Relatório do Projecto

CampusPark – Smart Campus Parking Platform

Índice

1 Introdução 1

2 Providers 1

3 Data Acquisition and Content Enricher (ParkDACE)   4

4 Parking Spot Transmission Unit (ParkTU) 5

5 Parking Spot Storage App (ParkSS) 7

6 Smart Park RESTful Web Service 8

7 Admin Dashboard (ParkDashboard) 8

8 Requisitos não implementados 9

9 Contribuição de cada elemento do grupo 9

10 Conclusão 10

*Anexo A* – Listagem dos Ficheiro XML utilizados 10

*Anexo B* – Outras informações 11

# Introdução

A solução CampusPark implementada reúne um conjunto de micro aplicações que desempenham diferentes funcionalidades.

* **BOT-SpotSensors (SOAP)** – Serviço Web SOAP que cria um ficheiro com formato XML que inclui dados específicos de um determinado Parking Spot. Implementa a interface de comunicação entre um sensor de um Parking Spot com a aplicação ParkDACE.
* **ParkDACE** – Recebe toda a informação fornecida pelos diversos sensores espalhados pelo Campus da ESTG. A informação recebida seria por vários serviços Web SOAP na vida real. Para simular tal comportamento, foi adicionada uma DLL, ParkingSensorNodeDLL, que envia múltiplos dados aleatórios relativos a um determinado Parking Spot. Além disso, adiciona informação da localização (em coordenadas) do sensor que emite a informação e a reúne num ficheiro XML, ParkingNodesConfig.xml. Todo o conteúdo deste ficheiro será encaminhado para a aplicação ParkTU;
* **ParkTU (Parking Spots Transmission Unit) –** Plataforma de comunicação entre a fonte de dados dos sensores (ParkDACE) e outras aplicações cliente que pretendem ter acesso aos dados. Aproveita a informação fornecida no ficheiro ParkingNodesConfig.xml e preenche o ficheiro parks.xml com informação relevante dos Parks e ParkingSpots existentes no Campus. Disponibiliza o seu conteúdo em domínio público;
* **ParkSS (Parking Spot Storage App)** – Linha de consola que automatiza a receção de dados fornecidos pelo ParkTU (o conteúdo XML do ficheiro parks.xml) e os armazena numa database relacional (ParkDB);
* **SmartPark (RESTful Web Service) –** RESTful web service disponível a dispositivos móveis e utilizadores finais.
* **Admin Dashboard (PaskDashboard)** – Aplicação Dashboard que monitoriza e configura Parks e Parking Spots existentes em toda a plataforma.

# Providers

Existem dois tipos de Providers que criam dados sobre a ocorrência de eventos de múltiplos Parking Spots de um determinado Park:

**ParkingSensorNodeDLL**

Ficheiro DLL disponível na aplicação ParkDACE.Desempenha como um gerador de dados aleatórios com um *timeout* inerente. O tempo definido pela obtenção de novos dados poderá ser obtido no ficheiro ParkingNodesConfig.xml. Caso não esteja definido no ficheiro, foi estipulado um tempo *default* com duração de 1 minuto, o que permite simular a ocorrência de um evento de um dado sensor na vida real a cada 30 segundos.

O conjunto de dados recolhidos são obtidos em formato *string* que obedecem à seguinte estrutura:

*ParkId;SpotId;Timestamp;ParkingSpotStatus;BatteryStatus*

Após a sua receção, a *string* é dividida num vetor de *strings* em que “;” assume como divisor dos dados previamente gerados.

Definido o vetor, procede-se à criação de um novo Parking Spot que será convertido futuramente em formato XML para recolha de dados por outras aplicações.

**BOT-SpotSensors (SOAP) :**

Serviço Web SOAP com a funcionalidade de estipular dados e gerar um novo Parking Spot. Após a definição de dados, o Parking Spot será convertido em dois formatos: XML e Data Contract.

A estrutura do conteúdo em formato XML do Parking Spot será previamente analisada por um ficheiro *schema*, ParkingSpotConfig.xsd e armazenado localmente num ficheiro XML, ParkingSpotConfig.xml. A estrutura implementada pelo serviço fornece a seguinte informação:

1. **ID:** O nome identificador do Park;
2. **Tipo:** Corresponde ao tipo de Parking Spot. Pode assumir diferentes tipos: *ParkingSpot, SpecialSpot, TeacherSpot, BicycleSpot, MotoSpot;*
3. **Localização:** As coordenadas do sensor que monitoriza o Parking Spot
4. **Estado:** Ocupado ou Livre (“occupied”/“free”)
5. **Data e hora do último evento**
6. **Estado da bateria do sensor:** Corresponde a um *boolean* que controla a bateria restante do sensor

Exemplo momentâneo do ficheiro ParkingSpotConfig.xml:

<?xml version="1.0"?>

<parkingSpot>

<id>Campus\_2\_A\_Park1</id>

<type>ParkingSpot</type>

<name>A-4</name>

<location></location>

<status>

<value>occupied</value>

<timestamp>2018-12-05T16:18:00.0000000</timestamp>

</status>

<batteryStatus>0</batteryStatus>

</parkingSpot>

O ficheiro ParkingSpotConfig.xml criado ficará disponível para recolha do seu conteúdo através da aplicação ParkDACE.

# Data Acquisition and Content Enricher (ParkDACE)

O ficheiro ParkingNodesConfig.xml mantém toda a informação atualizada de cada Parking Spot/sensor desde o arranque do sistema de sensores. Não haverá no ficheiro dois Parking Spots com o mesmo nome identificador e data do último envio de dados do sensor que os emite.

O ParkDACE redireciona informação que recebe de derivadas fontes de dados (após enriquecê-la com dados provenientes de outros sensores, como por exemplo a geolocalização do sensor) para a plataforma de comunicação, o ParkTU.

Nesta aplicação, começámos por definir quais os provedores da informação dos sensores – pela DLL ou pelo serviço SOAP, já mencionados no ponto anterior.

Na nossa base de dados, tanto o estado da bateria como o estado do parking spot são guardados e representados em valor binário (0 ou 1, *true* ou *false*) consoante um determinado estado:

* **Estado da bateria:** 0 se a bateria se encontra em bom estado, 1 se necessita de ser substituída.
* **Estado do spot:** 0 se o spot está livre, 1 se está ocupado.

O estado do Parking Spot, para além do seu valor referido em 2, também guarda a data e hora da sua última ocorrência – Timestamp. Este atributo é representado pelo tipo de dados *DateTime*.

Cada Parking Spot é caracterizado por um tipo: este pode ser ParkingSpot, SpecialSpot, TeacherSpot, BicycleSpot ou MotoSpot. Para representar este atributo utilizámos uma enumeração.

Para além disso, cada spot tem uma localização, definida por dados do tipo *string*. Por fim, cada spot também é caracterizado pelo seu ID (spotId) e pelo ID do Park a que pertence (ParkID), ambos do tipo *int*.

Após obter todos estes dados, a aplicação ParkDACE junta-a num só ficheiro do tipo XML e envia-a para a aplicação ParkTU. Este envio é feito através de uma Unidade de Transmissão de Dados: ServiceParkTransmissionUnitClient. Antes do envio, os dados são mostrados ao Utilizador.

Para a transmissão dos dados, foi utilizada na aplicação ParkTU o formato de dados XML porque é um formato de fácil compreensão e organização da informação, conhecido por muitas aplicações, o que grande interoperabilidade entre diversas aplicações.

# Parking Spot Transmission Unit (ParkTU)

Com a implementação do sistema de mensagens Mosquitto, a aplicação ParkTU recebe em formato XML o conteúdo do ficheiro ParkingNodesConfig.xml com informação detalhada dos Parks e Parking Spots.

Inclui:

1. **O tipo de Provider** que gerou a informação de determinado Park (DLL ou SOAP)
2. **O endpoint** que corresponde ao URI do *endpoint* de acesso aos dados no caso do serviço SOAP ou do nome do gerador de dados aleatórios, *ParkingSensorNodeDLL***.**
3. A informação do Park que confere a seguinte estrutura:
   1. **Id:** Nome identificador do Park
   2. **Descrição:** Breve descrição da sua localização
   3. **Número de Parking Spots existentes**
   4. **Número de Parking Spots destinados a outros utentes e meios de transporte**
   5. **Descrição do horário de funcionamento do Park**
   6. **Nome do ficheiro Excel que dispõe da informação da localização dos sensores** espalhados pelo Campus. O ficheiro lista um conjunto de coordenadas (latitude, longitude) que correspondem a cada nome identificador de determinado Parking Spot.
   7. **Listagem dos seus Parking Spots** em que a estrutura de cada item constitui a estrutura do ficheiro ParkingSpotConfig.xml criado anteriormente pelo serviço SOAP *BOT-SpotSensors***.**

Exemplo momentâneo do ficheiro ParkingNodesConfig.xml:

<parkingLocation refreshRate="10" units="seconds">

<!-- Example for the DLL provider -->

<provider>

<connectionType>DLL</connectionType>

<endpoint>ParkingSensorNodeDll.ParkingSensorNodeDll</endpoint>

<parkInfo>

<id>Campus\_2\_A\_Park1</id>

<description>Parking in the campus 2 near the building A entrance</description>

<numberOfSpots>15</numberOfSpots>

<operatingHours>Always open</operatingHours>

<numberOfSpecialSpots>2</numberOfSpecialSpots>

<geoLocationFile>Campus\_2\_A\_Park1.xlsx</geoLocationFile>

<parkingSpots>

<parkingSpot>

<type>ParkingSpot</type>

<name>A-2</name>

<location>39.734884,-8.820745</location>

<status>

<value>it has a vehicle</value>

<timestamp>2018-12-16T19:47:08.0000000</timestamp>

</status>

<batteryStatus>0</batteryStatus>

<parkingSpot>

<type>ParkingSpot</type>

<name>A-1</name>

<location>39.734859,-8.820784</location>

<status>

<value>free</value>

<timestamp>2018-12-16T20:25:11.0000000</timestamp>

</status>

<batteryStatus>0</batteryStatus>

</parkingSpot>

</parkingSpots>

</parkInfo>

</provider>

<!-- Example for a SOAP provider -->

<provider>

<connectionType>SOAP</connectionType>

<endpoint>http://localhost:49325/ServiceParkingSpots.svc</endpoint>

<parkInfo>

<id>Campus\_2\_B\_Park2</id>

<description>Parking between the A and the B building</description>

<numberOfSpots>10</numberOfSpots>

<operatingHours>Never closes</operatingHours>

<numberOfSpecialSpots>1</numberOfSpecialSpots>

<geoLocationFile>Campus\_2\_B\_Park2.xlsx</geoLocationFile>

</parkInfo>

</provider>

<provider>

<connectionType>SOAP</connectionType>

<endpoint>http://localhost:49325/ServiceParkingSpots.svc</endpoint>

<parkInfo>

<id>Campus\_2\_A\_Park1</id>

<description>Parking in the campus 2 near the entrance</description>

<numberOfSpots>15</numberOfSpots>

<operatingHours>Always open</operatingHours>

<numberOfSpecialSpots>2</numberOfSpecialSpots>

<geoLocationFile>Campus\_2\_A\_Park1.xlsx</geoLocationFile>

<parkingSpots>

<parkingSpot>

<type>ParkingSpot</type>

<name>A-4</name>

<location>39,734905,-8.820718</location>

<status>

<value>occupied</value>

<timestamp>2018-12-05T16:18:00.0000000</timestamp>

</status>

<batteryStatus>0</batteryStatus>

</parkingSpot>

</parkingSpots>

</parkInfo>

</provider>

</parkingLocation>

Somente a informação dos Parks e Parking Spots será concebida num ficheiro externo, parks.xml, que lista os Parks (com a informação existente do ParkingNodesConfig.xml) e Parking Spots distribuídos em cada Park pertencente. A listagem de Parking Spots incluída em cada Park assume um conjunto de estruturas em que cada uma obedece ao formato XML gerado pelo ficheiro ParkingSpotConfig.xml criado anteriormente pelo serviço SOAP BOT-SpotSensors.

Exemplo momentâneo do ficheiro parks.xml:

<?xml version="1.0"?>

<parks>

<parkInfo>

<id>Campus\_2\_A\_Park1</id>

<description>Parking in the campus 2 near the building A</description>

<numberOfSpots>15</numberOfSpots>

<operatingHours>Always open</operatingHours>

<numberOfSpecialSpots>2</numberOfSpecialSpots>

<geoLocationFile>Campus\_2\_A\_Park1.xlsx</geoLocationFile>

<parkingSpots>

<parkingSpot>

<type>ParkingSpot</type>

<name>A-7</name>

<location>39.734964,-8.820616</location>

<status>

<value>free</value>

<timestamp>2018-12-05T20:57:04.0000000</timestamp>

</status>

<batteryStatus>0</batteryStatus>

</parkingSpot>

<parkingSpot>

<type>ParkingSpot</type>

<name>A-3</name>

<location>39.734909,-8.820708</location>

<status>

<value>free</value>

<timestamp>2018-12-16T19:48:25.0000000</timestamp>

</status>

<batteryStatus>0</batteryStatus>

</parkingSpot>

</parkInfo>

<parkInfo>

<id>Campus\_2\_B\_Park2</id>

<description>Parking between the A and the B building</description>

<numberOfSpots>10</numberOfSpots>

<operatingHours>Never closes</operatingHours>

<numberOfSpecialSpots>1</numberOfSpecialSpots>

<geoLocationFile>Campus\_2\_B\_Park2.xlsx</geoLocationFile>

</parkInfo>

</parks>

O conteúdo do ficheiro parks.xml será futuramente encaminhado para outras aplicações clientes, por via do sistema Mosquitto, que terão acesso exclusivo aos dados dos sensores em tempo real.

# Parking Spot Storage App (ParkSS)

O ParkSS terá acesso aos dados fornecidos pelo ParkTU através do sistema de mensagens Mosquitto. Após a receção dos dados em formato XML (que corresponde ao conteúdo do ficheiro parks.xml), a informação será distribuída de modo a obtenção de informação correspondente a Parks e Parking Spots.

Primeiramente, os dados obtidos de cada Park serão armazenados numa base de dados relacional, CampusParkDB. Cada Park inserido devolve o seu ID gerado na base de dados a fim de atribuir esse ID aos seus Parking Spots. Em cada transação de inserção de um Park, serão realizadas as inserções dos Parking Spots respetivos ao Park com o ID retornado na transação anterior. Cada Park e Parking Spot só serão armazenados após uma análise prévia dos Parks e Parking Spots já existentes na base de dados. Verifica-se a igualdade entre dados de Parks pelo seu nome identificador (coluna “Name”).

Quanto aos Parking Spots, verifica-se a sua igualdade pelos seguintes campos da tabela *Parking Spots*:

1. Data do último evento do sensor (coluna “Timestamp”)
2. Nome identificador do Parking Spot (coluna “spotId”)
3. Id do Park (coluna “parkId”)

O ParkSS consegue, desde modo, reunir informação nova de sensores incluída na última atualização do ficheiro parks.xml no ParkTU.

Evita a duplicação de dados: a última atualização de determinados sensores não será novamente inserida.

Ao verificar a data e hora da última atualização, inibe-se a reinserção dos mesmos dados para o mesmo Parking Spot.

A explicação seguinte divide a execução da solução CampusPark em diferentes cenários que ocorrem durante a transmissão dos dados e a sua inserção na database relacional CampusParkDB:

**Cenário 1:**

A execução da solução começa primeiramente com as aplicações Park SS e ParkTU à espera da recepção dos ficheiros ParkingNodesConfig.xml e Parks.xml, respectivamente.

**Cenário 2:**

ParkTU recebe conteúdo do ficheiro, cria ou faz overwrite do ficheiro parks.xml já existente incluindo somente os nós XML da informação de Parks e Parking Spots.

**Cenário 3:**

Park TU cria um novo “forúm” do sistema de mensagens Mosquitto chamado “PARK\_SS” de acesso exclusivo somente pela aplicação ParkSS para a recepção do ficheiro Parks.xml.

**Cenário 4:**

Park SS ao receber o ficheiro organiza devidamente os dados recebidos e transforma-os em variáveis do tipo Parking Spot e ParkInfo. Através do serviço Restful API da aplicação SmartPark se sucede a transmissão das variáveis previamente criadas ao executar pedidos HTTP com o seu conteúdo em formato JSON através da metodologia POST.

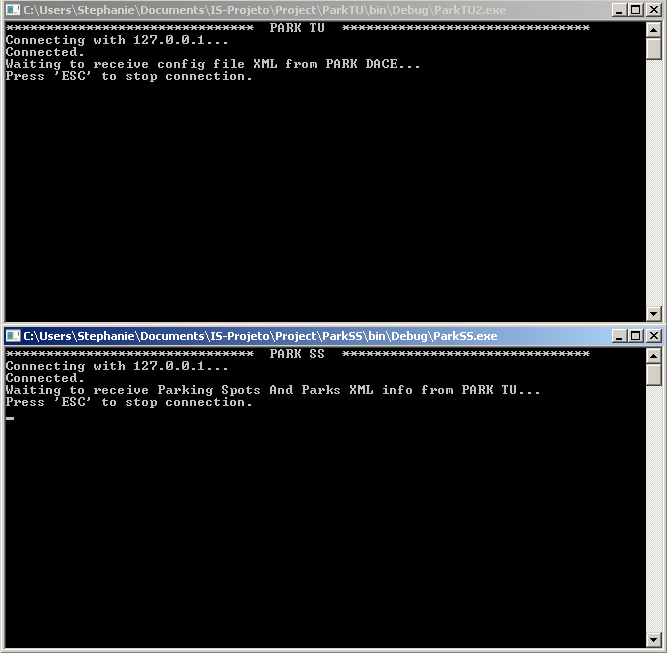
**Cenário 5:**

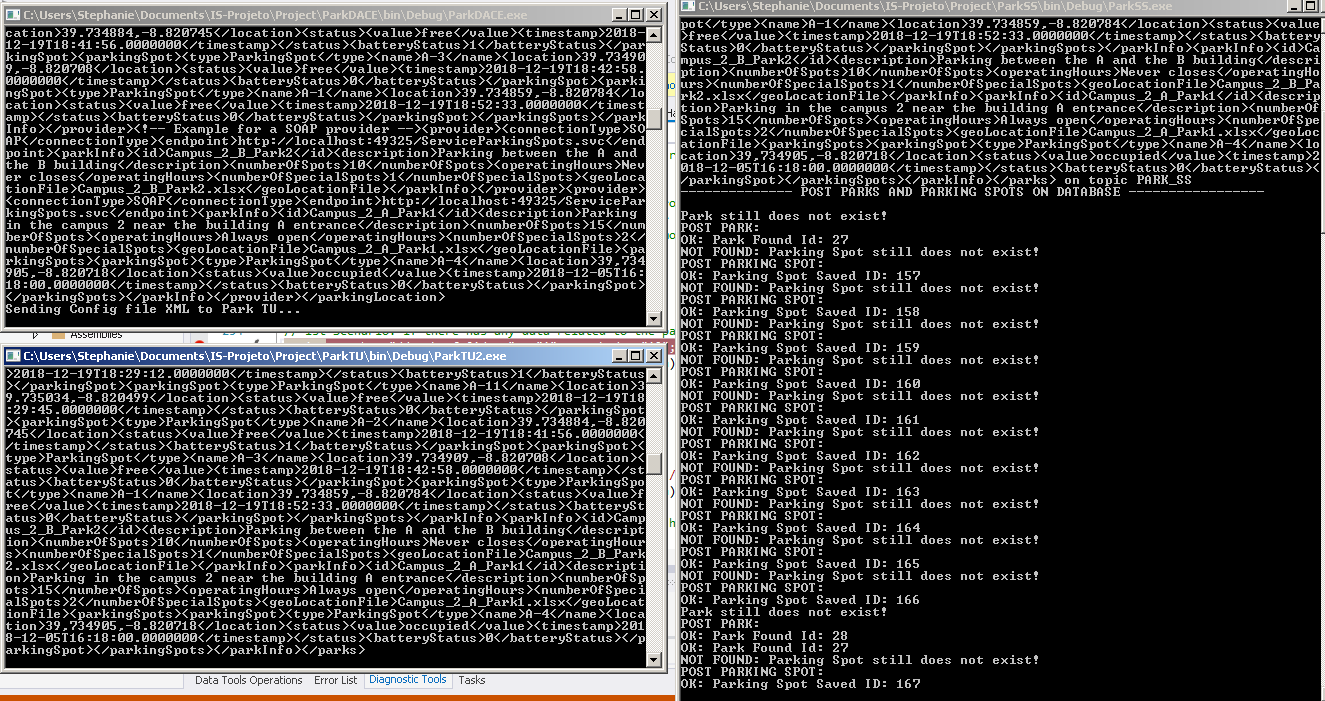
A aplicação SmartPark após receber dados do tipo ParkingSpot e ParkInfo, desformata o seu conteúdo de modo a realizar transações de inserção na database CampusParkDB ali alocada. Envia uma resposta HTTP ao ParkSS para certificar se a operação foi bem sucedida.

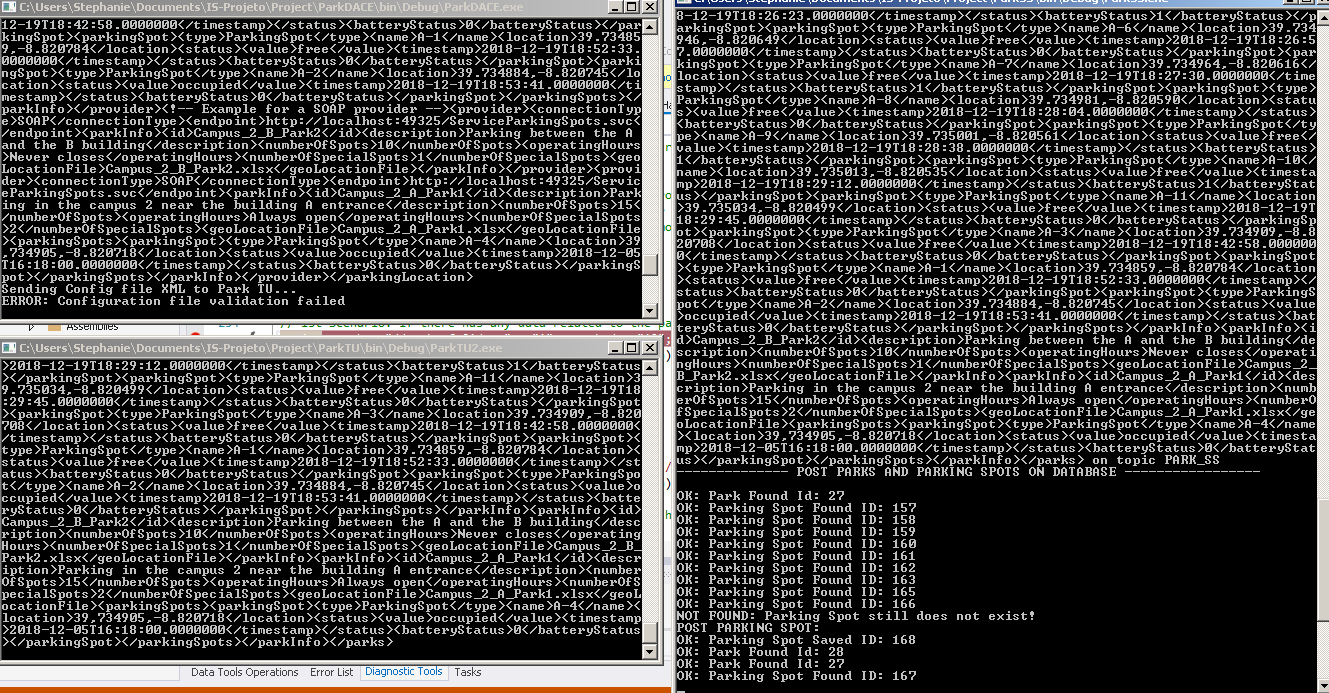
**Cenário 6:**

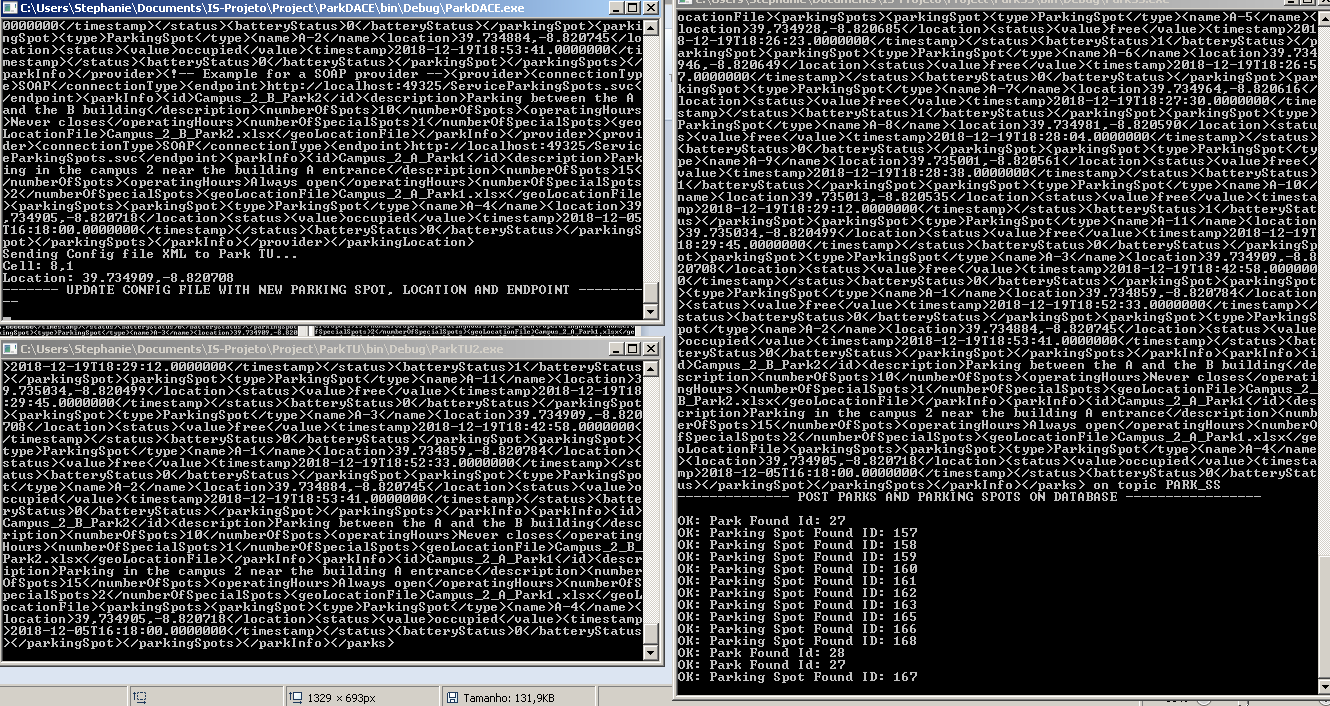
ParkSS notifica o utilizador na sua Linha de Consola consoante as respostas HTTP obtidas. Poderá haver casos em que a database contenha linhas com dados já armazenados de Parks e Parking Spots e informa na consola a ocorrência desse tipo de situações.

Seguem-se em baixo imagens explicitas do comportamento de transmissão de dados através do sistema de mensagens Mosquitto:









**Estrutura de dados da Base de Dados:**

Inclui duas tabelas:

*Parking Spots*

**Colunas:**

1. **ParkId**, tipo *Int*;
2. **SpotId**, tipo *Int*;
3. **Type**, tipo *Varchar*;
4. **Status**, tipo *Bool*;
5. **Timestamp**, tipo *DateTime*;
6. **BatteryStatus**, tipo *Bool*;
7. **Location**, tipo *Varchar*;

**Primary Key: Id,** Incremental, tipo *Int*;

**Foreign Key: parkId** que corresponde à Primary Key Id da tabela *Parks*

*Parks*

**Colunas:**

1. **Name**, tipo *Varchar*;
2. **Description**, tipo *Varchar*;
3. **NumberOfSpots**, tipo *Int*;
4. **OperatingHours**, tipo *Varchar*;
5. **NumberOfSpecialSpots**, tipo *Int*;
6. **GeoLocationFile,** , tipo *Varchar*;

**Primary Key: Id,** Incremental, tipo *Int*;

# Smart Park RESTful Web Service

O SmartPark é uma RESTful API que utiliza HTTP requests para operações CRUD (GET, POST, PUT e DELETE).

Uma RESTful API é um tipo de arquitetura e uma forma de comunicação muito utilizada no desenvolvimento de serviços WEB.

No geral é constituída por controladores (lidam com pedidos HTTP) e por modelos (definem os dados a trocar).

Para além disso, existe também um index.html onde é mostrada a informação proveniente dos pedidos HTTP.

No nosso projeto implementámos dois controladores: um para os Parks e outro para os Parking Spots.

Quanto aos modelos implementámos 3, um para os Parks, outro para os Parking Spots e o outro relativo ao Status do Parking Spot, uma vez que este é composto por dois atributos distintos: o value do status e o timestamp.

Criámos 3 documentos HTML para mostrar a informação dos pedidos HTTP. Um tem a informação relativa aos Parks, outro aos Parking Spots e o outro tem apenas os links para redirecionar o utilizador para cada uma das outras páginas WEB.

No nosso projeto a implementação da RESTful API assentou na utilização de rotas: cada operação disponível nos controladores tem uma rota correspondente através da qual é feito o pedido HTTP.

Em cada um dos controladores estão disponíveis operações CRUD, sendo que existem várias operações onde o método GET é utilizado.

A informação disponibilizada nas páginas WEB é proveniente dos controladores, os quais vão buscar a informação à base de dados (CampusParkDB).

# Admin Dashboard (ParkDashboard)

No ParkDashboard, de modo a disponibilizar a visualização do estado global do sistema, recorremos à utilização de uma Windows Forms App (.NET Framework).

O que é mostrado aos utilizadores são informações gerais sobre os Parks(ID, número de ParkingSpots, número de Special Spots, etc) e Parking Spots(ParkID, SpotID, Type, GeoLocation File, etc).

Para além destas informações também é permitido ao utilizador criar, editar e remover Parks e Parking Spots.

É possível obter informações sobre os Parks e Parking Spots num determinado momento (timestamp) e num determinado período de tempo (diferença entre dois timestamps).

# Requisitos não implementados

Em termos gerais, todas as funcionalidades que nos foram requisitadas no enunciado do projeto, para todos os módulos, foram implementadas na sua totalidade à exceção das credenciais (login e password).

# Contribuição de cada elemento do grupo

Luís Miguel Pires:

ParkSS

Luís Maria Pimentel:

SmartPark

Park Dashboard

Maria Teresa Pereira:

SmartPark

ParkTU

Stéphanie Gaspar:

BOT-SpotSensors

ParkDACE

ParkSS

ParkTU

# Conclusão

Ao longo do desenvolvimento do projeto surgiram-nos algumas dúvidas de como implementar certas partes do projeto e algumas dificuldades derivadas dessas mesmas dúvidas. Porém recorremos ao auxílio dos docentes da unidade curricular que nos ajudaram a dizimar essas dúvidas e nos permitiram clarificar como deveria ser feita a implementação das partes do projeto onde foram sentidas dificuldades.

Para concluir, com este projeto foi-nos permitido perceber melhor e de uma forma diferente os conteúdos lecionados nas aulas práticas e teóricas da Unidade Curricular de Integração de Sistemas.

# Anexos

Ficheiros XML utilizados e dos ficheiros XSD respetivos para cada um deles:

**Anexo A:**

**ParkingSpotConfig.xml:** exibe um modelo Parking Spot em formato XML gerado pelo Serviço Web SOAP, BOT-SpotSensors.

**Anexo B:**

**ParkingSpotConfig.xsd**:

Ficheiro schema .xsd de validação do ficheiro PakingSpotConfig.xml. As restrições aos dados presentes neste ficheiro são:

* **minOccurs:** “1”, para que todos os campos do ParkingSpot sejam obrigatórios;
* **enumeration**:

1. No campo “Type” do ParkingSpot, para que valide se o campo contém ou não um dos tipos previamente definidos: *ParkingSpot, SpecialSpot, TeacherSpot, BicycleSpot, MotoSpot;*
2. No campo “Status” do ParkingSpot, para que assuma apenas valores: “occupied” ou “free”;

**Anexo C:**

**ParkingNodesConfig.xml:**

Contém informação sobre os Parks e sobre os Parking Spots espalhados pelo diversos Campus do IPLeiria. A organização dos dados deste ficheiro em formato XML facilitará a sua interpretação e clareza por diversas aplicações. A informação de cada Parking Spot e Park está estruturada de acordo com a seguinte hierarquia base de elementos ficheiro XML: o elemento <park> terá sempre como nós filhos um conjunto de elementos <parkingSpots>. Além disso, a estrutura deste ficheiro garante benefícios aquando do armazenamento dos dados na database relacional CampusParkDB pela aplicação Consola do Windows ParkSS.

**Anexo D:**

**ParkingNodesConfig.xsd**:

Ficheiro schema .xsd de validação do ficheiro PakingNodesConfig.xml. As restrições aos dados presentes neste ficheiro .xml são:

* As restrições presentes no ficheiro *ParkingSpotConfig.xsd,* para validar os campos dos nós <parkingSpots>
* **Required:** Nos atributos @refreshRate e @units,para que o tempo timeout de envio do ficheiro para o ParkTU seja definido obrigatoriamente.
* **Enumeration:** No elemento <connectionType> do <provider>, que corresponde ao tipo de conecção do provedor dos dados de um determinado Park : *SOAP ou DLL*
* **minOccurs:** “1”: Em todos os elementos do Park, excepto a listagem de Parking Spots exibida no elemento <parkingSpots>, sejam obrigatórios;
* **maxOccurs="unbounded":** No elemento <parkingSpots> de modo a aceitar um conjunto infinito de nós <parkingSpot> de modo a não limitar um Park à recepção de novos Parking Spots.

**Anexo E:**

**Parks.xml**:

Inclui somente a parte do conteúdo que contém informação dos Parks e Parking Spots do ficheiro ParkingNodesConfig.xml: os nós <parkInfo> e <parkingSpots>, mantendo a sua hierarquia de nós XML.

**Anexo F:**

**Parks.xsd**:

Ficheiro schema .xsd de validação do ficheiro Parks.xml. As restrições aos dados presentes neste ficheiro .xml são as mesmas presentes no ficheiro ParkingNodesConfig.xsd de modo a validar os elementos <parkInfo>, <parkingSpots> e <parkingSpot>.

Outras informações

Para o correto funcionamento do projeto em qualquer ambiente, na aplicação SmartPark, logo no início de ambos os controladores, a string estática **‘str\_connectionDB’** deve ser alterada para que corresponda ao caminho absoluto para a base de dados no computador em questão. Tem de se alterar esta string porque estamos a utilizar uma base de dados está alojada localmente na solução.

<?xml version="1.0"?>

<parkingSpot>

<id>Campus\_2\_A\_Park1</id>

<type>ParkingSpot</type>

<name>A-4</name>

<location></location>

<status>

<value>occupied</value>

<timestamp>2018-12-05T16:18:00.0000000</timestamp>

</status>

<batteryStatus>0</batteryStatus>

</parkingSpot>

***Anexo A*** *– ParkingSpotConfig.xml*

<?xml version="1.0" encoding="utf-8"?>

<xs:schema attributeFormDefault="unqualified" elementFormDefault="qualified" xmlns:xs="http://www.w3.org/2001/XMLSchema">

<xs:element name="parkingSpot" type="parkingSpotInfo" />

<xs:complexType name="parkingSpotInfo">

<xs:sequence>

<xs:element name="id" type="xs:string" minOccurs="1"/>

<xs:element name="type" type="typeType" minOccurs="1"/>

<xs:element name="name" type="xs:string" minOccurs="1"/>

<xs:element name="location" type="xs:string" />

<xs:element name="status" type="statusType" minOccurs="1"/>

<xs:element name="batteryStatus" type="xs:boolean" minOccurs="1"/>

</xs:sequence>

</xs:complexType>

<xs:complexType name="statusType">

<xs:sequence>

<xs:element name="value" type="valueType" minOccurs="1"/>

<xs:element name="timestamp" type="xs:dateTime" minOccurs="1"/>

</xs:sequence>

</xs:complexType>

<xs:simpleType name="valueType">

<xs:restriction base="xs:string">

<xs:enumeration value="free"/>

<xs:enumeration value="occupied"/>

</xs:restriction>

</xs:simpleType>

<xs:simpleType name="typeType">

<xs:restriction base="xs:string">

<xs:enumeration value="ParkingSpot"/>

<xs:enumeration value="SpecialSpot"/>

<xs:enumeration value="MotoSpot"/>

<xs:enumeration value="BicycleSpot"/>

<xs:enumeration value="TeacherSpot"/>

</xs:restriction>

</xs:simpleType>

</xs:schema>

***Anexo B*** *– ParkingSpotConfig.xsd*

<parkingLocation refreshRate="10" units="seconds">

<!-- Example for the DLL provider -->

<provider>

<connectionType>DLL</connectionType>

<endpoint>ParkingSensorNodeDll.ParkingSensorNodeDll</endpoint>

<parkInfo>

<id>Campus\_2\_A\_Park1</id>

<description>Parking in the campus 2 near the building A entrance</description>

<numberOfSpots>15</numberOfSpots>

<operatingHours>Always open</operatingHours>

<numberOfSpecialSpots>2</numberOfSpecialSpots>

<geoLocationFile>Campus\_2\_A\_Park1.xlsx</geoLocationFile>

<parkingSpots>

<parkingSpot>

<type>ParkingSpot</type>

<name>A-2</name>

<location>39.734884,-8.820745</location>

<status>

<value>it has a vehicle</value>

<timestamp>2018-12-16T19:47:08.0000000</timestamp>

</status>

<batteryStatus>0</batteryStatus>

<parkingSpot>

<type>ParkingSpot</type>

<name>A-1</name>

<location>39.734859,-8.820784</location>

<status>

<value>free</value>

<timestamp>2018-12-16T20:25:11.0000000</timestamp>

</status>

<batteryStatus>0</batteryStatus>

</parkingSpot>

</parkingSpots>

</parkInfo>

</provider>

<!-- Example for a SOAP provider -->

<provider>

<connectionType>SOAP</connectionType>

<endpoint>http://localhost:49325/ServiceParkingSpots.svc</endpoint>

<parkInfo>

<id>Campus\_2\_B\_Park2</id>

<description>Parking between the A and the B building</description>

<numberOfSpots>10</numberOfSpots>

<operatingHours>Never closes</operatingHours>

<numberOfSpecialSpots>1</numberOfSpecialSpots>

<geoLocationFile>Campus\_2\_B\_Park2.xlsx</geoLocationFile>

</parkInfo>

</provider>

<provider>

<connectionType>SOAP</connectionType>

<endpoint>http://localhost:49325/ServiceParkingSpots.svc</endpoint>

<parkInfo>

<id>Campus\_2\_A\_Park1</id>

<description>Parking in the campus 2 near the entrance</description>

<numberOfSpots>15</numberOfSpots>

<operatingHours>Always open</operatingHours>

<numberOfSpecialSpots>2</numberOfSpecialSpots>

<geoLocationFile>Campus\_2\_A\_Park1.xlsx</geoLocationFile>

<parkingSpots>

<parkingSpot>

<type>ParkingSpot</type>

<name>A-4</name>

<location>39,734905,-8.820718</location>

<status>

<value>occupied</value>

<timestamp>2018-12-05T16:18:00.0000000</timestamp>

</status>

<batteryStatus>0</batteryStatus>

</parkingSpot>

</parkingSpots>

</parkInfo>

</provider>

</parkingLocation>

***Anexo C*** *– ParkingNodesConfig.xml*

<?xml version="1.0" encoding="utf-8"?>

<xs:schema attributeFormDefault="unqualified" elementFormDefault="qualified" xmlns:xs="http://www.w3.org/2001/XMLSchema">

<xs:element name="parkingLocation" type="parkingLocationType"/>

<xs:complexType name="parkingLocationType">

<xs:sequence>

<xs:element name="provider" type="providerType" minOccurs="1" maxOccurs="unbounded"/>

</xs:sequence>

<xs:attribute name="refreshRate" type="xs:unsignedShort" use="required" />

<xs:attribute name="units" type="xs:string" use="required"/>

</xs:complexType>

<xs:complexType name="parkingSpotType">

<xs:sequence>

<xs:element name="type" type="typeType" minOccurs="1"/>

<xs:element name="name" type="xs:string" minOccurs="1"/>

<xs:element name="location" type="xs:string" />

<xs:element name="status" type="statusType" minOccurs="1"/>

<xs:element name="batteryStatus" type="xs:boolean" minOccurs="1"/>

</xs:sequence>

</xs:complexType>

<xs:complexType name="parkInfoType">

<xs:sequence>

<xs:element name="id" type="xs:string" minOccurs="1"/>

<xs:element name="description" type="xs:string" minOccurs="1"/>

<xs:element name="numberOfSpots" type="xs:unsignedShort" minOccurs="1"/>

<xs:element name="operatingHours" type="xs:string" minOccurs="1"/>

<xs:element name="numberOfSpecialSpots" type="xs:unsignedShort" minOccurs="1"/>

<xs:element name="geoLocationFile" type="xs:string" minOccurs="1"/>

<xs:element name="parkingSpots" type="parkingSpotsType" minOccurs="0"/>

</xs:sequence>

</xs:complexType>

<xs:complexType name="parkingSpotsType">

<xs:sequence>

<xs:element name="parkingSpot" type="parkingSpotType" maxOccurs="unbounded"/>

</xs:sequence>

</xs:complexType>

<xs:complexType name="providerType">

<xs:sequence>

<xs:element name="connectionType" type="connectionTypeType" minOccurs="1"/>

<xs:element name="endpoint" type="xs:anyURI" minOccurs="1"/>

<xs:element name="parkInfo" type="parkInfoType" minOccurs="1"/>

</xs:sequence>

</xs:complexType>

<xs:complexType name="statusType">

<xs:sequence>

<xs:element name="value" type="valueType" minOccurs="1"/>

<xs:element name="timestamp" type="xs:dateTime" minOccurs="1"/>

</xs:sequence>

</xs:complexType>

<xs:simpleType name="valueType">

<xs:restriction base="xs:string">

<xs:enumeration value="free"/>

<xs:enumeration value="occupied"/>

</xs:restriction>

</xs:simpleType>

<xs:simpleType name="typeType">

<xs:restriction base="xs:string">

<xs:enumeration value="ParkingSpot"/>

<xs:enumeration value="SpecialSpot"/>

<xs:enumeration value="MotoSpot"/>

<xs:enumeration value="BicycleSpot"/>

<xs:enumeration value="TeacherSpot"/>

</xs:restriction>

</xs:simpleType>

<xs:simpleType name="connectionTypeType">

<xs:restriction base="xs:string">

<xs:enumeration value="DLL"/>

<xs:enumeration value="SOAP"/>

</xs:restriction>

</xs:simpleType>

</xs:schema>

***Anexo D*** *– ParkingNodesConfig.xsd*

<?xml version="1.0"?>

<parks>

<parkInfo>

<id>Campus\_2\_A\_Park1</id>

<description>Parking in the campus 2 near the building A</description>

<numberOfSpots>15</numberOfSpots>

<operatingHours>Always open</operatingHours>

<numberOfSpecialSpots>2</numberOfSpecialSpots>

<geoLocationFile>Campus\_2\_A\_Park1.xlsx</geoLocationFile>

<parkingSpots>

<parkingSpot>

<type>ParkingSpot</type>

<name>A-7</name>

<location>39.734964,-8.820616</location>

<status>

<value>free</value>

<timestamp>2018-12-05T20:57:04.0000000</timestamp>

</status>

<batteryStatus>0</batteryStatus>

</parkingSpot>

<parkingSpot>

<type>ParkingSpot</type>

<name>A-3</name>

<location>39.734909,-8.820708</location>

<status>

<value>free</value>

<timestamp>2018-12-16T19:48:25.0000000</timestamp>

</status>

<batteryStatus>0</batteryStatus>

</parkingSpot>

</parkInfo>

<parkInfo>

<id>Campus\_2\_B\_Park2</id>

<description>Parking between the A and the B building</description>

<numberOfSpots>10</numberOfSpots>

<operatingHours>Never closes</operatingHours>

<numberOfSpecialSpots>1</numberOfSpecialSpots>

<geoLocationFile>Campus\_2\_B\_Park2.xlsx</geoLocationFile>

</parkInfo>

</parks>

***Anexo E*** *– Parks.xml*

<?xml version="1.0" encoding="utf-8"?>

<xs:schema attributeFormDefault="unqualified" elementFormDefault="qualified" xmlns:xs="http://www.w3.org/2001/XMLSchema">

<xs:element name="parks" type="parksType"/>

<xs:complexType name="parksType">

<xs:sequence>

<xs:element name="parkInfo" type="parkInfoType" minOccurs="1" maxOccurs="unbounded"/>

</xs:sequence>

</xs:complexType>

<xs:complexType name="parkingSpotType">

<xs:sequence>

<xs:element name="type" type="typeType" minOccurs="1"/>

<xs:element name="name" type="xs:string" minOccurs="1"/>

<xs:element name="location" type="xs:string" />

<xs:element name="status" type="statusType" minOccurs="1"/>

<xs:element name="batteryStatus" type="xs:boolean" minOccurs="1"/>

</xs:sequence>

</xs:complexType>

<xs:complexType name="parkInfoType">

<xs:sequence>

<xs:element name="id" type="xs:string" minOccurs="1"/>

<xs:element name="description" type="xs:string" minOccurs="1"/>

<xs:element name="numberOfSpots" type="xs:unsignedShort" minOccurs="1"/>

<xs:element name="operatingHours" type="xs:string" minOccurs="1"/>

<xs:element name="numberOfSpecialSpots" type="xs:unsignedShort" minOccurs="1"/>

<xs:element name="geoLocationFile" type="xs:string" minOccurs="1"/>

<xs:element name="parkingSpots" type="parkingSpotsType" minOccurs="0"/>

</xs:sequence>

</xs:complexType>

<xs:complexType name="parkingSpotsType">

<xs:sequence>

<xs:element name="parkingSpot" type="parkingSpotType" maxOccurs="unbounded"/>

</xs:sequence>

</xs:complexType>

<xs:complexType name="statusType">

<xs:sequence>

<xs:element name="value" type="valueType" minOccurs="1"/>

<xs:element name="timestamp" type="xs:dateTime" minOccurs="1"/>

</xs:sequence>

</xs:complexType>

<xs:simpleType name="valueType">

<xs:restriction base="xs:string">

<xs:enumeration value="free"/>

<xs:enumeration value="occupied"/>

</xs:restriction>

</xs:simpleType>

<xs:simpleType name="typeType">

<xs:restriction base="xs:string">

<xs:enumeration value="ParkingSpot"/>

<xs:enumeration value="SpecialSpot"/>

<xs:enumeration value="MotoSpot"/>

<xs:enumeration value="BicycleSpot"/>

<xs:enumeration value="TeacherSpot"/>

</xs:restriction>

</xs:simpleType>

</xs:schema>

***Anexo F*** *– parks.xsd*