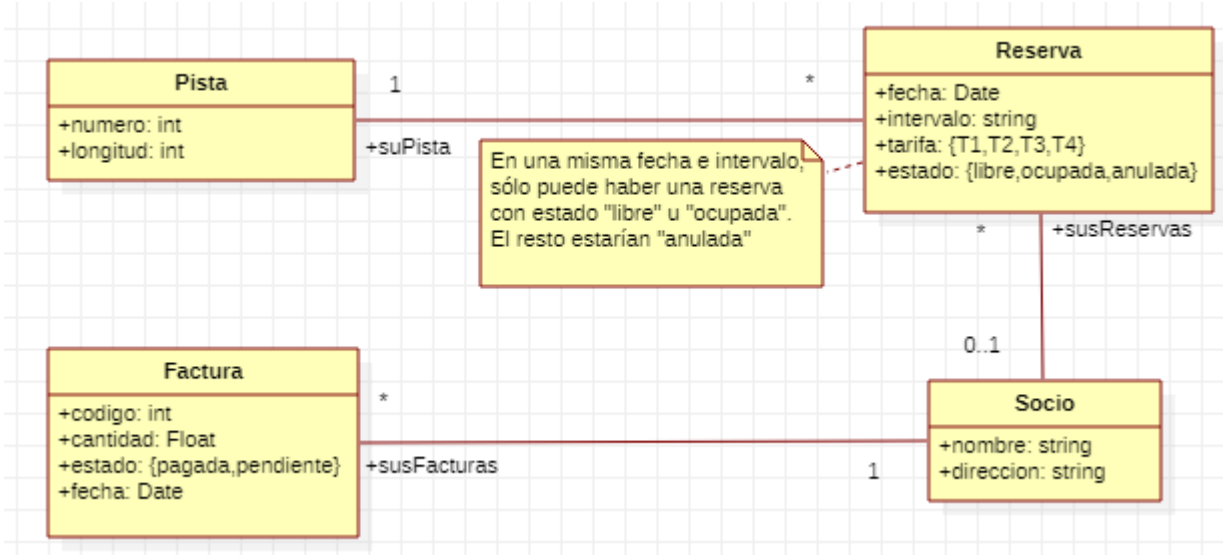
Diseño CU “Reservar Pista”

Flujo de Eventos: Reservar Pista

1. El socio proporciona su nombre y la fecha de reserva
2. El sistema muestra la ocupación de las pistas en esa fecha: por cada pista, a qué horas está libre
3. El socio escoge una hora libre de una pista
4. Se guarda la reserva: nombre del socio, fecha, hora, pista y la tarifa que corresponda (T1 si es fin de semana, o bien hora nocturna de un día laborable; o T2 en otro caso)

Flujos de Eventos Alternativos

- Si la fecha no es dentro de 30 días, no se puede reservar. Fin
- Si las reservas de ese día no se han creado, se crean y se continúa con el flujo de eventos normal



Flujo de Eventos: Reservar Pista

1. El socio proporciona su nombre y la fecha de reserva
2. El sistema muestra la ocupación de las pistas en esa fecha: por cada pista, a qué horas está libre
3. El socio escoge una hora libre de una pista
4. Se guarda la reserva: nombre del socio, fecha, hora, pista y la tarifa que corresponda (T1 si es fin de semana, o bien hora nocturna de un día laborable; o T2 en otro caso)

Flujos de Eventos Alternativos

- Si la fecha no es dentro de 30 días, no se puede reservar. Fin
- Si las reservas de ese día no se han creado, se crean y se continúa con el flujo de eventos normal

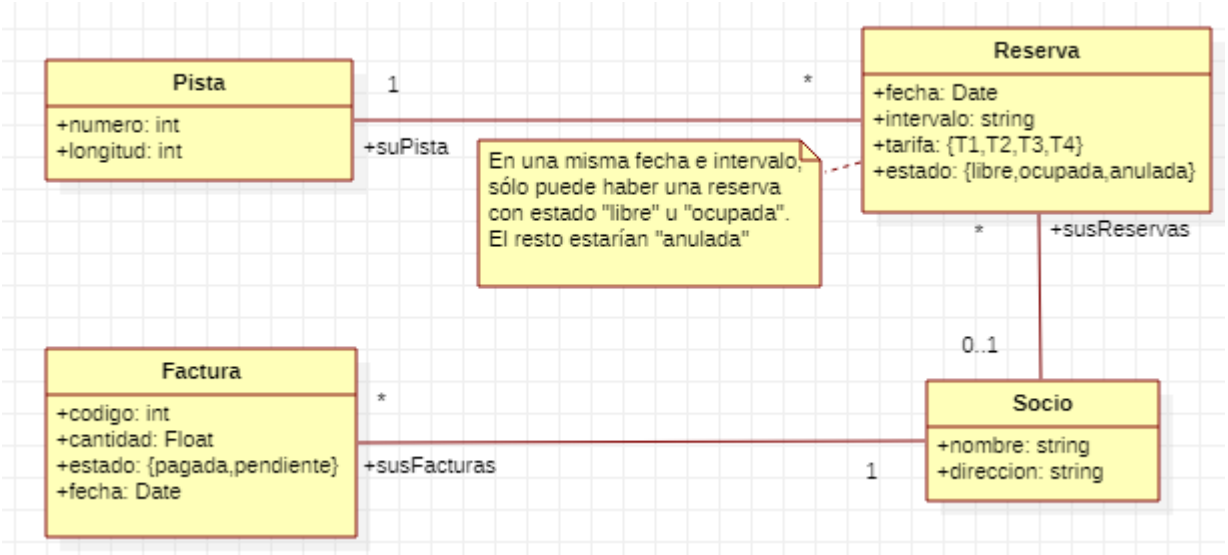
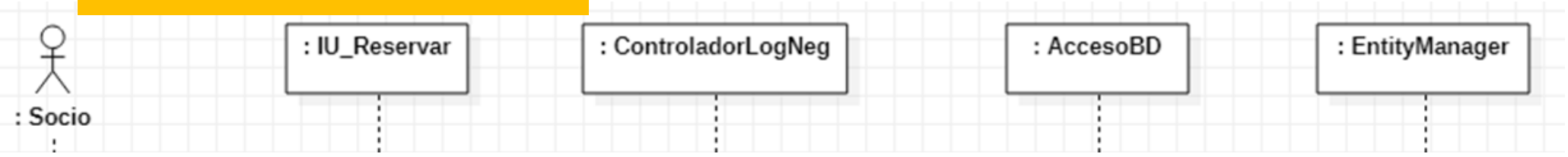


DIAGRAMA DE SECUENCIA



Flujo de Eventos: Reservar Pista

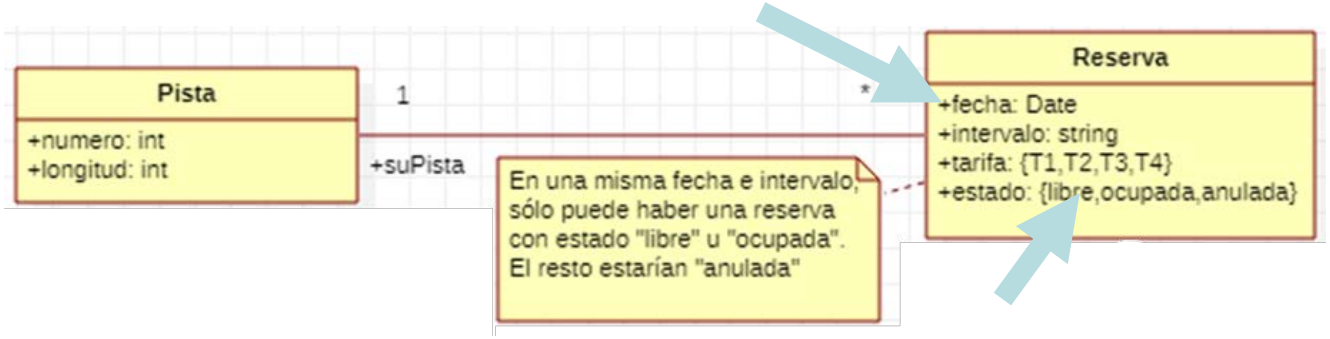
1. El socio proporciona su nombre y la fecha de reserva

DIAGRAMA DE SECUENCIA

FLUJO DE EVENTOS

2. El sistema muestra la ocupación de las pistas en esa fecha: por cada pista, a qué horas está libre

MODELO DEL DOMINIO



FLUJO DE EVENTOS

2. El sistema muestra la ocupación de las pistas en esa fecha: por cada pista, a qué horas está libre

MODELO DEL DOMINIO

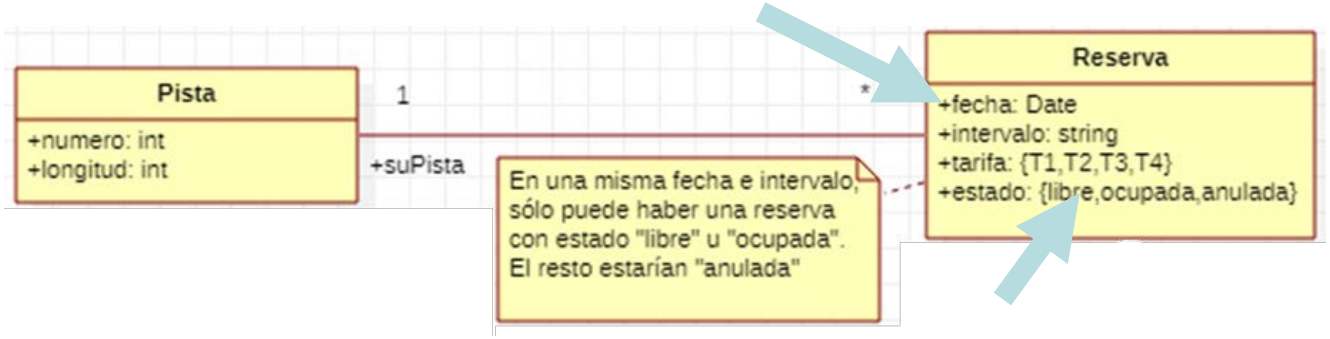
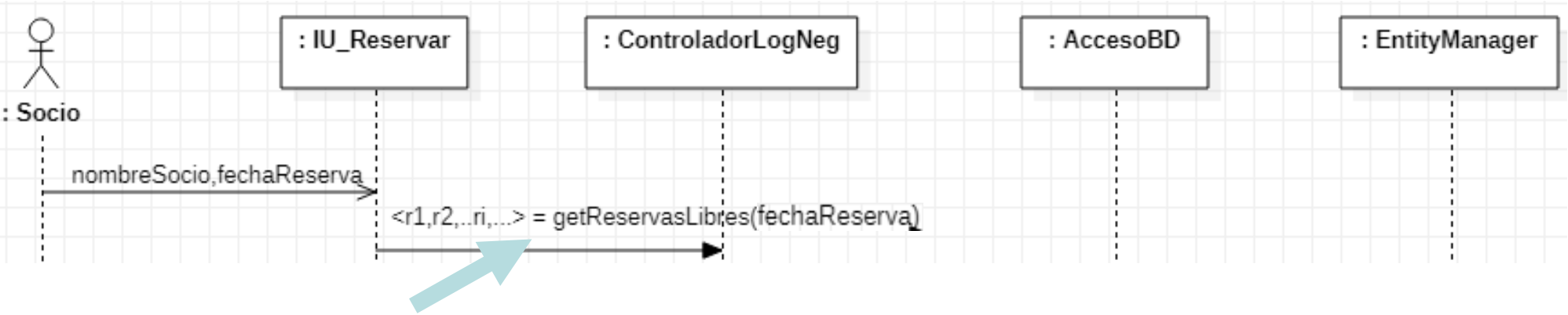


DIAGRAMA DE SECUENCIA



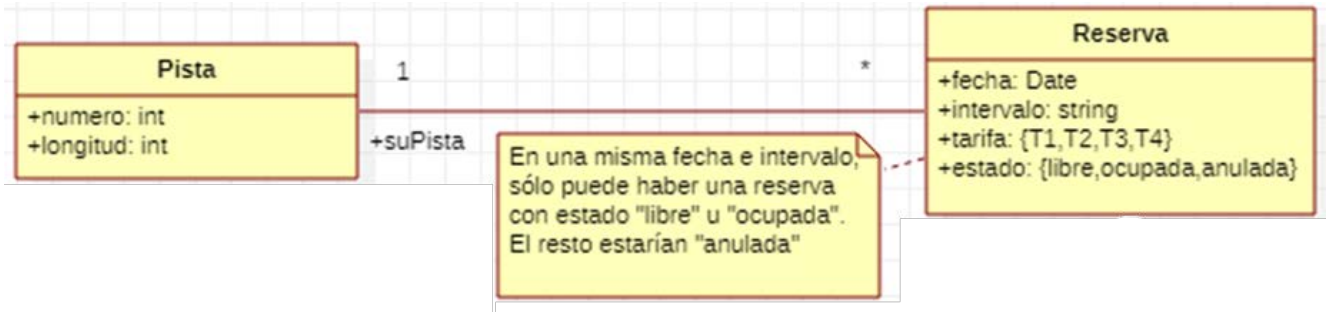
FLUJO DE EVENTOS

2. El sistema muestra la ocupación de las pistas en esa fecha: por cada pista, a qué horas está libre

Flujos de Eventos Alternativos

- Si la fecha no es dentro de 30 días, no se puede reservar. Fin

MODELO DEL DOMINIO



FLUJO DE EVENTOS

2. El sistema muestra la ocupación de las pistas en esa fecha: por cada pista, a qué horas está libre

Flujos de Eventos Alternativos

- Si la fecha no es dentro de 30 días, no se puede reservar. Fin

MODELO DEL DOMINIO

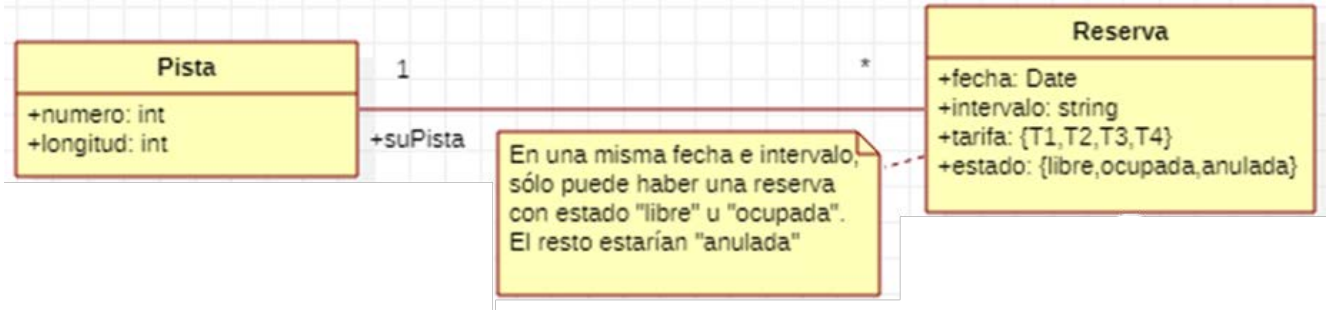
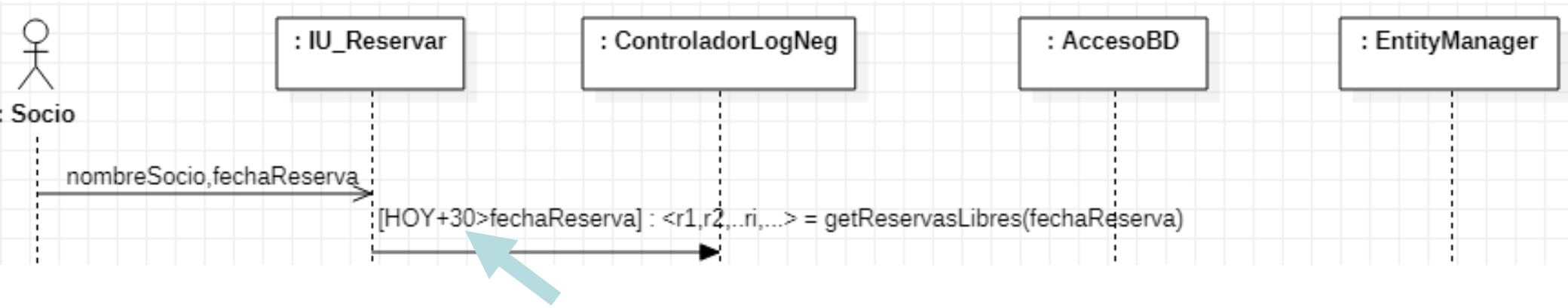


DIAGRAMA DE SECUENCIA



FLUJO DE EVENTOS

2. El sistema muestra la ocupación de las pistas en esa fecha: por cada pista, a qué horas está libre

Flujos de Eventos Alternativos

- Si la fecha no es dentro de 30 días, no se puede reservar. Fin

MODELO DEL DOMINIO

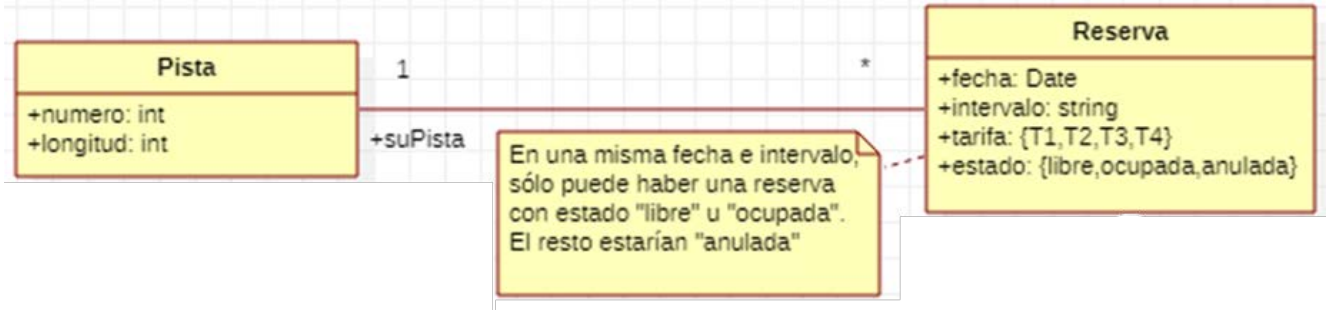
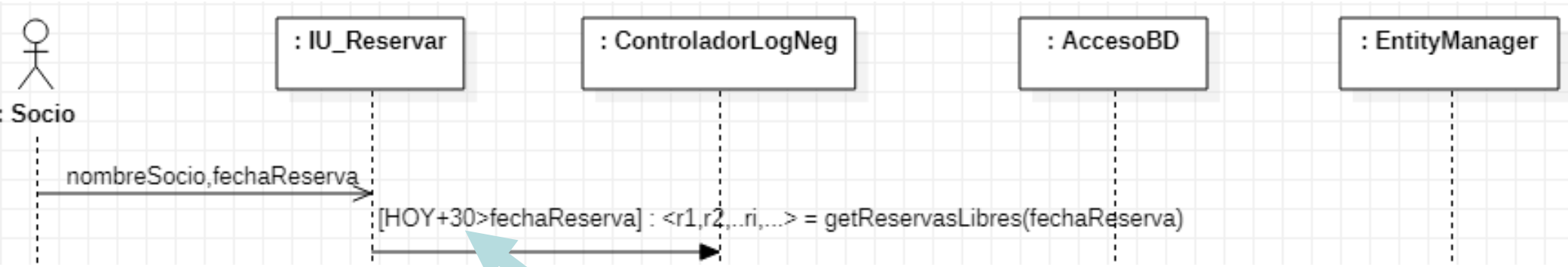


DIAGRAMA DE SECUENCIA



Esa validación de la entrada se puede hacer en la presentación o en la LN…

MODELO DEL DOMINIO

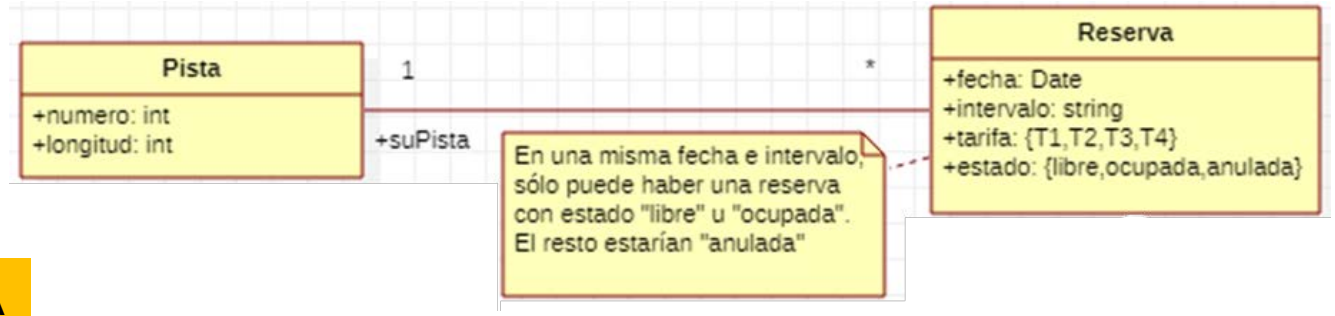
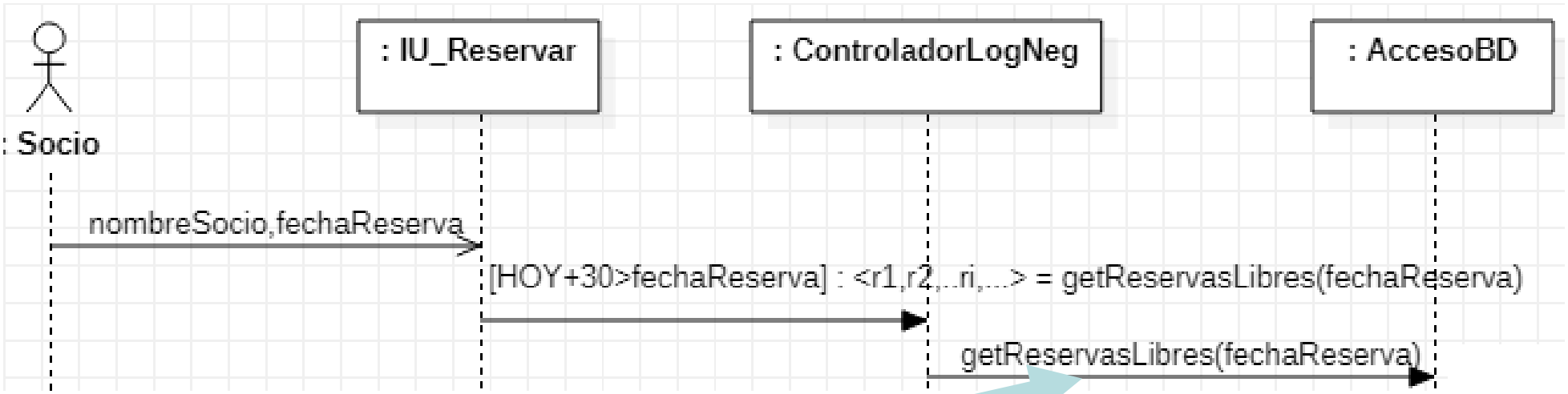


DIAGRAMA DE SECUENCIA



La lógica del negocio delega en el AccesoBD porque claramente las reservas libres habrá que buscarlas en la BD. Método homónimo en AccesoBD

MODELO DEL DOMINIO

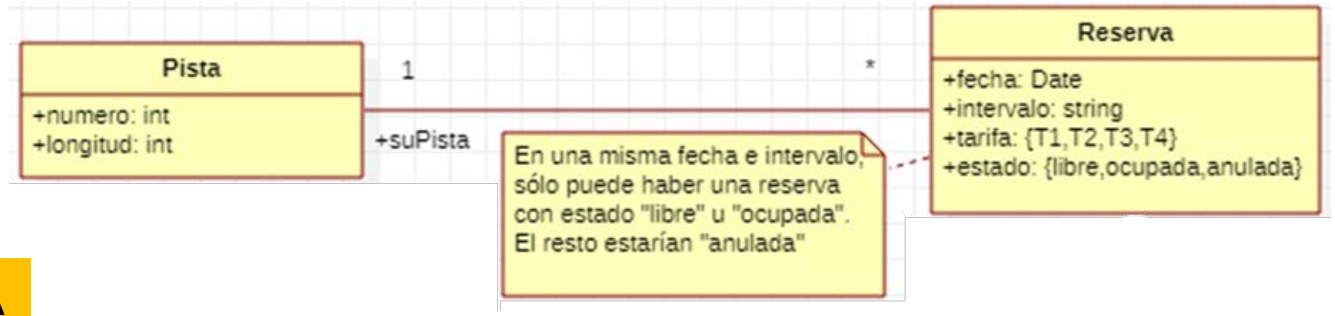
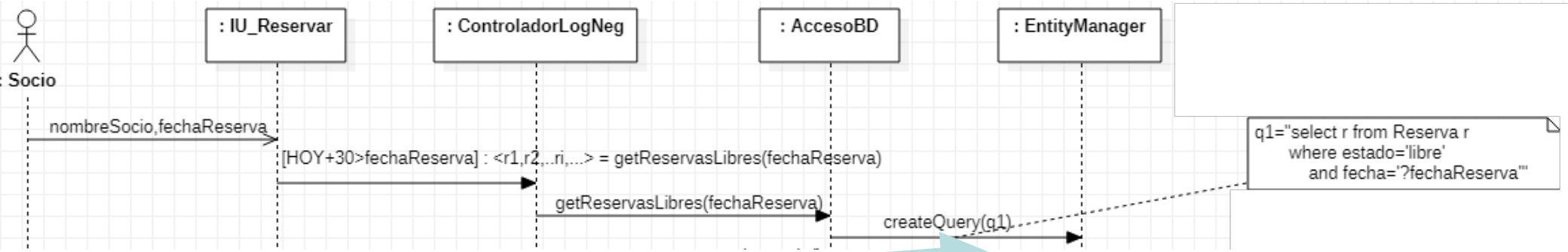


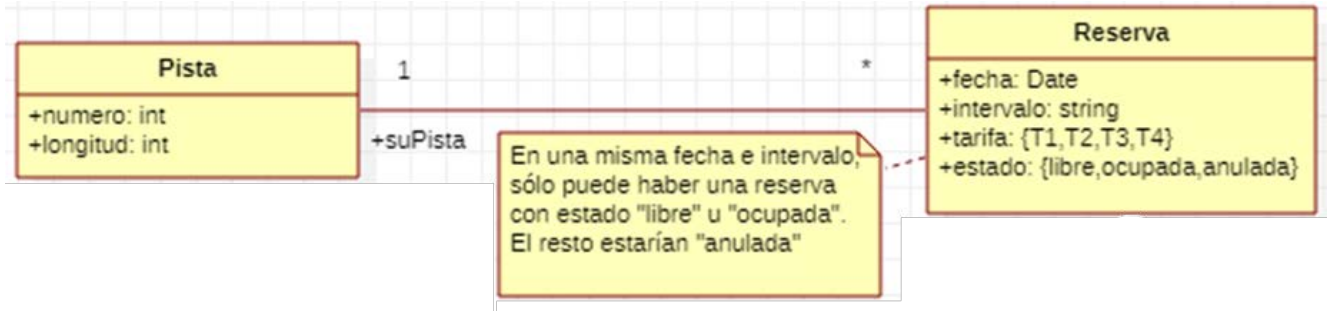
DIAGRAMA DE SECUENCIA



El AccesoBD hace la consulta a EntityManager (EM) de objectDB. Es la única clase que usa EM.

```
q1="select r from Reserva r
      where estado='libre'
        and fecha='?fechaReserva'"
```

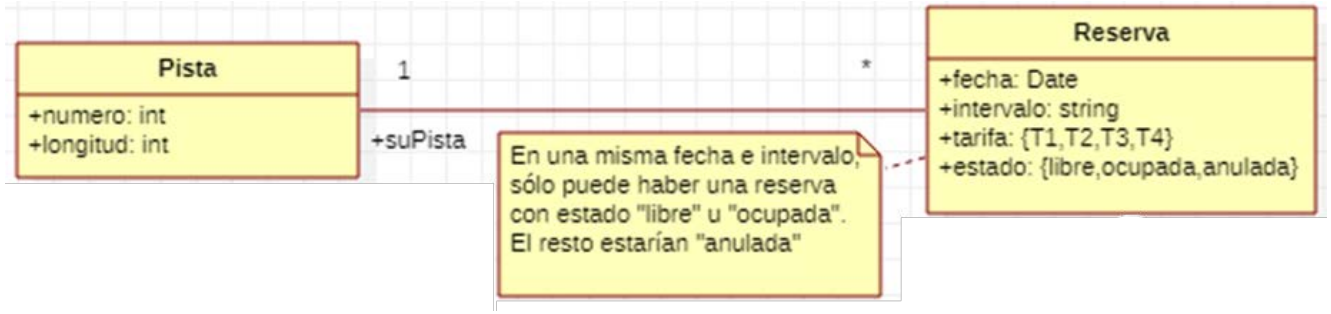
MODELO DEL DOMINIO



FLUJO DE EVENTOS

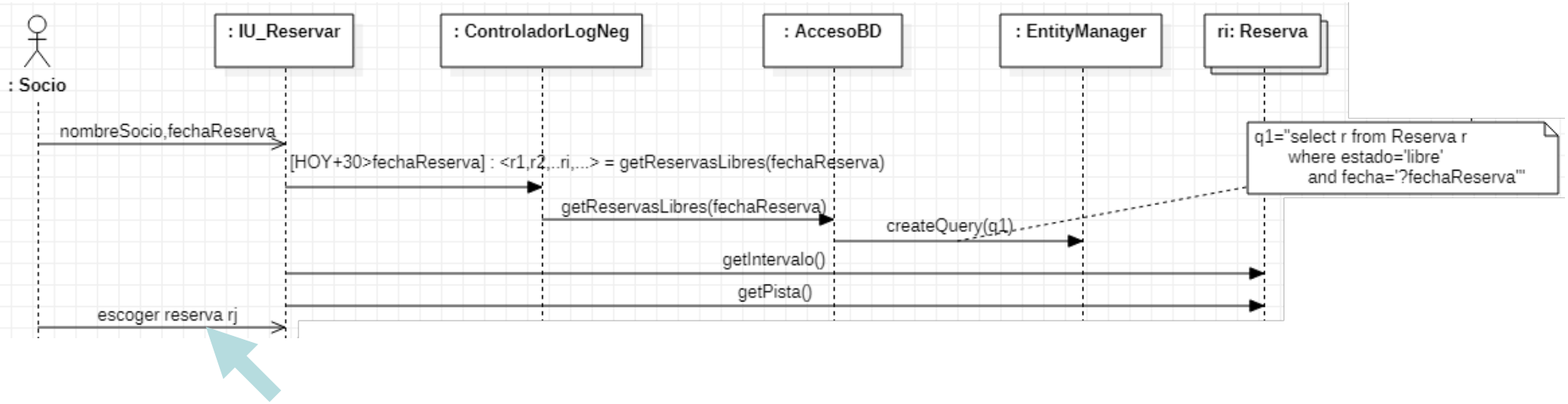
2. El sistema muestra la ocupación de las pistas en esa fecha: por cada pista, a qué horas está libre
3. El socio escoge una hora libre de una pista

MODELO DEL DOMINIO

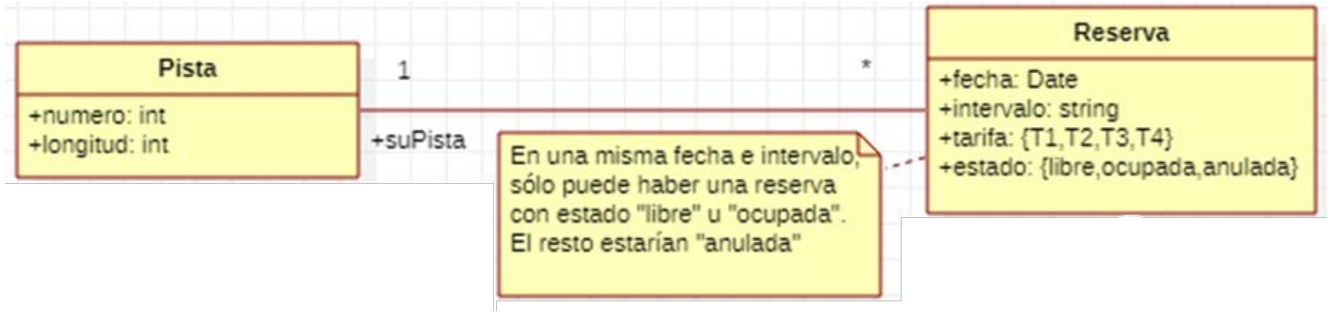


FLUJO DE EVENTOS

2. El sistema muestra la ocupación de las pistas en esa fecha: por cada pista, a qué horas está libre
3. El socio escoge una hora libre de una pista

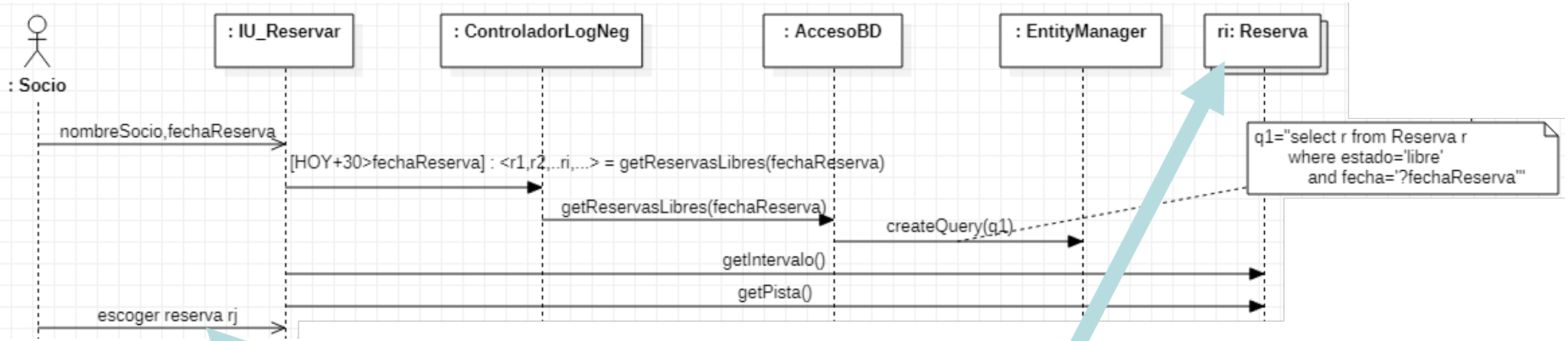


MODELO DEL DOMINIO



FLUJO DE EVENTOS

2. El sistema muestra la ocupación de las pistas en esa fecha: por cada pista, a qué horas está libre
3. El socio escoge una hora libre de una pista



En la interfaz :IU_Reservar está la relación entre las horas libres, pistas y reservas r1,r2,…ri,… El socio selecciona una hora libre de una pista, pero en la interfaz se sabe a qué reserva rj corresponde

Desde la interfaz se accede a los objetos porque ya se conocen sus referencias !!
En realidad habría que acceder a getNumero() de cada pista, pero esto no aporta mucho al diseño (son los típicos métodos accesores)

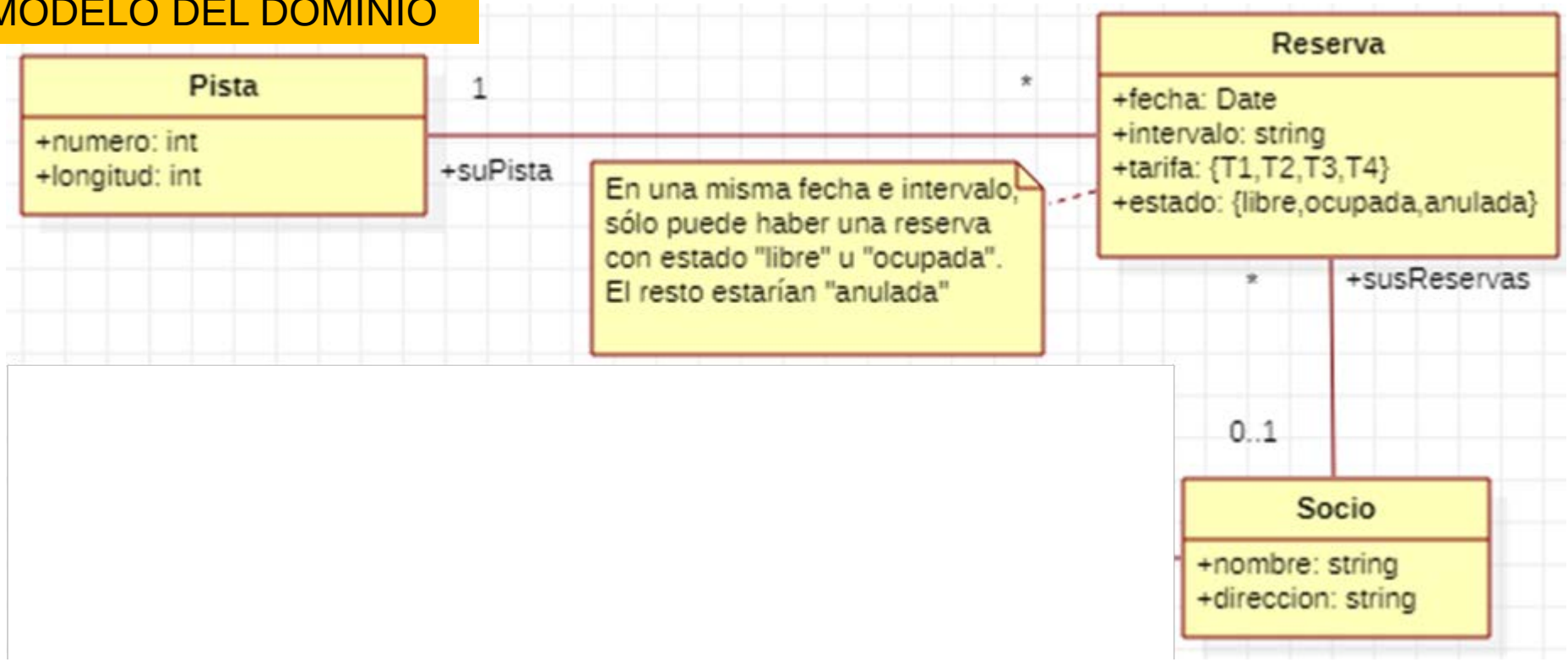
FLUJO DE EVENTOS

4. Se guarda la reserva: nombre del socio, fecha, hora, pista y la tarifa que corresponda (T1 si es fin de semana, o bien hora nocturna de un día laborable; o T2 en otro caso)

FLUJO DE EVENTOS

4. Se guarda la reserva: nombre del socio, fecha, hora, pista y la tarifa que corresponda (T1 si es fin de semana, o bien hora nocturna de un día laborable; o T2 en otro caso)

MODELO DEL DOMINIO



FLUJO DE EVENTOS

4. Se guarda la reserva: nombre del socio, fecha, hora, pista y la tarifa que corresponda (T1 si es fin de semana, o bien hora nocturna de un día laborable; o T2 en otro caso)

MODELO DEL DOMINIO

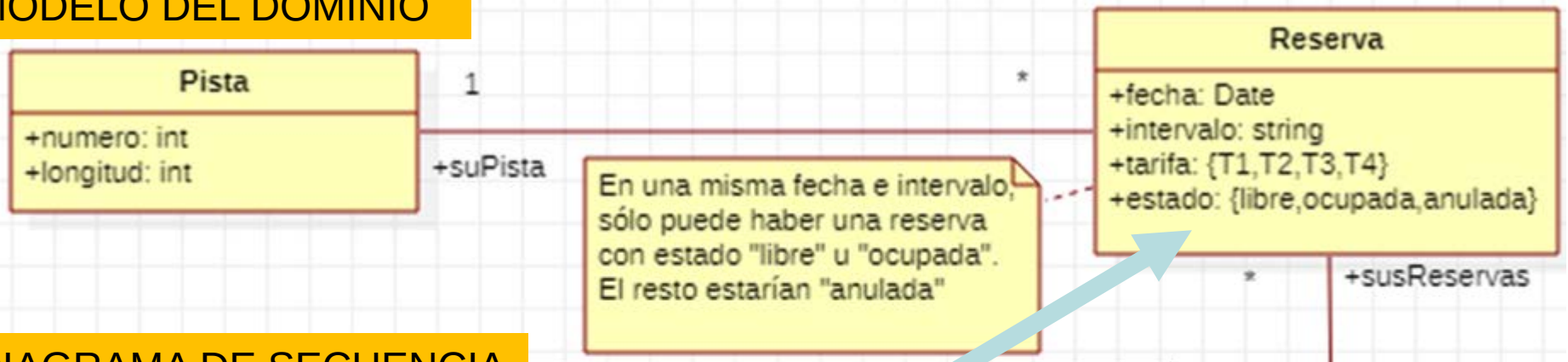
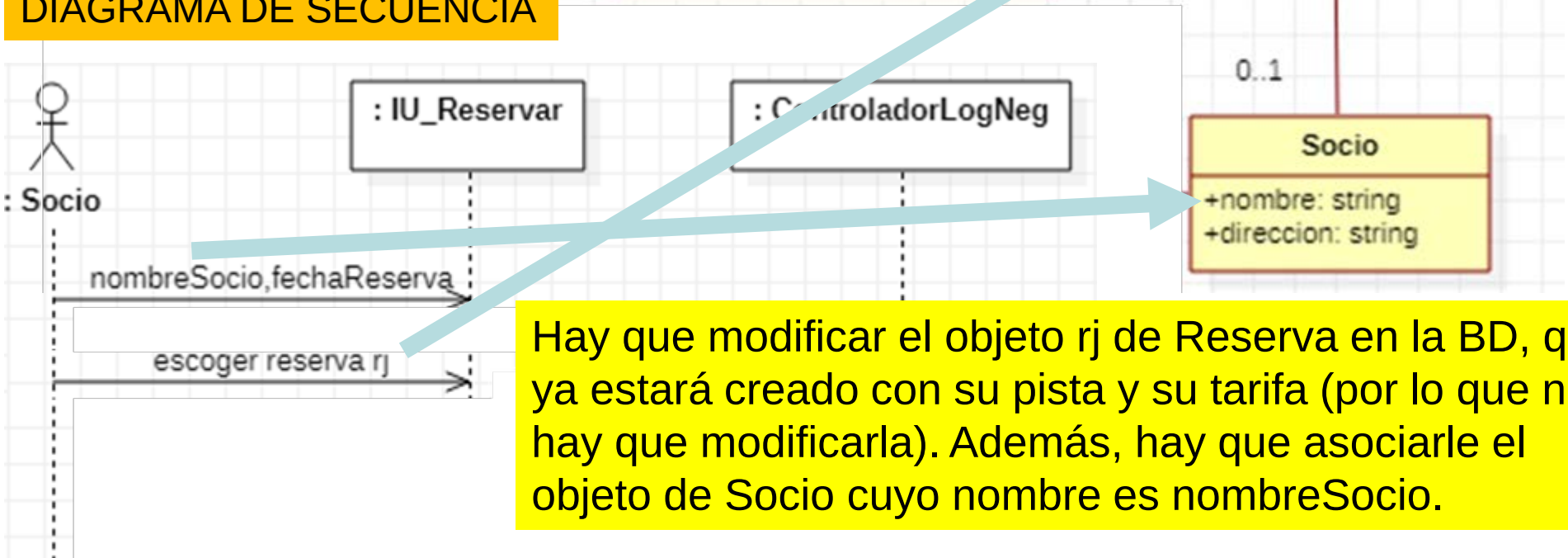


DIAGRAMA DE SECUENCIA



Hay que modificar el objeto rj de Reserva en la BD, que ya estará creado con su pista y su tarifa (por lo que no hay que modificarla). Además, hay que asociarle el objeto de Socio cuyo nombre es nombreSocio.

MODELO DEL DOMINIO

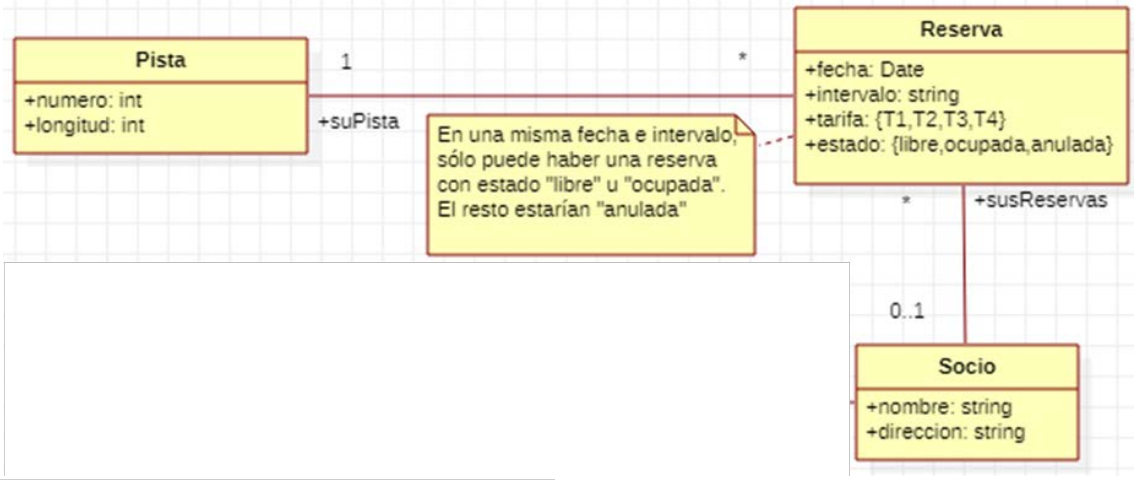
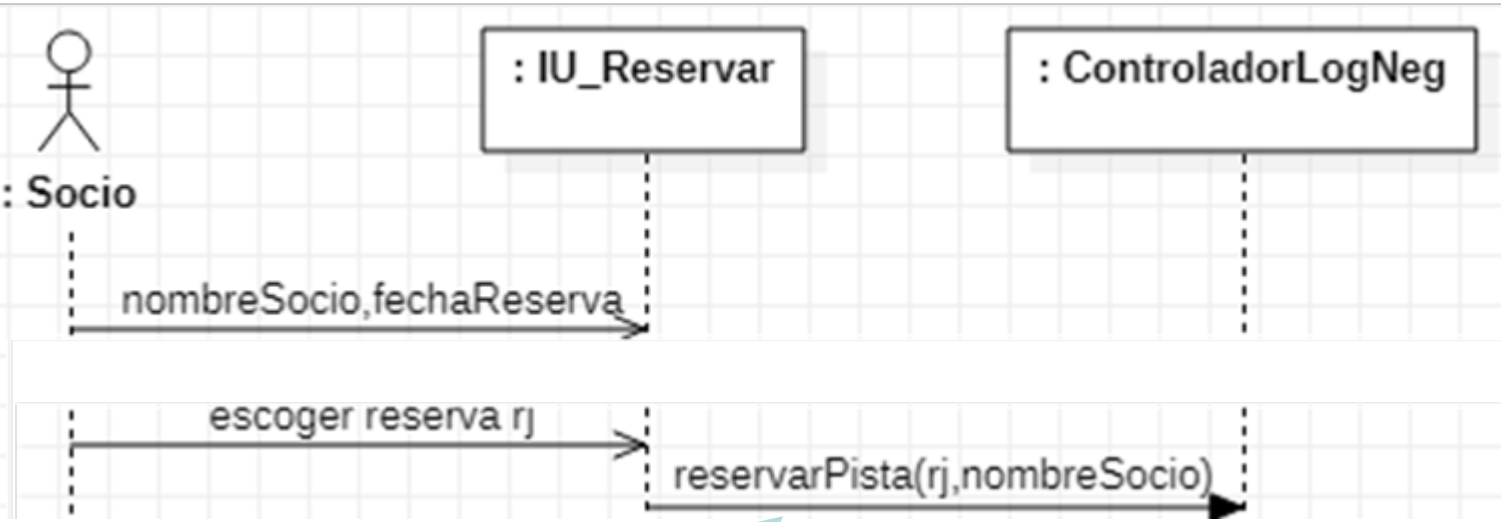


DIAGRAMA DE SECUENCIA



Aunque se trate de modificar algo en la BD, desde la presentación hay que pedírselo a la lógica del negocio

MODELO DEL DOMINIO

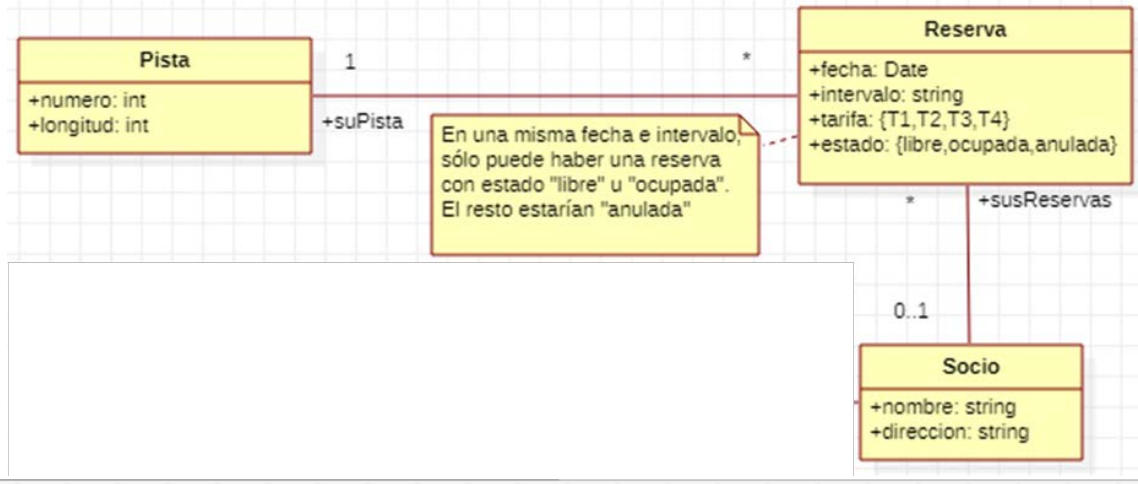
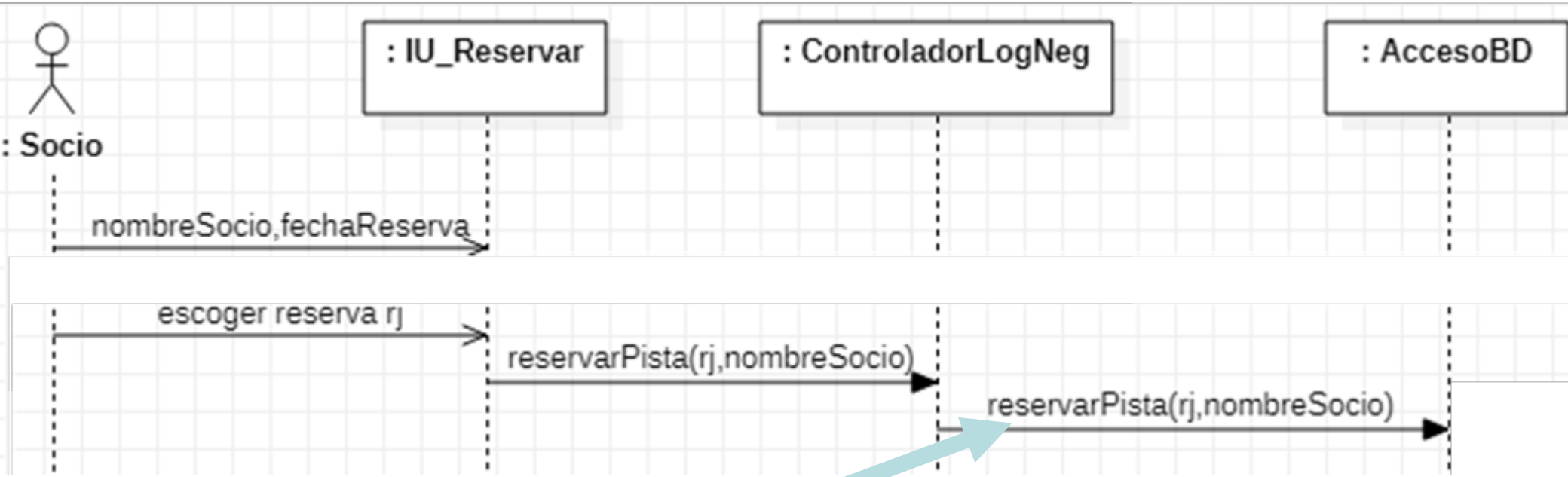


DIAGRAMA DE SECUENCIA



Quien delegará en el AccesoBD invocando a un método homónimo

MODELO DEL DOMINIO

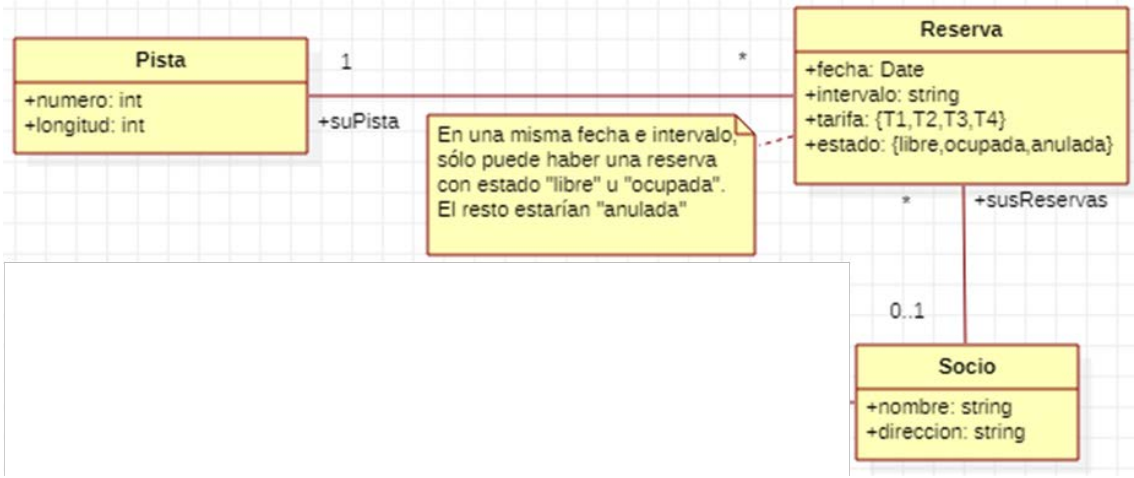
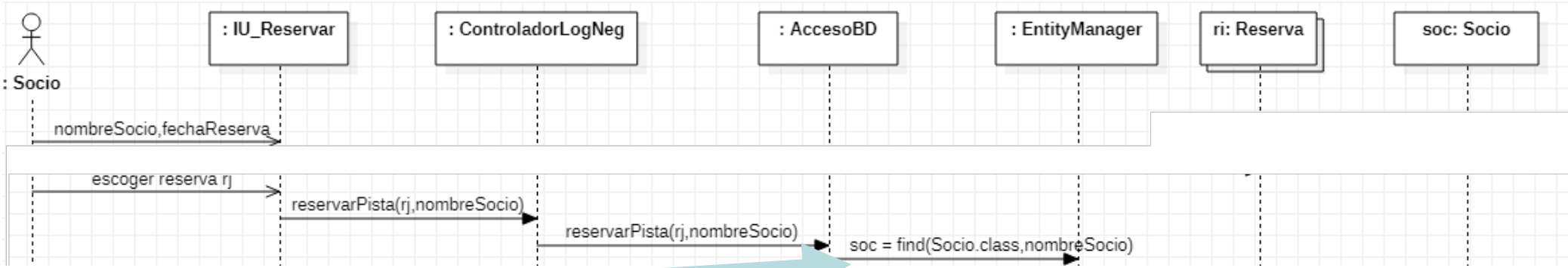


DIAGRAMA DE SECUENCIA



AccesoBD necesita encontrar el objeto de Socio cuyo nombre es nombreSocio

MODELO DEL DOMINIO

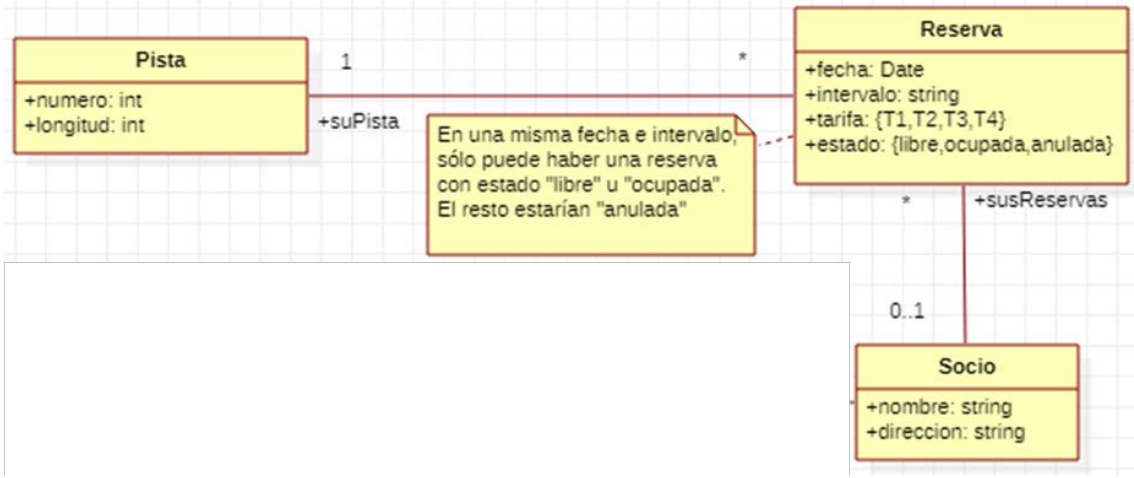
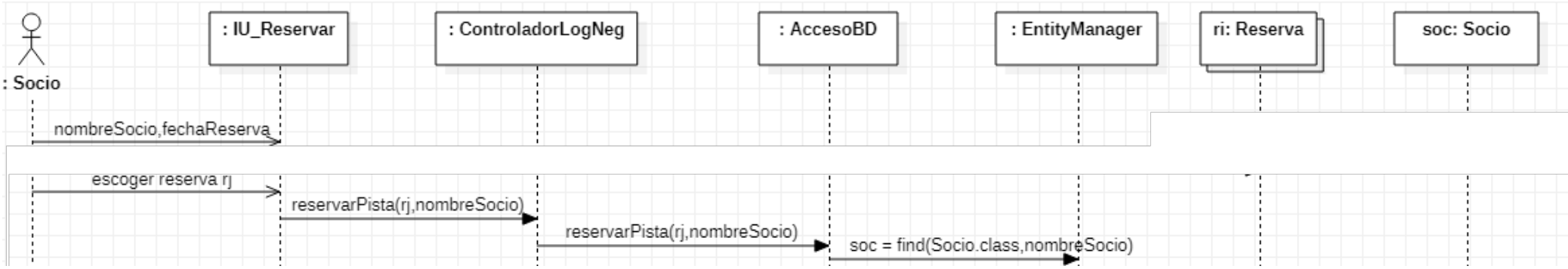
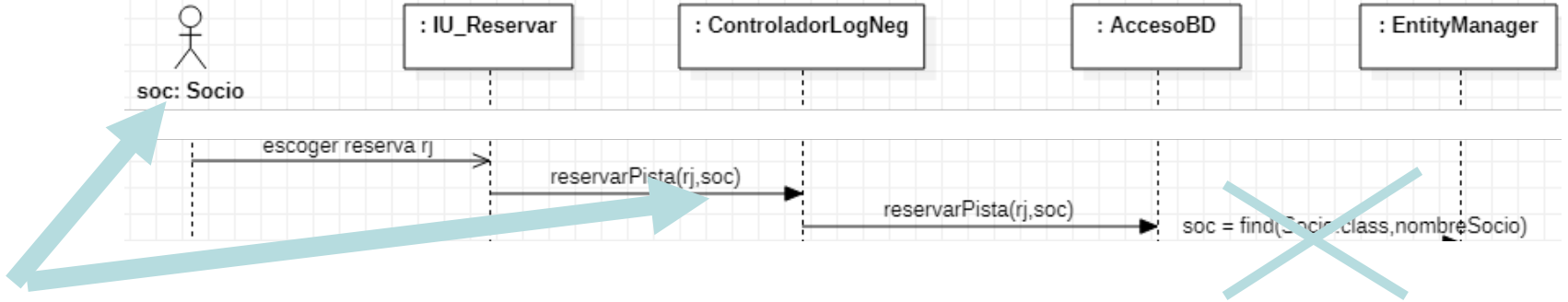


DIAGRAMA DE SECUENCIA



Nota: Si en este sistema hubiera Login, entonces no habría hecho falta…



MODELO DEL DOMINIO

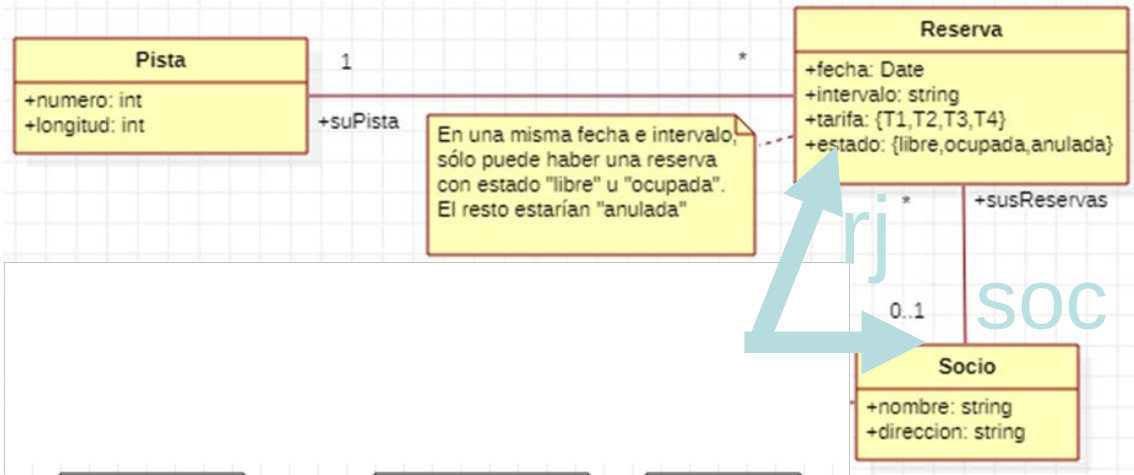
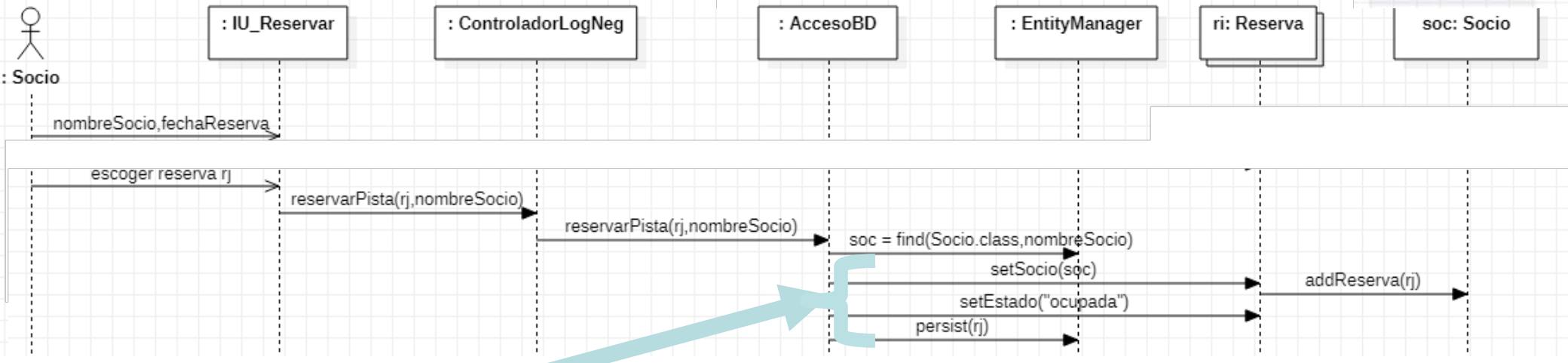
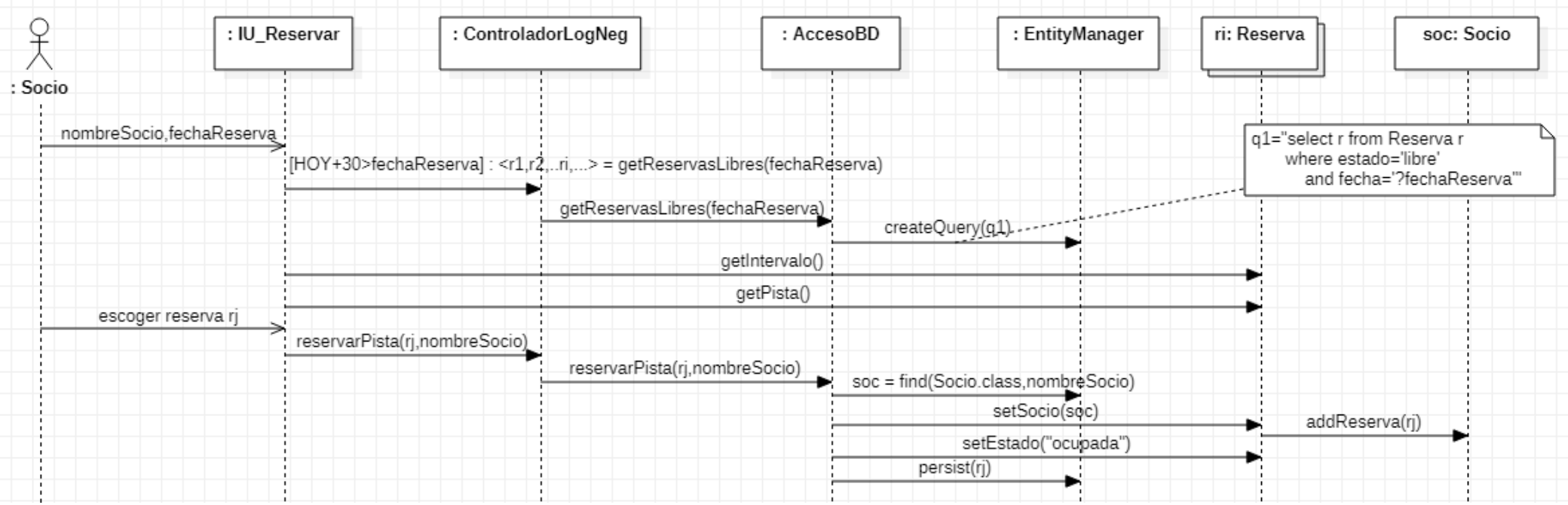


DIAGRAMA DE SECUENCIA



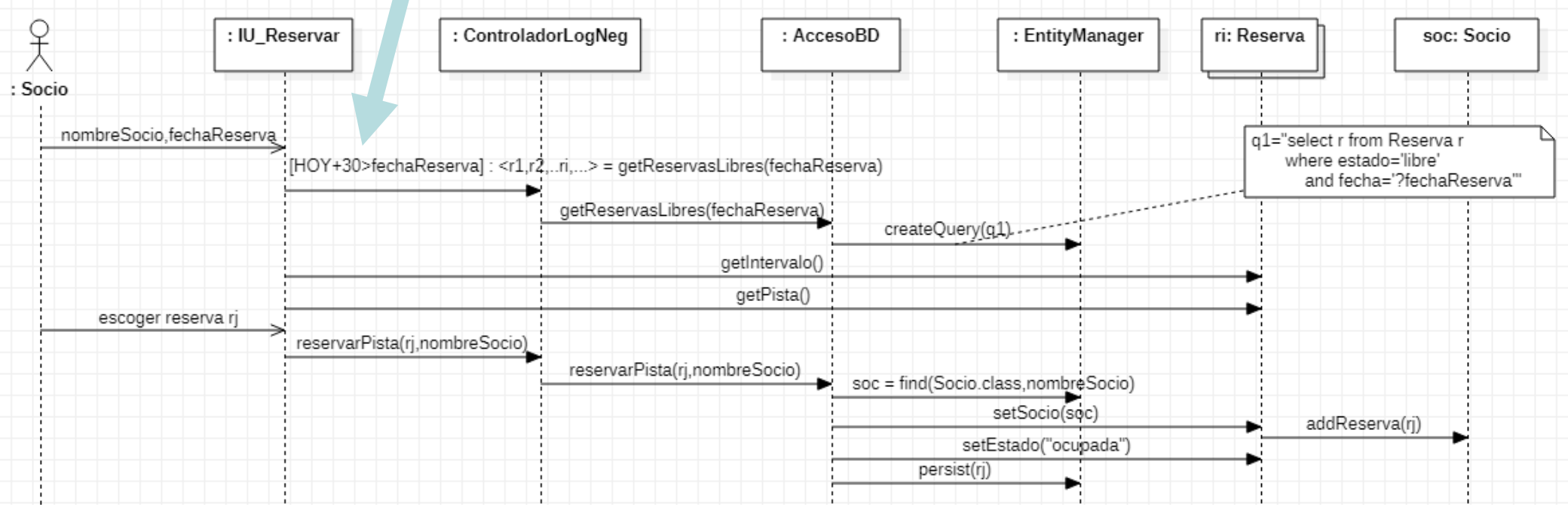
AccesoBD modifica el objeto rj de Reserva asignándole el socio soc y el estado “ocupada”, y le da persistencia a rj.
Se muestra explícitamente que al modificar el socio soc de la reserva rj, se mantiene la asociación inversa (en el socio soc se añade la reserva rj con addReserva(rj))
En realidad, se muestra sólo la llamada a persist(rj), presuponiendo que se hará dentro de una transacción con un commit (de hecho en ese caso, el persist(rj) no sería necesario…



Este diagrama de secuencia resuelve el “flujo normal de eventos” y puede considerarse ya como correcto.

Pero, en este caso, vamos a continuar con el mismo para tratar también todos los flujos alternativos. Por dos razones: 1) como ejercicio académico (para ver bloques “alt” y “loop”) y 2) porque cabe en una misma hoja (si no, sería mejor definir diagramas de secuencia separados para flujos alternativos)

Uno de los flujos alternativos ya se había añadido al DS

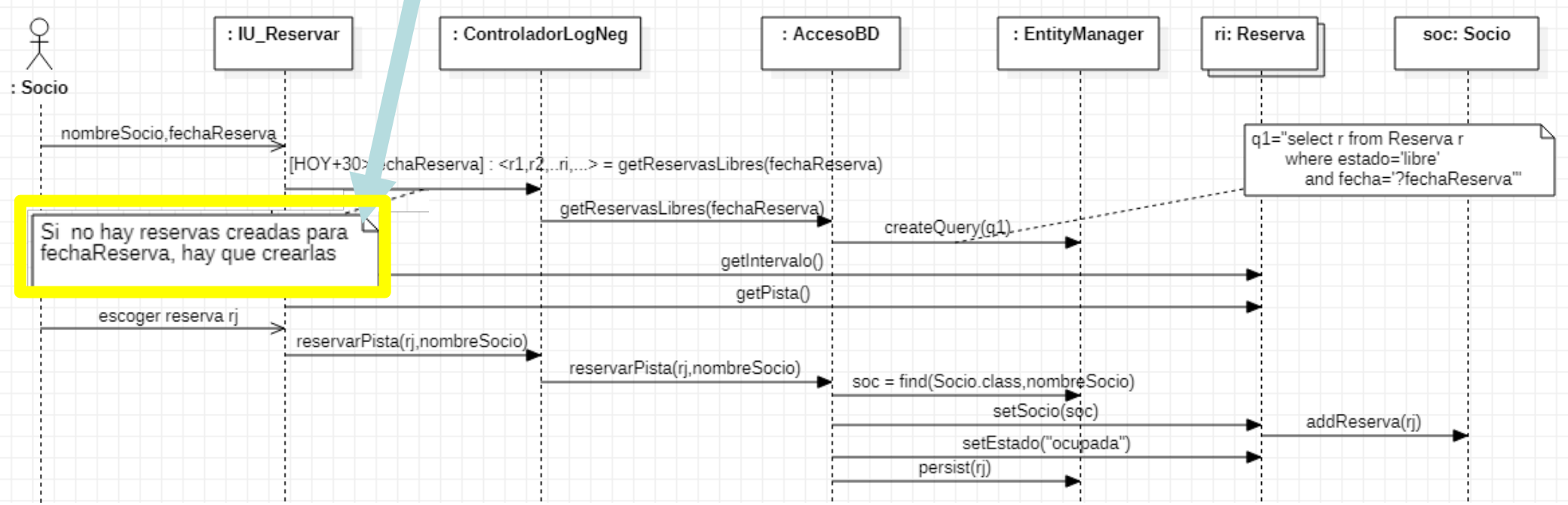


Y falta por tener en cuenta este otro

Flujos de Eventos Alternativos

- Si la fecha no es dentro de 30 días, no se puede reservar. Fin
- Si las reservas de ese día no se han creado, se crean y se continúa con el flujo de eventos normal

Una primera opción sencilla, podría consistir en añadir un comentario UML para que no se olvide

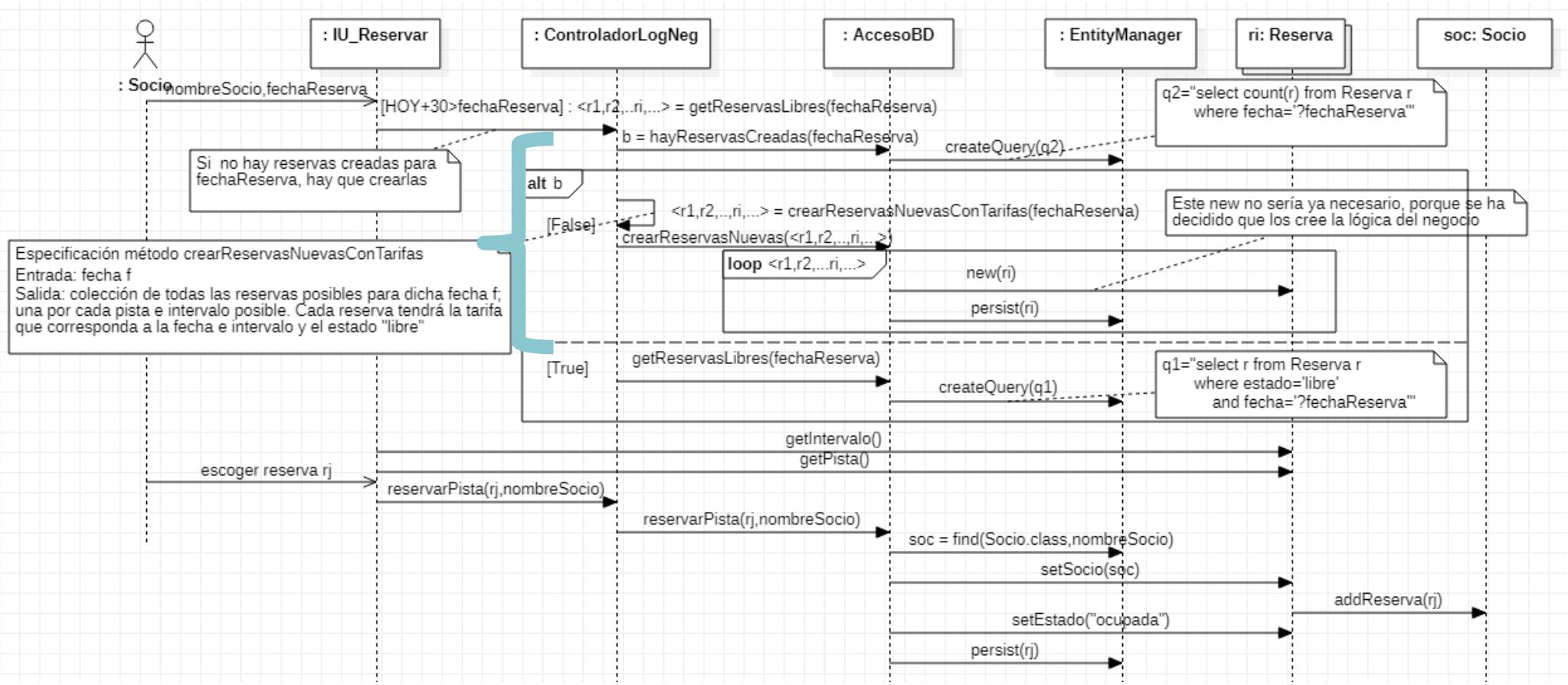


Flujos de Eventos Alternativos

- Si la fecha no es dentro de 30 días, no se puede reservar. Fin
- Si las reservas de ese día no se han creado, se crean y se continúa con el flujo de eventos normal

Y en esta se detalla mejor cuándo y cómo crear las nuevas reservas…

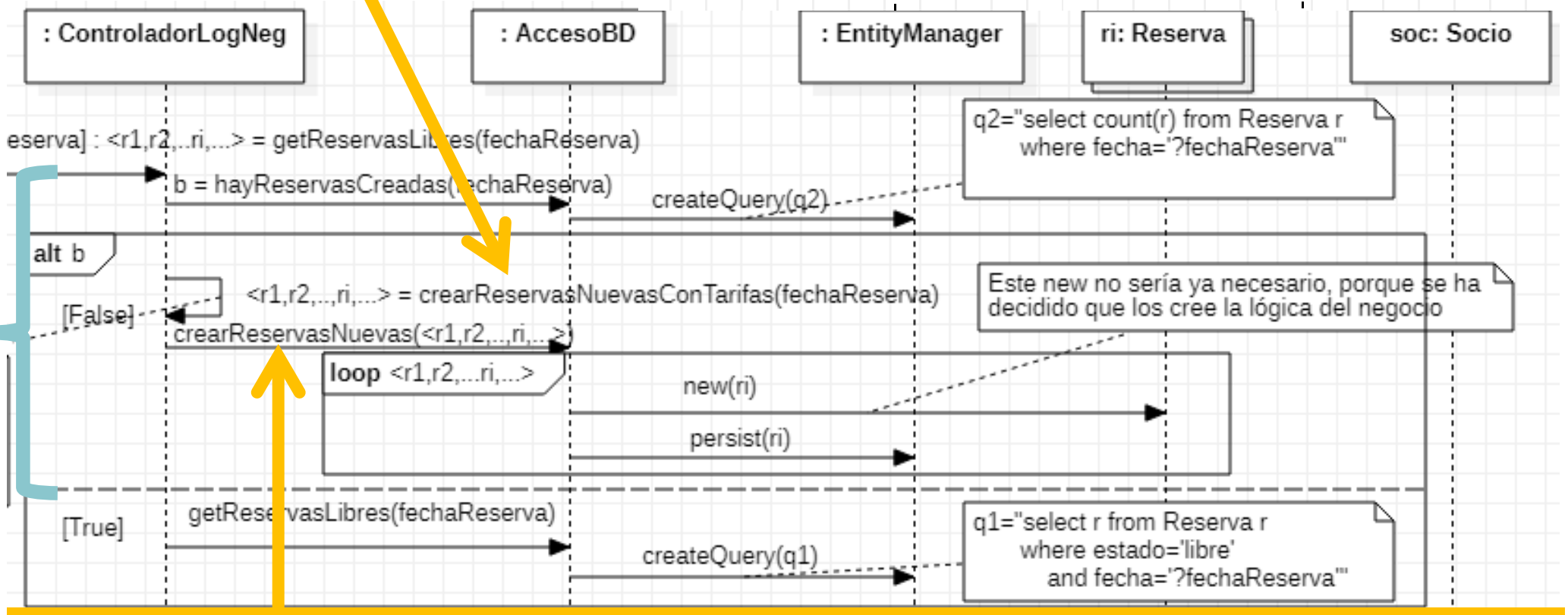
Usando una condición (con bloque “alt”) y una repetición (bloque “loop”)



En la siguiente diapositiva se muestra el detalle de esa parte del diagrama

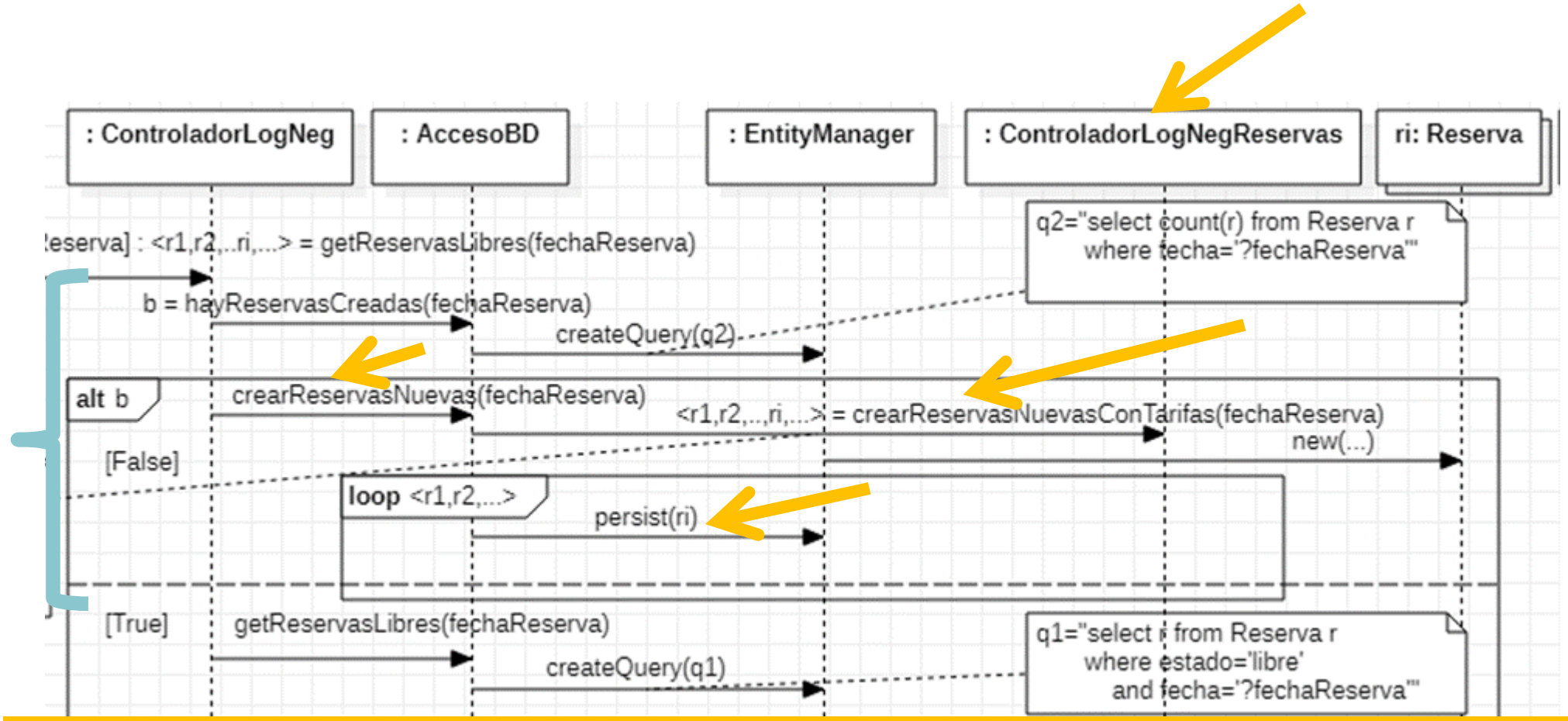
Como parte del diseño, se puede además especificar el método crearReservasNuevasConTarifas de la lógica del negocio:

Especificación método crearReservasNuevasConTarifas
Entrada: fecha f
Salida: colección de todas las reservas posibles para dicha fecha f; una por cada pista e intervalo posible. Cada reserva tendrá la tarifa que corresponda a la fecha e intervalo y el estado "libre"



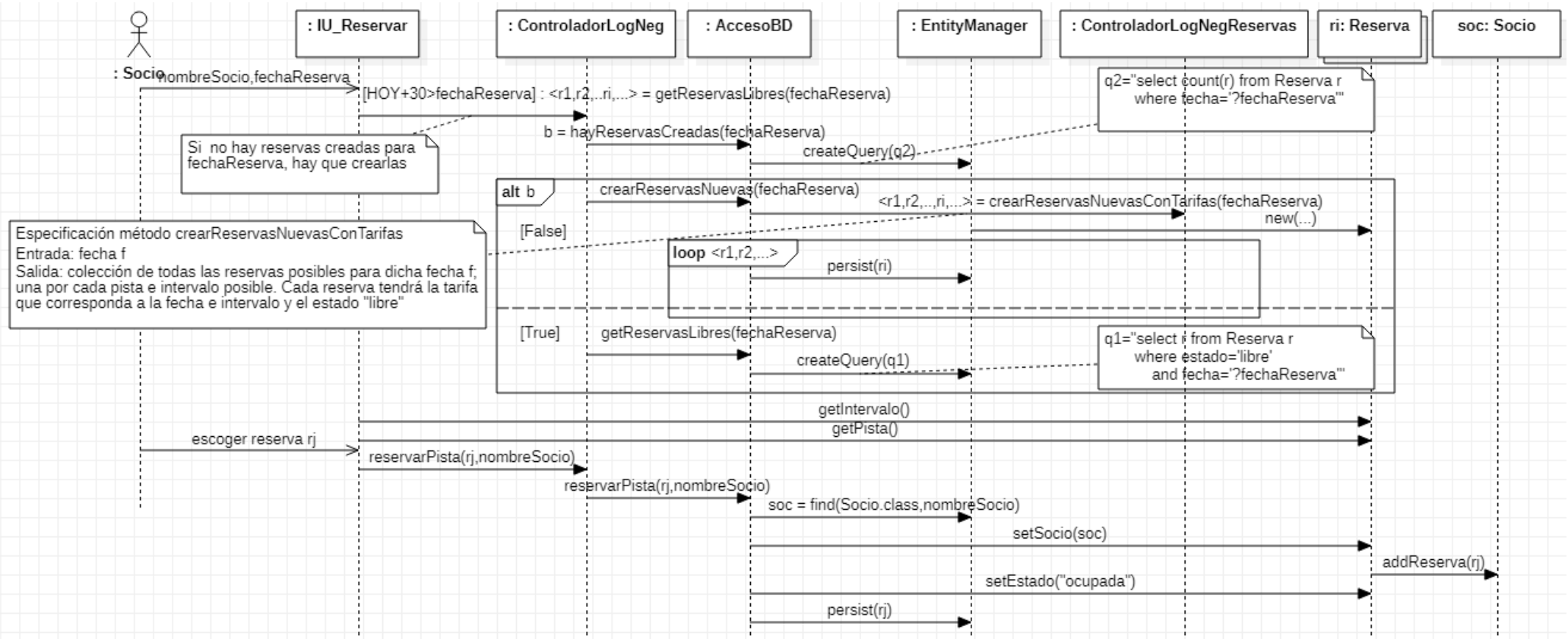
El método crearReservasNuevas es de AccesoBD (quien debe crear los objetos), pero en este caso se ha considerado que los valores de dichos objetos los debe definir la lógica del negocio (que es quien conoce de tarifas, intervalos…).

En la siguiente diapositiva se presenta una alternativa a este detalle del diagrama



Esta solución utiliza una clase (ControladorLogNegReservas) especializada en crear las reservas: es un controlador de reservas, que forma parte de la lógica del negocio y que es responsable de definir las tarifas, intervalos de fechas, etc. Si hubiera una nueva política de tarifas o cambio de horarios, sólo habría que modificar esta clase.

El AccesoBD la invocaría en vez de hacer los “new”, y luego daría persistencia a los objetos creados, y la clase ControladorLogNeg llamaría simplemente a AccesoBD.



Este diagrama de secuencia resuelve el flujo de eventos normal y los alternativos