UNIVERSIDAD POLITECNICA DE VALENCIA

Pràctiques

Butlletí Pràctica 2

Modelat OO en UML. Ferramentes

Enginyeria del Programari

ETS Enginyeria Informàtica
DSIC – UPV

Curs 2017-2018

1. Objectiu

L'objectiu de esta sessió de laboratori és utilitzar una ferramenta de modelat per representar diagrames de classe i diagrames de cas d'ús. Existeixen nombroses eines de suport a la notació gràfica UML (IBM Rational Rose, BoUML, StarUML, …, Microsoft Visio, etc). En aquest cas, *Visual Studio 2015 Enterprise* porta incorporat un editor gràfic per a especificar diferents diagrames UML i facilitar la integració dels mateixos en el nostre projecte.

En la nostra pràctica treballarem amb *Visual Studio 2015 Enterprise* i *Team Services*, utilitzant l'opció de crear *projectes de modelat*.

2. Tasques Prèvies del Team Leader

El treball continuarà on es va deixar en la sessió 1. Per a la realització d'aquesta sessió el Team Leader realitzarà els següents passos previs:

- Execute l'eina de desenvolupament Visual Studio 2015 Enterprise (Inici > Tots els programes > Microsoft Visual Studio 2015)
- Inicie sessió amb el seu compte de Microsoft prement a "Iniciar sesión" (dalt a la dreta)
- Connecte's al seu projecte d'equip definit en la sessió 1
- Obtinga l'última versió de la seua solució del repositori de Team Services

En aquella pràctica el team leader va crear una carpeta de solucions denominada *Testing*. El mateix team leader agregarà en aquesta carpeta de solucions una nova carpeta *Lab2*, on emmagatzemarem el treball d'aquesta sessió. En aquesta carpeta s'agregarà, al seu torn, dos projectes de modelatge *Lab2-DiagramaClases* i *Lab2-DiagramaCasosUso*, per al diagrama de classes i el diagrama de casos d'ús, respectivament (veure Figura 1). D'aquesta forma es podrà treballar sobre els dos diagrames en paral·lel sense conflictes en la sincronització. En particular, es decidirà què membres de l'equip treballaran en el diagrama de classes i quins membres en el diagrama de casos d'ús[1]. L'estructura del nostre projecte serà similar a la mostrada en la Figura 2.

Una vegada creada aquesta estructura del nostre projecte s'haurà de **protegir** la solució, perquè els projectes de modelatge s'incorporen a la versió del servidor i estiguen disponibles per a la seua descàrrega per qualsevol membre de l'equip de treball.

[1] També es podria haver agregat un únic projecte de modelatge *Lab2-Diagrames* que emmagatzemara tant el diagrama de classes com el de casos d'ús. No obstant açò, si distribuïm els diagrames entre diversos membres de l'equip de desenvolupament que treballen en paral·lel, en utilitzar arxius compartits pel mateix projecte, **podrien aparèixer problemes en la sincronització** quan tractem de protegir i fusionar. Per tant, **no es recomana treballar amb un únic projecte** de modelatge i, per simplicitat, **s'aconsella dividir el modelatge en dos projectes independents tal com es descriu en el butlletí i en la Figura 2.**

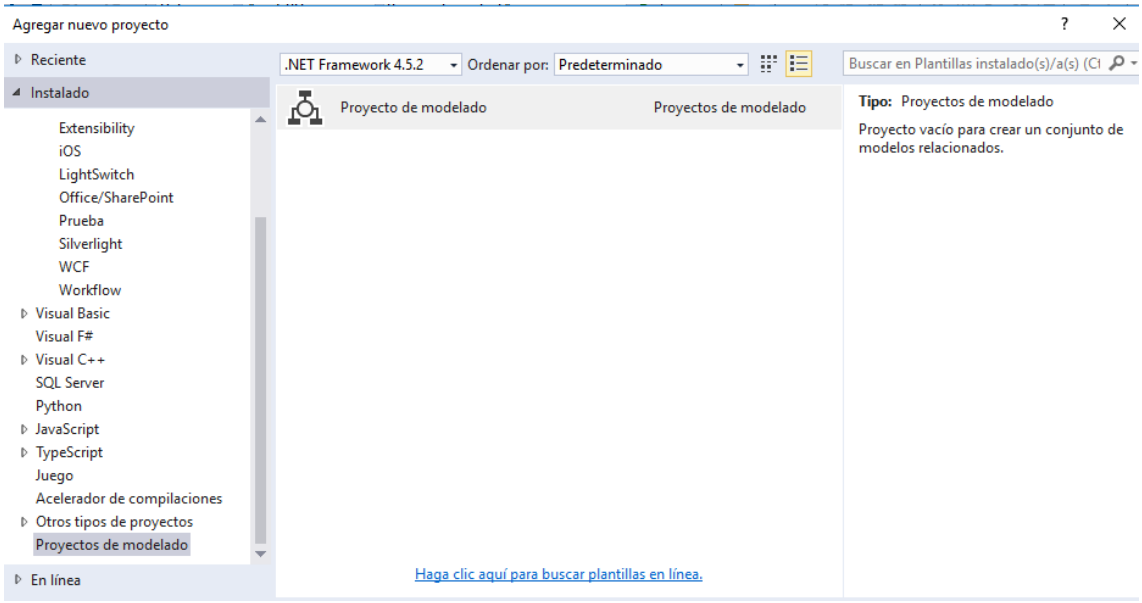

Figura 1. Creació d'un projecte de modelatge

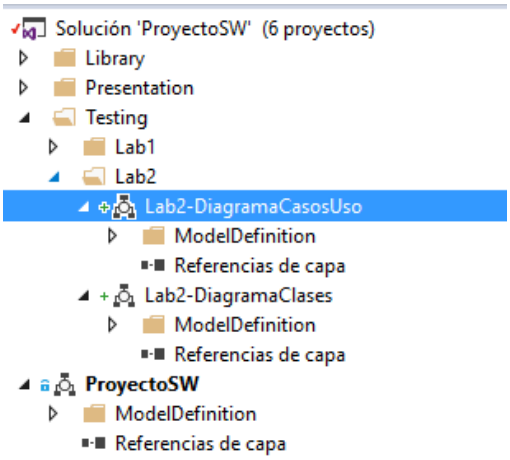

Figura 2. Estructura del nostre projecte després d'haver creat els dos projectes de modelatge (Lab2-DiagramaClases i Lab2-DiagramaCasosUso). NOTA: els noms de les soluciones/carpetes poden ser diferents als de la imatge presentada

3. Modelant diagrames amb Visual Studio

Una vegada el Team Leader ha creat els projectes de modelatge, els subequips poden treballar en paral·lel cadascun sobre el seu projecte de modelatge. En cada projecte agreguem un nou element (`Proyecto > Agregar nuevo elemento`) segons siga el diagrama de classes o el diagrama de casos d'ús (veure Figura 3). Òbviament, cada subequip treballarà amb un diagrama i agregarà el que li corresponga, siga el diagrama de classes o el diagrama de casos d'ús. Com s'ha indicat, aquesta forma d'assignar i distribuir les tasques evitarà que apareguen conflictes en el servidor en protegir la solució.

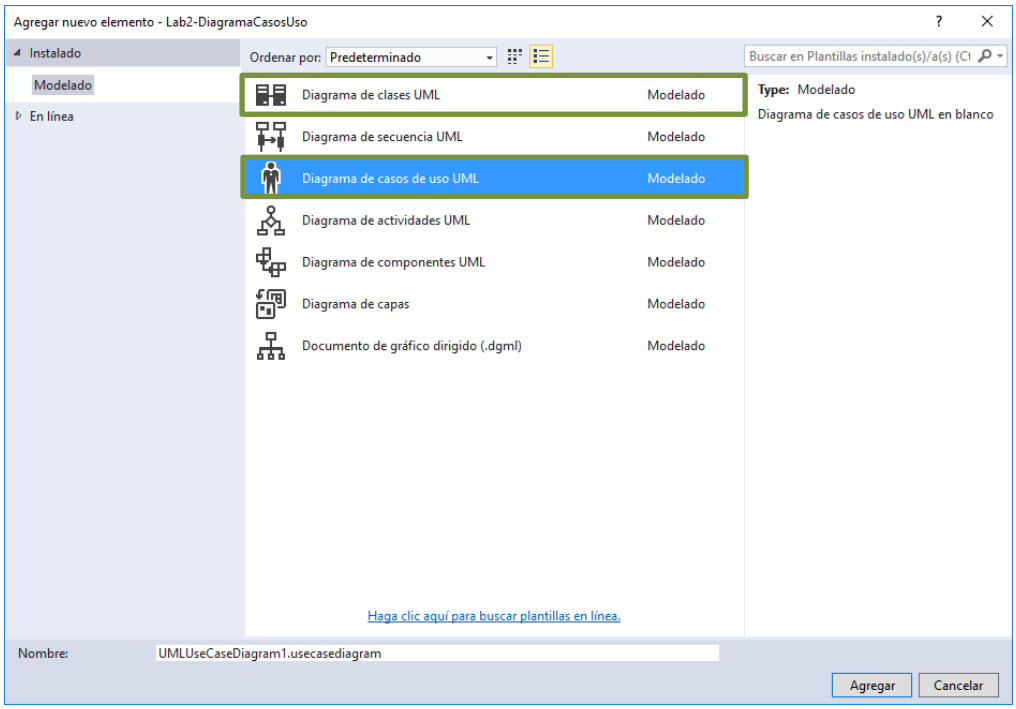

Figura 3. Agregant un nou diagrama de classes/casos d'ús UML

Una vegada agregats els diagrames veurem un àrea de treball buida en la qual podem arrossegar els controls o elements UML. Aquests controls apareixen en el quadre d'eines de l'esquerra, sent específics per al disseny d'un diagrama de classes o diagrama de casos d'ús, tal com es mostra en la Figura 4. Per exemple, en el diagrama de classes podem afegir classes, interfícies, relacions d'associació, agregació, etc. D'altra banda, en el diagrama de casos d'ús podem afegir actors, casos d'ús, relacions d'inclusió/extensió, etc.

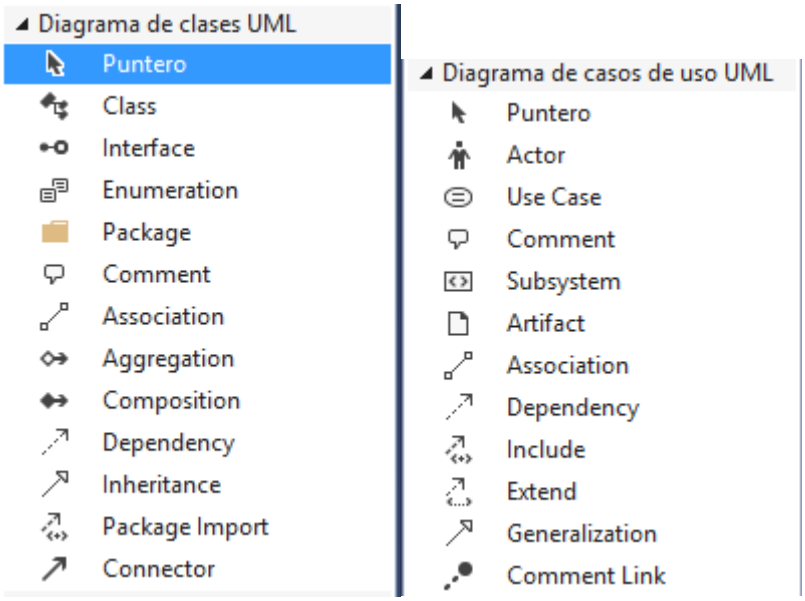

Figura 4. Quadre d'eines per a treballar amb un diagrama de classes o de casos d'ús, respectivament de Diagrama en edició

La forma de modificar aquests elements arrossegats al llenç és molt senzilla: n'hi ha prou amb editar les seues propietats (finestra en la part inferior dreta). Per exemple, podem editar el nom de la classe, el nom i tipus dels seus atributs, incloure associacions, agregacions, composicions

o herència entre classes. Similarment, podem editar el nom d'un cas d'ús, associar-li-ho a un actor, afegir relacions d'inclusió/extensió, etc.

Disseny de diagrames de casos d'ús

Una vegada arrosseguem els actors i els associem (relació Association en Visual Studio) als casos d'ús, l'única cosa que hem de fer és modificar les seues propietats. En un diagrama de casos d'ús aquestes propietats són molt simples i fàcils d'utilitzar, tal com s'observa en la Figura 5. Com es tracta d'un model gràfic, el mateix actor pot aparèixer representat diverses vegades si açò ajuda a visualitzar millor el model. De forma similar podem afegir relacions d'herència, d'inclusió i extensió.
NOTA: Visual Studio no permet afegir la condició en una relació d'extensió. Si es desitja afegir es pot agregar un comentari.

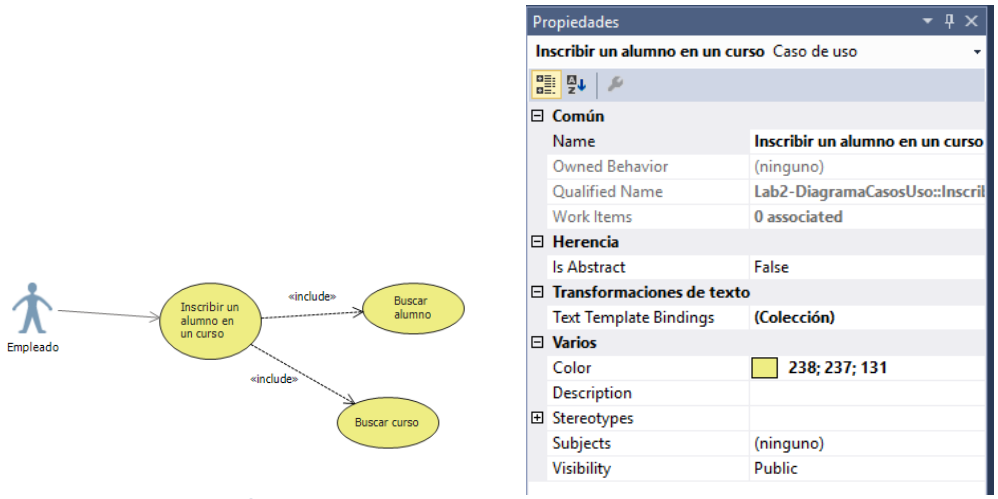

Figura 5. Exemple de cas d'ús "Inscriure un alumne en un curs" amb dues relacions de tipus "include"

Dissenyant un diagrama de classes

Una vegada arrosseguem les classes, hem d'afegir les relacions entre les mateixes, bé siguen associacions, agregacions o composicions. En qualsevol d'aquests tres casos és important indicar: nom de la relació, nom de rol en cadascun dels extrems, multiplicitat i navegabilitat. Els primers elements són acte-descriptius, mentre que la navegabilitat ens indica si una relació ens interessa fer-la bidireccional (isNavigable a False) o unidireccional (isNavigable a True). En el primer cas ens interessa mantenir que una classe es relaciona amb una altra i **també** al revés. En el segon cas solament ens interessa mantenir la relació en un sentit, però no en l'invers (per aquest motiu es diga que existeix una restricció de navegabilitat solament en un sentit). Gràficament, una relació bidireccional es visualitza com una línia sense fletxes en els extrems, i una relació unidireccional amb una fletxa en l'extrem/rol marcat com a navegable.

En la Figura 6 es mostra un exemple d'una relació de tipus composició entre dues classes. En primer lloc, disposem de les classes, amb els seus atributs i operacions/mètodes. Podem indicar si la classe és abstracta o no. Per a afegir un atribut/operació prou fer clic amb el botó dret sobre la classe i agregar l'atribut/operació que ens interesse, indicant totes les seues propietats com a nom, tipus, paràmetres, etc.

En el nostre exemple, un *Curso* es pot impartir en diverses Edicions, cadascuna d'elles representant un *CursoImpartido* en unes dates determinades. Açò es tradueix en una relació un-a-molts (un *Curso* a molts *CursosImpartidos*). En la Figura 6 es poden observar les propietats de la composició, amb el seu nom i els seus dos rols. Per a cada rol representem el seu nom, el seu cardinalitat i si és navegable o no.

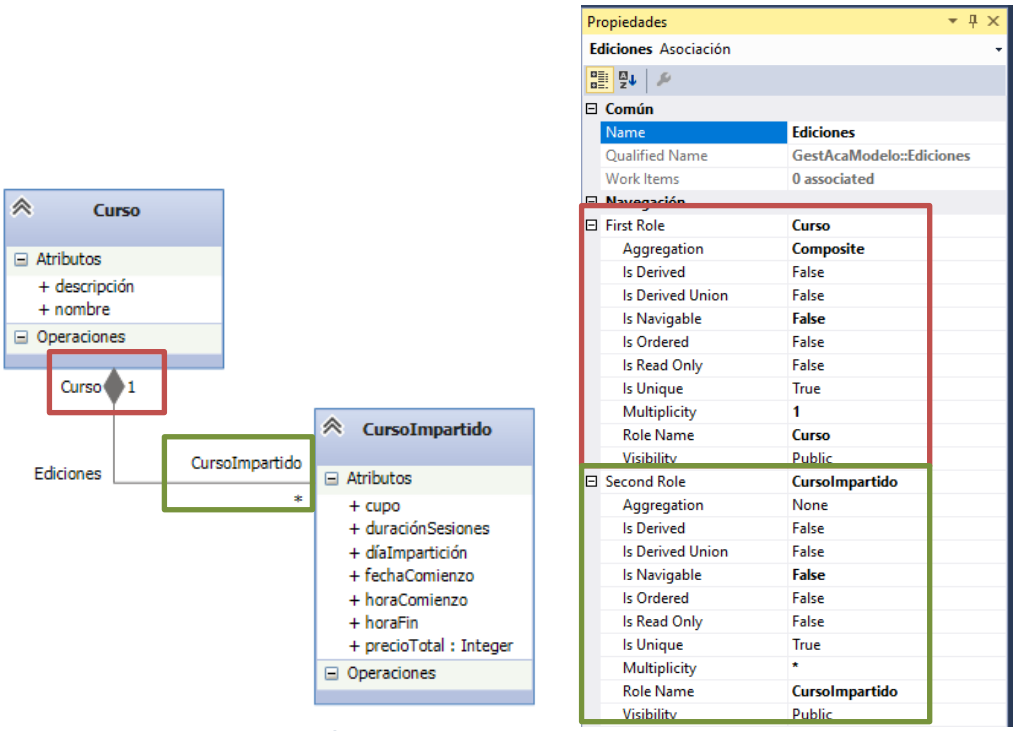

Figura 5. Exemple de composició bidireccional denominada Ediciones *que relaciona un* Curso *amb tots els seus* CursosImpartidos

4. Tasques a realitzar.

Treballarem amb un sistema que ha de gestionar una acadèmia on hi ha cursos, professors i alumnes. Les tasques a realitzar són les pròpies d'una acadèmia, com donar d'alta, realitzar cerques i llistats, així com gestionar les inscripcions, parts d'assistència, etc. Òbviament, ens interessa emmagatzemar la informació pròpia dels cursos, aula on s'imparteixen professors, parts d'assistència, etc.

Ens interessa utilitzar Visual Studio 2015 Enterprise per a modelar tant la part dinàmica com la part estàtica d'aquest sistema. Per tant, hem d'elaborar el diagrama de casos d'ús i el diagrama de classes UML mostrats en la Figura 7 i Figura 8, respectivament.

Important:

- **Aquesta tasca és un lliurament obligatori amb un pes del 10% respecte a la nota final de l'assignatura.**

- **Es lliurarà un document pdf amb els diagrames elaborats en PoliformaT (cada grup de laboratori tindrà habilitada una tasca en PoliformaT).**

- **Es realitzarà sobre *Team Services* una operació de protegir amb el comentari "Entregable 1" abans de la data de lliurament.**

- **Lliurament: El treball s'ha de lliurar abans de finalitzar la sessió de laboratori corresponent.**

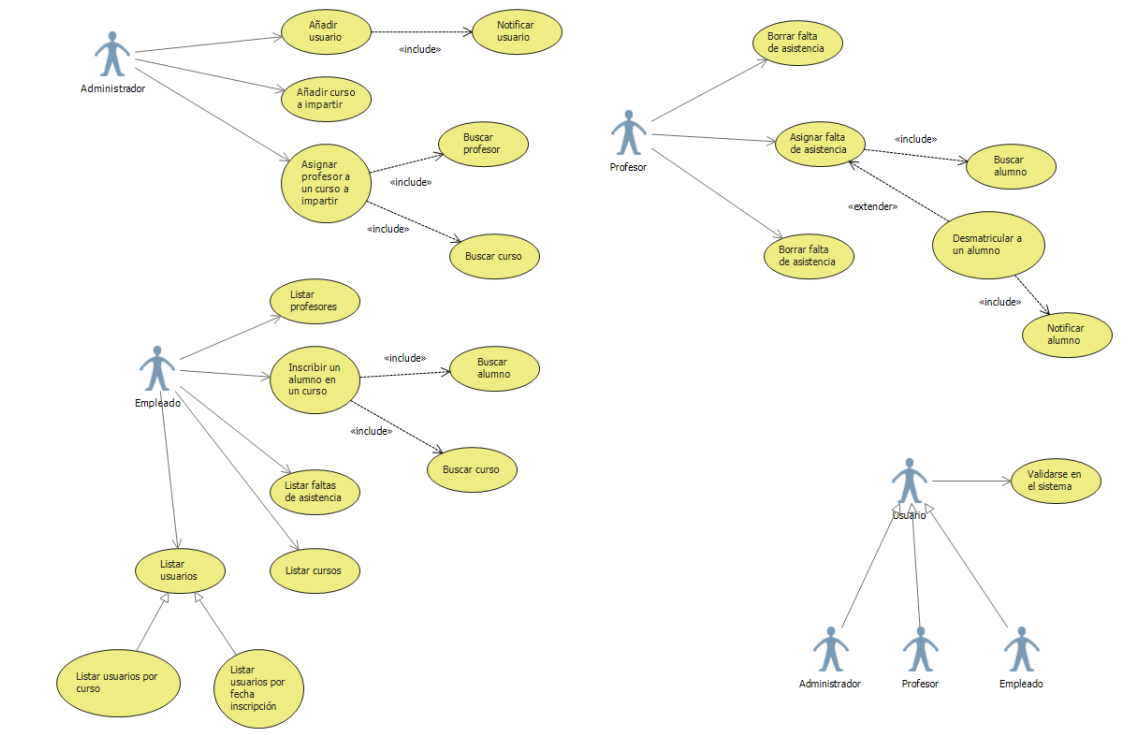

Figura 5. Diagrama de casos d'ús a modelar

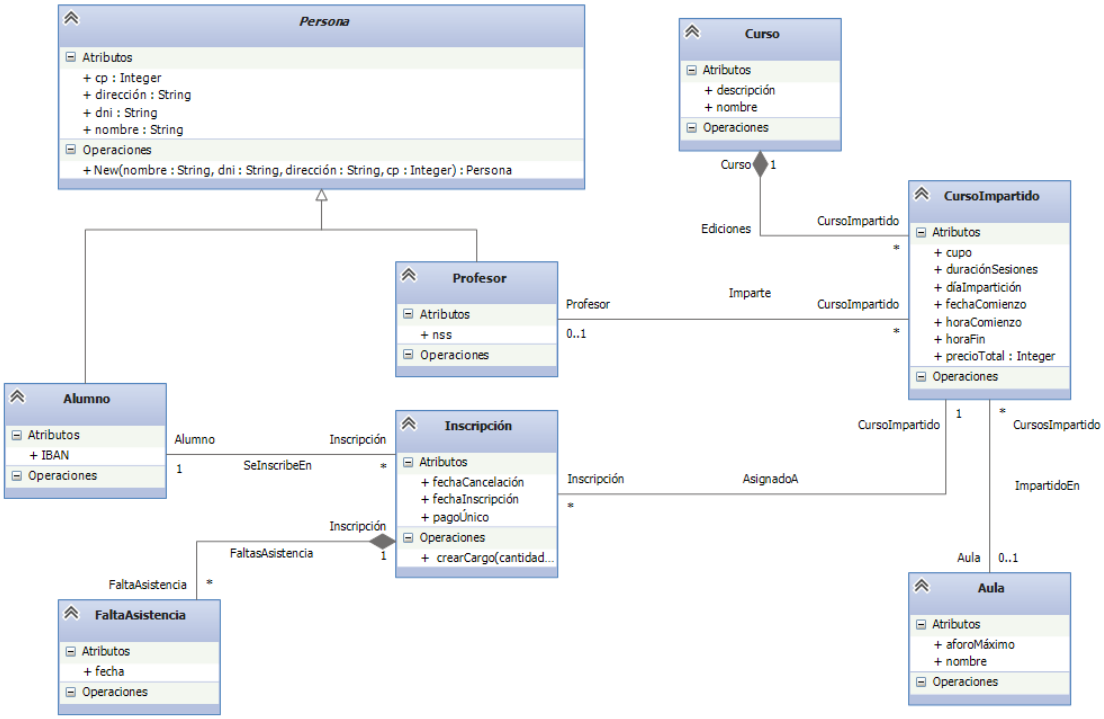

Figura 6. Diagrama de classes a modelar