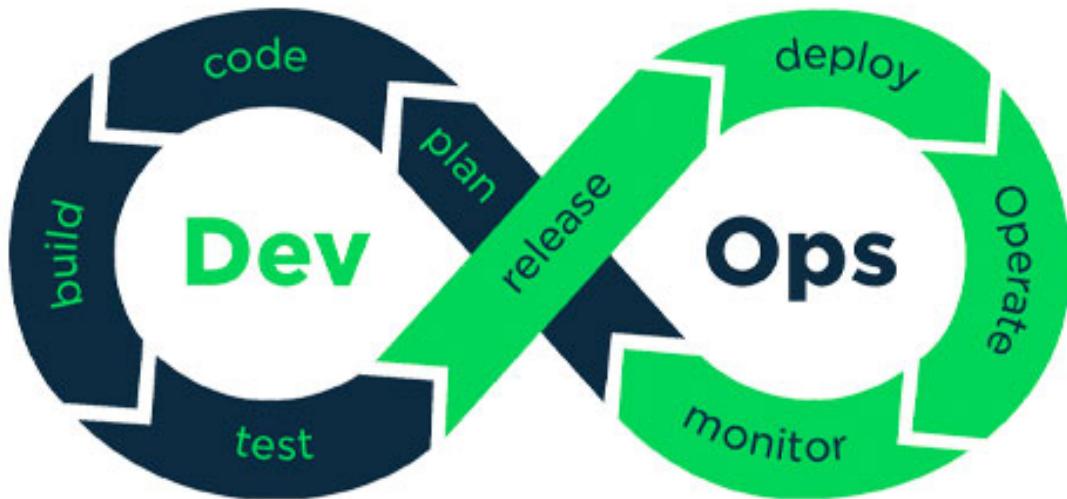


# Práctica reingeniería



María Fernández Herraiz, bmo553, 20% del trabajo.

Jose Carlos Gago Hernández, bmo031, 20% del trabajo.

David Ramajo Fernández, bno064, 20% del trabajo.

Rafael Sánchez Rodríguez, bmo323, 20% del trabajo.

Javier Aparicio León, bo0437, 20% del trabajo.



<b>PARTE 1</b>	<b>2</b>
<b>REQUISITOS FUNCIONALES</b>	2
<b>BAD SMELLS</b>	4
<b>DIAGRAMAS</b>	8
Clases	8
Entidad-Relación	9
Diagramas de secuencia	10
Secuencia (altaEvaluacion)	10
Secuencia (altaAlumno)	11
Secuencia (altaMatricula)	12
<b>PARTE 2</b>	<b>13</b>
<b>COMPRENSIÓN Y DEFINICIÓN DE REQUISITOS</b>	13
<b>COMPRENSIÓN DEL SISTEMA LEGADO</b>	15
Diagrama de clases	15
Diagrama entidad-relación	16
Diagrama de flujo	17
<b>COMPRENSIÓN DE LA TECNOLOGÍA OBJETIVO</b>	18
<b>COMPRENSIÓN DE LA ARQUITECTURA OBJETIVO</b>	19
<b>DEFINICIÓN DE LA ESTRATEGIA DE MIGRACIÓN</b>	19



## PARTE 1

### REQUISITOS FUNCIONALES

RU-1: Como usuario quiero ser capaz de acceder a la información de la aplicación desde la ventana principal pulsando en el ícono de la aplicación.

RF-1.1: El sistema permitirá que el usuario pueda volver a la ventana principal de la aplicación en cualquier momento pulsando el botón HOME.

RF-1.2: El sistema permitirá que el usuario pueda volver a la pantalla anterior en cualquier momento pulsando el botón ATRÁS.

RF-1.3: El sistema permitirá que el usuario pueda acceder a la información de la aplicación en cualquier momento pulsando sobre la pestaña ayuda y después el botón ABOUT.

RF-1.4: El sistema permitirá que el usuario pueda acceder al tutorial de la aplicación en cualquier momento pulsando primero sobre la pestaña ayuda y después el botón ayuda.

RF-1.5: El sistema permitirá que el usuario pueda acceder al tutorial de la aplicación pulsando sobre el botón HELP de la ventana principal.

RU-2: Como usuario quiero ser capaz de modificar las calificaciones de un alumno.

RF-2.1: El sistema permitirá la usuario modificar las calificaciones de un alumno introduciendo el DNI, numero de matricula o grupo de clase del alumno del que se quiere modificar las calificaciones.

RU-3: Como usuario quiero ser capaz de consultar las calificaciones de un alumno.

RF-3.1: El sistema permitirá que el usuario pueda consultar las calificaciones de un alumno introduciendo DNI, número de matrícula, grupo de clase o convocatoria del alumno del que se quiere consultar las calificaciones.

RU-4: Como usuario quiero ser capaz de modificar un grupo de prácticas.

RF-4.1: El sistema permitirá al usuario modificar un grupo de prácticas introduciendo un código de grupo, DNI de un alumno perteneciente al grupo y el tutor del grupo.

RU-5: Como usuario quiero ser capaz de consultar un grupo de prácticas.

RF-5.1: El sistema permitirá al usuario consultar un grupo de prácticas introduciendo un código de grupo, DNI de un alumno perteneciente al grupo y el tutor del grupo.

RU-6: Como usuario quiero ser capaz de dar de alta un grupo de prácticas.

RF-6.1: El sistema permitirá al usuario dar de alta un grupo de prácticas introduciendo un código de grupo, DNI de alumno 1, Dni de alumno 2 (opcional), el tutor del grupo y la nota (opcional).

RU-6: Como usuario quiero ser capaz de dar de baja un grupo de prácticas.

RF-6.1: El sistema permitirá al usuario dar de baja un grupo de prácticas introduciendo un código de grupo.



POLITÉCNICA

UNIVERSIDAD  
POLITÉCNICA  
DE MADRID



**RU-7:** Como usuario quiero ser capaz de consultar profesores.

**RF-7.1:** El sistema permitirá al usuario consultar profesores por nombre, apellidos o grupo de clase.

**RU-8:** Como usuario quiero ser capaz de modificar profesores.

**RF-8.1:** El sistema permitirá al usuario modificar profesores por nombre, apellidos o grupo de clase.

**RU-9:** Como usuario quiero ser capaz de dar de alta profesores.

**RF-9.1:** El sistema permitirá al usuario dar de alta profesores por nombre, apellidos o grupo de clase.

**RU-9:** Como usuario quiero ser capaz de dar de baja profesores.

**RF-9.1:** El sistema permitirá al usuario dar de baja profesores por nombre, apellidos o grupo de clase.

**RU-10:** Como usuario quiero ser capaz de consultar el historial de alumnos.

**RF-10.1:** El sistema permitirá al usuario consultar el historial de alumnos por DNI, curso o convocatoria.

**RU-11:** Como usuario quiero ser capaz de consultar alumnos.

**RF-11.1:** El sistema permitirá al usuario consultar alumnos por nombre, apellidos, DNI o número de matrícula.

**RU-12:** Como usuario quiero ser capaz de modificar alumnos.

**RF-12.1:** El sistema permitirá al usuario modificar alumnos por nombre, apellidos, DNI o número de matrícula.

**RU-13:** Como usuario quiero ser capaz de dar de alta alumnos.

**RF-13.1:** El sistema permitirá al usuario dar de alta alumnos por nombre, apellidos, DNI o número de matrícula.

**RU-14:** Como usuario quiero ser capaz de dar de baja alumnos.

**RF-14.1:** El sistema permitirá al usuario dar de baja alumnos por nombre, apellidos, DNI o número de matrícula.



## BAD SMELLS

### Paquete capaDatos

- Algunas clases presentan demasiados métodos.
- Constructores vacío.
- Constructores sin comentarios.
- Constructores innecesarios.
- Comentarios muy largos.
- Uso innecesario de System.out.println en los casos como  
`System.out.println(ex.getMessage());`
- El argumento o parámetro Alumno alumno, no es utilizado en muchos métodos.
- Paréntesis innecesario en el return del método public boolean noEstaDadoDeAlta().
- Nombre de muchas variables demasiado cortos.
- Múltiples variables han podido declararse 'final'.
- Lanzamiento de múltiples excepciones que pueden hacer perder el seguimiento del programa.
- Violación en múltiples casos de la Ley de Demeter.

### Paquete capalInterfaz

- Errores menores como nombres de variables muy largos, que empiezan por mayúscula o que contiene símbolos como “\_”.

### Paquete capalInterfaz.ficheros

- Violaciones de la LawOfDemeter
- Falta de comentarios
- Nombres de variables muy largos
- Nombres de variables muy cortos
- Falta de Override
- Tamaño del comentario
- Evitar inicializar objetos en loops
- Evitar Raw Exceptions
- Paréntesis sin uso
- Problemas con las convenciones a la hora de nombrar variables
- Variables locales sin usar
- Variables importadas sin usar
- Anomalías en el análisis del Dataflow



POLITÉCNICA

UNIVERSIDAD  
POLITÉCNICA  
DE MADRID



## Paquete capalInterfaz.listados

- Errores menores como nombres de variables y de métodos muy largos, que empiezan por mayúscula o que contiene símbolos como “\_”.
- Evitar superponer lanzamiento de excepciones.
- Línea 50 y 53 de la clase ListadoGrupoPractica.

## Paquete capalInterfaz.menuAlumnos

- Comentarios muy largos.
- Falta de comentarios en algunos métodos.
- Violaciones potenciales de la ley de Demeter.
- Múltiples variables podrían haber sido declaradas final.
- Nombres de variables muy largos.
- Uso indebido del System.out.println.
- Uso de paréntesis innecesarios en algunos casos como en la función modificarboton() if ((y+(INCREMENTOY\*num\_filas) > 500)).
- Uso reiterado de métodos de tipo arraylist<> que favorecen al bajo acoplamiento.

## Paquete capalInterfaz.menuCalificaciones

- Errores menores como nombres de variables y de métodos muy largos, que empiezan por mayúscula o que contiene símbolos como “\_”.

## Paquete capalInterfaz.menuConfiguracion

- Violaciones de la LawOfDemeter
- Falta de comentarios
- Nombres de variables muy largos
- Nombres de variables muy cortos
- Falta de Override
- Tamaño del comentario
- Variables locales sin usar
- Evitar Raw Exceptions
- Método excesivamente largo
- Problemas con las convenciones a la hora de nombrar variables
- Anomalías en el análisis del Dataflow



## Paquete capalInterfaz.menuGruposPracticas

- Falta de comentarios.
- Declaración de variables que podrían haber sido declaradas estáticas y no lo han sido.
- Nombres de variables excesivamente cortos.
- Falta de utilización de ‘Diamonds operators’ como en el caso de “private ArrayList<JTextField> arraynombre = new ArrayList<JTextField>()”, que podría escribirse como private ArrayList<JTextField> arraynombre = new ArrayList<>().
- Nombres de variables excesivamente largos.
- Parámetros que podrían haber sido definidos como final.
- Potencial violación de la ley de Demeter.
- Detección anómala en la asignación de valores en algunas variables como fila = -1;

## Paquete capalInterfaz.MenuHistoricoAlumnos

- Violaciones de la LawOfDemeter
- Falta de comentarios
- Nombres de variables muy largos
- Nombres de variables muy cortos
- Falta de Override
- Tamaño del comentario
- Variables locales sin usar
- Paréntesis sin uso
- Método excesivamente largo
- Problemas con las convenciones a la hora de nombrar variables
- Anomalías en el análisis del Dataflow

## Paquete capalInterfaz.menuPrincipal

- La clase FrameMenuPrincipal solo tiene constructores por lo que podría ser final.
- Errores menores como nombres de variables y de métodos muy largos, que empiezan por mayúscula o que contiene símbolos como “\_”.

## Paquete capalInterfaz.menuProfesores

- Violaciones de la LawOfDemeter
- Falta de comentarios
- Nombres de variables muy largos
- Nombres de variables muy cortos
- Falta de Override
- Tamaño del comentario
- Variables locales sin usar
- Paréntesis sin uso
- Método excesivamente largo
- Problemas con las convenciones a la hora de nombrar variables
- Anomalías en el análisis del Dataflow



POLITÉCNICA

UNIVERSIDAD  
POLÍTÉCNICA  
DE MADRID

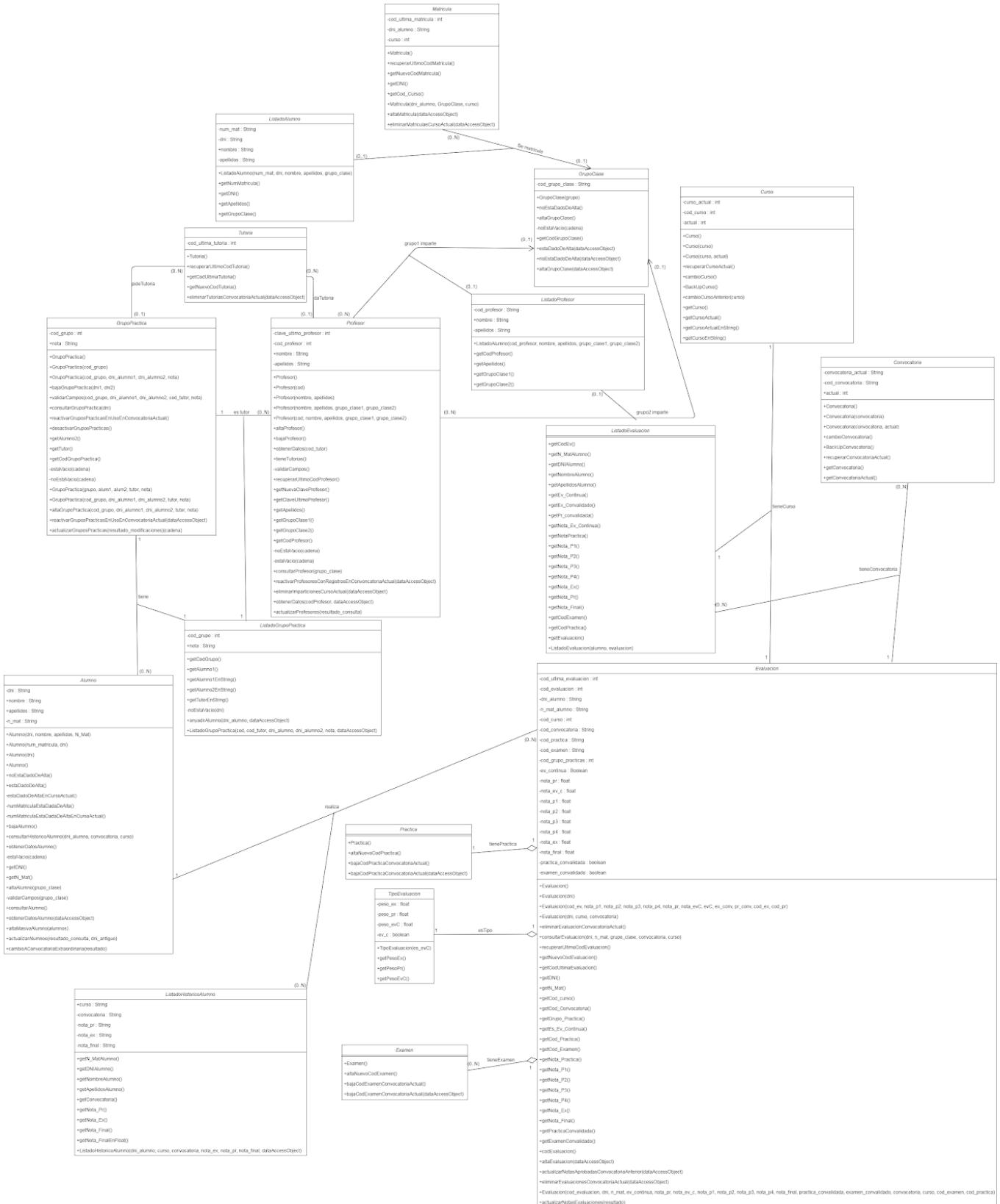


## Paquete capaLogicaNegocio

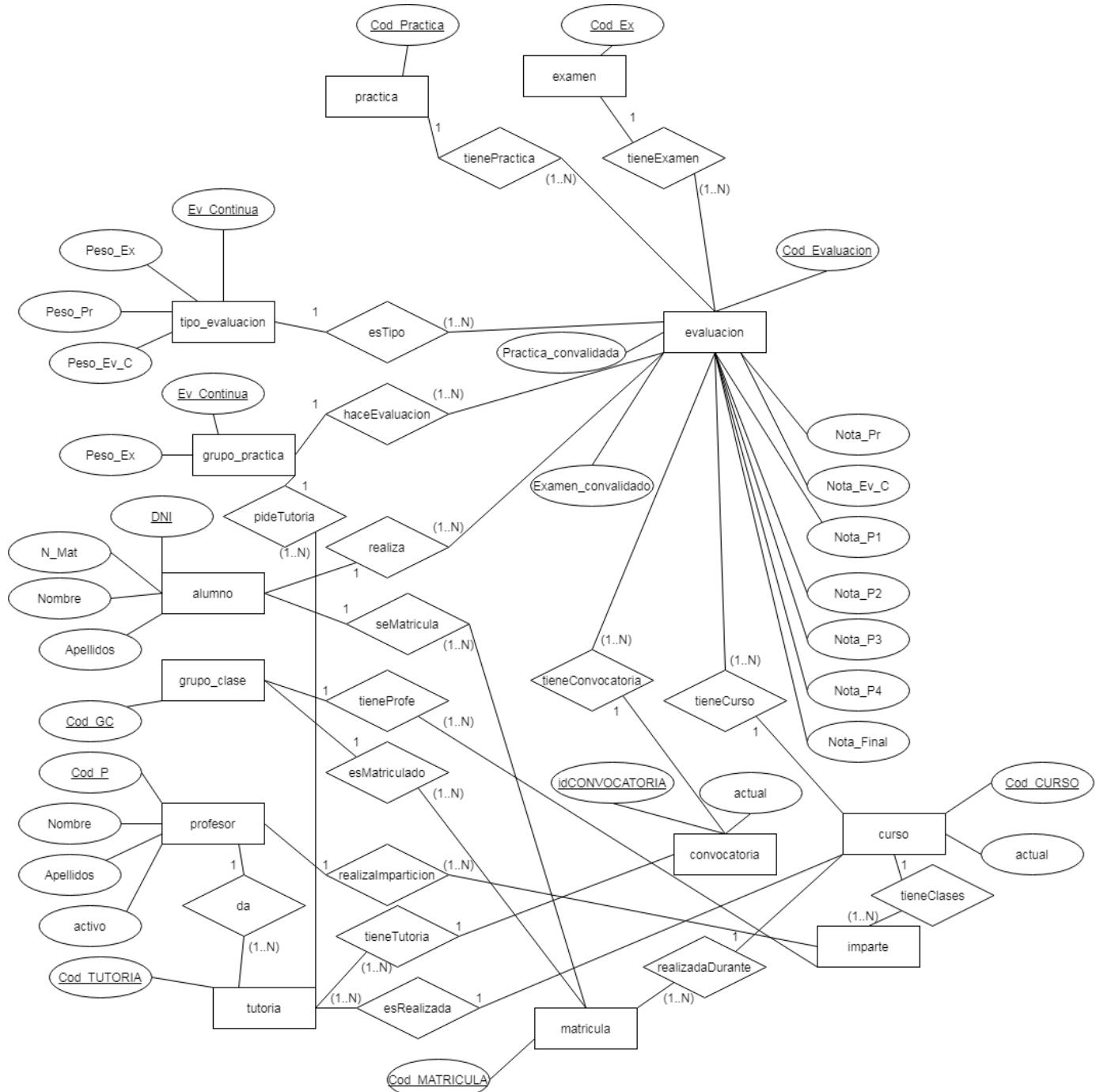
- Violaciones de la LawOfDemeter
- Falta de comentarios
- Nombres de variables muy largos
- Nombres de variables muy cortos
- Falta de Override
- Tamaño del comentario
- Evitar inicializar objetos en loops
- Evitar Raw Exceptions
- Paréntesis sin uso
- Problemas con las convenciones a la hora de nombrar variables
- Variables locales sin usar
- Variables importadas sin usar
- Anomalías en el análisis del Dataflow
- Muchos métodos
- For que puede ser foreach
- Constructores innecesarios
- Constructores vacíos sin comentarios
- Declaraciones prematuras
- Código muerto: la funcionalidad de tutorías no parece funcionar en la aplicación por lo que hemos podido ver.

## DIAGRAMAS

### Clases

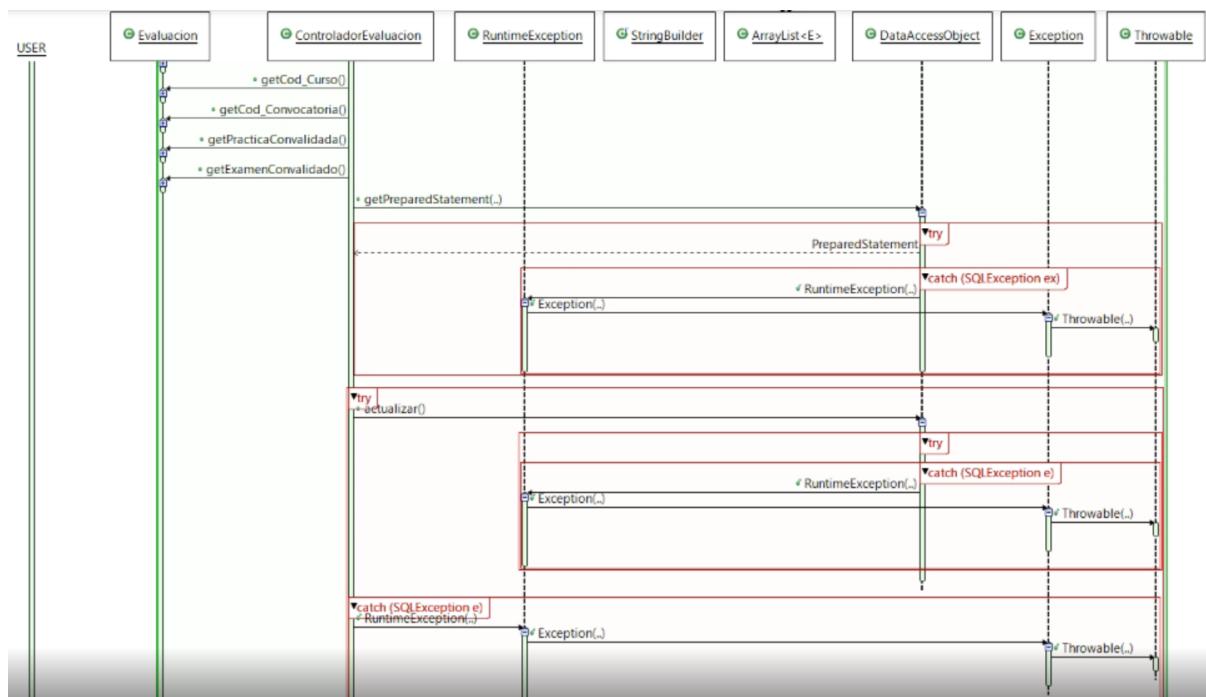
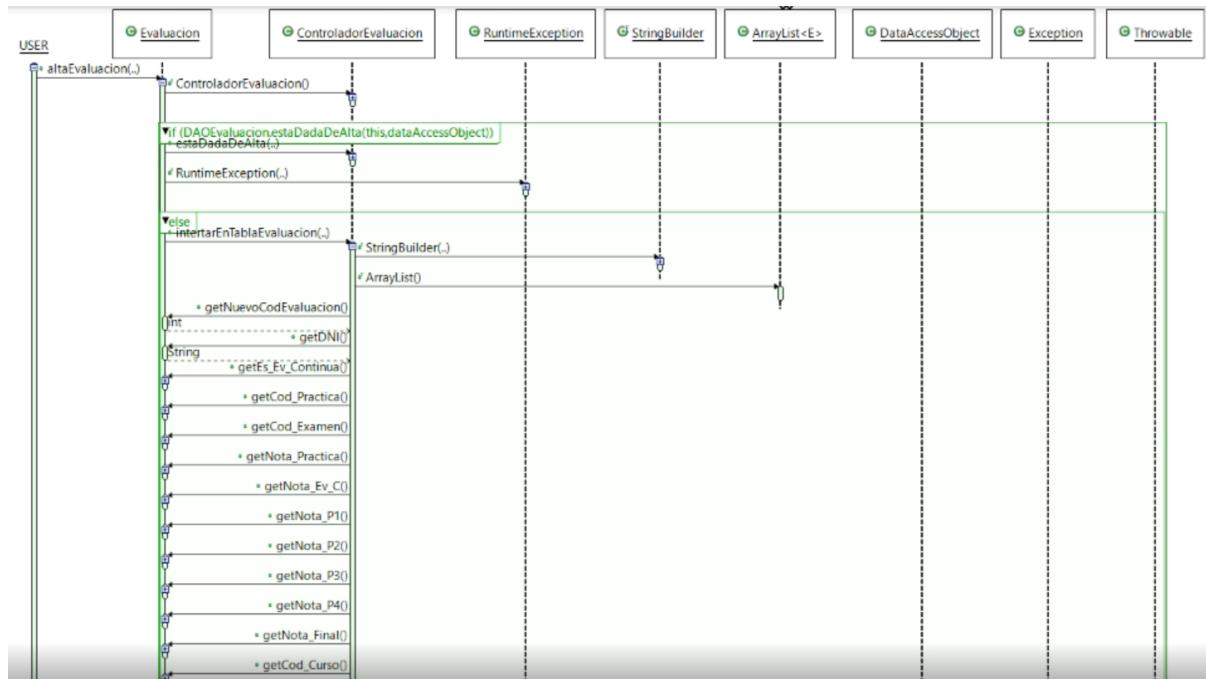


## Entidad-Relación

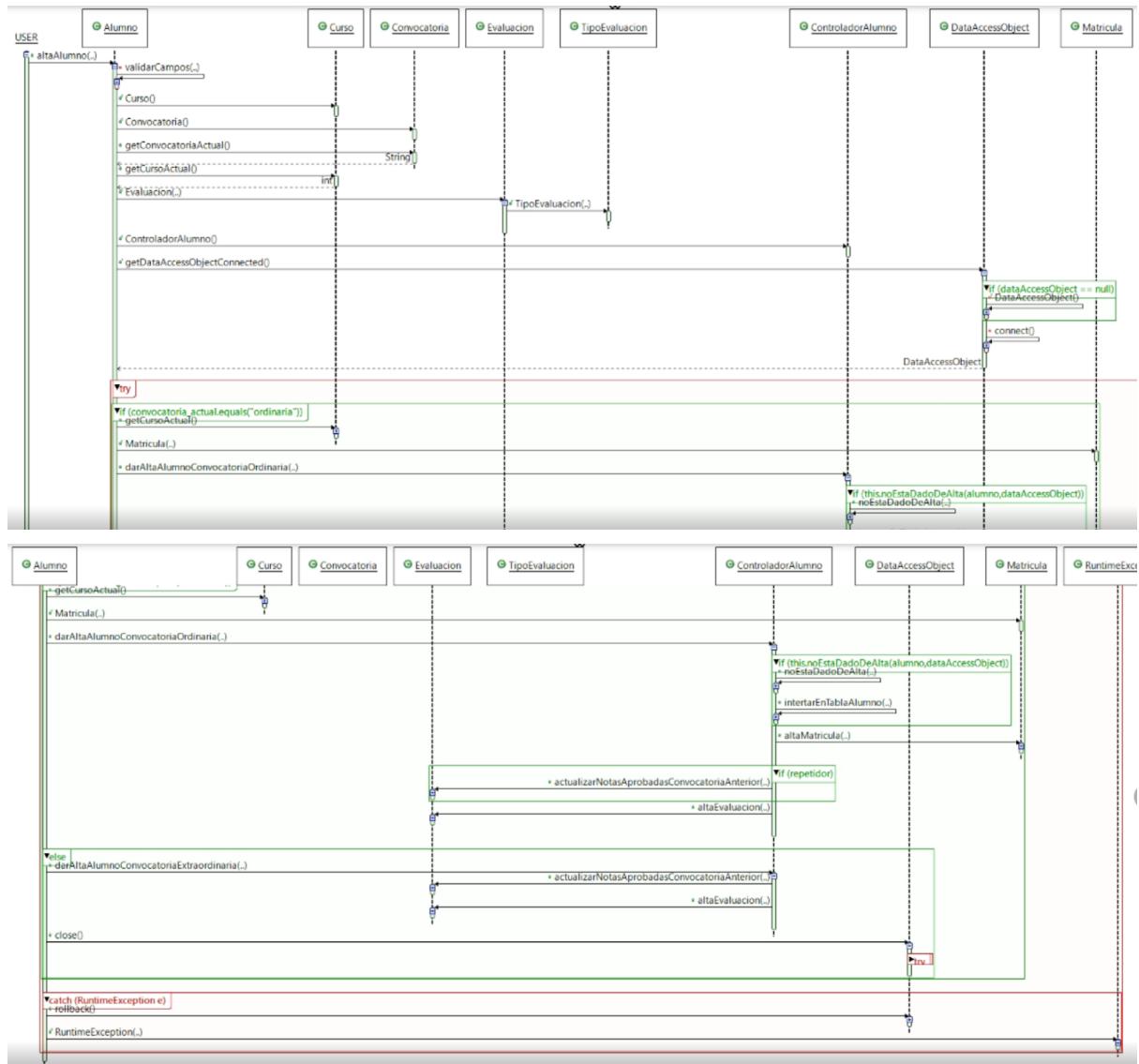


## Diagramas de secuencia

### Secuencia (altaEvaluacion)



## Secuencia (altaAlumno)



## Secuencia (altaMatricula)





## PARTE 2

### COMPRENSIÓN Y DEFINICIÓN DE REQUISITOS

RU-1: Como usuario quiero ser capaz de acceder a toda la funcionalidad de la aplicación desde cualquier ventana de la aplicación.

RF-1.1: El sistema permitirá que el usuario acceda a cualquier funcionalidad de la aplicación desde la barra de elementos superior.

RU-2: El usuario debe poder ser diferenciado según su rol.

RF-2.1: El sistema distinguirá entre tres tipos de roles para el usuario: Alumno, Profesor y Administrador.

RU-3: El usuario quiere que los administradores tengan acceso solo a las funcionalidades de su cargo.

RF-3.1: El sistema permitirá a los administradores poder definir el número máximo de alumnos por grupo.

RF-3.2: El sistema permitirá a los administradores poder crear, eliminar, consultar y editar cursos.

RF-3.3: El sistema permitirá a los administradores poder crear, eliminar, consultar y editar grupos.

RF-3.4: El sistema permitirá a los administradores poder asignar y eliminar a los profesores de los grupos.

RU-4: El usuario quiere que los profesores tengan acceso solo a las funcionalidades de su cargo.

RF-4.1: El sistema permitirá a los profesores poder matricular y desmatricular alumnos en grupos.

RF-4.2: El sistema permitirá a los profesores poder modificar y consultar las calificaciones de los alumnos.

RF-4.3: El sistema permitirá a los profesores poder consultar grupos.

RF-4.4: El sistema permitirá a los profesores poder realizar evaluaciones.

RU-5: El usuario quiere que los alumnos tengan acceso solo a consultar sus evaluaciones.

RF-5.1: Los alumnos podrán acceder a la funcionalidad mis evaluaciones de la aplicación para ver sus evaluaciones.

RF-5.2: Atributos de Alumno: Curso, Examen, Práctica y Trabajo.

RF-5.3: Los alumnos tienen tres calificaciones: Examen, Práctica y Trabajo.

RU-6: Como usuario quiero poder consultar los créditos ects asignados de un profesor.

RF-6.1: Atributos de Asignación Docente: créditos ects del profesor.

RF-6.2: Durante la Asignación Docente. Se creará una clave compuesta formada por los identificadores de profesor, grupo y curso.



POLITÉCNICA

UNIVERSIDAD  
POLITÉCNICA  
DE MADRID



RU-7: Como usuario quiero poder consultar el nombre, los apellidos, el DNI, el correo y su contraseña.

RF-7.1: Atributos de Usuario son: nombre, apellidos, dni, email y contraseña.

RU-8: Como usuario quiero poder consultar en una evaluación si el alumno se ha presentado, el tipo de convocatoria, ordinaria o extraordinaria y las notas del examen, trabajo y práctica.

RF-8.1: Atributos de Evaluación son: no presentado(boolean), tipo convocatoria, nota del examen, nota del trabajo, nota de la práctica.

RF-8.2: Hay dos tipos de convocatoria, Ordinaria o Extraordinaria.

RF-8.3: Solo se pueden crear dos convocatorias por cada alumno, para el mismo curso y asignatura y dichas convocatorias deben ser de distinto tipo.

RU-9: Como usuario quiero poder consultar los datos de un profesor.

RF-9.1: El sistema permitirá que cualquier usuario pueda consultar los datos de un profesor.

RU-10: Como usuario quiero diferenciar dos tipos de evaluaciones, final o continua.

RF-10.1: Hay dos tipos de evaluación, Final o Continua.

RU-11: Como usuario quiero poder consultar el año del curso.

RF-11.1: Atributos del Curso son: año.

RU-12: Como usuario quiero poder consultar el nombre y la capacidad de un grupo.

RF-12.1: Atributos de Grupos son: nombre y capacidad del grupo (nº de alumnos).

RU-13: Como usuario quiero que no haya inconsistencias durante la matriculación de grupo.

RF-13.1: El sistema tendrá una clave compuesta formada por los identificadores de alumno, grupos y curso.

RU-14: Como usuario quiero que no haya inconsistencias durante la asignación de docencias.

RF-14.1: El sistema tendrá una clave compuesta formada por los identificadores del profesor, grupo y curso.

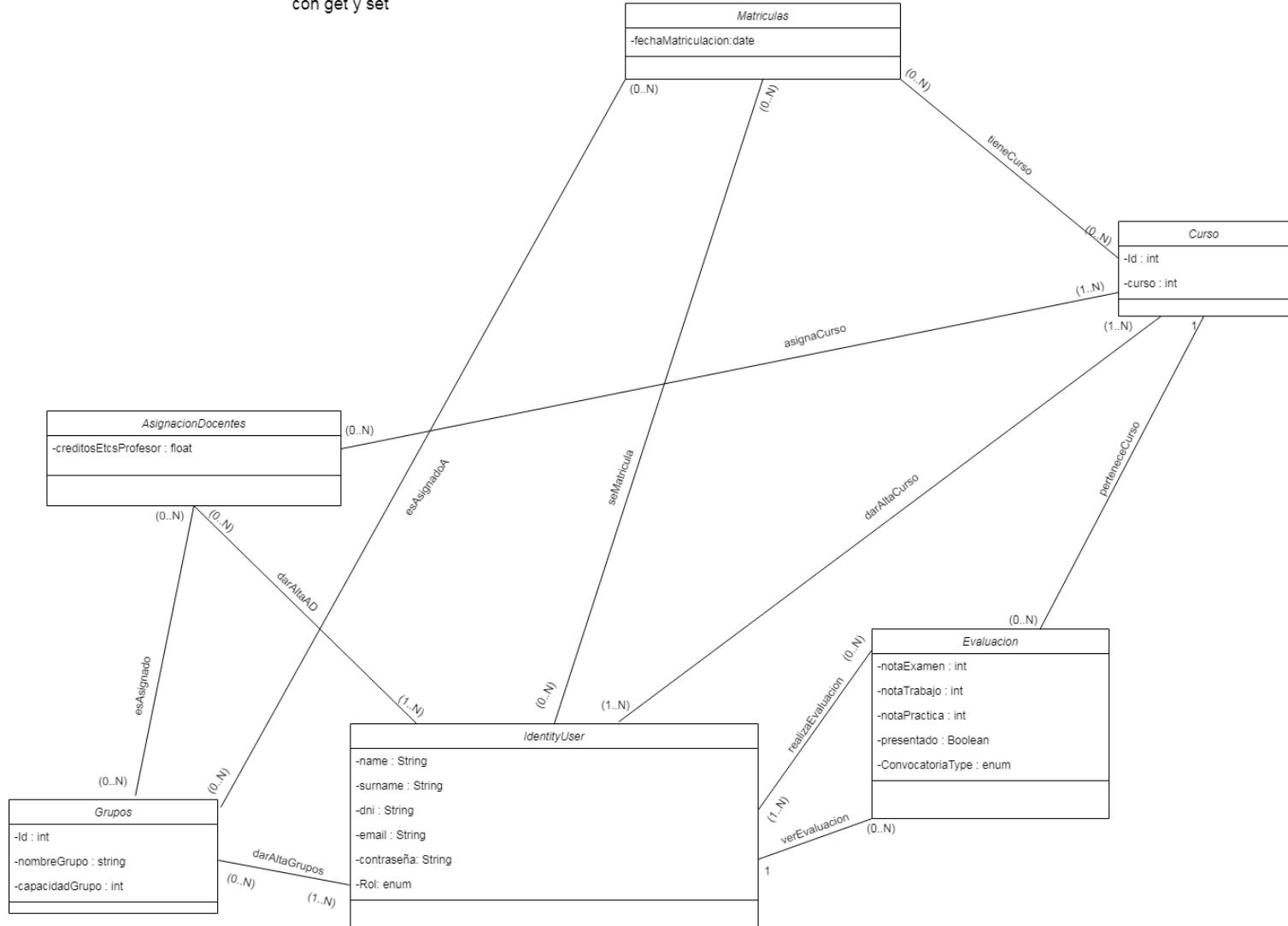
RU-15: Como usuario quiero poder consultar la fecha de matriculación.

RF-15.1: Atributos de Matrícula son: fecha de matriculación.

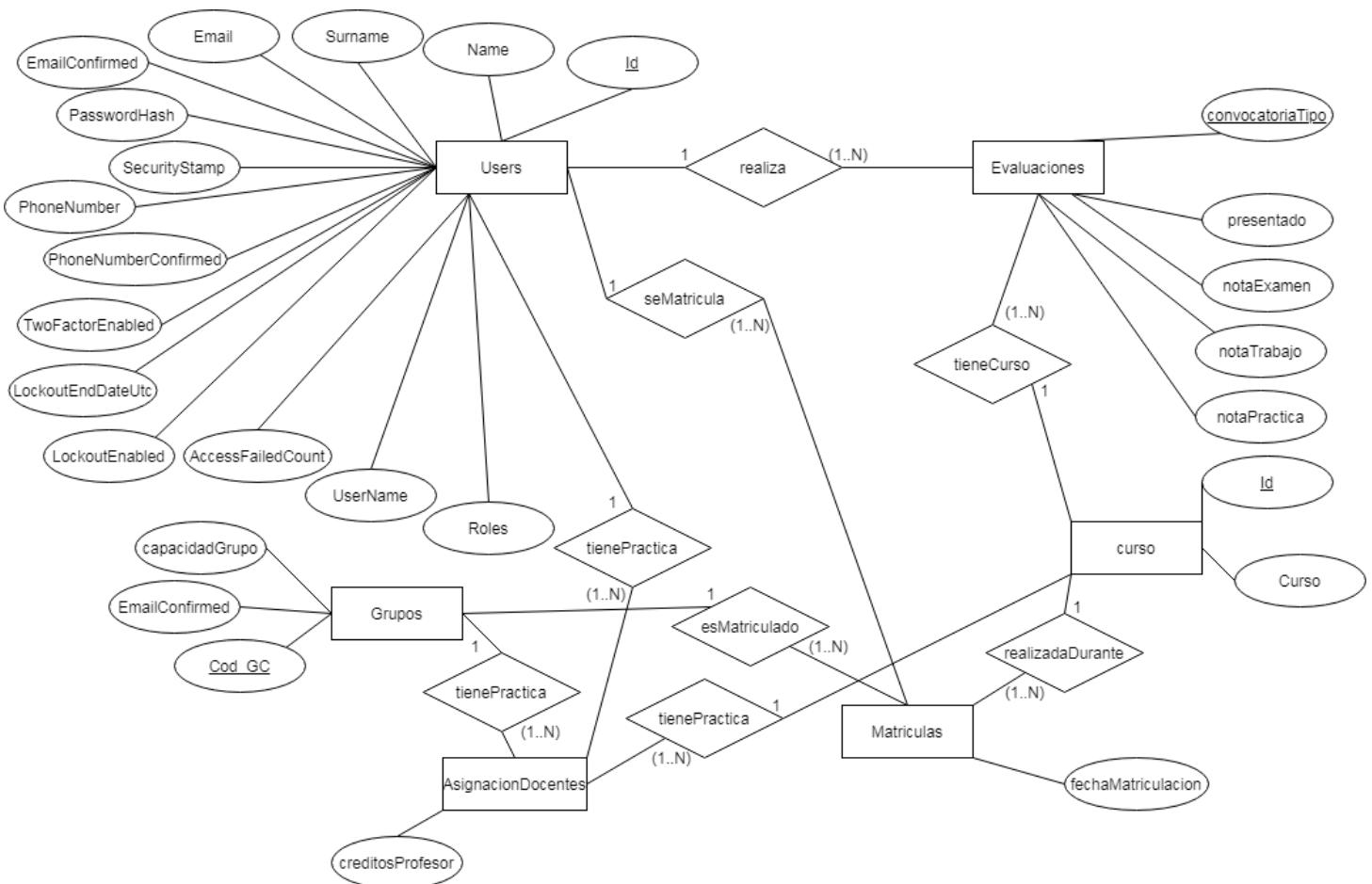
## COMPRENSIÓN DEL SISTEMA LEGADO

### Diagrama de clases

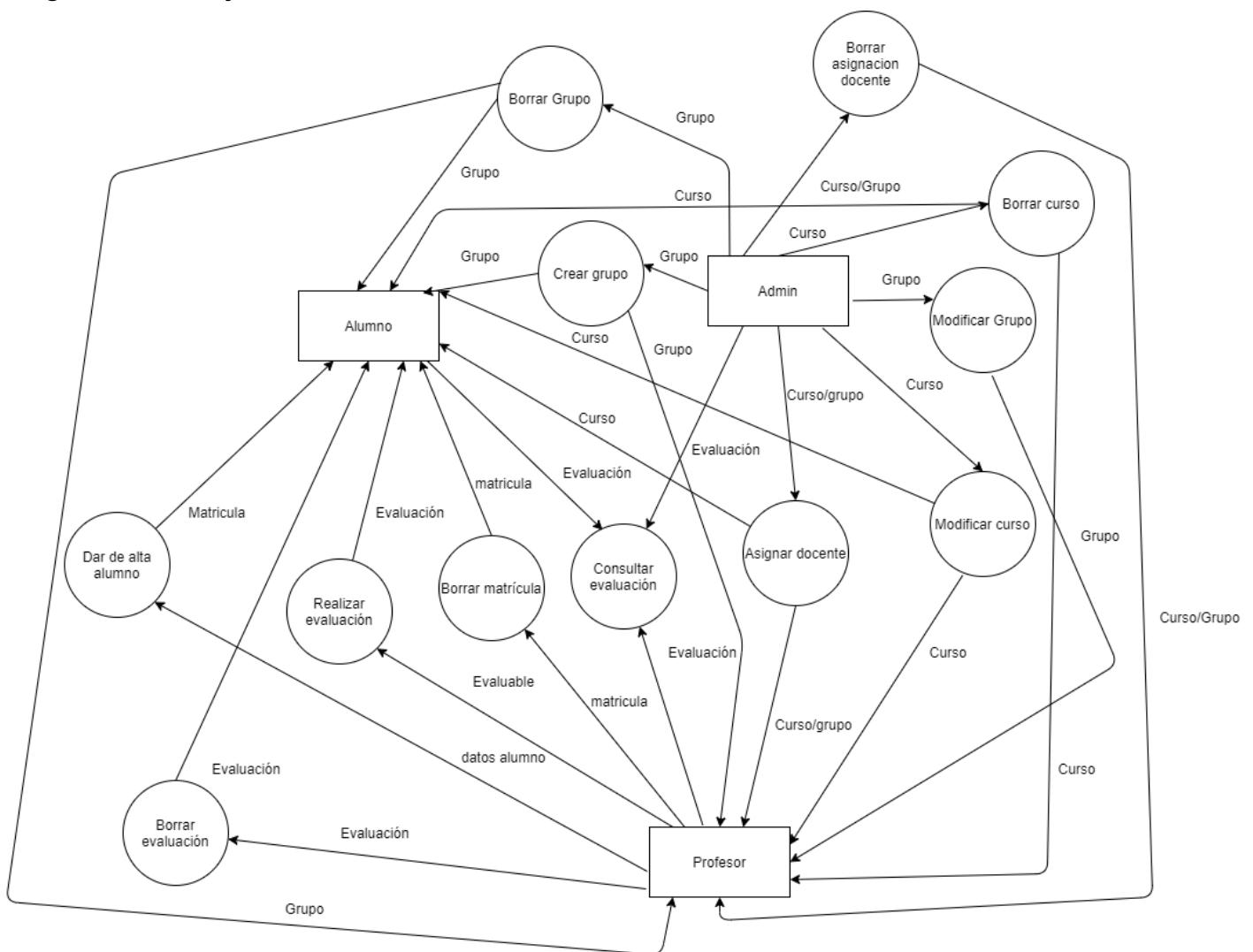
Todos los atributos contaran  
con get y set



## Diagrama entidad-relación



## Diagrama de flujo





## COMPRENSIÓN DE LA TECNOLOGÍA OBJETIVO

Vamos a usar ASP.NET para la nueva aplicación. ASP.NET Core es un marco de código abierto multiplataforma de alto rendimiento para crear aplicaciones modernas, habilitadas para la nube y conectadas a Internet.

ASP.NET Core nos proporciona los siguientes beneficios:

- Un historial unificado para construir UI y APIs web.
- Arquitectura testable.
- Blazor es una plataforma de trabajo para la creación de interfaces de usuario web interactivas del lado cliente con .NET.
- La habilidad de desarrollarse y funcionar en Windows, Linux y macOS.
- Herramientas que simplifican el desarrollo web.
- Inyección de dependencias incluida.
- Un pipeline de solicitudes HTTP ligero, modular y de alto rendimiento.

Vamos a utilizar ADO.NET Entity Framework para obtener acceso a la base de datos. Entity Framework (EF) es un ORM (Object/Relational Mapping) provisto por Microsoft que nos permitirá automatizar tareas con la BBDD como crear la base de datos, leer y actualizar datos.

ORM es una herramienta para almacenar datos desde objetos del dominio a bases de datos relacionales como MYSQL de forma automática y sin mucha programación. Incluye tres partes principales: objetos de clase de dominio, objetos de bases de datos relacionales y mapeo de información en cómo se mapean los objetos del dominio con objetos de bases de datos relacionales. Permite separar el diseño de la base de datos del diseño de las clases de dominio. Gracias a esto, la aplicación es mantenible y extensible. También automatiza las operaciones de CRUD (Create, Read, Update & Delete).

Escenarios de uso de EF:

- Database First: EF crea las clases de acceso a datos para una BBDD existente, por lo que se pueden usar estas clases para interaccionar con la BBDD en lugar de utilizar directamente ADO.Net
- Code First: EF crea la BBDD desde las clases del dominio de forma que nos podemos centrar en un diseño del dominio
- Model First: EF proporciona un diseñador del modelo de BBDD, a partir del cual se puede crear la BBDD y las clases basadas en este modelo de BBDD

En este desarrollo, vamos a utilizar Code First ya que nos permite probar la funcionalidad antes de la migración de la base de datos.



## COMPRENSIÓN DE LA ARQUITECTURA OBJETIVO

Vamos a usar el modelo MVC (modelo-vista-controlador). Es un patrón para desarrollo de aplicaciones con una buena arquitectura, testeables y fáciles de mantener. Las aplicaciones basadas en MVC contienen:

- Modelos: Clases que representan los datos de la aplicación y que usan lógica de validación para asegurarse de que las reglas de negocio se aplican con esos datos.
- Vistas: Archivos modelo que la aplicación usa para generar respuestas HTML dinámicas.
- Controladores: Clases que tratan las solicitudes entrantes del navegador, recupera datos del modelo y especifican modelos que devuelven una respuesta al navegador.

El flujo de control que se sigue generalmente al usar implementaciones MVC es:

1. El usuario interactúa con la interfaz de usuario.
2. El controlador recibe la notificación de la acción solicitada por el usuario. El controlador gestiona el evento que llega a través de eventos o callback.
3. El controlador accede al modelo, lo actualiza, modificando para que refleje la acción solicitada por el usuario.
4. El controlador delega a los objetos de la vista la tarea de desplegar la interfaz de usuario. La vista obtiene sus datos del modelo para generar la interfaz apropiada para el usuario donde se reflejan los cambios en el modelo. El modelo no tiene conocimiento directo sobre la vista.

Este flujo se realiza cíclicamente cada vez que el usuario interacciona con la interfaz.

## DEFINICIÓN DE LA ESTRATEGIA DE MIGRACIÓN

A continuación analizamos diversas estrategias de migración y a relacionarlas con el problema para llegar a la mejor opción. La ETSISI pide que se puedan comprobar ciertas funcionalidades en fases tempranas del desarrollo para poder tener un feedback y estar abiertos a los cambios. Además el sistema legado y el nuevo tienen que estar funcionando en paralelo hasta que dé el visto bueno los profesores de EMS.

- Cold Turkey: Esta estrategia no es apta ya que no podemos permitir que el sistema legado esté inaccesible durante el largo periodo de tiempo que supondría. Además, esta opción no nos permite probar funcionalidades hasta que la migración esté completa.
- Database Last: Aunque nos permite comprobar las funcionalidades, los datos se transmiten en una única transferencia que es larga y compleja, dejando el sistema legado inaccesible durante el proceso, no siendo aceptable
- Database First: Primero transfiere los datos y luego la funcionalidad, por razones obvias esto es inaceptable, pues no solo hay un largo tiempo hasta que la base de datos se transfiere, sino además hay que esperar a que se transfiera la funcionalidad.
- Chicken Little: Ofrece una migración gradual de la funcionalidad y la base de datos, durante la cual ambos sistemas están operativos, hasta que el nuevo sistema tiene todo y se puede apagar el legado. Esto cumple todas los requerimientos del cliente.



Como conclusión, se usará Chicken Little pues es el único que cumple con los criterios necesarios.

El proceso de migración que vamos a utilizar, basado en Chicken Little, es incremental. Los incrementos que vamos a usar son:

1. Login y gestión de usuarios: Se añadirán la funcionalidad del login y las de la gestión de los usuarios, y la migración de las tablas de usuarios.
2. Cursos y grupos: Se añadirán las funcionalidades de la gestión de cursos y las de grupos, junto con la migración de sus respectivas tablas.
3. Asignaciones docentes: Se añadirá la funcionalidad de la asignación docente y la migración de su tabla.
4. Matrículas: Se añadirá la funcionalidad del proceso de matriculación de alumnos y se añadirá la correspondiente tabla.
5. Evaluaciones: Se añadirá la funcionalidad de la gestión de las evaluaciones y su correspondiente tabla.

Durante la migración del sistema será necesario el uso de una gateway bidireccional que permita al sistema legado acceder a las partes migradas de la base de datos y al sistema nuevo acceder a la base de datos antigua. Las entradas y salidas de esta gateway serán:

- Entrada: Atributos entidad User(nombre, dni, rol...)  
Salida: Atributos clase profesor y alumno dependiendo del rol (nombre, dni, teléfono...)
- Entrada: Atributos entidad curso (id, curso).  
Salida: Atributos entidad curso con comparación de curso para comprobar si es el actual. (id, actual).
- Entrada: Atributos entidad grupo (id, nombreGrupo, capacidad).  
Salida: Atributos entidad grupoclase (cod\_gc), atributos clase grupo practica(cod\_gp).
- Entrada: Atributos entidad asignacionDocentes(UserId, Cursold, Grupold, creditosECTS)  
Salida: Atributos entidad imparte (Profesor\_cod\_P, Grupo\_clase\_cod\_gc, Curso\_cod\_curso)
- Entrada: Atributos entidad matrículas (UserId, Cursold, Grupold, fechaMatricula)  
Salida: Atributos entidad matrícula (Se ignora el código de matricula) (ALUMNO\_DNI, GRUPO\_CLASE\_Cod\_GC, CURSO\_Cod\_CURSO)
- Entrada: Atributos entidad Evaluaciones(UserId, cursold...)(Las notas se han simplificado a tres de examen, trabajo, prácticas y el tipo de convocatoria es un atributo)  
Salida: Atributos entidad Evaluación (Alumno\_DNI, GRUPO\_PRACTICA\_Cod\_GP...)



POLITÉCNICA

UNIVERSIDAD  
POLITÉCNICA  
DE MADRID



La puesta en producción seguirá este orden:

1. El login y la gestión de usuarios junto con la implementación de las gateway de acceso para los datos migrados.
2. Los cursos y grupos y actualizar las gateways para acceder a las nuevas migraciones.
3. Las asignaciones docentes y la actualización de las gateways.
4. Las matrículas y actualizar las gateways.
5. Las evaluaciones y la actualización de las gateways.
6. Finalmente, tras verificar el correcto funcionamiento del sistema nuevo, se apagará el sistema legado y las gateways.