Auxiliar IV

METODOLOGÍAS DE DISEÑO Y PROGRAMACIÓN

CC3002 @ 2009

Ejercicio 1

Construir los diagramas de colaboración necesarios para reflejar las siguientes interacciones:

a) Un objeto de clase *ClaseA* recibe como punto de entrada un mensaje *mensaje1()* y si la condición *cond* se satisface, envía un mensaje *mensaje2()* a un objeto de clase *ClaseB*, y en caso contrario, envía un mensaje *mensaje3()* a un objeto de clase *ClaseC*.

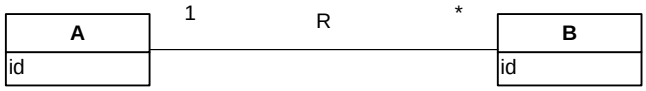
b) Un objeto de clase *ClaseA*, al recibir el mensaje *mensaje1()*, crea una instancia de clase *ClaseB*, y se lo envía a un objeto de clase *ClaseC*, el cual envía el mensaje *mensaje3()* a la instancia recién creada.

c) Un objeto de clase *ClaseA*, al recibir el mensaje *mensaje1()*, envía *m* veces el mensaje *mensaje2()* a una instancia de clase *ClaseB*, el cual a su vez, por cada mensaje recibido, envía un mensaje *mensaje3()* a un objeto de clase *ClaseC* y otro mensaje *mensaje4()*, a uno de clase *ClaseD*.

d) Un objeto de clase *ClaseA* recibe como punto de entrada un mensaje *mensaje1()*, y envía a cada objeto de una colección de objetos de clase *ClaseB* el mensaje *mensaje2()*.

Ejercicio 2

Considerando el siguiente modelo de dominio, realizar un diagrama de colaboración para cada una de las interacciones:



a) una instancia *a : A* crea una instancia *b : B* con un link unidireccional desde *a* hacia *b*.

b) una instancia *a : A* crea una instancia *b : B* con un link unidireccional desde *b* hacia *a*.

c) una instancia *b : B* crea una instancia *a : A* con un link unidireccional desde *b* hacia *a*.

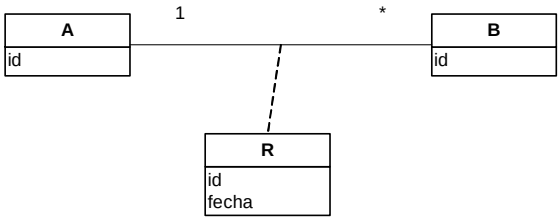
d) una instancia *b : B* crea una instancia *a : A* con un link unidireccional desde *a* hacia *b*.

e) una instancia *a : A* crea una instancia *b : B* con un link bidireccional.

f) una instancia *b : B* crea una instancia *a : A* con un link bidireccional.

Ejercicio 3

Considerar el siguiente modelo de dominio:



Diseñar la operación que retorne todos los ids de instancias de A que participen en la asociación R cuyas fechas sean mayores a una dada.

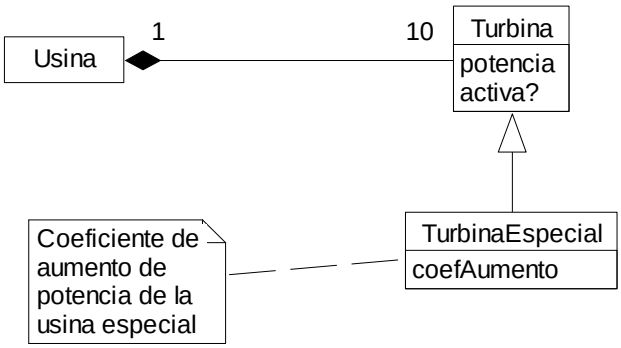
a) Realizar el diagrama de colaboración sabiendo que R es navegable de A a B.

b) Realizar el diagrama de colaboración sabiendo que R es navegable de B a A.

c) Realizar el diagrama de colaboración asignando la responsabilidad de realizar la operación a R.

Ejercicio 4

Considerando el siguiente modelo de dominio, realizar el diagrama de colaboración con el mensaje potenciaUsina() como punto de entrada. Esta operación debe devolver la potencia total que son capaces de generar todas las turbinas activas en ese momento.



Ejercicio 5

Considerar la siguiente realidad:

> **Visión**
> El Ministerio de Salud ha decidido verificar la calidad de los medicamentos producidos por ciertas empresas del sector. A tales efectos se ha dispuesto que sus inspectores estudien la composición de los medicamentos producidos por estas empresas (realizando un análisis químico en un laboratorio del ministerio), a fin de comprobar que la misma coincida con la especificada para cada medicamento.

Se quiere diseñar una operación mediante la cual el jefe de la investigación le pide a un inspector que inspeccione algunas de las empresas, verificando la calidad de una muestra de los medicamentos producidos por esa empresa. El inspector debe entregar un informe en el que conste, para cada empresa inspeccionada, el nombre de todos los medicamentos verificados junto con el porcentaje de medicamentos defectuosos de cada tipo.

El informe debe permitir llenar fácilmente el siguiente formulario:

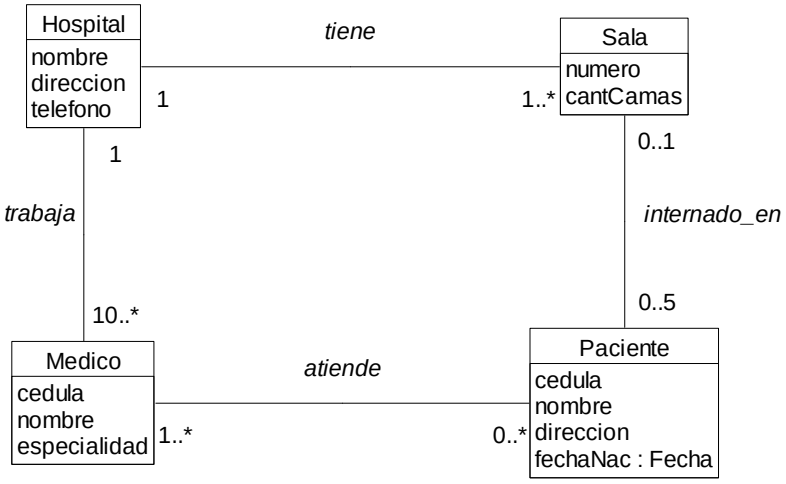
> **Informe**
>
> *Nombre Empresa A*
>
> Nombre Medicamento A Porcentaje defectuoso
>
> Nombre Medicamento B Porcentaje defectuoso
>
> …
>
> *Nombre Empresa B*
>
> …

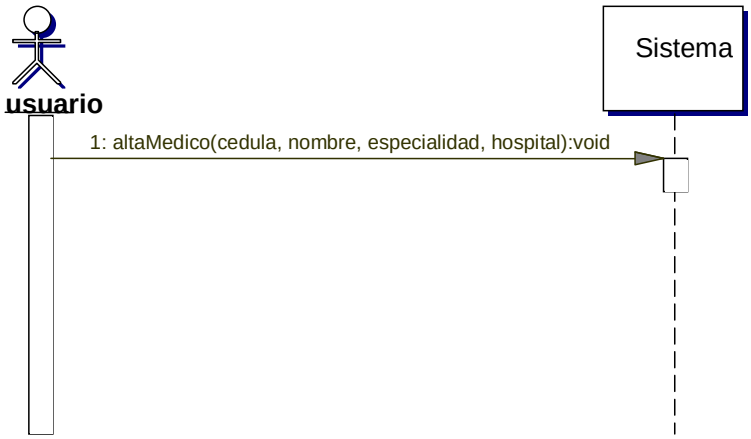
Representar la operación indicada mediante diagramas de colaboración en UML de manera que refleje fielmente la visión (tal como ocurre en el dominio del problema, sin tener en cuenta consideraciones de diseño).

Nota: Suponer existentes todos los conceptos que necesite, como por ejemplo: Medicamento, Inspector, Informe, Empresa, Laboratorio.

Ejercicio 6

Considerando los siguientes artefactos de análisis, realizar el diseño de la operación del sistema:

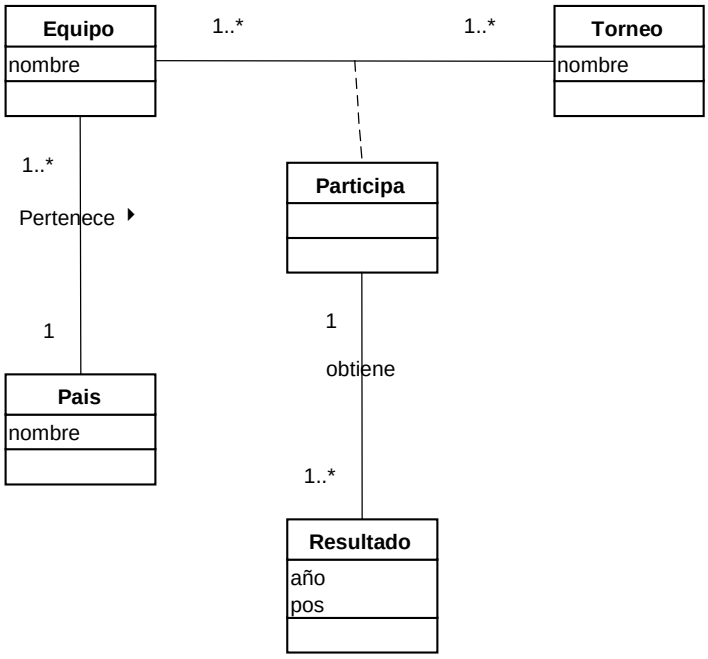


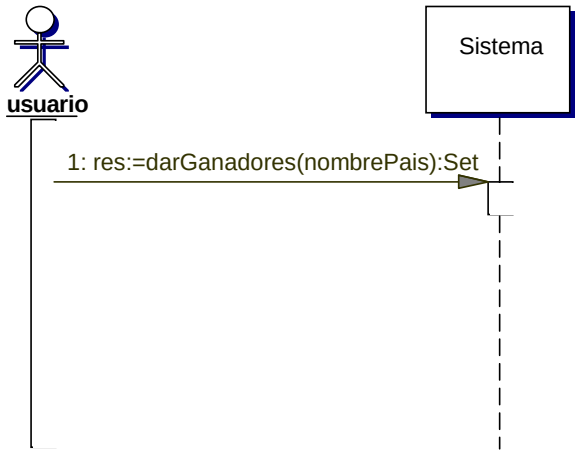


Nombre	**Alta Médico**
Operación	altaMedico(cedula:String, nombre:String, hospital:String, especialidad:String)
Entrada	*cedula* es la cédula del médico, *nombre* es el nombre del médico, *hospital* es el identificador del hospital en el que trabaja el médico, y *especialidad* es la especialidad del médico.
Salida	No tiene.
Descripción	Dar de alta el médico con los datos dados (*cedula*, *nombre*, *especialidad*) e indicar que trabaja en el hospital identificado por *hospital*.
Precondiciones y Postcondiciones	
context System::altaMedico(cedula:String, nombre:String, hospital:String, especialidad:String) **pre1**: El médico identificado por *cedula* no existe en el sistema. **pre2:** El hospital identificado por *hospital* existe en el sistema. **post:** Existe una instancia de Médico cuyos atributos son iguales a los dados y existe un link entre esta instancia y el Hospital correspondiente al nombre dado.	

Ejercicio 7

Considerando los siguientes artefactos de análisis, realizar el diseño de las operaciones del sistema:





Nombre	**Dar Equipos Ganadores**
Operación	darGanadores(nombrePais:String):Set(String)
Entrada	*nombrePais* es el nombre del país del cual se desean conocer los resultados.
Salida	El conjunto de nombres de los equipos que han ganado algún torneo en el país de nombre *nombrePais*.
Descripción	Lista todos los equipos del país dado por *nombrePais* que han ganado algún torneo. Es posible que éste conjunto sea vacío.

Precondiciones y Postcondiciones

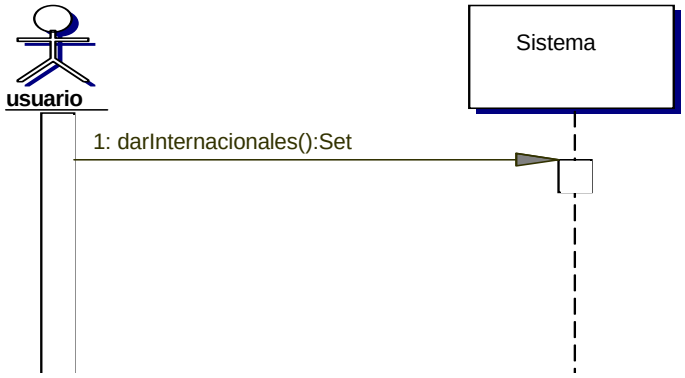
context System::darGanadores(nombrePais:String):Set(String)

pre: El país dado por *nombrePais* existe en el sistema.

post: El conjunto resultado contiene los nombres de todos los equipos del país dado por *nombrePais* que han ganado algún torneo. Son todos los equipos tales que existe alguna instancia de Participa que cumple:

- la instancia de Participa relaciona al equipo con algún torneo, y
- la instancia de Participa tiene al menos un link con una instancia de Resultado tal que *pos = 1*.

El estado del sistema no cambia.



Nombre	**Dar Torneos Internacionales**
Operación	darInternacionales():Set(String)
Entrada	No tiene.
Salida	El conjunto de nombres de los torneos internacionales. Es posible que dicho conjunto sea vacío.
Descripción	Lista todos los torneos internacionales.

Precondiciones y Postcondiciones

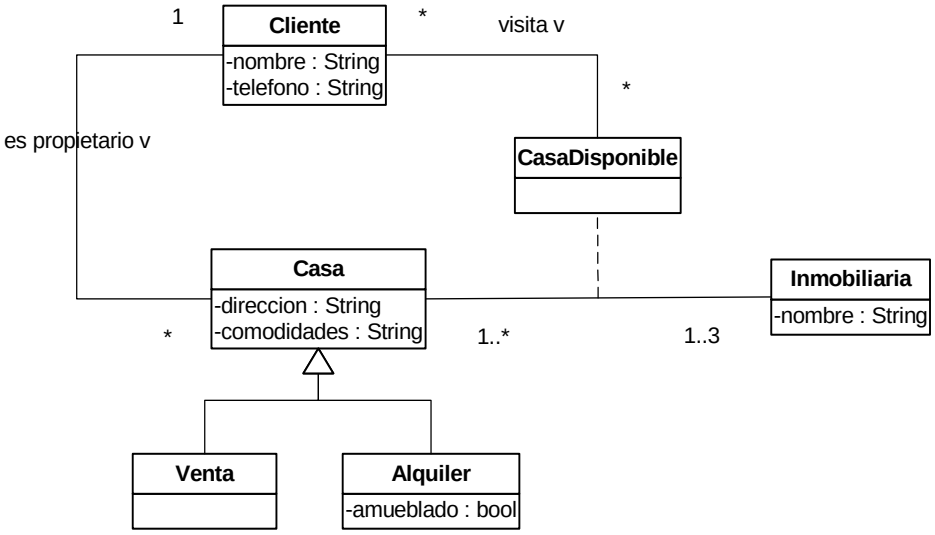
context System::darInternacionales():Set(String)

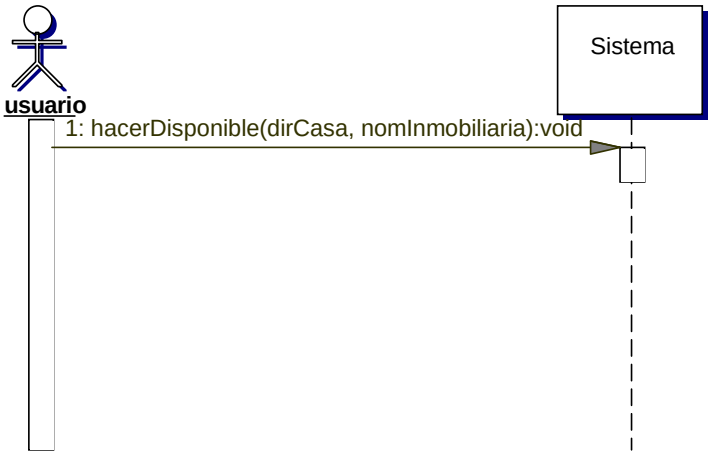
post: El conjunto resultado contiene los nombres de todos los torneos tales que no todos los equipos que participan en ellos pertenecen al mismo país.

El estado del sistema no cambia.

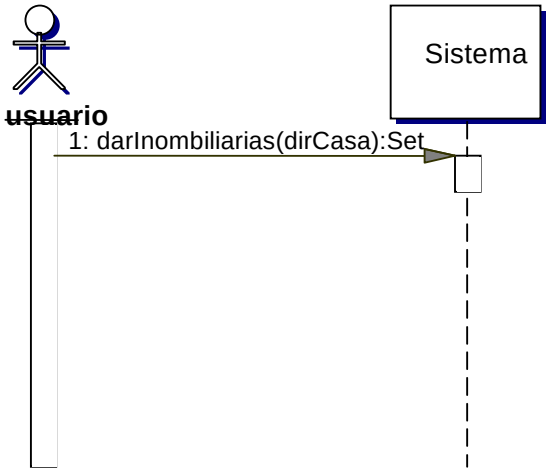
Ejercicio 8

Considerando los siguientes artefactos de análisis, realizar el diseño de las operaciones del sistema:

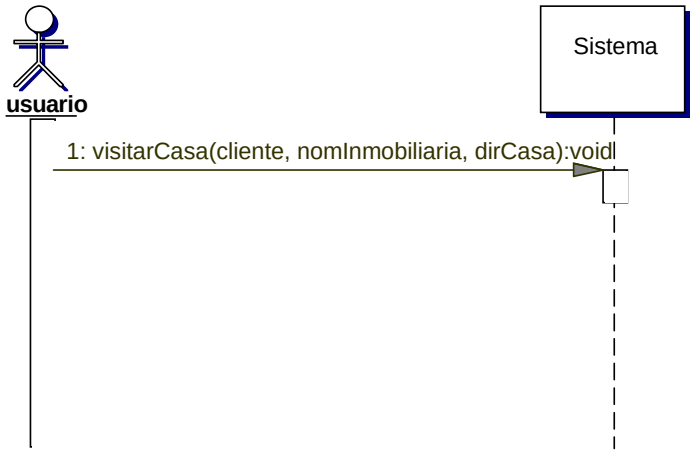




Nombre	**Hacer Casa Disponible**
Operación	hacerDisponible(dirCasa:String, nomInmobiliaria:String)
Entrada	*dirCasa* es la dirección de la casa que se desea hacer disponible, y *nomInmobiliaria* es el nombre de la inmobiliaria que tendrá disponible a dicha casa.
Salida	No tiene.
Descripción	Hacer que la casa identificada por *dirCasa* quede disponible dentro de la inmobiliaria dada por *nomInmobiliaria*.
Precondiciones y Postcondiciones	
context System::hacerDisponible(dirCasa:String, nomInmobiliaria:String) **pre1:** La casa dada por *dirCasa* existe en el sistema. **pre2:** La inmobiliaria dada por *nomInmobiliaria* existe en el sistema. **post:** Existe un link entre la casa dada por *dirCasa* y la inmobiliaria dada por *nomInmobiliaria*.	



Nombre	**Dar Inmoniliarias Con Disponibilidad De Casa**
Operación	darInmobiliarias(dirCasa:String):Set(String)
Entrada	*dirCasa* es la dirección de la casa de la cual se desea conocer las inmobiliarias
Salida	El conjunto de inmobiliarias que tienen a la casa disponible.
Descripción	Lista los nombres de todas las inmobiliarias que tienen disponible la casa dada por *dirCasa*.
Precondiciones y Postcondiciones	
context System::darInmobiliarias(dirCasa:String):Set(String) **pre:** La casa dada por *dirCasa* existe en el sistema. **post:** El conjunto de salida debe contiene los nombres de todas las inmobiliarias que tienen disponible la casa dada por *dirCasa*. El estado del sistema no cambia.	



Nombre	**Visitar Casa**
Operación	visitarCasa(cliente:String, nomInmobiliaria:String, dirCasa:String)
Entrada	*cliente* es el identificador del cliente que visitará la casa, *nomInmobiliaria* es el nombre de la inmobiliaria a través de la cual se visitará la casa, y *dirCasa* es la dirección de la casa a visitar.
Salida	No tiene.
Descripción	Indica al sistema que el cliente dado visita la casa dada por medio de la inmobiliaria dada.
Precondiciones y Postcondiciones	
context System::visitarCasa(cliente:String, nomInmobiliaria:String, dirCasa:String) **pre1:** La casa dada por *dirCasa* existe en el sistema. **pre2:** La inmobiliaria identificada por *nomInmobiliaria* existe en el sistema. **pre3:** El cliente *cliente* existe en el sistema. **post:** Existe un link entre el cliente *cliente*, la inmobiliaria *nomInmobiliaria* y la casa *dirCasa* que representa que el cliente visita la casa mostrada por esa inmobiliaria.	

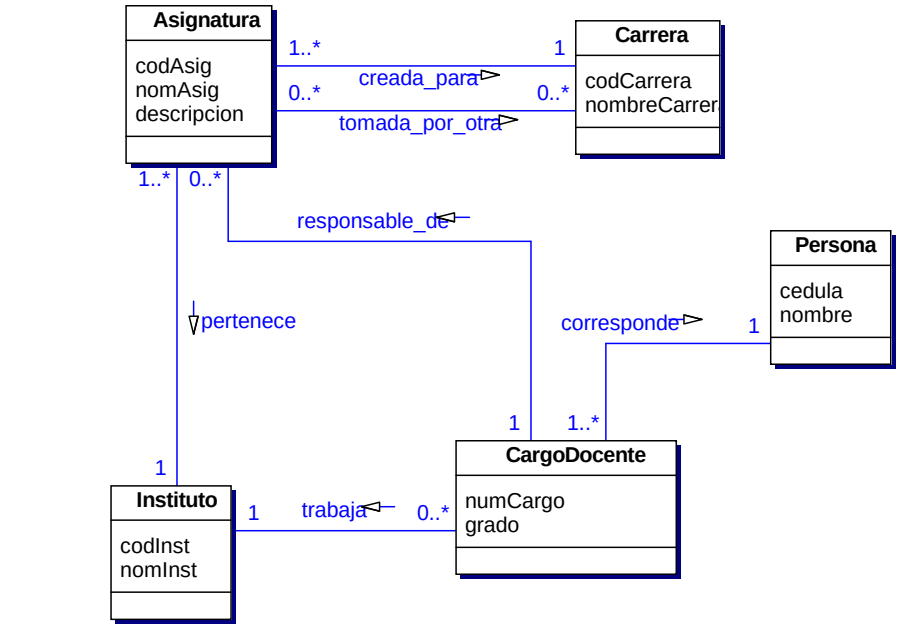
Ejercicio 9

Considerar los siguientes artefactos de requisitos y análisis:

Visión

La Facultad de Ingeniería está organizada en Institutos. Cada Instituto cuenta con un plantel docente que realiza tareas de enseñanza, entre otras cosas. Las asignaturas que se dictan en la facultad son responsabilidad de los institutos; cada asignatura pertenece a un único instituto, éste es el responsable de crear las asignaturas a su cargo, generando su contenido y demás detalles, asociándole un docente como responsable desde ese momento. Las asignaturas son creadas para su uso por una carrera determinada de la facultad; después que existen, sin embargo, pueden ser tomadas por otras carreras. A la facultad no le interesa mantener información de las asignaturas, cuando la necesita, se la pide al instituto que corresponda.

Las personas que trabajan como docentes pueden hacerlo en varios institutos (por lo menos en uno). Estas personas ocupan cargos docentes en cada instituto. Los cargos son propiedad de los institutos, o sea que un cargo docente pertenece a un único instituto. De los cargos docentes se conoce el número de cargo y el grado.



Nombre	**Alta Asignatura**
Operación	altaAsignatura(codigoInstituto:String, codAsig:String, nomAsig:String, descAsig:String, codCarrera:String, numCargoDocente:int)
Entrada	*codigoInstituto* es el código del instituto a la que pertenece la nueva asignatura, *codAsig* es el código de la nueva asignatura, *nomAsig* es el nombre de la nueva asignatura, *descAsig* es la descripción de la nueva asignatura, *codCarrera* es el código de la carrera a la cual pertenece la nueva asignatura, y *numCargoDocente* es el número de cargo del docente que dictará la materia.
Salida	No tiene.
Descripción	Da de alta la asignatura.
Precondiciones y Postcondiciones	
context System::altaAsignatura(codigoInstituto:String, codAsig:String, nomAsig:String, descAsig:String, codCarrera:String, numCargoDocente:int) **pre1:** El instituto identificado por *codigoInstituto* existe en el sistema. **pre2:** El cargo docente identificado por *numCargoDocente* existe en el sistema. **pre3:** La carrera identificada por *codCarrera* existe en el sistema. **pre4:** En el sistema no existe ninguna asignatura cuyo código de asignatura sea *codAsig*. **post:** Existe una instancia de Asignatura cuyos atributos son los dados (*codAsig*, *nomAsig* y *descAsig*). Existe un link entre la instancia de Asignatura correspondiente a *codAsig* y el instituto identificado por *codigoInstituto*. Existe un link entre la instancia de Asignatura correspondiente a *codAsig* y la carrera identificada por *codCarrera* mediante la asociación *Creada_para*. Existe un link entre la instancia de Asignatura correspondiente a *codAsig* y el cargo docente identificado por *numCargoDocente*.	

Se pide:

a) Diseñar la operación *altaAsignatura()*.

b) Explicar en el diagrama cuáles criterios de asignación de responsabilidades fueron utilizados.

Ejercicio 10

Se quiere modelar y diseñar un proceso de armado y registro de tours turísticos a medida para los clientes de una agencia de viajes. Un tour turístico para la agencia consiste en:

- la reserva de vuelos de aviones hacia las ciudades que componen el tour
- la reserva de hoteles en las ciudades y de los paseos locales que dichos hoteles ofrecen
- la información que indique si el tour fue pagado o no.

Una particularidad de los paseos locales es que ellos se realizan todos los días del año y no hay límite en la cantidad de personas que pueden hacerlos.

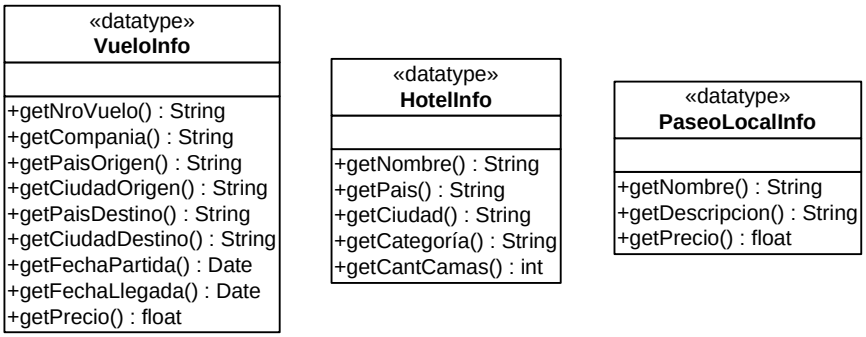
La descripción del escenario típico del proceso es la siguiente:

Actor	Sistema
1. Inicia un tour turístico para un cliente dada su cédula de identidad, ciudad de partida (identificada por su nombre y país), fecha de comienzo y cantidad de personas	
	2. Inicia el tour con los datos ingresados
3. Para cada ciudad (ver nota 1) que el cliente desea visitar:	
3a. Pide los vuelos con lugares disponibles para la fecha (ver nota 2)	
	3b. Presenta los vuelos pedidos de acuerdo a la cantidad de asientos requeridos
3c. Reserva un vuelo	
	3d. Registra la reserva del vuelo
3e. Pide la lista de hoteles con lugares disponibles dando la cantidad de días de la estadía en la ciudad	
	3f. Presenta los hoteles pedidos de acuerdo a la cantidad de personas a hospedar
3g. Reserva un hotel	
	3h. Registra la reserva del hotel
3i. Pide la lista de paseos locales ofrecidos por el hotel reservado	
	3j. Presenta los paseos pedidos
3k. Reserva cero o más paseos	
	3l. Reserva el(los) paseos indicados
4. Indica terminación del itinerario del tour	
	5. Presenta el valor del tour
6. Indica si paga ahora o no	
	7. Registra si se hace el pago o no
8. Termina el tour turístico	

Notas:

1. Para simplificar el proceso, no se requiere que el sistema liste ciudades en ningún momento. Las ciudades que el actor provee al sistema son ciudades válidas para el sistema. Asumir lo mismo para los clientes.
2. A excepción de la ciudad de partida del tour, esta fecha se determina teniendo en cuenta la fecha de llegada a una ciudad y la cantidad de días de la estadía.

Para modelar el proceso, se han definido y se encuentran a disposición los siguientes datatypes representando la información sobre vuelos, hoteles y paseos respectivamente: VueloInfo, HotelInfo, PaseoLocalInfo.



Asumir que los vuelos quedan identificados por su número de vuelo, país de origen, ciudad de origen y fecha de partida; los hoteles por su nombre, ciudad y país; y los paseos, dentro de un hotel, por su nombre.

Se pide:

a) Realizar un Diagrama de Secuencia del Sistema (DSS) para este proceso en cada una de las siguientes situaciones.

   i. El sistema mantiene el estado de dicho proceso de manera que cada operación provea solamente los datos del paso en que se encuentra.

   ii. El sistema no mantiene estado.

   Debe especificarse por completo la signatura de cada una de las operaciones del sistema y dar una breve descripción de los parámetros referentes a las fechas y ciudades.

b) Se quiere estructurar la capa lógica en dos subcapas.

   - Capa Alta: Se encarga de controlar que se respete el orden de las operaciones definido en el DSS de la parte *(a.i)*.
   - Capa Baja: Se encarga de manejar los datos del modelo de dominio (las operaciones de esta capa son las identificadas en el DSS de la parte (a.ii).

   Asumir que se encuentran definidos los siguientes elementos correspondientes a la Capa Baja:

Figura 2

   Con estos nuevos elementos:

   i. Considerar las operaciones de la Capa Alta identificadas en el DSS de la parte *(a.i)* involucradas con los pasos 1 al 3d y con los pasos 4 al 5 del proceso. Realizar el diagrama de colaboración de dichas operaciones utilizando las operaciones de la Capa Baja para llevar a cabo sus responsabilidades.

   ii. Realizar el diagrama de colaboración de la operación de la fábrica de la capa alta encargada de devolver el controlador de dicha capa.

   Observación: Todo lo referente a la Capa Baja ya está definido y por lo tanto NO debe ser diseñado.