Universidad Nacional Autónoma de México

Programación Orientada a Objetos

Diseño del software

V0.5

Fecha: 15/12/2022

Integrantes

- **Aranda Solis Ricardo**
- **García González Alejandro**
- **Rodríguez Rodríguez Arturo**
- **Sosa Cortez Misael Ivan**

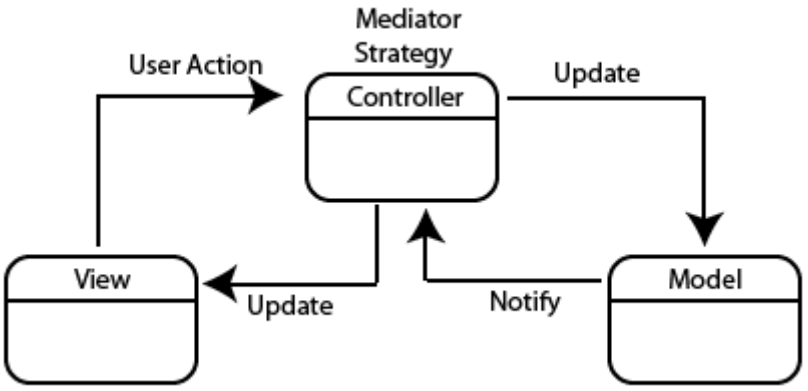
Contenido

Histórico de Cambios

Versión	Descripción	Responsable de Actualización	Fecha de actualización
V0.1	Creación del documento (secciones 1 y 2)	Aranda Solis Ricardo	01/11/22
V0.2	Actualización del documento (secciones 3, 4 y 5.3)	Rodriguez Rodriguez Arturo	01/11/22
V0.3	Actualización del documento(secciones 4 y 5.1-5.2)	García González Alejandro	02/11/22
V0.4	Modificación del documento	Aranda Solís Ricardo	02/11/22
V0.5	Modificación del documento	García González Alejandro	04/01/23

1. Descripción de la Arquitectura

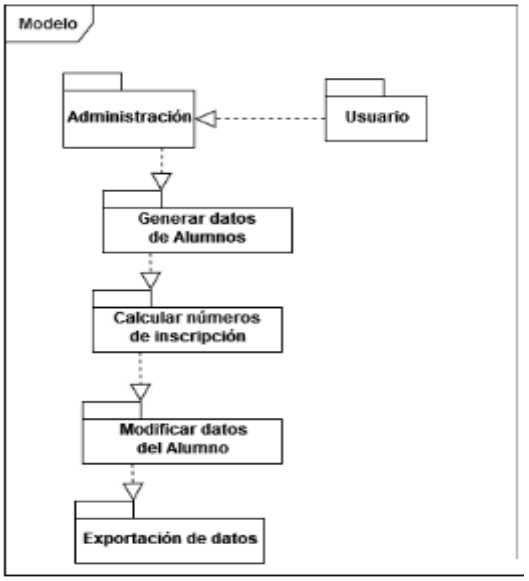
En el sistema se requiere priorizar las características de facilidad de comprensión, modularidad y garantizar la portabilidad para que los usuarios puedan acceder desde cualquier dispositivo de diferentes plataformas. Tomando en cuenta principalmente estas características, el sistema será implementado utilizando una arquitectura basada en el patrón de arquitectura *ModelViewController.*



1.1. Paquetes de la arquitectura

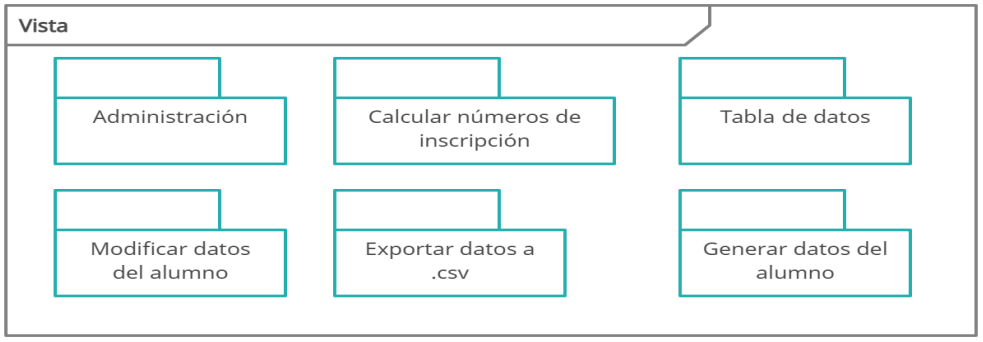
1.1.1. Modelo (Model)

Es la representación de la información, con la que nosotros en la implementación del sistema de gestión de datos de los alumnos podemos manejar la información de los alumnos, en este caso particular, también de la generación de los datos, por lo que este modelo debe de cumplir con ambos enfoques como lo es la parte de la administración escolar, como lo es de los datos de consulta por parte del alumno.



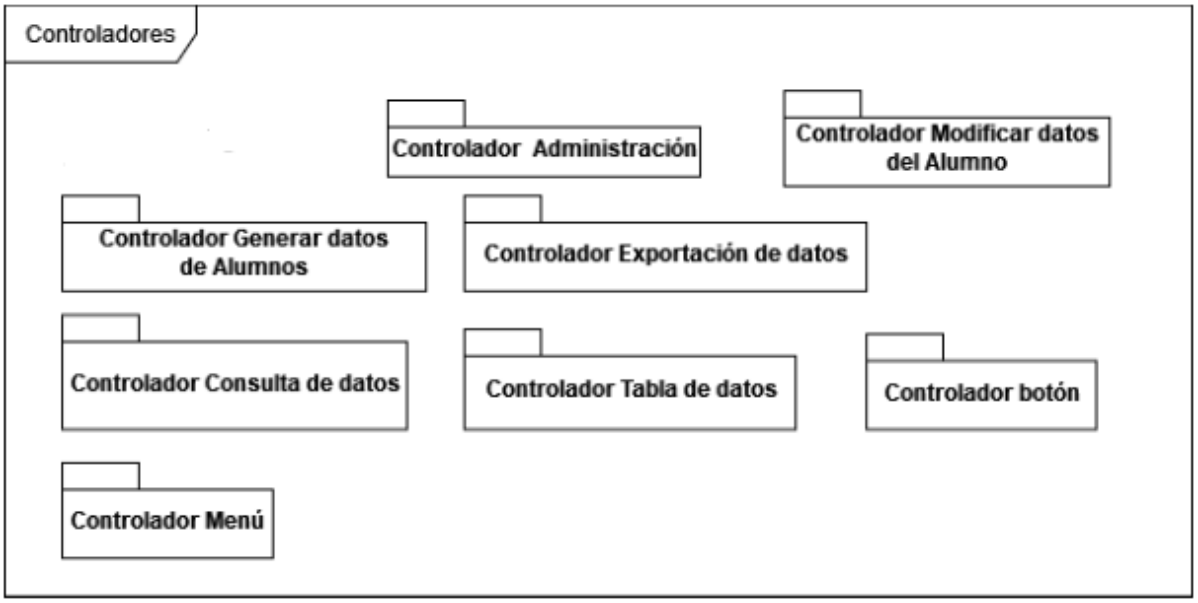
1.1.2. Vista (View)

Presenta el 'modelo' (información y lógica) en un formato adecuado para interactuar (usualmente la interfaz de usuario), por tanto requiere de dicho 'modelo' la información que debe representar como salida.



1.1.3. Controlador (Controller)

Responde a eventos (usualmente acciones del usuario desde la vista) e invoca peticiones al 'modelo' cuando se hace alguna solicitud sobre algunas de las opciones del modelo. También puede enviar comandos a su 'vista' asociada si se solicita un cambio en la forma en que se presenta el 'modelo', por tanto se podría decir que el 'controlador' hace de intermediario entre la 'vista' y el 'modelo'.



1.2. Definición del ambiente de implementación

Concepto	Herramienta	Versión
Lenguaje de programación	Java	18
Diagramador UML	Draw.io, creately	N/A
Manejador archivos exportados	Excel	16.0.15028.20160
IDE	Netbeans	14
Control de versiones	Github	9

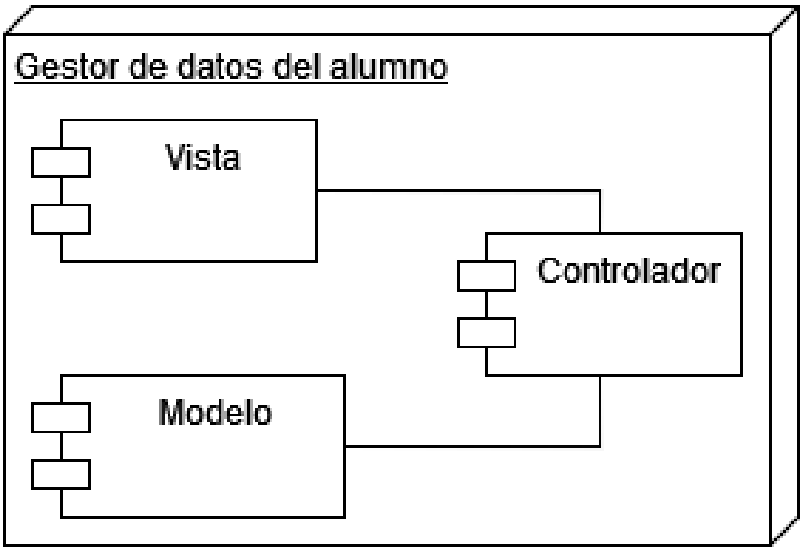
2. Vista de despliegue

2.1. Descripción

El sistema tendrá 1 nodos principal:

- Programa de gestión escolar: ejecutable (ya sea en .jar y .exe) que el interesado deberá de correrlo en su computadora, utilizando el modelo MVC para implementar los modelos necesarios para su correcto funcionamiento, incluyendo ambos usuarios (Administrador y alumno), además que el modelo pueda leer una base de datos local (excel) .

2.2. Modelo de despliegue



3. Vista lógica

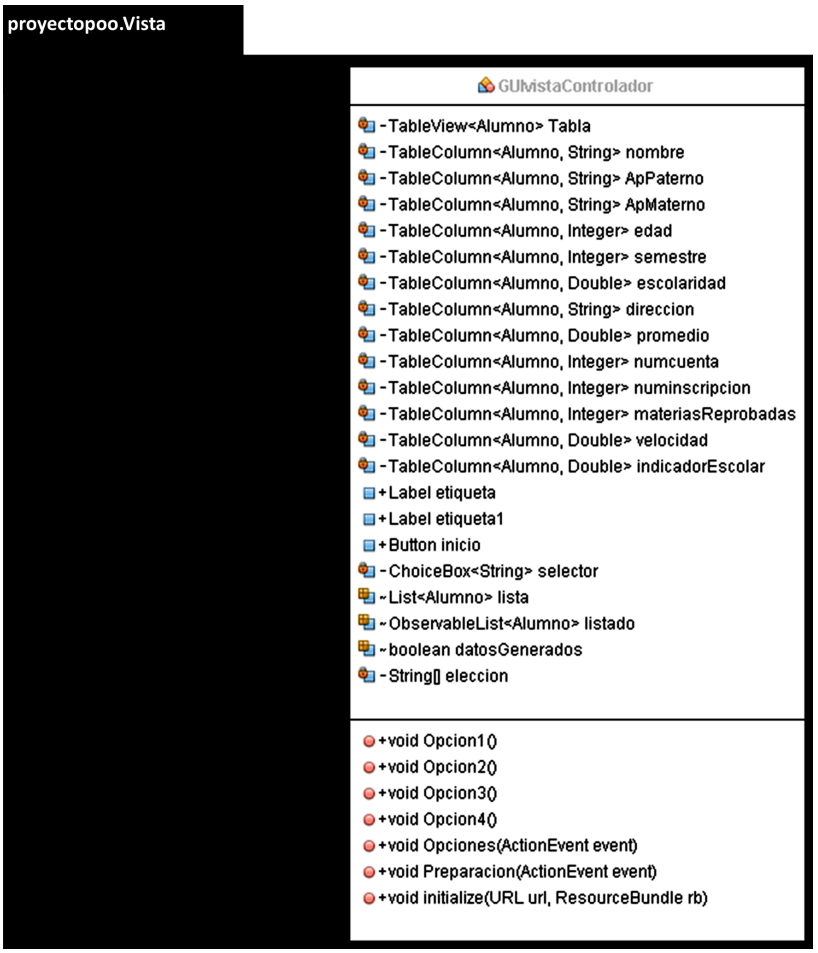
3.1. Descripción

Esta sección describe de manera general la descomposición del modelo mediante jerarquía de paquetes.

3.2. Identificación de las clases

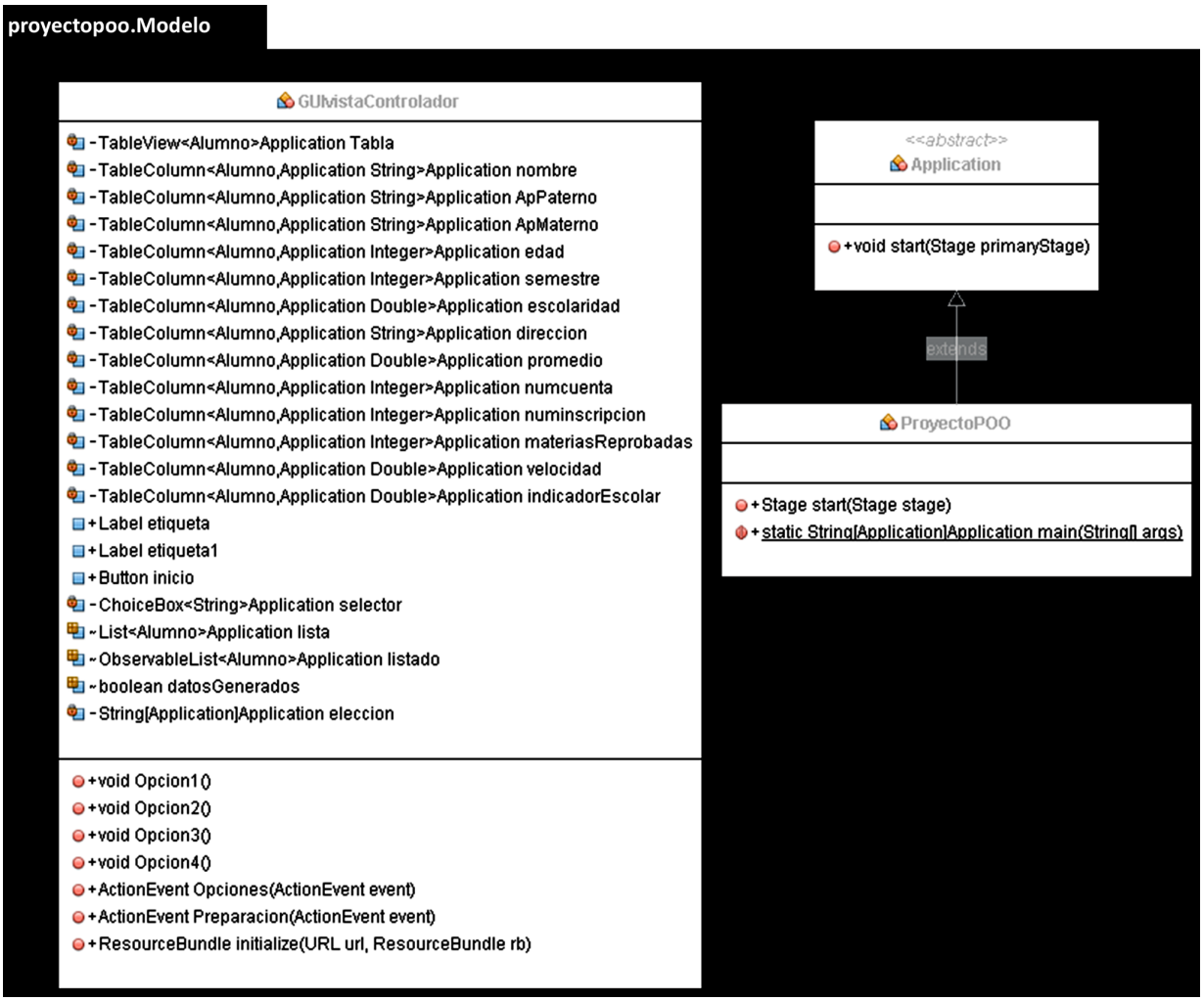
Esta sección contendrá los diagramas que describen el comportamiento estático del sistema y los tipos de relaciones existentes entre las clases. Se generará un diagrama de clases por cada paquete que contenga el sistema.

3.2.1. Clases de vista

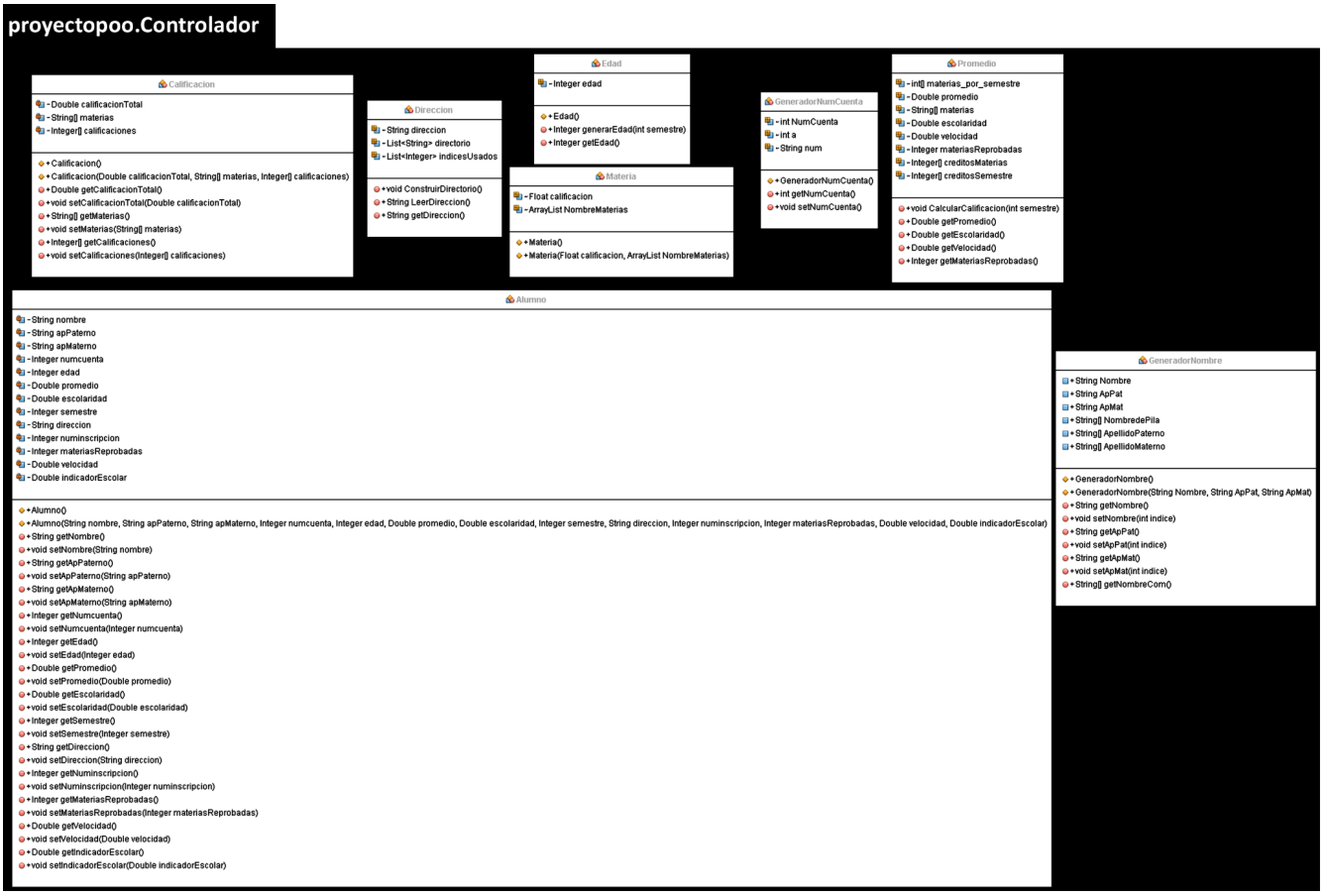


Es necesario recalcar el uso de *FXL* en la parte de vista, ya que *FXML* es un lenguaje de marcado basado en *XML* y programable para definir interfaces de usuario (UI) en *JavaFX*, una plataforma de software para crear y entregar aplicaciones de escritorio con gráficos y multimedia enriquecidos. FXML le permite definir el diseño y el comportamiento de su IU de manera declarativa, separando el diseño de la IU de la lógica de la aplicación. Esto puede facilitar el mantenimiento y la actualización de su interfaz de usuario, además de permitir que los diseñadores trabajen en el diseño de la interfaz de usuario sin necesidad de conocimientos de programación. *FXML* se puede usar junto con el lenguaje de programación Java para crear aplicaciones *JavaFX*, o se puede usar con otros lenguajes de programación que tengan un enlace *JavaFX*, como Scala o Groovy.

3.2.2. Clases de modelo



3.2.3. Clases de controlador



4. Vista de datos

4.1. Descripción

Para hacer persistente la información del sistema de administración de alumnos se utilizará una base de datos elaborada en una vista de tabla, que es un conjunto de datos pertenecientes a un mismo contexto y almacenados sistemáticamente para su posterior uso alrededor de todo el ciclo de vida del programa y llevar a cabo la gestión escolar de los alumnos .

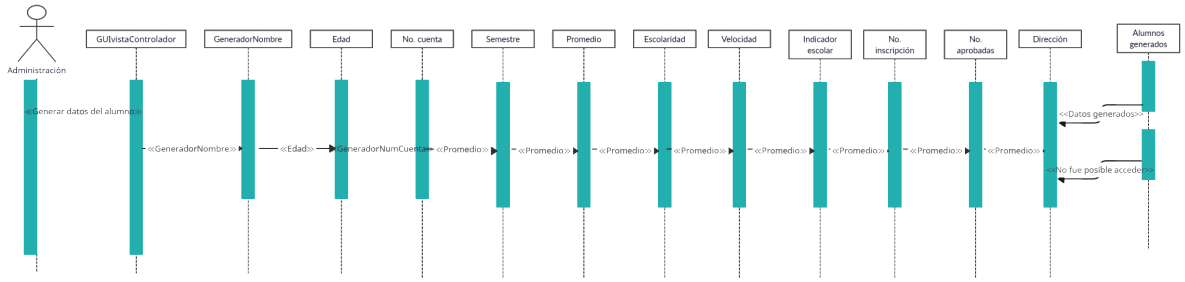
5. Vista dinámica

5.1. Descripción

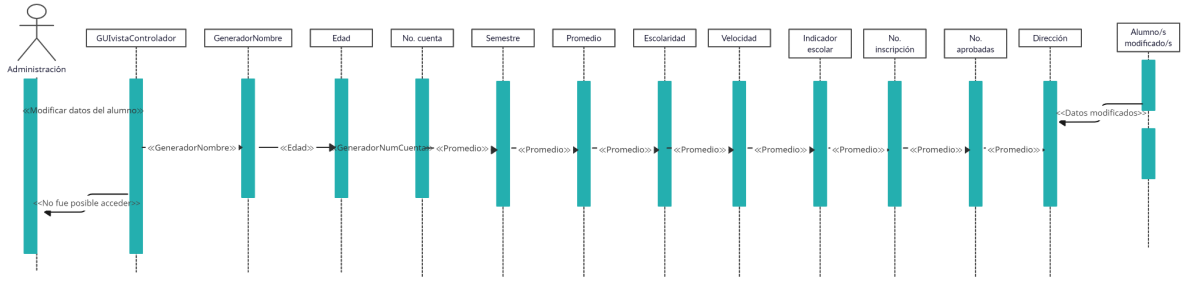
Cada caso de uso es representado con un diagrama de secuencia, éste muestra cómo se comportan los objetos entre ellos a través del tiempo. Los diagramas de secuencia representarán el comportamiento dinámico de los casos de uso. El diagrama de estados, indicará cómo navega el usuario(Administrador o Alumno) a través de las interfaces del sistema. Cada diagrama tendrá un estado inicial que se identificará con una circunferencia negra que rodea un punto negro, los estados serán representados por un óvalo con el nombre de la interfaz en la que se encuentra el usuario al realizar alguna acción, las acciones estarán determinadas en el documento por flechas con un estado origen, una acción y el estado destino.

5.2. Diagramas de secuencia

5.2.1. Generar alumno



5.2.2. Modificar Alumno



5.3. Diagramas de navegación