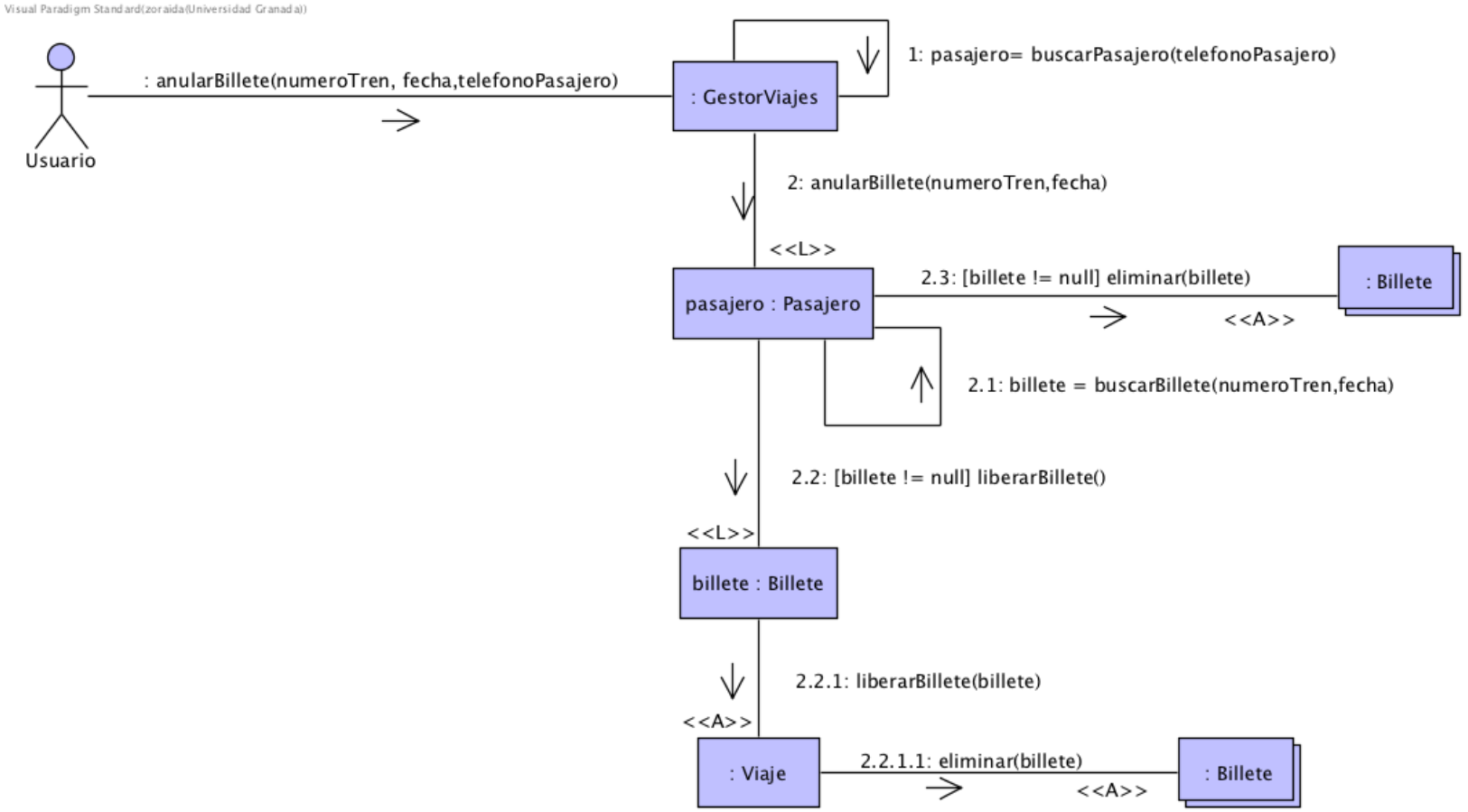
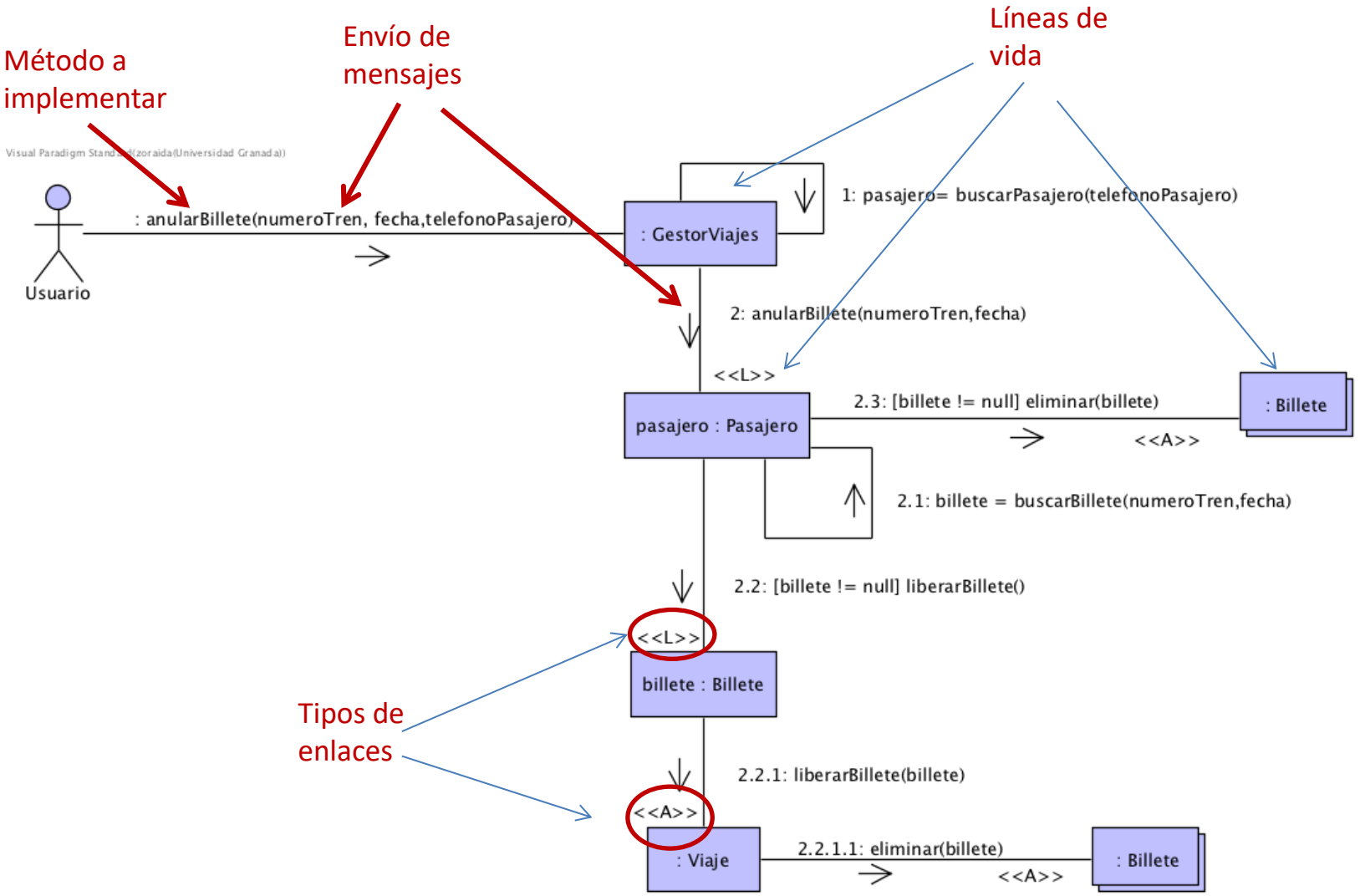
Programación y Diseño Orientado a Objetos. ETSIIT. UGR

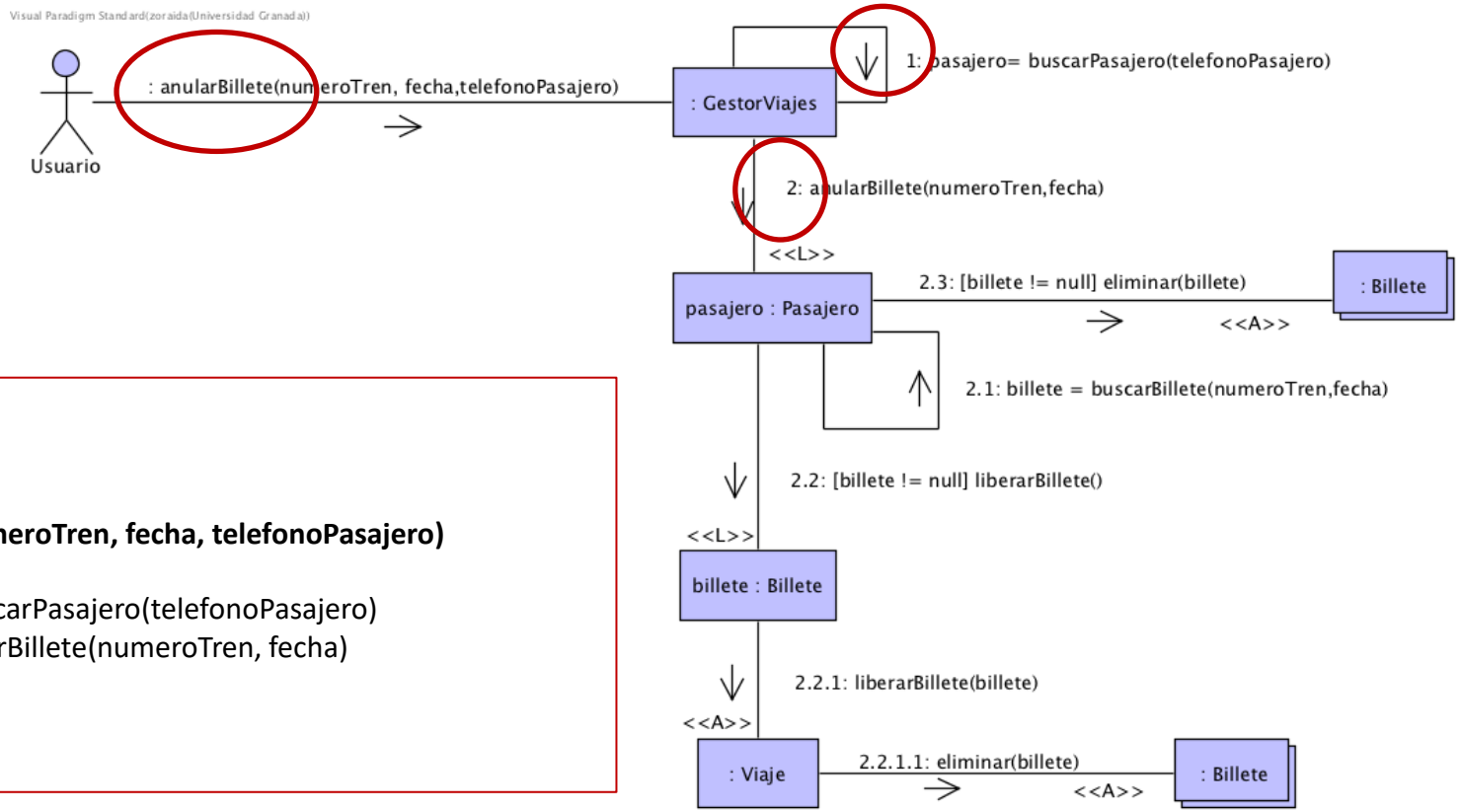
Paso de Diagrama de Comunicación UML a código Java y Ruby

María José Rodríguez Fórtiz. LSI.

Diagrama de comunicación: Énfasis en envío de mensajes




Método a implementar
Envío de mensajes
Líneas de vida
Visual Paradigm Standard(zoraida(Universidad Granada))
Usuario
: anularBillete(numeroTren, fecha,telefonoPasajero)
: GestorViajes
1: pasajero= buscarPasajero(telefonoPasajero)
2: anularBillete(numeroTren,fecha)
<<L>>
pasajero : Pasajero
2.3: [billete != null] eliminar(billete)
<<A>>
: Billete
2.1: billete = buscarBillete(numeroTren,fecha)
2.2: [billete != null] liberarBillete()
<<L>>
billete : Billete
Tipos de enlaces
2.2.1: liberarBillete(billete)
<<A>>
: Viaje
2.2.1.1: eliminar(billete)
<<A>>
: Billete

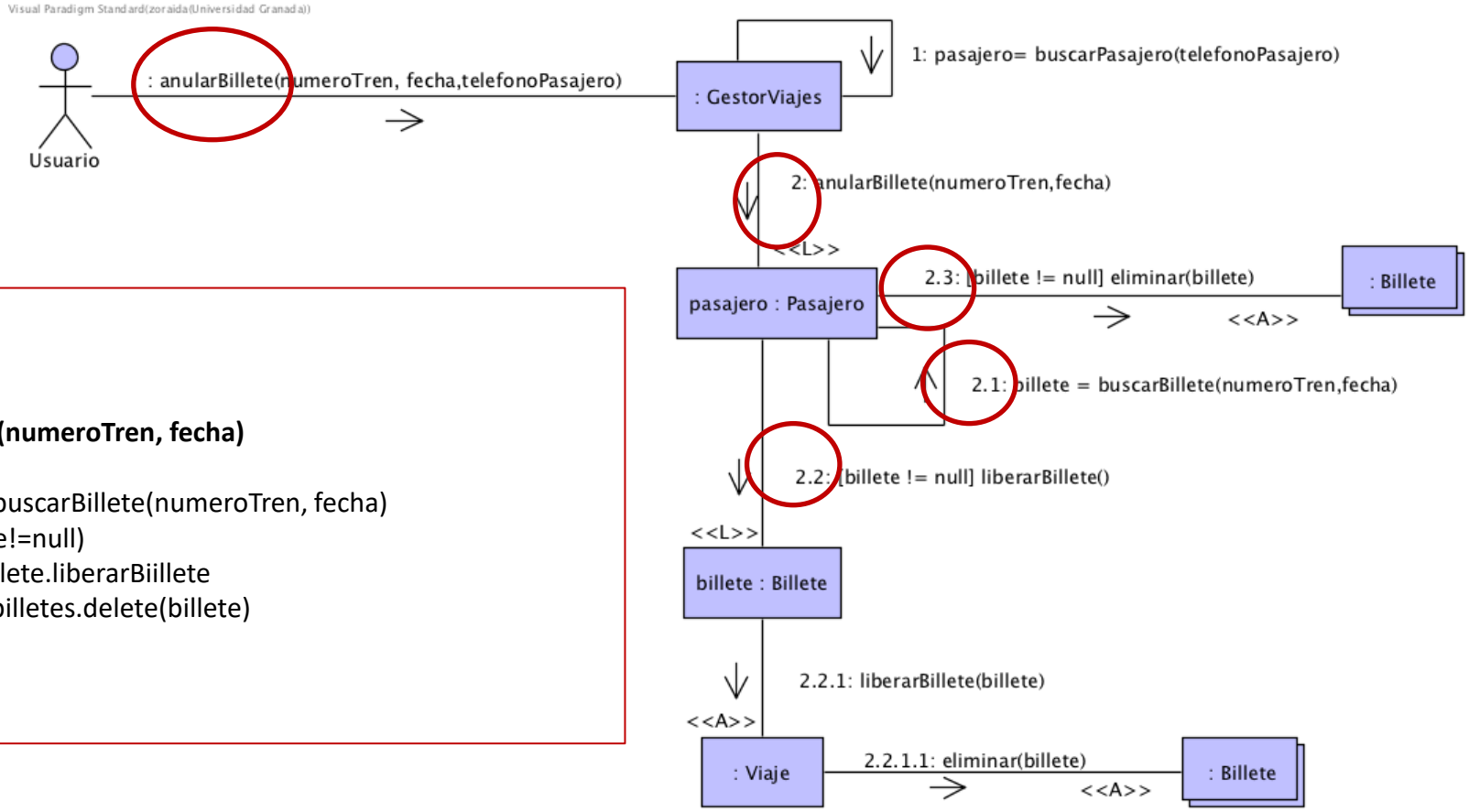


Ruby

```
class GestorViajes
  …
 def anularBillete(numeroTren, fecha, telefonoPasajero)

(1)      pasajero = buscarPasajero(telefonoPasajero)
(2)      pasajero.anularBillete(numeroTren, fecha)

end
end
```

Java

```
class GestorViajes{
  …
  public void anularBillete(int numeroTren, Date fecha, String telefonoPasajero){

(1)       Pasajero pasajero = buscarPasajero(telefonoPasajero);
(2)       pasajero.anularBillete(numeroTren, fecha);
  }
}
```

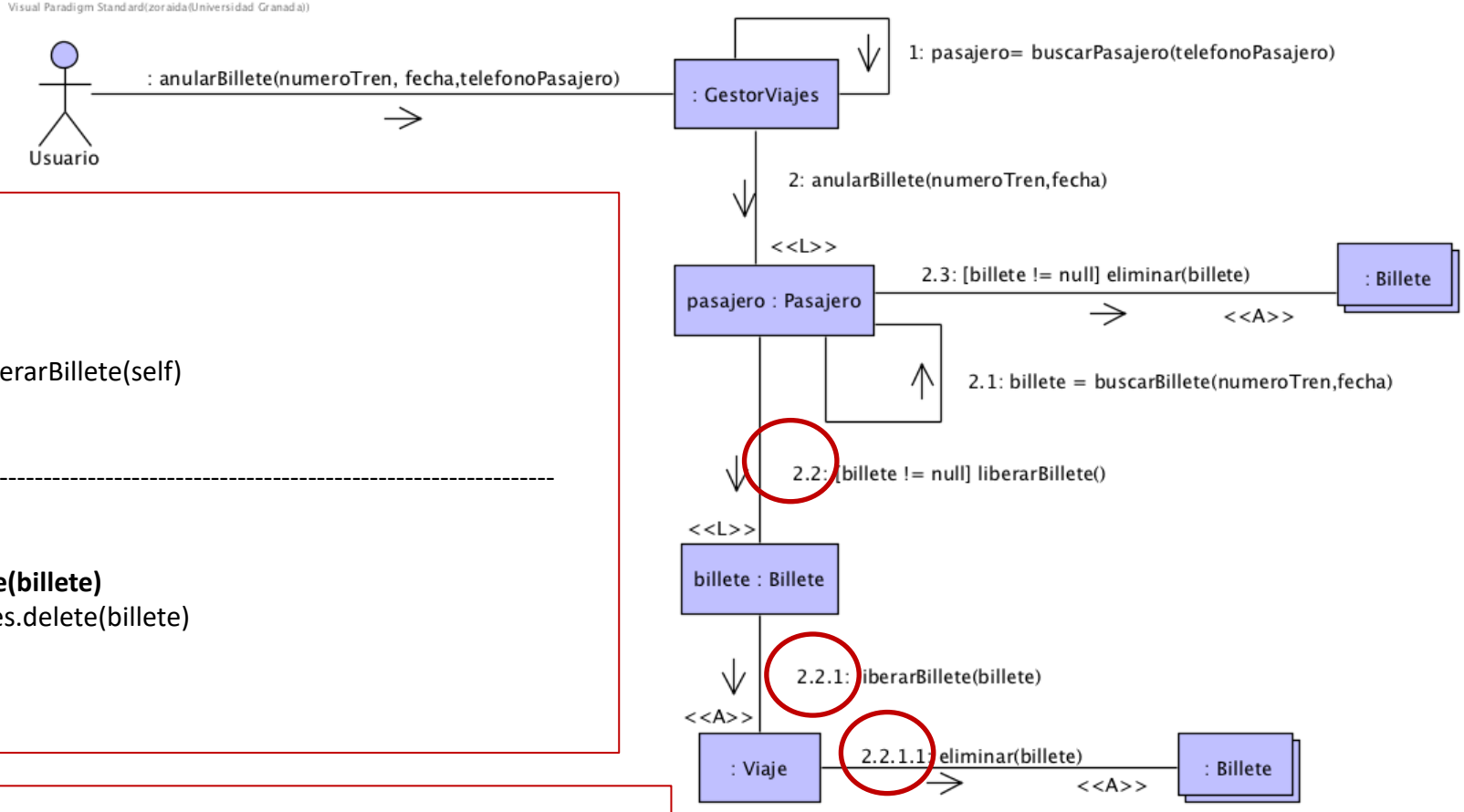


Ruby

```
class Pasajero
  …
 def anularBillete(numeroTren, fecha)

(2.1)       billete= buscarBillete(numeroTren, fecha)
            if  (billete!=null)
(2.2)               billete.liberarBiillete
(2.3)               @billetes.delete(billete)
            end
 end
end
```

Java

```
class Pasajero{
 …
  public void anularBillete(int numeroTren, Date fecha, String telefonoPasajero){

(2.1)       Billete billete= buscarBillete(numeroTren, fecha);
            if  (billete!=null){
(2.2)               billete.liberarBiillete();
(2.3)               billetes.remove(billete); //billetes es un atributo de referencia
            }
  }
}
```



Ruby

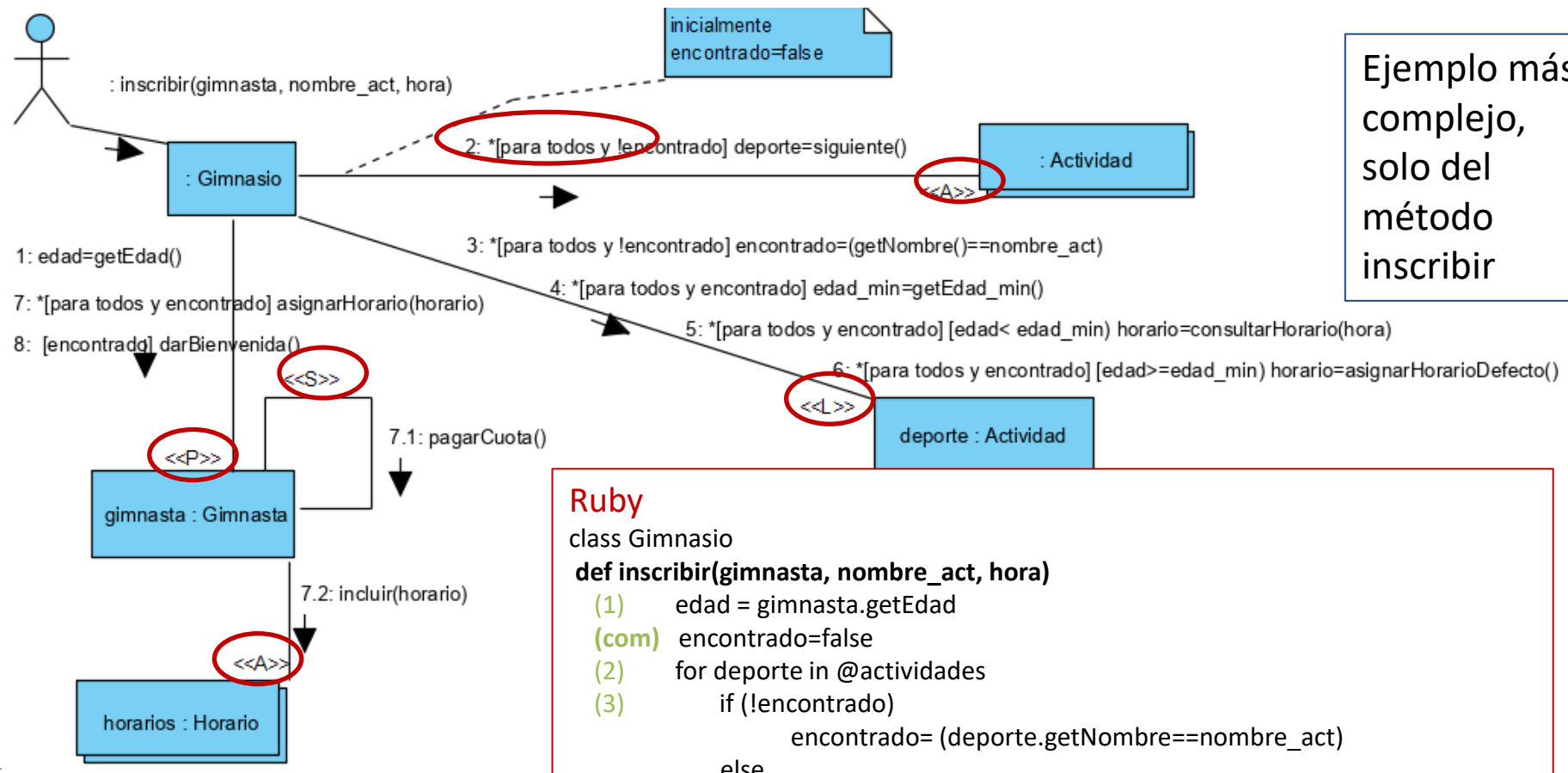
```
class Billete
  ...
 def  liberarBillete
(2.2.1)   @viaje.liberarBillete(self)
 end
end
```

```
Class Viaje
...
   def liberarBillete(billete)
(2.2.1.1)   @billetes.delete(billete)
end
end
```

Java

```
class Billete{
 ...
  public void liberarBillete( ) {
(2.2.1)       viaje.liberarBillete(this); }
}
```

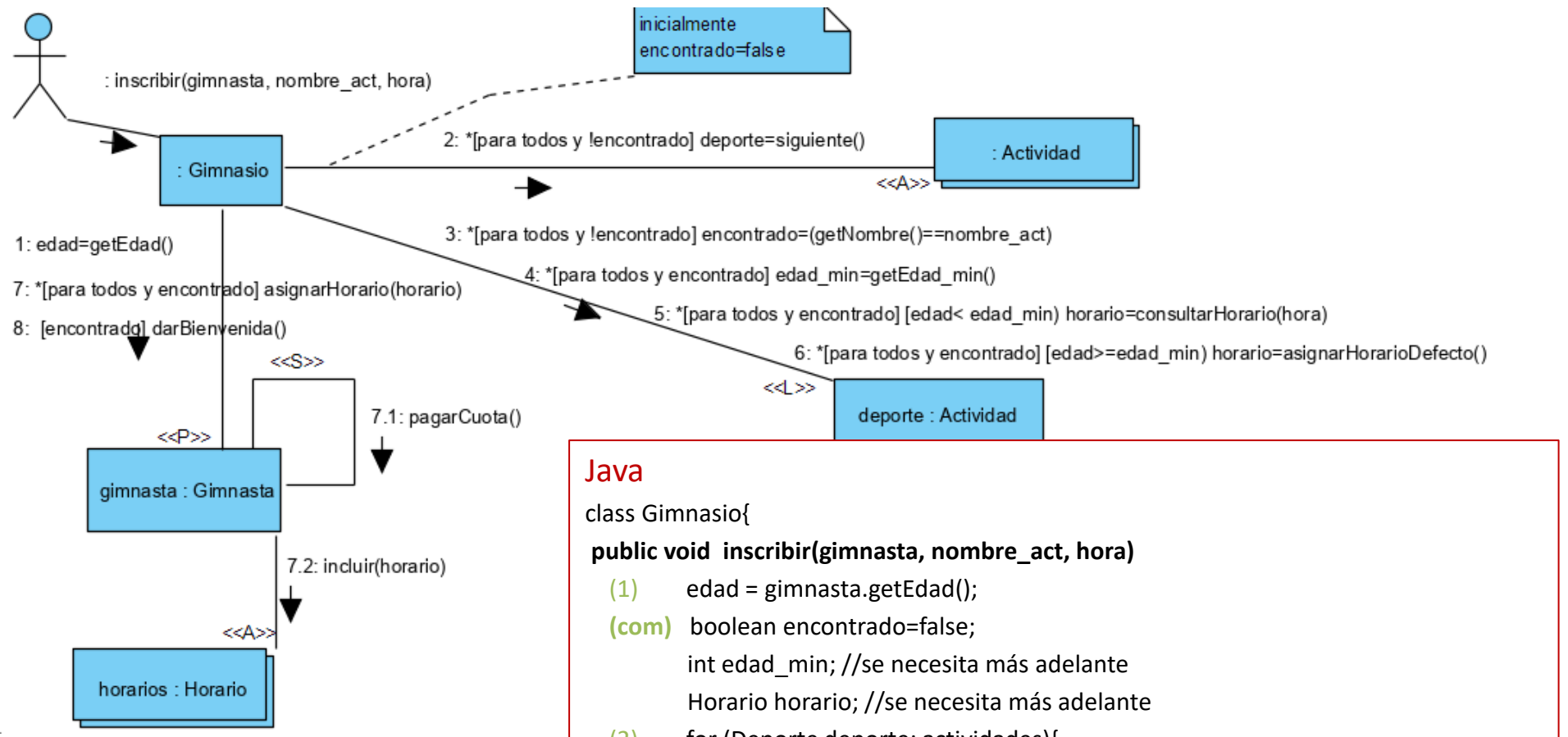
```
Class Viaje{
...
   public void liberarBillete(Billete billete)
(2.2.1.1)   billetes.remove(billete);}
}
```



Ejemplo más complejo, solo del método inscribir

Ruby

```
class Gimnasio
 def inscribir(gimnasta, nombre_act, hora)
   (1)      edad = gimnasta.getEdad
   (com)    encontrado=false
   (2)      for deporte in @actividades
   (3)           if (!encontrado)
                        encontrado= (deporte.getNombre==nombre_act)
                 else
   (4)               edad_min= deporte.getEdad_min
   (5)               if (edad<edad_min)
                            horario= deporte.consultaHorario(hora)
                     else
   (6)                       horario=deporte.asignarHorarioDefecto()
                     end
   (7)               gimnasta.asignarHorario(horario)
                 end
            end
   (8)      if encontrado
                  gimnasta.darBienvenida
            end
end
```



Java

```
class Gimnasio{
 public void  inscribir(gimnasta, nombre_act, hora)
   (1)      edad = gimnasta.getEdad();
   (com)    boolean encontrado=false;
            int edad_min; //se necesita más adelante
            Horario horario; //se necesita más adelante
   (2)      for (Deporte deporte: actividades){
   (3)           if (!encontrado)
                       encontrado= (deporte.getNombre()==nombre_act);
                 else {
   (4)                edad_min= deporte.getEdad_min();
   (5)                if (edad<edad_min)
                             horario= deporte.consultaHorario(hora);
   (6)                else    horario=deporte.asignarHorarioDefecto();
   (7)                gimnasta.asignarHorario(horario); }
             }
   (8)       if encontrado
                       gimnasta.darBienvenida();  }
```