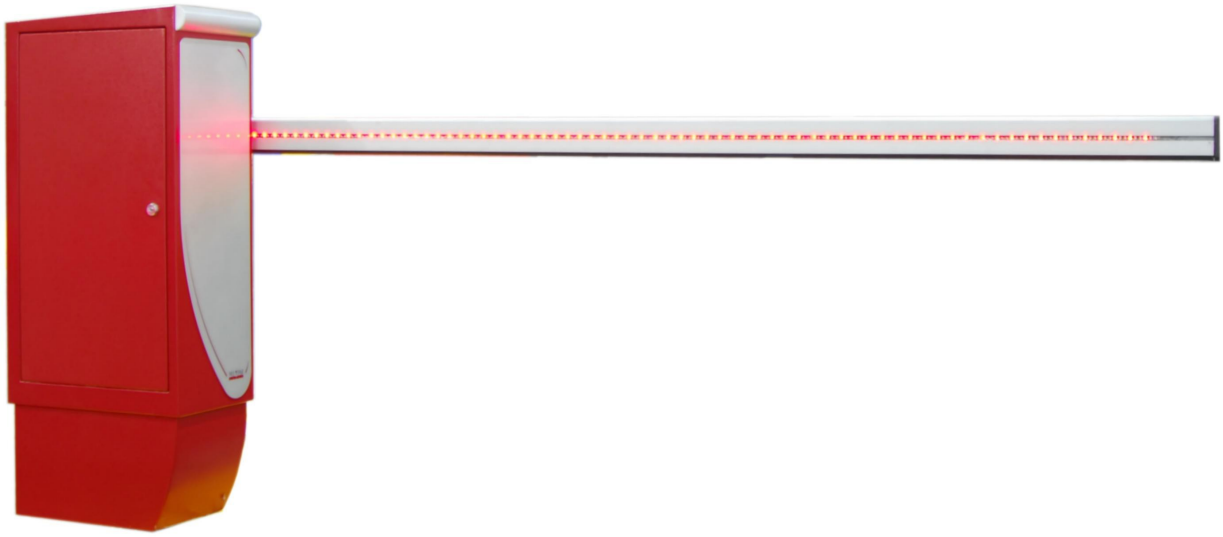
Proyecto Servicios Web

Sistema de Control de Acceso a un Párking



Sistemas Distribuidos y Servicios Web

Ana Lucero Fernández
Carlos Rodríguez Hernández

Índice

Índice de ilustraciones

1.- Motivación

En este segundo trabajo de la asignatura se pide realizar un proyecto basado en Servicios Web, por tanto usando los conceptos que se han dado en las prácticas 9 y 10, así como en las clases teóricas de la asignatura.

2.- Introducción

Se pretende diseñar un sistema de control y gestión remota de un aparcamiento de coches. En concreto se va a realizar una implementación basada en ***Custom Deployment*** o instalación personalizada, debido que para este servicio avanzado se pretende hacer uso de tipos complejos.

Por tanto, será necesario generar un fichero de configuración XML con toda la información necesaria para la instalación. Este fichero es lo que se denomina un ***Web Service Deployment Descriptor (WSDD)*** o descriptor de instalación de servicio Web.

Mediante este sistema de párking, se llevará a cabo el control de las entradas y salidas de coches de un párking. En él, se podrá acceder de forma temporal o se podrá tener una plaza reservada y alquilarla de forma mensual como abonado.

A grandes rasgos, el objetivo de este servicio es:

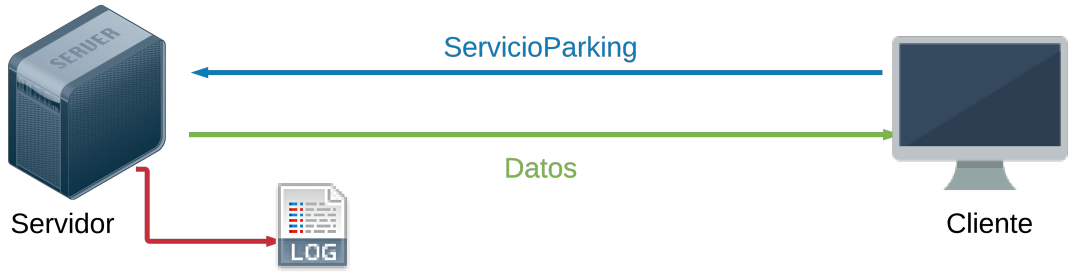
- Controlar el número de coches que entran y salen del aparcamiento, para así mostrar información del número de plazas libres u ocupadas.
- Guardar información/estadísticas de los coches y propietarios que usan el estacionamiento.
- Realizar un log con todas las transacciones económicas que se realizan a lo largo del tiempo.

Todo lo anterior se realiza haciendo uso de 3 clases principales, las cuales tienen diversos métodos accesibles por parte del cliente y otros métodos privados de uso interno para realizar las gestiones, estas clases son: Parking, Coche y Propietario.

Para implementarlo se diferencian dos tipos de roles en función de la forma de pago del servicio, la diferenciación se produce en base a la matrícula del vehículo:

- Propietarios/coches que hacen uso **estándar** del párking, a los cuales se les computa el tiempo de entrada y salida para establecer un coste de su estancia, acorde a la política de tarificación del párking.
- Propietarios/coches que están **abonados** a dicho párking, los cuales pagan una cuota fija por su estacionamiento y no hace falta tarificar en función del tiempo de entrada.

Se muestra, a continuación, el esquema general del funcionamiento del servicio, en el que se pueden distinguir los distintos componentes del sistema (cliente y servidor), cómo se comunican estos entre ellos, las clases que conforman el sistema, y algunos de los atributos que contienen estos.



- Matrícula
- Marca
- Modelo
- Color
- Propietario
- Abonado
- Tiempo entrada
- Tarifa

- Plazas libres
- Plazas ocupadas
- Plazas totales
- Dirección

- Nombre
- Apellidos
- DNI
- Teléfono

Ilustración 1: Esquema de funcionamiento general

3.- Funcionamiento

Este proyecto consta de dos partes diferenciadas:

3.1.- Cliente

Es la parte que se encarga de la ejecución del programa. Simula la entrada de un vehículo al párking. Puede ejecutar varias opciones, en función de lo que desee:

1. Dar de **alta** un vehículo en el registro del párking, lo cual permite al usuario hacer uso de los servicios del párking, sin estar registrado no se puede acceder para aparcar.
2. Dar de **baja** un vehículo del registro del párking. En este caso, el propietario y su coche dejarían de formar parte de la base de datos del párking y, en caso de querer volver acceder a él, sería necesario un nuevo registro completo.
3. **Aparcar** el vehículo en el párking.
4. **Sacar** el vehículo del párking, previo pago del importe correspondiente si el propietario del vehículo no tiene una plaza asociada como abonado del mismo.
5. Obtener los **datos** de un **coche** a partir del DNI de su propietario.
6. Obtener los **datos** del **propietario** de un coche a partir de la matrícula del mismo.
7. Obtener el **estado** de un **coche** a partir de la matrícula del mismo.
8. Obtener los **datos** de todos los **coches** actualmente **aparcados**.
9. Obtener la relación de **datos** entre **coches y propietarios** de dados de **alta** en el párking.

Para iniciar cualquiera de los procedimientos anteriormente indicados, el cliente debe, antes que nada, dar de alta su vehículo en el párking. Cuando lleva esto a cabo, se establece una

comunicación entre el servidor y el cliente en la que se indican varios datos. Por parte del coche, se solicitan:

- **Matrícula**: este parámetro actuará como identificador único del vehículo dentro de todos los que formen parte del párking.
- **Marca**: marca del vehículo que se va a registrar.
- **Modelo**: modelo del vehículo que se va a registrar.
- **Color**: color del vehículo que se va a registrar.
- **Propietario**: en este campo se solicitan los datos del usuario que aparecerá como propietario del vehículo. Se detallan más adelante.
- **Abonado**: este parámetro indica si el usuario tiene una plaza reservada o si no es así.

Por parte del propietario del vehículo, se solicitan los siguientes datos:

- **Nombre**: se pide la introducción del nombre del propietario del vehículo.
- **Apellidos**: se solicitan, además, los apellidos del propietario.
- **DNI**: este campo es el que actuará como identificador único del propietario del vehículo.
- **Teléfono**: se solicita el teléfono propietario del vehículo.

En la ejecución del programa el usuario mantendrá una conversación guiada con el servidor, que será el que, uno a uno, le irá solicitando los datos anteriores y los va registrando y asociando el vehículo con el propietario. Una vez que finaliza la entrada de texto, si todo ha ido correctamente, se almacena el vehículo y su propietario en la base de datos del servidor. Gracias a esto se guardará registro de cada vez que dicho vehículo entre o salga del párking, de las horas a las que lo hace y, en caso de que no sea abonado, de la cantidad de dinero que debe pagar por su estancia en el párking.

3.2.- Servidor

El servidor funciona, según la orden que se le mande, de la siguiente forma:

1. Dar de **alta** un vehículo en el párking:
   I. Recibe un mensaje por parte de algún cliente solicitando añadirse al párking.
   II. Extrae los diferentes campos de ese mensaje (nombre, apellidos, teléfono, DNI, matrícula del vehículo a registrar, marca, modelo, color, etc) y, si todo ha ido bien, registra al vehículo y al propietario que han solicitado su adhesión al párking.

2. **Aparcar** un vehículo que ya registrado en el párking:
   I. Recibe una solicitud por parte del cliente.
   II. Solicita la matrícula del coche a aparcar.
   III. Comprueba si tiene una plaza asociada o no. En caso de tenerla, da paso al vehículo sin más. En caso de no tenerla, comprueba la disponibilidad de plazas libres. Si no hubiese, no deja que el vehículo acceda. En caso contrario, da paso al vehículo. En cualquiera de las situaciones, el párking recoge la hora a la que el vehículo ha accedido al interior del aparcamiento.

3. **Sacar** un vehículo ya aparcado en el párking:
   I. Recibe una solicitud por parte del cliente.
   II. Solicita la matrícula del coche que va a salir.
   III. Comprueba si tiene una plaza asociada o no. En caso de tenerla, le deja salir sin más. En caso contrario, calcula el pago que debe abonar el propietario antes de abandonar el párking. De nuevo, en este caso, el servidor anota a qué hora sale el vehículo del párking.

4. **Eliminar** un vehículo del párking:
   I. Recibe una solicitud por parte del cliente.
   II. Solicita la matrícula del coche que desea eliminarse.
   III. Comprueba que si debe o no dinero al párking. En caso de deberlo, se le solicita.
   IV. Se elimina el coche indicado y todos los datos asociados a él de la base de datos en la que estaba almacenado.

5. **Obtener** los **coches** asociados a un DNI:
   I. Recibe una solicitud por parte del cliente.
   II. Solicita el DNI del propietario que se quiere buscar.
   III. Devuelve todos los coches que dicho propietario posee en la base de datos del párking.

6. **Obtener** el **propietario** de un coche:
   I. Recibe una solicitud por parte del cliente.
   II. Solicita la matricula del vehículo del que se quiere conocer el propietario.
   III. Devuelve el propietario del coche con la matrícula deseada.

7. **Obtener** el **estado** de un **coche**:
   I. Recibe una solicitud por parte del cliente.
   II. Solicita la matricula del vehículo del que se quiere conocer el estado.
   III. Devuelve si dicho vehículo está actualmente dentro o fuera del párking

8. **Obtener** los datos de todos los **coches** actualmente **aparcados**:
   I. Recibe una solicitud por parte del cliente.
   II. Comprueba el estado de los vehículos registrados en el párking y devuelve aquellos que se encuentran aparcados en ese momento.

9. **Obtener** la relación completa de **coches y propietarios registrados** en el párking:
   I. Recibe una solicitud por parte del cliente.
   II. Devuelve la relación completa de vehículos y propietarios existentes en el parking.

Aparte de llevar un control en tiempo real de lo que ocurre en el párking, el servidor también realiza un control interno tanto de los mensajes recibidos como de los mensajes enviados haciendo uso de servicios de log.

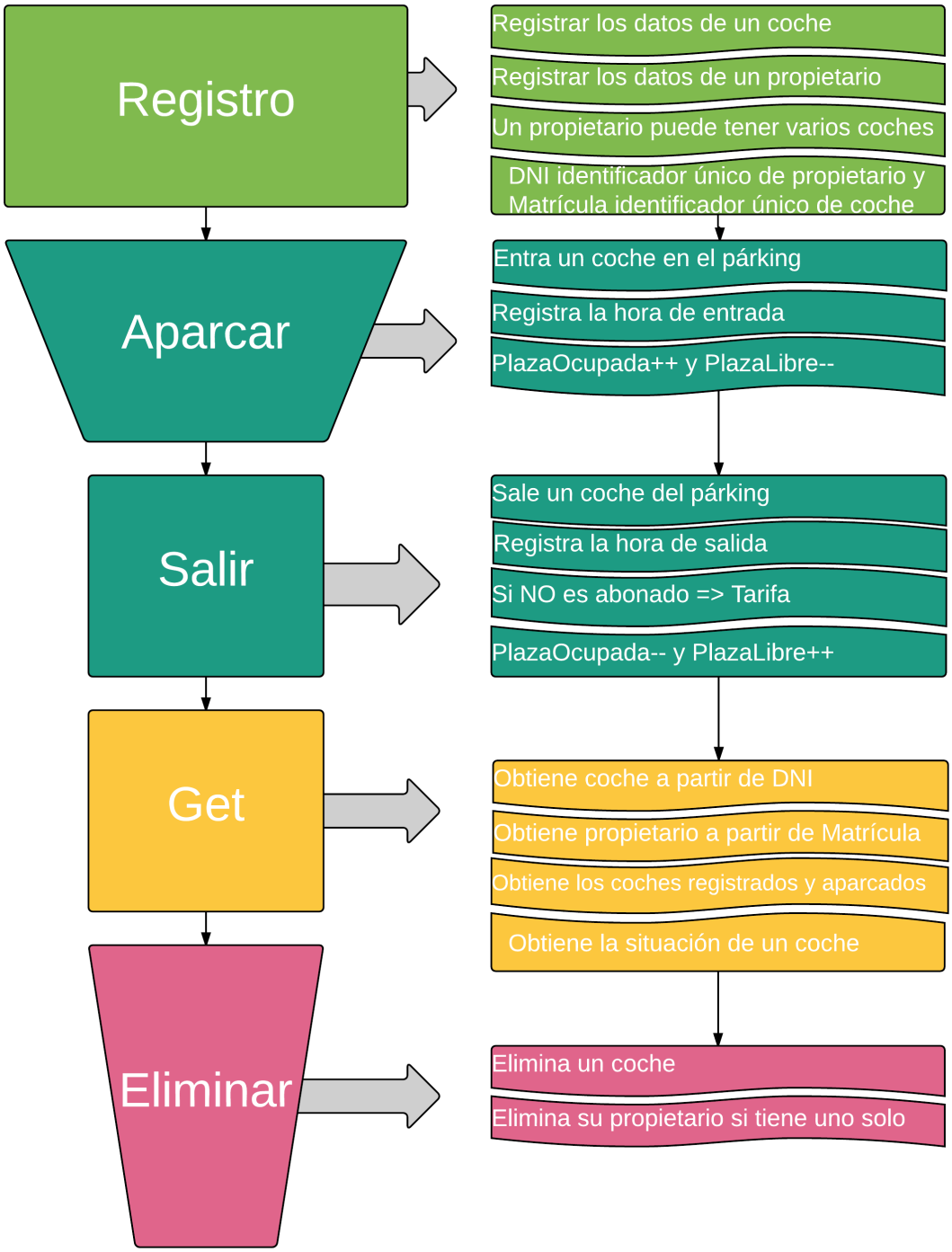


Ilustración 2: Secuencia

4.- Implementación

Para llevar a cabo este proyecto se han usado los dos métodos vistos en las prácticas.

En primer lugar se ha desarrollado mediante **WSDD**, para ello se han implementado las 3 clases de las que consta el proyecto en un package llamado *parkingwebservice*, en estas clases se llevan a cabo todas las tareas descritas a lo largo de esta memoria. Posteriormente se ha desarrollado el fichero deploy.wsdd, en el cual mediante XML se ha definido la estructura de las invocaciones remotas haciendo uso de Servicios Web. Por último, se ha desarrollado un Cliente en el que se implementan los métodos que invocan a los métodos alojados en el servidor. Estas invocaciones se realizan con los servicios proporcionados por axis, para de esta manera poder pasar tipos de datos complejos.

En segundo lugar se ha realizado también un desarrollo usando **WSDL**. Para este caso se ha hecho uso de los servicios proporcionados por axis para generar las clases a partir de la base. Una vez que se generan dichos ficheros en dos directorios *(/es/* y */carlos/) en nuestro caso, se ha procedido a realizar un nuevo Cliente para usar estos métodos.*

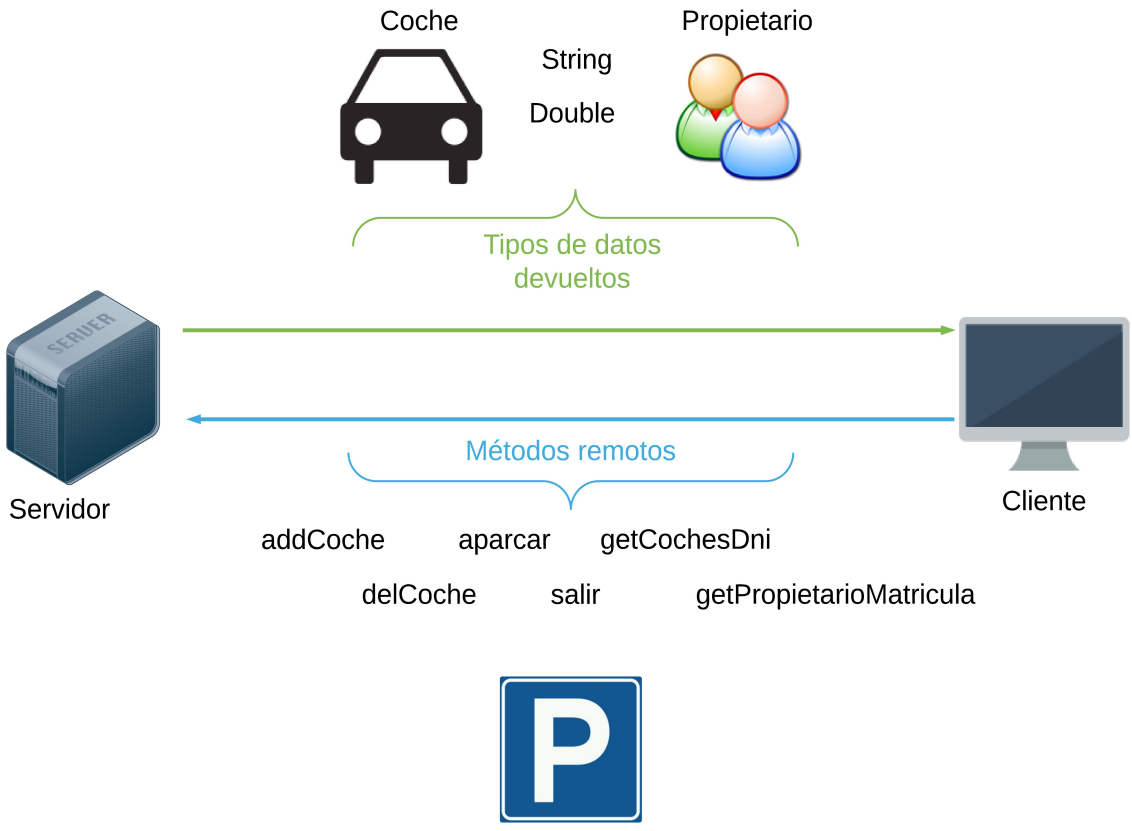


Ilustración 3: Paso de tipos y Métodos remotos

5.- Compilación y ejecución

Para realizar la compilación, se llevarían a cabo los siguientes comandos:

```
# 1 General
        ## 1.1 Limpia lo anterior y compila todas las clases
        make clean
        make all
```

Para ejecutarlo se abren diferentes terminales. Una para el axis y otra para la interacción entre cliente – servidor:

- Terminal 1 (axis):

```
        ## 1.2 Ejecuta axis (Nueva terminal (Siempre abierta))
        make axis
```

- Terminal 2 (cliente):

```
# 2 Método WSDD
        ## 2.1 Ejecuta Deploy
        make deploy
        ## 2.2 Compila Cliente
        make parkingClient
```

Para ejecutar los distintos métodos mediante **WSDD**:

```
        ## 2.3 Ejecuta métodos
        make añadir         # Añade un coche a la BBDD de coches
        make eliminar       # Elimina un coche de la BBDD de coches
        make aparcar        # Entra coche en el párking en función de la matrícula
        make salir          # Sale coche del párking en función de la matrícula
        make getCoches      # Obtiene los coches asociados a un DNI
        make getPropietario # Obtiene el propietario de un coche en base a la matrícula
        make getEstado      # Obtiene el estado de un coche en base a la matrícula
        make getAparcados   # Obtiene los datos de todos los vehículos actualmente aparcados
        make getCoches      # Obtiene los datos de los coches dados de alta en el párking
```

Para ejecutar los distintos métodos mediante **WSDL**:

```
# 3 Método WSDL
        ## 3.1 Ejecuta WSDL
        make parkingWsdl
        ## 3.2 Compila Cliente
        make parkingClient2
        ## 3.3 Ejecuta cliente
        make parkingClient2Exe
```

6.- Relación con la asignatura

Como se puede observar, en este proyecto se ha hecho uso de casi todos los elementos vistos en las prácticas, utilizando conceptos de todas ellas, tales como aspectos relacionados con el periódico, la agenda o el banco que se han desarrollado en las dos prácticas relativas a Servicios Web. Además se ha realizado la implementación mediante los dos métodos (WSDD y WSDL) que se puede hacer teniendo en cuenta el carácter complejo de los datos y métodos que se usan remotamente.