|  |
| --- |
| Aplicación de escritorio para la gestión de alumnos y certificados |
| Memoria |
| [Escriba aquí una descripción breve del documento. Normalmente, una descripción breve es un resumen corto del contenido del documento. Escriba aquí una descripción breve del documento. Normalmente, una descripción breve es un resumen corto del contenido del documento.] |

17-2-2016

# https://riunet.upv.es/bitstream/handle/10251/38745/Memoria.pdf?sequence=1

# Resumen

Desarrollo de una aplicación de escritorio a medida para la empresa ADESOS para gestionar los certificados y los alumnos de la empresa y facilitarles acceder a la información de sus certificados y los alumnos que los reciben, también poder obtener pdf´s con la información de los alumnos, los certificados que actualmente se imparten y los que se han impartido, las notas de los alumnos y las estadísticas que puedan surgir del cruce de datos.

# Análisis

## Objetivos

El objetivo de esta aplicación es construir una interfaz gráfica para gestionar los certificados que se imparten en ADESOS, facilitar la gestión de los certificados como es el papeleo de la formación de los grupos, la gestión de las pruebas a realizar, además de cálculo de las notas, así como de las notas finales.

Los **pasos del proyecto** son los siguientes:

* Negociación con el cliente de las características que tendrá la aplicación para generar la hoja de requerimientos.
* Elección de las tecnologías que vamos a usar para la realización del proyecto.
* Creación de la versión alfa, que será mostrada al cliente.
* Modificación del proyecto con las aportaciones del cliente
* Creación de la versión beta e inicio fase de testeo:
  + Testear la aplicación y obtener información
  + Entregar al cliente para que pueda testearla y dar información
  + Discusión con empresa y cliente sobre puntos críticos, mejoras importantes y mejoras para un futuro teniendo en cuenta la información obtenida.
* Creación de la versión final del producto.

(Hablar sobre si le hacemos un manual de instrucciones)

Las **tareas de desarrollo**, a grandes rasgos, son las siguientes:

* Creación del LOGIN principal para acceder a la aplicación.
* Creación del formulario con sus respectivos botones de acceso a cada una de las áreas, según seas administrador o usuario normal, a saber:
  + Acceso a certificados.
  + Acceso a alumnos.
  + Creación de usuarios (opción solo accesible desde el usuario que sea de tipo administrador)
  + Gestión de notas de los alumnos en dichos certificados.
* Creación de formulario donde se tenga visible la lista de los certificados y cuando cliquees en él muestre la información de cada certificado y de sus módulos asociados, además de poder introducir y modificar.
* Creación de formulario donde se tenga visible la lista de los alumnos y cuando cliquees en él muestre la información de dichos alumnos y los estudios que posean, además de poder introducir y modificar.
* Creación de formulario para la gestión de notas en el que se pueda visualizar las notas por modulo realizado por cada alumno, además de poder introducir y modificar.
* Creación de formulario para la gestión de usuarios en el que se pueda visualizar todos los usuarios, pudiendo eliminar, insertar y modificar usuarios además de poder indicarle de que tipo de usuario se trata.

## Requisitos funcionales

Cuando se nos encargó el proyecto, lo primero que fue necesario fue reunirnos con los encargados de la empresa para que llegáramos a redactar una lista con los requisitos funcionales y no funcionales del sistema, es decir, los comportamientos y requisitos del sistema.

Los requisitos funcionales definen una función concreta del sistema y/o de uno de sus componentes. Para establecer un requisito funcional es necesario establecer las entradas que tiene dicha funcionalidad, comportamiento y salidas que genera.

## Requisitos Funcionales de la aplicación

|  |  |
| --- | --- |
| **Código del Requisito** | **Nombre del requisito** |
| LoginGeneral | Autenticación de usuario |
| **Descripción o características** | |
| Antes de poder usar la aplicación el usuario debe autenticarse en la aplicación con un nombre de usuario y contraseña. | |

|  |  |
| --- | --- |
| **Código del Requisito** | **Nombre del requisito** |
| InterfazPrincipalAdministrador | Interfaz principal-ADMINISTRADOR |
| **Descripción o características** | |
| El administrador se introducirá con su login y tendrá acceso a una interfaz gráfica donde tendrá acceso a distintos botones donde podrá gestionar notas, alumnos y certificados, además de poder crear usuarios (opción que solo puede realizar el administrador). | |

|  |  |
| --- | --- |
| **Código del Requisito** | **Nombre del requisito** |
| LoginAñadir | Añadir usuarios que puedan autenticarse |
| **Descripción o características** | |
| Los administradores pueden añadir otros usuarios y establecer su nivel (ROL) y contraseña. | |

|  |  |
| --- | --- |
| **Código del Requisito** | **Nombre del requisito** |
| InterfazPrincipalUsuario | Interfaz principal-USUARIO |
| **Descripción o características** | |
| El usuario se introducirá con su login y tendrá acceso a una interfaz gráfica donde tendrá acceso a distintos botones donde podrá gestionar notas, alumnos y certificados. | |

|  |  |
| --- | --- |
| **Código del Requisito** | **Nombre del requisito** |
| GestionNotas | Gestión de las notas obtenidas por los alumnos |
| **Descripción o características** | |
| El usuario o administrador podrán visualizar, introducir o modificar desde esta pantalla las notas obtenidas por los alumnos en cada módulo para tratar estos datos como ellos consideren oportuno. | |

|  |  |
| --- | --- |
| **Código del Requisito** | **Nombre del requisito** |
| GestionAlumnos | Gestión de los alumnos matriculados en los módulos |
| **Descripción o características** | |
| El usuario o administrador podrán visualizar, introducir o modificar desde esta pantalla los datos de los alumnos matriculados en los distintos módulos que componen los certificados para tratar estos datos como ellos consideren oportuno. | |

|  |  |
| --- | --- |
| **Código del Requisito** | **Nombre del requisito** |
| GestionCertificados | Gestión de los certificados que imparte esta entidad |
| **Descripción o características** | |
| El usuario o administrador podrán visualizar, introducir o modificar desde esta pantalla los datos de los distintos módulos que componen los certificados para tratar estos datos como ellos consideren oportuno. | |

**¿ 2.1.2 Requisitos Funcionales de la interfaz de usuario?**

## Diagrama de casos de uso

naziha

## Requisitos no funcionales

Esa aplicación va a constar de una base de datos hecha en Oracle, una aplicación de escritorio realizada en Java FX y la integración de los dos medios. La aplicación va a poder correr en varios ordenadores, así que la base de datos tiene que estar en todos los ordenadores, o en un ordenador que sea visible por todos los demás.

La primera opción de tener la base de datos en cada ordenador no es viable porque tenemos que tenerla sincronizada. La segunda opción es la más viable porque si la base de datos la tenemos bien configurada y controlada, los cambios realizados por el usuario en la aplicación tienen que cumplir el **ACID**(Atomicidad, Consistencia, Aislamiento y Durabilidad) . Pero nos surge la necesidad de tener un ordenador centralizado con la base de datos integrada y la aplicación hecha, lo que nos lleva a ver las características o requisitos que necesita el equipo.

* Una dirección IP estática en el equipo en el que va a instalar Oracle 11g.
* Un mínimo de 125 MB de espacio libre en el disco para archivos temporales de instalación.
* 1 GB como mínimo de memoria RAM física para Oracle 11g Standard Edition (1,3 GB de RAM para Oracle 11g Enterprise Edition) y 2 GB de memoria virtual.
* 4,55 GB como mínimo de espacio libre en el disco para la instalación básica y 4,92 GB de espacio libre en el disco para la instalación avanzada.
* Los discos de instalación de Oracle 11g.
* Las licencias y contratos de asistencia apropiados de Oracle.

El sistema operativo que puede correr la aplicación no depende de la versión del sistema operativo sino que depende que tengamos la máquina virtual de java, además de tener la última versión de la máquina virtual de java lo más actualizada posible.

* La aplicación debe ser accesible desde cualquier ordenador autorizado.
* Se autenticará al usuario mediante usuario y contraseña a indicar por el administrador.
* En todo momento se mantendrá control de quien hace las cosas para futura referencia en caso de necesitarlo por temas de seguridad.

# Diseño

Explicar MVC y detallar cada una de las capas: modelo, vista y los controladores.

Se utilizará el lenguaje de modelos UML (Unified Modeling Language) para facilitar la organización y estructuración de la aplicación mediante un diagrama de interfaz, el cual mostrará los distintos menús y cómo se relacionan entre sí, y nos ayudará a tener una visión global de la aplicación. Es un diagrama de flujo, que nos sirve para ver los distintos pasos que seguiría un usuario desde que accede al menú principal hasta que sale de la aplicación.

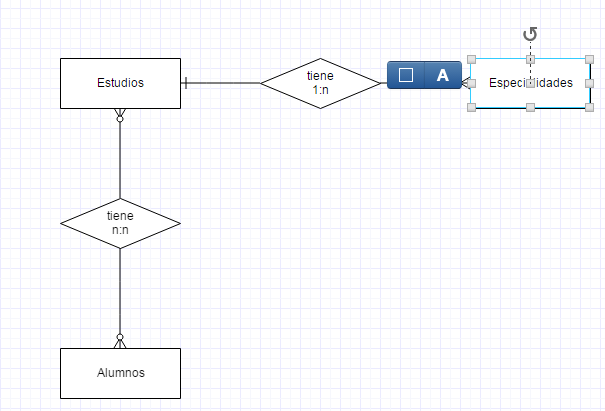
En nuestro diagrama de interfaz podemos apreciar que el grueso de la aplicación sucede en los formularios a los que se accede a través del menú principal,

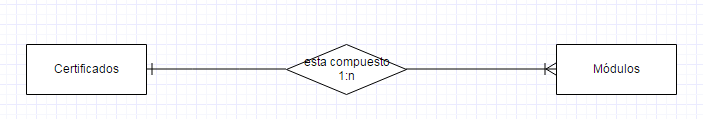
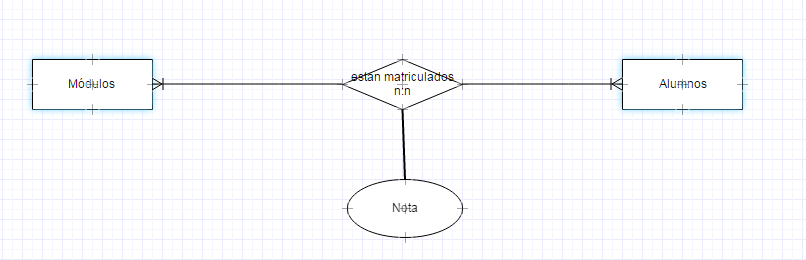
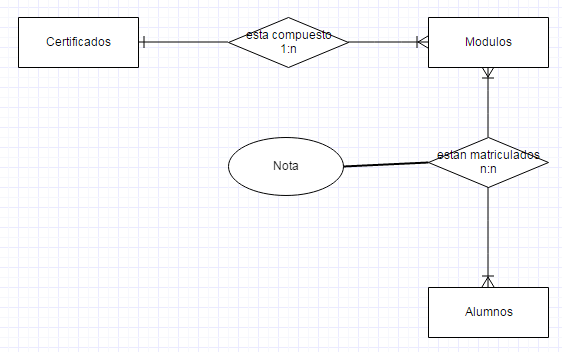
Explicación del funcionamiento de la base de datos propuesta

## Análisis del modelo entidad relación

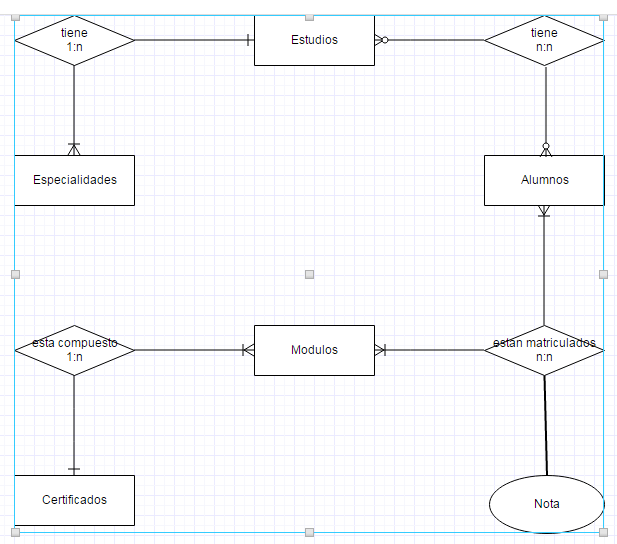
Según la información propuesta por el cliente y los objeto que nos ha propuesto tratar.

* **En el tema de los alumnos hemos dicho que nos interesa recoger datos de sus estudios teniendo en cuenta su especialidad.**Hemos contemplado que un alumno puede tener ninguno o varios estudios mientras que un estudio puede haberlo realizado ninguno o varios alumnos.  
  Así que:  
    
  A su vez hemos contemplado que una especialidad puede ser de un estudio y solo de uno mientras que un estudio puede tener entre una o varias especialidades, Así que:  
    
  En suma el resultado del análisis del tema de alumnos es:

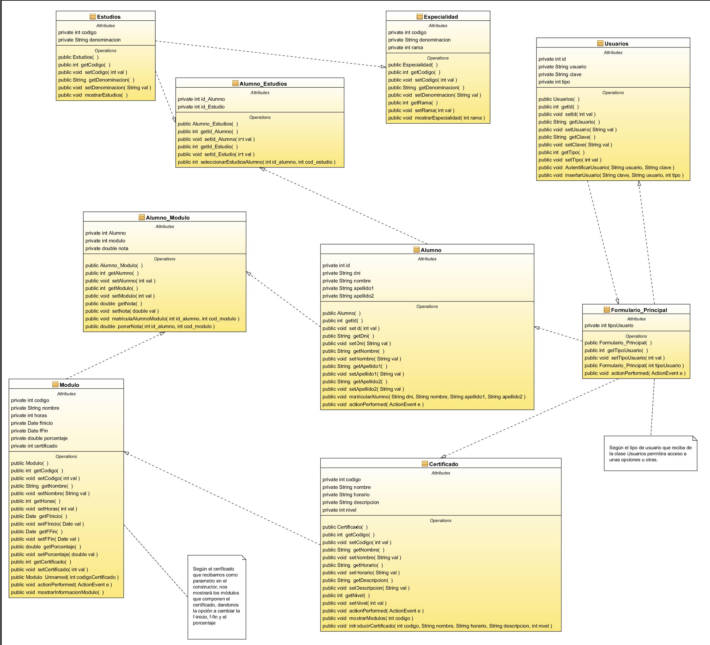
****

* En el tema de los certificados hemos dicho que nos interesa saber que módulos tiene que alumnos están matriculados en el certificado y que notas han sacado en cada módulo.  
  Hemos contemplado que un certificado puede tener como mínimo un módulo o varios mientras un módulo puede estar en un solo certificado.   
  A su vez hemos contemplado que un alumno pueda hacer dos certificados a la vez. Vemos más eficiente que un alumno esté relacionado con módulos porque como hemos visto anteriormente el certificado puede tener varios módulos.   
  Si unimos el alumno con el certificado, puede llegar el caso de que un alumno haya comenzado varias veces el mismo certificado porque por causas ajenas al alumno no se haya finalizado el certificado. Si unimos el alumno con el módulo tenemos claro en qué fecha comenzó el certificado y en qué fecha terminó. Pero creando un identificador en la entidad módulos que sea creado por la aplicación teniendo en cuenta el año y el mes.   
  Entonces con estos precedentes vamos a analizar las relaciones entre las entidades: por ejemplo, un alumno puede estar apuntado a uno o varios módulos mientras que un módulo puede tener uno o varios alumnos. En la relación que tenemos vamos a tener un atributo que se va a llamar nota.  
    
  En suma el resultado del análisis del tema de certificados y módulos y la unión con el tema de los alumnos.  
  

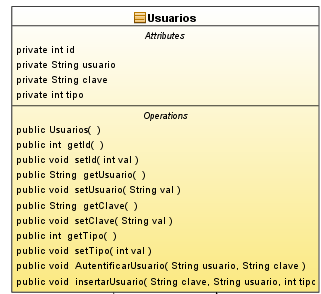
En definitiva, después del análisis del modelo entidad relación de los apartados alumnos, certificados y módulos, hemos llegado a la conclusión que el modelo entidad relación queda así:



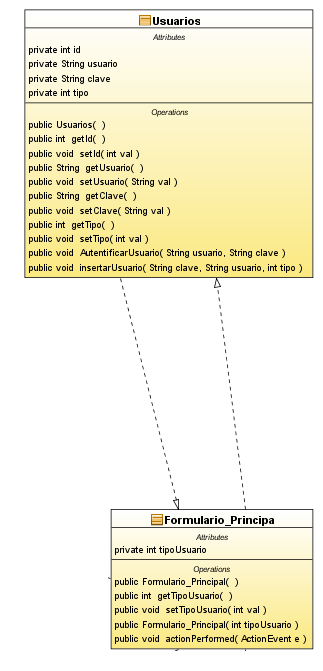
Capturas interfaz

Diagrama de clases: explicar cada 

**CASO USUARIO:**



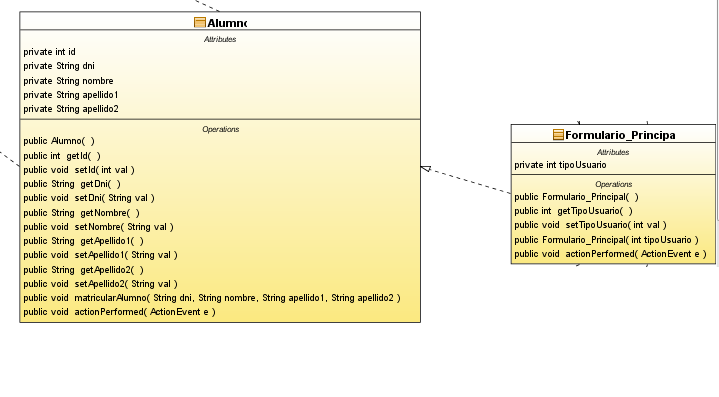
Dentro del caso usuario lo primero que le vamos a permitir, ya que como mínimo tenemos que tener un administrador es autentificarse el usuario para poder crear más usuarios, poder crear certificados, módulos, alumnos (dentro de los alumnos podrá asignar estudios y sus especialidades correspondientes)



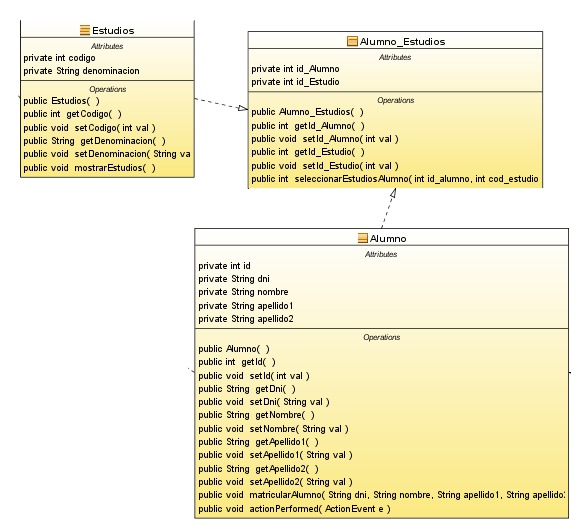
A un usuario después de autentificarse se le da acceso al formulario principal que según el tipo de usuario que sea podrá ver un tipo de formulario u otro, según sea usuario o administrador.

En caso de autentificarse como un usuario-administrador podrás crear usuarios. Así que la relación o implementación es recíproca.

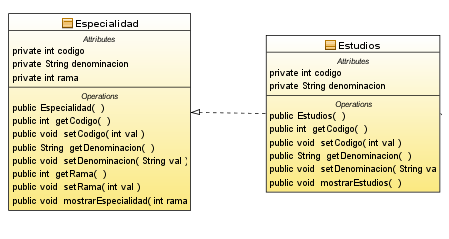
**CASO ALUMNO**



Desde el formulario principal nos va a dar la opción de gestionar el tema de los alumnos, no importa el tipo de usuario que se autentifique. Cualquier usuario puede matricular alumnos.



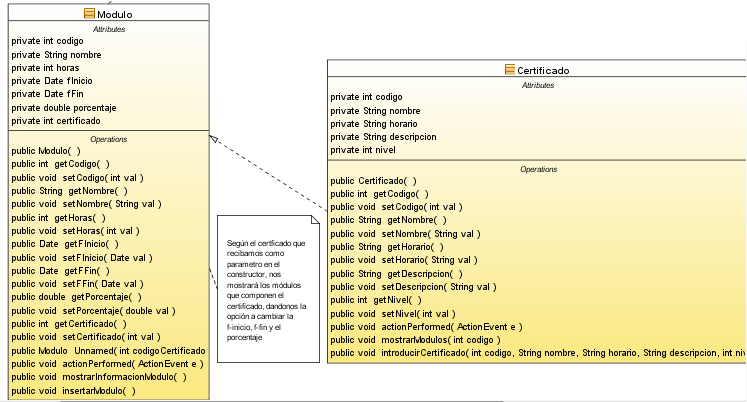
Para matricular a un alumno, a la aplicación le interesa guardar datos sobre los estudios que ha cursado el alumno y cómo hemos comentado anteriormente en el modelo entidad-relación, un alumno puede haber cursado varios estudios y un estudio puede tenerlo varios alumnos o ninguno. Así que en la relación n-n que surge, aparecería la entidad alumnos-estudios que se traduciría en el diagrama de clases en una clase que se llamaría Alumnos-Estudios en la cual el usuario podrá seleccionar los estudios que posee cada alumno. En la clase estudios solo se va a permitir actualizar o mostrar los estudios, almacenados en la base de datos.



Cada estudio tiene sus especialidades y por especialidades nos referimos por ejemplo si indicamos en el estudio “licenciado” en la especialidad indicaremos el tipo “Derecho, Ciencias del mar, etc” o si indicamos “Grado superior de Formación profesional” en la especialidad indicaremos “Desarrollo de aplicaciones web, mecánica y automoción, etc”.

Dentro de la clase especialidad nos interesa cuando el usuario dé de alta a un alumno filtre por estudios las especialidades en cuestión. En la clase especialidad sólo se va a permitir actualizar o mostrar las especialidades introduciendo en cada atributo los datos.

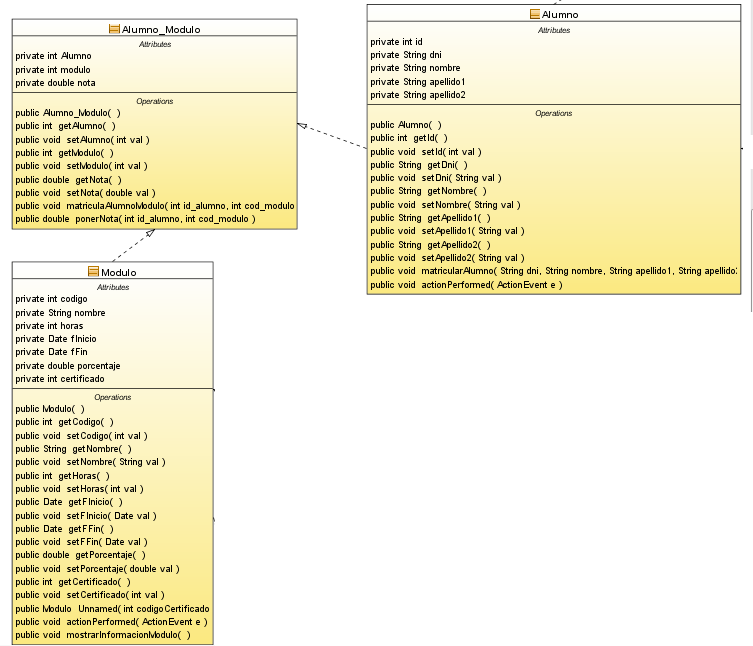
**CASO CERTIFICADO**

****

En el caso de los certificados el usuario normal o el administrador puede crear certificados, actualizarlos, modificarlos o borrarlos, además de poder asociar los módulos a los certificados a los que pertenece el módulo (esto se realizará mediante la función mostrarModulos en la clase certificados a la que se pasa por parámetro el código del certificado).

En la clase módulo se podrá insertar los módulos que sería coger los módulos maestros( módulos con una plantilla común pero sin fechas de inicio y fecha fin), introduciendoles las fechas de inicio, fecha fin y el porcentaje.

**CASO MATRÍCULA/PONER NOTA**

****

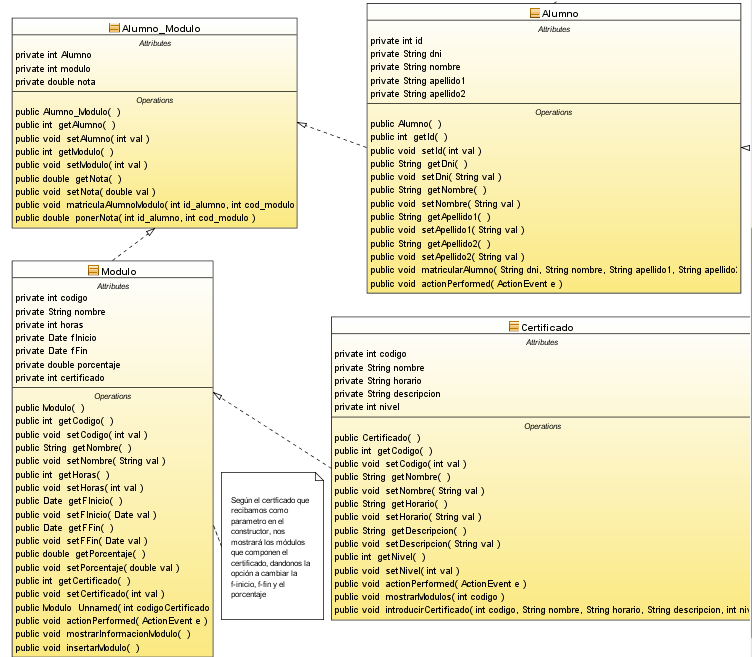
Hemos equiparado los casos de poner nota y matricular alumnos porque intervendrían las mismas clases para los dos procesos.

A la hora de matricular se crea una relación entre las clases alumno y módulo, entendiendo que un alumno puede realizar varios módulos a lo largo de la relación con la entidad formativa, y un módulo puede haberlo cursado varios alumnos, así que nacería una entidad de la relación n-n, que sería alumno-módulo. La relación a la hora de matricular se gestiona pasándole como parámetro a la función o método el código del alumno y el código del módulo.

A la hora de poner las notas adquiridas por los alumnos se gestiona creando un método al que se pasa como parámetro la nota del módulo.

Cómo nos interesa recoger la nota final del alumno en el certificado, se puede calcular con la relación que existe entre las clases certificados, módulos y alumnos-módulos ya que un certificado está compuesto por varios módulos y un módulo solo puede pertenecer a un certificado, y un módulo se identifica por su porcentaje dentro del certificado y como tenemos la nota de alumno-modulo así que :

-Nota final = sumatorio(modulo.porcentaje\*alumno-modulo.nota)/100 .



# Implementación