SISTEMA DE ADMINISTRACIÓN DE CLASES Y ALUMNOS

Sistema de Administración de Clases y Alumnos

CREADO POR:

DRAGOS GEORGE, STAN

DANIEL, CASTILLO BELLO

ANGEL, MARTÍN MORENO

Este documento recogerá todo el proceso que hemos utilizado para el diseño, creación e implementación del Sistema de Administración de Clases y Alumnos.

Para la creación de este proyecto, hemos seguido la metodología SCRUM, que es una metodología basada en ciclos temporales cortos y de duración fija en el que se aplican de manera regular una serie de prácticas para trabajar colaborativamente en equipo y obtener el mejor resultado posible.

Realizada de la especificación de requisitos del cliente/usuario, se establecieron una serie de necesidades que el sistema tendría que cumplir conforme a las peticiones del mismo. Formalizados estos requisistos fueron generadas las historias de usuario, donde aparecen recogidas estas peticiones.

Con las historias de usuario generadas se pudo realizar un estudio del sistema para establecer las funcionalidades que el sistema tendría que realizar y los actores que intervendrían en él. Para modularizar este comportamiento se creó el diagrama de casos de uso. Una vez creado el diagrama de casos de uso y utilizando el paradigma de orientación a objetos, UML define dentro del modelo estructural el diagrama de clases para, una vez más, modularizar el sistema y así facilitar su entendimiento. Asimismo, para cada funcionalidad establecida en el diagrama UML define otro modelo, el de comportamiento, en el que es posible describir el comportamiento de dichas funcionalidades por medio de los diagramas de secuencia. Definido esto se establece pues un diagrama de secuencia para cada caso de uso en el que vienen descritas las interacciones entre las llamadas sociedades de objetos (actores del sistema).

A partir de las funcionalidades que se debían cumplir, su comportamiento y como interaccionaban entre sí, se aplicó el modelo estructural de UML y se realizó previamente y de manera adicional un estudio sobre que clases tomarían lugar en el sistema, así como los datos que estos manejarían y de qué forma para desarrollar posteriormente el diagrama de clases de nuestro sistema. Además de todo esto, las clases del sistema podrían estar o no asociadas entre sí. Una vez realizados estos tres diagramas, comenzaría la parte de implementación del sistema en el lenguaje de programación elegido.

Nuestro cliente nos encargó un sistema para poder gestionar sus alumnos y sus clases de una manera más cómoda. Para ello hemos contactado con el cliente y a cambio él nos indicó las historias de usuario que recoge las funcionalidades que se necesitan del sistema. A partir de aquí se generan las **HISTORIAS DE USUARIO**.

Historias de usuario

Las historias de usuario se indican en forma de tarjetas, donde cada tarjeta expone una funcionalidad por delante (ANVERSO) junto con el identificador de la tarjeta, y por detrás el objetivo de dicha funcionalidad.

(Anverso)

ID: Identificador **Nombre de la funcionalidad**

Información primaria sobre la funcionalidad

Prioridad: Prioridad a asignar a la hora de priorizar la importancia de las funcionalidades

(REVERSO)

- Objetivos de dicha funcionalidad

Empezaremos con las historias de usuario que nuestro cliente nos ha indicado.

Historias de usuario

(ANVERSO)

ID: 1 **Almacenar usuario**

Quiero tener almacenada la información de los alumnos.

Prioridad: 2

(REVERSO)

- Quiero tener almacenados todos los datos relevantes sobre los alumnos.
- Se debe distinguir si un alumno es lider de grupo o no.

(ANVERSO)

ID: 2 **Guardar clases**

Quiero poder guardar la información de las distintas clases.

Prioridad: 2

(REVERSO)

- Preferiría tener almacenada la información de las distintas clases en ficheros.

(ANVERSO)

ID: 3 **Cargar clases**

Quiero poder utilizar cuando quiera la información de las clases.

Prioridad: 2

(REVERSO)

*Quiero poder cargar la información de las distintas clases.

(ANVERSO)

ID: 4 **Buscar alumno**

Quiero tener la posibilidad de buscar a un alumno por su DNI o por el grupo al que pertenece

Prioridad: 3

(REVERSO)

- Debo poder buscar alumnos usando como posibles parámetros:
  - DNI
  - Grupo al que pertenece

(ANVERSO)

ID: 5 **Actualizar alumno**

Quiero tener la posibilidad de actualizar la información de un alumno.

Prioridad: 4

(REVERSO)

- Quiero poder modificar cualquier tipo de información referente a cada alumno.

(ANVERSO)

ID: 6 **Eliminar alumno**

Quiero poder eliminar a un alumno.

Prioridad: 4

(REVERSO)

- Quiero poder borrar la información de un alumno y eliminar sus registros.

(ANVERSO)

ID: 7 **Listar alumnos**

Debe haber una opción para ver todos los alumnos de una clase

Prioridad: 5

(REVERSO)

- Quiero tener una opción que liste a todos los alumnos de una clase.
- Los alumnos que sean líderes de algún grupo deben estar marcados de alguna forma especial.

(ANVERSO)

ID: 8 **Sistema GNU/Linux**

El programa debe funcionar en Linux.

Prioridad: 1

(REVERSO)

- El programa debe funcionar en cualquier sistema GNU/Linux.

(ANVERSO)

ID: 9 **Campos *Equipo* y *Líder* no obligatorios**

Quiero que la información relacionada con el equipo al que pertenece un alumno y si es líder o no de dicho equipo sean opcionales.

Prioridad: 2

(REVERSO)

- Los campos *Equipo* y *Líder* de la información de los alumnos no serán obligatorios.

(ANVERSO)

ID: 10 **Máximo 150 alumnos**

No podrán haber más de 150 alumnos por clase.

Prioridad: 2

(REVERSO)

- El máximo de alumnos que se pueden almacenar por cada clase será de 150.

(ANVERSO)

ID: 11 **Ficheros binarios**

La información de los alumnos se tendrá que almacenar en ficheros binarios.

Prioridad: 2

(REVERSO)

- Los ficheros en los que se guarde la información de los alumnos deberán ser binarios.

A partir de estas historias de usuario, nuestro equipo ha podido extraer las **ESPECIFICACIONES DE REQUISITOS** que es un documento que recoge de una forma más detenida y detallada las historias de usuario. Aquí podemos distinguir entre dos tipos de requisitos: Requisitos Funcionales y Requisitos no Funcionales.

Los Requisitos Funcionales son especificaciones técnicas sobre las funciones con las cuales el usuario puede interactuar.

Los Requisitos no Funcionales son restricciones que debe tener el sistema.

Especificación de requisitos

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

Requisitos

La prioridad de cada uno de los requisitos vendrá determinada por una numeración ascendente desde el 1 (siendo la prioridad más alta) a 5 (siendo la más baja).

Requisitos Funcionales

1. (**2**) Se deberán guardar datos personales para cada alumno. Los datos a guardar son los siguientes:
    - DNI
    - Nombre
    - Apellidos
    - Correo Electrónico
    - Teléfono
    - Dirección
    - Curso más alto en el que está matriculado
    - Fecha de nacimiento
    - Equipo al que pertenece
    - Líder/No líder
2. (**2**) Deberá ser posible grabar los datos de los alumnos en ficheros.
3. (**2**) Deberá ser posible cargar los datos de los alumnos en ficheros.
4. (**3**) Deberá ser posible buscar a un usuario:
    - Por DNI
    - Por Apellido
    - Por Grupo
5. (**4**) Deberá ser posible actualizar los datos de un alumno.
6. (**4**) Deberá ser posible eliminar a un alumno.
7. (**5**) Cuando se listen los usuarios de la clase, deberá ser posible mostrar los datos ordenados:
    - Por DNI
    - Por nombre
    - Por apellido
    - Por curso matriculado
8. (**2**) El profesor deberá ser capaz de registrarse en el sistema.
9. (**2**) El profesor deberá ser capaz de iniciar sesión en el sistema.

Requisitos no Funcionales

1. (**1**) La herramienta deberá funcionar bajo sistemas GNU/Linux.
2. (**2**) Los campos *Líder* y *Equipo* no serán obligatorios.
3. (**2**) Solo se podrán guardar un máximo de 150 alumnos.
4. (**2**) El tipo de fichero en el que se guardarán los datos deberá ser de tipo binario.

Una vez analizado las historias de usuario y creado el documento de la especificación de requisitos, toca entrar con más profundidad en los requisitos funcionales, analizándolos con detenimiento para poder conocer con mayor precisión el desempeño técnico de las funciones. Este análisis se denomina **CASO DE USO** y, al igual que las historias de usuario, se representa por tarjetas.

Cada requisito funcional debe tener asignado un caso de uso y es mejor hacerlos de forma organizada, siguiendo el orden impuesto por los identificadores de las historias de usuario o de la especificación de requisitos. En nuestro caso, hemos organizado ambas de la misma manera.

Nombre de la función

ID: Identificador
Breve Descripción: Una breve descripción sobre la funcionalidad

---

Actores principales: Usuario que interactúa con la función de forma directa

Actores secundarios: Usuario que interactúa con la función de forma indirecta

1. **Precondiciones**

- Condiciones que se deben dar para que la función desempeñe correctamente.

2. **Flujo Principal**

- Uso y funcionalidad que debe tener la función

3. **Postcondiciones**

- Resultado que ha de lograr.

4. **Flujos alternativos**

- Formas alternativas que tiene el sistema de interactuar, por ejemplo, en caso de errores.

Ahora que hemos definido los casos de uso y su funcionalidad, mostraremos los casos de uso que han resultado en nuestro proceso:

Casos de Uso

Almacenar alumno

ID: 01

Breve Descripción: Se pueden almacenar los alumnos

Actores principales: Usuario

Actores secundarios: Alumno

1. **Precondiciones**

- El alumno debe estar matriculado en el curso del profesor.
- El alumno no debe estar ya cargado en el sistema.

2. **Flujo Principal**

- El caso de uso comienza cuando el usuario necesita almacenar un alumno.
- El sistema recoge los datos del alumno.
- Se podrá indicar si el alumno pertenece a algún grupo y si es líder o no.

3. **Postcondiciones**

- La información del alumno ha de estar almacenada.
- Debe quedar distinguido si un alumno es líder de grupo o no.

4. **Flujos alternativos**

- Si el sistema no puede guardar un alumno y sus datos, mostrará un mensaje de error por pantalla, tras lo cual se volverá al menú principal.
- Si otro alumno tiene el mismo DNI o e-mail, se mostrará un mensaje de error por pantalla, tras lo cual se volverá al menú principal.
- Si el alumno es líder de un grupo en el que ya hay un líder con anterioridad, se mostrará un mensaje de error por pantalla, tras lo cual se volverá al menú principal.

Guardar clase

ID: 02

Breve Descripción: Almacenar información con respeto a las clases

---

Actores principales: Usuario Actores secundarios: Clase

1. **Precondiciones**

- La clase debe existir.

2. **Flujo Principal**

- El caso de uso da comienzo cuando se necesita guardar una clase.
- El sistema recoge los datos y guarda una clase en un fichero binario.

3. **Postcondiciones**

- La información de la clase debe estar almacenada en un fichero binario.

4. **Flujos alternativos**

- Si una clase no se puede guardar, se imprimirá un error por pantalla, tras lo cual se volverá al menú principal.
- Si una clase no se puede encontrar entre los ficheros, se imprimirá un error por pantalla.
- Si se introduce un nombre de fichero que tiene un fichero ya existente, el sistema sobrescribirá la información en dicho fichero.

Cargar Clase

ID: 03

Breve Descripción: Poder usar la información de una clase guardada en un fichero

---

Actores principales: Usuario Actores secundarios: Clase

1. **Precondiciones**

- La información de la clase debe estar almacenada en un fichero.

2. **Flujo Principal**

- El caso de uso da comienzo cuando el usuario quiere importar una clase desde un fichero.
- El sistema recoge la información del fichero binario.

3. **Postcondiciones**

- La información de la clase ha de haberse cargado satisfactoriamente.

4. **Flujos alternativos**

- Si no se puede leer el fichero, se mostrará un error por pantalla, tras lo cual se volverá al menú principal.

Búsqueda de alumno

ID: 04

Breve Descripción: Se podrá buscar un alumno.

Actores principales: Usuario Actores secundarios: Alumno

1. **Precondiciones**

- Debe haber una clase cargada.
- El alumno debe existir en dicha clase.

2. **Flujo Principal**

- El caso de uso comienza cuando el usuario desea buscar un alumno.
- El alumno se debe buscar por DNI, por apellido o por el grupo al que pertenece.
- Se muestra la información del alumno por pantalla.

3. **Postcondiciones**

- La información del alumno debe aparecer en la pantalla.
- Si hay varios alumnos con el mismo apellido se mostrará por pantalla todos los alumnos con el apellido.
- Si buscamos por grupos, se mostrará por pantalla todos los alumnos pertenecientes a dicho grupo.

4. **Flujos alternativos**

- Si el alumno no se puede encontrar, se mostrará un mensaje de error por pantalla, tras lo cual se volverá al menú principal.
- Si existe más de un alumno cuyos datos coinciden con los de la búsqueda, se mostrarán todos ellos por pantalla.

Actualizar alumno

ID: 05

Breve Descripción: Se podrá actualizar la información de un alumno.

Actores principales: Usuario Actores secundarios: Alumno

1. **Precondiciones**

- Debe haber una clase cargada.
- El alumno deberá existir en dicha clase.
- El alumno deberá tener información guardada anteriormente para poder modificarla.

2. **Flujo Principal**

- El caso de uso comienza cuando el usuario desea cambiar la información de alguno de sus alumnos.
- Se buscará al alumno cuya información se desea modificar por el DNI o por el apellido.
- El sistema recogerá la información nueva y la guardará, sobreescribiendo la anterior.

3. **Postcondiciones**

- Los datos del alumno se deberán haber modificado con éxito.

4. **Flujos alternativos**

- Si no se puede modificar la información, se mostrará un error por pantalla, tras lo cual se volverá al menú principal.

Eliminar alumno

ID: 06

Breve Descripción: Podremos eliminar un alumno.

Actores principales: Usuario Actores secundarios: Alumno

1. **Precondiciones**

- Debe haber una clase cargada.
- El alumno debe existir en dicha clase.

2. **Flujo Principal**

- El caso de uso da comienzo cuando el usuario desea borrar un alumno y sus datos permanentemente.
- Se podrá buscar al alumno cuya información se desea eliminar por el DNI o por el apellido.
- El sistema elimina cualquier información relacionada a ese alumno.

3. **Postcondiciones**

- La información del alumno deberá haber sido borrada de forma satisfactoria.

4. **Flujos alternativos**

- Si un alumno no se puede borrar, se mostrará un mensaje de error por pantalla, tras lo cual se volverá al menú principal.

Listar Alumnos

ID: 07

Breve Descripción: Se mostrarán todos los alumnos de una clase

---

Actores principales: Usuario Actores secundarios: Clase

1. **Precondiciones**

- Deberá haber una clase cargada.
- Dicha clase debe contener al menos un alumno.

2. **Flujo Principal**

- El caso de uso comienza cuando el usuario desea ver todos los alumnos de una clase.
- El sistema mostrará por pantalla la información de los alumnos perteneciente a la clase.

3. **Postcondiciones**

- La información de los alumnos de la clase debe aparecer en la pantalla por orden alfabético del primer apellido.

4. **Flujos alternativos**

- Si el sistema no puede encontrar la clase, mostrará un error por pantalla, tras lo cual se volverá al menú principal.
- Si el sistema no puede mostrar la información de la clase, mostrará un error por pantalla, tras lo cual se volverá al menú principal.

Registro del Profesor

ID: 08

Breve Descripción: El profesor debe ser capaz de registrarse en el sistema

---

Actores principales: Usuario Actores secundarios: Sistema

1. **Precondiciones**

- El profesor no debe estar registrado en el sistema.
- El nombre de usuario escogido no debe estar ocupado.

2. **Flujo Principal**

- El caso de uso comienza cuando un usuario quiere registrarse en el sistema.
- Se pedirá un nombre de usuario y una contraseña, que se guardarán en un fichero binario.
- La comprobación de la contraseña se hará con una comparativa con el fichero binario.

3. **Postcondiciones**

- El usuario debe quedar registrado en el sistema.

4. **Flujos alternativos**

- Si el nombre de usuario ya está recogido en el sistema, se mostrará un error por pantalla y se volverá a la pantalla de registro.
- Si el usuario ya está registrado en el sistema, se le mostrará un error por pantalla indicándole que el nombre de usuario ya está registrado y se vuelve a la pantalla de registro.

Inicio de sesión

ID: 09

Breve Descripción: El usuario debe ser capaz de acceder al sistema

---

Actores principales: Usuario Actores secundarios: Sistema

1. **Precondiciones**

- El nombre de usuario debe estar registrado en el sistema.
- La contraseña corresponda con el nombre de usuario.
- Que no haya otra sesión iniciada.

2. **Flujo Principal**

- El caso de uso comienza cuando el usuario quiere acceder al sistema.
- El usuario debe introducir un nombre de usuario y una contraseña que ya existan en el sistema.
- Tras introducir la información requerida, se llevará el usuario al menú principal.

3. **Postcondiciones**

- Tras una correcta validación, el usuario habrá accedido al menú principal.

4. **Flujos alternativos**

- Si el usuario ha introducido una contraseña errónea, se mostrará un error por pantalla y se llevará al usuario al menú de inicio de sesión
- Si el usuario ha introducido un nombre de usuario que no está registrado en el sistema se mostrará un error por pantalla y se llevará al usuario al menú de inicio de sesión.

Una vez hemos creado los casos de uso, podemos adentrarnos ya en la parte técnica de la programación. Lo primero que haremos será elegir el lenguaje de programación que usaremos para el programa. En este caso se ha establecido que el lenguaje de programación será C++.

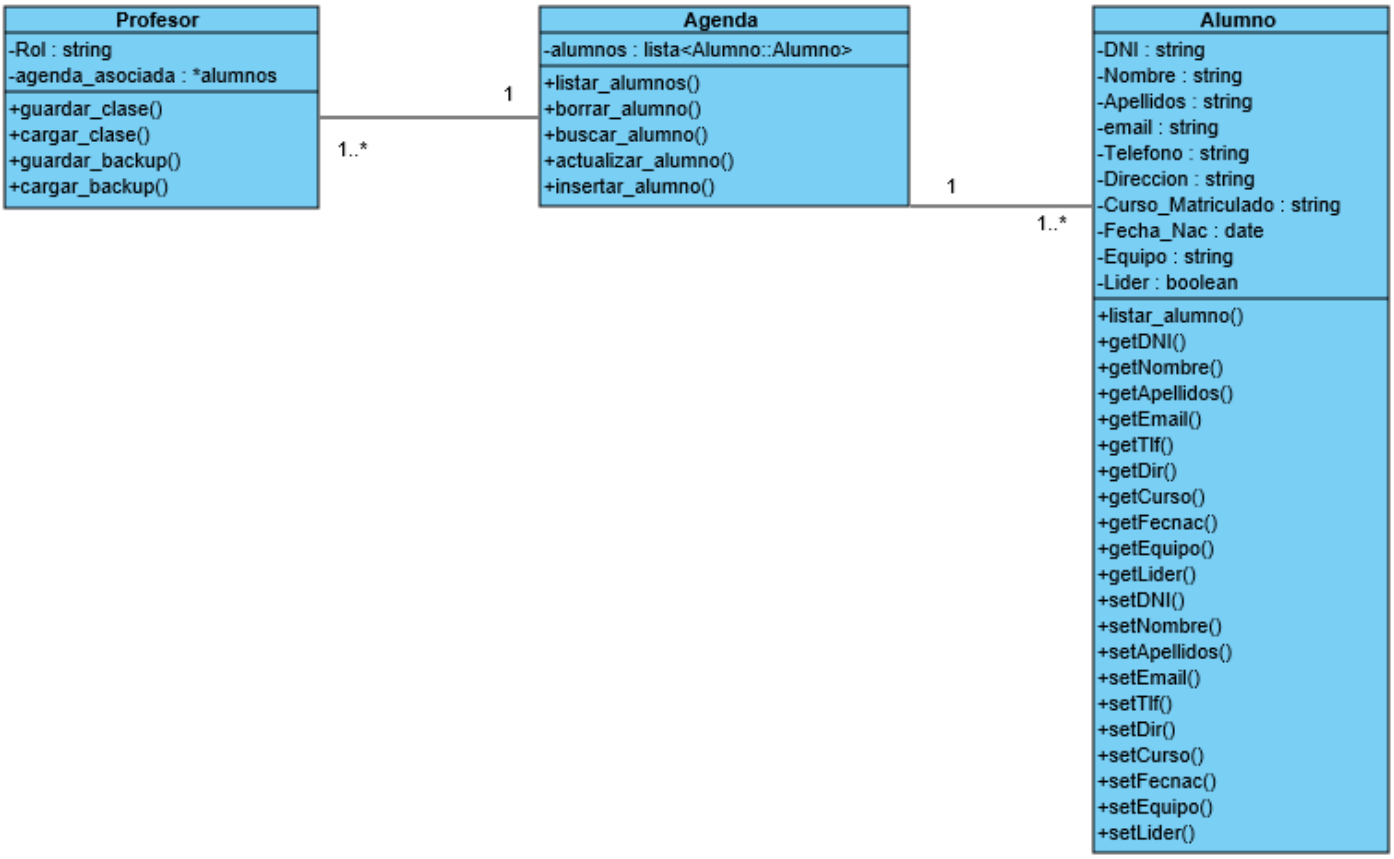
Ahora, con toda la información acumulada, podemos diseñar los diagramas. Hay dos tipos de diagramas: los diagramas de clase y los diagramas de casos de uso.

Los diagramas de clase muestran las relaciones entre los distintos elementos del sistema.

Los diagramas de casos de uso muestran el desempeño técnico de los casos de uso.

Empezaremos con el diagrama de clase.

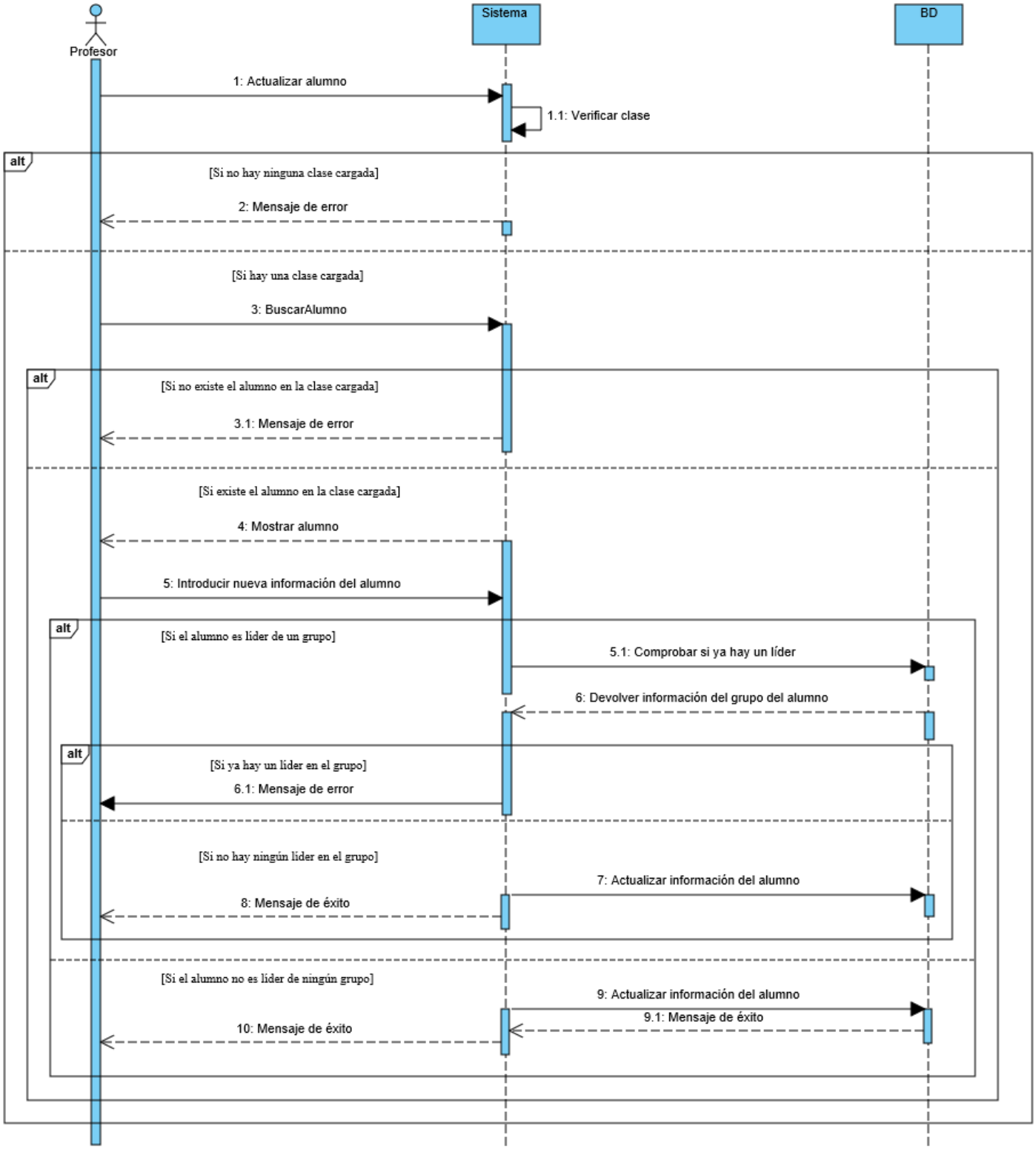
Diagrama de clase



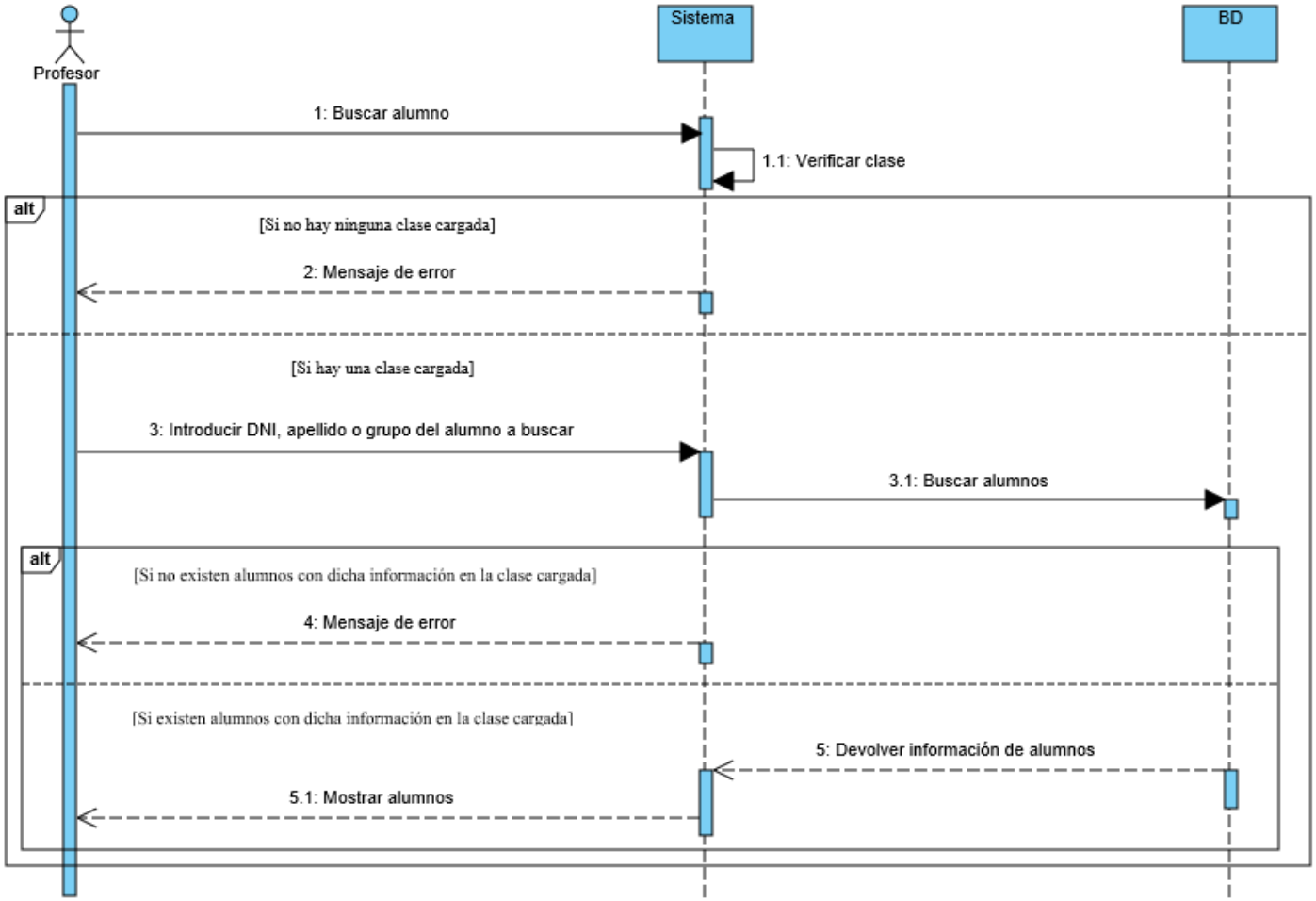
Una vez diseñado el diagrama de clase, podemos pasar a los diagramas de casos de uso.

Diagramas de casos de uso

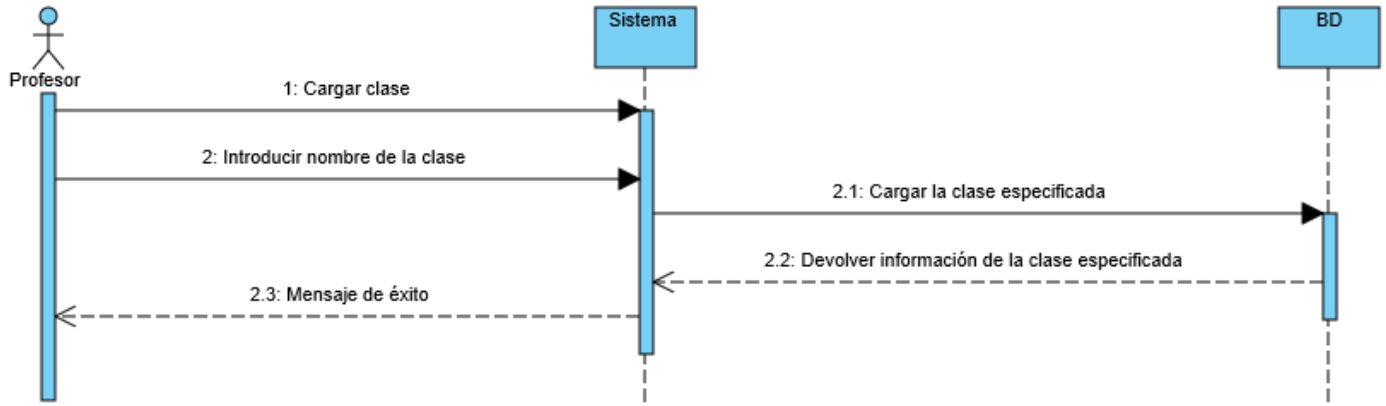
Actualizar Alumno



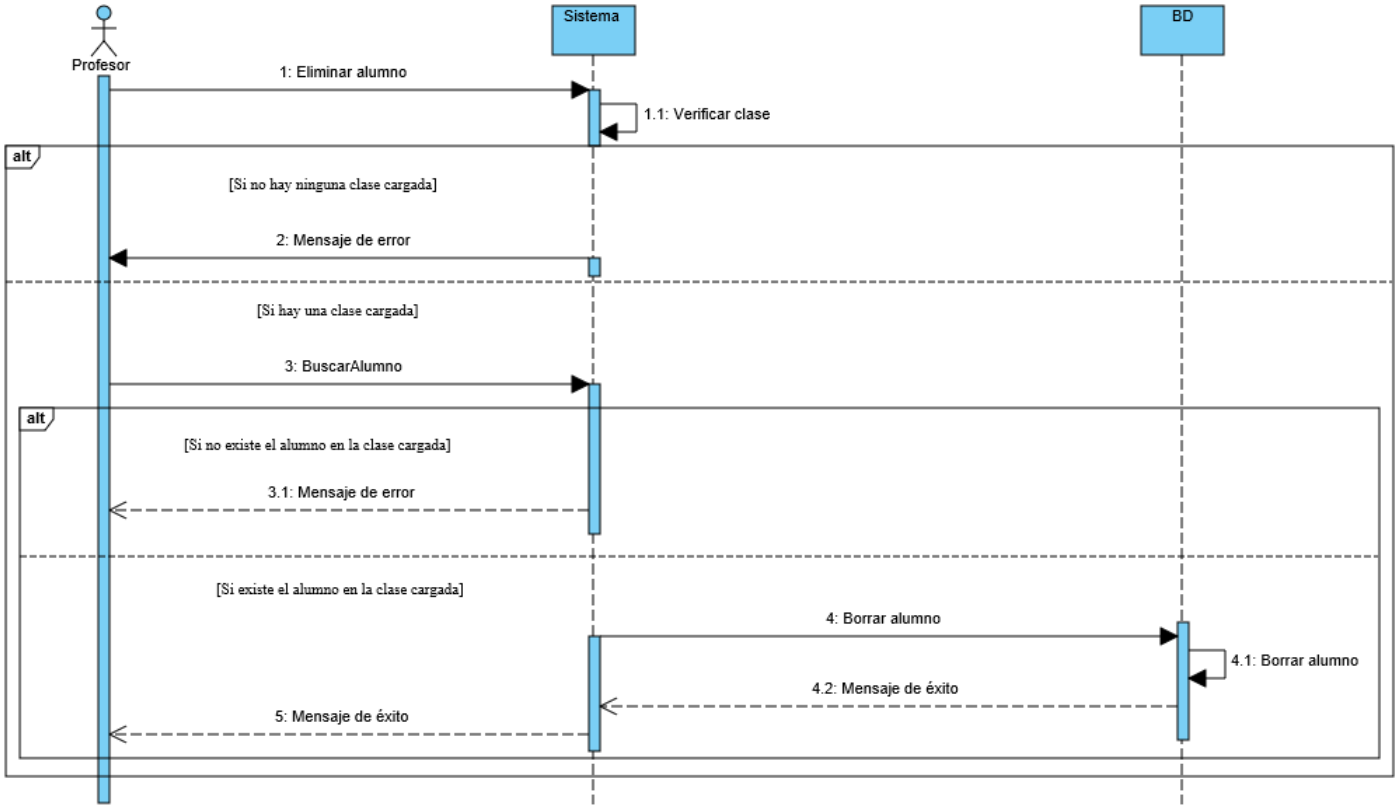
Buscar Alumno



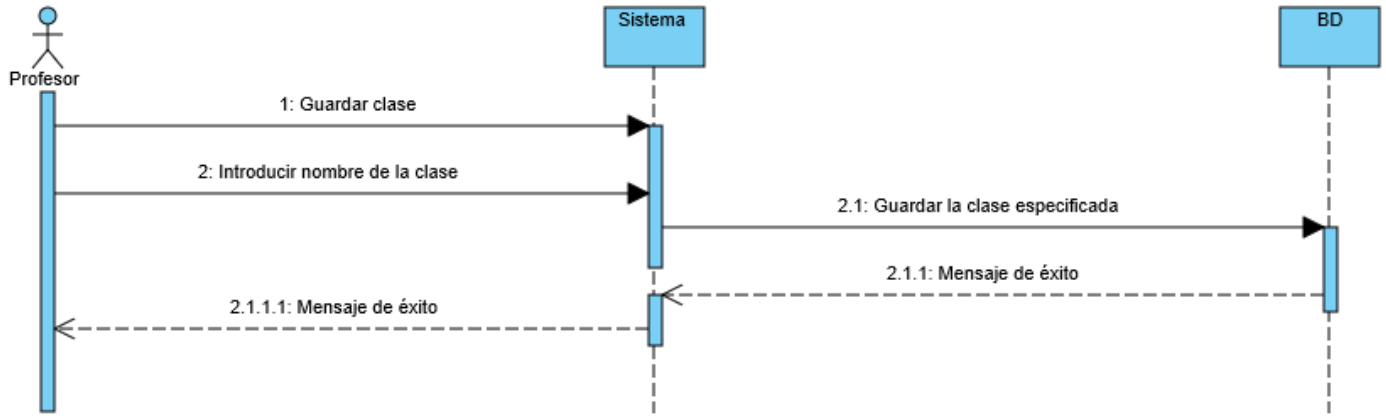
Cargar Clase



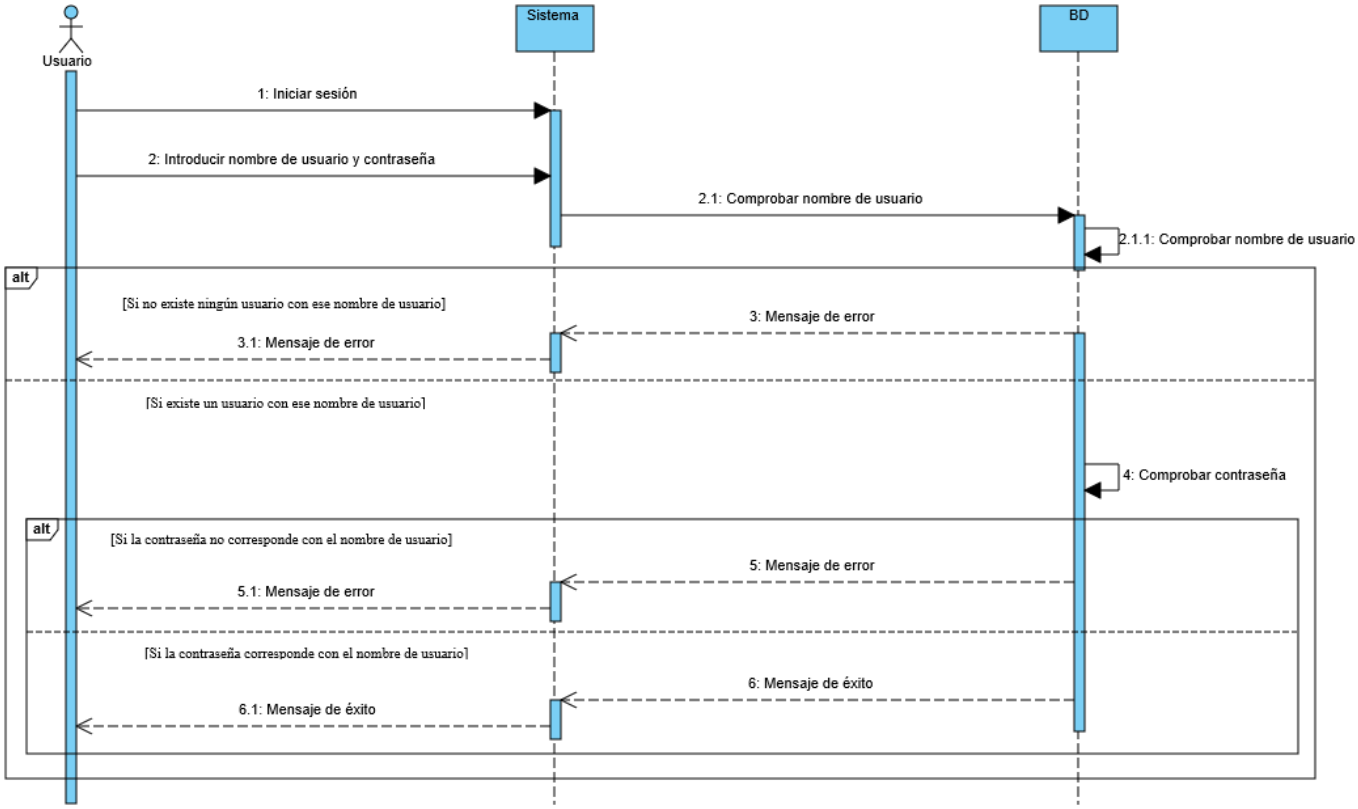
Eliminar Alumno



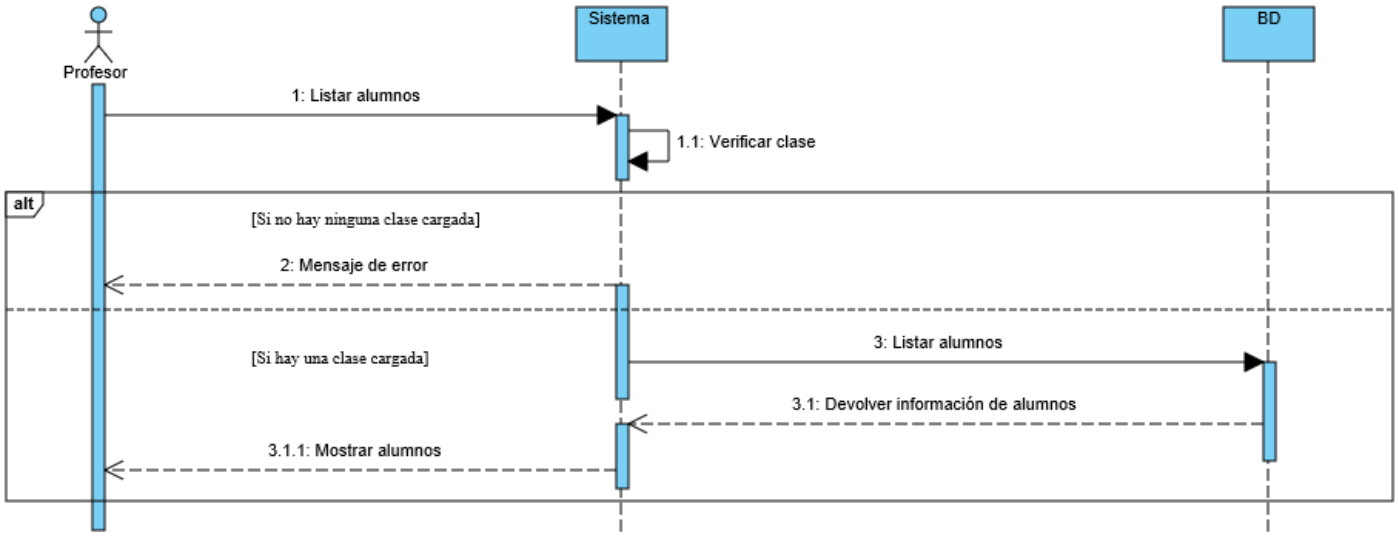
Guardar Clase



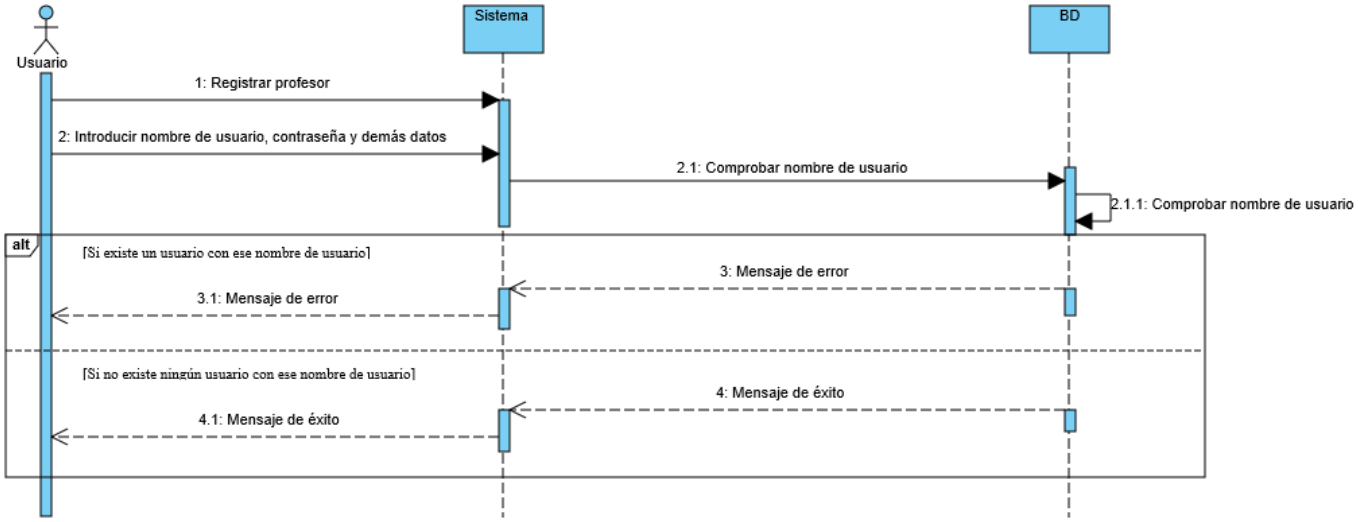
Iniciar Sesión



Listar Alumno



Registrar Profesor



Una vez finalizado con los diagramas y detallado la creación de las distintas funcionalidades del sistema, podemos proceder a la implementación del programa. Para ello vamos a usar el método **SCRUM** descrito al principio de este documento para organizar el trabajo del equipo de programación y así lograr la máxima eficiencia y eficacia posible.

Procedimiento SCRUM

Para poder usar este procedimiento, lo primero que hemos realizado es el denominado **Product Backlog**, documento que recoge todos los requisitos funcionales y no funcionales del sistema.

Product Backlog

ID: 8 Sistema GNU/Linux

(ANVERSO)

El programa debe funcionar en Linux.

Prioridad: 1

(REVERSO)

- El programa debe funcionar en cualquier sistema GNU/Linux.

ID: 3 Cargar clases

(ANVERSO)

Quiero poder utilizar cuando quiera la información de las clases.

Prioridad: 2

(REVERSO)

- Quiero poder cargar la información de las distintas clases.

ID: 11 Ficheros binarios

(ANVERSO)

La información de los alumnos se tendrá que almacenar en ficheros binarios.

Prioridad: 2

(REVERSO)

- Los ficheros en los que se guarde la información de los alumnos deberán ser binarios.

ID: 1 Almacenar usuario
(ANVERSO)

Quiero tener almacenada la información de los alumnos.

Prioridad: 2

(REVERSO)

- Quiero tener almacenados todos los datos relevantes sobre los alumnos.
- Se debe distinguir si un alumno es lider de grupo o no.

ID: 2 Guardar clases
(ANVERSO)

Quiero poder guardar la información de las distintas clases.

Prioridad: 2

(REVERSO)

- Preferiría tener almacenada la información de las distintas clases en ficheros.

ID: 10 Máximo 150 alumnos
(ANVERSO)

No podrán haber más de 150 alumnos por clase.

Prioridad: 2

(REVERSO)

- El máximo de alumnos que se pueden almacenar por cada clase será de 150.

ID: 9 Campos *Equipo* y *Líder* no obligatorios

(ANVERSO)

Quiero que la información relacionada con el equipo al que pertenece un alumno y si es líder o no de dicho equipo sean opcionales.

Prioridad: 2

(REVERSO)

- Los campos *Equipo* y *Líder* de la información de los alumnos no serán obligatorios.

ID: 4 Buscar alumno

(ANVERSO)

Quiero tener la posibilidad de buscar a un alumno por su DNI, nombre, apellido, la clase o por el grupo al que pertenece

Prioridad: 3

(REVERSO)

- Debo poder buscar alumnos usando como posibles parámetros:
  - DNI
  - Grupo al que pertenece

ID: 6 Eliminar alumno

(ANVERSO)

Quiero poder eliminar a un alumno.

Prioridad: 4

(REVERSO)

- Quiero poder borrar la información de un alumno y eliminar sus registros.

ID: 5 Actualizar alumno
(ANVERSO)

Quiero tener la posibilidad de actualizar la información de un alumno.

Prioridad: 4

(REVERSO)

- Quiero poder modificar cualquier tipo de información referente a cada alumno.

ID: 7 Listar alumnos
(ANVERSO)

Debe haber una opción para ver todos los alumnos de una clase

Prioridad: 5

(REVERSO)

- Quiero tener una opción que liste a todos los alumnos de una clase.
- Los alumnos que sean líderes de algún grupo deben estar marcados de alguna forma especial.

Una vez recogidos todos los requisitos, tanto funcionales como no funcionales, en el Product Backlog, podemos proceder a la programación del sistema. Para ello, dividiremos el trabajo según funciones. Cada semana se intentará hacer un número específico de funciones dentro de las horas establecidas. Este procedimiento se llama **Sprint** y para organizarlo se crea una tabla que indica las funciones a crear durante el Sprint.

En nuestro proyecto, hemos tenido dos Sprints, uno por semana, que ha dado lugar a las siguientes tablas:

Sprint Backlog #1

Historia de Usuario (ID)	Enunciado de la historia	Alias	Estado	Esfuerzo	Sprint
7	Como profesor quiero ver la información de un alumno en concreto.	Listar un Alumno	Finalizado	2	1
4	Como profesor quiero poder buscar las distintas clases que hay.	Buscar un alumno	Finalizado	1	1
6	Como profesor quiero poder eliminar un alumno de una clase .	Eliminar un alumno	En proceso	1	1
1	Como profesor quiero poder añadir un alumno a una clase.	Almacenar un alumno	Finalizado	3	1

Developer/Team	Horas/Sprint
3	10

En Progreso	Estado
Crear la base de datos para que el profesor pueda almacenar los alumnos de una clase en una clase (fichero binario)	Finalizado
Crear la interfaz de usuario para que el profesor pueda acceder a las distintas opciones que ofrece el sistema	Finalizado
Crear las distintas funciones para cada una de las opciones del sistema para este sprint	Finalizado

Sprint Backlog #2

Historia de Usuario (ID)	Enunciado de la historia	Alias	Estado	Esfuerzo	Sprint
2	Como profesor quiero poder guardar una clase en un fichero binario junto con la información de dicha clase	Guardar Clase	Finalizado	4	2
5	Como profesor quiero poder actualizar los datos de un alumno si en algún momento cambian	Actualizar Alumno	Finalizado	2	2
6	Como profesor quiero poder eliminar un alumno de una clase .	Eliminar un alumno	Finalizado	2	2
7	Como profesor quiero ver todos los alumnos de una clase por pantalla.	Listar Alumnos	Finalizado	2	2

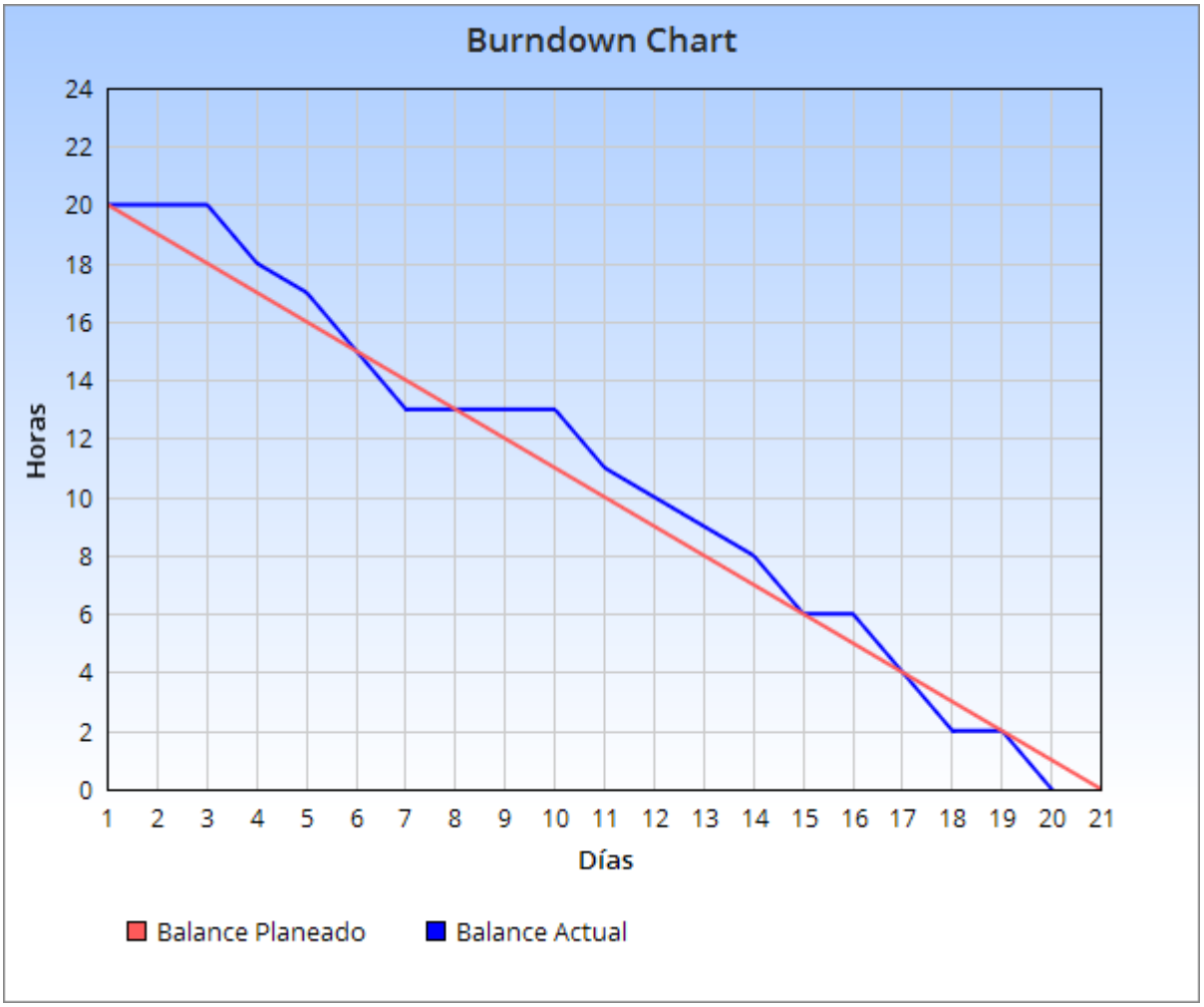
Developer/Team	Horas/Sprint
3	10

En Progreso	Estado
Crear la base de datos para que el profesor pueda almacenar los alumnos de una clase en una clase (fichero binario) y la información necesaria	Finalizado
Crear la interfaz de usuario para que el profesor pueda acceder a las distintas opciones que ofrece el sistema	Finalizado
Crear las distintas funciones para cada una de las opciones del sistema para este sprint	Finalizado

Durante esas dos semanas, hay que tener un seguimiento del trabajo del equipo y de las horas trabajadas. Para ello, se crea un gráfico denominado **Burndown Chart**. Este gráfico recoge las horas planeadas para el proyecto y las horas actuales trabajadas cada semana.

A continuación, veremos el Burndown Chart que hemos creado para el seguimiento de nuestro trabajo.

Burndown Chart



Matriz de Trazabilidad de Requisitos

Para finalizar el procedimiento, debemos verificar que, de forma correcta, cada caso de uso debe quedar cubierto por un requisito funcional. Para realizar esto, debemos crear una matriz llamada **matriz de trazabilidad de requisitos** donde cubriremos cada caso de uso con un requisito funcional. En este paso nos daremos cuenta si tenemos todo cubierto o hay que añadir algún otro requisito funcional.

Requisitos /Casos de usos	Caso 1	Caso 2	Caso 3	Caso 4	Caso 5	Caso 6	Caso 7	Caso 8	Caso 9
Req 1	X								
Req 2		X							
Req 3			X						
Req 4				X					
Req 5					X				
Req 6						X			
Req 7							X		
Req 8								X	
Req 9									X

Ahora que ya hemos comprobado el correcto desarrollo de los casos de uso en relación con los requisitos funcionales, podemos pasar al **manual**.

Manual

El manual recoge las instrucciones técnicas de uso del sistema, entre las cuales destacan el uso de las opciones y funcionalidades que proporciona el sistema además de los errores y fallos que se pueden dar a la hora de usarlo.

Sistema de Administración de Clases y Alumnado

Manual de usuario

El sistema ACA es un sistema desarrollado para uso educativo, principalmente por profesores, que permite la organización del alumnado y de las clases de cada profesor. Este documento cubrirá todas las funciones del sistema ACA, su correcto uso y las opciones disponibles para la realización de dichas tareas.

El sistema está escrito en el lenguaje de programación C++ y es completamente compatible con cualquier sistema operativo basado en GNU/Linux.

Registro de los usuarios del sistema

El sistema permite la creación de más de una cuenta de usuario para los distintos profesores existentes o más de una cuenta para un mismo profesor.

Inicio de sesión

El usuario deberá introducir el usuario y la contraseña indicadas en el registro para poder acceder al sistema. Si el usuario introduce una contraseña erronea se le imprimirá un mensaje por pantalla diciendole que la contraseña no es correcta. En dicho caso, para recuperar su contraseña, debe contactar con un administrador, indicandole el nombre de cuenta al administrador.

Si por cualquier razón no puede recuperar su cuenta, siempre puede crear otra cuenta.

Si el usuario se ha autentificado correctamente, podrá acceder al menú principal del sistema y a sus bases de datos (clases y alumnos).

Si el usuario no está registrado en el sistema, se le imprimirá un mensaje por pantalla indicandole que primero debe registrarse en el sistema y volverá a la pantalla de acreditación o registro.

Menú principal

Una vez introducido las credenciales y accedido al sistema, el usuario verá el menú principal del sistema, donde podrá ver las distintas opciones que le permite el sistema.

1. Asignar clase

Esta es la opción principal para poder crear clases o cambiar entre las clases bajo las cuales se agrupan los alumnos que pertenezcan a una clase.

Las clases no podrán tener más de 150 alumnos.

Si intenta introducir más de 150 alumnos en una clase, se le indicará un error y volverá al menú principal.

Para crear una nueva clase simplemente usamos esta opción y asignamos un nombre a la clase.

Para cambiar a otra clase mientras ya hay una cargada, volvemos a usar esta opción e indicamos el nombre de la otra clase. Una vez hayamos hecho esto, volveremos al menú principal y podremos cargar la otra clase.

Atención: Es recomendable guardar los cambios realizados en una clase antes de cambiar a otra.

2. Cargar clase

Esta opción se usa en conjunto con la primera opción (Asignar clase). Una vez indicado el nombre de la clase a cargar en la opción "Asignar clase", podremos usar esta opcion para realizar el cargado de la clase.

3. Guardar clase

Esta opción nos permite guardar una clase después de haberla creado (para crear una clase, volver al punto 1 "Asignar clase"). Guardará la clase y su información (alumnos e información de esos alumnos) en un fichero binario.

Si la clase no ha podido ser guardada nos dará un error y volverá al menú principal.

4. Mostrar clase

Esta opción nos permite ver todos los alumnos de una clase, además nos mostrará toda la información de los alumnos.

Si la clase indicada es errónea o no existe, se indicará un mensaje por pantalla con el error y se volverá al menú principal.

5. Borrar clase

Esta opción nos permite eliminar una clase y toda la información adjunta a ella. Para borrar una clase, primero tenemos que cargarla en el sistema.

Si no se ha podido eliminar una clase, nos imprimirá un error por pantalla y se volverá al menú principal.

6. Buscar alumno

Esta opción permitirá buscar un alumno por su DNI, nombre, apellido o grupo al que pertenece. Se mostrará por pantalla la información de dicho alumno: su nombre y apellido, DNI y grupo al que pertenece y si es lider de grupo o no

Si la información indicada para realizar la búsqueda no existe o es erróneo, se indicará por pantalla y será devuelto al menú principal.

7. Insertar alumno

Esta opción permitirá almacenar un alumno en el sistema, con la información de ese alumno, como DNI, nombre y apellido. Los campos de nombre, apellidos, DNI y grupo al que pertenecen serán obligatorios para poder almacenar el alumno en el sistema.

Si no se puede almacenar un alumno por por un fallo del usuario o del sistema, se indicará el error y se volverá al menú principal.

8. Actualizar alumno

Esta opción nos permite modificar la información de un alumno previamente buscado por DNI.

Podremos modificar cualquier campo relacionado con ese alumno.

Hay que rellenar todos los campos para que la modificación se haga correctamente.

9. Eliminar alumno

Esta opción permite borrar a un alumno y los registros e información de dicho alumno. Para poder hacerlo, buscaremos al alumno por su DNI y activaremos la opción "Eliminar alumno".

Dicho alumno debe existir en el sistema para poder ser eliminado, de lo contrario, se indicará un error por pantalla y se volverá al menú principal.

Base de datos

Las clases y los usuarios se guardarán en ficheros binarios en la carpeta donde se ubica el sistema.

Las clases serán guardadas bajo el nombre de la clase y los profesores registrados bajo un fichero llamado "profesores"