# Caso de estudio del proyecto de Grado

El sistema de parqueo de la Pontifica Universidad Javeriana Cali ha sido elegido como caso de estudio para emplear en conjunto las verificaciones formales y pruebas, con el fin de valorar y minimizar los errores en el dise\~no de un sistema distribuido. La figura 1 muestra el sistema de parqueo de veh\’iculos de la Pontificia Universidad Javeriana Cali, PUJC, y cuenta con lo siguiente:

1. Por lo menos con una entrada y una salida principal, en el caso de la figura 1 est\'an sobre la avenida Ca\~nasgordas.
2. Cada entrada y salida principal cuenta con un sistema que permite identificar las placas de los carros, un lector de carn\’e y un sistema b\’asico de sensores que garantiza que el usuario pueda ingresar o salir del sistema de parqueo.
3. Cuenta con un conjunto de zonas[[1]](#footnote-1) que est\’an bajo el mando de un controlador, en la figura 1 se muestran 5 controladores del sistema de parqueo de la PUJC.
4. Repositorios donde se encuentran los usuarios que tienen permitido el acceso al sistema de paqueo y sus placas registradas.

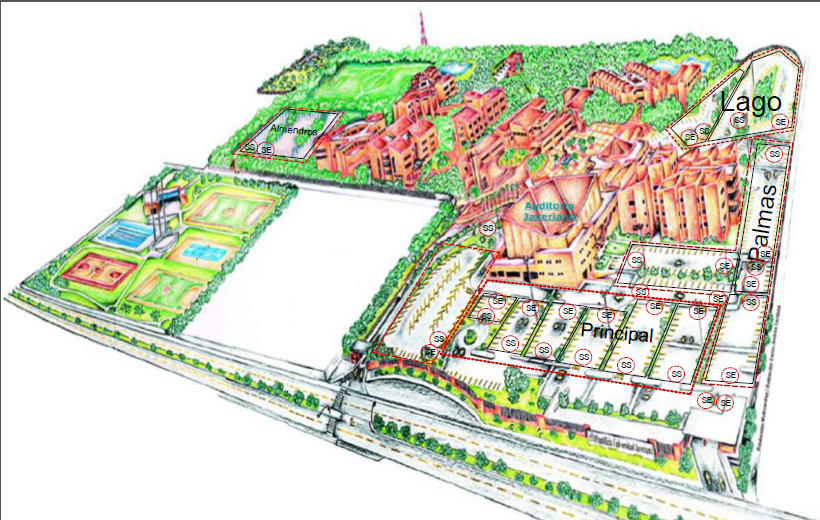


Figura 1. Sistema de Parqueo de la PUJC

La figura 1 muestra b\’asicamente 5 controladores de zonas que son:

1. Controlador “Principal” que tiene 6 zonas de parqueo.
2. Controlador “Docentes” que tiene 1 zona de parqueo.
3. Controlador “Palmas” que tiene 3 zonas de parqueo.
4. Controlador “Lago” que tiene 2 zonas de parqueo.
5. Controlador “Almendros” que tiene 1 zona de parqueos.

La figura 2 ilustra los componentes del sistema de parqueos de la PUJC. B\’asicamente un sistema de parqueo de veh\’iculos cuenta con *C* controladores de zonas y a su vez puede tener hasta *Z* zonas, estas últimas poseen un sistema de sensores capaces de informar a su controlador si un veh\’iculo ingres\’o o sali\’o de dicha zona. Adicionalmente el sistema puede tener hasta *M* entrada y *N* salidas principales respectivamente. Para garantizar que en el sistema solo ingresen usuarios registrados, por ejemplo: Estudiantes, docentes, personal administrativo, colaboradores, etc. Es indispensable contar con elementos que permitan cumplir dicho objetivo; Por ello, el sistema est\’a dise\~nado para que cada entrada y salida principal cuenten con una c\’amara cuyo prop\’osito es detectar la placa del veh\’iculo que esta por ingresar o salir del sistema, adem\’as de tener un lector de carn\’es que permite validar si el usuario hace parte del sistema o no; finalmente se requiere un sistema de sensores que permitan que el veh\’iculo ingrese o salga del sistema(PROFUNDIZAR MAS EL CASO QUE LA TALANQUERA TOQUE EL VEHICULO).

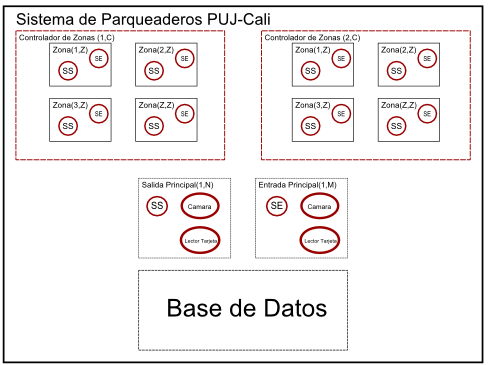


Figura 2. Componentes del Sistema de Parqueo de la PUJC

La tabla 1 describe las funciones que posee el sistema de parqueo de la PUJC y el tipo de usuario que puede hacer uso de \’estas.

|  |  |  |
| --- | --- | --- |
| **Funci\’on** | **Descripci\’on de la funci\’on** | **Usuario** |
| Crear Controlador de Zona | Permite que el sistema tenga m\’as controladores y a su vez m\’as capacidad de zonas que se traduce en aparcar m\’as carros. | ADMINISTRADOR |
| Crear una Zona en un Controlador Especifico | Permite anexar una zona en un controlador espec\’ifico ampliando la capacidad de \’este último. | ADMINISTRADOR |
| Crear una Entrada/Salida principal | Permite que el sistema tenga m\’as entradas y salidas para el acceso de usuarios al sistema de parqueo. | ADMINISTRADOR |
| Solicitar informaci\’on a los controladores de zona | Permite saber las plazas libres que posee cada uno de los controladores lo que posibilita que el usuario sepa en qu\’e controlador hay plazas libres. | ADMINISTRADOR/AUTOGESTI\’ON DEL SISTEMA |
| Inicializar las plazas libres y totales de una zona de un controlador en espec\’ifico | Permite inicializar las plazas libres de una zona de un controlador en espec\’ifico, posibilita tener zonas de diferentes capacidades. | ADMINISTRADOR |
| Reportar ingreso o salida de un usuario al sistema de parqueo | Reporta si un usuario ha ingresado al sistema de parqueo si y solo si: El c\’odigo del usuario y la placa de su veh\’iculo se encuentra registrado en la base de datos. | USUARIO |
| Reportar el ingreso o salida de un usuario en una zona | Reporta si un veh\’iculo ha ingresado o ha dejado una zona cuando se detecte una secuencia espec\’ifica en los sensores de la zona. | USUARIO |

Tabla 1 Funciones del sistema de parqueo de la PUJC

## Modelado

Para el modelado del sistema de parqueo de la PUJC se hizo uso del lenguaje SDL sobre la herramienta RTDS. El modelo del sistema cuenta con las funciones descritas en la tabla 1. La arquitectura del sistema se divide en dos grandes bloques y estos a su vez se dividen en m\’as bloques y procesos. La figura 3 detalla la arquitectura general del sistema de parqueo de la PUJC.

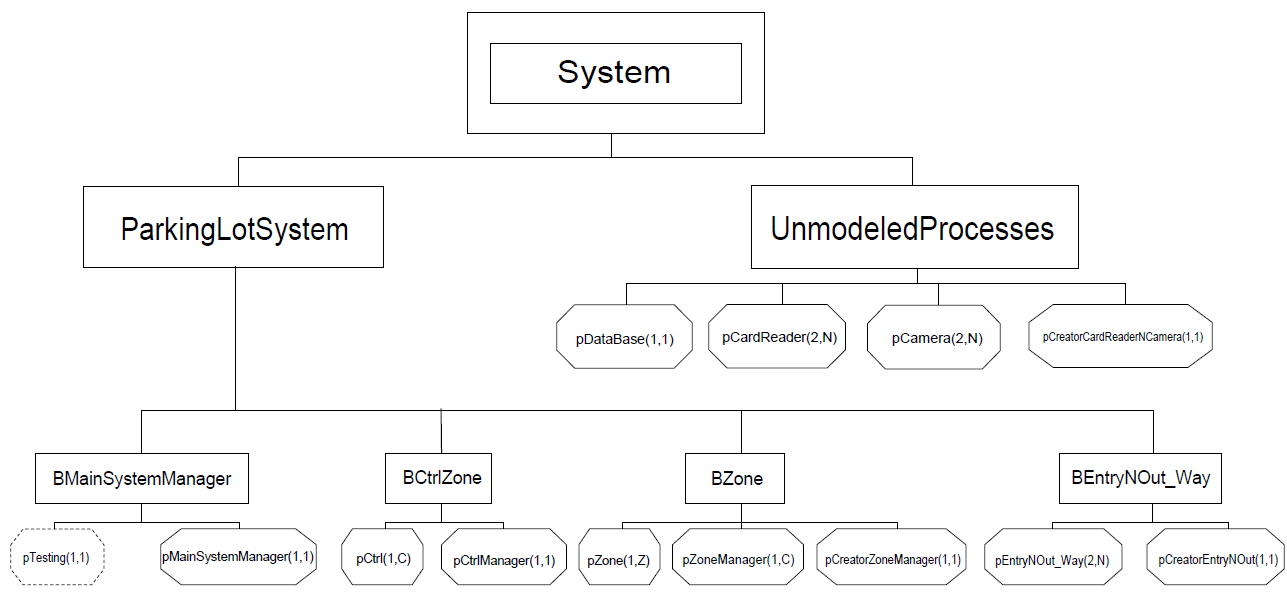


Figura 3. Arquitectura del modelo del sistema de parqueo de la PUJC

Como se aprecia en la figura 3, el sistema cuenta con dos bloques principales que se llaman: ParkingLotSystem y UnmodeledProcesses. El primer bloque cuenta con los procesos y elementos que se han modelado con un mayor nivel de detalle; por el contrario en el segundo bloque se encuentran los elementos del sistema que no se ha modelado sino que se han considerado como procesos los cuales reciben un valor y retornan otro pero sin haber un procesamiento espec\’ifico de la informaci\’on, dado que no es de inter\’es del proyecto modelar estos.

Las tablas 2 y 3 describen las funciones generales de cada bloque y proceso del sistema de parqueo respectivamente.

|  |  |
| --- | --- |
| **Nombre del bloque** | **Prop\’osito/Funci\’on** |
| ParkingLotSystem | Tiene como prop\’osito agrupar los bloques BMainSystemManager, BCtrlZone y BZone. Adicionalmente es el bloque que tiene m\’as interacci\’on con el Administrador. |
| UnmodeledProcesses | Tiene como prop\’osito agrupar los procesos no modelados que son: pDataBase, pCardReader, pCamera y pCreatorCardReaderNCamera. |
| BMainSystemManager | Agrupa los procesos pMainSystemManager y el proceso pTesting[[2]](#footnote-2), este bloque interactúa principalmente con el Administrador. |
| BCtrlZone | Agrupa los procesos pCtrl y pCtrlManager, interactua b\’asicamente con el bloque BMainSystemManager y el bloque BZone. |
| BZone | Agrupa los procesos pZone, pZoneManager, pCreatorZoneManager. Interactua principalmente con el bloque BCtrlZone y el usuario. |
| BEntryNOut\_Way | Agrupa los procesos pEntryNOut\_Way y pCreatorEntryNOut\_Way. Interactúa principalmente con los procesos del bloque UnmodeledProcesses y con el usuario. |

Tabla 2. Descripci\’on de los bloques del sistema de parqueo.

|  |  |
| --- | --- |
| **Nombre del proceso** | **Funciones** |
| pDataBase | Este proceso representa la base de datos del sistema, adem\’as est\’a encargado de evaluar si un usuario est\’a habilitado para ingresar o salir del sistema de parqueo. |
| pCardReader | Este proceso representa al lector de carn\’es y se encarga de entregar el c\’odigo del usuario que est\’a por ingresa o salir del sistema de parqueo. |
| pCamera | Este proceso representa la c\’amara que captura la placa del veh\’iculo y la entrega al sistema para evaluar si el usuario est\’a permitido para ingresar o salir del sistema. |
| pCreatorCardReaderNCamera | Proceso encargado de crear los procesos pCardReader y pCamera, y asignarlos a su respectivo proceso pEntryNOut\_Way. |
| pEntryNOut\_Way | Este proceso representa una entrada o salida principal, es el encargado de interactuar con el usuario. Adicionalmente es el único proceso que interactúa con los procesos pDataBase, pCardReader, pCamera y pCreatorCardReaderNCamera. |
| pCreatorEntryNOut\_Way | Proceso encargado de crear el proceso pEntryNOut\_Way. |
| PZone | Este proceso representa una zona del sistema e interactúa con el controlador de zonas reportando el ingreso o salida de un veh\’iculo, dependiendo de la secuencia de se\~nales del conjunto de sensores que se reciba. |
| pZoneManager | Proceso encargado de crear instancias del proceso pZone y a su vez asignarle su respectivo proceso pCtrl. |
| pCreatorZoneManager | Proceso encargado de crear los procesos pZoneManager. |
| PCtrl | Este proceso representa un controlador de zonas y est\’a encargado de controlar los procesos pZone, adicionalmente reporta el estado de las plazas libres y totales que posee cada zona. |
| pCtrlManager | Proceso encargado de crear los procesos pCtrl. |
| pMainSystemManager | Este proceso representa la interfaz directa entre el administrador y el sistema, a trav\’es de este proceso se puede crear entradas y salidas principales, controladores de zona, zonas, hacer requerimientos de informaci\on, inicializar tanto plazas libres como plazas totales. |
| PTesting | Proceso que permite acoplar el sistema para hacer pruebas funcionales de caja negra. Este proceso no debe ser considerado en la fase de implementaci\’on del sistema de parqueo. |

Tabla 3. Descripci\’on los procesos del sistema de parqueo.

Dentro de la arquitectura del sistema de parqueo presentada en la figura 3 se detallan algunos procesos que tienen como funci\’on crear instancias como se aprecia en la tabla 3. Dado que la arquitectura del sistema de parqueo se compone de bloques no es posible crear instancias de procesos si \’estos no se encuentran en el mismo nivel, lo anterior justifica la existencia de dichos procesos como *puentes* para crear instancias de procesos que se encuentren en su mismo nivel.

### Especificaci\’on del sistema de parqueo

El modelado del sistema de parqueo empieza desde los requerimientos de especificaci\’on del sistema, en este caso se desea que \’este posea las funciones descritas en la tabla 1. La herramienta RTDS tiene la posibilidad de anexar al proyecto diagramas MSC, \’estos son fundamentales para describir las especificaciones del sistema. Existen estrategias para el dise\~no y desarrollo de modelos, entre las cuales encontramos Top-Down y Bottom-Up. Top-Down es una estrategia donde se empieza desarrollando el modelo desde un alto nivel con una m\’inima abstracci\’on y termina en bloques o procesos con altos niveles de abstracci\’on. Bottom-Up es una estrategia que consiste en empezar a dise\~nar los elementos del sistema que tiene un alto nivel de abstracci\’on y a partir de \’estos se crean bloques m\’as complejos. En el modelado del sistema de parqueo de la PUJC se hace uso de la estrategia Top-Down para la especificaci\’on de requerimientos del sistema que se encuentran descritos en MSC, y para la implementaci\’on del sistema se hace uso de la metodolog\’ia Botton-Up empleando SDL.

#### MSCs de las especificaciones del sistema

En la figura 3 detalla que el sistema de parqueo posee dos grandes bloques y a su vez estos contienen sub-bloques y procesos. Dado que la practicidad de especificar el sistema en alto nivel, sin tener en cuenta el nivel de abstracci\’on, es mucho m\’as sencillo que hacer el proceso contrario, se elabora un MSC que tiene como finalidad mostrar un escenario en el cual se describa la interacci\’on de los dos bloques principales: ParkingLotSystem y UnmodeledProcesses. La figura 4 muestra la interacci\’on entre los dos bloques ParkingLotSystem y UnmodeledProcesses, el escenario que describe es el acceso de un veh\’iculo al sistema de parqueo. En este MSC encontramos tres agentes interactuando entre s\’i los cuales son: Env, ParkingLotSystem y UnmodeledProcesses. El agente Env representa el ambiente o medio por el cual el sistema intercambia mensajes con el usuario o con los sensores. El procedimiento que describe el acceso de un veh\’iculo al sistema de parqueo de la PUJC est\’a dado por la siguiente secuencia:

1. El proceso Env env\’ia una se\~nal al bloque ParkingLotSystem llamada sLoopInductive\_Entrance que indica que un veh\’iculo est\’a en la porter\’ia principal.
2. El bloque ParkingLotSystem necesita verificar que el usuario que desea ingresar al sistema de parqueo est\’a autorizado. Para ello, necesita saber su identificaci\’on y la placa del veh\’iculo. Con la se\~nal sEnableCarReader habilita el lector de carn\’es.
3. Una vez que el bloque ParkingLotSystem tenga el c\’odigo del usuario, habilita la toma de la foto a la placa, para ello intercambia el mensaje sRequestPlate con el bloque UnmodeledProcesses.
4. El bloque UnmodeledProcesses env\’ia la placa como un charstring al bloque que hizo la consulta.
5. Una vez que se tenga la informaci\’on del usuario es indispensable validarla y determinar si se le otorga el acceso o no al sistema. Para ello env\’ia un requerimiento al bloque UnmodeledProcesses esperando la se\~nal sOkUser que confirma que el usuario est\’a habilitado para usar el sistema de parqueo o sNoRegis\_User que indica que no puede usar el sistema de parqueo.
6. Si el usuario est\’a habilitado para usar el sistema de parqueo entonces se efectúa una serie de pasos para darle bienvenida al usuario. Para ello se env\’ia una se\~nal a la talanquera para que permita el paso del veh\’iculo, una vez que \’este haya pasado, un sensor env\’ia dicha se\~nal indicando que es seguro bajar la talanquera y el bloque ParkingLotSystem culmina enviando la se\~nal sDownBarrier, quedando as\’i disponibles los dos bloques para interactuar nuevamente.

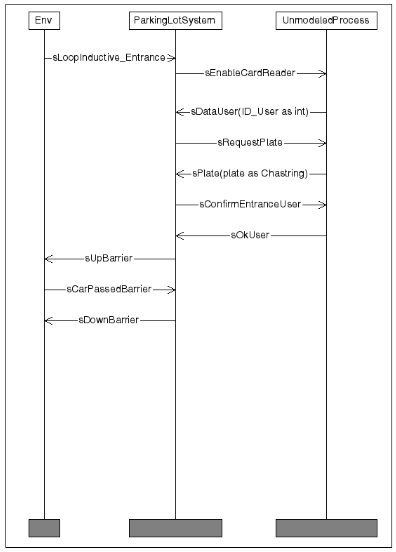


Figura 4. MSC ingreso veh\’iculo al sistema de parqueo de la PUJC

El escenario que representa la salida de un veh\’iculo del sistema de parqueo de la PUJC es similar al comportamiento descrito en la figura 4, la diferencia radica que el agente Env o entorno env\’ia una se\~nal sLoopInductive\_Exit y en el momento que el bloque ParkingLotSystem desea verificar si el usuario est\’a autorizado para salir, env\’ia una se\~nal sConfirmOutUser.

Los MSCs son útiles para describir la interacci\’on entre agentes, que si bien pueden ser bloques tambi\’en sirven para describir el comportamiento a nivel de procesos. Como se muestran en las figuras 5 y 6 que representan los MSCs de la inicializaci\’on del sistema cuando \’este empieza por primera vez. Los requerimientos del sistema consisten en lo siguiente:

* El sistema debe de tener por lo menos una entrada y una salida principal.
* Cada entrada y salida principal debe de tener su conjunto de sensores para gestionar el acceso o salida del usuario del sistema, una c\’amara la cual retorna el valor de la placa del veh\’iculo y un lector de carn\’es.
* Todo el sistema tiene por lo menos una zona y su respectivo controlador. La zona cuenta con sus respectivos sensores para indicar si un veh\’iculo ha ingresado o salido de dicha zona.
* Cuenta con un repositorio de datos donde se encuentran los c\’odigos y sus placas asociadas que est\’an autorizadas para el acceso o salida del sistema de parqueo de la PUJC.

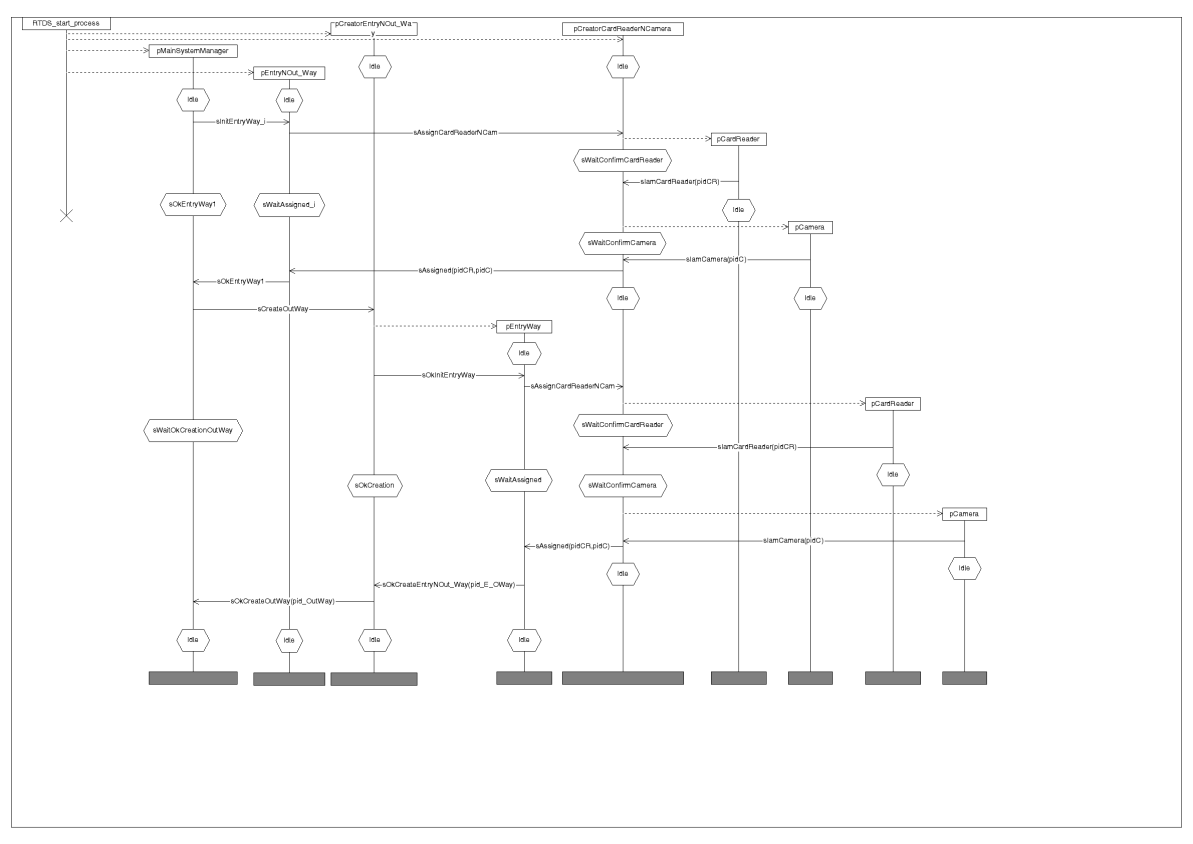


Figura 5. Inicializaci\’on entrada y salida principal del sistema

La figura 5, detalla la interacci\’on entre procesos del bloque ParkingLotSystem y los procesos del bloque UnmolededProcesses. Inicialmente el sistema contiene una instancia de cada uno de los proceso de la figura 3. El proceso pMainSystemManager se encarga de asociar el proceso pEntryNOut\_Way a su tabla de entradas principales; para ello env\’ia una se\~nal llamada sInitEntryWay\_i al proceso pEntryNOutWay. A partir de esta solicitud lo que va ha pasar es que \’este proceso necesita que se le sea asigando un lector de carn\’es, pCardReader, y una c\’amara, pCamera. El proceso que tiene la facultad de instanciar estos procesos es pCreatorCardReaderNCamera, una vez \’este proceso haya creado el lector de carn\’es y la c\’amara, enviar\’a una se\~nal al proceso pEntryNOut\_Way con los identificadores correspondientes. El proceso pMainSystemManager cuando reciba la se\~nal sOkEntryWay1 por parte de pEntryNOut\_Way indicar\’a que la primera entrada principal ha sido exitosamente creada, entonces se procede a crear una salida principal. La interacci\’on de los procesos es muy parecida a la inicializaci\’on de la primera entrada principal, solo cambia que el proceso pMainSystemManager solicita al proceso pCreatorCardReaderNCamera instanciar un proceso pEntryNOut\_Way y env\’ie la se\~nal correspondiente a este último para que se le sea asociado un lector de carn\’es y una c\’amara. El proceso de inicializaci\’on termina cuando el proceso de creaci\’on de entradas y salidas principales env\’ia un mensaje al proceso pMainSystemManager llamada sOkCreateOutWay asociando el identificador de dicha salida para que sea anexado en su lista de salidas principales.

La figura 6 describe la inicializaci\’on de los procesos del bloque ParkingLotSystem, el objetivo de esta interacci\’on es asociar a la zona y al proceso pZoneManager su respectivo controlador. El sistema de parqueo cuenta con controladores de tienen a su mando un conjunto de zonas; dado que el sistema es din\’amico es posible que un controlador pueda tener m\’as zonas, ya que el proceso pCtrl no se encuentra en el mismo bloque que pZone se necesita un proceso que sea capaz de crear m\’as zonas, para ello se requiere del proceso pZoneManager. Lo anterior implica que cada proceso pCtrl tendr\’a asociando hasta *Z* zonas y un proceso pZoneManager.

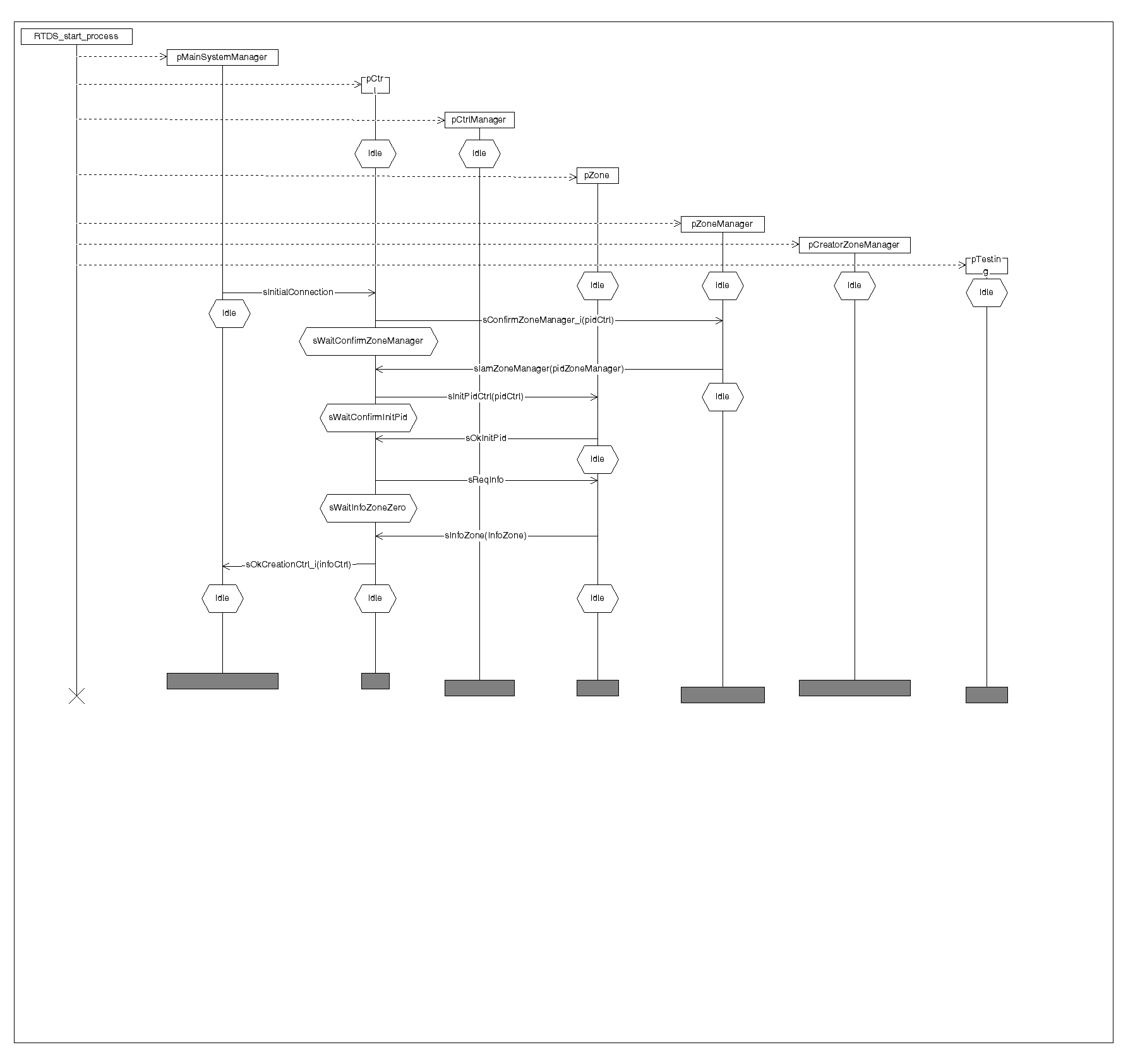


Figura 6. Inicializaci\’on de procesos del bloque ParkingLotSystem

La funci\’on de cada uno de las se\~nales presentadas a partir de este momento se puede detallar en el Anexo B.

La tabla 4 explica el prop\’osito de cada uno de los MSCs que se dise\~naron para la descripci\’on de las especificaciones del sistema de parqueo de la PUJC.

|  |  |
| --- | --- |
| **MSC** | **Descripci\’on** |
| Figura 7. Creaci\’on de una entrada principal al sistema de parqueo. | El administrador solicita la creaci\’on de una entrada principal al sistema de parqueo. B\’asicamente se crea el proceso que representa una entrada principal y a \’este se le asigna su respectiva c\’amara y lector de carn\’e. El sistema si ha creado exitosamente la entrada principal, retorna una se\~nal al administrador llamada sOkCreateE\_W. Si el sistema tiene el cupo m\’aximo de entradas principales retornar\’a al administrador la se\~nal sExcEntryWay en lúgar de sOkCreateE\_W y no se efectuar\’a el intercambio de se\~nales para la asignaci\’on de c\’amara y lector de carn\’es. |
| Figura 8. Creaci\’on de una salida principal al sistema de parqueo | El administrador solicita la creaci\’on de una entrada principal al sistema de parqueo. B\’asicamente se crea el proceso que representa una entrada principal y a \’este se le asigna su respectiva c\’amara y lector de carn\’e. El sistema si ha creado exitosamente la entrada principal, retorna una se\~nal al administrador llamada sOkCreateE\_O. Si el sistema tiene el cupo m\’aximo de entradas principales retornar\’a al administrador la se\~nal sExcOutWay en lúgar de sOkCreateE\_O y no se efectuar\’a el intercambio de se\~nales para la asignaci\’on de c\’amara y lector de carn\’es. |
| Figura 9. Creaci\’on de una zona de parqueo cuando el controlador de zonas tiene asociado un pZoneManager | El administrador del sistema env\’ia una se\~nal al proceso pMainSystemManager para crear una zona en un controlador espec\’ifico, con una inicializaci\’on de plazas libres y totales de la zona. Para la solicitud de la zona se usa la se\~nal sAddZone donde su primer par\’ametro es el número del controlador de zona, el suegundo y tercero son las plazas totales y libres de la zona respectivamente. Si la creaci\’on de la zona es exitosa el proceso pMainSystemManager lo har\’a saber al administrador a trav\’es de la se\~nal sOkCreateZone de lo contrario enviar\’a la se\~nal sExcLimitZones, \’esta se\~nal se da antes que el proceso pZoneManager haga una instancia del proceso pZone. |
| Figura 10. Creaci\’on de una zona de parqueo cuando el controlador de zonas no tiene asociado un pZoneManager | El administrador del sistema env\’ia una se\~nal al proceso pMainSystemManager para crear una zona en un controlador espec\’ifico. Este escenario es muy parecido al de la figura 9 la diferencia es que el controlador de zonas no tiene un proceso pZoneManager asociado, por lo cual solicita su creaci\’on, una vez creado el pZoneManager y asociado a su respectivo controlador de zonas el intercambio de se\~nales respecto a la figura 9 son los mismos. |
| Figura 11. MSC ajustes de par\’ametros a una zona de parqueo reci\’en creada. | Este MSC representa el intercambio de se\~nales para la inicializaci\’on de una zona de parqueo, en este diagrama a la zona se le asigna un controlador y se ajusta su capacidad de aparcamiento de veh\’iculos. |
| Figura 12. MSC creaci\’on de un controlador de zonas al sistema de parqueo | El administrador solicita al sistema de parqueo crear un controlador de zonas por medio de la se\~nal sCreateCtrl que recibe el proceso pMainSystemManager. Si la creaci\’on es exitosa el proceso pMainSystemManager retornar\’a una se\~nal sOkCreateCtrl al administrador, de lo contrario, har\’a saber al administrador que no es posible tener m\’as controladores de zonas con la se\~nal sExcLimitCtrl. |
| Figura 13. MSC ingreso de un veh\’iculo al sistema de parqueo. | Este escenario es similar al de la figura 4 presentada anteriormente pero a nivel de procesos. Dado que los procesos pCardReader, pCamera y pDataBase no han sido modelados, se ha colocado un temporizar en la vida del proceso pCamera que se llama timerProcessOCR el cual representa el tiempo de c\’omputo que tardar\’ia el proceso pCamera para entregar a partir de una foto la placa de un veh\’iculo en un tipo de dato charstring. |
| Figura 14. MSC Salida de un veh\’iculo del sistema de parqueo. | Este escenario es similar al de la figura 13. La dieferencia es que en este escenario el veh\’oculo est\’a por salir del sistema de parqueo, b\’asicamente el proceso de verificaci\’on de usuario habilitado para salir es el mismo que cuando el suaurio est\’a por ingresar con la diferencia que \’esta validaci\’on se hace a trav\’es de la se\~nal sConfirmOutUser. |
| Figura 15. MSC usuario no autorizado intentando ingresar al sistema de parqueo de la | Este escenario representa el caso cuando un usuario desea ingresar al sistema de parqueo de la PUJC pero \’este no est\’a habilitado para hacerlo. La din\’amica de verificaci\’on del usuario es la misma que la figura 13, solo que la respuesta por parte del proceso pDataBase es negativa y no se le da acceso al usuario de ingresar. |
| Figura 16. MSC usuario no autorizado intentando salir del sistema de parqueo de la PUJC. | Este escenario representa el caso cuando un usuario desea salir del sistema de parqueo de la PUJC pero \’este no est\’a habilitado para hacerlo. La din\’amica de verificaci\’on del usuario es la misma que la figura 14, solo que la respuesta por parte del proceso pDataBase es negativa y no se le da acceso al usuario de salir. |
| Figura 17. MSC usuario ingresando a una zona del sistema de parqueo. | Este MSC representa el escenario el cual un veh\’iculo va a ingresar a una zona del sistema. Para saber que un usario ha ingresado se tiene que cumplir la siguiente secuencia: Se interrumpe el primero sensor infrarrojo, sIR1\_Zone, posteriormente el veh\’oculo interrumpe el sensor infrarrojo 2, sIR2\_Zone y finalmente se verifica que es un vehiculo por la se\~nal capturada en el sensor inductivo sLoopInductive\_Zone. La zona reportar\’a que ha ingresado un veh\’iculo a su respectivo controlador con la se\~nal sEntered\_Car. Otra posibilidad que es v\’alida para el ingreso del vehiculo es cuando se interrumpe el sensor infrarrojo 4, luego el sensor infrarrojo 3 y finalmente la recepci\’on de la se\~nal del loop inductivo. |
| Figura 18. MSC usuario saliendo de una zona del sistema de parqueo. | Este MSC representa el escenario el cual un veh\’iculo va a salir de una zona del sistema. Para saber que un usario ha ingresado se tiene que cumplir la siguiente secuencia: Se interrumpe el primero sensor infrarrojo 3, sIR3\_Zone, posteriormente el veh\’iulo interrumpe el sensor infrarrojo 4, sIR4\_Zone y finalmente se verifica que es un vehiculo por la se\~nal capturada en el sensor inductivo sLoopInductive\_Zone. La zona reportar\’a que ha salido un veh\’iculo de su zona a su respectivo controlador con la se\~nal sOut\_Car. Otra posibilidad que es v\’alida para reconocer que un vehiculo va a salir de la zona es cuando se interrumpe el sensor infrarrojo 2, luego el sensor infrarrojo 1 y finalmente la recepci\’on de la se\~nal del loop inductivo. |
| Figura 19. MSC Ingreso y salida de un veh\’iculo a trav\’es del proceso Testing | Este MSC representa la entrada y salida de veh\’iculos en una zona del sistema, imagen lado izquierod y derecho respectivamente. La diferencia de este escenario a los de la figura 17 y 18 es que las se\~nales de los sensores son simuladas desde el proceso pMainSystemManager. El controlador y la zona que se quiere entrar y sacar veh\’iculos se da a trav\’es del proceso temporal pTesting. Cabe destacar que las se\~nales de sensores generadas por el proceso pMainSystemManager no deben ser considerados en la implementaci\’on del sistema de parqueo. |

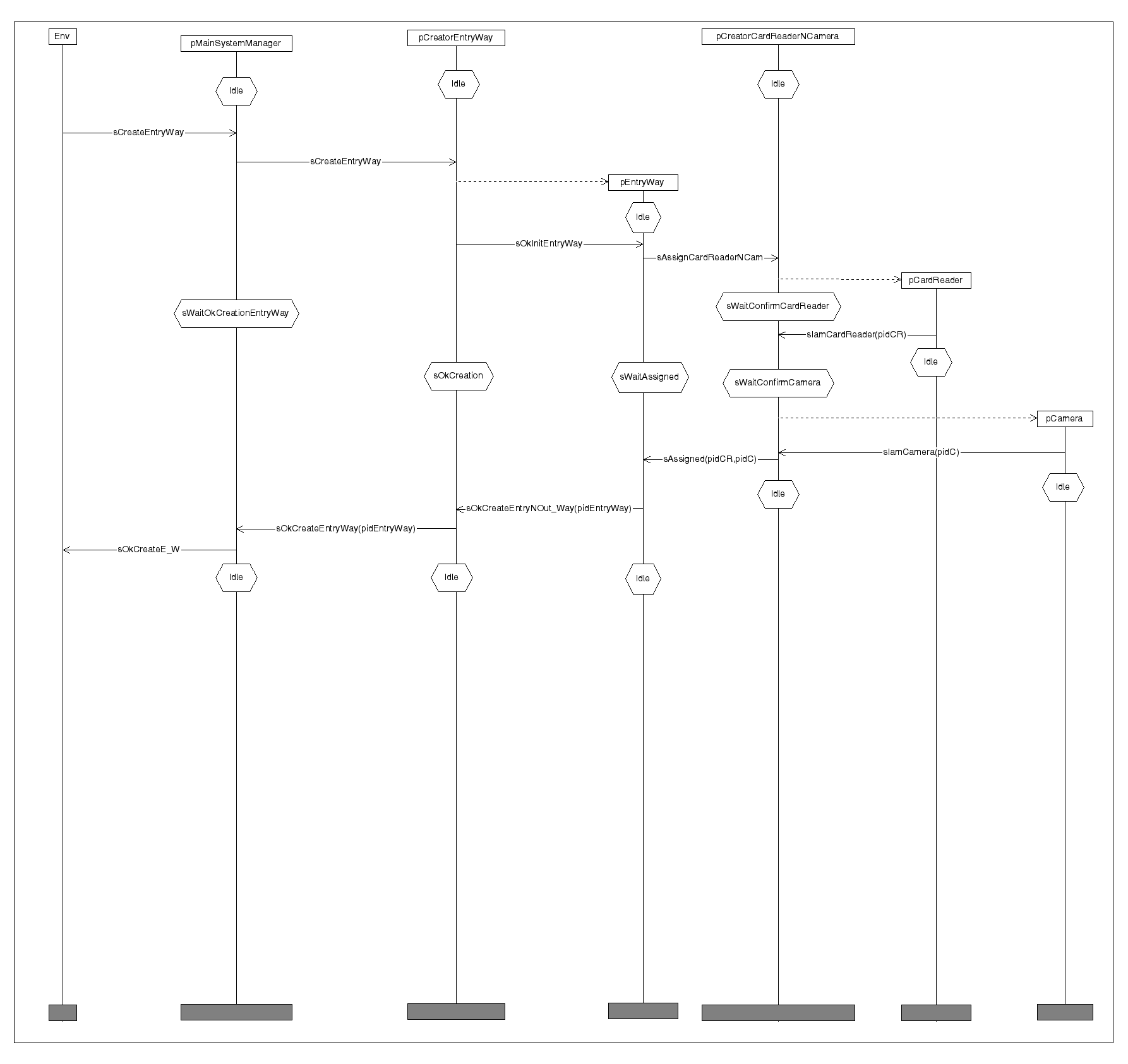


Figura 7. Creaci\’on de una entrada principal al sistema de parqueo.

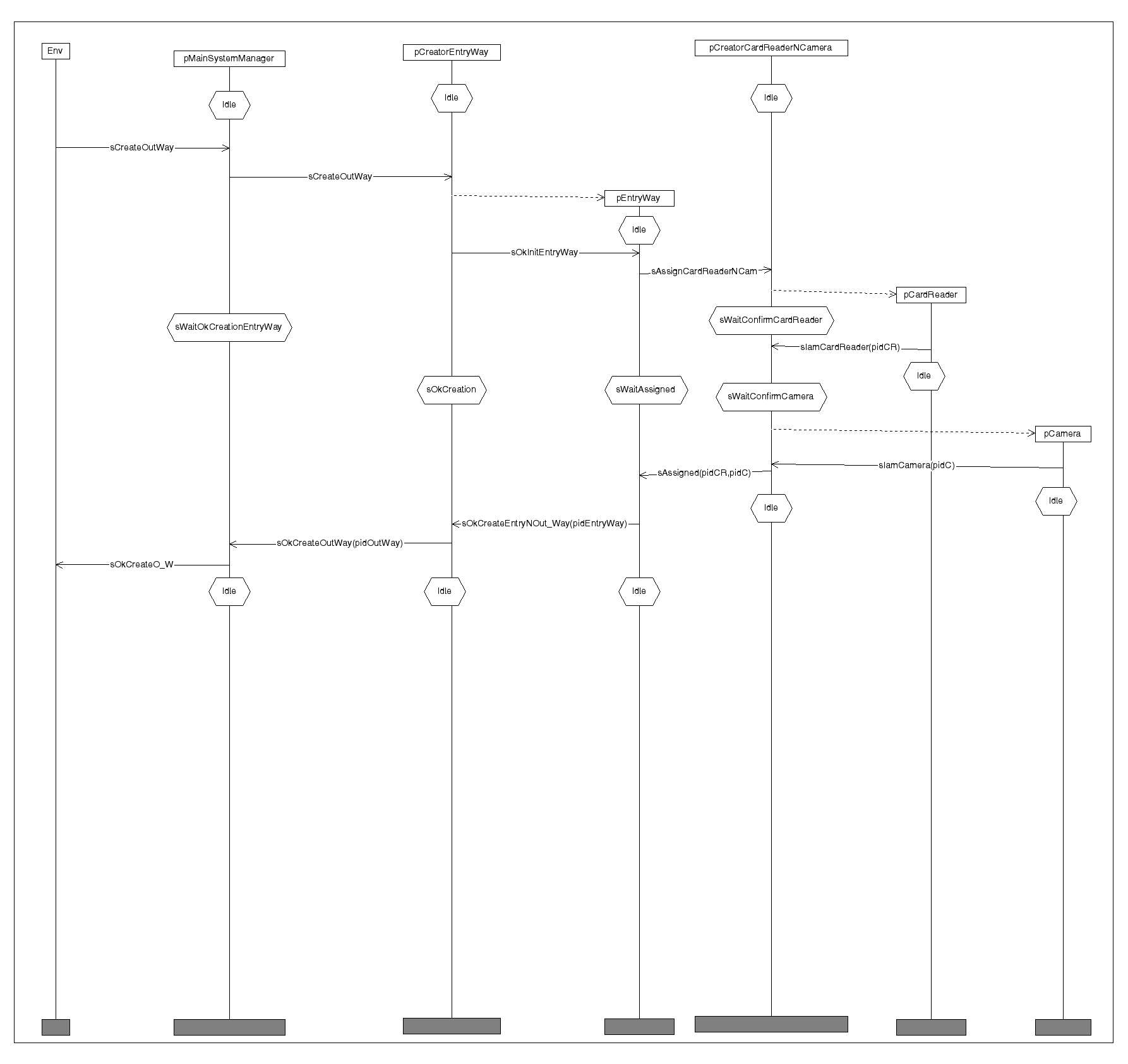


Figura 8. Creaci\’on de una salida principal al sistema de parqueo

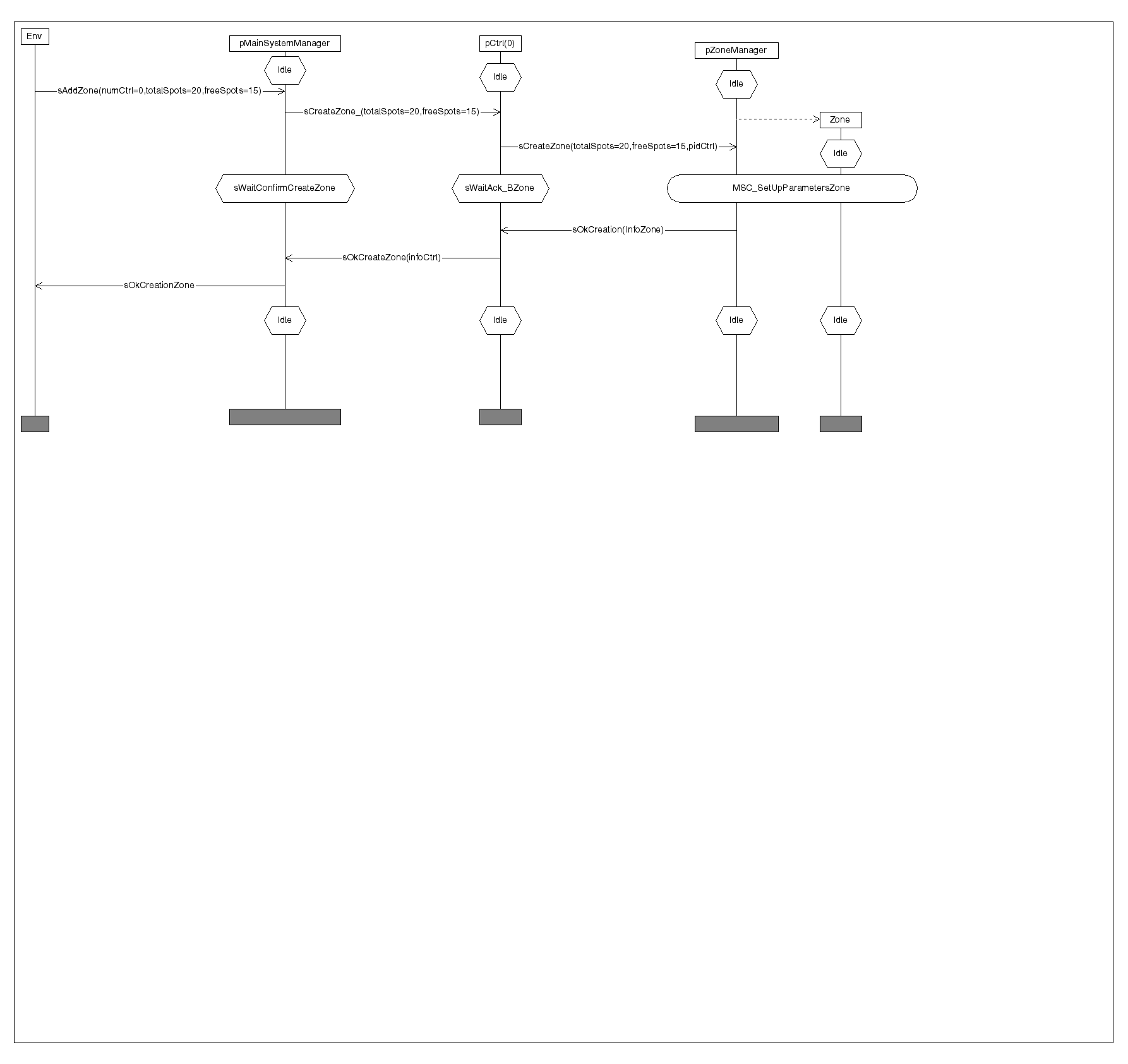


Figura 9. Creaci\’on de una zona de parqueo cuando el controlador de zonas tiene asociado un pZoneManager

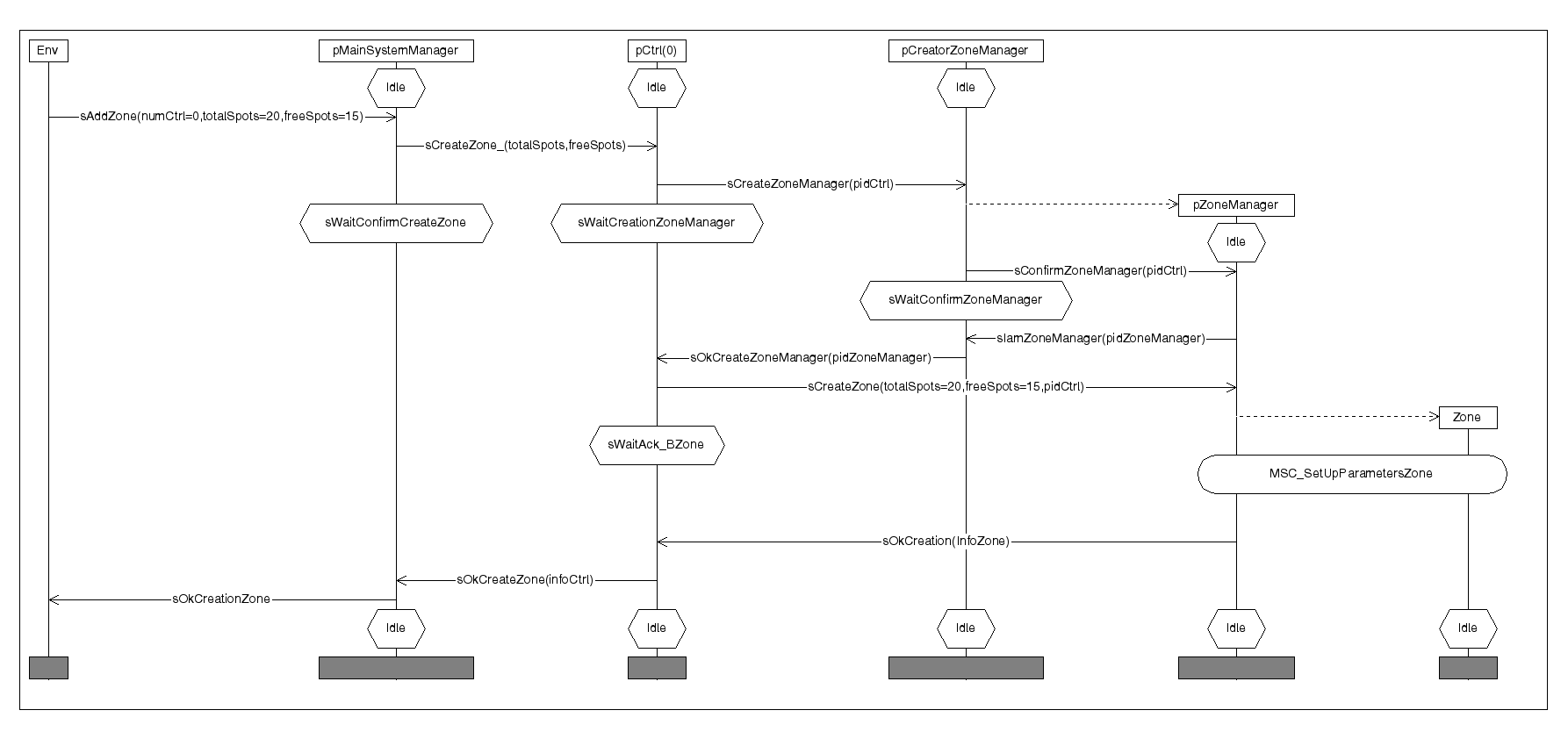


Figura 10. Creaci\’on de una zona de parqueo cuando el controlador de zonas no tiene asociado un pZoneManager

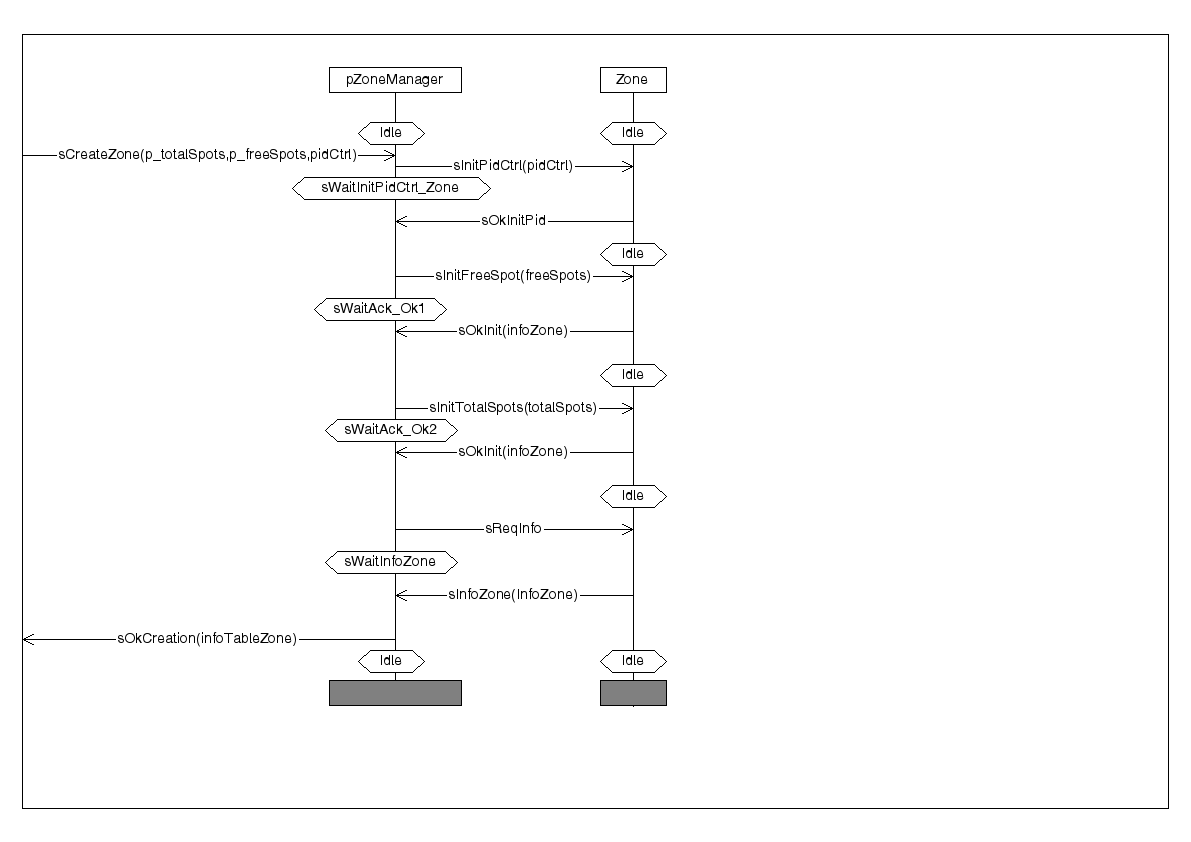


Figura 11. MSC ajustes de par\’ametros a una zona de parqueo reci\’en creada.

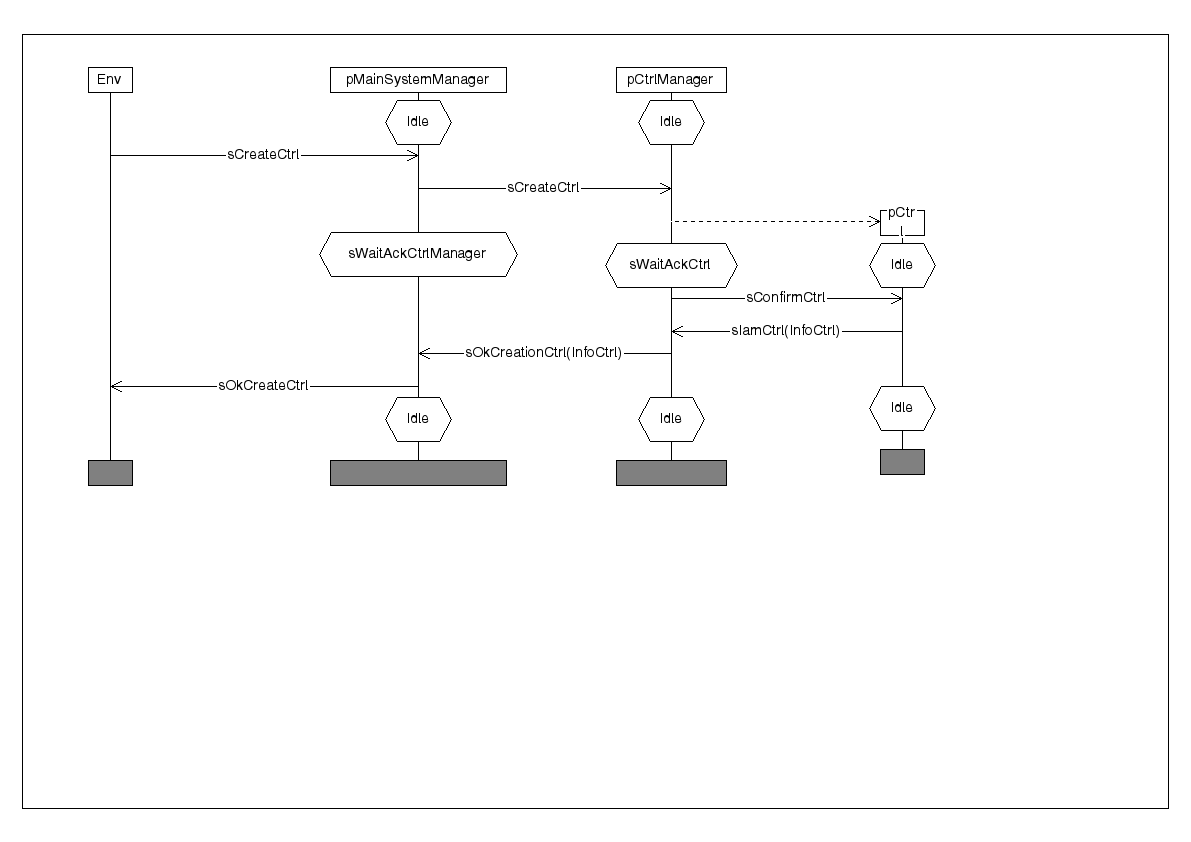


Figura 12. MSC creaci\’on de un controlador de zonas al sistema de parqueo.

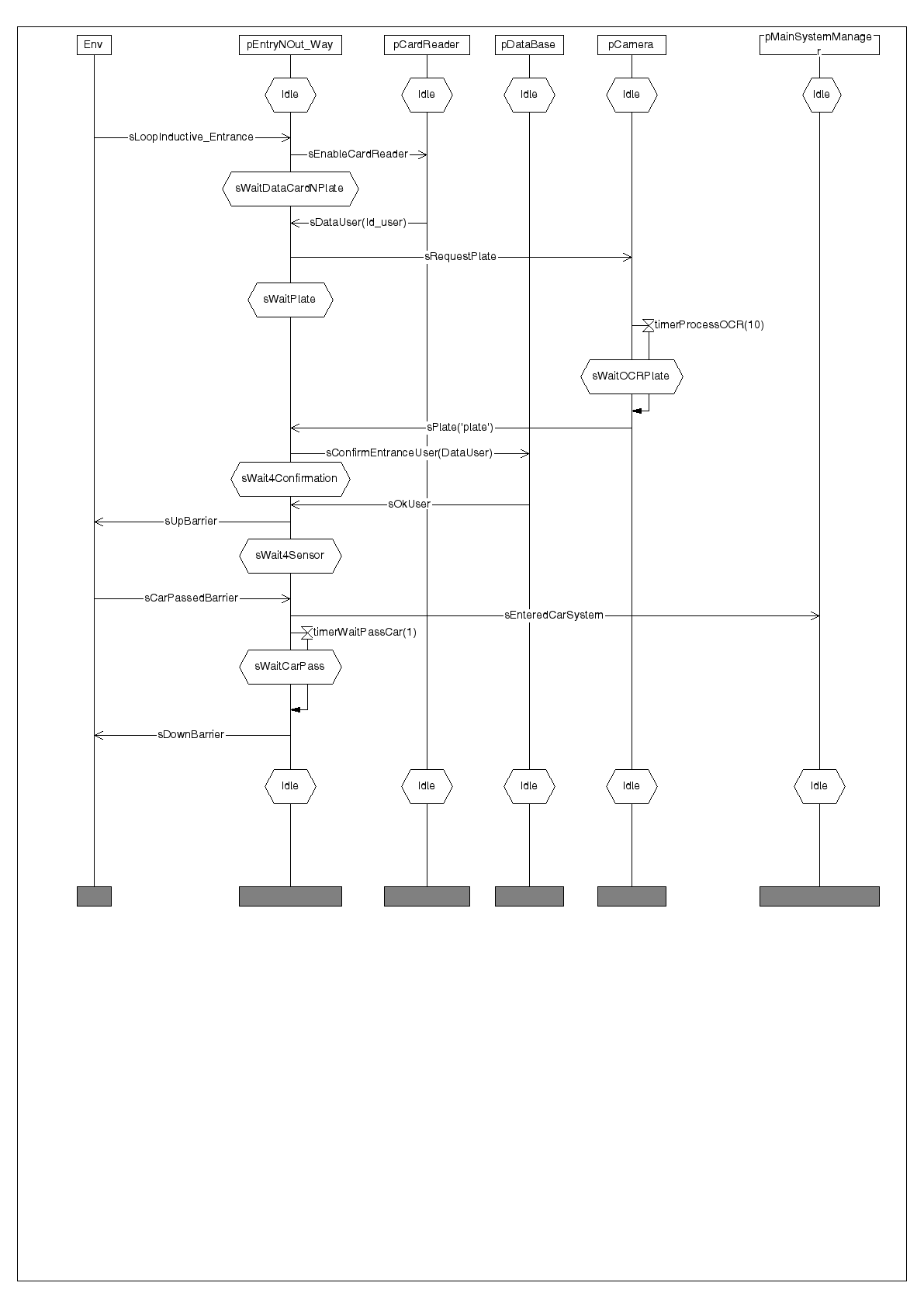


Figura 13. MSC ingreso de un veh\’iculo al sistema de parqueo.

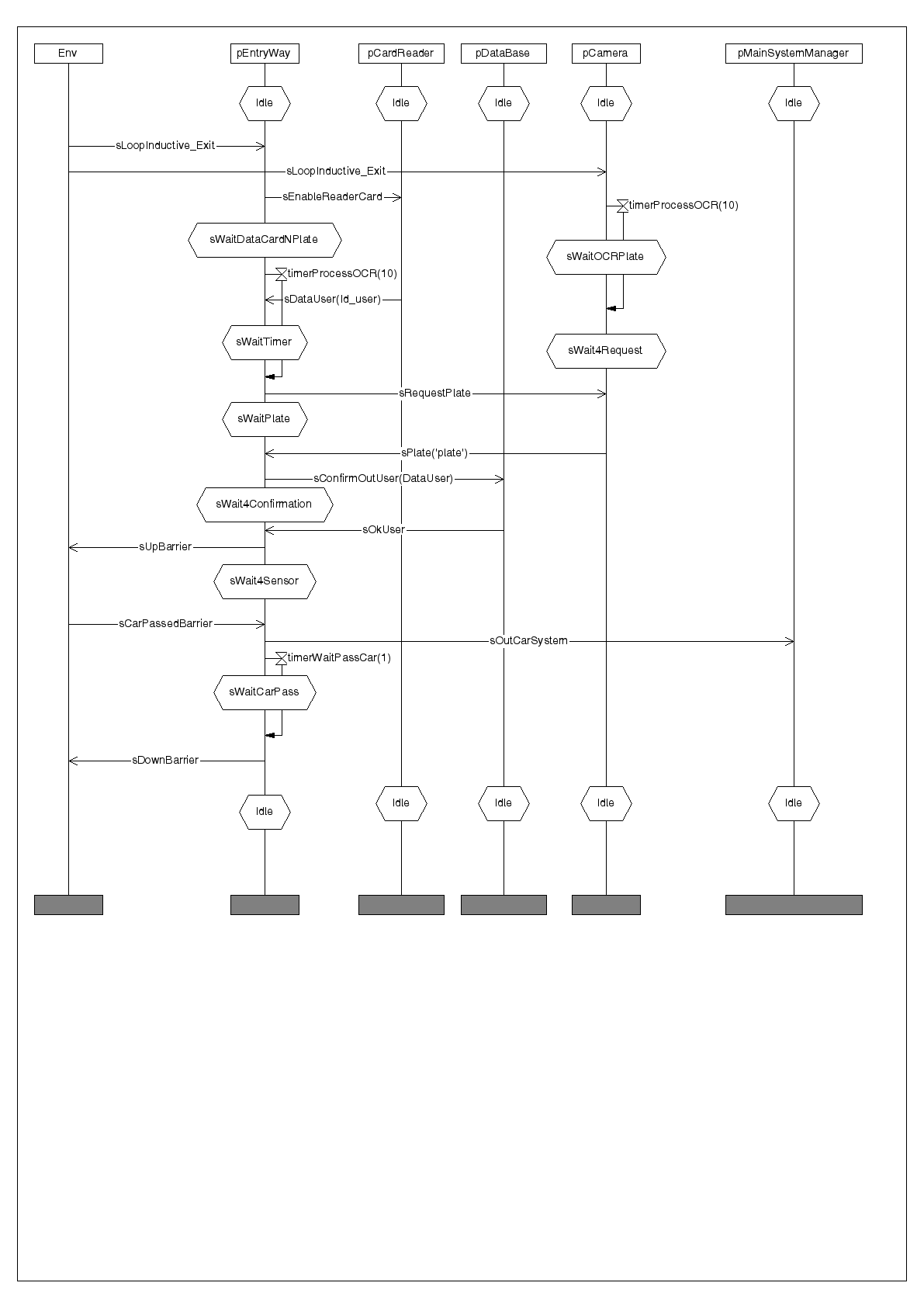


Figura 14. MSC Salida de un vehiculo del sistema de parqueo.

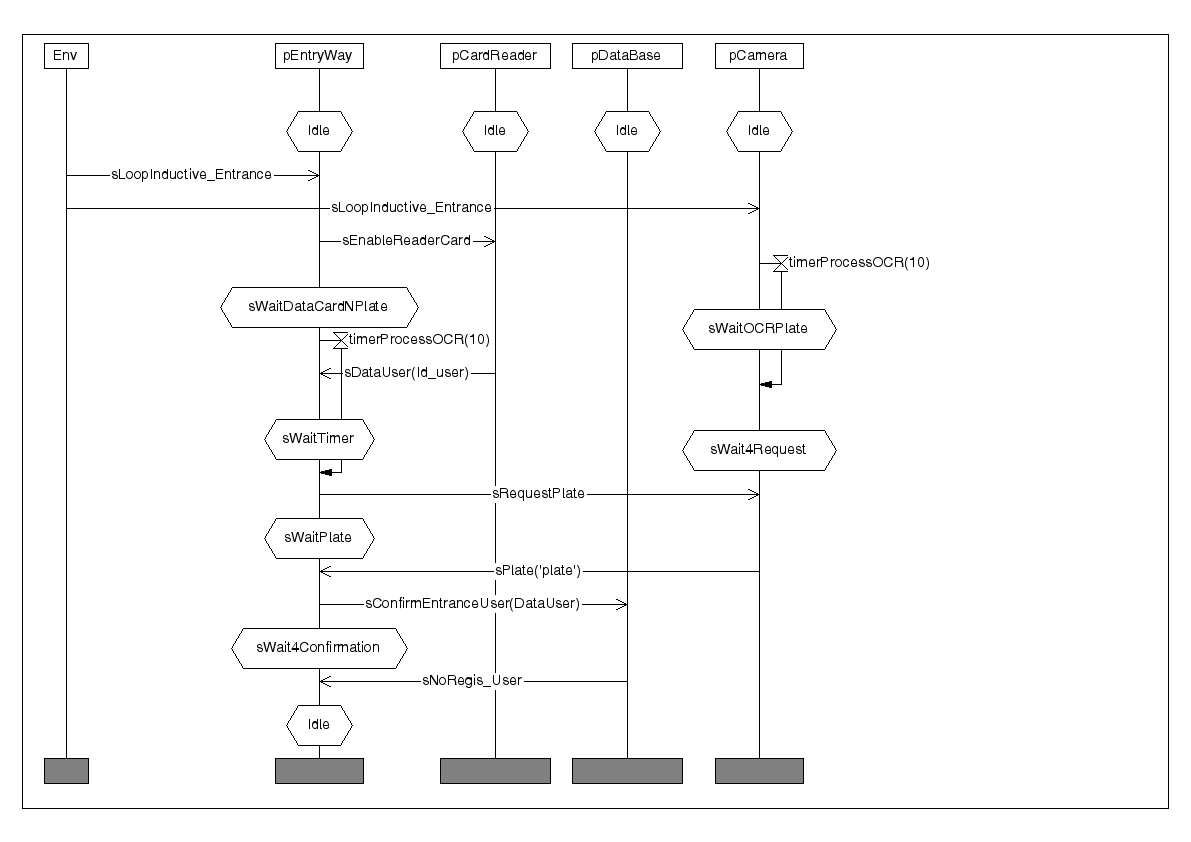


Figura 15. MSC usuario no autorizado intentando ingresar al sistema de parqueo de la PUJC.

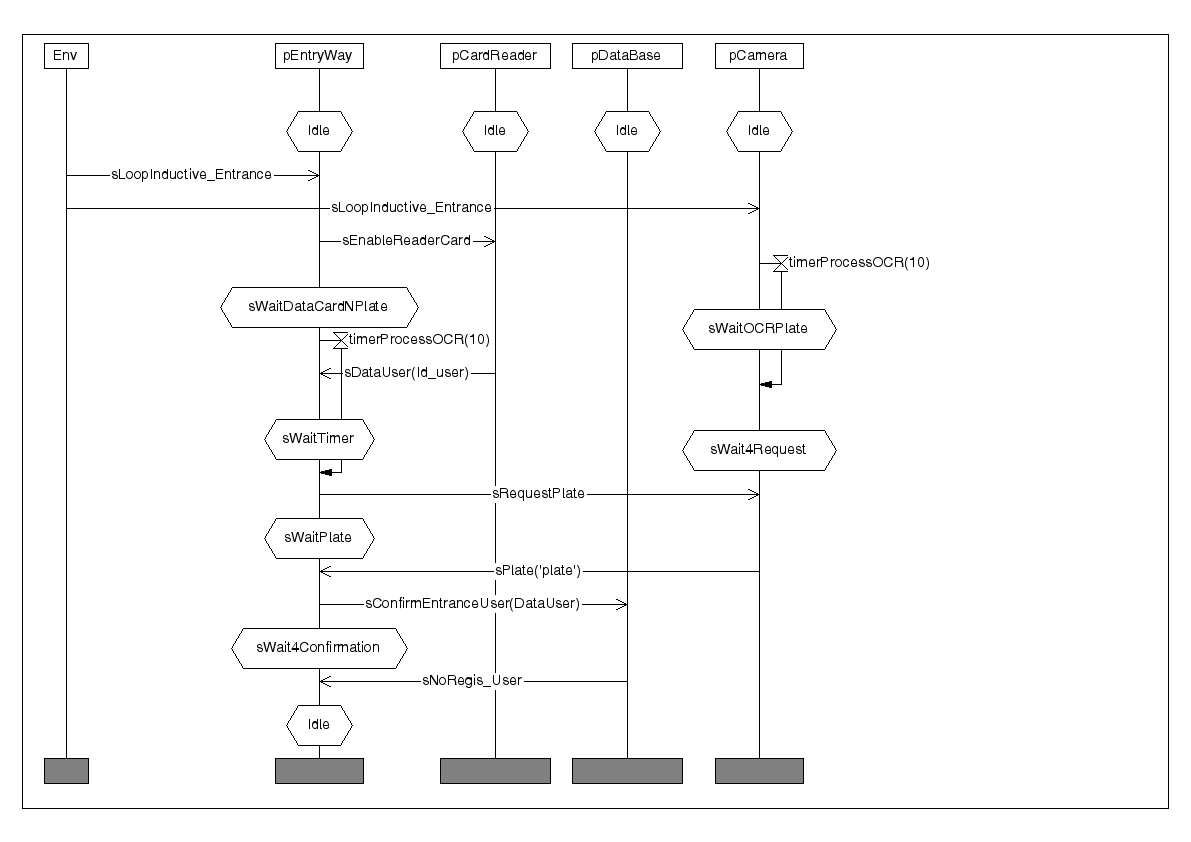


Figura 16. MSC usuario no autorizado intentando salir del sistema de parqueo de la PUJC.

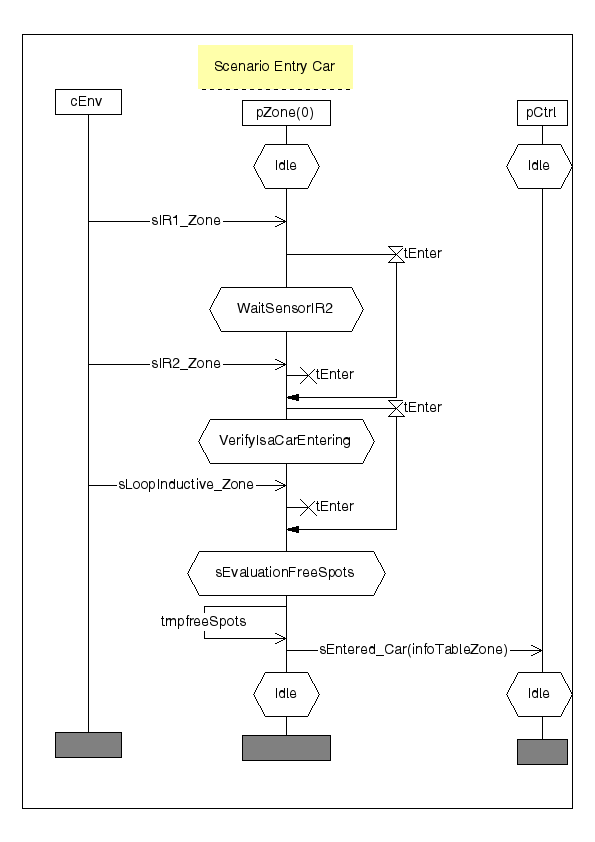


Figura 17. MSC usuario ingresando a una zona del sistema de parqueo.

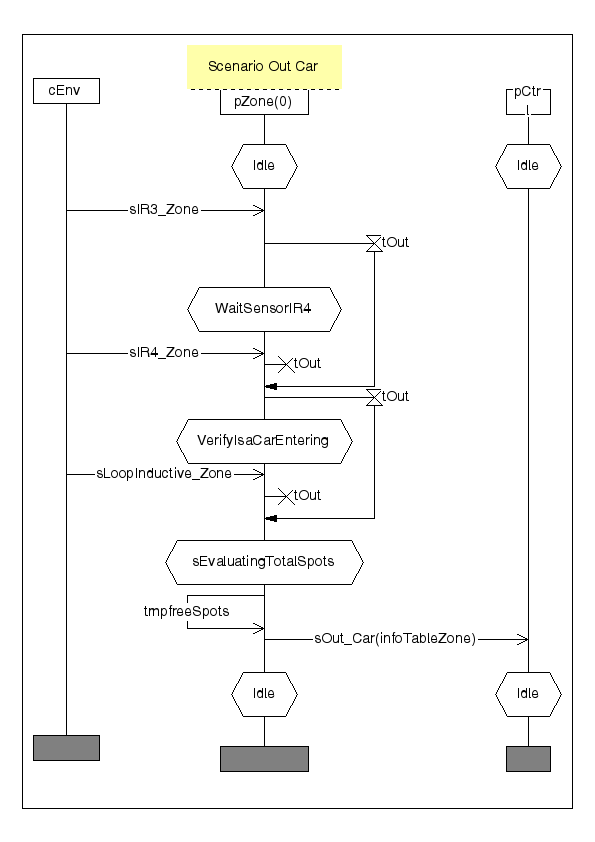


Figura 18. MSC usuario saliendo de una zona del sistema de parqueo.

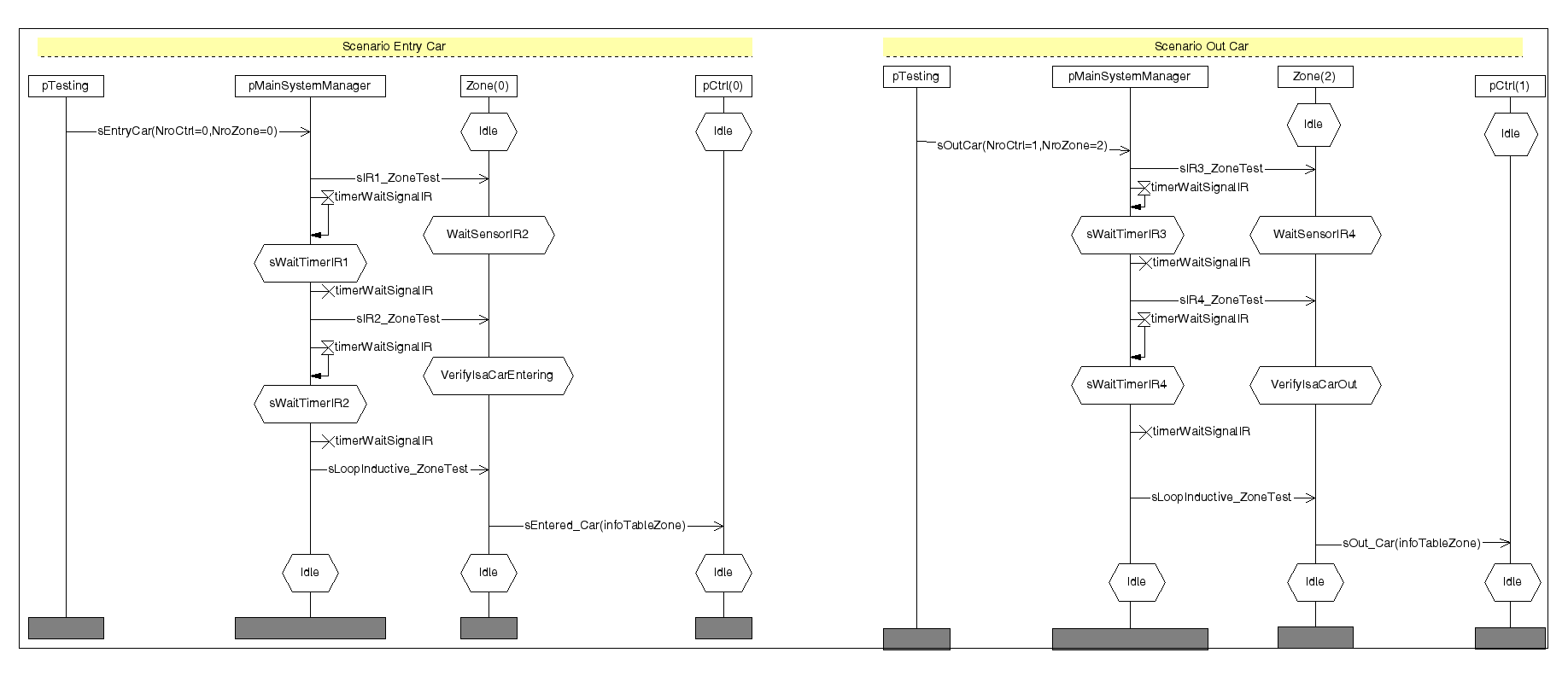


Figura 19. MSC Ingreso y salida de un veh\’iculo a trav\’es del proceso Testing

1. Una zona es un área donde se pueden aparcar vehículos y cuenta con un sistema de sensores para determinar cuando entran o salen los vehículos. [↑](#footnote-ref-1)
2. El proceso pTesting solo tiene sentido en la fase de pruebas del modelo, pero no debe ser considerado en la fase de implementación. [↑](#footnote-ref-2)