error.

5. Diagrama de clases

El diagrama de clases propuesto para la resolución del problema se puede dividir en 3 clases diferentes:

1. Clase **Agenda**, la cual representa el objeto de una lista de alumnos. Donde crearemos todas las funciones de gestión de alumnos. Usaremos un vector de alumnos para guardar los datos de la agenda.
2. Clase **Alumno**, la cual representa un alumno de la vida real. Clase que usaremos para almacenar los datos de cada alumno (atributos) y para cada uno de ellos sus getters y setters.
3. Clase **Profesor**, la cual representa un profesor que gestiona la lista de alumnos e interactúa con la Agenda. Cada profesor podrá loguearse en el sistema y dependiendo de su rol podrá crear/cargar copias de seguridad o no.

Como se muestra en la siguiente imagen:

- La clase Alumno se relaciona con la clase Agenda (cardinalidad 1:N)y con la clase Grupo (cardinalidad 0:N).
- La clase Profesor se relaciona con la clase Agenda con cardinalidad 1:N.

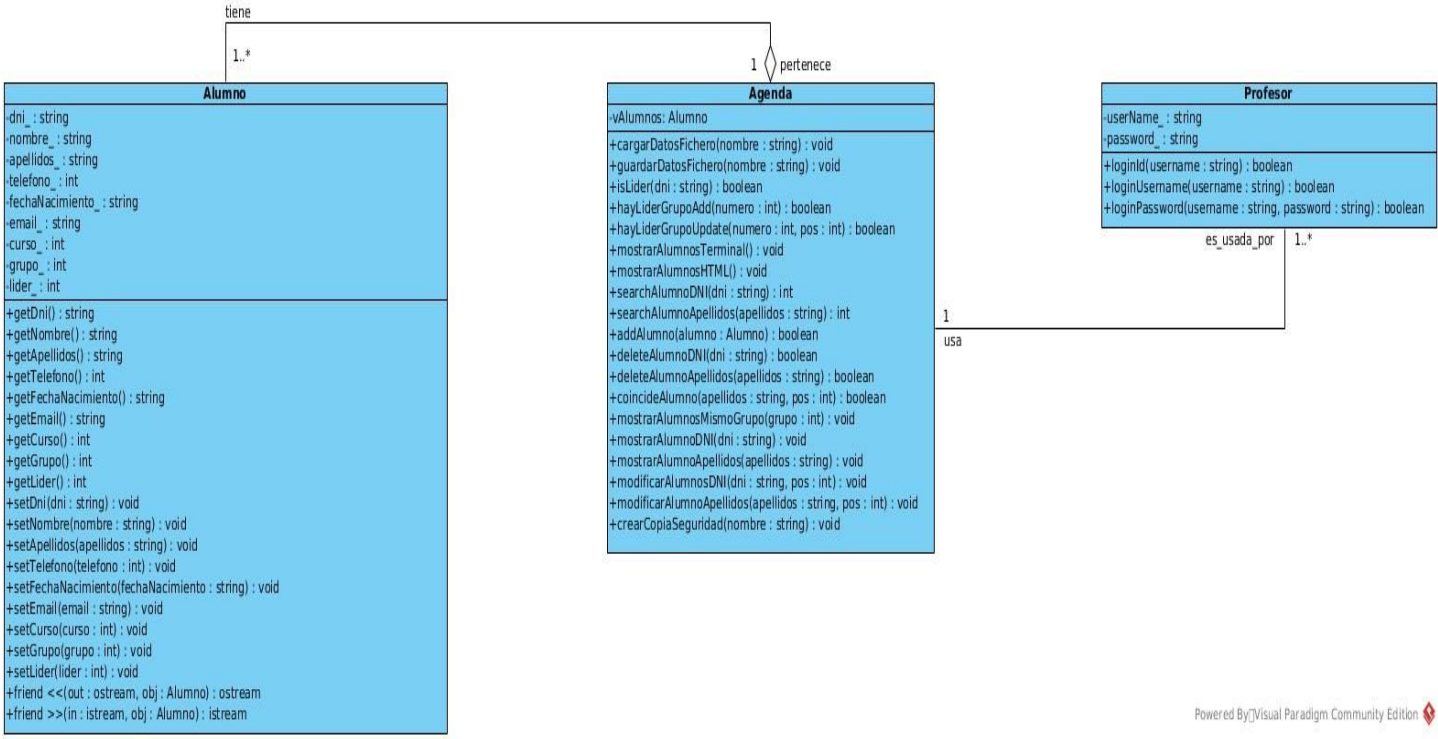

Figura 5.1: Diagrama de Clases

(*https://github.com/i62cogag/practica_IS/blob/master/practica3/diagramaClase.jpg*)

6. Diagrama de secuencia

DS-00 Login

El diagrama comienza justo al iniciar el sistema , cuando nos pide introducir el usuario para poder identificarnos como profesor ayudante o profesor coordinador. Se le manda un mensaje al sistema con el nombre de usuario , después el sistema comprueba en el fichero de profesores que el usuario existe. El sistema solicita una contraseña y el profesor al introducirla , se comprueba que sea correcta. El sistema muestra el menú y las opciones de hacer copia de seguridad en caso de que sea coordinador , y solo el

menu en caso de que sea ayudante. Si se introducen mal los datos de usuario , se muestra un mensaje de error.

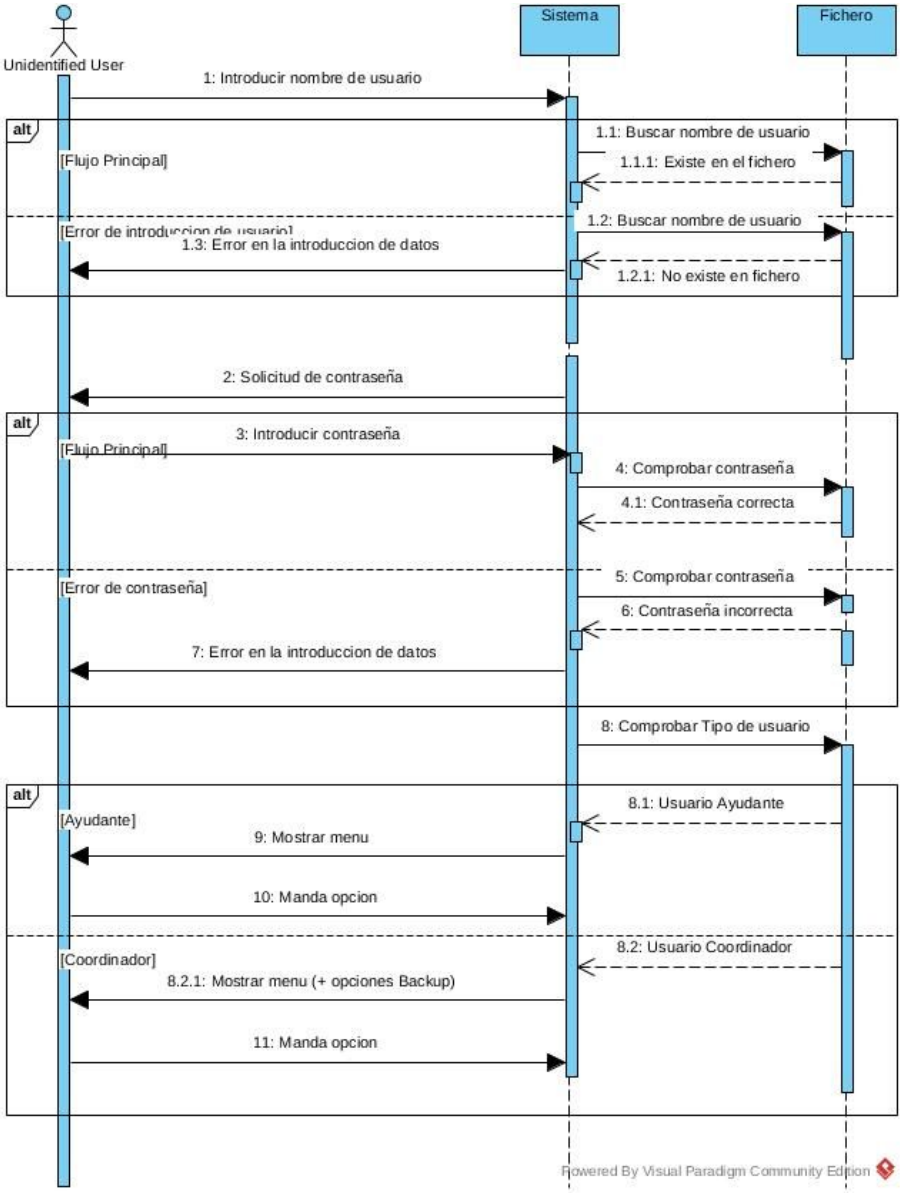

Figura 6.0 Diagrama de Secuencia del Login de usuarios

DS-01 Cargar Fichero

El diagrama comienza cuando el usuario (bien coordinador o bien ayudante) solicita la opción de Cargar Fichero de alumnos en el sistema. El sistema solicita el nombre del fichero de alumnos a cargar. Si el fichero existe, los alumnos son cargados en el sistema. En caso de no existir el fichero indicado se informa al usuario con un mensaje de error.

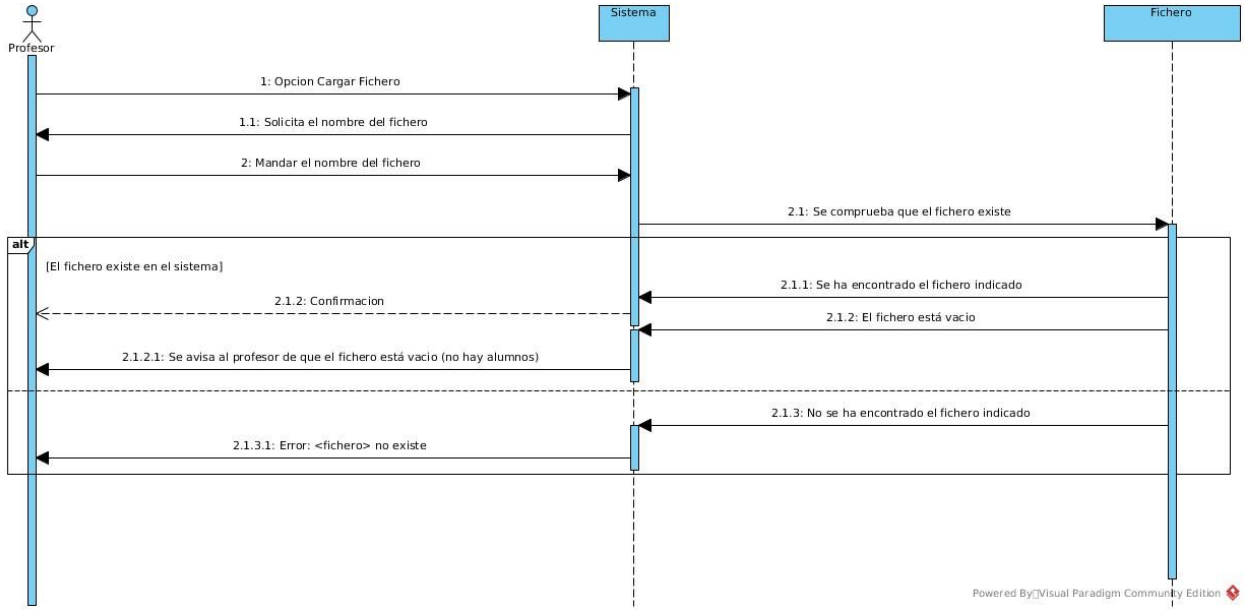

Figura 6.1: Diagrama de Secuencia-Cargar Fichero

DS-02 Añadir Alumno

El diagrama comienza cuando el usuario (bien coordinador o bien ayudante) solicita la opcion de Añadir un nuevo alumno al sistema. El sistema comprueba que hay menos de 150 alumnos en ese momento, ya que es el máximo permitido. En caso de haber 150, se informa con un mensaje de error al usuario. En caso de que cupiese un nuevo alumno, se solicitan sus datos. Una vez introducidos los datos se comprueba que estén correctos sin que coincidan con los datos de un alumno ya introducido (DNI).

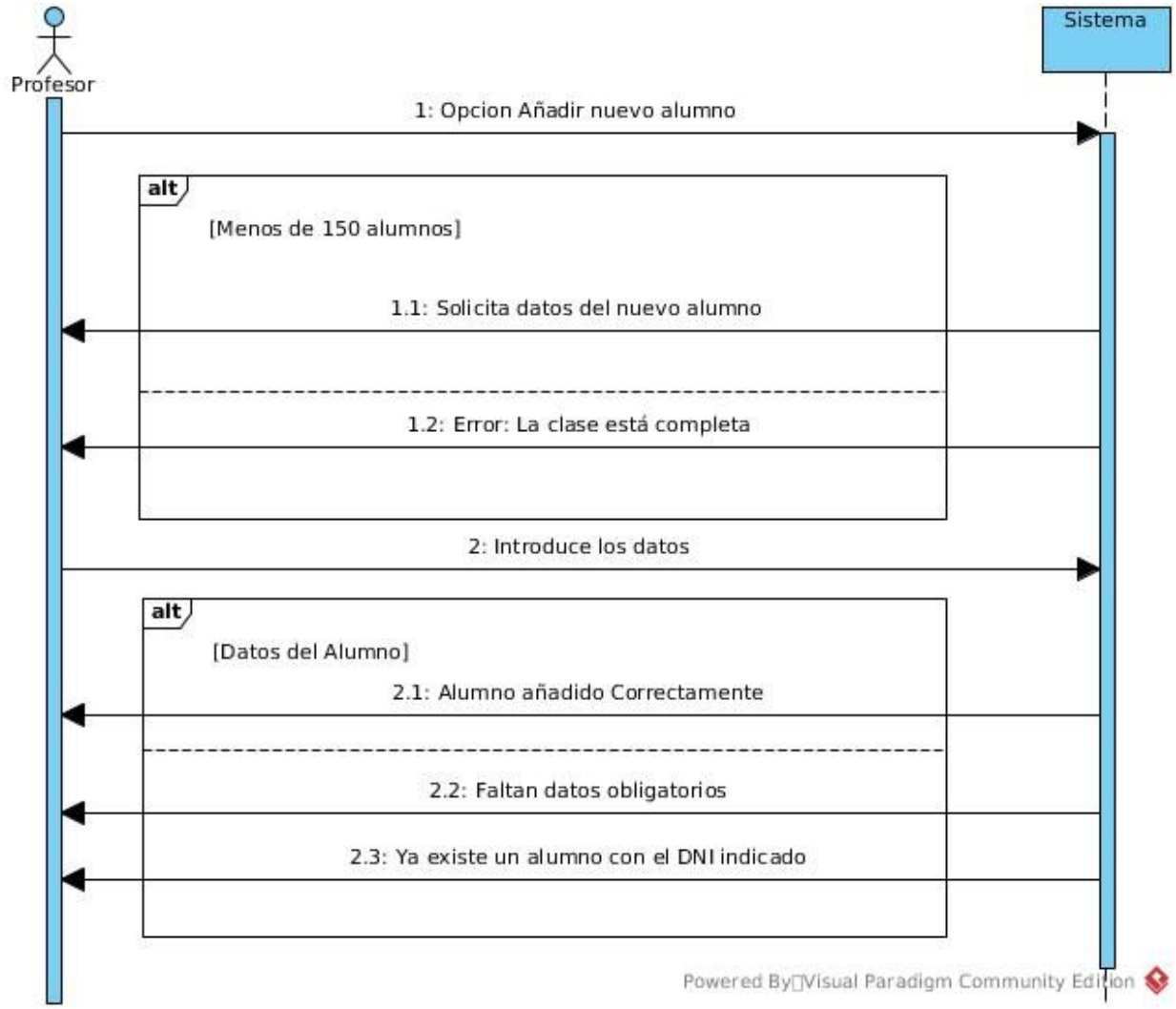

Figura 6.2: Diagrama de Secuencia-Añadir alumno

DS-03 Guardar Fichero

El diagrama comienza cuando el usuario (bien coordinador o bien ayudante) solicita la opción de guardar los datos del sistema en un fichero. En primer lugar se comprueba que el sistema tenga datos. Se solicita al usuario el nombre del fichero en el que se insertarán los datos del sistema. El sistema crea un fichero binario con los datos y a continuación borra los datos del sistema.

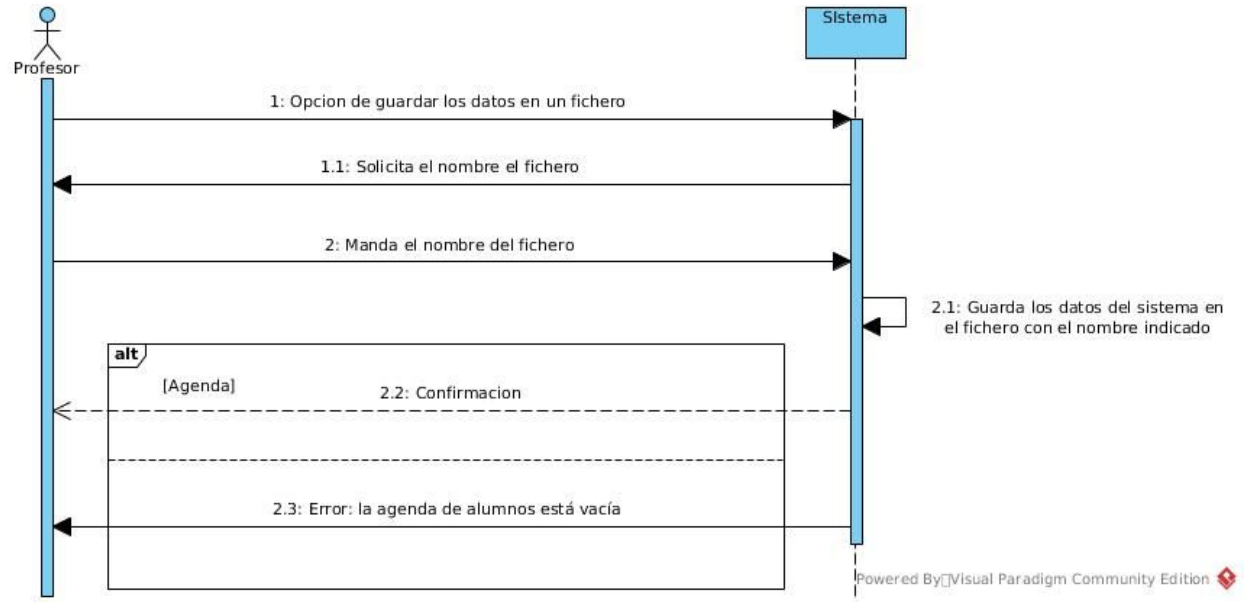

Figura 6.3: Diagrama de Secuencia-Guardar Fichero

DS-04 Mostrar Un Alumno

El diagrama comienza cuando el usuario (bien coordinador o bien ayudante) solicita la opción de mostrar un solo alumno del sistema. El sistema solicita el DNI o los apellidos del alumno que se desea visualizar. En caso de que los datos introducidos no se encuentren, se informa al usuario de que el alumno no existe mediante un mensaje de error. En caso de que el usuario haya introducido sus Apellidos y coincidan con los de otro alumno del sistema, se solicitará el DNI. En caso de encontrarlo se muestra al alumno en cuestión.

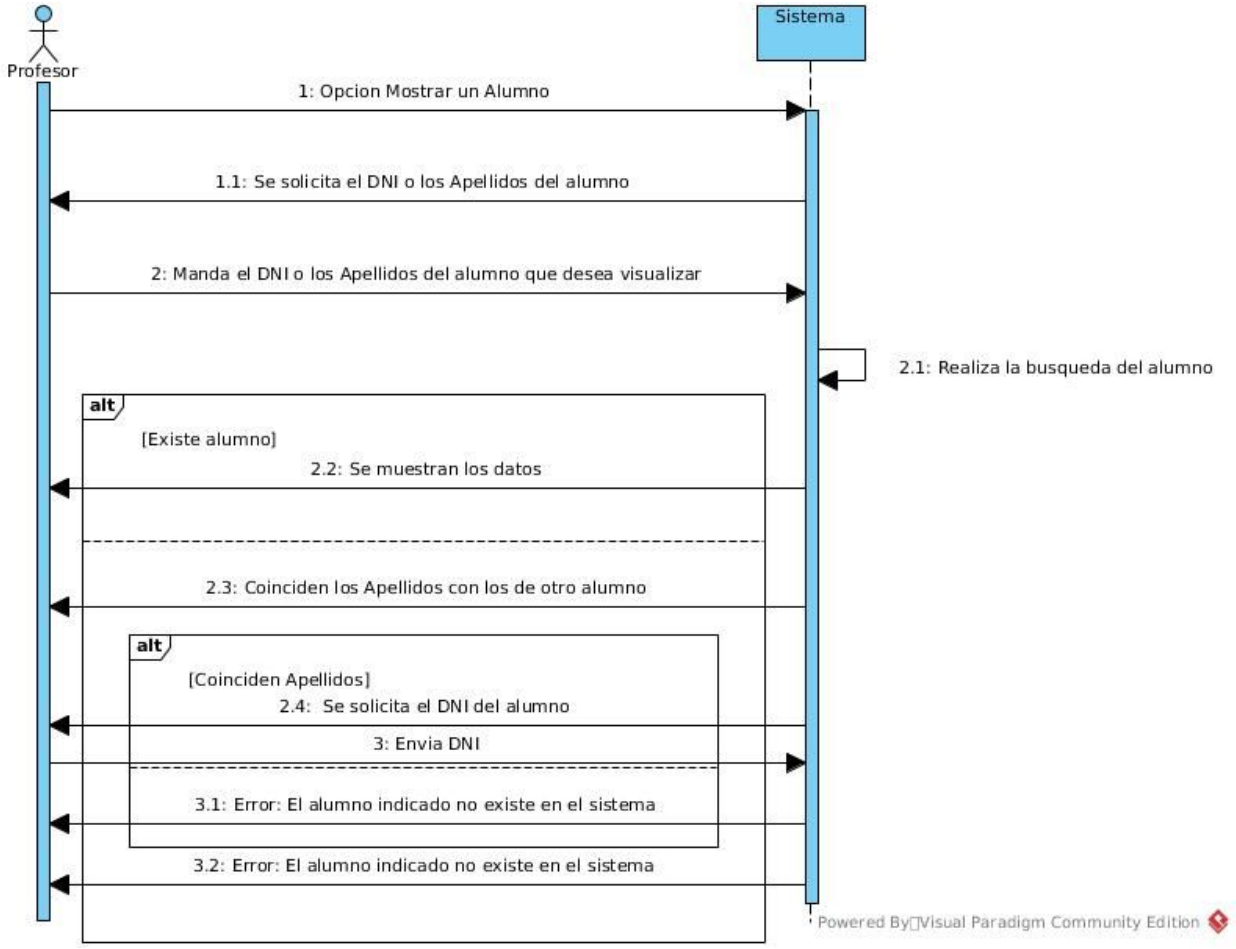

Figura 6.4: Diagrama de Secuencia-Mostrar un Alumno

DS-05 Mostrar Todos los Alumnos

El diagrama comienza cuando el usuario (bien coordinador o bien ayudante) solicita la opción de mostrar todos los alumnos. En primer lugar el sistema solicita al usuario el formato de visualización(Markdown o Terminal). Si el usuario desea visualizarlo en formato markdown, se copian los datos a un archivo .md y se enviará al usuario. Si el usuario desea visualizarlo en formato terminal, se mostrará por pantalla la lista de alumnos y sus datos. En caso de que en ese momento no existan alumnos en el sistema se informará al usuario con un mensaje de error.

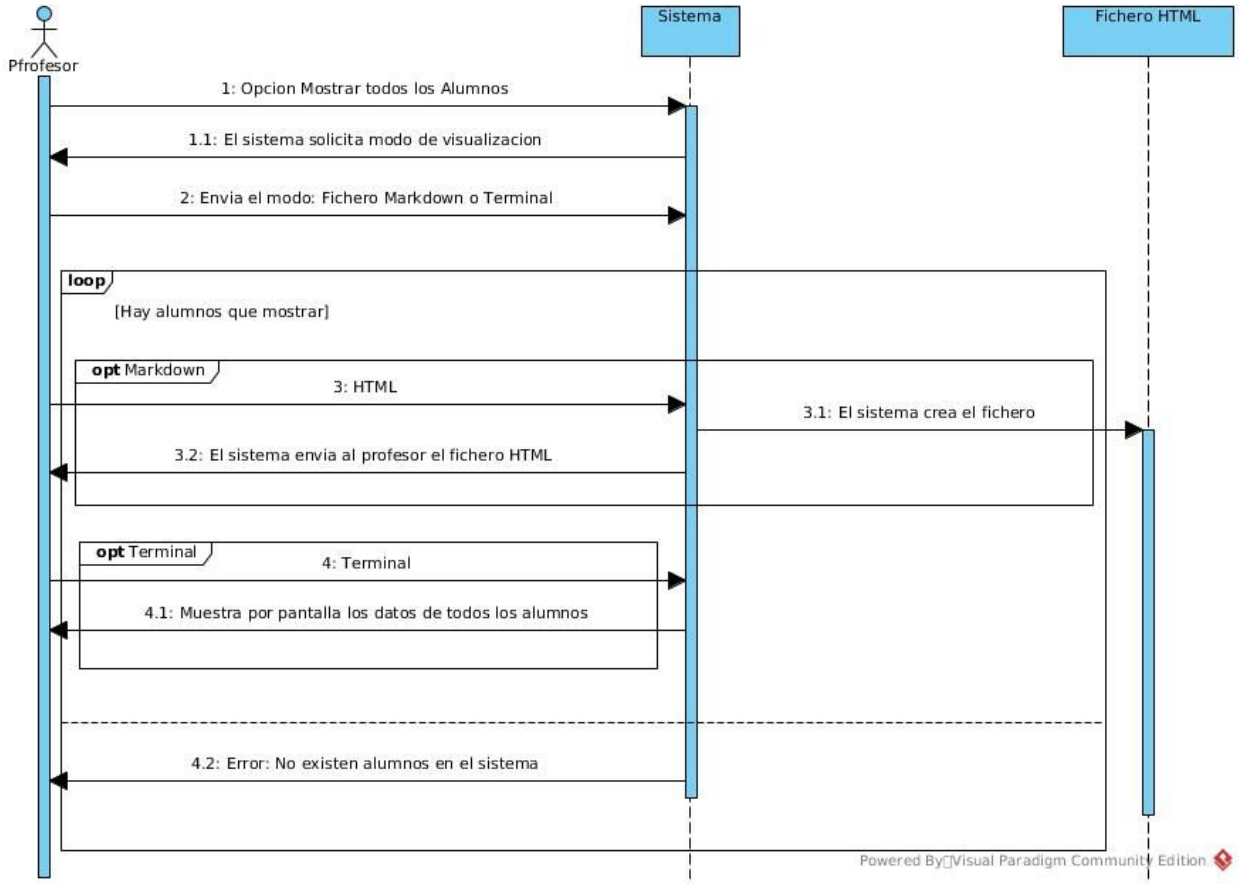

Figura 6.5: Diagrama de Secuencia-Mostrar todos Alumnos

DS-06 Mostrar Alumnos de Un Grupo

El diagrama comienza cuando el usuario (bien coordinador o bien ayudante) solicita la opción de mostrar los alumnos de 1 grupo concreto. El sistema solicita el número del grupo que desea visualizar (recordamos que hay grupos que pueden estar formados por un único alumno). El sistema recoge la información de los alumnos del grupo indicado y los muestra por pantalla. En caso de que el grupo indicado no exista se

informará al usuario con un mensaje de error.

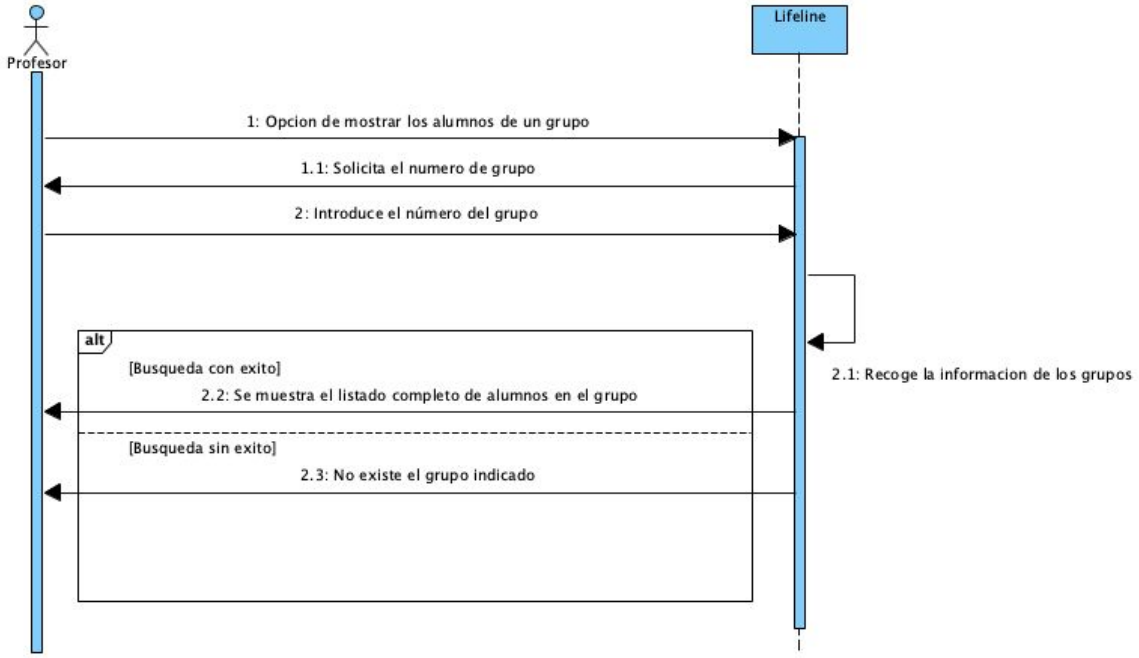

Figura 6.6: Diagrama de Secuencia-Mostrar Alumnos de Un Grupo

DS-07 Modificar Alumno

El diagrama comienza cuando el usuario (bien coordinador o bien ayudante) solicita la opción de modificar los datos de un alumno. El sistema solicita, al igual que en la opción de mostrar UN solo alumno, el DNI o los apellidos del alumno que desea modificar. En caso de que los apellidos coincidan, se solicitará el DNI. En caso de que no se encuentren los datos indicados en el sistema, se notificará al usuario con un mensaje de error indicando que el alumno no existe. Una vez encontrado el alumno, se solicitan los nuevos datos y se modifican.

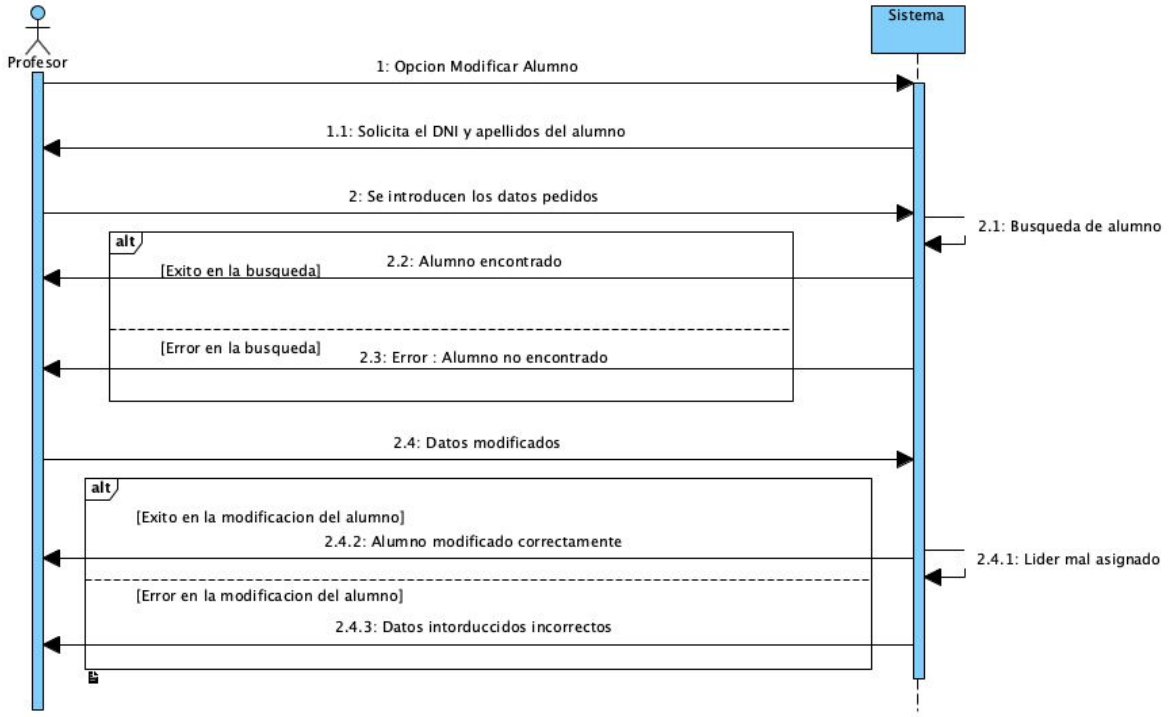

Figura 6.7: Diagrama de Secuencia-Modificar Alumno

DS-08 Borrar Alumno

El diagrama comienza cuando el usuario (bien coordinador o bien ayudante) solicita la opción de eliminar un alumno. El sistema solicita los Apellidos o el DNI del alumno que se desea eliminar. En caso de no encontrarlo se notifica al usuario de que dicho alumno no existe. En caso de encontrarlo se eliminan sus datos del sistema.

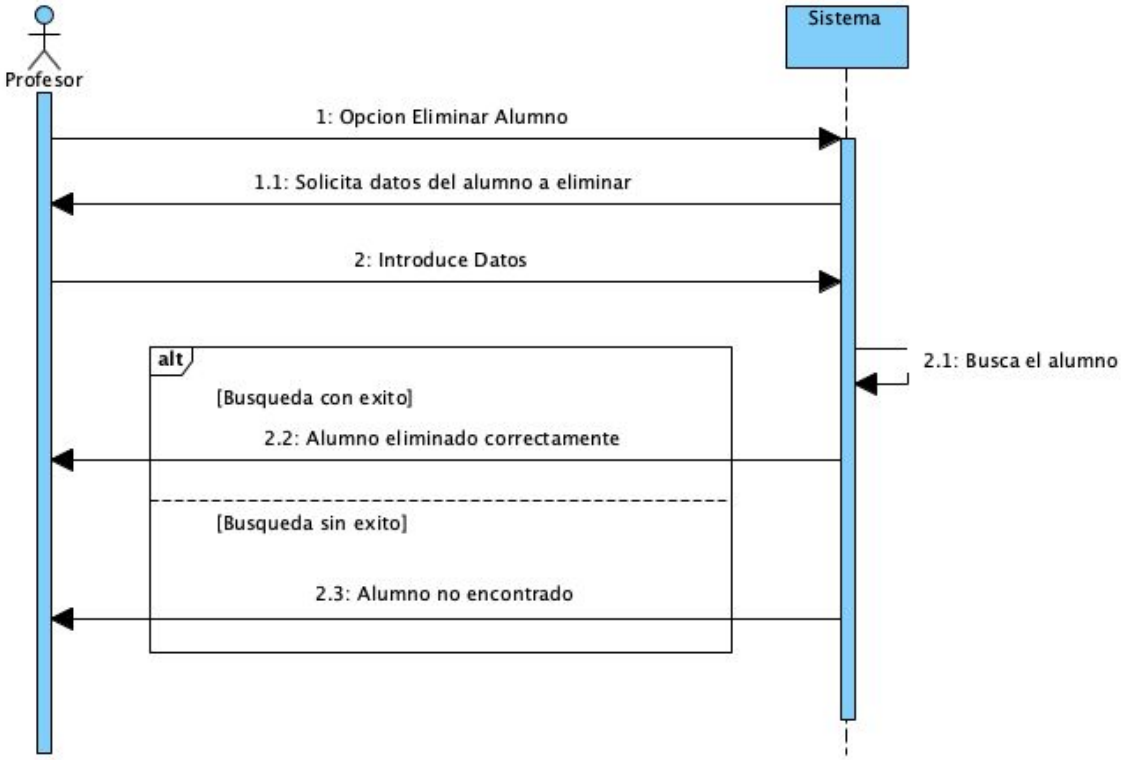

Figura 6.8: Diagrama de Secuencia-Eliminar Alumno

DS-09 Crear Copia Seguridad

El diagrama comienza cuando el usuario solicita la opción de crear copia de seguridad. El sistema comprueba que el usuario tiene los permisos adecuados. En caso de que no tenga permisos no podrá acceder a crear la copia de seguridad. En caso de tener permisos, solicitará un nombre del fichero al cual se le quiere hacer la copia. Si el fichero indicado no existe se dará un mensaje de error. Si el fichero indicado existe se crea la copia de seguridad.

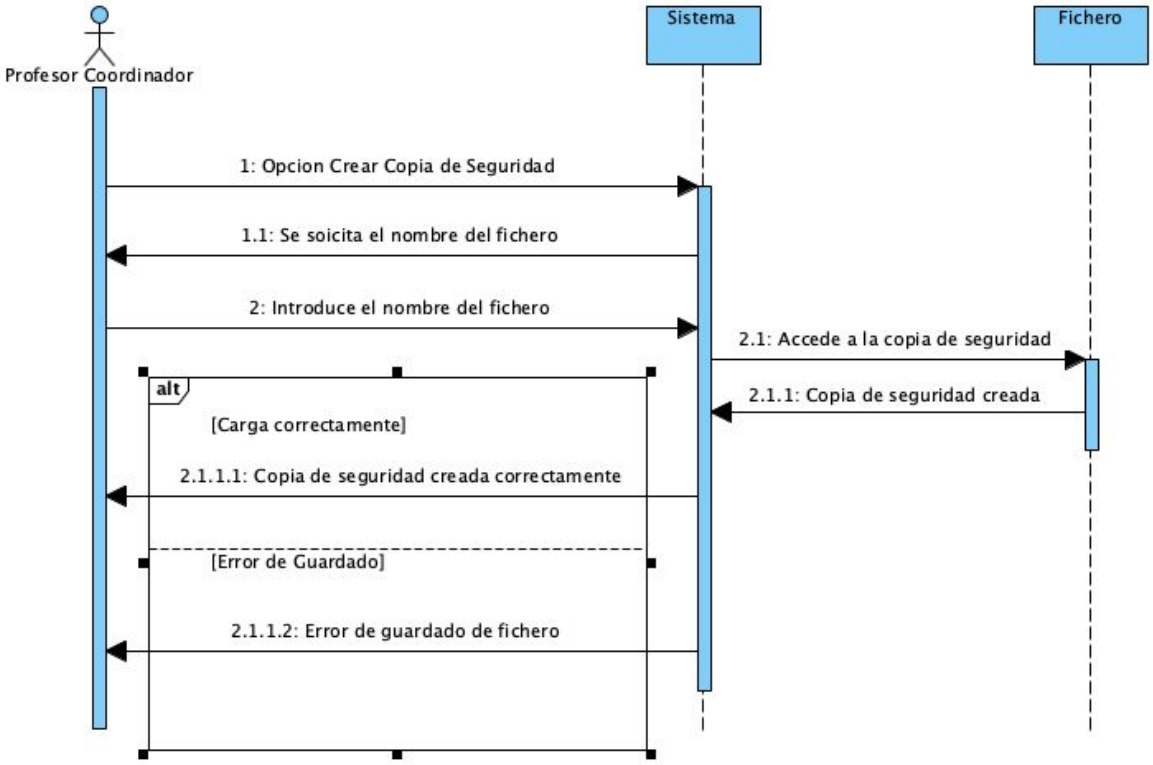

Figura 6.9: Diagrama de Secuencia-Crear Copia Seguridad

DS-10 Cargar Copia Seguridad

El diagrama comienza cuando el usuario solicita la opción de cargar copia de seguridad. El sistema comprueba que el usuario tiene los permisos adecuados. En caso de que no tenga permisos no podrá acceder a cargar la copia de seguridad. En caso de tener permisos, solicitará un nombre del fichero que contiene la copia de seguridad. Si el fichero no existe se da un mensaje de error. Si el fichero existe, se cargan en el sistema los datos de dicha copia de seguridad.

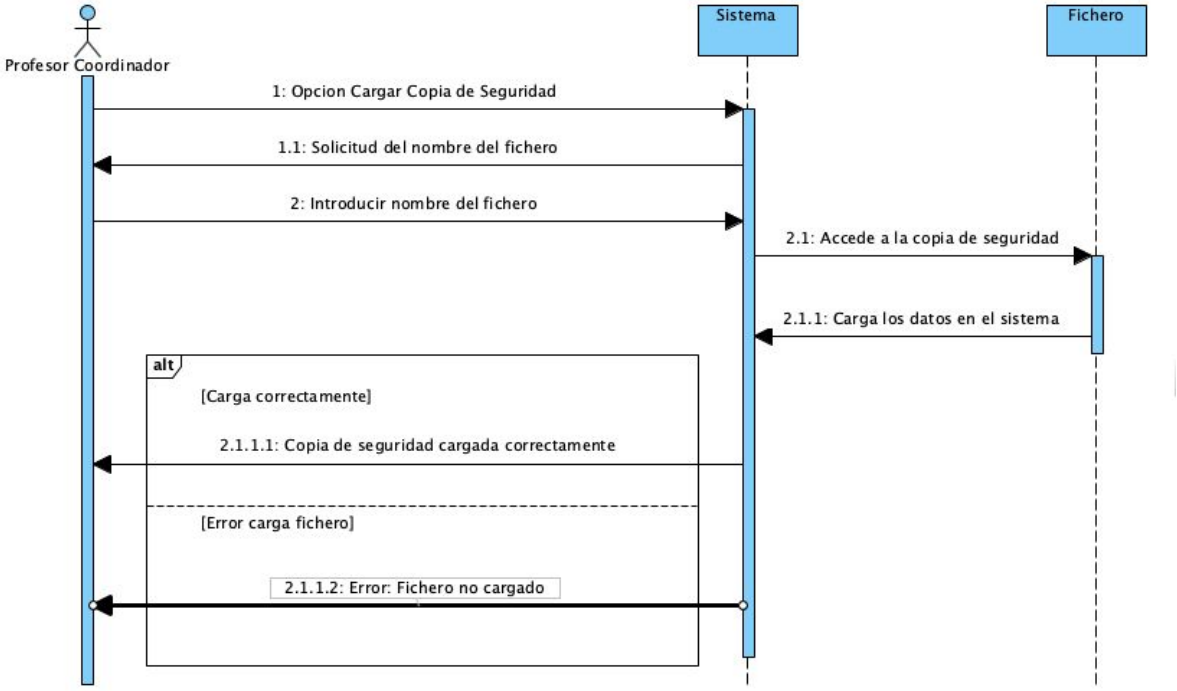

Figura 6.10: Diagrama de Secuencia-Cargar Copia Seguridad

7. Metodología SCRUM

PRODUCT BACKLOG:

En esta sección hemos organizado las tareas mediante su ID, asignando unas prioridades que tendremos en cuenta a la hora de desarrollar los diferentes Sprints de la práctica de implementación de código. A continuación se muestran dichas tareas con sus respectivas prioridades:

1. Login: ID:00 - Prioridad Cliente: 0 - Horas estimadas: 45min - Prioridad SCRUM: 3.
2. Cargar Fichero: ID:01 - Prioridad Cliente: 1 - Horas estimadas: 3.5h - Prioridad SCRUM: 1.
3. Añadir Alumno: ID:02 - Prioridad Cliente: 1 - Horas estimadas: 1h - Prioridad SCRUM: 1.
4. Guardar En Fichero: ID:03 - Prioridad Cliente: 2 - Horas estimadas: 2h - Prioridad SCRUM: 1
5. Mostrar Un Alumno: ID:04 - Prioridad Cliente: 3 - Horas estimadas: 50min - Prioridad SCRUM: 2
6. Mostrar Todos Alumnos: ID:05 - Prioridad Cliente: 3 - Horas estimadas: 1h - Prioridad SCRUM: 1
7. Mostrar Alumnos de un Grupo: ID:06 - Prioridad Cliente: 3 - Horas estimadas: 1h - Prioridad SCRUM: 2
8. Modificar Alumno: ID:07 - Prioridad Cliente: 4 - Horas estimadas: 50min - Prioridad SCRUM: 2
9. Borrar Alumno: ID:08 - Prioridad Cliente: 5 - Horas estimadas: 30min - Prioridad SCRUM: 1
10. Crear copia de seguridad: ID:09 - Prioridad Cliente: 6 - Horas estimadas: 40min - Prioridad SCRUM: 3

11. Cargar copia de seguridad ID:10 - Prioridad Cliente: 6 - Horas estimadas: 20min - Prioridad SCRUM: 3

Para representar de una manera más visual la gestión y organización de tareas en el Sprint, podemos ver en la siguiente tabla como se quedan éstas justo antes de empezar el desarrollo de la práctica

Tareas por Empezar	Tareas por Hacer	Tareas en Desarrollo	Tareas Finalizadas
ID: 00	ID: 01		
ID: 04	ID: 02		
ID: 06	ID: 03		
ID: 07	ID: 05		
ID: 09	ID: 08		
ID: 10			

Figura 7.1: Ejemplo de primera estimación Sprint 1

SPRINT BACKLOG (I)

En este apartado se explican las tareas que hemos desarrollado durante el primer Sprint, así como el reparto de las mismas para cada uno de los integrantes del grupo. Para poder tener un código mínimamente funcional de cara a la entrega del primer Sprint (lo que se conoce como mvp o producto mínimo viable) hemos decidido realizar las siguientes tareas:

Días de Desarrollo: 9.

Tareas a Desarrollar: 5.

1. Cargar Fichero: ID:01 - Alejandro.
2. Añadir Alumno: ID:02 - Javier.
3. Guardar Fichero: ID:03 - Guillermo y Alejandro.
4. Mostrar Todos los Alumnos: ID:05 - Guillermo
5. Borrar Alumno: ID:08 - Javier.

La siguiente imagen refleja el resultado de las tareas realizadas el dia de la entrega del primer Sprint.

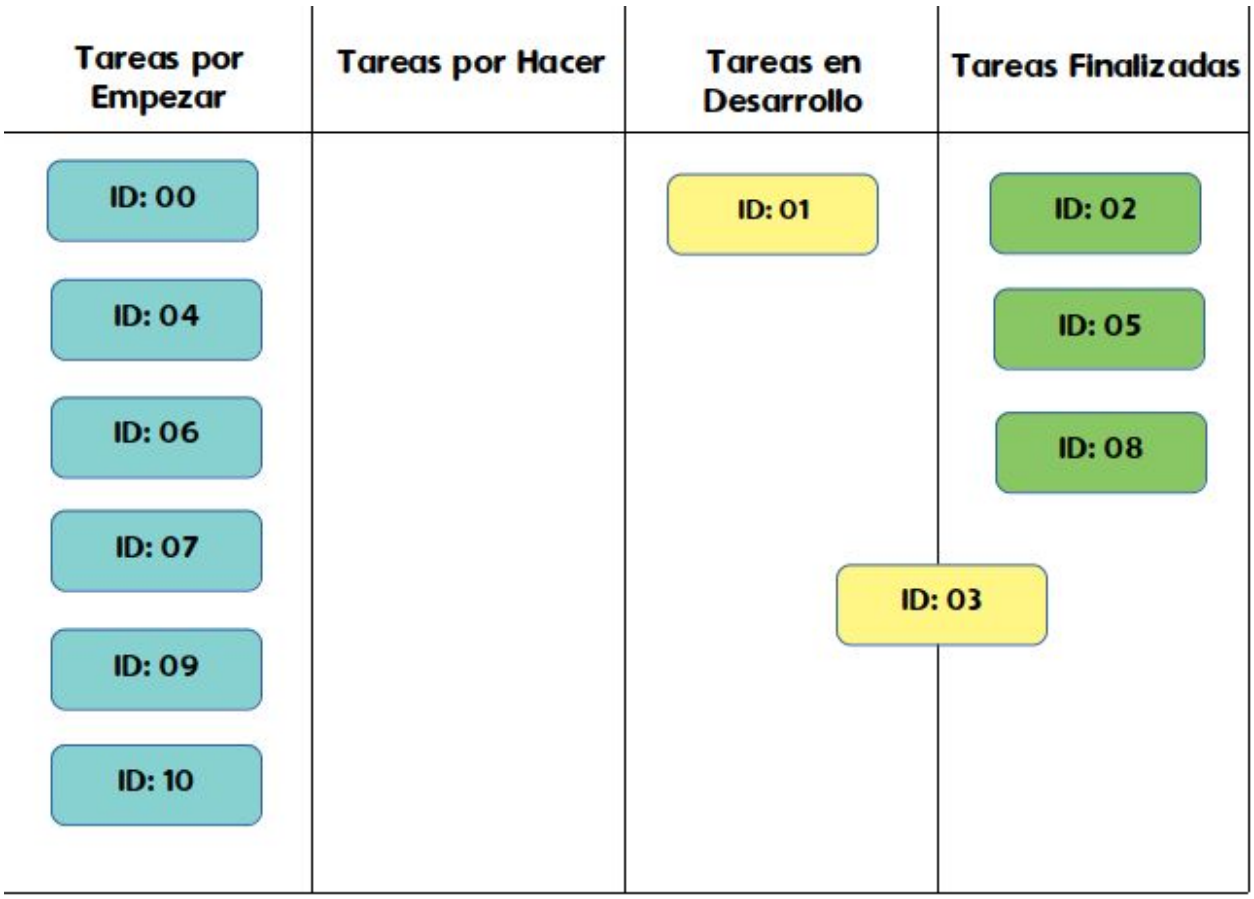

Figura 7.2: Ejemplo de Final de Sprint 1.

BURNDOWN CHART(I):

La gráfica que se muestra a continuación representa el seguimiento del trabajo realizado durante el primer Sprint:

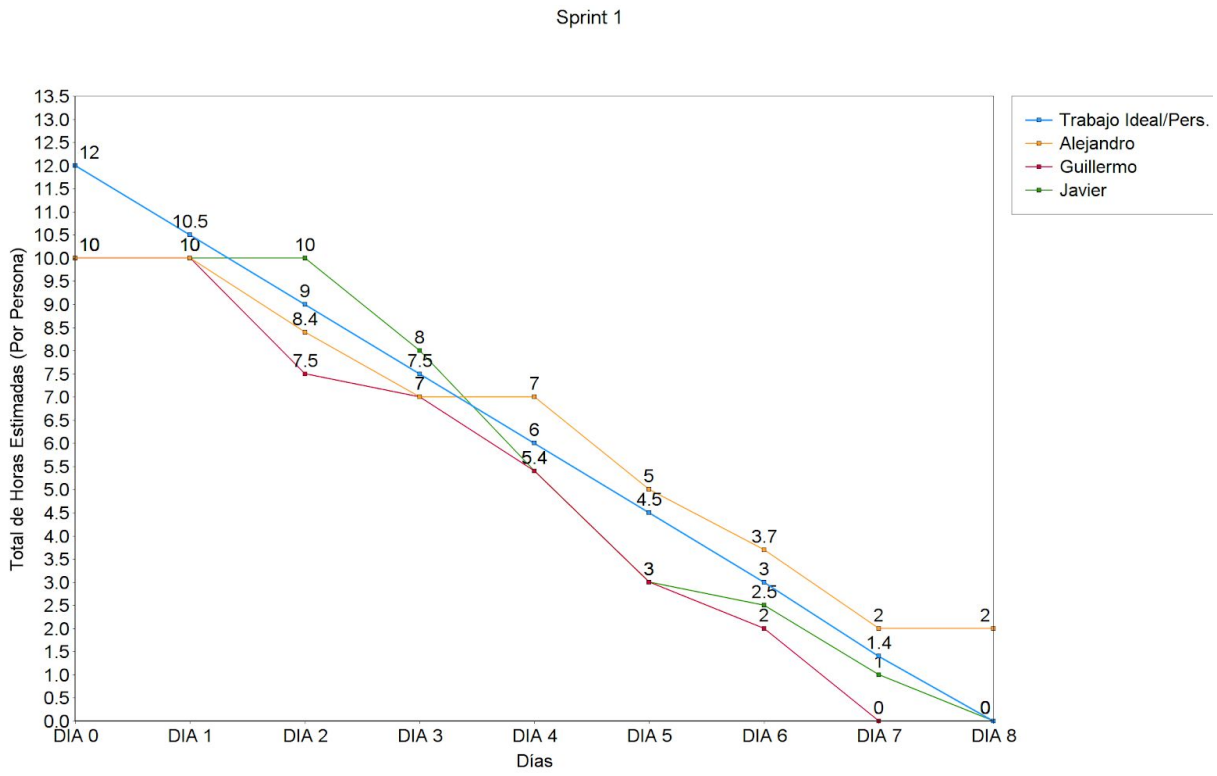

Figura 7.3: Ejemplo de BurnDown Chart individual

SPRINT BACKLOG (II)

En este apartado se explican las tareas que hemos desarrollado durante el segundo Sprint.

Días de Desarrollo: 7.

Tareas a Desarrollar: 6.

1. Login: ID:00 - Alejandro.
2. Mostrar Un Alumno: ID:04 - Javier.
3. Mostrar Alumnos de un Grupo: ID:06 - Javier.
4. Modificar Alumno: ID:07 - Javier.
5. Crear Copia Seguridad: ID:09 - Guillermo.
6. Cargar Copia Seguridad: ID:10 - Guillermo.

Además durante el segundo Sprint, hemos tenido que corregir fallos del Sprint anterior y completar las tareas que faltaron por terminar. A esto hay que sumar la funcionalidad de Borrar, Mostrar y Modificar por Apellidos del alumno ya que en el primer Sprint lo hicimos solo por DNI

Correcciones:

1. Cargar Fichero: ID:01 - Guillermo y Alejandro.
2. Control de Errores y funciones: isLider(), coincideAlumno(), etc. - Todos los miembros.

Finalmente el DashBoard una vez terminado el proyecto:

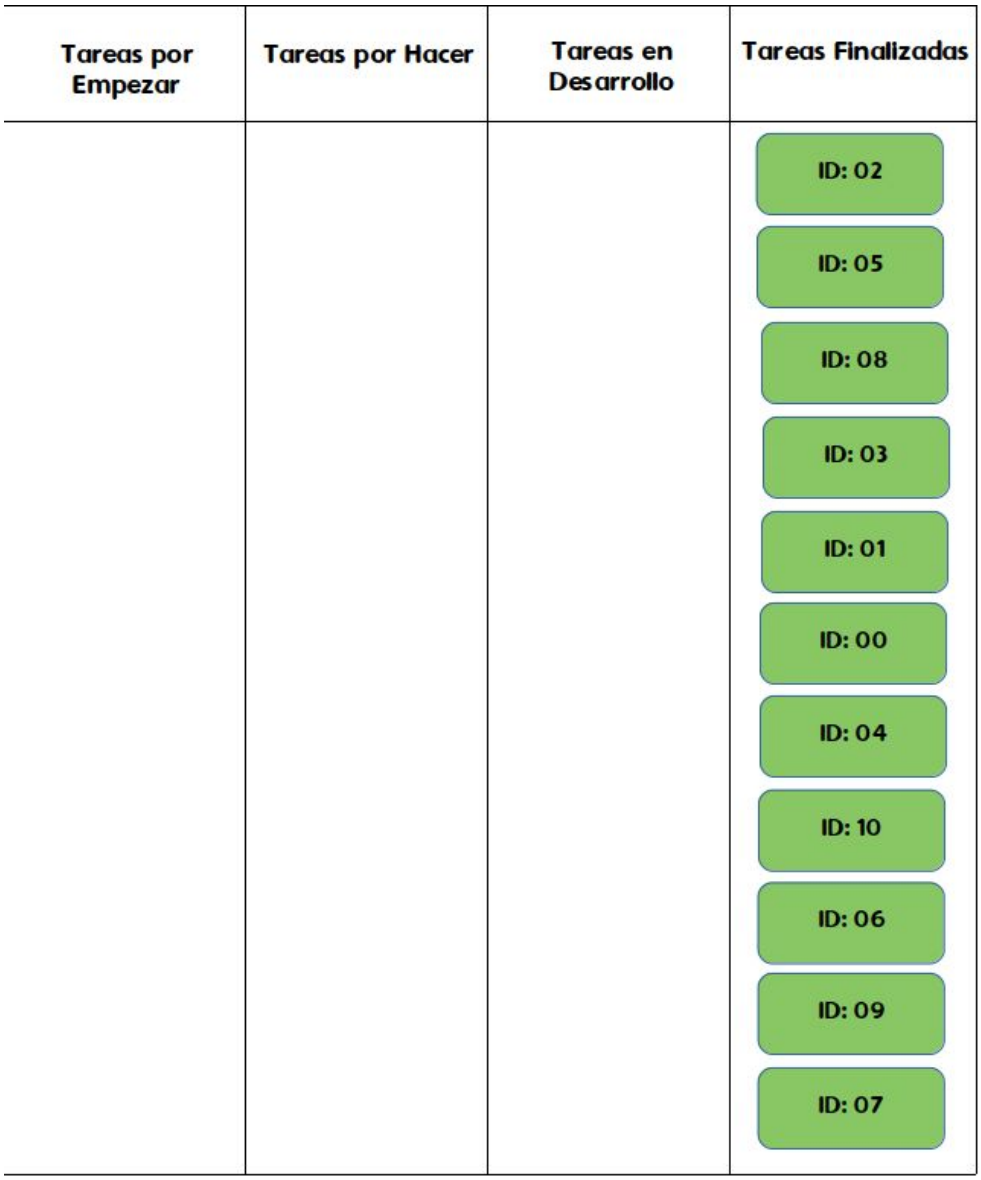

Figura 7.4: Ejemplo de Dashboard una vez terminado el Sprint

BURNDOWN CHART(II):

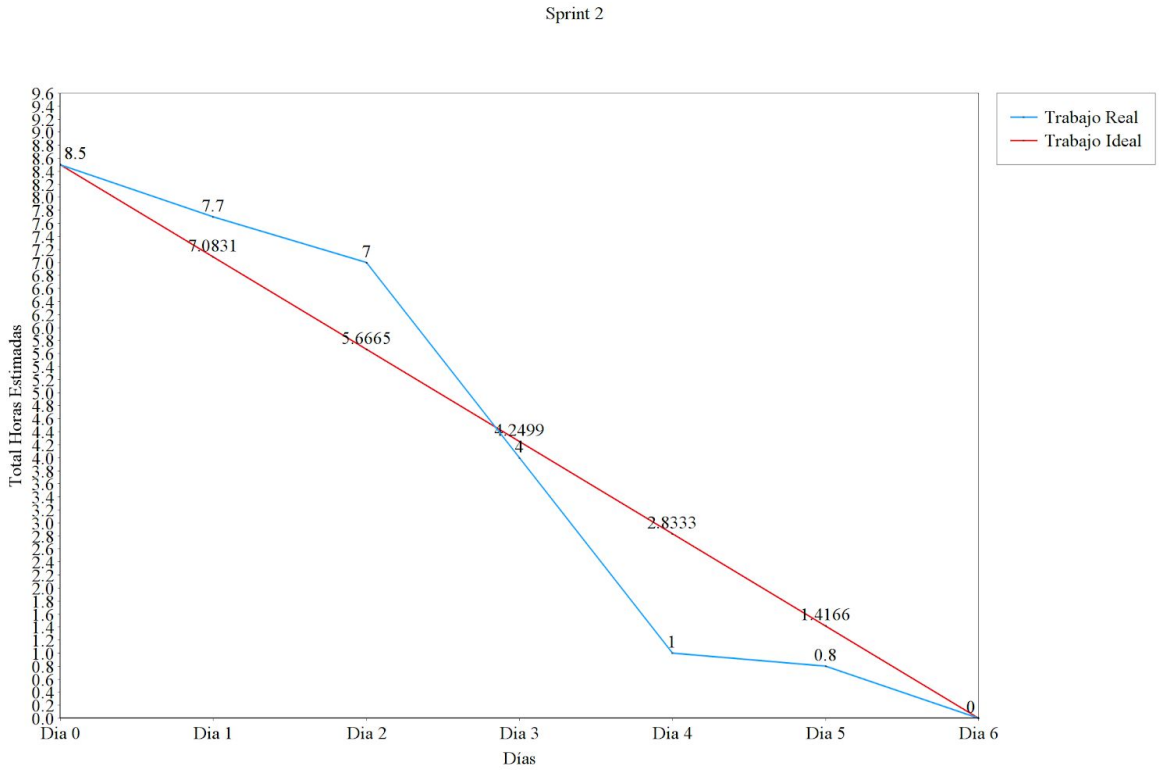

Figura 7.5: Ejemplo BurnDown Chart del segundo Sprint.

8. Matrices de Validación

Matriz Clases/CU

	00	01	02	03	04	05	06	07	08	09	10
Agenda		✓	✓	✓	✓	✓	✓	✓	✓		
Alumno			✓		✓	✓	✓	✓	✓		
Profesor	✓		✓		✓	✓	✓	✓	✓	✓	✓

Tabla 8.1: Matriz Clases/Casos de Uso

Matriz RF/CU

RF	CU

	00	01	02	03	04	05	06	07	08	09	10
00	✓										
01		✓	✓								
02			✓			✓					
03				✓							
04				✓	✓			✓			
05						✓					
06							✓				✓
07					✓			✓			
08		✓							✓		
09							✓		✓	✓	
10	✓									✓	✓

Tabla 8.2: Matriz Requisitos Funcionales/Casos de Uso

9. Bibliografía

Repositorio Github del proyecto: https://github.com/i62cogag/practica_IS

Apuntes moodle: https://moodle.uco.es/m1819/course/view.php?id=2230

Diagramas de Secuencia: https://manuel.cillero.es/doc/metrica-3/tecnicas/diagrama-de-interaccion/diagrama-de-secuencia/

Diagramas Visual Paradigm: https://www.visual-paradigm.com/guide/uml-unified-modeling-language/uml-class-diagram-tutorial/

Github de los miembros del grupo:

https://github.com/i62cogag

https://github.com/Javibu5

https://github.com/p42argea