Departamento de Computación,
Facultad de Ciencias Exactas y Naturales,
Universidad de Buenos Aires

Trabajo Práctico 2

Ingeniería de Software I

Primer Cuatrimestre de 2014

Apellido y Nombre	LU	E-mail
Delgado, Alejandro N.	601/11	nahueldelgado@gmail.com
Lovisolo, Leandro	645/11	leandro@leandro.me
Petaccio, Lautaro José	443/11	lausuper@gmail.com
Requeni, Gastón	400/11	grequeni@hotmail.com
Vita, Sebastián	149/11	sebastian_vita@yahoo.com.ar

Índice

1. Introducción

En el siguiente informe, mostraremos en detalle nuevos aspectos y puntos de vista del proyecto de bicisendas de Mar Chiquita.

Mostraremos cómo se relacionan los conceptos del proyecto mediante un diagrama de modelo conceptual y mostraremos mediante la utilización de OCL los invariantes que se aplican sobre el mismo.

Utilizaremos el diagrama del modelo de casos de uso para detallar la interacción de tanto usuarios como el personal autorizado para la utilización del software y describiremos cada caso de uso a continuación del mismo.

Por último, nos valeremos de diagramas de actividades y FSM para brindar detalle tanto a las interacciones entre agentes como a eventos específicos como las penalizaciones.

2. Desarrollo

Debido a que la continuidad del proyecto requería de la elección de una de las alternativas propuestas anteriormente para la distribución de bicicletas en hora pico, elegimos la propuesta de que las distribuciones se realicen de manera automática utilizando los datos estadísticos recaudados por el sistema.

Esta elección se debe a que la distribución automática propone una manera más simple y precisa, ya que no se requiere de nadie calculando o decidiendo para cada estación, que cantidad de bicicletas le toca a cada una, permitiendo así, un mejor uso de los recursos disponibles.

3. Diferencias con especificaciones anteriores

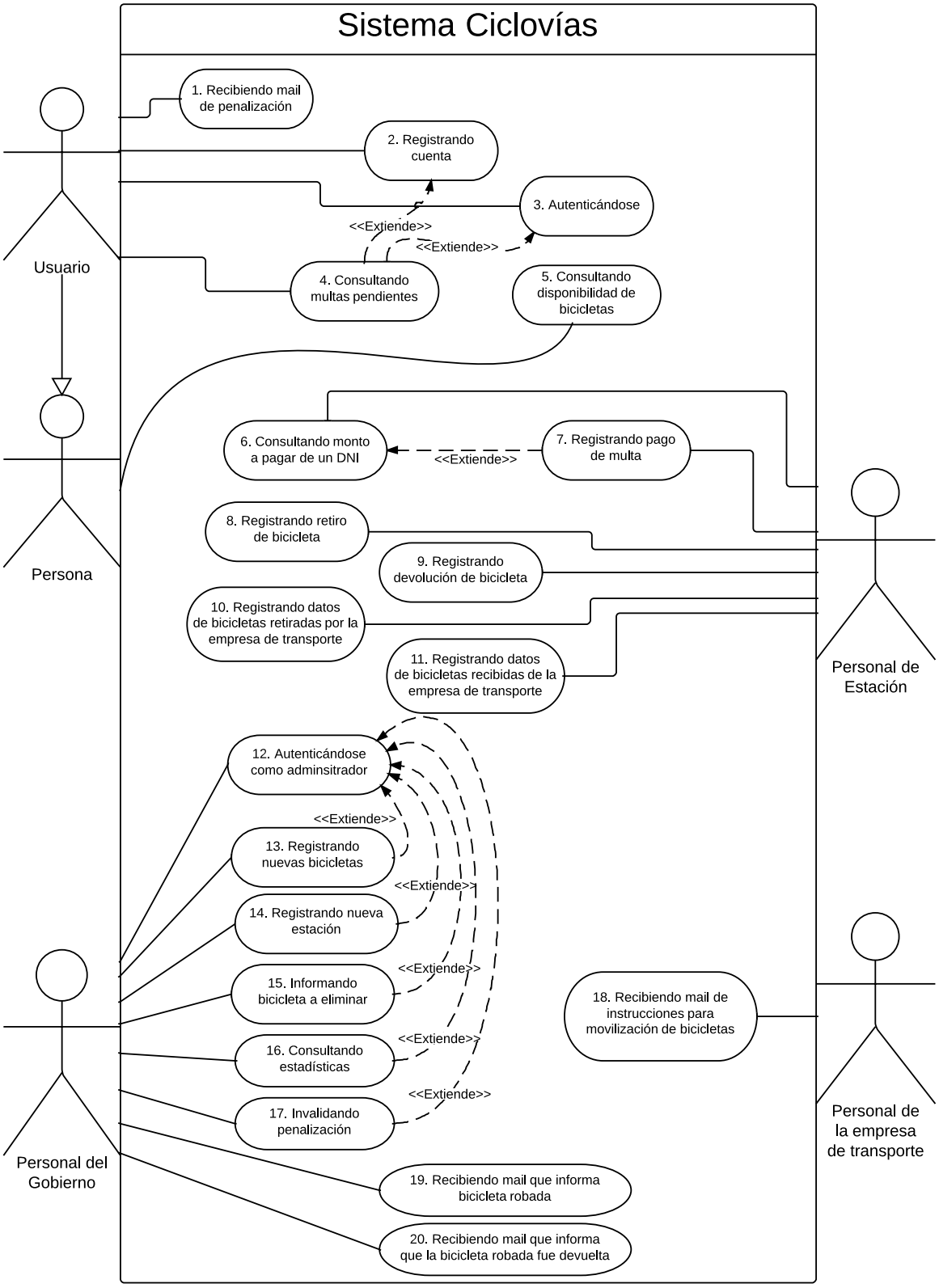
A continuación listaremos algunos cambios realizados a la solución propuesta con respecto a las especificaciones presentadas previamente.

- Cuando el empleado de estación consulta el monto de una multa que debe ser pagada por un usuario, ingresa en el sistema el DNI de este último. Aquí agregamos que el sistema valide ese DNI e indique por pantalla que hubo un error si este no pertenecía a ningún usuario registrado.
- Las operaciones de registro (no de consulta) que pueden ser realizadas a través de un radio requieren que un empleado de otra estación ingrese los datos al sistema. Para esto, fue necesario agregar la posibilidad de que el empleado de estación ingrese el ID de ésta para realizar dichas operaciones y así permitir que ingrese datos de otra estación. Además se agregó una validación de ese ID (debe corresponder a una estación registrada en el sistema).
- Cuando sale o llega un camión con bicicletas, el empleado de estación registra los IDs de las bicicletas que envía o recibe. Las bicicletas que llegan deben coincidir con las que salieron. Pero podría suceder que en la estación origen se informaran ciertas bicicletas pero se enviaran otras. Para poder corregir este error, debimos agregar:
  - Las órdenes de transporte de bicicletas tienen un ID para poder identificar qué envío es el que se está informando en la estación destino, y así asociarlo con la estación origen. Este ID es validado (debe corresponder a una orden de transporte registrada).
  - Agregamos una validación de IDs de bicicletas ingresadas en la estación destino para poder detectar los errores e informarlos.
- Cuando el gobierno registra bicicletas nuevas en el sistema, puede hacerlo por cantidad (antes sólo podía ingresar de a una). Además el sistema le muestra los IDs asignados a las bicicletas nuevas para que el personal del gobierno pueda grabarlos en las bicicletas y se indica también la estación inicial de las bicicletas. Por último, el personal del gobierno se encarga ahora de la distribución de las nuevas bicicletas, eliminando el requerimiento de enviar un mail a la empresa de transporte.
- Cuando el gobierno ingresa una nueva estación al sistema, agregamos la validación del nombre y dirección, para evitar estaciones duplicadas.

- Cuando el gobierno elimina bicicletas del sistema, agregamos la validación del ID de la bicicleta (debe existir).
- Cuando el gobierno consulta las estadísticas, ademas de poder consultar la demanda de cada estación, ahora puede:
  - Consultar que usuarios estan registrados.
  - Consultar que usuarios estan penalizados.
  - Averiguar cuales son las 5 estaciones con más solicitudes en hora pico.
  - Averiguar cuales son las 5 estaciones con más solicitudes fuera de hora pico.
- El gobierno ahora puede invalidar penalizaciones de determinado usuario.
- Cuando un usuario no devuelve la bicicleta por más de 24hs se informa al gobierno. Pero si la bicicleta es devuelta, agregamos un nuevo aviso al gobierno indicando esto.

4. Casos de uso

4.1. Diagrama



4.2. Detalle

Caso de uso 1: Recibiendo mail de penalización
Pre: True
Post: El usuario conoce vía mail la penalización otorgada por el sistema
Actores: Usuario

Curso normal	Curso alternativo
1. El sistema envía un mail al usuario informando las infracciones cometidas, indicando el motivo, el monto individual y total a pagar por las mismas.	
2. El usuario recibe el mail enviado con la información de su penalización.	
3. Fin caso de uso.	

Caso de uso 2: Registrando cuenta
Pre: True
Post: El usuario está registrado y autenticado en el sistema
Actores: Usuario

Curso normal	Curso alternativo
1. El usuario ingresa su número de DNI, email, nombre y contraseña.	
2. El sistema verifica que no esté registrado otro usuario con el email o DNI ingresado, que el campo DNI esté compuesto de números, que el email esté compuesto como usuario@dominio y que el nombre y la contraseña no se hayan dejado en blanco.	
3. El sistema guarda los datos ingresados.	3.1. Si los datos ingresados ya existían o fueron ingresados de manera errónea, mostrar que no es posible realizar el registro, y volver a 1.
4. El sistema muestra al usuario que el registro se realizó correctamente.	
5. El sistema autentica al usuario.	
6. Si lo desea, el usuario puede clickear un enlace para consultar sus multas pendientes. Es extendido por **CU 4**.	
7. Fin caso de uso.	

Caso de uso 3: Autenticándose
Pre: True
Post: El usuario está autenticado en el sistema
Actores: Usuario

Curso normal	Curso alternativo
1. El usuario ingresa su número de DNI y su contraseña.	
2. El sistema verifica que el usuario exista y que los datos ingresados sean correctos.	
3. El sistema muestra al usuario que la autenticación fue satisfactoria.	3.1. Si los datos ingresados son incorrectos, el sistema indica que la autenticación no fue satisfactoria, y vuelve a 1.
4. Si lo desea, el usuario puede clickear un enlace para consultar sus multas pendientes. Es extendido por **CU 4**.	
5. Fin caso de uso.	

Caso de uso 4: Consultando multas pendientes
Pre: El usuario está autenticado

Post: El usuario conoce las multas que tiene pendientes
Actores: Usuario

Curso normal	Curso alternativo
1. El sistema muestra una tabla informando las infracciones cometidas, indicando el motivo, el monto individual y total a pagar por las mismas. Si no tiene infracciones, se muestra un mensaje informándolo.	
2. Fin caso de uso.	

Caso de uso 5: Consultando disponibilidad de bicicletas
Pre: True
Post: El usuario conoce la disponibilidad de la estación deseada
Actores: Persona

Curso normal	Curso alternativo
1. El sistema muestra una lista de las estaciones a consultar por disponibilidad.	
2. La persona selecciona la estación deseada.	
3. El sistema muestra la disponibilidad de la estación deseada.	
4. Fin caso de uso.	

Caso de uso 6: Consultando monto a pagar de un DNI
Pre: True
Post: El sistema muestra las multas pendientes por pagar de un determinado DNI
Actores: Personal de la estación

Curso normal	Curso alternativo
1. El personal de la estación ingresa el DNI del usuario a consultar las multas.	
2. El sistema verifica que el DNI ingresado corresponda a un usuario.	
3. Si existen multas por abonar, el sistema muestra que tipo de multas y el importe total. Si no existen multas, el sistema muestra que está libre de deudas.	3.1. Si el DNI ingresado es incorrecto, mostrar mensaje de DNI equivocado y volver a 1.
4. Si el personal de la estación desea registrar el pago de las multas, hace click en el botón "Pagar". Es extendido por **CU 7**.	
5. Fin caso de uso	

Caso de uso 7: Registrando pago de multa
Pre: La persona con el DNI provisto registraba una multa sin abonar
Post: Se registra el cobro de la multa
Actores: Personal de la estación

Curso normal	Curso alternativo
1. El personal de la estación ingresa el DNI de un usuario que registra multas sin abonar.	
2. El sistema registra el pago de la multa y despenaliza al usuario.	
3. El sistema informa que la acción fue realizada exitosamente.	
4. Fin caso de uso.	

Caso de uso 8: Registrando retiro de bicicleta
Pre: True
Post: Se registra el retiro de bicicleta

Actores: Personal de la estación

Curso normal	Curso alternativo
1. El personal de estación ingresa el número de estación y presiona "Siguiente". Si no lo ingresa, por default se toma el número de la estación en la que se encuentra.	
2. El sistema registra la petición de una bicicleta.	2.1. Si el número de estación no es válido, el sistema lo indica por pantalla. Fin CU.
3. El sistema verifica el stock de la estación indicada.	
4. El sistema reserva una bicicleta del stock hasta el fin del CU.	4.1. Si no hay stock, muestra que no hay stock. Fin CU.
5. El personal de la estación ingresa el DNI.	
6. El sistema verifica que el usuario esté registrado.	
7. El sistema verifica que el usuario no esté penalizado.	7.1. Si el usuario no está registrado, muestra que no existe en el sistema. Fin CU.
8. El personal de la estación ingresa el número de la bicicleta a asignar al usuario.	8.1. Si el usuario está penalizado, se informa que lo está. Fin CU.
9. El sistema verifica que el ID de la bicicleta ingresada pertenezca a una bicicleta en la estación.	
10.El sistema registra la entrega de la bicicleta guardando ID de estación, DNI, ID de bicicleta, fecha y hora actual.	10.1.Si el ID ingresado es erróneo, muestra que es incorrecto y vuelve a 7.
11.Fin caso de uso.	

Detallamos las actividades e interacciones que realizan los actores involucrados en el proceso que incluye la utilización del caso de uso en el **DA Entrega de Bicicleta** 6.1.

Caso de uso 9: Registrando devolución de bicicleta
Pre: El usuario había retirado una bicicleta
Post: Se registra la devolución de la bicicleta entregada
Actores: Personal de la estación

Curso normal	Curso alternativo
1. El personal de estación ingresa el número de estación y presiona "Siguiente". Si no lo ingresa, por default se toma el número de la estación en la que se encuentra.	
2. El personal de la estación puede ingresar o no el número de DNI del usuario que devuelve la bicicleta. Si no lo ingresa, el sistema muestra una advertencia de posible penalización al usuario que retiró la bicicleta.	2.1. Si el número de estación no es válido, el sistema lo indica por pantalla. Fin CU.
3. El personal de la estación ingresa el ID de la bicicleta devuelta y el estado de la misma ("Buen Estado" o "Mal Estado").	
4. El sistema valida que el usuario que entregó la bicicleta sea el mismo que la retiró, que no haya usado la bicicleta más de una hora y que la bicicleta devuelta no esté en mal estado.	
5. Si falla alguna de las validaciones del paso 4, se penaliza al usuario y se informa por pantalla el motivo. Ver **DA "Penalizaciones"**.	
6. El sistema registra la devolución de la bicicleta, aumenta el stock y muestra que la devolución se realizó correctamente.	
7. Fin caso de uso.	

Detallamos las actividades e interacciones que realizan los actores involucrados en el proceso que incluye la utilización del caso de uso en el **DA Devolución de bicicleta** 6.2.

Caso de uso 10: Registrando datos de bicicletas retiradas por la empresa de transporte
Pre: El sistema dió la orden de mover bicicletas y descontó el stock de las mismas de la estación (y marcó esa cantidad como "reservada")
Post: El personal de la estación registra las bicicletas que se retirarán
Actores: Personal de la estación

Curso normal	Curso alternativo
1. El personal de estación ingresa el número de estación y presiona "Siguiente". Si no lo ingresa, por default se toma el número de la estación en la que se encuentra.	
2. El personal de la estación ingresa el ID relacionado al envío.	2.1. Si el número de estación no es válido, el sistema lo indica por pantalla. Fin CU.
3. El sistema verifica que el ID ingresado del envío sea un envío que salga de la estación donde se ingresó y que no haya sido completado anteriormente.	
4. El sistema muestra la cantidad de bicicletas que se necesitan trasladar y los campos para ingresar los ID's de las mismas.	4.1. Si el ID de envío ingresado es incorrecto, volver a 3) y mostrar mensaje.
5. El personal de la estación ingresa los ID de las bicicletas a entregar a la empresa de transporte.	
6. El sistema verifica que los IDs de las bicicletas ingresadas pertenezcan a bicicletas en la estación.	
7. El sistema registra el retiro de las bicicletas con los ID ingresados.	7.1. SSi alguno de los IDs ingresados es erróneo, muestra que es incorrecto y vuelve a 2).
8. El sistema muestra que la operación fue realizada exitosamente.	
9. Fin caso de uso.	

Detallamos las actividades e interacciones que realizan los actores involucrados en el proceso que incluye la utilización del caso de uso en el **DA Empresa de transporte: retiro de bicicletas** 6.6.

Caso de uso 11: Registrando datos de bicicletas recibidas de la empresa de transporte
Pre: Llega un camión con bicicletas y las descarga en la estación
Post: Se registra en el sistema la llegada de las bicicletas
Actores: Personal de la estación

Curso normal	Curso alternativo
1. El personal de estación ingresa el número de estación y presiona "Siguiente". Si no lo ingresa, por default se toma el número de la estación en la que se encuentra.	
2. El personal de la estación ingresa el ID relacionado al envío.	
3. El sistema verifica que el ID ingresado del envío sea un envío que tuviera como destino el ID de la estación ingresada y no se haya completado anteriormente.	
4. El sistema muestra la cantidad de bicicletas que deberían llegar y los campos para ingresar los ID's de las mismas.	4.1. Si el ID de envío ingresado es incorrecto, volver a 3) y mostrar mensaje.
5. El personal de la estación ingresa los ID de las bicicletas recibidas.	5.1. Si el número de estación no es válido, el sistema lo indica por pantalla. Fin CU.
6. El sistema verifica que los ID de las bicicletas ingresadas estuvieran siendo transportadas hacia esta estación.	
7. El sistema registra la ubicación de las bicicletas con los ID ingresados y actualiza el stock de la estación.	7.1. Si existe algún ID que no concuerda con el transporte realizado, se muestra por pantalla cuál ID está erróneo y el personal de la estación puede optar por volver a 1) y re ingresar correctamente el ID o en el caso que los IDs ingresados sean correctos, clickear "Omitir" y el sistema se encargará de corregir el error usando los datos de registro.
8. El sistema muestra que la operación fue realizada exitosamente.	
9. Fin caso de uso.	

Aclaración de 7.1: Veamos cómo el sistema realiza la corrección con un ejemplo: Tenemos la estación A y la estación B. Llega un camión a la estación A con la orden de transportar 5 bicicletas desde allí hasta la estación B. El personal de la estación en A ingresa los 5 IDs (que corresponden a bicicletas en la estación A) y el sistema las valida. Luego le entrega al camión 5 bicicletas, de las cuales 3 son erróneas (no se corresponden con ninguno de los IDs ingresados). Luego el camión viaja y las entrega en la estación B. El personal de la estación B ingresa los 5 IDs de las bicicletas que le llegaron y el sistema rechaza 3 de ellas porque no coinciden con el registro de envío. Entonces el personal clickea "Omitir" y el sistema intercambia las ubicaciones de las 3 bicicletas que llegaron con las 3 que estaban registradas en el envío. Observar que las 3 que viajaron y que según el sistema estaban en la estación A, no podrán ser retiradas hasta que no se haga la corrección (ver paso 9 de **CU 8**).

Detallamos las actividades e interacciones que realizan los actores involucrados en el proceso que incluye la utilización del caso de uso en el **DA Empresa de transporte: entrega de bicicletas** 6.7.

Caso de uso 12: Autenticándose como administrador
Pre: True
Post: El personal del gobierno está autenticado como administrador
Actores: Personal del gobierno

Curso normal	Curso alternativo
1. El personal del gobierno ingresa su usuario y su contraseña.	
2. El sistema verifica que el usuario exista y que los datos ingresados sean correctos.	
3. El sistema muestra al usuario que la autenticación fue satisfactoria.	3.1. Si los datos ingresados son incorrectos, el sistema indica que la autenticación no fue satisfactoria, y vuelve a 1.
4. Si lo desea, el personal del gobierno puede hacer click en alguno de los siguientes enlaces: ▪ Registrar nuevas bicicletas. Es extendido por **CU 13**. ▪ Registrar una nueva estación. Es extendido por **CU 14**. ▪ Informar la eliminación de una bicicleta. Es extendido por **CU 15**. ▪ Estadísticas del sistema. Es extendido por **CU 16**. ▪ Invalidar penalizaciones de un usuario. Es extendido por **CU 17**.	
5. Fin caso de uso.	

Caso de uso 13: Registrando nuevas bicicletas
Pre: El personal del estado está autenticado
Post: Nuevas bicicletas registradas en el sistema y el personal del gobierno conoce los IDs asignados a las bicicletas.
Actores: Personal del gobierno

Curso normal	Curso alternativo
1. El personal del gobierno de mar chiquita ingresa la cantidad de bicicletas nuevas y el ID de la estación inicial.	
2. El sistema registra el número de bicicletas ingresado por el personal, asignándole a cada bicicleta registrada un ID único.	
3. El sistema informa al personal los IDs de las bicicletas registradas.	
4. Fin caso de uso.	

Detallamos las actividades e interacciones que realizan los actores involucrados en el proceso que incluye la utilización del caso de uso en el **DA Agregar bicicletas al sistema** 6.8.

Caso de uso 14: Registrando nueva estación
Pre: El personal del gobierno está autenticado
Post: Se registra en el sistema la nueva estación
Actores: Personal del gobierno

Curso normal	Curso alternativo
1. El personal del gobierno ingresa el nombre de la nueva estación y su dirección, indicando si pertenece al centro o a la periferia.	
2. El sistema verifica si ya existe una estación con el mismo nombre o la misma dirección.	
3. El sistema muestra que el ingreso de la nueva estación fue correcto.	3.1. Si existe una estación con el mismo nombre o la misma dirección, mostrar cuál fue el ingreso erróneo y volver a 1.
4. Fin caso de uso.	

Caso de uso 15: Informando bicicleta a eliminar
Pre: El personal del gobierno está autenticado
Post: Una bicicleta es eliminada del sistema
Actores: Personal del gobierno

Curso normal	Curso alternativo
1. El personal del gobierno ingresa el ID de la bicicleta a eliminar del sistema.	
2. El sistema verifica que el ID de la bicicleta a eliminar corresponda a una bicicleta en el sistema.	
3. El sistema elimina la bicicleta con ID ingresado.	3.1. Si el ID es incorrecto, mostrar que el ID ingresado no es válido y volver a 1.
4. El sistema muestra que la operación se realizó correctamente e informa la última ubicación de la bicicleta.	
5. Fin caso de uso.	

Caso de uso 16: Consultando estadísticas
Pre: El personal del gobierno está autenticado
Post: El personal del gobierno conoce estadísticas del sistema
Actores: Personal del gobierno

Curso normal	Curso alternativo
1. El sistema muestra en pantalla una tabla con la siguiente información: ▪ Información general actualizada: • Cantidad de usuario registrados. • Cantidad de usuarios en infracción. ▪ Información de los últimos 7 días: • Por cada estación: ◦ Promedio de bicicletas solicitadas (retiradas y no retiradas) en hora pico. ◦ Promedio de bicicletas solicitadas fuera de hora pico. • Las 5 estaciones con mayor promedio en hora pico. • Las 5 estaciones con mayor promedio fuera de hora pico.	
2. Fin caso de uso.	

Caso de uso 17: Invalidando penalización
Pre: El personal del gobierno está autenticado
Post: El personal del gobierno despenaliza a un usuario.
Actores: Personal del gobierno

Curso normal	Curso alternativo
1. El personal del gobierno indica el DNI del usuario que desea despenalizar.	
2. El sistema verifica que el DNI esté registrado y tenga alguna penalización.	
3. El sistema muestra una tabla informando las infracciones cometidas por el usuario, indicando el motivo, el monto individual y total a pagar por las mismas.	3.1. Si el DNI no estaba registrado, el sistema muestra por pantalla que el DNI es erróneo y regresa a 1. Si el DNI estaba registrado pero no tenía ninguna penalización, muestra un mensaje informándolo y vuelve a 1.
4. El personal del gobierno selecciona las penalizaciones que desea eliminar de la tabla y hace clic en "Eliminar".	
5. El sistema elimina todo registro de las penalizaciones seleccionadas por el personal del gobierno.	
6. Fin caso de uso.	

Detallamos las actividades e interacciones que realizan los actores involucrados en el proceso que incluye la utilización del caso de uso en el **DA Justificar penalizaciones** 6.4.

Caso de uso 18: Recibiendo mail de instrucciones para movilización de bicicletas
Pre: True
Post: El personal de la empresa de transporte recibe el mail con las indicaciones de cómo mover las bicicletas
Actores: Personal de la empresa de transporte

Curso normal	Curso alternativo
1. El sistema envía mail al personal de la empresa de transporte informando cómo mover las bicicletas. En una tabla, por cada entrada indica: ID del envío, estación orígen, estación destino, la dirección de cada estación y la cantidad de bicicletas a trasladar.	
2. El personal de la empresa de transporte recibe el mail.	
3. Fin caso de uso.	

Caso de uso 19: Recibiendo mail que informa bicicleta robada
Pre: True
Post: El personal del gobierno se entera de que una bicicleta fue robada y del responsable
Actores: Personal del gobierno

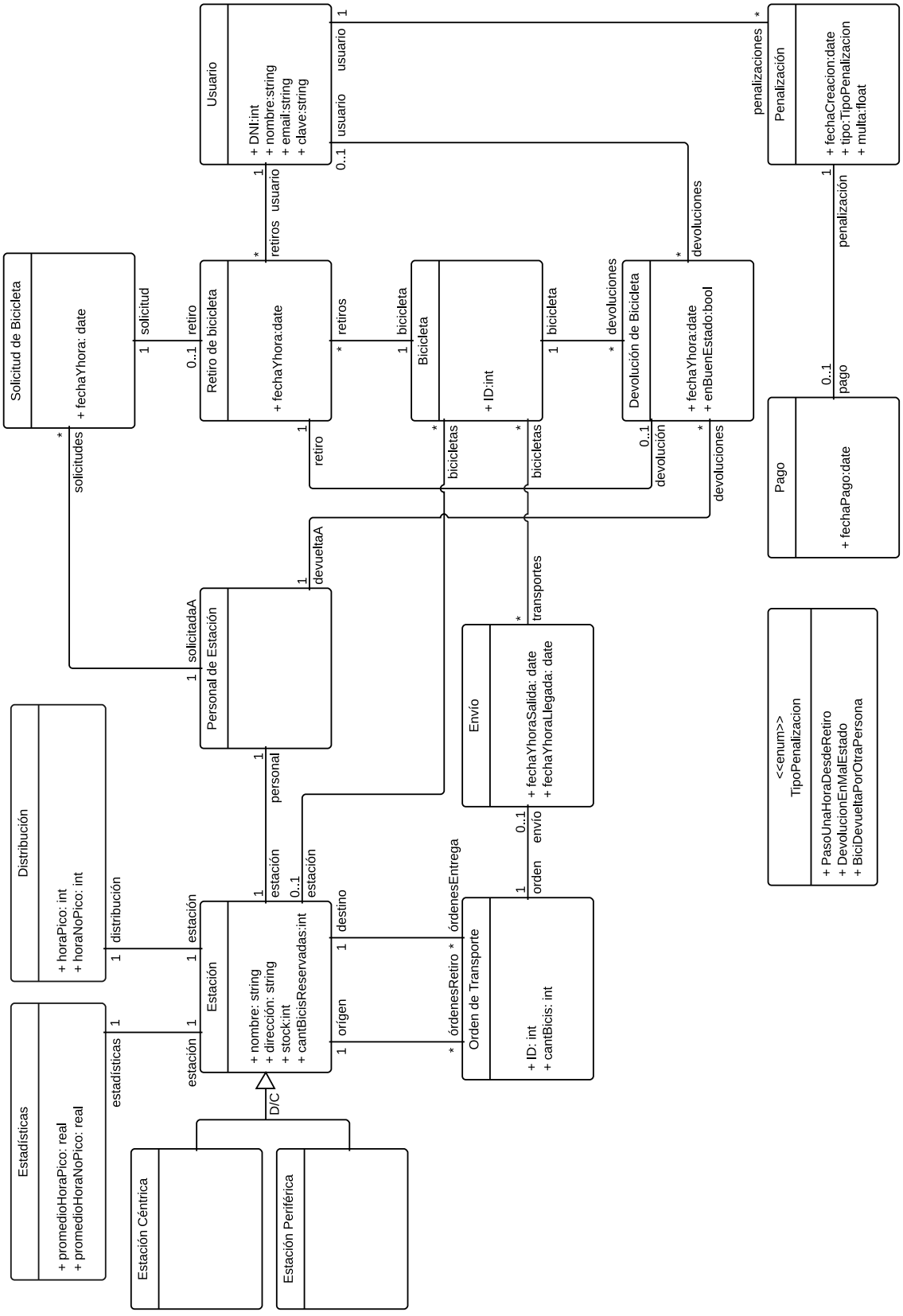
Curso normal	Curso alternativo
1. El sistema envía mail al personal del gobierno informando el ID de una bicicleta que fue retenida por un usuario por más de 24hs. Le indica también el nombre, el DNI y el mail del usuario que la retiró por última vez.	
2. El personal del gobierno recibe el mail.	
3. Fin caso de uso.	

Caso de uso 20: Recibiendo mail que informa que la bicicleta robada fue devuelta
Pre: True
Post: El personal del gobierno se entera de que una bicicleta anteriormente reportada como robada ya fue devuelta
Actores: Personal del gobierno

Curso normal	Curso alternativo
1. El sistema envía mail al personal del gobierno informando el ID de una bicicleta que fue retenida por un usuario por más de 24hs y ya fue devuelta. Le indica también el nombre, el DNI y el mail del usuario que la retiró por última vez y del usuario que la devolvió (sólo en el caso que existiera esta información).	
2. El personal del gobierno recibe el mail.	
3. Fin caso de uso.	

5. Modelo conceptual

5.1. Diagrama



A continuación realizaremos aclaraciones sobre algunas de las clases y asociaciones presentadas previamente:

Solicitud de Bicicleta: Representa una solicitud que puede ser realizada por cualquier persona (registrada o no) en el sistema. Observar que puede tener 0 o 1 retiros, porque la solicitud puede ser aceptada o no.

Retiro de Bicicleta: Si una solicitud es aceptada por el sistema, entonces se realiza el retiro (en este caso el "Usuario" asociado es el que retira la bicicleta y también el que realizó la solicitud previamente). Observar que puede tener 0 o 1 devoluciones porque la bicicleta podría no ser devuelta.

Devolución de Bicicleta: Representa la devulución de una bicicleta. Está asociada al retiro correspondiente a la misma bicicleta. Puede ser devuelta por una persona que no sea usuario o por un usuario registrado. Si es usuario, podría ser tanto el mismo usuario que la retiró como un usuario que había retirado otra bicicleta y la intercambió con el que la retiró. La bicicleta asociada a la devolución es la misma asociada al retiro, pero incluímos esta información redundante para simplificar la lectura del diagrama y de los invariantes en OCL.

Orden de Transporte: Representa la orden enviada por el sistema a la empresa de transporte para movilizar bicicletas (ver **CU 18**). Tiene una estación orígen, una estación destino y puede tener o no un envío.

Envío: Representa el envío de las bicicletas según una orden de transporte. Tiene una fecha de salida y, si ya llegó a destino, una fecha de llegada. Ambas fechas se corresponden con el momento en que el envío/recepción es registrado (ver **CU 10** y **CU 11**). El envío también tiene asociada la lista de bicicletas que transporta (que son elegidas en la estación orígen de la orden de transporte).

Estadísticas: Son las estadísticas que se calculan por estación, correspondientes a los datos de los últimos 7 días. Ver detalles de los cálculos en OCL.

Distribución: Por cada estación, indica la cantidad de bicicletas que debe tener cuando se realiza la distribución en hora pico y por la mañana (hora no pico). Observar que esto *no* es la cantidad de bicicletas que el sistema le enviará sino la cantidad que debería tener luego del envío.

5.2. OCL

5.2.1. Penalizaciones

Ningún campo de las penalizaciones es null

```
context Penalización
inv: self.fechaDeCreación <> null and
     self.tipoDePenalización <> null and
     self.multa <> null and
     (self.pago <> null implies self.pago.fechaPago <> null)
```

Si hay una penalización en mal estado, existe una devolución en mal estado

```
context Penalización
inv: self.tipo = DevoluciónEnMalEstado implies self.usuario.devoluciones->exists(d |
         d.enBuenEstado = false and d.fechaYHora = self.fechaCreación)
```

Si hay una multa por devolución por otra persona, existe una devolución sin usuario o con un usuario distinto

```
context Penalización
inv: self.tipo = BiciDevueltaPorOtraPersona implies
```

```
          self.usuario.retiros->select(r | r.devolución <> null)
                               ->exists(r | (r.devolución.usuario = null or
                                             r.devolución.usuario != r.usuario) and
                                            r.devolución.fechaYHora = self.fechaCreación)
```

Si hay una multa por pasar más de una hora, existe un retiro sin devolución por más de una hora

```
context Penalización
inv: self.tipo = PasoUnaHoraDesdeRetiro implies self.usuario.retiros->exists(r |
         (r.devolución = null or (r.devolución <> null and
                                  r.devolución.fechaYHora >= r.fechaYHora + 1h)) and
         self.fechaCreación = r.fechaYHora + 1h)
```

Para cada devolución en mal estado existe una penalización

```
context DevoluciónDeBicicleta
inv: self.enBuenEstado = false implies
         self.retiro.usuario.penalizaciones->exists(p |
             p.fechaCreación = self.fechaYHora and
             p.tipo = DevoluciónEnMalEstado)
```

Si existe una devolución sin usuario o con un usuario distinto, hay multa por devolución de otro usuario

```
context Devolución de Bicicleta
inv: (self.usuario = null or self.retiro.usuario <> self.usuario) implies
         self.retiro.usuario.penalizaciones->exists(p |
             p.tipo = BiciDevueltaPorOtraPersona and p.fechaCreación = self.fechaYHora)
```

Si existe un retiro de bicicleta donde ya pasó más de una hora, hay una multa (automática)

```
context RetiroDeBicicleta
inv: (self.devolución = null and today() >= self.fechaYHora + 1h) or
     (self.devolución <> null and self.fechaYHora <= self.devolución.fechaYHora + 1h) implies
         self.usuario.penalizaciones->exists(p |
             p.fechaCreación = self.fechaYHora + 1h and
             p.tipo = PasoUnaHoraDesdeRetiro)
```

Los pagos ocurren en una fecha después de la penalización

```
context Penalización
inv: self.pago <> null implies self.fechaCreación <= self.pago.fechaPago
```

5.2.2. Retiros

Todo retiro tiene una solicitud con la misma hora

```
context RetiroDeBicicleta
inv: self.solicitud.fechaYHora = self.fechaYHora
```

No puede existir un retiro de bicicletas si existía alguna penalización sin pagar antes del retiro

```
context RetiroDeBicicleta
inv: self.usuario.penalizaciones->forAll(p |
         (p.fechaCreación < self.fechaYHora) implies
             (p.pago <> null and
             p.pago.fechaPago <= self.fechaYHora))
```

Un usuario no puede tener más de una bicicleta

```
context Usuario
inv: self.retiros->select(r1 | r1.devolución = null)->size() <= 1
```

No hay solapamientos entre los retiros (franjas de entrega y devolución que se solapen)

```
context RetiroDeBicicleta
inv: RetiroDeBicicleta->allInstances()->forAll(r1, r2 |
         (r1 <> r2 and r1.devolución <> null and r2.devolución <> null) implies
             ((r1.fechaYHora < r2.fechaYHora implies
                 r1.devolución.fechaYHora < r2.fechaYHora) and
              (r1.fechaYHora > r2.fechaYHora implies
                 r1.devolución.fechaYHora > r2.fechaYHora)))
```

Los retiros tienen fecha mayor o igual a la de las devoluciones

```
context RetiroDeBicicleta
inv: self.devolucion <> null implies self.fechaYHora >= self.devolución.fechaYHora
```

5.2.3. Bicicletas

Los IDs de las bicicletas no son nulos

```
context Bicicleta
inv: self.id <> null
```

No hay otra bicicleta con el mismo ID

```
context Bicicleta
inv: Bicicleta.allInstances()->forAll(b1, b2 | b1 <> b2 implies b1.id <> b2.id)
```

Una bicicleta no puede estar en más de dos envíos (sin fecha de llegada)

```
context Bicicleta
inv: self.transportes->select(t | t.fechaYHoraLlegada = null)->size() <= 1
```

Una bicicleta está en una estación, en transporte o en posesión de un usuario

```
context Bicicleta
inv: -- Está en una estación
     (self.estación <> null and not self.transportes.exists(t |
         t.fechaYHoraLllegada = null and
         self.retiros->size() = self.devoluciones->size())) or

     -- Está en transporte
     (self.estación = null and
      self.transportes->exists(t | t.fechaYHoraLllegada = null) and
      self.retiros->size() = self.devoluciones->size()) or

     -- Está en posesión de un usuario
     (self.estación = null and
      not self.transportes->exists(t | t.fechaYHoraLllegada = null) and
      self.retiros->size() = self.devoluciones->size() + 1)
```

5.2.4. Orden de transporte

Los IDs de las ordenes de transporte nunca son null

```
context OrdenDeTransporte
inv: self.id <> null
```

Toda orden de transporte tiene un ID distinto de los otros

```
context OrdenDeTransporte
inv: OrdenDeTransporte.allInstances()->forAll(o1, o2 |
         o1 <> o2 implies o1.id <> o2.id)
```

Una orden de transporte no puede tener el mismo origen que destino

```
context OrdenDeTransporte
inv: self.orígen <> self.destino
```

5.2.5. Envíos

Los envíos tienen fecha de llegada mayor a la fecha de salida en el caso de que hayan llegado o null si no llegaron a destino

```
context Envío
inv: self.fechaYHoraLLegada = null or self.fechaYHoraSalida < self.fechaYHoraLlegada
```

Los envíos siempre existen con fecha de inicio

```
context Envío
inv: self.fechaYHoraSalida <> null
```

No hay solapamientos en envíos por cada bicicleta

```
context Bicicleta
inv: self.transportes->forAll(e1, e2 |
         (e1 <> e2 and e1.fechaYHoraLlegada <> null and e2.fechaYHoraLlegada <> null) implies
             ((e1.fechaYHoraSalida  < e2.fechaYHoraSalida implies
               e1.fechaYHoraLlegada < e2.fechaYHoraSalida) and
              (e1.fechaYHoraSalida  > e2.fechaYHoraSalida implies
               e1.fechaYHoraLlegada > e2.fechaYHoraSalida)))
```

Debe de haber la cantidad especificada de bicicletas por la orden en el envío

```
context Envío
inv: self.orden.cantBicis = self.bicicletas->size()
```

5.2.6. Usuarios

Ningún campo de usuarios puede ser Null

```
context Usuario
inv: self.dni <> null and
     self.nombre <> null and
     self.email <> null and
     self.email <> null and
     self.clave <> null
```

No existen usuarios con el mismos DNI y mail

```
context Usuario
inv: Usuario.allInstances()->forAll(u1, u2 |
        u1 <> u2 implies (u1.dni <> u2.dni and u1.email <> u2.email))
```

5.2.7. Estaciones

Las bicicletas reservadas concuerdan con la cantidad de bicicletas reservadas para envíos en las órdenes de transporte

```
context Estación
inv: self.órdenesRetiro->select(e | e.envío = null)
                       ->collect(e.cantBicis)
                       ->sum() = self.cantBicisReservadas
```

El stock más la cantidad de bicis reservadas es igual a la cantidad de bicicletas en la estación

```
context Estación
inv: self.stock + self.cantBicisReservadas = self.bicicletas->size()
```

Las estaciones no tienen el mismo nombre o dirección

```
context Estación
inv: Estación.allInstances()->forAll(e1, e2 |
        (e1 <> e2) implies (e1.nombre <> e2.nombre and e1.dirección <> e2.dirección))
```

Demanda promedio en hora pico y hora no-pico

```
context Estación
inv: self.estadísticas.promedioHoraPico =
     self.personal.solicitudes
         ->select(fechaYHora->dia >= today()->dia - 7 and
                  fechaYHora->hora >= 17 and
                  fechaYHora->hora <= 20)
         ->size() / 7
inv: self.estadísticas.promedioHoraNoPico =
     self.personal.solicitudes
         ->select(fechaYHora->dia >= today()->dia - 7 and
                  fechaYHora->hora < 17 or
                  fechaYHora->hora > 20)
         ->size() / 7
```

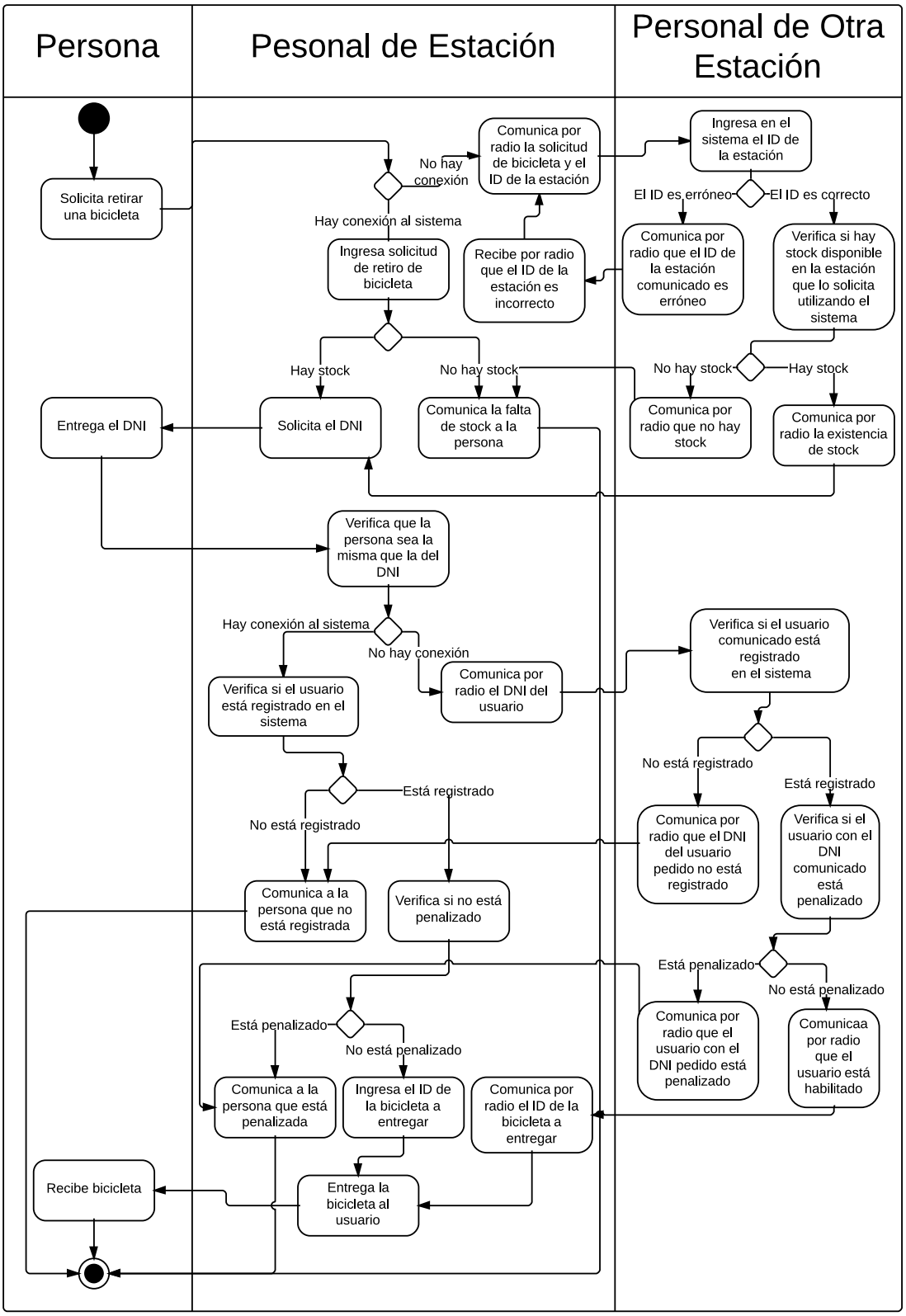
Distribución

```
context Estación
inv: self.distribución.horaNoPico =
         Bicicleta.allInstances()->size() /
         Estación.allInstances()->size()
inv: self.distribución.horaPico =
         (if self.oclIsTypeOf(EstaciónPeriférica) then
             self.estadísticas.promedioHoraPico * 0.25
         else
             self.estadísticas.promedioHoraPico
         endif) /
         Estación.allInstances()
             ->collect(if oclIsTypeOf(EstaciónPeriférica) then
```

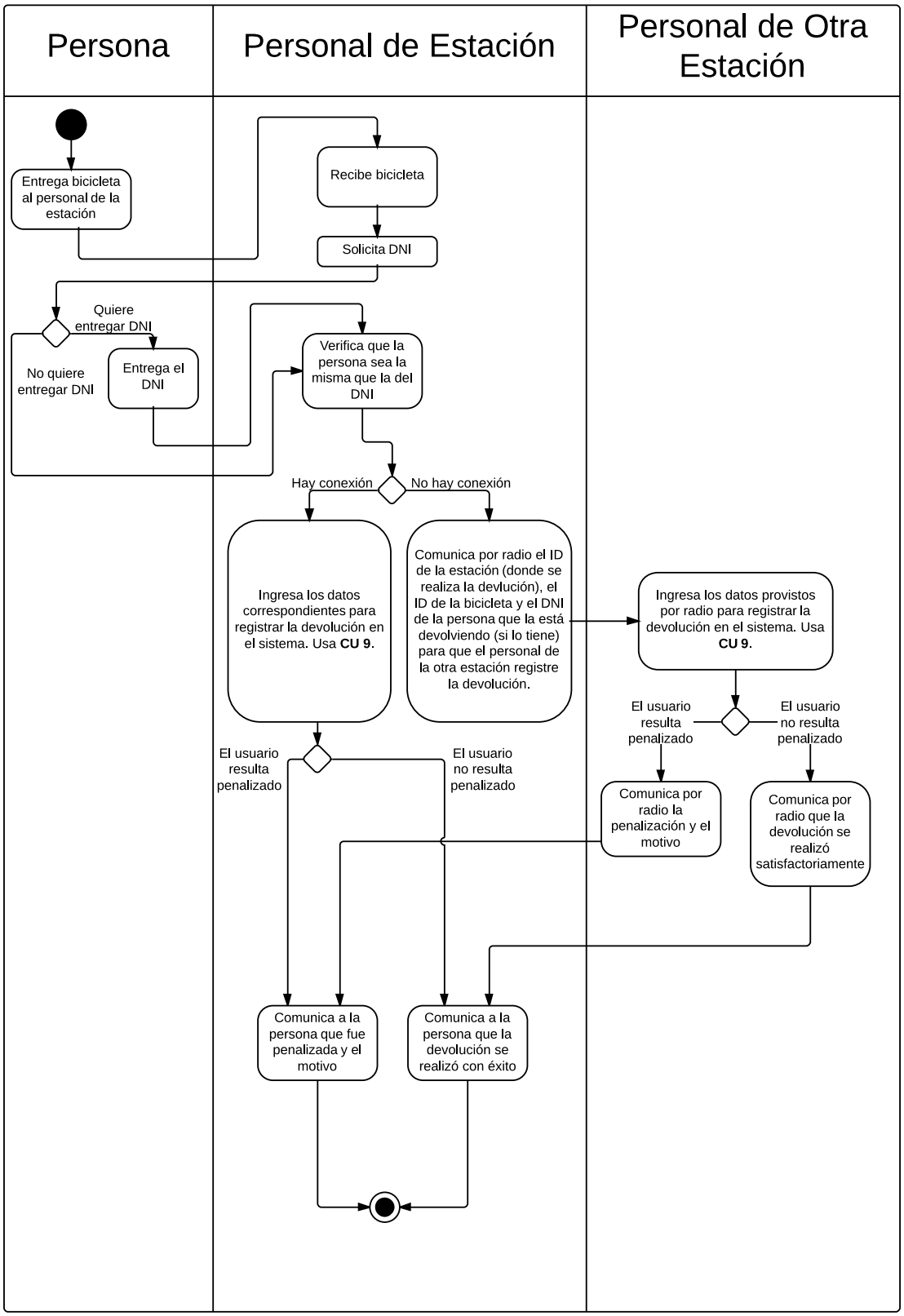
```
                    estadísticas.promedioHoraPico * 0.25
                else
                    estadísticas.promedioHoraPico
                endif)
    ->sum() *
Bicicleta.allInstances()->size()
```

6. Diagramas de actividad

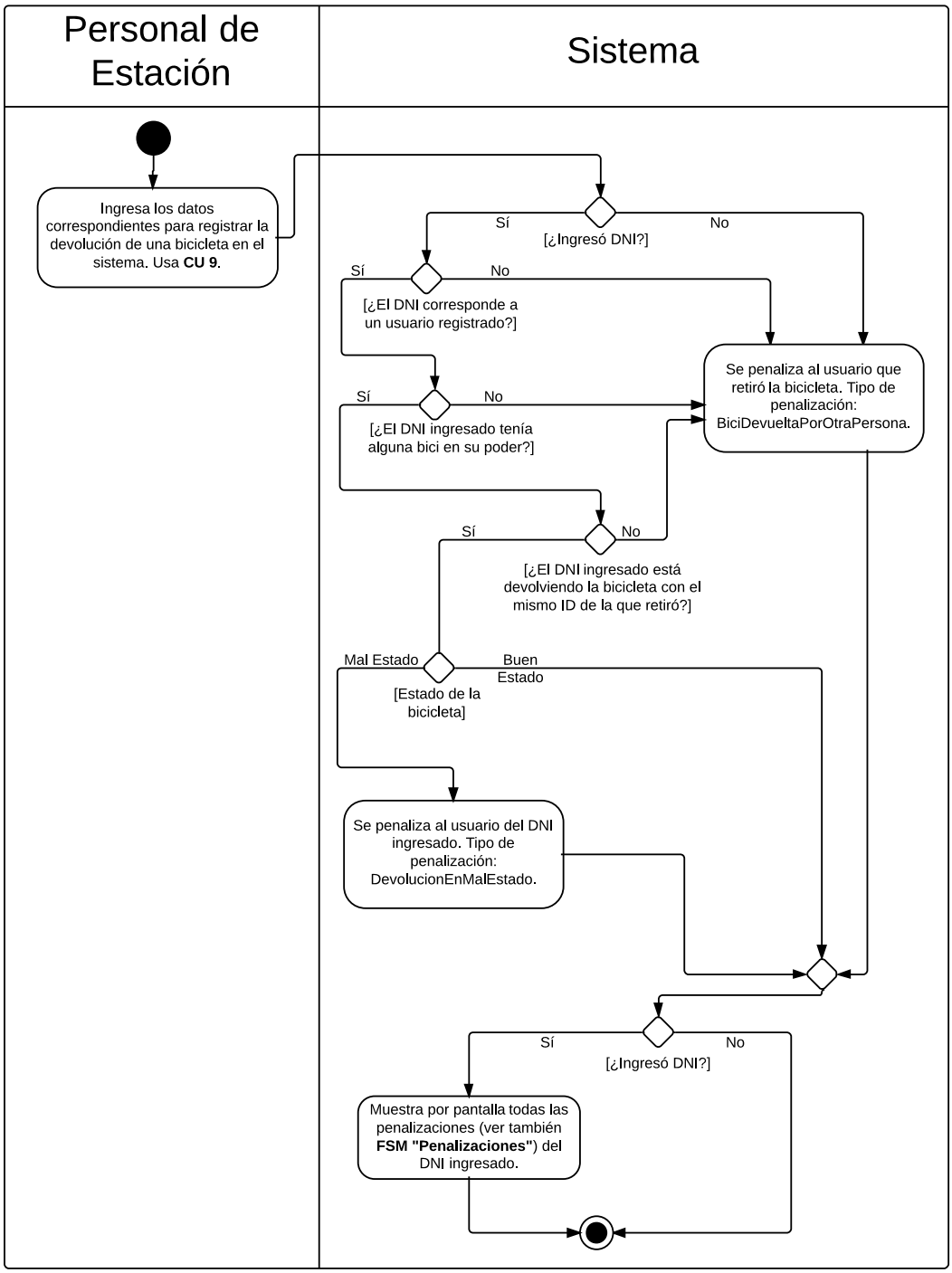
6.1. Entrega de bicicleta



6.2. Devolución de bicicleta



6.3. Penalizaciones



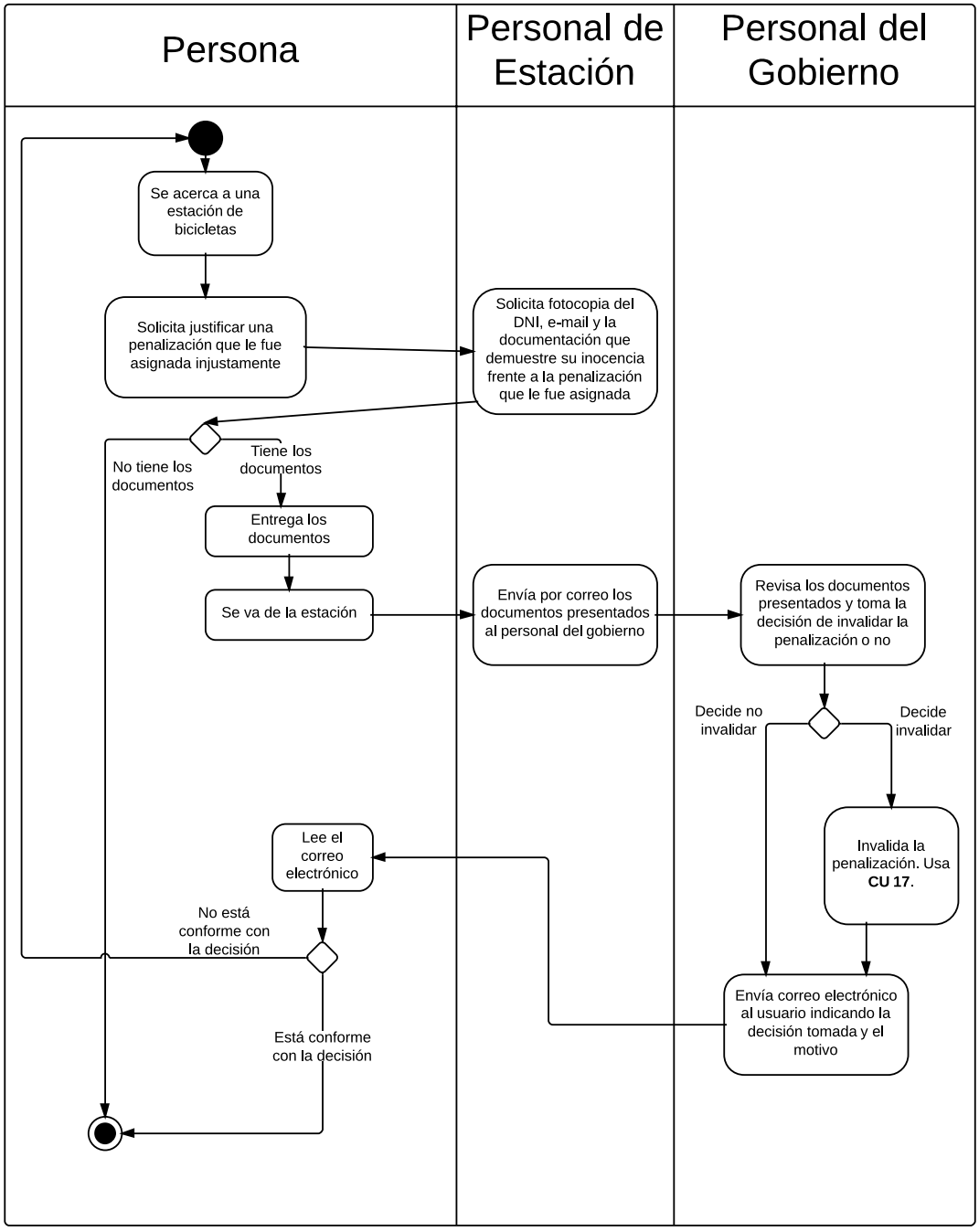
Este diagrama describe la asignación de penalizaciones ante la devolución de una bicicleta. Las reglas de penalización son las siguientes:

- Si la bicicleta es devuelta por una persona que no la retiró (sea usuario o no), el usuario que la retiró es penalizado con "BiciDevueltaPorOtraPersona".
- Si la bicicleta es devuelta por el usuario que la retiró y está en mal estado, el usuario (que la retiró y también devolvió) es penalizado con "DevoluciónEnMalEstado".

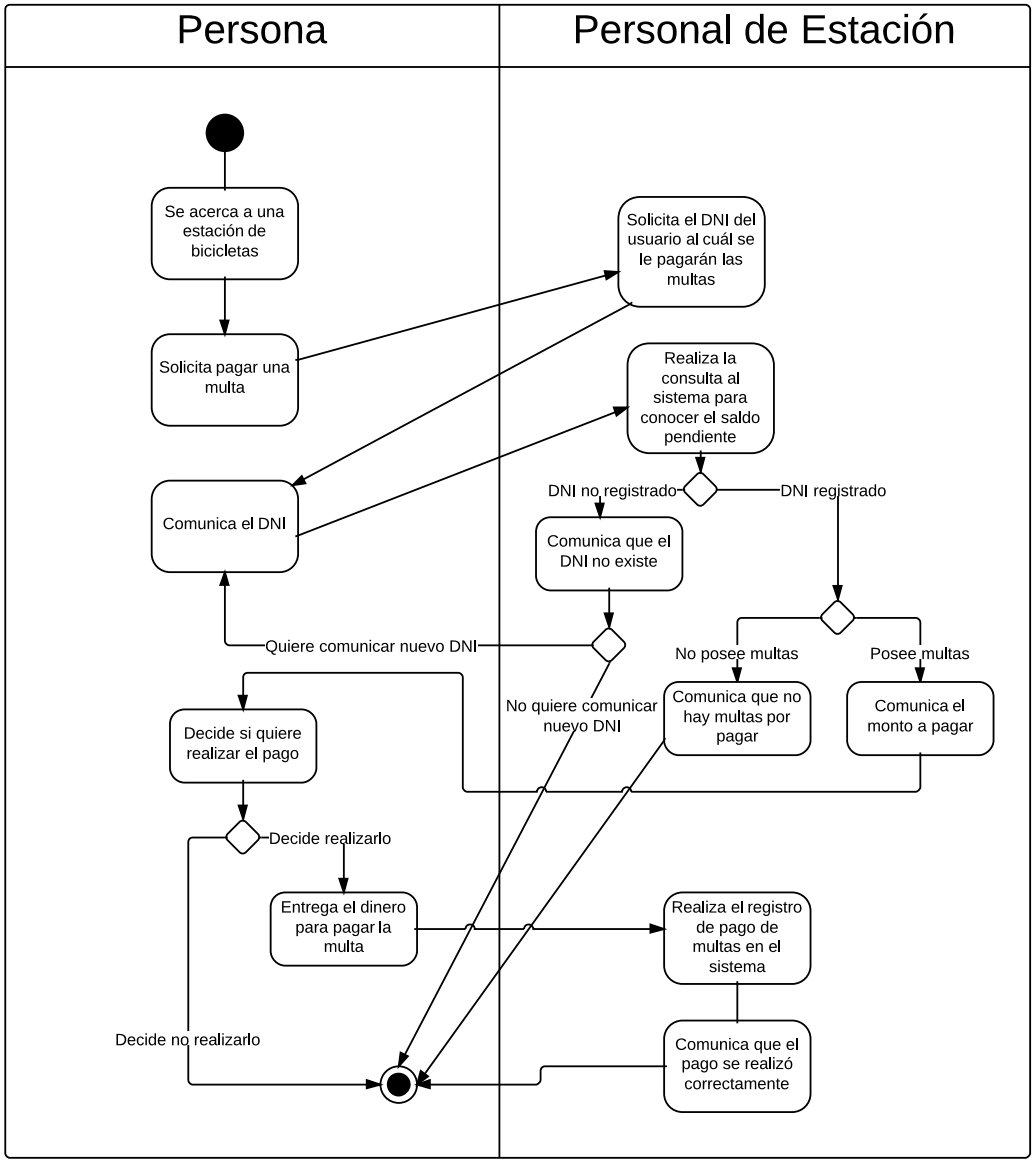
Observar que asumimos que siempre que una bicicleta es devuelta, había sido retirada por alguien (el "usuario que la retiró" siempre existe). Esto se debe a que no tenemos en cuenta en nuestro modelo la baja de usuarios ni tampoco el robo de bicicletas directamente de la estación (sin que hayan sido entregadas a un usuario).

Cabe aclarar que estas no son todas las penalizaciones posibles, sino que son únicamente las que se desencadenan ante la devolución de una bicicleta. Ver **FSM "Penalizaciones"** (Sección 7.3).

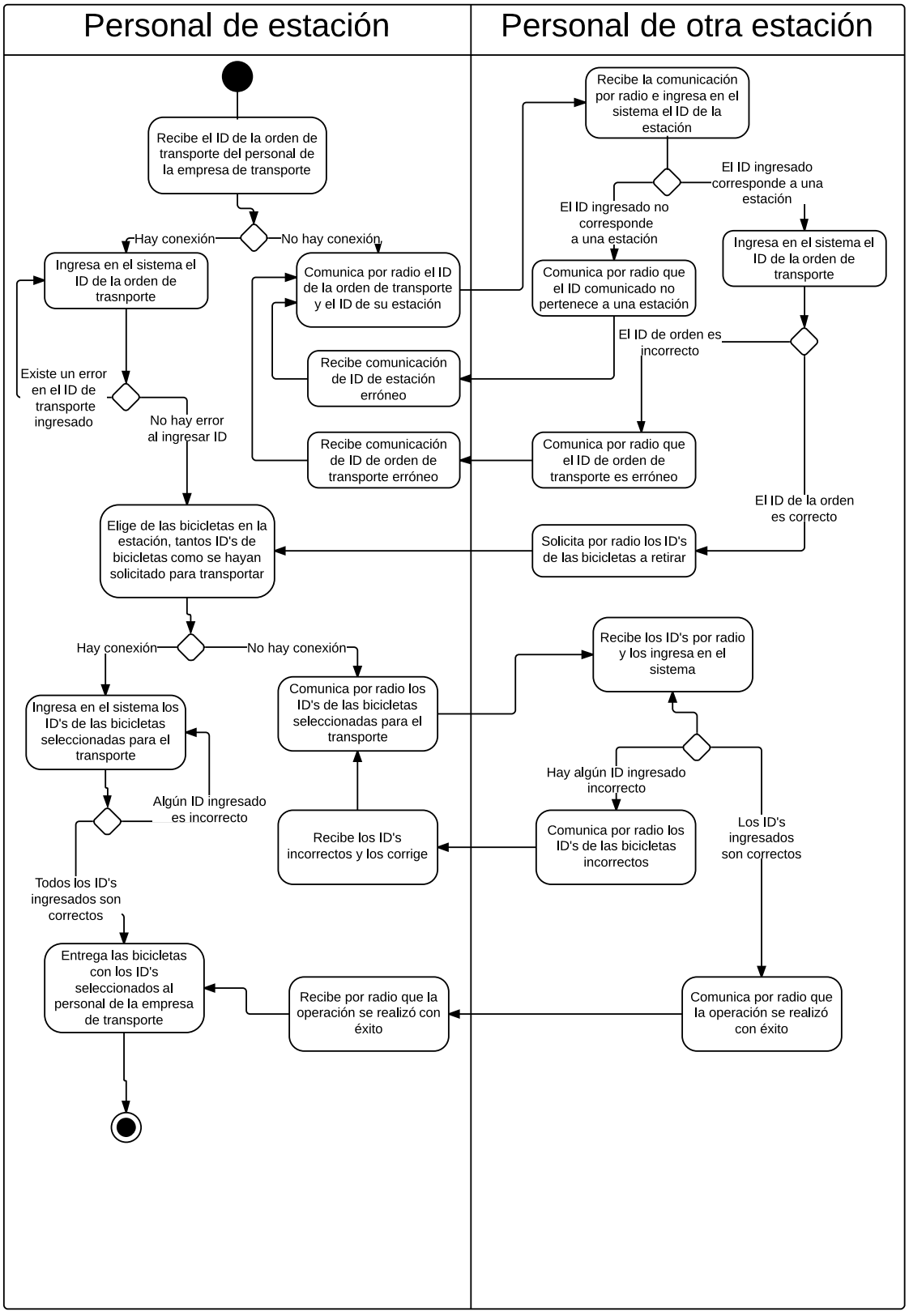
6.4. Justificar penalizaciones



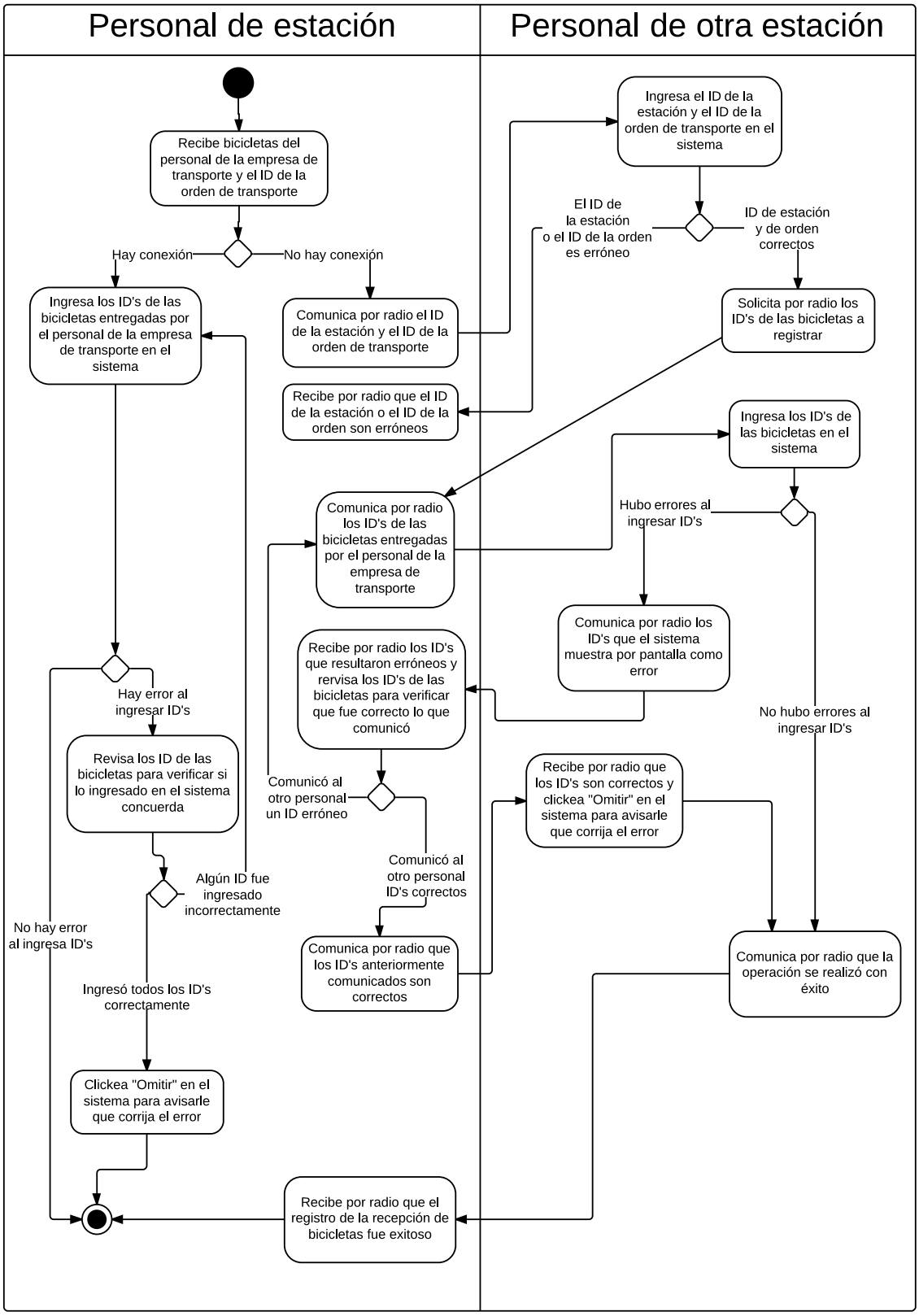
6.5. Pago de multas



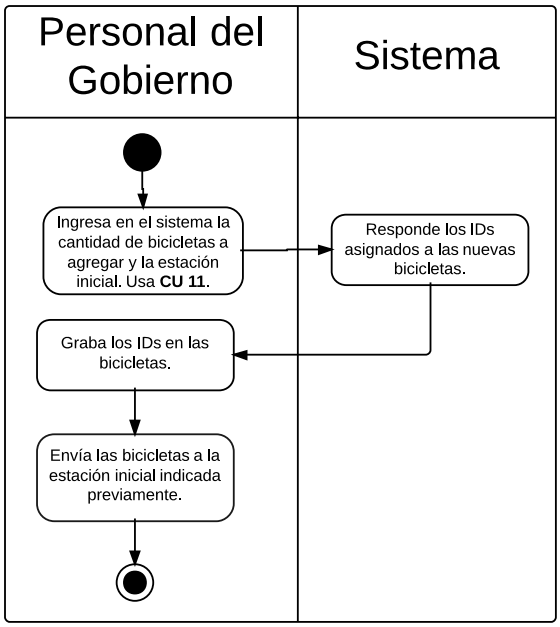
6.6. Empresa de transporte: retiro de bicicletas



6.7. Empresa de transporte: entrega de bicicletas

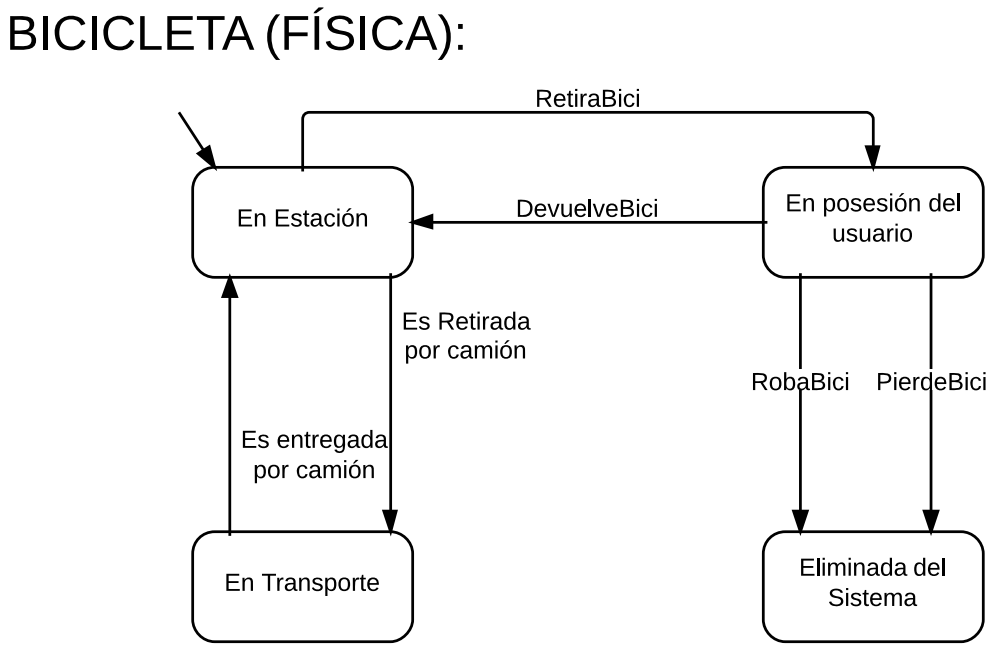


6.8. Agregar bicicletas al sistema

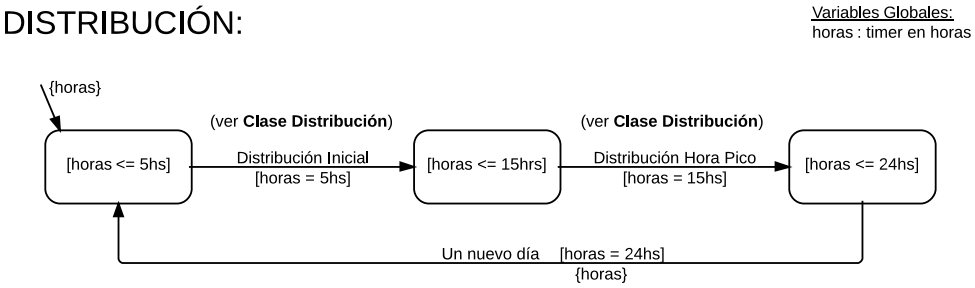


7. Máquinas de Estado

7.1. Bicicleta



7.2. Distribución de bicicletas



7.3. Penalizaciones