



Estacionamiento Inteligente

Informe del prototipo final, detalles de instalación y uso

Versión: 0108

Año: 2020

♦ Sistemas Distribuidos

Mediante el uso de los recursos aprendidos durante la cursada, se desea simular un estacionamiento inteligente a partir de un sistema distribuido que controle el mismo.

ÍNDICE

1. ¿Qué es?.....	1
2. Descripción.....	2
3. Instalación.....	3
3.1 Módulo proceso barrera.....	3
3.1.1 Descripción.....	3
3.1.2 Interfaz gráfica.....	3
3.1.3 Configuración.....	4
3.2 Módulo proceso sensor.....	5
3.2.1 Descripción.....	5
3.2.2 Interfaz gráfica.....	6
3.3 Módulo manipulador sensor	7
3.3.1 Descripción.....	7
3.3.2 Interfaz gráfica.....	7
3.4 Módulo manipulador barrera.....	8
3.4.1 Descripción.....	8
3.4.2 Interfaz gráfica.....	9
3.4.3 Configuración.....	10
3.5 Objeto distribuido remoto (RMI).....	11
3.5.1 Descripción.....	11
3.6 Servidor DNS.....	12
3.6.1 Configuración.....	12
3.6.2 Mensajes y estados.....	13
4. FAQ 's.....	15
5. Glosario.....	15
6. Grupos e integrantes.....	16

1. ¿Qué es?

«**Estacionamiento Inteligente**» es un software desarrollado para empresas que busquen actualizar su clásico estacionamiento tradicional. Posee un entorno visual muy atractivo, de fácil utilización y empleo. Además, cuenta con un sofisticado sistema de monitoreo constante por medio de videos que brindarán una cuota extra de seguridad al centro comercial, restaurante, entre otros.

En esta guía encontrará caracterizaciones e ilustraciones de cada uno de los procesos que realiza el sistema internamente desarrollado para entender el funcionamiento del mismo¹.

El sistema compuesto está constituido por:

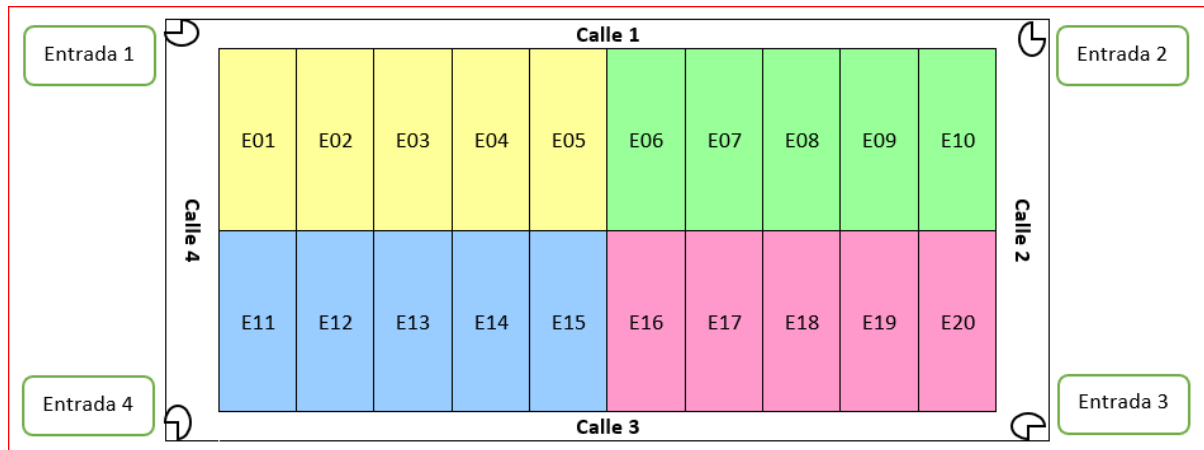
- Procesos manipuladores;
- Procesos barreras;
- Procesos sensores;
- Objeto distribuido;
- Proceso servidor del objeto distribuido;
- Tareas Webservice;

Por otro lado, este objeto distribuido recién mencionado se ejecutará en el equipo que funcione como servidor. En el mismo, se configurarán el DNS y los puertos a utilizar. Luego, podrán editarse la disponibilidad de slots para estacionar, el listado de los dominios permitidos y será quien lleve un registro del estado de ocupación del estacionamiento. Cabe señalar que cambiará el estado de las parcelas a medida que se reciben los cambios realizados a través de los procesos manipuladores.

¹ Es de vital importancia que los puertos de los módulos que el sistema necesita para su correcto funcionamiento estén disponibles y libres. No puede haber ninguna otra aplicación utilizando alguno de ellos. A su vez, es imperioso que la conexión multicast esté habilitada para lograr la comunicación exitosa entre los nodos.

2. Descripción

El boceto sobre el cual se elaboró esta guía de instalación y usuario se ilustra a continuación:



Se simula un estacionamiento con cuatro entradas intervenido por cuatro calles que, en cuya intersección, se generan los ingresos vehiculares al mismo::

- Entrada 1 (superior izquierda);
- Entrada 2 (superior derecha);
- Entrada 3 (inferior derecha);
- Entrada 4 (inferior izquierda),

Y que posee veinte slots, o parcelas, distribuidas uniformemente para dar lugar a los lugares en el estacionamiento:

- Entrada 1: E01 a E05 (amarillo);
- Entrada 2: E06 a E10 (verde);
- Entrada 3: E11 a E15 (rojo);
- Entrada 4: E16 a E20 (azul).

En términos generales, al llegar un vehículo a una de las cuatro entradas se genera una consulta al *objeto distribuido*, encargado de verificar el estado del estacionamiento, más precisamente si hay disponibilidad o no. Si la hay, se levanta la barrera pertinente y habrá una pantalla que muestre cuántos lugares quedan libres, cuántos ocupados y se le indica al conductor a qué slot y calle debe dirigirse. En caso de no haber lugar, es decir, que los veinte slots estén ocupados se mostrará un mensaje de «estacionamiento lleno» y no se levantará la barrera.

Cada slot puede tener hasta tres estados distintos:

- Libre: no hay un vehículo estacionado;

- Ocupado: hay un vehículo estacionado;
- Reservado: estado transitorio de *libre* a *ocupado* desde que el vehículo ingresa al estacionamiento y procede a ocupar el lugar indicado por el sistema. Cuando el sistema detecta el auto dentro del slot, además de cambiar el estado a *ocupado*, actualiza el listado de los lugares libres y notifica a los demás procesos involucrados. Cuando se abandona una parcela se aggiorna el estado del mismo a *libre* y también el conteo de lugares libres.

3. Instalación

3.1 Módulo proceso barrera

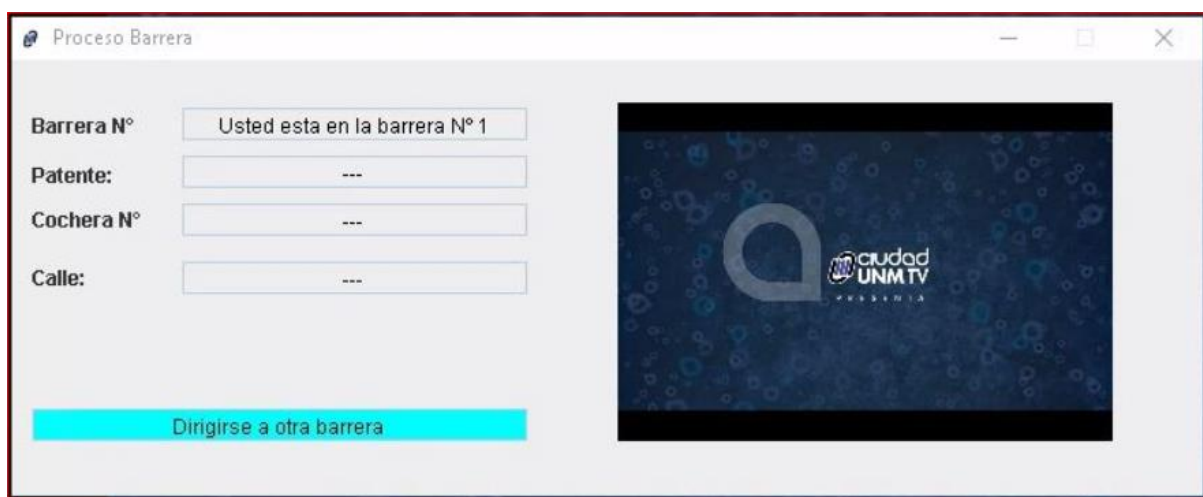
3.1.1 Descripción

Existe un *proceso barrera* por cada entrada del estacionamiento. Se recibe el dominio, o patente, de algún vehículo a través del *manipulador barrera* y la enviará al sistema, junto con el número de entrada para que este pueda hacer la reserva del lugar dentro del estacionamiento. Cuando recibe el resultado del *objeto distribuido* lo exhibe por pantalla indicando el número de slot reservado, por lo que el cliente sabrá qué parcela lo espera. Además de informar al *manipulador barrera*, que debe mostrar el mensaje «barrera abierta» o «estacionamiento lleno». Cabe destacar que se dispondrá de un video obtenido por medio de streaming a través de la red utilizando *sockets multicast*.

3.1.2 Interfaz gráfica



A la izquierda se observa la información del vehículo y lugar del estacionamiento. A la derecha, el video



Conexión fallida con el proceso distribuido

3.1.3 Configuración

Al momento de instalar el presente software, el usuario deberá ubicar el archivo de configuración «*BarrerasConfiguracion.txt*» en su directorio de trabajo. Éste dependerá del sistema operativo de la máquina, por lo que si se trata de Windows será, por defecto, en C://Users//"NombreUsuario" o bien en Linux /Home/"NombreUsuario".

Dentro de ese directorio es donde radica el archivo y es donde irá a buscarlo el programa para su posterior ejecución. Existirán cuatro copias de este programa y en cada máquina o PC el programa será el mismo. Lo que se debe modificar es en el .txt ya mencionado:

- IP del DNS server: todas las máquinas tendrán el mismo DNS server;
- El número de barrera: (de 1 a 4);
- Puerto de escucha del *proceso barrera*: «3000» en todos los casos;
- IP del servidor objeto distribuido: igual en todos los casos;
- Puerto de escucha del servidor objeto distribuido (RMI): igual en todos los casos;
- Puerto servicio «keep alive» entre *manipulador* y *barrera* N°1 -- 3004 // N°2 -- 3005 // N°3 -- 3006 // N°4 -- 3007;
- Puerto servicio «keep alive» entre el *servidor objeto distribuido* y *barrera*: N°1 -- 5051 // N°2 -- 5052 // N°3 -- 5053 // N°4 -- 5054.

3.2 Módulo proceso sensor

3.2.1 Descripción

Este módulo no requiere la intervención del usuario. Existen cuatro procesos que involucran a los sensores y cada uno corresponde a los cuatro sectores del estacionamiento.

Cada uno de los *procesos sensores* tiene cinco hilos (threads) y reciben los datos de los autos que ingresan por medio del *manipulador de sensores*. Ésto se realiza mediante sockets UDP. La información se procesa y se envía como mensaje con los datos del vehículo al *objeto distribuido* esperando una respuesta por *Java² RMI* del mismo. En base a esa respuesta se observa si el vehículo pudo estacionar, o no, y se le responde al *manipulador de sensores* nuevamente por sockets UDP.

Se presentará por pantalla una interfaz gráfica que mostrará el bloque y la parcela que ocupará el vehículo en el estacionamiento, la acción a realizar (estacionar o liberar) y el estado final, si efectivamente se estacionó o se liberó el lugar.

Una vez que se reciben los datos del *manipulador de sensores*, se procesan los mismos y se comunican por medio de *Java RMI* con el *objeto distribuido*. Esta práctica es necesaria para que presenten en su interfaz gráfica dicha acción, mientras se actualiza y consulta a la base de datos.

² <https://www.oracle.com/java/technologies/javase-jre8-downloads.html>

Cada uno de los cuatro *procesos sensores* cuenta con dos réplicas que, junto con el *proceso manipulador* por medio del *algoritmo del Grandulón* generan redundancia. De ocurrir algún inconveniente se inicia un algoritmo de selección para que alguna de las dos réplicas pueda reemplazar al original y que el programa pueda continuar su óptimo funcionamiento. Estas dos réplicas son independientes, pueden estar en la misma PC o en otra para realizar las tareas de recibir los datos por UDP y enviarlos por *Java RMI* al *objeto distribuido*.

Para finalizar la descripción del módulo de *proceso sensor*, cabe destacar la importancia del servidor DNS. Las direcciones IP del *manipulador de sensores* y del *objeto distribuido* se obtienen por el DNS creado por el *proceso barrera*.

3.2.2 Interfaz gráfica

The image displays four overlapping windows of the 'Proceso Sensor' graphical interface. Each window contains four labeled input fields: 'Bloque', 'Parcela', 'Acción', and 'Estado'. The 'Bloque' field is populated with the numbers 1, 2, 3, and 4 in the four windows respectively. The other three fields ('Parcela', 'Acción', and 'Estado') are currently empty in all instances. The windows are arranged in a 2x2 grid, with the top-left window being the most prominent and the others slightly offset behind it.

Cada proceso sensor cuenta con su respectivo bloque, además de la parcela, acción y estado en que se encuentra

3.3 Módulo manipulador sensor

3.3.1 Descripción

El *manipulador sensor* se comunica con el *proceso sensor* a través de sockets UDP. Capaz de brindar información del dominio, si desea estacionar o liberar un slot y selecciona qué lugar ocupará el vehículo que ingresa al estacionamiento. Se hace énfasis en el sensor de ubicación como eje central en este proceso y su correspondiente simulación. La respuesta confirmará, o no, la operación.

Mediante la simulación, un sensor puede determinar la posición del automóvil, ya sea para estacionar o liberar la parcela, mientras que otro capturará la patente por medio de streaming. Por ende, se dice que este proceso actúa como si fuese una interfaz con la que interactúa con el *proceso sensor*.

3.3.2 Interfaz gráfica



Interfaz donde se presenta la información del vehículo y lugar del estacionamiento

The screenshot shows a window titled 'Manipulador Sensor'. It contains the following elements:

- A text field labeled 'PATENTE' with the value 'ABC123'.
- A dropdown menu labeled 'BLOQUE' with the value '1'.
- A dropdown menu labeled 'ACCION' with the value 'LIB'.
- Five buttons labeled 'SEN 1', 'SEN 2', 'SEN 3', 'SEN 4', and 'SEN 5'.
- A text field at the bottom containing the text 'LIBERADO E:01'.
- The version 'V 1.0' in the bottom right corner.

Ejemplo de LIBERACIÓN del lugar donde se encontraba el vehículo estacionado

The screenshot shows the same 'Manipulador Sensor' window, but with the following changes:

- The 'ACCION' dropdown menu now shows the value 'EST'.
- The text field at the bottom now contains the text 'AUTO ESTACIONADO E:01'.
- All other elements (PATENTE, BLOQUE, buttons, and version) remain the same.

Ejemplo de vehículo ESTACIONADO según bloque y slot

3.4 Módulo manipulador barrera

3.4.1 Descripción

En el programa *manipulador barrera* existen tres clases: clase ventana, hilo cliente e hilo keep alive.

La clase ventana es quien genera la interfaz gráfica mostrando los datos al usuario e instancia a las otras dos clases -hilo cliente e hilo keep alive-. La clase ventana es la de mayor importancia.

Por su parte, se invoca al hilo cliente al presionar el botón para que ingrese a una barrera. Tal acción provoca que el hilo envíe un mensaje al *proceso barrera* correspondiente a la información de la patente del vehículo. Luego espera recibir una respuesta y en función de ésta modificar la interfaz. Cada botón debe ingresar la patente de cada vehículo que llama a un hilo diferente. Además de realizar una consulta y en función de ello tomar una decisión.

En cuanto al hilo keep alive recibe mensajes de las barreras para ver si se tiene conexión al *servidor RMI* (donde se encuentra la información necesaria de los vehículos del estacionamiento). Si recibe que sí se está conectado al *servidor RMI* habilita los botones. Este acontecimiento se produce para evitar que el manipulador envíe peticiones de ingreso cuando se ha perdido el contacto con el *proceso distribuido*.

3.4.2 Interfaz gráfica



Interfaz donde se presenta una BAJA de barrera para el ingreso de un auto



Interfaz donde se presenta un ejemplo de ALTA de barrera e ingreso de un auto

3.4.3 Configuración

Dentro del archivo de configuración de este módulo se encuentran los puertos de comunicación con los cuatro *proceso barrera* y la IP del DNS-Server:

Ej:

xxx.xxx.xxx.xxx -- IP del DNS-Server:

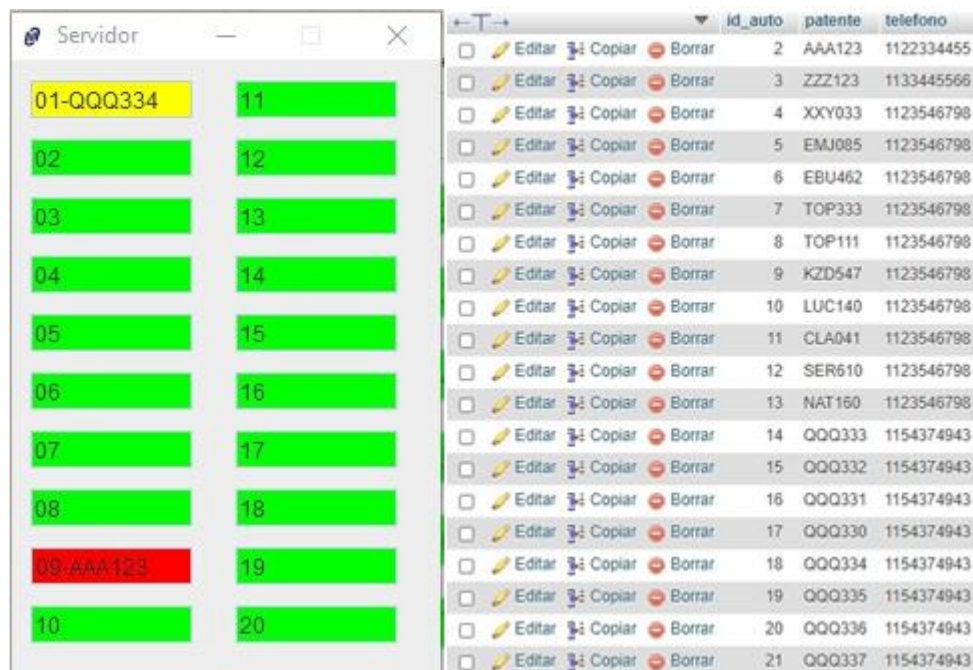
- 3000: Puerto por el que se envían los datos al *proceso barrera* correspondiente;
- 3001: Puerto por el que se reciben los datos al *proceso barrera* correspondiente;
- 3004: Puerto por donde se escuchan los Keep Alive de la barrera 1;
- 3005: Puerto por donde se escuchan los Keep Alive de la barrera 2;
- 3006: Puerto por donde se escuchan los Keep Alive de la barrera 3;
- 3007: Puerto por donde se escuchan los Keep Alive de la barrera 4.

3.5 Objeto distribuido remoto (RMI)

3.5.1 Descripción

El dispositivo que contiene la base de datos del estacionamiento es accedido a través del RMI. Éste se instala en el dispositivo que funcionará como servidor a través del Webservice. De esta manera se podrá editar en el archivo SQL la disponibilidad de slots para estacionar, el listado de patentes permitidas y quien lleva registro del estado de ocupación del estacionamiento. Se modificará el estado de los slots a medida que se reciben los cambios realizados a través de los procesos.

- Se debe instalar la aplicación XAMPP³;
- Iniciar el Apache Server;
- Utilizar el archivo SQL (edición según la necesidad del cliente);
- Los clientes RMI son configurados en el *proceso barrera y sensor*.



	id_auto	patente	telefono
<input type="checkbox"/> Editar <input type="checkbox"/> Copiar <input type="checkbox"/> Borrar	2	AAA123	1122334455
<input type="checkbox"/> Editar <input type="checkbox"/> Copiar <input type="checkbox"/> Borrar	3	ZZZ123	1133445566
<input type="checkbox"/> Editar <input type="checkbox"/> Copiar <input type="checkbox"/> Borrar	4	XXY033	1123546798
<input type="checkbox"/> Editar <input type="checkbox"/> Copiar <input type="checkbox"/> Borrar	5	EMJ085	1123546798
<input type="checkbox"/> Editar <input type="checkbox"/> Copiar <input type="checkbox"/> Borrar	6	EBU462	1123546798
<input type="checkbox"/> Editar <input type="checkbox"/> Copiar <input type="checkbox"/> Borrar	7	TOP333	1123546798
<input type="checkbox"/> Editar <input type="checkbox"/> Copiar <input type="checkbox"/> Borrar	8	TOP111	1123546798
<input type="checkbox"/> Editar <input type="checkbox"/> Copiar <input type="checkbox"/> Borrar	9	KZD547	1123546798
<input type="checkbox"/> Editar <input type="checkbox"/> Copiar <input type="checkbox"/> Borrar	10	LUC140	1123546798
<input type="checkbox"/> Editar <input type="checkbox"/> Copiar <input type="checkbox"/> Borrar	11	CLA041	1123546798
<input type="checkbox"/> Editar <input type="checkbox"/> Copiar <input type="checkbox"/> Borrar	12	SER610	1123546798
<input type="checkbox"/> Editar <input type="checkbox"/> Copiar <input type="checkbox"/> Borrar	13	NAT160	1123546798
<input type="checkbox"/> Editar <input type="checkbox"/> Copiar <input type="checkbox"/> Borrar	14	QQQ333	1154374943
<input type="checkbox"/> Editar <input type="checkbox"/> Copiar <input type="checkbox"/> Borrar	15	QQQ332	1154374943
<input type="checkbox"/> Editar <input type="checkbox"/> Copiar <input type="checkbox"/> Borrar	16	QQQ331	1154374943
<input type="checkbox"/> Editar <input type="checkbox"/> Copiar <input type="checkbox"/> Borrar	17	QQQ330	1154374943
<input type="checkbox"/> Editar <input type="checkbox"/> Copiar <input type="checkbox"/> Borrar	18	QQQ334	1154374943
<input type="checkbox"/> Editar <input type="checkbox"/> Copiar <input type="checkbox"/> Borrar	19	QQQ335	1154374943
<input type="checkbox"/> Editar <input type="checkbox"/> Copiar <input type="checkbox"/> Borrar	20	QQQ336	1154374943
<input type="checkbox"/> Editar <input type="checkbox"/> Copiar <input type="checkbox"/> Borrar	21	QQQ337	1154374943

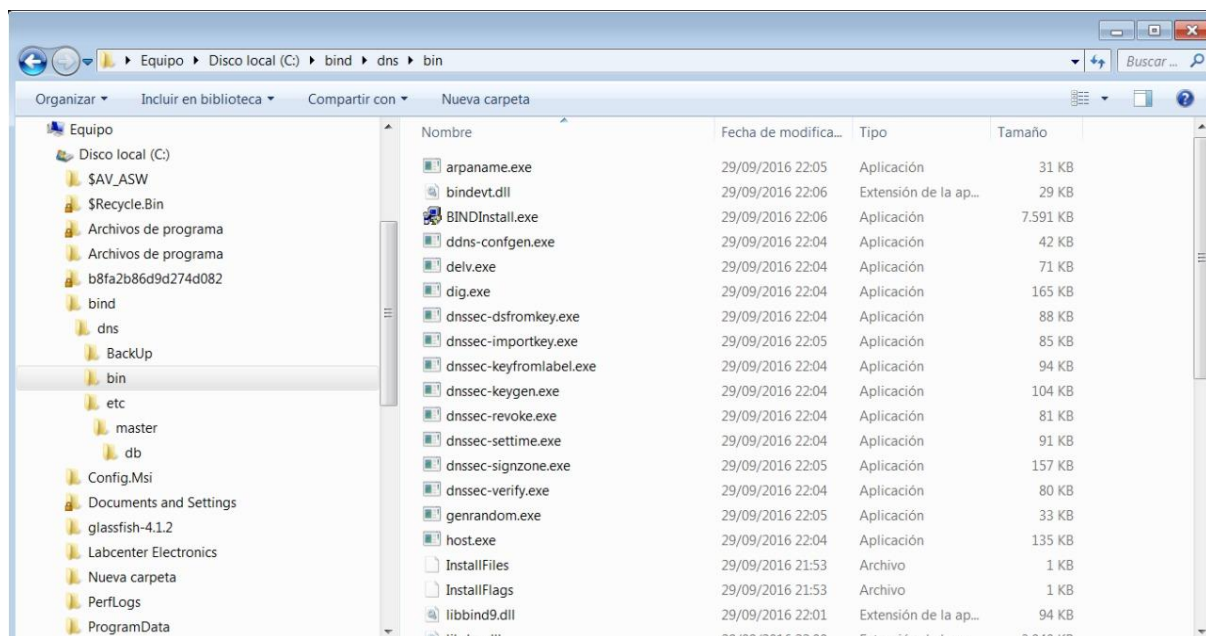
Vista de configuración de datos del estacionamiento

³ <https://www.apachefriends.org/download.html>

3.6 Servidor DNS

3.6.1 Descripción

Un servidor DNS es un método utilizado para traducir de forma sencilla de recordar los nombres de dominio. Es factible la instalación de cualquier tipo de servidor DNS. Se recomienda hacer uso del DNS BIND9⁴ para plataformas que operan con Windows. El cliente deberá ver una carpeta similar a la siguiente:



En cuanto a la base de datos que almacenará la información, el cliente realizará las modificaciones necesarias dependiendo de dónde se instalen los equipos, ya que al desplegar las máquinas cada uno tendrá una dirección IP diferente. En otras palabras, se les asignará los *procesos barrera* direcciones IP que serán diferentes entre tales y en comparación al *servidor objeto distribuido*.

Por consiguiente, el cliente tendrá la tarea de configurar la dirección IP del servidor DNS en cada máquina para que puedan visualizarlo. La misma se encuentra en la carpeta principal, donde existe un archivo llamado «IP» que debe contener la dirección del servidor DNS.

⁴ <https://www.isc.org/download/#BIND>

3.6.2 Mensajes y estados

Para establecer una conexión exitosa con el servidor DNS BIND9 deberá dirigirse, en Windows, al «cmd» y escribir los comandos que se muestran a continuación::

```
C:\Windows\system32\cmd.exe
Microsoft Windows [Versión 6.1.7601]
Copyright (c) 2009 Microsoft Corporation. Reservados todos los derechos.

C:\Users\UNM>dns

C:\Users\UNM>cd..

C:\Users>cd..

C:\>cd bind

C:\bind>cd dns

C:\bind\dns>cd bin

C:\bind\dns\bin>start named.exe -f
```

También se necesitarán realizar pruebas del óptimo funcionamiento del servidor DNS:

```
C:\bind\dns\etc>nslookup dns.unm.ar
Servidor: UnKnown
Address: 192.168.1.40

Nombre: dns.unm.ar
Address: 25.3.179.86

C:\bind\dns\etc>nslookup barrera1.unm.ar
Servidor: UnKnown
Address: 192.168.1.40

Nombre: barrera1.unm.ar
Address: 25.0.85.115
```

Por último, deberá contemplar la importancia de la base de datos. El cliente la deberá configurar dependiendo de las direcciones IP que posean las otras computadoras donde estarán los *procesos barrera* 1, 2, 3, 4; además del manipulador, entre otros. La base de datos se encontrará en “C:\bind\dns\etc\master\db\db.unm.ar”, y de realizarse alguna modificación en ésta se procederá al reinicio del servidor DNS para que los cambios estén disponibles.

```
$ttl 15s
@ IN SOA dns.unm.ar root.unm.ar. (
    2022211122 ;Serial number
    3H         ;Refresh Interval
    30M        ;Retrytry
    2W         ;Expiration date
    15S)       ;Min TTL
TXT "unm web"
NS dns.unm.ar.
dns A 25.3.179.86
barrera1 A 25.0.85.115
barrera2 A 25.7.177.111
barrera3 A 25.7.177.121
barrera4 A 25.7.177.211
servidorobjdistribuidos A 25.106.211.49
manipulador A 25.144.84.70
sensor1 A 25.7.177.131
sensor2 A 25.7.177.141
sensor3 A 25.7.177.151
sensor4 A 25.7.177.161
videomulticast A 231.1.1.1
```

Ejemplo de una base de datos modelo

4. FAQ's

1. Uno de los puertos que el software necesita para funcionar está ocupado. ¿Qué debo hacer?

Si uno de los puertos que el sistema exige se encuentra ocupado deberá liberarlo y volver a intentar la instalación.

2. Una de las barreras permanece baja aún viendo que existe un lugar desocupado. ¿Qué puede suceder?

Los sensores poseen un tiempo determinado de respuesta.

Probablemente el sistema todavía no ha registrado el retiro de un vehículo de la parcela.

3. No estoy de acuerdo con el slot que me han asignado. ¿Qué hago?

Por el momento el presente software no contempla esta clase de decisiones. Sólo debe dirigirse al campo que se le ha atribuido.

4. Me ha aparecido el cartel «Patente duplicada. Comuníquese con el administrador». ¿Qué significa?

No es posible que existan dos vehículos con la misma patente. El sistema lo rechazará. Por lo tanto, se debe dirigir al administrador para solucionar el inconveniente.

5. Glosario

A continuación, se indican los términos utilizados en el presente documento:

Término	Descripción
UDP	Protocolo a nivel capa de transporte (capa 4) basado en el intercambio de datagramas (paquetes de información).
Socket	Método para la comunicación entre programas o procesos en una red.
IP Multicast	Protocolo que permite la transmisión de información desde un punto a múltiples destinatarios.
RMI	Forma parte del entorno de Java y proporciona un mecanismo simple para la comunicación de manera remota.
DNS	Sistema de nomenclatura jerárquico descentralizado para dispositivos conectados a redes IP como Internet o una red privada.

6. Grupos e integrantes

3.1 Módulo proceso barrera: Grupo 1

Alfredo Rábago - Lucas Giménez - Christian Polo - Facundo Souza

3.2 Módulo proceso sensor: Grupo 2

Claudio Sandoval - Natalia Gonza - Luis Gutiérrez

3.3 Módulo manipulador sensor: Grupo 3

Sergio Sandoval - Luciano Gutiérrez - Juan Kromm

3.4 Módulo manipulador barrera: Grupo 4

Agustín Cerávol - Pablo Valdéz - Alejandro Dotti