



## (Analysis Multi-Scenario to Achieve Reliable Indicators)



30-NOV-2020 MINECO19P09-EP09V1.0

## E9 - Protocolo IOT para Ecodriving

**TEKIA Ingenieros, S.A.**

Calle Chile, 4

28290 Las Rozas, Madrid (ESPAÑA)

Tel. +34 916300505





# CONTENIDO

<b>1 INTRODUCCIÓN</b>	<b>4</b>
1.1 RESUMEN DE ENTREGABLES DEL PROYECTO AMSARI	4
1.2 CRITERIOS DE ACEPTACIÓN DEL ENTREGABLE	4
1.3 REQUISITOS MÍNIMOS DEL ENTREGABLE	4
<b>2 CONTEXTO DE SISTEMAS</b>	<b>5</b>
2.1 ARQUITECTURA DE SISTEMAS	6
<b>3 PROTOCOLO IOT PARA UN SISTEMA ECODRIVING</b>	<b>9</b>
3.1 TRANSFERIR FICHERO ECOFILE	9
3.1.1 Mensaje FileTransferRequest	10
3.1.2 Mensaje FileTransferResponse	10
3.2 OBTENER DATOS DE CONFIGURACIÓN	11
3.2.1 Mensaje ConfigurationRequest	11
3.2.2 Mensaje ConfigurationResponse	12
3.3 OBTENER DATOS DE INFORME PARA EL CONDUCTOR	12
3.3.1 Mensaje DriverReportDataRequest	13
3.3.2 Mensaje DriverReportDataResponse	13
<b>4 API REST PARA OBTENER DATOS DE LA EMPRESA</b>	<b>15</b>
4.1 ESTABLECER DATOS DE RECURSOS	15
4.1.1 Establecer datos de centros de operación	15
4.1.2 Establecer datos de concesiones	17
4.1.3 Establecer datos de conductores	19
4.1.4 Establecer datos de modelos de vehículos	22
4.1.5 Establecer datos de vehículos	23
4.1.6 Establecer datos de asignaciones de vehículos	26
4.2 ESTABLECER DATOS DE CALENDARIO	28
4.2.1 Establecer datos de grupos de calendario	28
4.2.2 Establecer datos de temporadas	30
4.2.3 Establecer datos de tipos de día de movimiento	31
4.2.4 Establecer datos de calendario	33
4.3 ESTABLECER DATOS DE PERÍODOS DE DEMANDA	36
4.3.1 Establecer datos de grupos de franjas horarias	36
4.3.2 Establecer datos de franjas horarias	38
4.3.3 Establecer datos de periodos de demanda	40
4.4 ESTABLECER DATOS DE FRANJAS HORARIAS NATURALES	43
4.5 ESTABLECER DATOS DE LA RED TOPOLÓGICA	45



4.5.1 Establecer datos de grupos de explotación .....	45
4.5.2 Establecer datos de subgrupos de explotación.....	47
4.5.3 Establecer datos de líneas.....	49
4.5.4 Establecer datos de sublíneas .....	51
4.5.5 Establecer datos de rutas.....	53
4.5.6 Establecer datos de secuencia de paradas de rutas .....	55
4.5.7 Establecer datos de paradas .....	57
4.6 ESTABLECER DATOS DE EXPEDICIONES REALIZADAS .....	59
4.7 OBTENER DATOS DE INFORME PARA EL CONDUCTOR .....	61
<b>5 EVENTOS DE ECODRIVING .....</b>	<b>63</b>
1.1 MV – MOVIMIENTO .....	64
1.2 DT – DETENCIÓN .....	64
1.3 FB – FRENADA BRUSCA.....	65
1.4 AF – ACELERACIÓN SEGUIDA DE FRENADA .....	65
1.5 DB – DETENCIÓN BRUSCA.....	66
1.6 GB – GIRO BRUSCO .....	66
1.7 AB – ACELERACIÓN BRUSCA .....	67
1.8 PB – PASO BRUSCO POR RESALTE (FIRME IRREGULAR).....	67
5.1 TIPOS DE DATOS.....	67
5.2 CONCEPTO DE VEC, FVEC Y AVEC.....	68

# 1 INTRODUCCIÓN

## 1.1 Resumen de entregables del proyecto AMSARI

PAQUETE DE TRABAJO		ENTREGABLES	AÑO
PT0	GESTIÓN DEL PROYECTO	• E1: PLAN DE PROYECTO	2019
		• E2: INFORME FINAL DE PROYECTO	2020
PT1	DESARROLLO DE UN MÉTODO, AMSARI, DE OBTENCIÓN DE INDICADORES COMPLEJOS	• E3: INDICADOR COMPLEJO DE CONSUMO	2020
		• E4: INDICADOR COMPLEJO DE EFICIENCIA DE LOS ESTILOS DE CONDUCCIÓN	2020
PT2	DESARROLLO DE UN ENTORNO DE PRUEBA. IMPLEMENTACIÓN, PRUEBAS Y VALIDACIÓN DE LOS PROCESOS DESARROLLADOS EN PT1	• E5: ENTORNO BIG DATA CONFIGURADO (Software)	2019
		• E6: ARQUITECTURA IMPLEMENTADA (Software)	2019
		• E7: ENTORNO DE PRUEBA (SW)	2020
		• E8: INFORMES DE PRUEBAS	2020
PT3	DISEÑO DE UNA SOLUCIÓN DE ECODRIVING EN LA NUBE	• E12: ARQUITECTURA DE SOFTWARE PARA UN SISTEMA EN LA NUBE	2020
		• E9: PROTOCOLO IOT PARA UN SISTEMA DE ECODRIVING	2020
PT4	DIFUSIÓN DEL PROYECTO Y SUS RESULTADOS	• E10: PÁGINA WEB DEL PROYECTO, CON CONTENIDOS	2020
		• E11: ARTÍCULO PARA DIVULGACIÓN DE LOS RESULTADOS DEL PROYECTO	2020

Este documento contiene la definición de un protocolo IoT para Ecodriving, entregable E9.

## 1.2 Criterios de aceptación del entregable

ENTREGABLE	Descripción	Clasificación	Criterio de aceptación
E9	PROTOCOLO IOT PARA ECODRIVING	Documentación	El documento describe los mensajes de intercambio entre el sistema y los dispositivos embarcados, así como su estructura, además de los tipos y longitud de los datos contenidos en dichos mensajes.

## 1.3 Requisitos mínimos del entregable

REQUISITO	Descripción	Criterio de aceptación
R18	ADECUACIÓN DEL	El protocolo contempla todos los datos requeridos para aplicar el
		Existe una manera de verificar que los mensajes definidos en el protocolo se



	<p>PROTOCOLO AL MÉTODO AMSARI</p>	<p>método AMSARI (salvo los que se obtengan de otro backoffice)</p>	<p>corresponden con los datos requeridos por el BackOffice usando el método AMSARI y se identifica cuales deben necesariamente provenir de los vehículos y cuales pueden obtenerse de otro sistema.</p>
--	-----------------------------------	---	---

## 2 CONTEXTO DE SISTEMAS

Una de las aplicaciones inmediatas del método AMSARI es en la implementación de un sistema de ecodriving que permita caracterizar el estilo de conducción de múltiples conductores de forma comparable (en escenarios similares) de forma que se puedan obtener indicadores de eficiencia de la conducción, de confort y de consumo que reflejen la realidad de la conducción con independencia de factores externos que la influyan.

No se desea, sin embargo, implementar un sistema de ecodriving completo (incluyendo las soluciones de adquisición de datos de los vehículos) que pudiera ofrecerse llave en mano, sino una solución de analítica de ecodriving que pueda ofrecerse como servicio y permita utilizar soluciones de adquisición de datos de terceros, o dicho de otra forma, una solución a disposición de cualquiera que pueda obtener los datos necesarios de los vehículos, normalmente integradores de sistemas proveedores de sistemas de validación y venta y sistemas de ayuda a la explotación u otros integradores que instalan equipamiento en autobuses.

Dicho de forma simple, la solución ideal sería una en la que un tercero pudiera conectar sus equipos, y automáticamente disponer de los sofisticados indicadores de ecodriving obtenidos con el método AMSARI, todo ello sin que fuera necesaria la intervención de TEKIA como proveedor del servicio.

En la práctica, esta "conexión de equipos de un tercero" implica varios tipos de intercambios de información, no solamente con equipamiento embarcado para obtener las medidas necesarias para el cálculo de los indicadores, sino también con otros sistemas centrales para obtener toda la información de contexto necesaria (vehículos, conductores, rutas, tipos de días y otra). Lamentablemente, mientras para esta última información existen especificaciones públicas y estándares normalizados (como SIRI, NeTex o GTFS) para las medidas de ecodriving, que constituyen gran parte de la información a obtener, no existe ninguna normalización.

Consecuentemente, el proyecto ha llevado a cabo dos líneas de actuación para diseñar una solución de ecodriving como servicio basada en el método AMSARI:

- El diseño de una **arquitectura de software** que posibilite por un lado la conexión de múltiples fuentes de información, procedente del equipamiento en vehículos provisto por terceros y por otro que ofrezca el servicio de analítica a múltiples clientes, cada uno titular de un conjunto de los vehículos conectados. Este es el objeto de E12.
- La especificación de un **protocolo de datos de ecodriving** que facilite la conexión de dicho equipamiento embarcado, a falta de especificaciones normalizadas. Este es el objeto de E9.

Este documento constituye E9.



## 2.1 ARQUITECTURA DE SISTEMAS

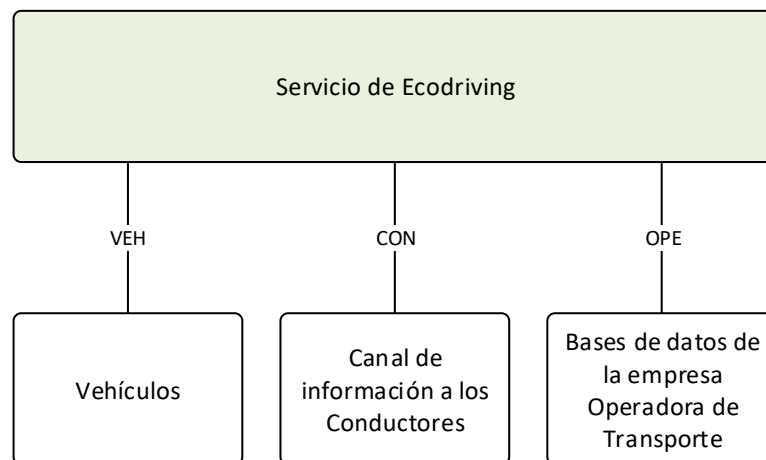
El protocolo de datos ecodriving soportará el principal intercambio de datos entre los sistemas embarcados y el servicio en la nube de analítica de ecodriving, pero existen otros intercambios de datos necesarios. Este apartado ofrece una visión general de los sistemas involucrados y los intercambios de datos necesarios.

Aunque el detalle de esta arquitectura puede variar dependiendo de las soluciones de terceros, en el caso más general incluiría siempre los siguientes elementos:

- El sistema central AMSARI ecodriving ofrecido como servicio en la nube
- El sistema central de la empresa operadora de transporte (Bases de datos de la empresa)
- El equipamiento de adquisición de datos en los vehículos.
- Una o varias soluciones para hacer llegar la información a los conductores (web, App o una pantalla que forma parte del equipamiento embarcado)

Identificamos tres tipos de interfaces de intercambio de datos necesarios

- Interfaces con los vehículos (VEH)
- Interfaces con los canales de información a los conductores (CON)
- Interfaces con las bases de datos del operador de transporte (OPE)



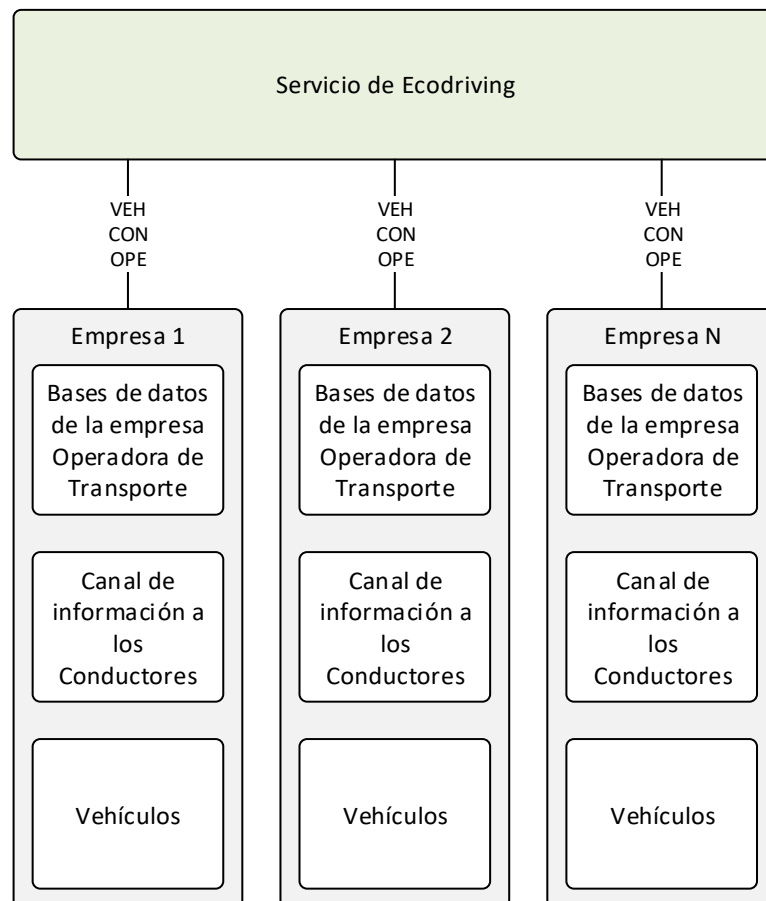
Cada uno de estos tipos de interfaces incluyen varios protocolos para intercambiar datos específicos:

- VEH
  - Datos de ecodriving. Estos datos representan la mayor parte de la información del sistema, tanto en tipos de datos como, sobre todo, en cantidad. Su especificación es objeto de E9.
  - Datos de configuración de los equipos embarcados que generan los eventos de ecodriving (por ejemplo umbrales de aceleración) y comandos enviados desde el servicio de ecodriving a los vehículos (por ejemplo para solicitar la descarga de información, o para desactivar determinadas indicaciones a los conductores o u otras)
  - Eventos de tiempo real enviados desde los equipos embarcados al servicio de ecodriving (por ejemplo averías detectadas en el vehículo)



- CON
  - Servidor de App y Web para mostrar a los conductores informes sobre su conducción
  - Web service ofreciendo el contenido de los informes de conductores para que un tercero implemente el canal (por ejemplo para generar un informe en el pupitre del conductor)
- OPE
  - Datos de contexto. Planificación y catálogos (Conductores, Vehículos, Centros de Operación, Topología de la red, Calendario)
  - Datos sobre los servicios realizados que puedan cruzarse con los datos de ecodriving (Hora inicio de cada expedición, Ruta, Vehículo y Conductor) para determinar a posteriori a qué conductor perteneces dichos datos y en qué rutas ha conducido.

En un contexto con múltiples operadores de transporte y múltiples proveedores de sistemas embarcados, el servicio de Ecodriving tendría que implementar una gran cantidad de interfaces. Consecuentemente se desea utilizar interfaces estandarizados cuando existan, y especificar y hacer público un interfaz, basado en E9, especializado en información de ecodriving, para el que no existen especificaciones abiertas.

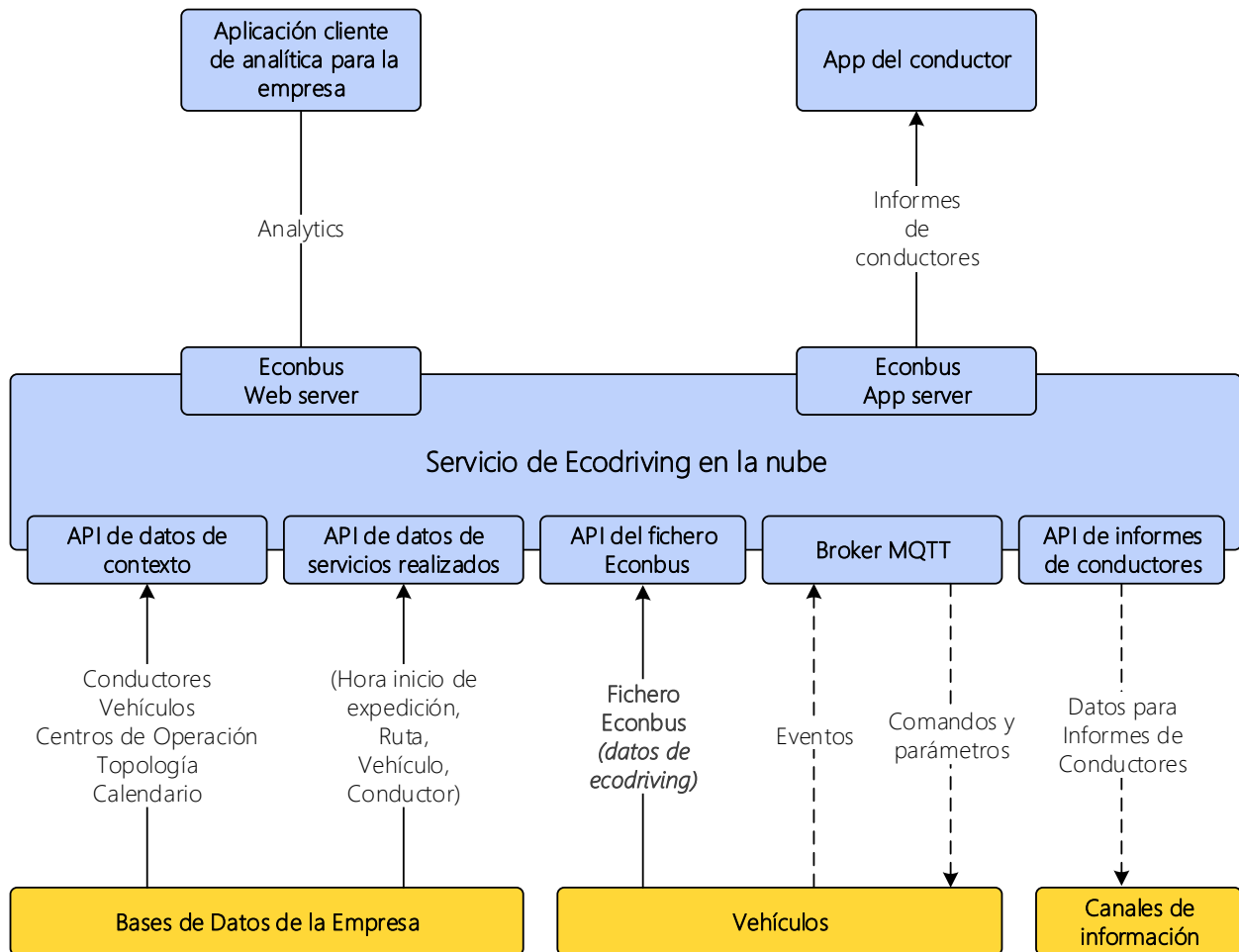


En suma, la arquitectura general de sistemas en la que se enmarca el servicio de ecodriving se muestra en el siguiente diagrama.

Los elementos en amarillo representan sistemas externos.



Las líneas discontinuas representan intercambios de datos no esenciales para la prestación del servicio.



Para las API de datos de contexto y servicios actualizados se propone la utilización de web services implementando los protocolos normalizados SIRI y GTFS.

Los datos de ecodriving se enviarán al servicio de ecodriving mediante ficheros, según se describe en E9.

Para el intercambio de información en tiempo real con los vehículos se propone la utilización de una arquitectura de comunicaciones IoT basada en un bróker de mensajes y un modo de comunicaciones basado en la publicación y suscripción. Estos datos no son imprescindibles para la arquitectura objeto de AMSARI pero se prevé que formen parte del servicio ecodriving.

La API de informes de conductores será un web service que publicará los datos para los informes de conductores en ficheros estructurados (uno por conductor). Su uso por parte de terceros es opcional ya que se prevé que una App para los conductores forme parte del servicio ofrecido.



### 3 PROTOCOLO IOT PARA UN SISTEMA ECODRIVING

Mediante este protocolo se reciben los datos de eventos de la conducción registrados por los dispositivos embarcados y que son necesarios para aplicar el método AMSARI. El protocolo también incluye mensajes para la configuración de dichos dispositivos y además permite el envío de datos del informe del conductor.

Se trata de un protocolo de la capa de aplicación basado en TCP/IP.

El protocolo está orientado a mensajes binarios cuya estructura general se muestra en la siguiente tabla.

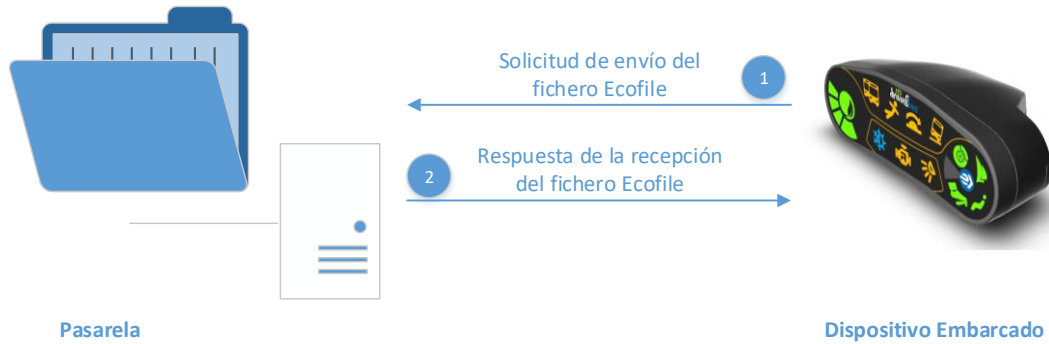
Campo	Tipo	Longitud	Descripción
messageType	unsigned byte	1	Tipo del mensaje
messageLength	unsigned int	4	Longitud del mensaje
payload	binary	messageLength	Payload del mensaje

Codificación de los tipos de mensajes:

Código	Tipo de Mensaje
0x00	FileTransferRequest
0x01	FileTransferResponse
0x02	ConfigurationRequest
0x03	ConfigurationResponse
0x04	DriverReportDataRequest
0x05	DriverReportDataResponse

#### 3.1 Transferir fichero Ecofile

La transferencia de ficheros Ecofile desde el dispositivo Embarcado al Servidor Pasarela se realiza mediante los mensajes se ilustran a continuación.



### Ilustración 1: Protocolo de transferencia del fichero Ecofile entre el dispositivo embarcado y el servidor pasarela

Inicialmente el dispositivo embarcado inicia la comunicación enviando al Servidor Pasarela un mensaje de solicitud de envío de un fichero Ecofile. Como respuesta al mensaje anterior el Servidor Pasarela envía un mensaje de respuesta informando la recepción correcta o no del fichero Ecofile.

A continuación se describe el formato de los mensajes de solicitud y respuesta.

#### 3.1.1 Mensaje FileTransferRequest

En la tabla se describen los campos del payload del mensaje FileTransferRequest.

Campo	Tipo	Longitud	Descripción
fileNameLength	unsigned byte	1	Longitud del nombre del fichero
fileName	string	fileNameLength	Nombre del fichero
contentLength	unsigned byte	4	Longitud del contenido del fichero
content	binary	messageLength - 1 - 1 - fileNameLength	Contenido del fichero
crc		4	CRC (Cyclic Redundancy Check) de 32 bit para detectar errores de transmisión.

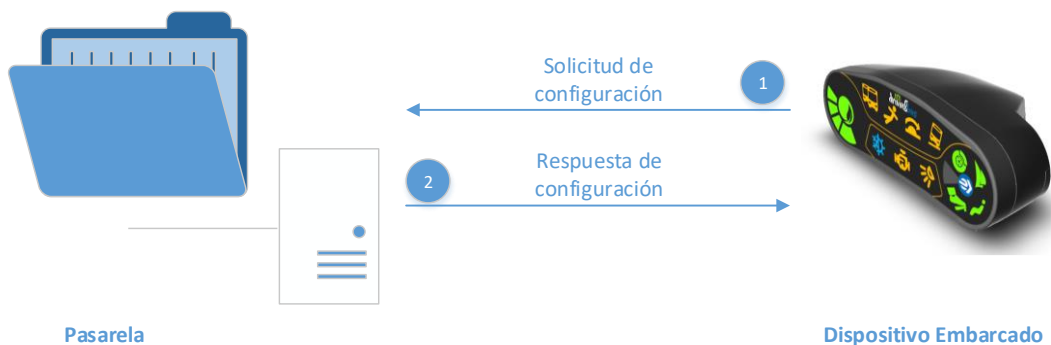
#### 3.1.2 Mensaje FileTransferResponse

En la tabla se describen los campos del payload del mensaje FileTransferResponse.

Campo	Tipo	Longitud	Descripción
fileNameLength	unsigned byte	1	Longitud del nombre del fichero
fileName	string	fileNameLength	Nombre del fichero
responseCode	unsigned byte	1	Código de respuesta del mensaje 0: OK 1: ERROR

### 3.2 Obtener datos de configuración

El envío de la configuración del dispositivo embarcado se realiza mediante los mensajes mostrados en el siguiente diagrama.



#### Ilustración 2: Protocolo para el envío de la configuración del dispositivo embarcado

En primer lugar el dispositivo embarcado inicia la comunicación enviando al servidor pasarela un mensaje de solicitud de configuración. Como respuesta al mensaje anterior el servidor pasarela envía un mensaje de respuesta con el contenido de parámetros de configuración.

A continuación se describe el formato de los mensajes de envío y respuesta.

#### 3.2.1 Mensaje ConfigurationRequest

En la tabla se describen los campos del payload del mensaje ConfigurationRequest.

Campo	Tipo	Longitud	Descripción
empresaId	unsigned byte	2	Identificador de la empresa
vehicleId	unsigned byte	2	Identificador del vehículo



deviceId	unsigned byte	2	Identificador del dispositivo
configurationVersion	unsigned byte	1	Versión de configuración actual

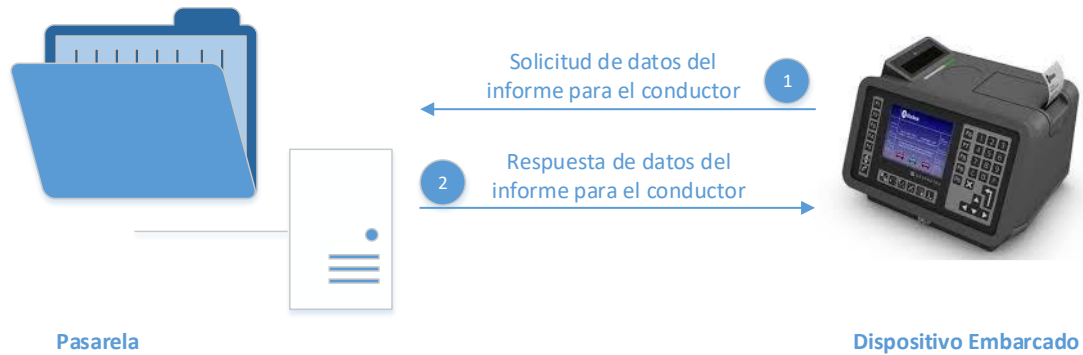
### 3.2.2 Mensaje ConfigurationResponse

En la tabla se describen los campos del payload del mensaje ConfigurationResponse.

Campo	Tipo	Longitud	Descripción
empresaId	unsigned byte	2	Identificador de la empresa
vehicleId	unsigned byte	2	Identificador del vehículo
deviceId	unsigned byte	2	Identificador del dispositivo
configurationVersion	unsigned byte	1	Versión de configuración
configurationLength	unsigned byte	2	Longitud del contenido de la configuración
Parameter1Length	unsigned byte	1	Longitud del parámetro 1
Parameter1Length	unsigned byte	1	Valor del parámetro 1
...			
ParameterNLength	unsigned byte	1	Longitud del parámetro n
ParameterNLength	unsigned byte	1	Valor del parámetro n

### 3.3 Obtener datos de informe para el conductor

La imagen a continuación muestra los mensajes intercambiados entre el dispositivo embarcado y la pasarela para la obtención de los datos del informe para el conductor.



### Ilustración 3: Protocolo para el envío de los datos del informe para el conductor

Primeramente el dispositivo embarcado inicia la comunicación enviando al servidor pasarela un mensaje de solicitud de datos del informe para el conductor. Como respuesta al mensaje anterior el servidor pasarela envía un mensaje de respuesta de dichos datos.

A continuación se describe el formato de los mensajes de envío y respuesta.

#### 3.3.1 Mensaje DriverReportDataRequest

En la tabla se describen los campos del payload del mensaje DriverReportDataRequest.

Campo	Tipo	Longitud	Descripción
empresaId	unsigned byte	2	Identificador de la empresa
driverId	unsigned byte	2	Identificador del conductor

#### 3.3.2 Mensaje DriverReportDataResponse

En la tabla se describen los campos del payload del mensaje DriverReportDataResponse.

Campo	Tipo	Longitud	Descripción
empresaId	unsigned byte	2	Identificador de la empresa
driverId	unsigned byte	2	Identificador del conductor
reportDataLength	unsigned byte	2	Longitud de los datos del informe
viajesLength	unsigned byte	1	Longitud del campo "viajes"



viajes	unsigned byte	viajesLength	Cantidad de viajes realizados en el periodo de datos analizado
fechaInicioLength	unsigned byte	1	Longitud del campo "fechaInicio"
fechaInicio	unsigned byte	fechaInicioLength	Fecha de inicio del periodo de datos analizado
fechaFinLength	unsigned byte	1	Longitud del campo "fechaFin"
fechaFin	unsigned byte	fechaFinLength	Fecha de finalización del periodo de datos analizado
frenadasLength	unsigned byte	1	Longitud del campo "frenadas"
frenadas	unsigned byte	frenadasLength	Nivel KPI Frenadas (0, 1, 2, 3, 4)
aceleracionesLength	unsigned byte	1	Longitud del campo "aceleraciones"
aceleraciones	unsigned byte	aceleracionesLength	Nivel KPI Aceleraciones (0, 1, 2, 3, 4)
inerciaLength	unsigned byte	1	Longitud del campo "inercia"
inercia	unsigned byte	inerciaLength	Nivel KPI Inercia (0, 1, 2, 3, 4)
confortLength	unsigned byte	1	Longitud del campo "confort"
confort	unsigned byte	confortLength	Nivel KPI Confort (0, 1, 2, 3, 4)
mínimoLength	unsigned byte	1	Longitud del campo "mínimo"
mínimo	unsigned byte	mínimoLength	Mínimo global (KPI peores conductores)
máximoLength	unsigned byte	1	Longitud del campo "máximo"
máximo	unsigned byte	máximoLength	Máximo global (KPI mejores conductores)



mediaLength	unsigned byte	1	Longitud del campo "media"
media	unsigned byte	mediaLength	Media global del KPI
propioLength	unsigned byte	1	Longitud del campo "propio"
propio	unsigned byte	propioLength	KPI Propio del conductor

## 4 API REST PARA OBTENER DATOS DE LA EMPRESA

La API REST de la pasarela contiene funcionalidades necesarias para obtener los datos de contexto de los sistemas externos. Estos datos son necesario para la definición de los escenarios de referencia y además sirven para la realización de estudios de los indicadores de la conducción, decisiones

### 4.1 Establecer datos de recursos

#### 4.1.1 Establecer datos de centros de operación

Este de método de la API de la pasarela implementa las funcionalidades para la recepción de los datos de los centros de operación.

**Nombre:** setOperationCentersData

**Parámetros de entrada:** Recibe una cadena de caracteres de un objeto JSON con los datos de los centros de operación.

**Parámetros de Salida:** Este método devuelve un objeto JSON con los parámetros:

- Código: Código del error. "0" indica que no hubo error, valores diferentes de "0" indican que ha ocurrido un error.
- Descripción: Contiene la descripción del error.

A continuación se define la estructura del objeto JSON recibido como parámetro de entrada.

```

{
  "empresa_id": "1",
  "fecha_actualizacion": "2019-02-19 00:03:25.000",
  "centros_operacion": [
    {
      "id": "1",
      "codigo": "00001",
      "nombre": "Centro de Operación 1"
    },
    {
      "id": "2",
      "codigo": "00002",
      "nombre": "Centro de Operación 2"
    }
  ]
}

```

**Ilustración 4: JSON del parámetro de entrada con datos de centros de operación**

En las tablas siguientes se describe los objetos contenidos en el JSON de datos de los centros de operación.

Empresa:

Atributo	Tipo	Null	Descripción
empresa_id	String	N	Identificador de la empresa en el sistema Econbus

Fecha de actualización:

Atributo	Tipo	Null	Descripción
fecha_actualizacion	String	N	Fecha de actualización de los datos

Centro de operación:

Atributo	Tipo	Null	Descripción
id	String	N	Identificador del centro de operación en el sistema externo
codigo	String	N	Código del centro de operación
nombre	String	S	Nombre del centro de operación



El siguiente cuadro de texto contiene el JSON con los parámetros de salida del método.

```
{
  "codigo": 0,
  "descripcion": ""
}
```

#### **Ilustración 5: JSON del parámetro de salida con datos de la respuesta del método**

En la siguiente tabla se describe el objeto contenido en el JSON del parámetro de salida.

Atributo	Tipo	Descripción
codigo	Entero	Código del error
descripcion	String	Descripción del error

### **4.1.2 Establecer datos de concesiones**

Este método de la API de la pasarela implementa las funcionalidades para la recepción de los datos de las concesiones.

**Nombre:** setConcessionsData

**Parámetros de entrada:** Recibe una cadena de caracteres de un objeto JSON con los datos de las concesiones.

**Parámetros de Salida:** Este método devuelve un objeto JSON con los parámetros:

- Código: Código del error. "0" indica que no hubo error, valores diferentes de "0" indican que ha ocurrido un error.
- Descripción: Contiene la descripción del error.

A continuación se define la estructura del objeto JSON recibido como parámetro de entrada.

```

{
  "empresa_id": "1",
  "fecha_actualizacion": "2019-02-19 00:03:25.000",
  "concesiones": [
    {
      "id": "1",
      "codigo": "00001",
      "nombre": "Concesión 1"
    },
    {
      "id": "2",
      "codigo": "00002",
      "nombre": "Concesión 2"
    }
  ]
}

```

**Ilustración 6: JSON del parámetro de entrada con datos de las concesiones**

En las tablas siguientes se describe los objetos contenidos en el JSON de datos de las concesiones.

Empresa:

Atributo	Tipo	Null	Descripción
empresa_id	String	N	Identificador de la empresa en el sistema Econbus

Fecha de actualización:

Atributo	Tipo	Null	Descripción
fecha_actualizacion	String	N	Fecha de actualización de los datos

Concesión:

Atributo	Tipo	Null	Descripción
id	String	N	Identificador la concesión en el sistema externo
codigo	String	N	Código de la concesión
nombre	String	S	Nombre de la concesión



El siguiente cuadro de texto contiene el JSON con los parámetros de salida del método.

```
{
  "codigo": 0,
  "descripcion": ""
}
```

### Ilustración 7: JSON del parámetro de salida con datos de la respuesta del método

En la siguiente tabla se describe el objeto contenido en el JSON del parámetro de salida.

Atributo	Tipo	Descripción
codigo	Entero	Código del error
descripcion	String	Descripción del error

#### 4.1.3 Establecer datos de conductores

Este método de la API de la pasarela implementa las funcionalidades para la recepción de los datos de los conductores.

**Nombre:** setDriversData

**Parámetros de entrada:** Recibe una cadena de caracteres de un objeto JSON con los datos de los conductores.

**Parámetros de Salida:** Este método devuelve un objeto JSON con los parámetros:

- Código: Código del error. "0" indica que no hubo error, valores diferentes de "0" indican que ha ocurrido un error.
- Descripción: Contiene la descripción del error.

A continuación se define la estructura del objeto JSON recibido como parámetro de entrada.

```

{
  "empresa_id": "1",
  "fecha_actualizacion": "2019-02-19 00:03:25.000",
  "conductores": [
    {
      "id": "1",
      "codigo": "1",
      "nombre": "Juana",
      "apellido1": "González",
      "apellido2": "González",
      "sexo": "F",
      "email": "pglez@empresa.es",
      "fecha_nacimiento": "1980-07-20",
      "fecha_alta": "2018-08-01",
      "fecha_baja": null,
      "fecha_formacion": "2018-10-01",
      "centro_operacion_id": "1"
    },
    {
      "id": "2",
      "nombre": "Pedro",
      "apellido1": "Fernández",
      "apellido2": "Fernández",
      "sexo": "M",
      "email": "jfdez@empresa.es",
      "fecha_nacimiento": "1950-05-25",
      "fecha_alta": "1980-04-01",
      "fecha_baja": "2019-02-18",
      "fecha_formacion": "2018-09-14",
      "centro_operacion_id": "2"
    }
  ]
}

```

**Ilustración 8: JSON del parámetro de entrada con datos de conductores**

En las tablas siguientes se describe los objetos contenidos en el JSON de datos de conductores.

Empresa:

Atributo	Tipo	Null	Descripción
empresa_id	String	N	Identificador de la empresa en el sistema Econbus

Fecha de actualización:

Atributo	Tipo	Null	Descripción
fecha_actualizacion	String	N	Fecha de actualización de los datos

Conductor:



Atributo	Tipo	Null	Descripción
id	String	N	Identificador del conductor en el sistema externo
codigo	String	N	Código del conductor
nombre	String	S	Nombre
apellido1	String	S	Primer apellido
apellido2	String	S	Segundo apellido
sexo	String	S	Sexo ("M", "F")
fecha_nacimiento	String	S	Fecha de nacimiento
fecha_alta	String	S	Fecha de alta
fecha_baja	String	S	Fecha de baja
fecha_formacion	String	S	Fecha de formación
centro_operacion_id	String	N	Identificador del centro de operación en el sistema externo

El siguiente cuadro de texto contiene el JSON con los parámetros de salida del método.

```
{
  "codigo": 0,
  "descripcion": ""
}
```

**Ilustración 9: JSON del parámetro de salida con datos de la respuesta del método**

En la siguiente tabla se describe el objeto contenido en el JSON del parámetro de salida.

Atributo	Tipo	Descripción
codigo	Entero	Código del error
descripcion	String	Descripción del error

#### 4.1.4 Establecer datos de modelos de vehículos

Este método de la API de la pasarela implementa las funcionalidades para la recepción de los datos de los modelos de vehículos.

**Nombre:** setVehicleModelsData

**Parámetros de entrada:** Recibe una cadena de caracteres de un objeto JSON con los datos de los modelos de vehículos.

**Parámetros de Salida:** Este método devuelve un objeto JSON con los parámetros:

- **Código:** Código del error. "0" indica que no hubo error, valores diferentes de "0" indican que ha ocurrido un error.
- **Descripción:** Contiene la descripción del error.

A continuación se define la estructura del objeto JSON recibido como parámetro de entrada.

```

{
  "empresa_id": "1",
  "fecha_actualizacion": "2019-02-19 00:03:25.000",
  "modelos": [
    {
      "id": "1",
      "codigo": "IRIZAR.I2E.1",
      "marca_codigo": "IRIZAR",
      "tipo_combustible_codigo": "ELECTRICO"
    },
    {
      "id": "2",
      "codigo": "MERCEDES.CIT-GNC.4",
      "marca_codigo": "MERCEDES",
      "tipo_combustible_codigo": "GNC"
    }
  ]
}

```

#### Ilustración 10: JSON del parámetro de entrada con datos de los modelos de vehículos

En las tablas siguientes se describe los objetos contenidos en el JSON de datos de vehículos.

Empresa:

Atributo	Tipo	Null	Descripción
empresa_id	String	N	Identificador de la empresa en el sistema Econbus

Fecha de actualización:



Atributo	Tipo	Null	Descripción
fecha_actualizacion	String	N	Fecha de actualización de los datos

Modelo de vehículo:

Atributo	Tipo	Null	Descripción
id	String	N	Identificador del modelo en el sistema externo
codigo	String	N	Código del modelo
marca_codigo	String	N	Código de la marca en el sistema Econbus
tipo_combustible_codigo	String	N	Código del tipo de combustible primario en sistema Econbus ('ELECTRICO', 'GNC')

El siguiente cuadro de texto contiene el JSON con los parámetros de salida del método.

```
{
  "codigo": 0,
  "descripcion": ""
}
```

#### **Ilustración 11: JSON del parámetro de salida con datos de la respuesta del método**

En la siguiente tabla se describe el objeto contenido en el JSON del parámetro de salida.

Atributo	Tipo	Descripción
codigo	Entero	Código del error
descripcion	String	Descripción del error

### **4.1.5 Establecer datos de vehículos**

Este método de la API de la pasarela implementa las funcionalidades para la recepción de los datos de los vehículos.

**Nombre:** setVehiclesData



**Parámetros de entrada:** Recibe una cadena de caracteres de un objeto JSON con los datos de los vehículos.

**Parámetros de Salida:** Este método devuelve un objeto JSON con los parámetros:

- Código: Código del error. "0" indica que no hubo error, valores diferentes de "0" indican que ha ocurrido un error.
- Descripción: Contiene la descripción del error.

A continuación se define la estructura del objeto JSON recibido como parámetro de entrada.

```

{
  "empresa_id": "1",
  "fecha_actualizacion": "2019-02-19 00:03:25.000",
  "vehiculos": [
    {
      "id": "1",
      "codigo": "00001",
      "matricula": "1234ABC",
      "fecha_matriculacion": "18/12/2017 0:00:00",
      "fecha_alta": "02/02/2018 0:00:00",
      "fecha_baja": null,
      "modelo_id": "1"
    },
    {
      "id": "2",
      "codigo": "00002",
      "matricula": "5678DEF",
      "fecha_matriculacion": "06/03/2018 0:00:00",
      "fecha_alta": "02/02/2018 0:00:00",
      "fecha_baja": null,
      "modelo_id": "2"
    }
  ]
}

```

**Ilustración 12: JSON del parámetro de entrada con datos de vehículos**

En las tablas siguientes se describe los objetos contenidos en el JSON de datos de vehículos.

Empresa:

Atributo	Tipo	Null	Descripción
empresa_id	String	N	Identificador de la empresa en el sistema Econbus

Fecha de actualización:

Atributo	Tipo	Null	Descripción
----------	------	------	-------------



fecha_actualizacion	String	N	Fecha de actualización de los datos
---------------------	--------	---	-------------------------------------

Vehículo:

Atributo	Tipo	Null	Descripción
id	String	N	Identificador del vehículo en el sistema externo
codigo	String	N	Código o número del vehículo
matricula	String	N	Matrícula
fecha_matriculacion	String	N	Fecha de matriculación
fecha_alta	String	N	Fecha de alta
fecha_baja	String	Y	Fecha de baja
modelo_id	String	N	Identificador del modelo de vehículo en el sistema externo

El siguiente cuadro de texto contiene el JSON con los parámetros de salida del método.

```
{
  "codigo": 0,
  "descripcion": ""
}
```

**Ilustración 13: JSON del parámetro de salida con datos de la respuesta del método**

En la siguiente tabla se describe el objeto contenido en el JSON del parámetro de salida.

Atributo	Tipo	Descripción
codigo	Entero	Código del error
descripcion	String	Descripción del error



#### 4.1.6 Establecer datos de asignaciones de vehículos

Este método de la API de la pasarela implementa las funcionalidades para la recepción de los datos de asignaciones de vehículos a centros de operación.

**Nombre:** setVehicleAssignmentsData

**Parámetros de entrada:** Recibe una cadena de caracteres de un objeto JSON con los datos de asignaciones de vehículos a centros de operación.

**Parámetros de Salida:** Este método devuelve un objeto JSON con los parámetros:

- **Código:** Código del error. "0" indica que no hubo error, valores diferentes de "0" indican que ha ocurrido un error.
- **Descripción:** Contiene la descripción del error.

A continuación se define la estructura del objeto JSON recibido como parámetro de entrada.

```

{
  "empresa_id": "1",
  "fecha_actualizacion": "2019-02-19 00:03:25.000",
  "asignaciones": [
    {
      "vehiculo_id": "1",
      "centro_operacion_id": "10000",
      "fecha_ini_asignacion": "02/02/2018 0:00:00",
      "fecha_fin_asignacion": "31/12/9999 0:00:00"
    },
    {
      "vehiculo_id": "2",
      "centro_operacion_id": "10000",
      "fecha_ini_asignacion": "12/06/2018 0:00:00",
      "fecha_fin_asignacion": "31/12/9999 0:00:00"
    }
  ]
}

```

**Ilustración 14: JSON del parámetro de entrada con datos de asignaciones de vehículos a centros de operación**

En las tablas siguientes se describe los objetos contenidos en el JSON de datos de asignaciones vehículos a centros de operación.

Empresa:

Atributo	Tipo	Null	Descripción
empresa_id	String	N	Identificador de la empresa en el sistema Econbus

Fecha de actualización:



Atributo	Tipo	Null	Descripción
fecha_actualizacion	String	N	Fecha de actualización de los datos

Asignación:

Atributo	Tipo	Null	Descripción
vehiculo_id	String	N	Identificador del vehículo en el sistema externo
centro_operacion_id	String	N	Identificador del centro de operación en el sistema externo
fecha_ini_asignacion	String	N	Fecha inicial de la asignación
fecha_fin_asignacion	String	Y	Fecha final de la asignación. El valor por defecto es 31/12/9999 0:00:00

El siguiente cuadro de texto contiene el JSON con los parámetros de salida del método.

```
{
  "codigo": 0,
  "descripcion": ""
}
```

**Ilustración 15: JSON del parámetro de salida con datos de la respuesta del método**

En la siguiente tabla se describe el objeto contenido en el JSON del parámetro de salida.

Atributo	Tipo	Descripción
codigo	Entero	Código del error
descripcion	String	Descripción del error



## 4.2 Establecer datos de calendario

### 4.2.1 Establecer datos de grupos de calendario

Un grupo de calendario se define como una agrupación de líneas que comparten un mismo calendario.

Este método de la API de la pasarela implementa las funcionalidades para la recepción de los datos de los grupos de calendario.

**Nombre:** setCalendarGroupsData

**Parámetros de entrada:** Recibe una cadena de caracteres de un objeto JSON con los datos de los grupos de calendario.

**Parámetros de Salida:** Este método devuelve un objeto JSON con los parámetros:

- **Código:** Código del error. "0" indica que no hubo error, valores diferentes de "0" indican que ha ocurrido un error.
- **Descripción:** Contiene la descripción del error.

A continuación se define la estructura del objeto JSON recibido como parámetro de entrada.

```
{
  "empresa_id": "1",
  "fecha_actualizacion": "2019-02-19 00:03:25.000",
  "grupos_calendario": [
    {
      "id": "1",
      "codigo": "00001",
      "nombre": "Grupo Calendario 1"
    },
    {
      "id": "2",
      "codigo": "00002",
      "nombre": "Grupo Calendario 2"
    }
  ]
}
```

#### **Ilustración 16: JSON del parámetro de entrada con datos de grupos de calendario**

En las tablas siguientes se describe los objetos contenidos en el JSON de datos de los grupos de calendario.



Empresa:

Atributo	Tipo	Null	Descripción
empresa_id	String	N	Identificador de la empresa en el sistema Econbus

Fecha de actualización:

Atributo	Tipo	Null	Descripción
fecha_actualizacion	String	N	Fecha de actualización de los datos

Grupo Calendario:

Atributo	Tipo	Null	Descripción
id	String	N	Identificador del grupo calendario en el sistema externo
codigo	String	N	Código del grupo calendario
nombre	String	N	Nombre del grupo calendario

El siguiente cuadro de texto contiene el JSON con los parámetros de salida del método.

```
{
  "codigo": 0,
  "descripcion": ""
}
```

### **Ilustración 17: JSON del parámetro de salida con datos de la respuesta del método**

En la siguiente tabla se describe el objeto contenido en el JSON del parámetro de salida.

Atributo	Tipo	Descripción
codigo	Entero	Código del error
descripcion	String	Descripción del error



## 4.2.2 Establecer datos de temporadas

Este método de la API de la pasarela implementa las funcionalidades para la recepción de los datos de las temporadas.

**Nombre:** setSeasonsData

**Parámetros de entrada:** Recibe una cadena de caracteres de un objeto JSON con los datos de las temporadas.

**Parámetros de Salida:** Este método devuelve un objeto JSON con los parámetros:

- **Código:** Código del error. "0" indica que no hubo error, valores diferentes de "0" indican que ha ocurrido un error.
- **Descripción:** Contiene la descripción del error.

A continuación se define la estructura del objeto JSON recibido como parámetro de entrada.

```

{
  "empresa_id": "1",
  "fecha_actualizacion": "2019-02-19 00:03:25.000",
  "temporadas": [
    {
      "id": "1",
      "codigo": "IV",
      "nombre": "Invierno"
    },
    {
      "id": "2",
      "codigo": "VN",
      "nombre": "Verano"
    }
  ]
}

```

### Ilustración 18: JSON del parámetro de entrada con datos de las temporadas

En las tablas siguientes se describe los objetos contenidos en el JSON de datos de las temporadas.

Empresa:

Atributo	Tipo	Null	Descripción
empresa_id	String	N	Identificador de la empresa en el sistema Econbus

Fecha de actualización:

Atributo	Tipo	Null	Descripción
fecha_actualizacion	String	N	Fecha de actualización de los datos



Temporada:

Atributo	Tipo	Null	Descripción
id	String	N	Identificador del grupo calendario en el sistema externo
codigo	String	N	Código del grupo calendario
nombre	String	N	Nombre del grupo calendario

El siguiente cuadro de texto contiene el JSON con los parámetros de salida del método.

```
{
  "codigo": 0,
  "descripcion": ""
}
```

#### **Ilustración 19: JSON del parámetro de salida con datos de la respuesta del método**

En la siguiente tabla se describe el objeto contenido en el JSON del parámetro de salida.

Atributo	Tipo	Descripción
codigo	Entero	Código del error
descripcion	String	Descripción del error

### **4.2.3 Establecer datos de tipos de día de movimiento**

Este método de la API de la pasarela implementa las funcionalidades para la recepción de los datos de los tipos de día de movimiento.

**Nombre:** setDayTypeMovData

**Parámetros de entrada:** Recibe una cadena de caracteres de un objeto JSON con los datos de los tipos de día de movimiento.

**Parámetros de Salida:** Este método devuelve un objeto JSON con los parámetros:

- Código: Código del error. "0" indica que no hubo error, valores diferentes de "0" indican que ha ocurrido un error.
- Descripción: Contiene la descripción del error.



A continuación se define la estructura del objeto JSON recibido como parámetro de entrada.

```

{
  "empresa_id": "1",
  "fecha_actualizacion": "2019-02-19 00:03:25.000",
  "tipos_dia_movimiento": [
    {
      "id": "1",
      "codigo": "LA",
      "nombre": "Laborable"
    },
    {
      "id": "2",
      "codigo": "SA",
      "nombre": "Sábado"
    },
    {
      "id": "3",
      "codigo": "FE",
      "nombre": "Domingos y Festivos"
    }
  ]
}

```

**Ilustración 20: JSON del parámetro de entrada con datos de los tipos de día de movimiento**

En las tablas siguientes se describe los objetos contenidos en el JSON de datos de los tipos de día de movimiento.

Empresa:

Atributo	Tipo	Null	Descripción
empresa_id	String	N	Identificador de la empresa en el sistema Econbus

Fecha de actualización:

Atributo	Tipo	Null	Descripción
fecha_actualizacion	String	N	Fecha de actualización de los datos

Tipo de día movimiento:

Atributo	Tipo	Null	Descripción
id	String	N	Identificador del tipo de día de movimiento en el



			sistema externo
codigo	String	N	Código del tipo de día de movimiento
nombre	String	N	Nombre del tipo de día de movimiento

El siguiente cuadro de texto contiene el JSON con los parámetros de salida del método.

```
{
  "codigo": 0,
  "descripcion": ""
}
```

**Ilustración 21: JSON del parámetro de salida con datos de la respuesta del método**

En la siguiente tabla se describe el objeto contenido en el JSON del parámetro de salida.

Atributo	Tipo	Descripción
codigo	Entero	Código del error
descripcion	String	Descripción del error

#### 4.2.4 Establecer datos de calendario

Este método de la API de la pasarela implementa las funcionalidades para la recepción de los datos del calendario.

**Nombre:** setCalendarData

**Parámetros de entrada:** Recibe una cadena de caracteres de un objeto JSON con los datos del calendario.

**Parámetros de Salida:** Este método devuelve un objeto JSON con los parámetros:

- Código: Código del error. "0" indica que no hubo error, valores diferentes de "0" indican que ha ocurrido un error.
- Descripción: Contiene la descripción del error.

A continuación se define la estructura del objeto JSON recibido como parámetro de entrada.



```
{
  "empresa_id": "1",
  "fecha_actualizacion": "2019-01-07 00:00:00.000",
  "calendario": [
    {
      "id": "1",
      "temporada_id": "1",
      "tipo_dia_mov_id": "3",
      "grupo_calendario_id": "1",
      "dia_servicio": "2019-01-01"
    },
    {
      "id": "2",
      "temporada_id": "1",
      "tipo_dia_mov_id": "1",
      "grupo_calendario_id": "1",
      "dia_servicio": "2019-01-02"
    },
    {
      "id": "1",
      "temporada_id": "3",
      "tipo_dia_mov_id": "1",
      "grupo_calendario_id": "1",
      "dia_servicio": "2019-01-03"
    },
    {
      "id": "1",
      "temporada_id": "4",
      "tipo_dia_mov_id": "1",
      "grupo_calendario_id": "1",
      "dia_servicio": "2019-01-04"
    },
    {
      "id": "1",
      "temporada_id": "5",
      "tipo_dia_mov_id": "2",
      "grupo_calendario_id": "1",
      "dia_servicio": "2019-01-05"
    },
    {
      "id": "1",
      "temporada_id": "6",
      "tipo_dia_mov_id": "3",
      "grupo_calendario_id": "1",
      "dia_servicio": "2019-01-06"
    },
    {
      "id": "1",
      "temporada_id": "7",
      "tipo_dia_mov_id": "1",
      "grupo_calendario_id": "1",
      "dia_servicio": "2019-01-07"
    }
  ]
}
```

**Ilustración 22: JSON del parámetro de entrada con datos del calendario**

En las tablas siguientes se describe los objetos contenidos en el JSON de datos del calendario.

Empresa:



Atributo	Tipo	Null	Descripción
empresa_id	String	N	Identificador de la empresa en el sistema Econbus

Fecha de actualización:

Atributo	Tipo	Null	Descripción
fecha_actualizacion	String	N	Fecha de actualización de los datos

Calendario:

Atributo	Tipo	Null	Descripción
id	String	N	Identificador del calendario en el sistema externo
temporada_id	String	N	Identificador de la temporada en el sistema externo
tipo_dia_mov_id	String	N	Identificador del tipo de día de movimiento en el sistema externo
grupo_calendario_id	String	N	Identificador del grupo de calendario en el sistema externo
día_servicio	String	N	Día de servicio

El siguiente cuadro de texto contiene el JSON con los parámetros de salida del método.

```
{
  "código": 0,
  "descripcion": ""
}
```

### **Ilustración 23: JSON del parámetro de salida con datos de la respuesta del método**

En la siguiente tabla se describe el objeto contenido en el JSON del parámetro de salida.

Atributo	Tipo	Descripción
codigo	Entero	Código del error
descripcion	String	Descripción del error



## 4.3 Establecer datos de períodos de demanda

### 4.3.1 Establecer datos de grupos de franjas horarias

Un grupo de franja horaria se define como una agrupación de líneas que comparten periodos de demanda.

Este método de la API de la pasarela implementa las funcionalidades para la recepción de los datos de los grupos de franjas horarias.

**Nombre:** setTimeSlotGroupsData

**Parámetros de entrada:** Recibe una cadena de caracteres de un objeto JSON con los datos de los grupos de franjas horarias.

**Parámetros de Salida:** Este método devuelve un objeto JSON con los parámetros:

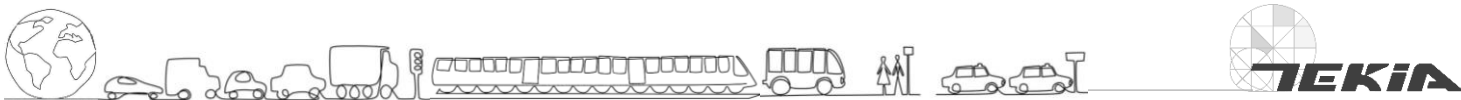
- Código: Código del error. "0" indica que no hubo error, valores diferentes de "0" indican que ha ocurrido un error.
- Descripción: Contiene la descripción del error.

A continuación se define la estructura del objeto JSON recibido como parámetro de entrada.

```
{
  "empresa_id": "1",
  "fecha_actualizacion": "2019-02-19 00:03:25.000",
  "grupos_franja_horaria": [
    {
      "id": "1",
      "codigo": "00001",
      "nombre": "Red diurna convencional"
    },
    {
      "id": "2",
      "codigo": "00002",
      "nombre": "Red nocturna"
    },
    {
      "id": "3",
      "codigo": "00003",
      "nombre": "Red universitaria"
    }
  ]
}
```

#### **Ilustración 24: JSON del parámetro de entrada con datos de grupos de franjas horarias**

En las tablas siguientes se describe los objetos contenidos en el JSON de datos de los grupos de franjas horarias.



Empresa:

Atributo	Tipo	Null	Descripción
empresa_id	String	N	Identificador de la empresa en el sistema Econbus

Fecha de actualización:

Atributo	Tipo	Null	Descripción
fecha_actualizacion	String	N	Fecha de actualización de los datos

Grupo de franjas horarias:

Atributo	Tipo	Null	Descripción
id	String	N	Identificador del grupo de franja horaria en el sistema externo
codigo	String	N	Código del grupo de franja horaria
nombre	String	N	Nombre del grupo de franja horaria

El siguiente cuadro de texto contiene el JSON con los parámetros de salida del método.

```
{
  "codigo": 0,
  "descripcion": ""
}
```

**Ilustración 25: JSON del parámetro de salida con datos de la respuesta del método**

En la siguiente tabla se describe el objeto contenido en el JSON del parámetro de salida.

Atributo	Tipo	Descripción
codigo	Entero	Código del error
descripcion	String	Descripción del error



### 4.3.2 Establecer datos de franjas horarias

Este método de la API de la pasarela implementa las funcionalidades para la recepción de los datos de franjas horarias.

**Nombre:** setTimeSlotData

**Parámetros de entrada:** Recibe una cadena de caracteres de un objeto JSON con los datos de franjas horarias.

**Parámetros de Salida:** Este método devuelve un objeto JSON con los parámetros:

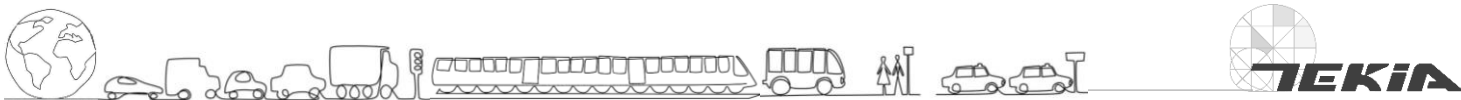
- Código: Código del error. "0" indica que no hubo error, valores diferentes de "0" indican que ha ocurrido un error.
- Descripción: Contiene la descripción del error.

A continuación se define la estructura del objeto JSON recibido como parámetro de entrada.

```
{
  "empresa_id": "1",
  "fecha_actualizacion": "2019-02-19 00:03:25.000",
  "franjas_horarias": [
    {
      "id": "1",
      "codigo": "00001",
      "nombre": "Franja Valle de Mañana"
    },
    {
      "id": "2",
      "codigo": "00002",
      "nombre": "Franja Punta Mediodía"
    },
    {
      "id": "3",
      "codigo": "00003",
      "nombre": "Franja Valle de Tarde"
    },
    {
      "id": "4",
      "codigo": "00004",
      "nombre": "Franja Punta de Tarde"
    }
  ]
}
```

#### **Ilustración 26: JSON del parámetro de entrada con datos de grupos de franjas horarias**

En las tablas siguientes se describe los objetos contenidos en el JSON de datos de las franjas horarias.



Empresa:

Atributo	Tipo	Null	Descripción
empresa_id	String	N	Identificador de la empresa en el sistema Econbus

Fecha de actualización:

Atributo	Tipo	Null	Descripción
fecha_actualizacion	String	N	Fecha de actualización de los datos

Franjas horarias:

Atributo	Tipo	Null	Descripción
id	String	N	Identificador de la franja horaria en el sistema externo
codigo	String	N	Código de la franja horaria
nombre	String	N	Nombre de la franja horaria

El siguiente cuadro de texto contiene el JSON con los parámetros de salida del método.

```
{
  "codigo": 0,
  "descripcion": ""
}
```

**Ilustración 27: JSON del parámetro de salida con datos de la respuesta del método**

En la siguiente tabla se describe el objeto contenido en el JSON del parámetro de salida.

Atributo	Tipo	Descripción
codigo	Entero	Código del error
descripcion	String	Descripción del error



### 4.3.3 Establecer datos de periodos de demanda

Este método de la API de la pasarela implementa las funcionalidades para la recepción de los datos de periodos de demanda.

**Nombre:** setDemandPeriodsData

**Parámetros de entrada:** Recibe una cadena de caracteres de un objeto JSON con los datos de periodos de demanda.

**Parámetros de Salida:** Este método devuelve un objeto JSON con los parámetros:

- **Código:** Código del error. "0" indica que no hubo error, valores diferentes de "0" indican que ha ocurrido un error.
- **Descripción:** Contiene la descripción del error.

A continuación se define la estructura del objeto JSON recibido como parámetro de entrada.

```

{
  "empresa_id": "1",
  "fecha_actualizacion": "2019-01-07 00:00:00.000",
  "periodos_demanda": [
    {
      "id": "1",
      "temporada_id": "1",
      "tipo_dia_mov_id": "1",
      "grupo_franja_horaria_id": "1",
      "franja_horaria_id": "1",
      "tiempo_desde": 36000,
      "tiempo_hasta": 46800
    },
    {
      "id": "1",
      "temporada_id": "1",
      "tipo_dia_mov_id": "1",
      "grupo_franja_horaria_id": "1",
      "franja_horaria_id": "2",
      "tiempo_desde": 46800,
      "tiempo_hasta": 57600
    },
    {
      "id": "1",
      "temporada_id": "1",
      "tipo_dia_mov_id": "1",
      "grupo_franja_horaria_id": "1",
      "franja_horaria_id": "3",
      "tiempo_desde": 57600,
      "tiempo_hasta": 61200
    },
    {
      "id": "1",
      "temporada_id": "1",
      "tipo_dia_mov_id": "1",
      "grupo_franja_horaria_id": "1",
      "franja_horaria_id": "4",
      "tiempo_desde": 61200,
      "tiempo_hasta": 72000
    }
  ]
}

```

**Ilustración 28: JSON del parámetro de entrada con datos de periodos de demanda**

En las tablas siguientes se describe los objetos contenidos en el JSON de datos de periodos de demanda

Empresa:

Atributo	Tipo	Null	Descripción
empresa_id	String	N	Identificador de la empresa en el sistema Econbus

Fecha de actualización:



Atributo	Tipo	Null	Descripción
fecha_actualizacion	String	N	Fecha de actualización de los datos

Periodos de demanda:

Atributo	Tipo	Null	Descripción
id	String	N	Identificador del periodo de demanda en el sistema externo
temporada_id	String	N	Identificador de la temporada en el sistema externo
tipo_dia_mov_id	String	N	Identificador del tipo de día de movimiento en el sistema externo
grupo_franja_horaria_id	String	N	Identificador del grupo de franja horaria en el sistema externo
franja_horaria_id	String	N	Identificador de la franja horaria en el sistema externo
tiempo_desde	Entero	N	Tiempo inicial expresado en segundos transcurridos desde las 00:00:00 horas del día de servicio
tiempo_hasta	Entero	N	Tiempo final expresado en segundos transcurridos desde las 00:00:00 horas del día de servicio

El siguiente cuadro de texto contiene el JSON con los parámetros de salida del método.

```
{
  "codigo": 0,
  "descripcion": ""
}
```

**Ilustración 29: JSON del parámetro de salida con datos de la respuesta del método**

En la siguiente tabla se describe el objeto contenido en el JSON del parámetro de salida.

Atributo	Tipo	Descripción
codigo	Entero	Código del error



descripcion	String	Descripción del error
-------------	--------	-----------------------

#### 4.4 Establecer datos de franjas horarias naturales

Este método de la API de la pasarela implementa las funcionalidades para la recepción de los datos de franjas horarias naturales.

**Nombre:** setNaturalTimeSlotsData

**Parámetros de entrada:** Recibe una cadena de caracteres de un objeto JSON con los datos de franjas horarias naturales.

**Parámetros de Salida:** Este método devuelve un objeto JSON con los parámetros:

- Código: Código del error. "0" indica que no hubo error, valores diferentes de "0" indican que ha ocurrido un error.
- Descripción: Contiene la descripción del error.

A continuación se define la estructura del objeto JSON recibido como parámetro de entrada.

```

{
  "empresa_id": "1",
  "fecha_actualizacion": "2019-01-07 00:00:00.000",
  "periodos_demanda": [
    {
      "id": "1",
      "codigo": "001",
      "nombre": "Franja horaria 00H de red diurna convencional",
      "grupo_franja_horaria_id": "1",
      "tiempo_desde": 0,
      "tiempo_hasta": 3599
    },
    {
      "id": "2",
      "codigo": "002",
      "nombre": "Franja horaria 01H de red diurna convencional",
      "grupo_franja_horaria_id": "1",
      "tiempo_desde": 3600,
      "tiempo_hasta": 7199
    },
    {
      "id": "3",
      "codigo": "003",
      "nombre": "Franja horaria 02H de red diurna convencional",
      "grupo_franja_horaria_id": "1",
      "tiempo_desde": 7200,
      "tiempo_hasta": 10799
    }
  ]
}

```

**Ilustración 30: JSON del parámetro de entrada con datos de periodos de demanda**



En las tablas siguientes se describe los objetos contenidos en el JSON de datos de franjas horarias naturales.

Empresa:

Atributo	Tipo	Null	Descripción
empresa_id	String	N	Identificador de la empresa en el sistema Econbus

Fecha de actualización:

Atributo	Tipo	Null	Descripción
fecha_actualizacion	String	N	Fecha de actualización de los datos

Franja horaria natural:

Atributo	Tipo	Null	Descripción
id	String	N	Identificador de la franja horaria natural en el sistema externo
codigo	String	N	Código de la franja horaria natural
nombre	String	N	Nombre de la franja horaria natural
grupo_franja_horaria_id	String	N	Identificador del grupo de franja horaria en el sistema externo
tiempo_desde	Entero	N	Tiempo inicial expresado en segundos transcurridos desde las 00:00:00 horas del día de servicio
tiempo_hasta	Entero	N	Tiempo final expresado en segundos transcurridos desde las 00:00:00 horas del día de servicio

El siguiente cuadro de texto contiene el JSON con los parámetros de salida del método.



```
{
  "código": 0,
  "descripcion": ""
}
```

**Ilustración 31: JSON del parámetro de salida con datos de la respuesta del método**

En la siguiente tabla se describe el objeto contenido en el JSON del parámetro de salida.

Atributo	Tipo	Descripción
codigo	Entero	Código del error
descripcion	String	Descripción del error

### 4.5 Establecer datos de la red topológica

Los siguientes métodos de la API de la pasarela implementan las funcionalidades para la recepción de los datos de la red topológica.

#### 4.5.1 Establecer datos de grupos de explotación

Este método de la API de la pasarela implementa las funcionalidades para la recepción de los datos de los grupos de explotación.

**Nombre:** setOperatingGroupsData

**Parámetros de entrada:** Recibe una cadena de caracteres de un objeto JSON con los datos de los grupos de explotación.

**Parámetros de Salida:** Este método devuelve un objeto JSON con los parámetros:

- Código: Código del error. "0" indica que no hubo error, valores diferentes de "0" indican que ha ocurrido un error.
- Descripción: Contiene la descripción del error.

A continuación se define la estructura del objeto JSON recibido como parámetro de entrada.

```

{
  "empresa_id": "1",
  "fecha_actualizacion": "2019-02-19 00:03:25.000",
  "grupos_explotacion": [
    {
      "id": "1",
      "codigo": "001",
      "nombre": "Red diurna convencional"
    },
    {
      "id": "2",
      "codigo": "002",
      "nombre": "Red nocturna"
    },
    {
      "id": "3",
      "codigo": "003",
      "nombre": "Red universitaria"
    }
  ]
}

```

**Ilustración 32: JSON del parámetro de entrada con datos de los grupos de explotación**

En las tablas siguientes se describe los objetos contenidos en el JSON de datos de los grupos de explotación.

Empresa:

Atributo	Tipo	Null	Descripción
empresa_id	String	N	Identificador de la empresa en el sistema Econbus

Fecha de actualización:

Atributo	Tipo	Null	Descripción
fecha_actualizacion	String	N	Fecha de actualización de los datos

Grupo de explotación:

Atributo	Tipo	Null	Descripción
id	String	N	Identificador del grupo de explotación en el sistema externo



codigo	String	N	Código del grupo de explotación
nombre	String	N	Nombre del grupo de explotación

El siguiente cuadro de texto contiene el JSON con los parámetros de salida del método.

```
{
  "codigo": 0,
  "descripcion": ""
}
```

**Ilustración 33: JSON del parámetro de salida con datos de la respuesta del método**

En la siguiente tabla se describe el objeto contenido en el JSON del parámetro de salida.

Atributo	Tipo	Descripción
codigo	Entero	Código del error
descripcion	String	Descripción del error

### 4.5.2 Establecer datos de subgrupos de explotación

Este método de la API de la pasarela implementa las funcionalidades para la recepción de los datos de los subgrupos de explotación.

**Nombre:** setOperatingSubgroupsData

**Parámetros de entrada:** Recibe una cadena de caracteres de un objeto JSON con los datos de los subgrupos de explotación.

**Parámetros de Salida:** Este método devuelve un objeto JSON con los parámetros:

- Código: Código del error. "0" indica que no hubo error, valores diferentes de "0" indican que ha ocurrido un error.
- Descripción: Contiene la descripción del error.

A continuación se define la estructura del objeto JSON recibido como parámetro de entrada.

```

{
  "empresa_id": "1",
  "fecha_actualizacion": "2019-02-19 00:03:25.000",
  "subgrupos_explotacion": [
    {
      "id": "1",
      "codigo": "001",
      "nombre": "Líneas Convencionales",
      "grupo_explotacion_id": "1"
    },
    {
      "id": "2",
      "codigo": "002",
      "nombre": "Líneas Minibuses",
      "grupo_explotacion_id": "1"
    },
    {
      "id": "3",
      "codigo": "003",
      "nombre": "Líneas Especiales",
      "grupo_explotacion_id": "1"
    }
  ]
}

```

**Ilustración 34: JSON del parámetro de entrada con datos de los subgrupos de explotación**

En las tablas siguientes se describe los objetos contenidos en el JSON de datos de los subgrupos de explotación.

Empresa:

Atributo	Tipo	Null	Descripción
empresa_id	String	N	Identificador de la empresa en el sistema Econbus

Fecha de actualización:

Atributo	Tipo	Null	Descripción
fecha_actualizacion	String	N	Fecha de actualización de los datos

Grupo de explotación:

Atributo	Tipo	Null	Descripción
----------	------	------	-------------

id	String	N	Identificador del grupo de explotación en el sistema externo
codigo	String	N	Código del grupo de explotación
nombre	String	N	Nombre del grupo de explotación
grupo_explotacion_id	String	N	Identificador del grupo de explotación en el sistema externo

El siguiente cuadro de texto contiene el JSON con los parámetros de salida del método.

```
{
  "codigo": 0,
  "descripcion": ""
}
```

**Ilustración 35: JSON del parámetro de salida con datos de la respuesta del método**

En la siguiente tabla se describe el objeto contenido en el JSON del parámetro de salida.

Atributo	Tipo	Descripción
codigo	Entero	Código del error
descripcion	String	Descripción del error

### 4.5.3 Establecer datos de líneas

Este método de la API de la pasarela implementa las funcionalidades para la recepción de los datos de las líneas.

**Nombre:** setLinesData

**Parámetros de entrada:** Recibe una cadena de caracteres de un objeto JSON con los datos de las líneas.

**Parámetros de Salida:** Este método devuelve un objeto JSON con los parámetros:

- Código: Código del error. "0" indica que no hubo error, valores diferentes de "0" indican que ha ocurrido un error.
- Descripción: Contiene la descripción del error.

A continuación se define la estructura del objeto JSON recibido como parámetro de entrada.

```

{
  "empresa_id": "1",
  "fecha_actualizacion": "2019-02-19 00:03:25.000",
  "lineas": [
    {
      "id": "1",
      "codigo": "L1",
      "nombre": "L1",
      "centro_operacion_id": "1",
      "subgrupo_explotacion_id": "1",
      "grupo_calendario": "1",
      "grupo_franja_horaria_id": "1",
      "concesion_id": null
    },
    {
      "id": "2",
      "codigo": "L2",
      "nombre": "L2",
      "centro_operacion_id": "1",
      "subgrupo_explotacion_id": "1",
      "grupo_calendario": "1",
      "grupo_franja_horaria_id": "1",
      "concesion_id": null
    }
  ]
}

```

**Ilustración 36: JSON del parámetro de entrada con datos de las líneas**

En las tablas siguientes se describe los objetos contenidos en el JSON de las líneas.

Empresa:

Atributo	Tipo	Null	Descripción
empresa_id	String	N	Identificador de la empresa en el sistema Econbus

Fecha de actualización:

Atributo	Tipo	Null	Descripción
fecha_actualizacion	String	N	Fecha de actualización de los datos

Línea:

Atributo	Tipo	Null	Descripción
id	String	N	Identificador de la línea en el sistema externo



codigo	String	N	Código de la línea
nombre	String	N	Nombre de la línea
centro_operacion_id	String	N	Identificador del centro de operación en el sistema externo
subgrupo_explotacion_id	String	N	Identificador del subgrupo de explotación en el sistema externo
grupo_calendario_id	String	N	Identificador del grupo calendario en el sistema externo
grupo_franja_horaria_id	String	N	Identificador del grupo de franja horaria en el sistema externo
concesion_id	String	Y	Identificador de la concesión en el sistema externo

El siguiente cuadro de texto contiene el JSON con los parámetros de salida del método.

```
{
  "codigo": 0,
  "descripcion": ""
}
```

**Ilustración 37: JSON del parámetro de salida con datos de la respuesta del método**

En la siguiente tabla se describe el objeto contenido en el JSON del parámetro de salida.

Atributo	Tipo	Descripción
codigo	Entero	Código del error
descripcion	String	Descripción del error

**4.5.4 Establecer datos de sublíneas**

Este de método de la API de la pasarela implementa las funcionalidades para la recepción de los datos de las sublíneas.

**Nombre:** setSublinesData



**Parámetros de entrada:** Recibe una cadena de caracteres de un objeto JSON con los datos de las sublíneas.

**Parámetros de Salida:** Este método devuelve un objeto JSON con los parámetros:

- Código: Código del error. "0" indica que no hubo error, valores diferentes de "0" indican que ha ocurrido un error.
- Descripción: Contiene la descripción del error.

A continuación se define la estructura del objeto JSON recibido como parámetro de entrada.

```

{
  "empresa_id": "1",
  "fecha_actualizacion": "2019-02-19 00:03:25.000",
  "sublineas": [
    {
      "id": "1",
      "codigo": "SL1",
      "nombre": "SL1",
      "linea_id": "1"
    },
    {
      "id": "2",
      "codigo": "SL2",
      "nombre": "SL2",
      "linea_id": "1"
    }
  ]
}

```

**Ilustración 38: JSON del parámetro de entrada con datos de las sublíneas**

En las tablas siguientes se describe los objetos contenidos en el JSON de la red topológica.

Empresa:

Atributo	Tipo	Null	Descripción
empresa_id	String	N	Identificador de la empresa en el sistema Econbus

Fecha de actualización:

Atributo	Tipo	Null	Descripción
fecha_actualizacion	String	N	Fecha de actualización de los datos

Sublínea:

Atributo	Tipo	Null	Descripción
id	String	N	Identificador de la sublínea en el sistema externo
codigo	String	N	Código de la sublínea
nombre	String	N	Nombre de la sublínea
linea_id	String	N	Identificador de la línea a la que pertenece la sublínea

El siguiente cuadro de texto contiene el JSON con los parámetros de salida del método.

```
{
  "codigo": 0,
  "descripcion": ""
}
```

**Ilustración 39: JSON del parámetro de salida con datos de la respuesta del método**

En la siguiente tabla se describe el objeto contenido en el JSON del parámetro de salida.

Atributo	Tipo	Descripción
codigo	Entero	Código del error
descripcion	String	Descripción del error

### 4.5.5 Establecer datos de rutas

Este método de la API de la pasarela implementa las funcionalidades para la recepción de los datos de las rutas.

**Nombre:** setRoutesData

**Parámetros de entrada:** Recibe una cadena de caracteres de un objeto JSON con los datos de las rutas.

**Parámetros de Salida:** Este método devuelve un objeto JSON con los parámetros:

- Código: Código del error. "0" indica que no hubo error, valores diferentes de "0" indican que ha ocurrido un error.
- Descripción: Contiene la descripción del error.

A continuación se define la estructura del objeto JSON recibido como parámetro de entrada.

```

{
  "empresa_id": "1",
  "fecha_actualizacion": "2019-02-19 00:03:25.000",
  "rutas": [
    {
      "id": "1",
      "codigo": "R1",
      "nombre": "R1",
      "sentido_codigo": "IDA",
      "sublinea_id": "1"
    },
    {
      "id": "2",
      "codigo": "R2",
      "nombre": "R2",
      "sentido_codigo": "VUELTA",
      "sublinea_id": "1"
    },
    {
      "id": "3",
      "codigo": "R3",
      "nombre": "R3",
      "sentido_codigo": "IDA",
      "sublinea_id": "2"
    },
    {
      "id": "4",
      "codigo": "R4",
      "nombre": "R4",
      "sentido_codigo": "VUELTA",
      "sublinea_id": "2"
    }
  ]
}

```

**Ilustración 40: JSON del parámetro de entrada con datos de las rutas**

En las tablas siguientes se describe los objetos contenidos en el JSON de las rutas.

Empresa:

Atributo	Tipo	Null	Descripción
empresa_id	String	N	Identificador de la empresa en el sistema Econbus

Fecha de actualización:

Atributo	Tipo	Null	Descripción
fecha_actualizacion	String	N	Fecha de actualización de los datos



Ruta:

Atributo	Tipo	Null	Descripción
id	String	N	Identificador de la ruta en el sistema externo
codigo	String	N	Código de la ruta
nombre	String	N	Nombre de la ruta
sentido_codigo	String	N	Código del sentido
sublinea_id	String	N	Identificador de la sublínea a la que pertenece la ruta

El siguiente cuadro de texto contiene el JSON con los parámetros de salida del método.

```
{
  "codigo": 0,
  "descripcion": ""
}
```

#### **Ilustración 41: JSON del parámetro de salida con datos de la respuesta del método**

En la siguiente tabla se describe el objeto contenido en el JSON del parámetro de salida.

Atributo	Tipo	Descripción
codigo	Entero	Código del error
descripcion	String	Descripción del error

### **4.5.6 Establecer datos de secuencia de paradas de rutas**

Este método de la API de la pasarela implementa las funcionalidades para la recepción de los datos de secuencia de paradas de rutas.

**Nombre:** setSequenceRouteStopsData

**Parámetros de entrada:** Recibe una cadena de caracteres de un objeto JSON con los datos de secuencia de paradas de rutas.

**Parámetros de Salida:** Este método devuelve un objeto JSON con los parámetros:

- **Código:** Código del error. "0" indica que no hubo error, valores diferentes de "0" indican que ha ocurrido un error.



- Descripción: Contiene la descripción del error.

A continuación se define la estructura del objeto JSON recibido como parámetro de entrada.

```

{
  "empresa_id": "1",
  "fecha_actualizacion": "2019-02-19 00:03:25.000",
  "secuencia_paradas_ruta": [
    { "ruta_id": "1", "parada_id": "1", "distancia": 0, "secuencia": 1, "tipo_parada": "1" },
    { "ruta_id": "1", "parada_id": "2", "distancia": 500, "secuencia": 2, "tipo_parada": "2" },
    { "ruta_id": "1", "parada_id": "3", "distancia": 1000, "secuencia": 3, "tipo_parada": "3" },

    { "ruta_id": "2", "parada_id": "3", "distancia": 0, "secuencia": 1, "tipo_parada": "1" },
    { "ruta_id": "2", "parada_id": "4", "distancia": 500, "secuencia": 2, "tipo_parada": "2" },
    { "ruta_id": "2", "parada_id": "1", "distancia": 1000, "secuencia": 3, "tipo_parada": "3"},

    { "ruta_id": "3", "parada_id": "1", "distancia": 0, "secuencia": 1, "tipo_parada": "1" },
    { "ruta_id": "3", "parada_id": "5", "distancia": 500, "secuencia": 2, "tipo_parada": "2" },
    { "ruta_id": "3", "parada_id": "3", "distancia": 1000, "secuencia": 3, "tipo_parada": "3" },

    { "ruta_id": "4", "parada_id": "3", "distancia": 0, "secuencia": 1, "tipo_parada": "1" },
    { "ruta_id": "4", "parada_id": "6", "distancia": 500, "secuencia": 2, "tipo_parada": "2" },
    { "ruta_id": "4", "parada_id": "1", "distancia": 1000, "secuencia": 3, "tipo_parada": "3"}
  ]
}

```

**Ilustración 42: JSON del parámetro de entrada con datos de secuencia de paradas de rutas**

En las tablas siguientes se describe los objetos contenidos en el JSON de secuencia de paradas de rutas.

Empresa:

Atributo	Tipo	Null	Descripción
empresa_id	String	N	Identificador de la empresa en el sistema Econbus

Fecha de actualización:

Atributo	Tipo	Null	Descripción
fecha_actualizacion	String	N	Fecha de actualización de los datos

Secuencia de paradas en ruta:

Atributo	Tipo	Null	Descripción
----------	------	------	-------------

ruta_id	String	N	Identificador de la ruta en el sistema externo
parada_id	String	N	Identificador de la parada en el sistema externo
distancia	Entero	N	Distancia de la parada desde el inicio de la ruta o parada inicial
secuencia	Númerico	N	Secuencia de la parada en la ruta
tipo_parada	String	N	Tipo de parada (Inicial, Intermedia, Final)

El siguiente cuadro de texto contiene el JSON con los parámetros de salida del método.

```
{
  "codigo": 0,
  "descripcion": ""
}
```

**Ilustración 43: JSON del parámetro de salida con datos de la respuesta del método**

En la siguiente tabla se describe el objeto contenido en el JSON del parámetro de salida.

Atributo	Tipo	Descripción
codigo	Entero	Código del error
descripcion	String	Descripción del error

### 4.5.7 Establecer datos de paradas

Este método de la API de la pasarela implementa las funcionalidades para la recepción de los datos de las paradas.

**Nombre:** seStopsData

**Parámetros de entrada:** Recibe una cadena de caracteres de un objeto JSON con los datos de las paradas.

**Parámetros de Salida:** Este método devuelve un objeto JSON con los parámetros:

- Código: Código del error. "0" indica que no hubo error, valores diferentes de "0" indican que ha ocurrido un error.
- Descripción: Contiene la descripción del error.

A continuación se define la estructura del objeto JSON recibido como parámetro de entrada.



```

{
  "empresa_id": "1",
  "fecha_actualizacion": "2019-02-19 00:03:25.000",
  "paradas": [
    { "id": "1", "codigo": "P1", "nombre": "P1", "coord_latitud": 0.000011, "coord_longitud": 0.000011 },
    { "id": "2", "codigo": "P2", "nombre": "P2", "coord_latitud": 0.000022, "coord_longitud": 0.000022 },
    { "id": "3", "codigo": "P3", "nombre": "P3", "coord_latitud": 0.000033, "coord_longitud": 0.000033 },
    { "id": "4", "codigo": "P4", "nombre": "P4", "coord_latitud": 0.000044, "coord_longitud": 0.000044 },
    { "id": "5", "codigo": "P5", "nombre": "P5", "coord_latitud": 0.000055, "coord_longitud": 0.000055 },
    { "id": "6", "codigo": "P6", "nombre": "P6", "coord_latitud": 0.000066, "coord_longitud": 0.000066 }
  ]
}

```

**Ilustración 44: JSON del parámetro de entrada con datos de las paradas**

Absolver

En las tablas siguientes se describe los objetos contenidos en el JSON de las paradas.

Empresa:

Atributo	Tipo	Null	Descripción
empresa_id	String	N	Identificador de la empresa en el sistema Econbus

Fecha de actualización:

Atributo	Tipo	Null	Descripción
fecha_actualizacion	String	N	Fecha de actualización de los datos

Parada:

Atributo	Tipo	Null	Descripción
id	String	N	Identificador de la parada en el sistema externo
codigo	String	N	Código de la parada
nombre	String	N	Nombre de la parada
coord_latitud	Númerico	N	Coordenada latitud
coord_longitud	Númerico	N	Coordenada longitud

El siguiente cuadro de texto contiene el JSON con los parámetros de salida del método.



```
{
  "código": 0,
  "descripcion": ""
}
```

**Ilustración 45: JSON del parámetro de salida con datos de la respuesta del método**

En la siguiente tabla se describe el objeto contenido en el JSON del parámetro de salida.

Atributo	Tipo	Descripción
codigo	Entero	Código del error
descripcion	String	Descripción del error

### 4.6 Establecer datos de expediciones realizadas

Este método de la API de la pasarela implementa las funcionalidades para la recepción de los datos de expediciones realizadas.

**Nombre:** setTripsData

**Parámetros de entrada:** Recibe una cadena de caracteres de un objeto JSON con los datos de las expediciones realizadas.

**Parámetros de Salida:** Este método devuelve un objeto JSON con los parámetros:

- Código: Código del error. "0" indica que no hubo error, valores diferentes de "0" indican que ha ocurrido un error.
- Descripción: Contiene la descripción del error.

A continuación se define la estructura del objeto JSON recibido como parámetro de entrada.

```

{
  "empresa_id": "1",
  "fecha_actualizacion": "2019-02-19 00:03:25.000",
  "expediciones_realizadas": [
    {
      "conductor_id": "1",
      "vehiculo_id": "1",
      "linea_id": "1",
      "ruta_id": "1",
      "dia_servicio": "2019-01-02",
      "fecha_inicio": "2019-01-02 09:00:00",
      "fecha_fin": "2019-01-02 09:55:00"
    },
    {
      "conductor_id": "1",
      "vehiculo_id": "1",
      "linea_id": "1",
      "ruta_id": "2",
      "dia_servicio": "2019-01-02",
      "fecha_inicio": "2019-01-02 10:00:00",
      "fecha_fin": "2019-01-02 10:53:00"
    }
  ]
}

```

**Ilustración 46: JSON del parámetro de entrada con datos de las expediciones realizadas**

En las tablas siguientes se describe los objetos contenidos en el JSON de datos de las expediciones realizadas.

Empresa:

Atributo	Tipo	Null	Descripción
empresa_id	String	N	Identificador de la empresa en el sistema Econbus

Fecha de actualización:

Atributo	Tipo	Null	Descripción
fecha_actualizacion	String	N	Fecha de actualización de los datos

Grupo de explotación:

Atributo	Tipo	Null	Descripción
----------	------	------	-------------

conductor_id	String	N	Identificador del conductor en el sistema externo
vehiculo_id	String	N	Identificador del vehículo en el sistema externo
linea_id	String	N	Identificador de la línea en el sistema externo
ruta_id	String	N	Identificador de la ruta en el sistema externo
dia_servicio	String	N	Día de servicio
fecha_inicio	String	N	Fecha y hora inicial de la expedición
fecha_fin	String	Y	Fecha y hora final de la expedición

El siguiente cuadro de texto contiene el JSON con los parámetros de salida del método.

```
{
  "codigo": 0,
  "descripcion": ""
}
```

**Ilustración 47: JSON del parámetro de salida con datos de la respuesta del método**

En la siguiente tabla se describe el objeto contenido en el JSON del parámetro de salida.

Atributo	Tipo	Descripción
codigo	Entero	Código del error
descripcion	String	Descripción del error

### 4.7 Obtener datos de informe para el conductor

Este método de la API de la pasarela implementa las funcionalidades para el envío de los datos del informe del conductor.

**Nombre:** getDriverReportData

**Parámetros de entrada:** Recibe una cadena de caracteres de un objeto JSON con los datos de solicitud del informe del conductor.

A continuación se define la estructura del objeto JSON recibido como parámetro de entrada.

```
{
  "empresa_id": "1",
  "conductor_id": "1"
}
```

**Ilustración 48: JSON del parámetro de entrada con datos de solicitud del informe para el conductor**

En la tabla siguiente se describe los atributos del objeto JSON del parámetro de entrada.

Atributo	Tipo	Null	Descripción
empresa_id	String	N	Identificador de la empresa en el sistema Econbus
conductor_id	String	N	Identificador de del conductor en el sistema externo

**Parámetros de Salida:** Devuelve una cadena de caracteres de un objeto JSON con los datos del informe para el conductor.

A continuación se define la estructura del objeto JSON de respuesta o parámetro de salida.

```
{
  "informe": {
    "empresa_id": 1,
    "conductor_id": 1,
    "viajes": 14,
    "fecha_inicio": "2019-10-01 17:00:00",
    "fecha_fin": "2019-10-08 17:00:00",
    "frenadas": 4,
    "aceleraciones": 3,
    "inercia": 1,
    "ralenti": 4,
    "confort": 2,
    "minimo": 57,
    "maximo": 83,
    "media": 71,
    "propio": 80
  }
}
```

**Ilustración 49: JSON del parámetro de salida con datos del informe para conductor**

En la tabla siguiente se describe los atributos del objeto JSON del parámetro de salida.

Atributo	Tipo	Null	Descripción
empresa_id	String	N	Identificador de la empresa en el sistema Econbus
conductor_id	String	N	Identificador de del conductor en el sistema externo



viajes	Entero	N	Cantidad de viajes o expediciones realizadas por el conductor en el periodo analizado
fecha_inicio	String	N	Fecha inicial del periodo de datos analizado
fecha_fin	String	N	Fecha inicial del periodo de datos analizado
frenadas	Entero	N	Nivel KPI Frenadas (0, 1, 2, 3, 4)
aceleraciones	Entero	N	Nivel KPI Aceleraciones (0, 1, 2, 3, 4)
inercia	Entero	N	Nivel KPI Inercia (0, 1, 2, 3, 4)
ralentí	Entero	N	Nivel KPI Ralenti (0, 1, 2, 3, 4)
confort	Entero	N	Nivel KPI Confort (0, 1, 2, 3, 4)
minimo	Entero	N	Mínimo global (KPI peores conductores)
maximo	Entero	N	Máximo global (KPI mejores conductores)
media	Entero	N	Media global del KPI
propio	Entero	N	KPI Propio del conductor

## 5 EVENTOS DE ECODRIVING

Las medidas correspondientes a eventos de ecodriving se agrupan en registros. Se consideran los siguientes tipos de registros:

- MV Movimiento
- DT Detención
- FB Frenada Brusca
- AF Aceleración seguida de Frenada
- DB Detención Brusca
- GB Giro Brusco
- AB Aceleración Brusca
- PB Paso Brusco por resalte / firme irregular



## 1.1 MV – Movimiento

El registro MV se crea cada vez que el vehículo se detiene, y contiene datos obtenidos desde la detención anterior; es decir, correspondientes a la marcha del vehículo entre dos detenciones.

Tipo	Campos del registro MV	Notas
Long	1 T_ini_mv	Tiempo transcurrido desde el inicio de la sesión
Word	2 T_mv	Duración del evento MV
Word	3 Distancia_ini_mv	Distancia recorrida desde el inicio de la sesión
Word	4 Distancia_mv	Distancia recorrida durante el evento MV
Word	5 N_Fvec_events_mv	Nº de deceleraciones sucedidas en el evento MV
DWord	6 Fvec_mv	Fvec de las deceleraciones sucedidas en el evento MV
Word	7 N_Fvec_mv	Nº de muestras utilizadas para el cálculo de Fvec_mv
Word	8 T_fvec_mv	Duración total de las deceleraciones sucedidas en el evento MV
DWord	9 Fvec_aprox_mv	Fvec de la aproximación
Word	10 N_fvec_aprox_mv	Nº de muestras utilizadas para el cálculo de Fvec_aprox_mv
Word	11 T_pedal_fr_mv	Tiempo acumulado frenando durante el evento MV
Word	12 Fvec_eco_mv	Fvec de las indicaciones al conductor durante el evento MV
Word	13 Dist_mpi_mv	Distancia recorrida aprovechando la inercia durante el evento MV
Word	14 T_kd_mv	Tiempo total de uso del KD durante el evento MV
Word	15 T_kd_indebido_mv	Tiempo de uso indebido del KD durante el evento MV
Word	16 T_kd_salida_mv	Tiempo de uso del KD en la salida MV
Word	17 T_kd_salida_indebido_mv	Tiempo de uso indebido del KD en la salida
DWord	18 Consumo_sesion	Consumo desde el inicio de la sesión hasta el inicio de MV
DWord	19 Consumo_mv	Consumo durante el evento MV
Word	20 Consumo_salida_mv	Consumo en la salida MV
Word	21 T_luces_mv	Tiempo con las luces encendidas durante el evento MV
Word	22 T_luces_indebido_mv	Tiempo con las luces encendidas indebidamente
Word	23 T_ac_mv	Tiempo con el compresor de A/C encendido
Word	24 T_hi_rpm1_mv	Tiempo con el motor a más de Umbral_rpm1 RPM
Word	25 T_hi_rpm2_mv	Tiempo con el motor a más de Umbral_rpm2 RPM
Integer	26 Pendiente_tramo	Pendiente media del tramo
Word	27 Vel_max_mv	Velocidad máxima alcanzada en el tramo
Byte	28 Temp_motor_max	Temperatura máxima del motor alcanzada
Integer	29 Umbral_fvec	Umbral de deceleración para el cálculo de Fvec
	30 [Expedición]	Línea, Ruta, Hora de inicio Opcional

## 1.2 DT – Detención

El registro DT se crea cada vez que el vehículo se pone en marcha, y contiene datos obtenidos durante la detención.



Tipo	Campos del registro DT	Notas
Long	1 T_ini_dt	Tiempo transcurrido desde el inicio de la sesión
Word	2 T_dt	Duración de la detención
DWord	3 Distancia_ini_dt	Distancia recorrida desde el inicio de la sesión
Word	5 Consumo_dt	Consumo en ralentí durante la detención
Word	6 T_ralenti_dt	Tiempo en ralentí durante la detención
Word	7 T_umbral_ralenti	Tiempo máximo que se debe permanecer en ralentí
Word	8 T_puertas_dt	Tiempo con las puertas abiertas durante la detención
Integer	9 Pendiente	Valor de Ax medida con el acelerómetro durante la detención
Word	10 Ruido_acel	Ruido del acelerómetro (precisión en el cálculo de la pendiente)
Long	11 Latitud	
Long	12 Longitud	
	30 [Expedición]	Línea, Ruta, Hora de inicio Opcional

### 1.3 FB – Frenada Brusca

El registro FB se crea al finalizar un evento de frenada brusca. Los eventos de frenada brusca se inician cuando el valor absoluto de la aceleración negativa de la frenada excede un umbral, Umbral\_fb, y finalizan cuando dicho valor absoluto es menor que Umbral\_fb menos un valor de histéresis (típicamente ~20mG).

Tipo	Campos del registro FB	Notas
Long	1 T_ini_fb	Tiempo transcurrido desde el inicio de la sesión
Word	2 T_fb	Duración de la frenada
DWord	3 Distancia_ini_fb	Distancia recorrida desde el inicio de la sesión
Word	4 Distancia_fb	Distancia recorrida durante el evento FB
DWord	5 Fvec_fb	Fvec de la frenada
Word	6 N_Fvec_fb	Nº de muestras acumuladas para obtener Fvec_fb
Integer	7 Fr_max_fb	Valor absoluto de la máxima deceleración registrada
Word	8 Velocidad_ini_fb	Velocidad al comienzo de la frenada
Word	9 Velocidad_fin_fb	Velocidad después de la frenada
Long	10 Latitud	
Long	11 Longitud	
	12 [Expedición]	Línea, Ruta, Hora de inicio Opcional

### 1.4 AF – Aceleración seguida de Frenada

Cualquier frenada cuya intensidad en valor absoluto supere **Umbral\_fr\_af** mG y que se produzca menos de **Umbral\_t\_af** segundos después de haber registrado una aceleración con intensidad superior a **Umbral\_ac\_af** mG generará un evento AF.

Tipo	Campos del registro	Notas
------	---------------------	-------



		AF	
Long	1	T_ini_af	Tiempo transcurrido desde el inicio de la sesión
Word	2	T_af	Duración de la frenada
DWord	3	Distancia_ini_af	Distancia recorrida desde el inicio de la sesión
Word	4	Distancia_af	Distancia recorrida desde el inicio de la aceleración hasta el final de la frenada
Integer	5	Acel_max_af	Maxima aceleración registrada
DWord	6	Fvec_af	Fvec de la frenada
Word	7	N_Fvec_af	Nº de muestras acumuladas para obtener Fvec_af
Word	8	T_fvec_af	Duración de la frenada mientras ha excedido Umbral_fr_af
Integer	9	Fr_max_af	Valor absoluto de la máxima deceleración registrada
Word	10	Velocidad_ini_af	Velocidad al inicio de la frenada
Word	11	Velocidad_fin_af	Velocidad después de la frenada
Word	12	T_ac_fr_af	Segundos desde el inicio de la aceleración hasta el inicio de la frenada
Long	13	Latitud	
Long	14	Longitud	
	15	[Expedición]	Línea, Ruta, Hora de inicio Opcional

## 1.5 DB – Detención Brusca

Tipo	Campos del registro DB		Notas
Long	1	T_ini_db	Tiempo transcurrido desde el inicio de la sesión
DWord	2	Distancia_ini_db	Distancia recorrida desde el inicio de la sesión
Integer	3	Vec_db	Valor acumulados de Ax_diff durante la detención brusca
Long	4	Latitud	
Long	5	Longitud	
	6	[Expedición]	Línea, Ruta, Hora de inicio Opcional

## 1.6 GB – Giro Brusco

Tipo	Campos del registro GB		Notas
Long	1	T_ini_gb	Tiempo transcurrido desde el inicio de la sesión
DWord	2	Distancia_ini_gb	Distancia recorrida desde el inicio de la sesión
DWord	3	gVEC	VEC de la aceleración lateral
Long	4	Latitud	
Long	5	Longitud	
	6	[Expedición]	Línea, Ruta, Hora de inicio Opcional



## 1.7 AB – Aceleración Brusca

Tipo	Campos del registro AB	Notas
Long	1 T_ini_ab	Tiempo transcurrido desde el inicio de la sesión
DWord	2 Distancia_ini_ab	Distancia recorrida desde el inicio de la sesión
DWord	3 Avec_ab	
Word	4 N_avec_ab	
Word	5 T_avec_ab	
Integer	6 Ax_max_ab	Aceleración máxima alcanzada
Long	7 Latitud	
Long	8 Longitud	
	9 [Expedición]	Línea, Ruta, Hora de inicio Opcional

## 1.8 PB – Paso Brusco por resalte (firme irregular)

Tipo	Campos del registro PB	Notas
Long	1 T_ini_pb	Tiempo transcurrido desde el inicio de la sesión
DWord	2 Distancia_ini_pb	Distancia recorrida desde el inicio de la sesión
Integer	3 Diff_ax_pb	Maxima variación de la aceleración longitudinal
Integer	4 Dixx_az_pb	Maxima variación de la aceleración vertical
Long	5 Latitud	
Long	6 Longitud	
	7 [Expedición]	Línea, Ruta, Hora de inicio Opcional

## 5.1 Tipos de datos

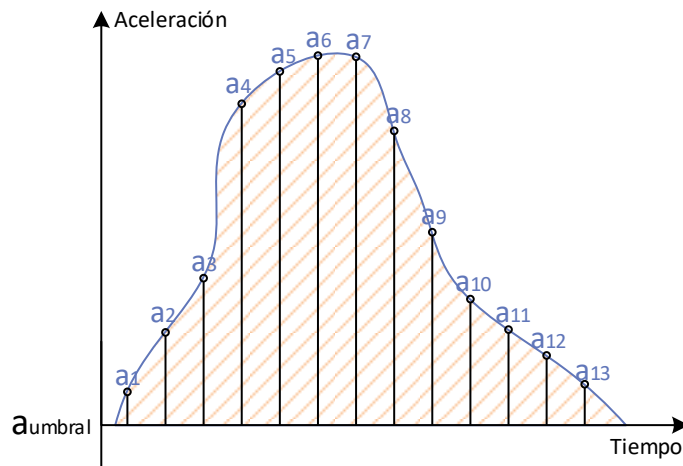
Tipo	Nº de Bytes	Signed/Unsigned	Rango*
Byte	1	Unsigned	0~255
Integer	2	Signed	-32.768 a +32.767
Word	2	Unsigned	0~65.535
DWord	4	Unsigned	0~4.294.967.295
Long	4	Signed	-2.147.483.648 a 2.147.483.647

\* Ningún dato puede adquirir el valor máximo. **El valor máximo se utiliza para indicar que el dato no está disponible.**

## 5.2 Concepto de VEC, fVEC y aVEC

Denominamos VEC (valor de eficiencia de la conducción) de frenada (fVEC) de aceleración (aVEC) o de cualquier otra medida al área que resulta de cortar con un umbral la curva que representa la variación de dicha medida en el tiempo.

Por ejemplo, en el caso de la aceleración se calcula a partir de los valores de la aceleración,  $a_i$ , que exceden un umbral determinado (Umbral VEC).



$$VEC = \sum_{i=1}^{13} \frac{|a_i - a_{umbral}|^n}{C}$$

También puede calcularse de la misma forma para otras variables diferentes de la aceleración (por ejemplo para calcular un VEC velocidad como un indicador que refleje en cuánto se ha excedido la velocidad y durante cuánto tiempo).