Alexandre Meslin (Bolsista CAPES/BRASIL)
Noemi Rodriguez
Markus Endler (meslin, noemi, endler)@inf.puc-rio.br

MUSANet – Mobile Urban Sensor and Actuator Network

Programação

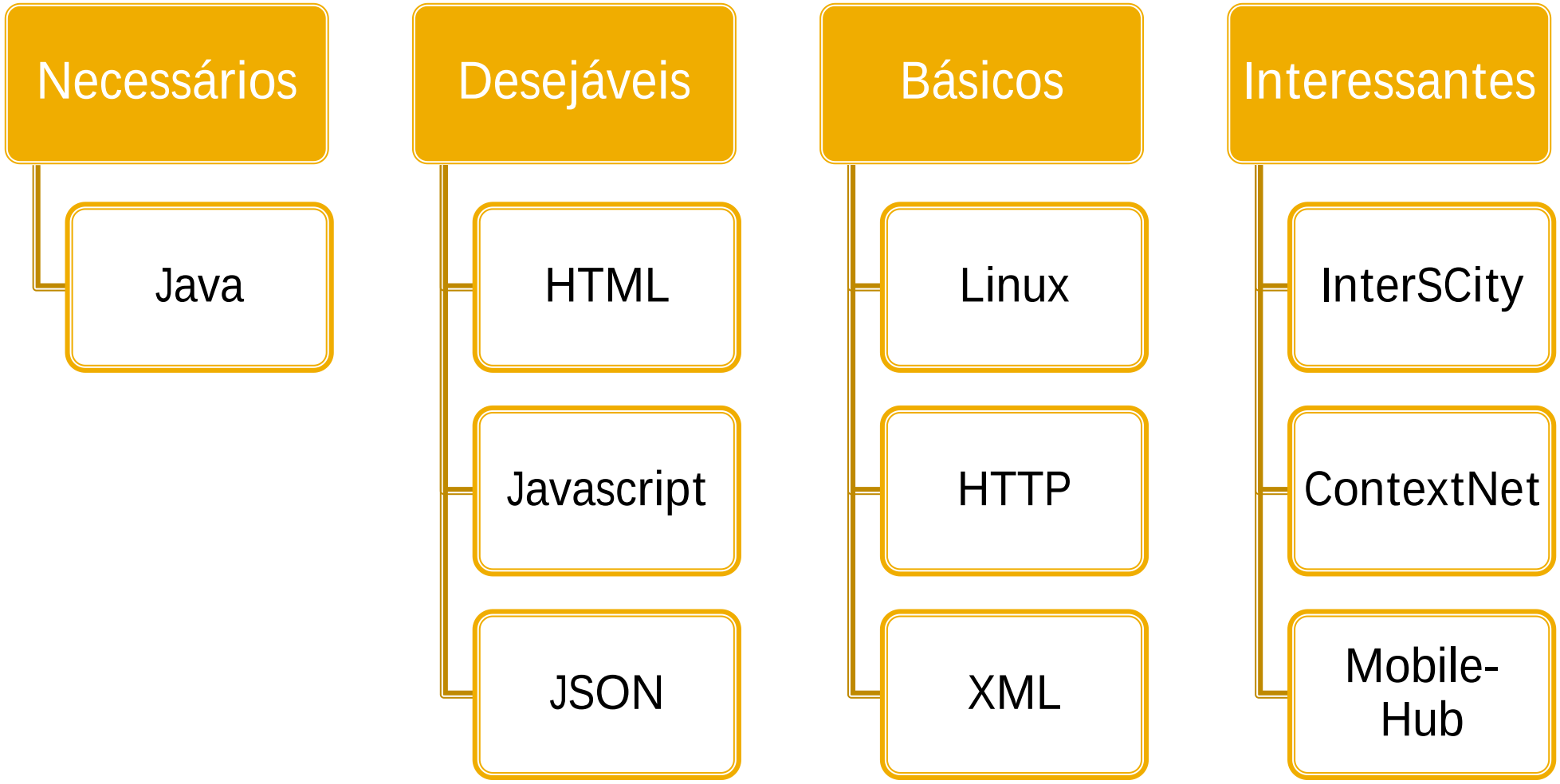
SEMANA 1

Dia 22	• REST e JSON • Introdução ao MUSANet
Dia 23	• Instalando e conhecendo a plataforma InterSCity
Dia 24	• Instalação, configuração do ContextNet. • Desenvolvendo primeiras aplicações.
Dia 25	• Desenvolvendo aplicações com o ContextNet
Dia 26	• Utilizando o Mobile-Hub para captura de dados – inclui transmissão da informação para o ContextNet

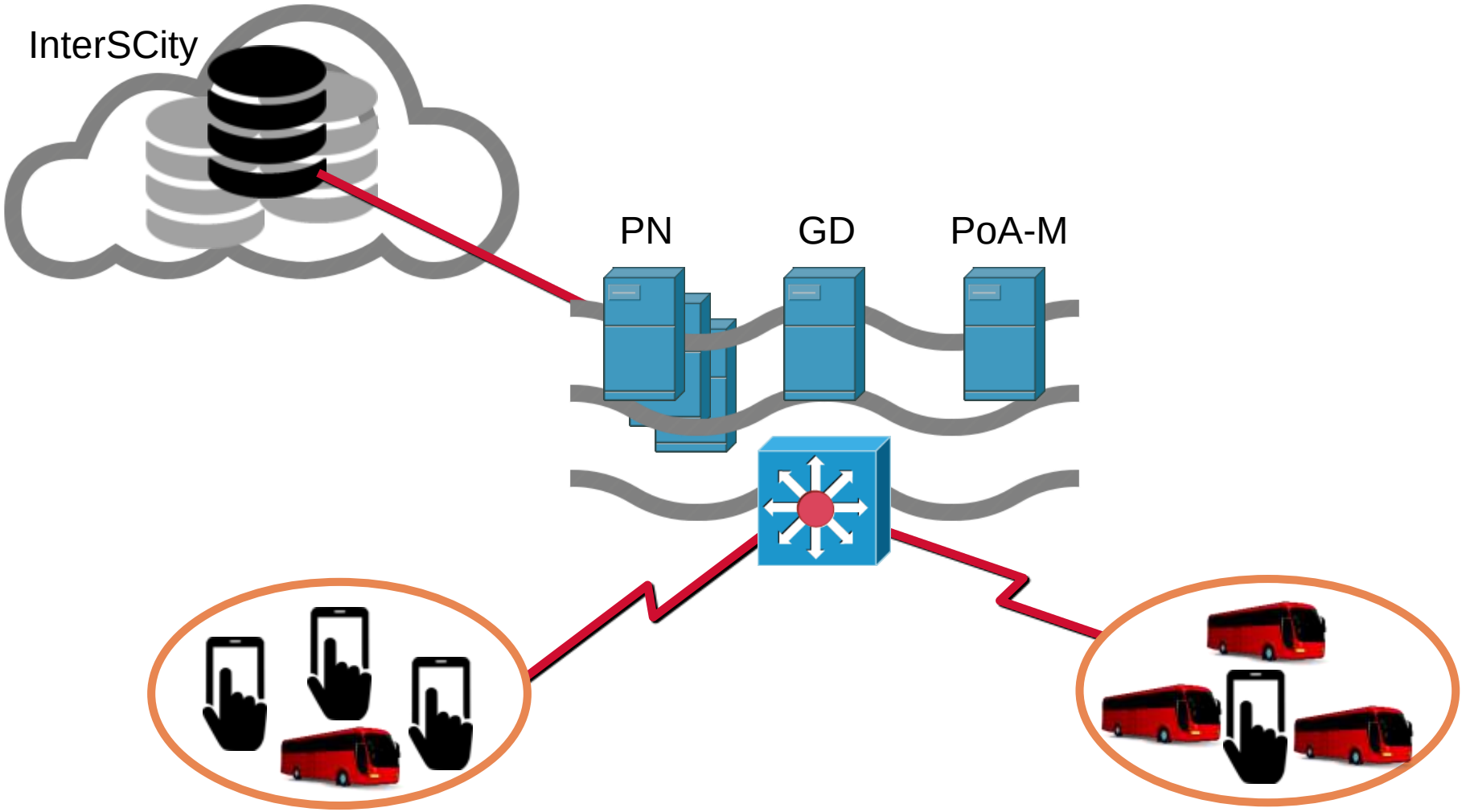
SEMANA 2

Dia 29	• Implementação de uma versão distribuída do ContextNet.
Dia 30	• Projetos usando o MUSANet
Dia 31	• Primeiros testes dos protótipos das aplicações para locais inteligentes
Dia 01	• Continuação dos testes
Dia 02	• Validação e avaliação das aplicações desenvolvidas

Conhecimentos Prévios



Mini Overview



Ambiente

APLICATIVOS

- Eclipse
  (ou equivalente genérico)
- Ubuntu
  (ou equivalente genérico)
- Java 8
- Tomcat (qualquer versão)
- ContextNet
- OpenSplice
- Browser
- VMWare ou VirtualBox

MAVEN

```
<repositories>
    <repository>
        <id>LAC PUC-Rio</id>
        <url>https://bitbucket.org/endler/contextnet-dependencies/raw/master</url>
    </repository>
</repositories>

<dependencies>
    <dependency>
        <groupId>br.pucrio.inf.lac</groupId>
        <artifactId>contextnet</artifactId>
        <version>2.7</version>
    </dependency>
    <dependency>
        <groupId>org.json</groupId>
        <artifactId>json</artifactId>
        <version>20180813</version>
    </dependency>
    <dependency>
        <groupId>com.googlecode.json-simple</groupId>
        <artifactId>json-simple</artifactId>
        <version>1.1</version>
    </dependency>
</dependencies>
```

Slides

- Alguns slides foram baseados no material preparado por:
  - Markus Endler
  - Felipe Carvalho
  - Arthur Del Esposte

Perguntas?

JSON

JavaScript Object Notation

O Que é JSON

Java**s**cript **O**bject **N**otation

Uma sintaxe para armazenar e transmitir informações em formato texto – assim como é feito com XML

Independente de linguagem de programação

Auto-descritivo e fácil de entender

Alguns dizem que JSON é menor, mais rápido e mais fácil do que XML – você decide

- https://www.youtube.com/watch?v=kc8BAR7SHJI

Exemplo de uma estrutura em JSON

```
{
  "aircraft": "A320",
  "pilot":   {
    "firstName": "John",
    "lastName": "Adams"
  },
  "passenger":   [
    "George Washington",
    "Thomas Jefferson"
  ]
}
```

Refere-se a um objeto com a seguinte estrutura e valores:

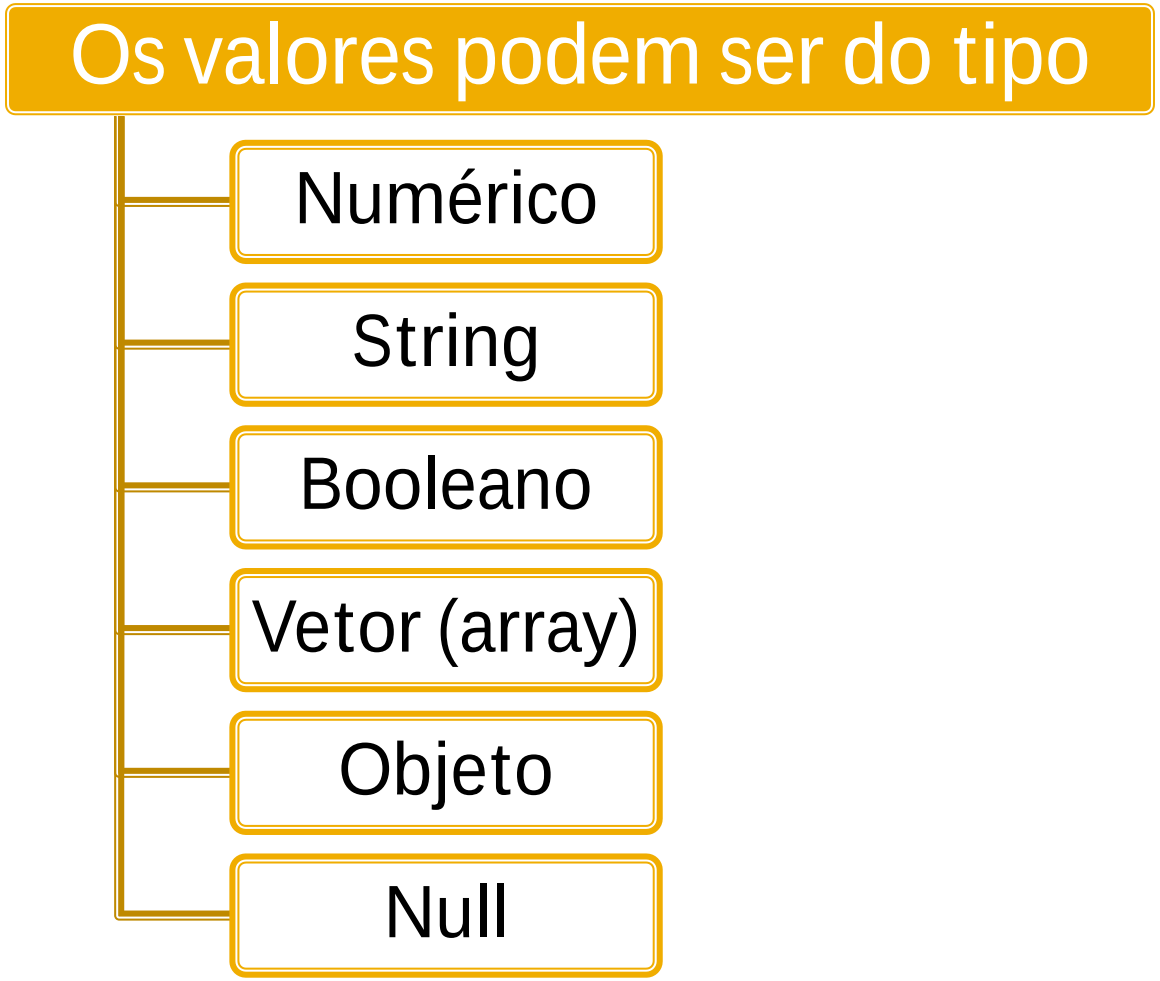
- objeto.aircraft = "A320"
  objeto.pilot.firstName = "John"
  objeto.pilot.lastName = "Adams"
  objeto.passenger[0] = "George Washington"
  objeto.passenger[1] = "Thomas Jefferson"

Sintaxe

Baseado em Javascript

- Dados aparecem em pares de nome/valor
- Dados são separados por vírgula
- Chaves {} separam objetos
- Colchetes [] seraparam vetores
- Sinal de igual = é substituido por dois pontos :

Sintaxe


Os valores podem ser do tipo
Numérico
String
Booleano
Vetor (array)
Objeto
Null

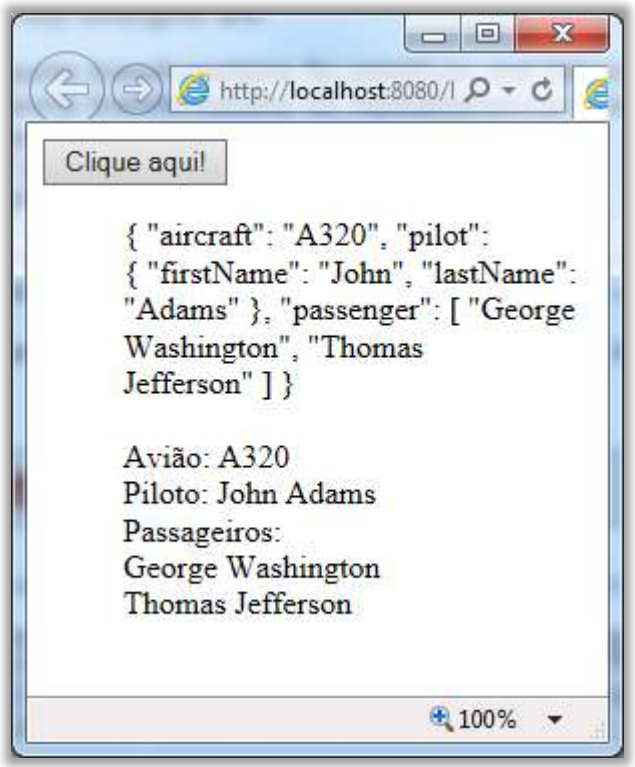
Exemplo: código do lado do servidor

```
public class JSON extends HttpServlet
{
        protected void doGet(HttpServletRequest request, HttpServletResponse response)
                throws ServletException, IOException
        {
                JSONObject json = new JSONObject();
                // Put a simple element
                json.put("aircraft", "A320");
                // Add a JSON Object
                JSONObject pilot = new JSONObject();
                pilot.put("firstName", "John");
                pilot.put("lastName", "Adams");
                json.put("pilot", pilot);
                // Accumulate values in an array
                json.accumulate("passenger", "George Washington");
                json.accumulate("passenger", "Thomas Jefferson");
                PrintWriter out = response.getWriter();
                response.setContentType("text/plain");
                out.println(json.toString());
        }
}
```

Exemplo extraido de http://answers.oreilly.com/topic/263-how-to-generate-json-from-java/

Resultados:

NAVEGADOR



CONSOLE

```
JSON: {
  "aircraft": "A320",
  "pilot":   {
    "firstName": "John",
    "lastName": "Adams"
  },
  "passenger":   [
    "George Washington",
    "Thomas Jefferson"
  ]
}
```

Exemplo
Obtendo posição dos ônibus no Rio de Janeiro

```
package br.com.meslin.ufma.main;

import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStreamReader;
import java.net.HttpURLConnection;
import java.net.URL;

import org.json.JSONArray;
import org.json.JSONObject;
```

```
public class GetBuses {
  // constants
  private static final String USER_AGENT =
"Mozilla/5.0";
  private static final int DATAHORA = 0;
  private static final int ORDEM = 1;
  private static final int LINHA = 2;
  private static final int LATITUDE = 3;
  private static final int LONGITUDE = 4;
  private static final int VELOCIDADE = 5;
```

Exemplo
Obtendo posição dos ônibus no Rio de Janeiro

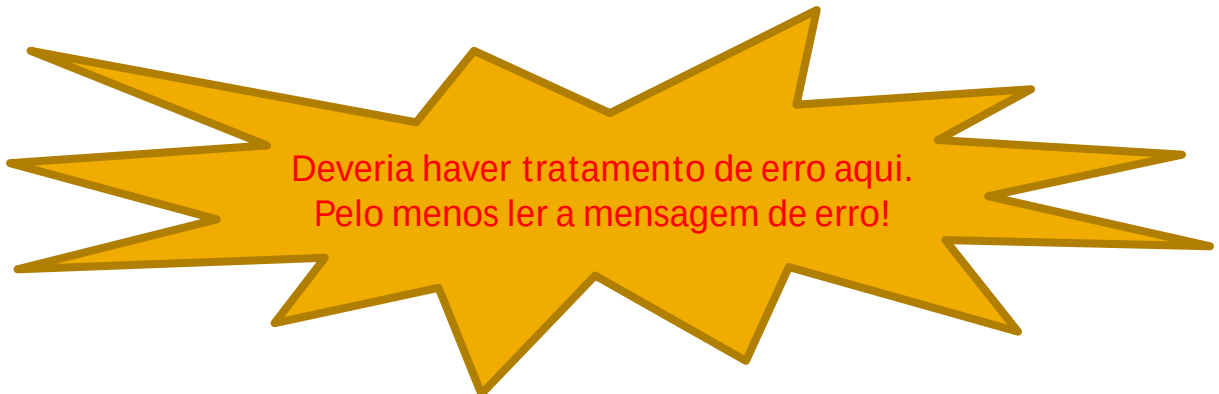
```
public class GetBuses {
  ...
  public static void main(String[] args) {
    GetBuses getBuses = new GetBuses();
    JSONArray jsonBuses = getBuses.production();

    for(int i=0; i<jsonBuses.length(); i++) {
      JSONArray jsonBus = jsonBuses.getJSONArray(i);
      // Houston, we have a problem...LINHA sometime is string, sometimes int
      try {
        System.out.println(jsonBus.getInt(LINHA) + ": " + jsonBus.getString(ORDEM) + " " +
          jsonBus.getString(DATAHORA) + "(" + jsonBus.getDouble(LATITUDE) + "," +
          jsonBus.getDouble(LONGITUDE) + ") @" + jsonBus.getDouble(VELOCIDADE) + "km/h");
      } catch(Exception e) {
        System.out.println(jsonBus.getString(LINHA) + ": " + jsonBus.getString(ORDEM) + " " +
          jsonBus.getString(DATAHORA) + " (" + jsonBus.getDouble(LATITUDE) + "," +
          jsonBus.getDouble(LONGITUDE) + ") @" + jsonBus.getDouble(VELOCIDADE) + "km/h");
      }
    }
  }
  ...
```

Exemplo
Obtendo posição dos ônibus no Rio de Janeiro

```
public class GetBuses {
  ...
  public JSONArray production() {
    // First step: get data from city hall website
    try {
      url = new
URL("http://dadosabertos.rio.rj.gov.br/apiTransporte/apresentacao/rest/index.cfm/obterTodasPosicoes");
      connection = (HttpURLConnection) url.openConnection();
      connection.setRequestMethod("GET");
      connection.setRequestProperty("User-Agent", USER_AGENT);
      responseCode = connection.getResponseCode();
    } catch (IOException e) {
      System.err.println("Fatal error");
    }
    ...
```

Exemplo
Obtendo posição dos ônibus no Rio de Janeiro

```
public class GetBuses {
  ...
  public JSONArray production() {
    ...
    // read bus data
    if(responseCode == 200) {
      try {
        BufferedReader reader = new BufferedReader(new InputStreamReader(connection.getInputStream()));
        String inputLine;
        buffer = new StringBuffer();
        while((inputLine=reader.readLine())!=null) {
          buffer.append(inputLine);
        }
        reader.close();
      } catch (IOException e) {
        System.err.println("Fatal error…");
      }
    } else {
      System.err.println("Responde code…");
      return null;
    }
    ...
```



Exemplo
Obtendo posição dos ônibus no Rio de Janeiro

```
public class GetBuses {
  ...
  public JSONArray production() {
    ...
    // Second step: create the buses structure
    // create a JSON object based on bus data
    if(buffer != null) {
      JSONObject jsonObject = new JSONObject(buffer.toString());
      JSONArray jsonData = jsonObject.getJSONArray("DATA");
      return jsonData;
    }
    else {
      System.err.println("Error code #" + responseCode);
    }
    return null;
  }
}
```

Exercícios

Exercício

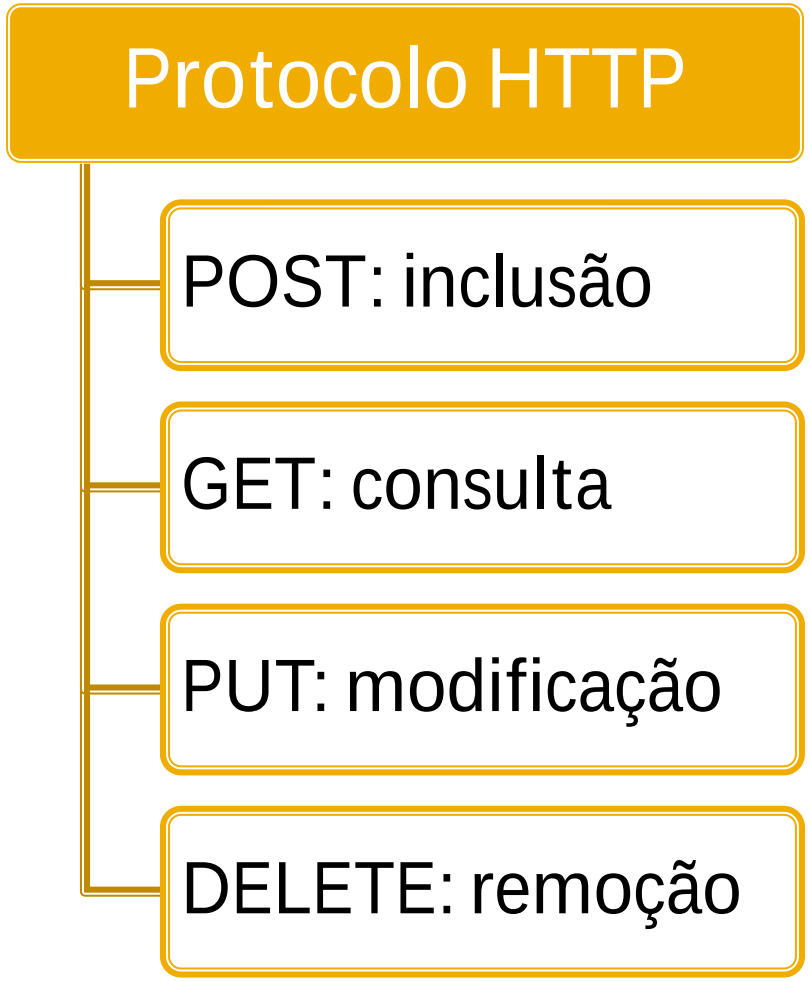
- Crie uma estrutura JSON para representar uma turma de alunos de determinada disciplina
  - Nome da disciplina
  - Nome do professor
  - Nome e matrícula dos alunos
  - 3 notas para cada aluno

Perguntas?

REST

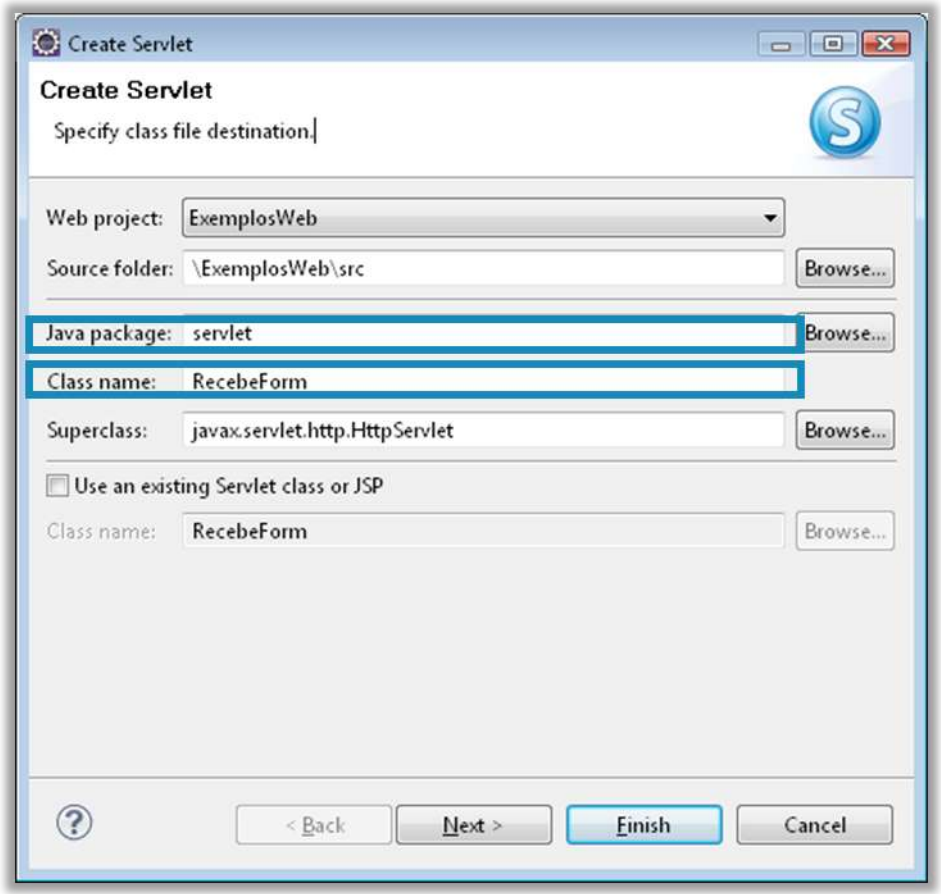
Representational State Transfer

REST: Representational State Transfer



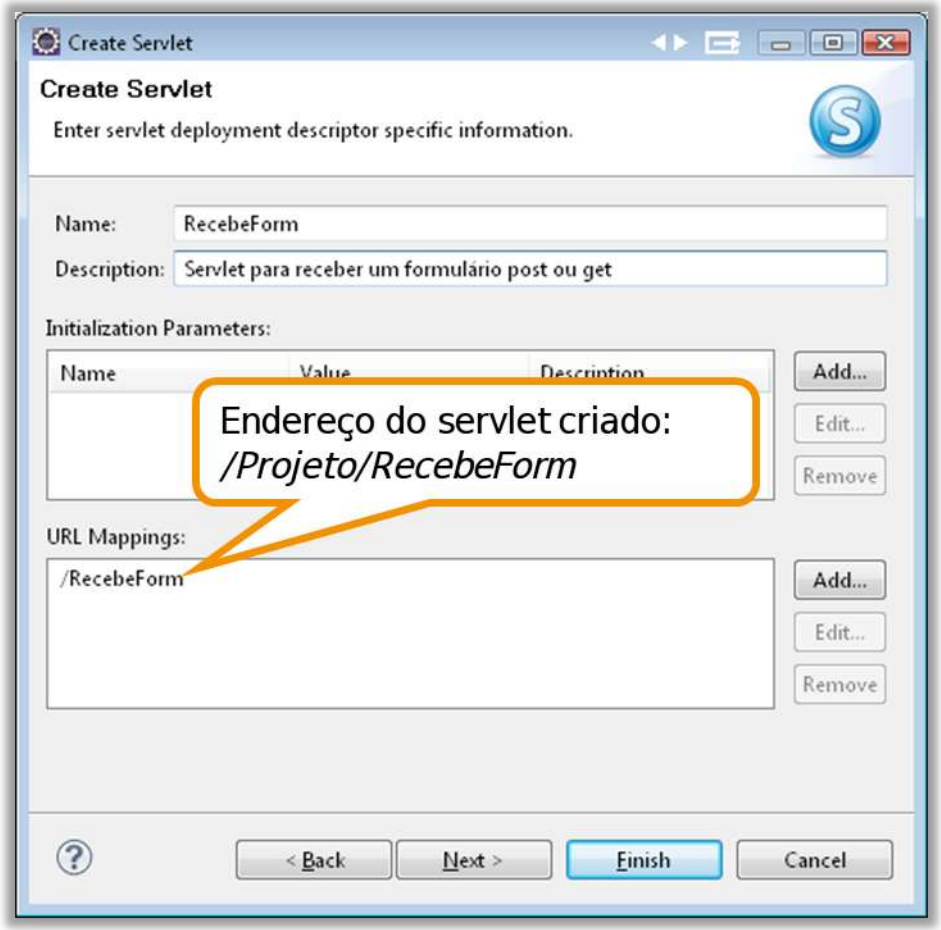
Criando nosso primeiro servlet

- Selecione File → New → Servlet
- Informe o nome do pacote onde o servlet será criado
- Informe o nome da classe (será o nome do arquivo .java também!)
- (normalmente não é necessário modificar a informação de superclasse)
- Clique Next



Criando nosso primeiro servlet

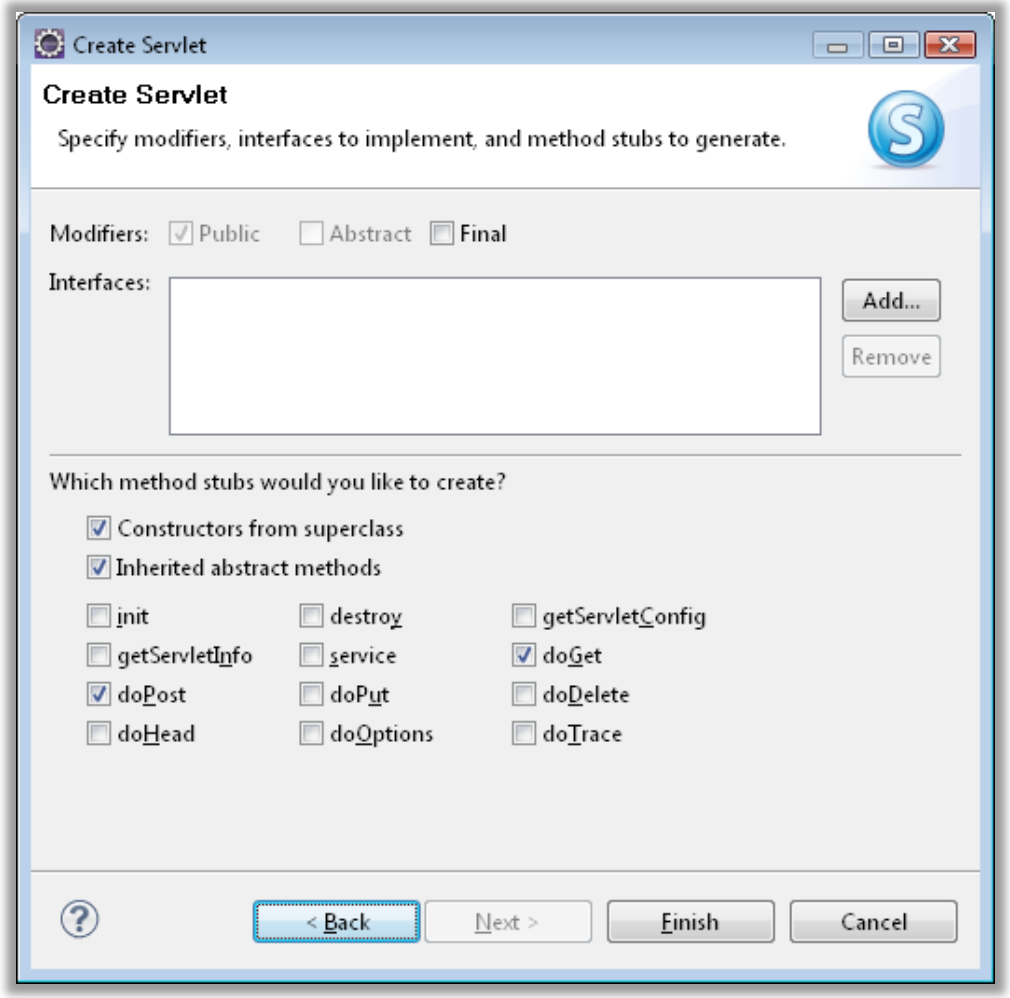
- Informe o nome pelo qual o servlet será conhecido no mundo (normalmente, o mesmo nome da classe)
- Opcionalmente, dê uma descrição ao seu servlet (é apenas um comentário, para seres humanos)
- Clique Next



Criando nosso primeiro servlet

- Informe algumas opções necessárias para o seu funcionamento
- No nosso caso, agora, não preciso de mais nada...

- Clique Finish



Consulta linha de ônibus

Pacotes

```
package br.com.meslin.ufma.servlet;

import java.io.IOException;

import javax.servlet.ServletException;
import javax.servlet.ServletOutputStream;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

import org.json.JSONArray;

import br.com.meslin.ufma.main.GetBuses;
```

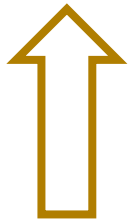
```
/**
 * Servlet implementation class GetBusesByLine
 */
@WebServlet("/GetBusesByLine")
public class GetBusesByLine extends HttpServlet {
  private static final long serialVersionUID=1L;

  // constants
  private static final int LINHA = 2;

    /**
     * @see HttpServlet#HttpServlet()
     */
    public GetBusesByLine() {
        super();
        // TODO Auto-generated constructor stub
    }
```

Consulta linha de ônibus

Métodos de POST e GET – obtenção dos dados

```
  /**
   * @see HttpServlet#doPost(HttpServletRequest
request, HttpServletResponse response)
   */
  protected void doPost(HttpServletRequest
request, HttpServletResponse response) throws
ServletException, IOException {
    // TODO Auto-generated method stub
  }
```

```
/**
 * @see HttpServlet#doGet(HttpServletRequest
request, HttpServletResponse response)
 */
protected void doGet(HttpServletRequest request,
HttpServletResponse response) throws
ServletException, IOException {
  GetBuses buses = new GetBuses();
  JSONArray busesJSON = buses.production();
```



Consulta linha de ônibus

Prepara os dados para enviar para o callback em AJAX

```
  JSONArray busesByLineJSON = new JSONArray();
  for(int i=0; i<busesJSON.length(); i++) {
    JSONArray aBus = busesJSON.getJSONArray(i);
    String linha;
    try {
      linha = Integer.toString(aBus.getInt(LINHA));
    } catch (Exception e) {
      linha = aBus.getString(LINHA);
    }
    if(linha.equals(request.getParameter("linha"))) {
      busesByLineJSON.put(aBus);
    }
  }

  ServletOutputStream out = response.getOutputStream();
  out.print(busesByLineJSON.toString(2));
  out.close();
}
```

Instalando o Postman
(se você ainda não tem)

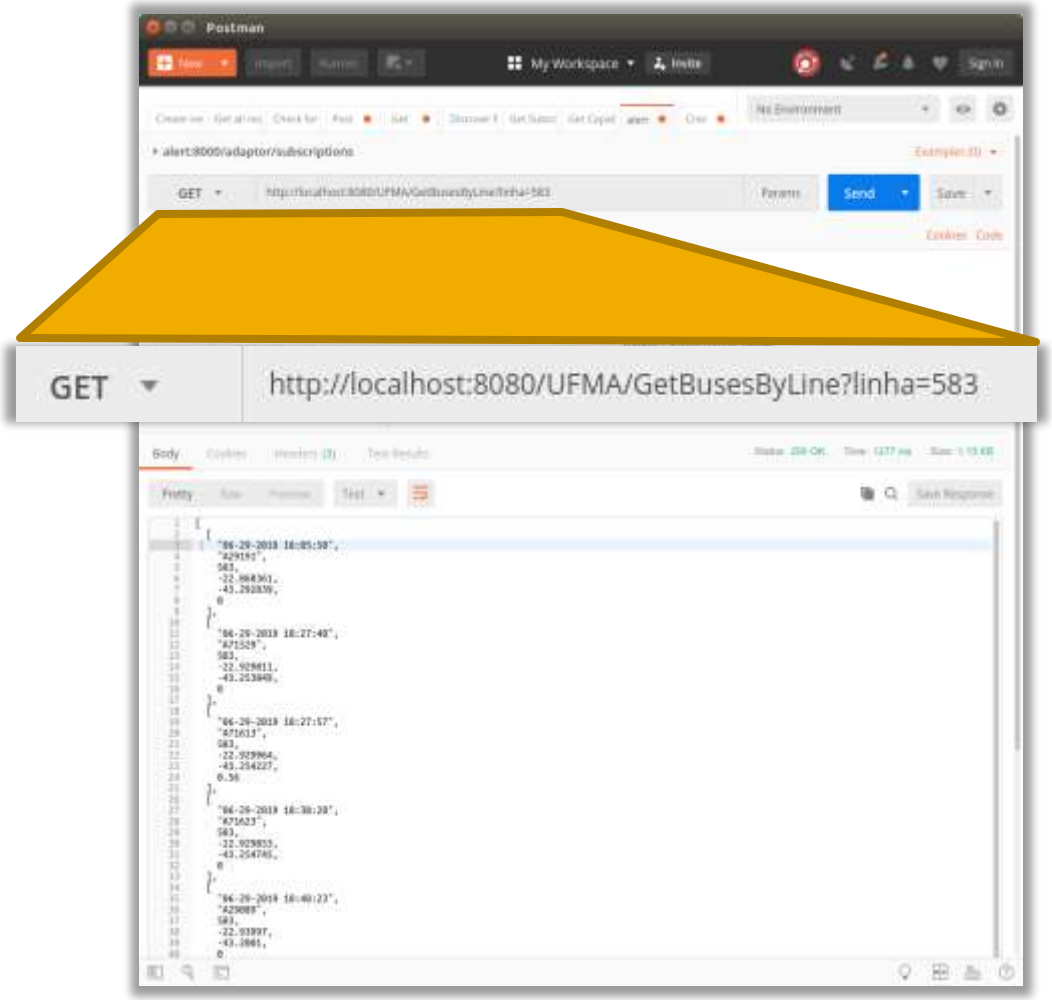
- Fazer download do Postman a partir de https://www.getpostman.com para o diretório de Downloads.

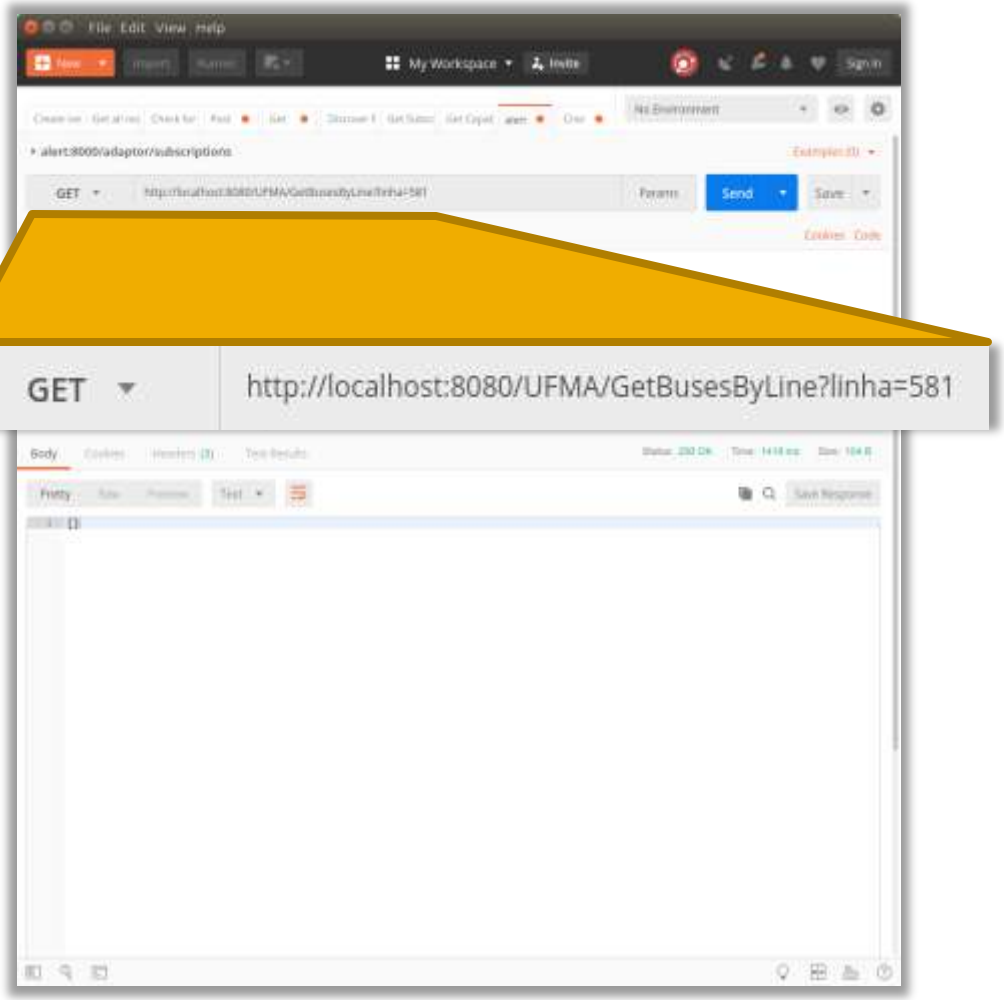
- Descompactar o arquivo, copiar para o diretório /opt e criar o link simbólico com os comandos a seguir (atenção para a versão correta):

```
$ tar -xvzf Postman-linux-x64-6.3.0.tar.gz
$ sudo mv Downloads/Postman /opt/
$ sudo ln -s /opt/Postman/Postman /usr/bin/postman
```

Usando o Postman (Linux ou Windows)

Testando...





Exercícios

Exercício

- Crie um servlet para retornar todos os ônibus que estão se locomovendo entre uma determinada faixa de velocidade passada como parâmetro


Perguntas?

MUSANet

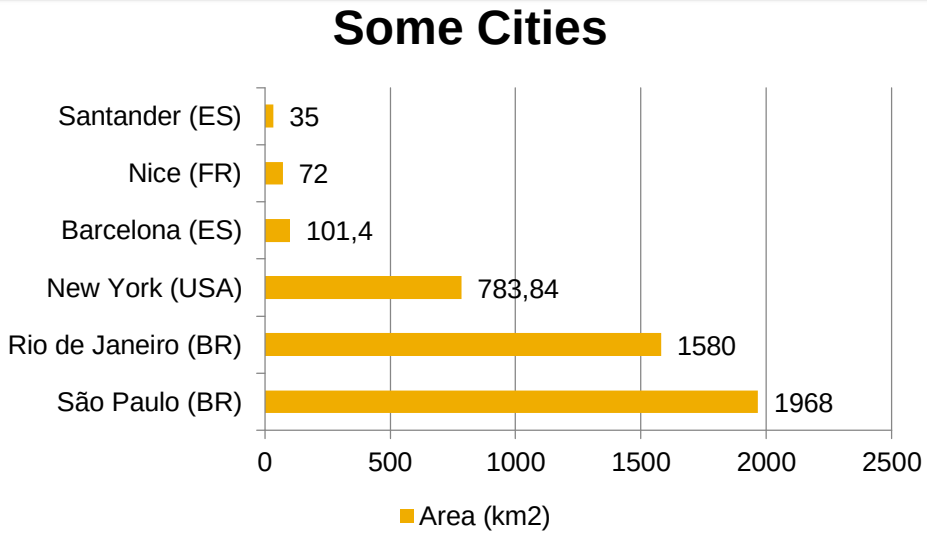
Mobile Urban Sensor and Actutator Network

What's the size of the problem?

Nice - France

Barcelona - Spain

Santander - Spain



Sources:
- Google Maps
- Wikipedia

Sensors used in Smart Cities

- Santander
  - ~ 1,500 sensors
- Barcelona
  - ~ 2,000 sensors
- Rio de Janeiro
  - ~ 8,000 sensors only in buses
- São Paulo
  - ~ 14,000 sensors only in buses

- Other problems:
  - Inhabitants can be sensors and/or actuators
  - Many sensors may be mobile
  - Data must be delivered in a timely manner

Our Goals

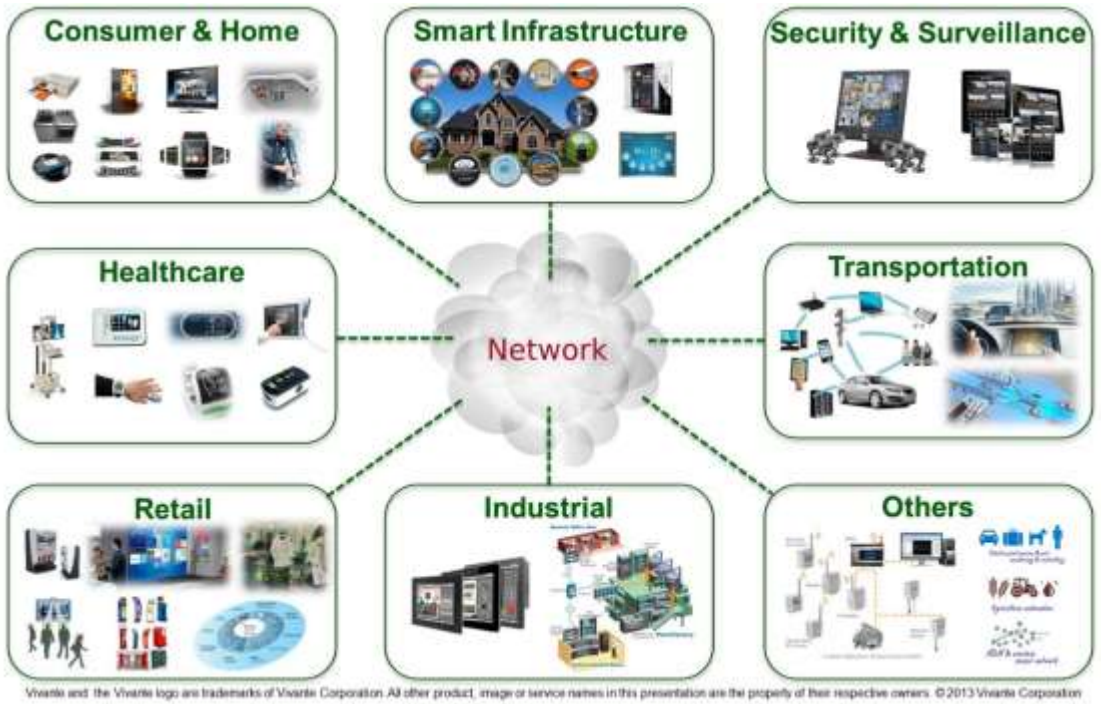
- Target
  - Very large city
  - Lots of sensors and actuators
  - Large population

Mobile!

- Create an middleware to support (large-scale) WSN for smart cities
  - Hierarchical
  - Distributed
  - Cloud-based
  - Context-aware
  - Mobile nodes

Internet of Things (IoT)

- Things = devices with embedded processor, and sensors and/or actuators, able to connect wirelessly



Internet of Things (IoT)
Ou melhor Internet of Mobile Things (IoMT)

- Objects that can move or be moved
- … possibly with few resources and only short/medium range connectivity (Bluetooth, ZigBee)
- … but they should be connected to the Internet whenever possible

Parcel + SmartTags


Domestic Robot


Car


Keychain


Quadcopter/ Drones



Health Sensosrs


Beacons and Sensor Tags

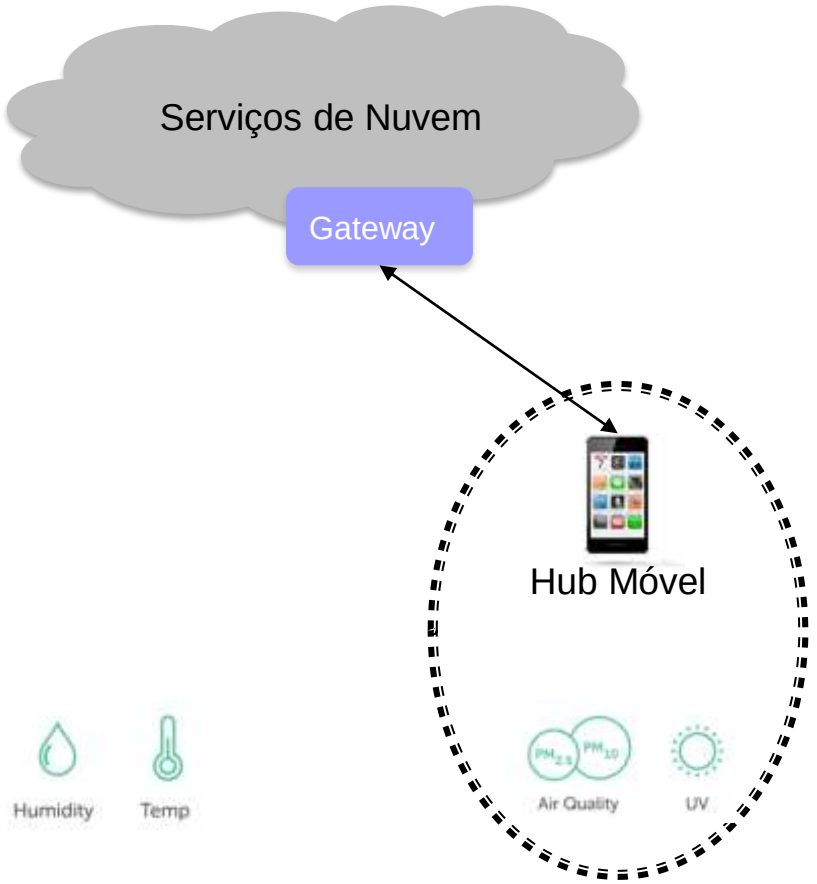


Toys

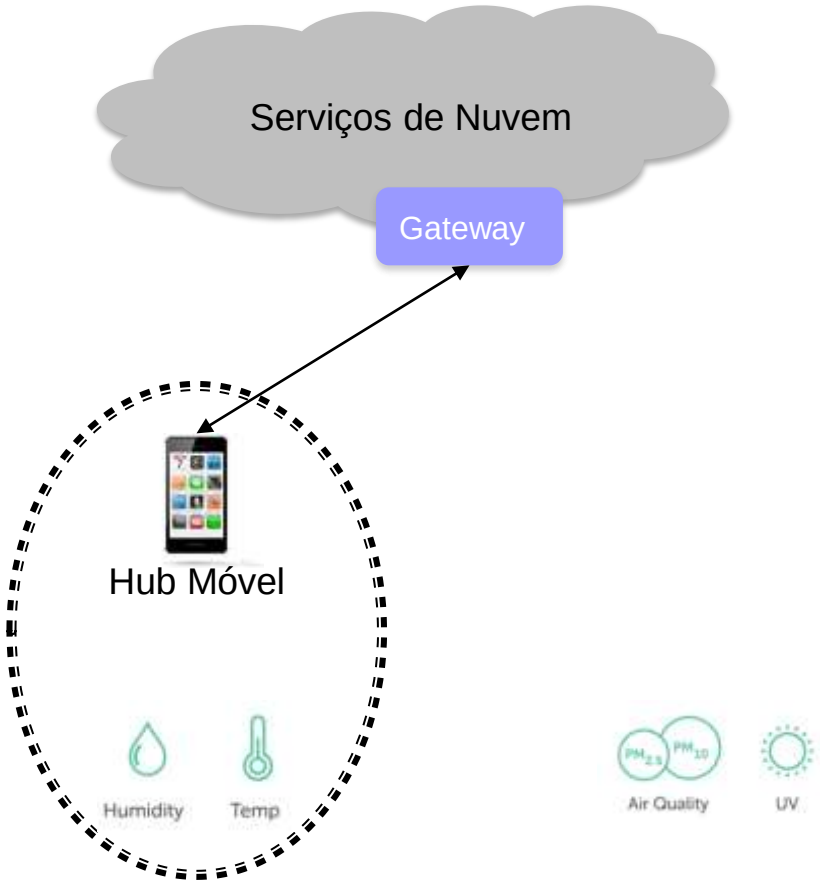
Internet of Mobile Things (IoMT)

- Abordagem: Usar smartphones como intermediários entre coisas inteligentes e serviços na nuvem.
- Grande presença de smartphones:
  - Mais de 2 bilhões de pessoas possuem pelo menos um smartphone. Até 2020 esse número deve alcançar 7 bilhões.
  - Brasil (2016): 38,3 milhões de usuários móveis.
  - Mais de 100 smartphones vendidos por hora no Brasil.
- Possuem:
  - Conectividade Internet (3G/4G/5G & Wi-Fi).
  - Razoável capacidade de processamento e armazenamento.
  - Vários sensores (GPS, acelerômetro, magnetômetro, etc.).
  - Crescente suporte à tecnologias WPAN:
    - Bluetooth 4.0/ Low Energy.
    - NFC.

Cenário em IoMT



Cenário em IoMT



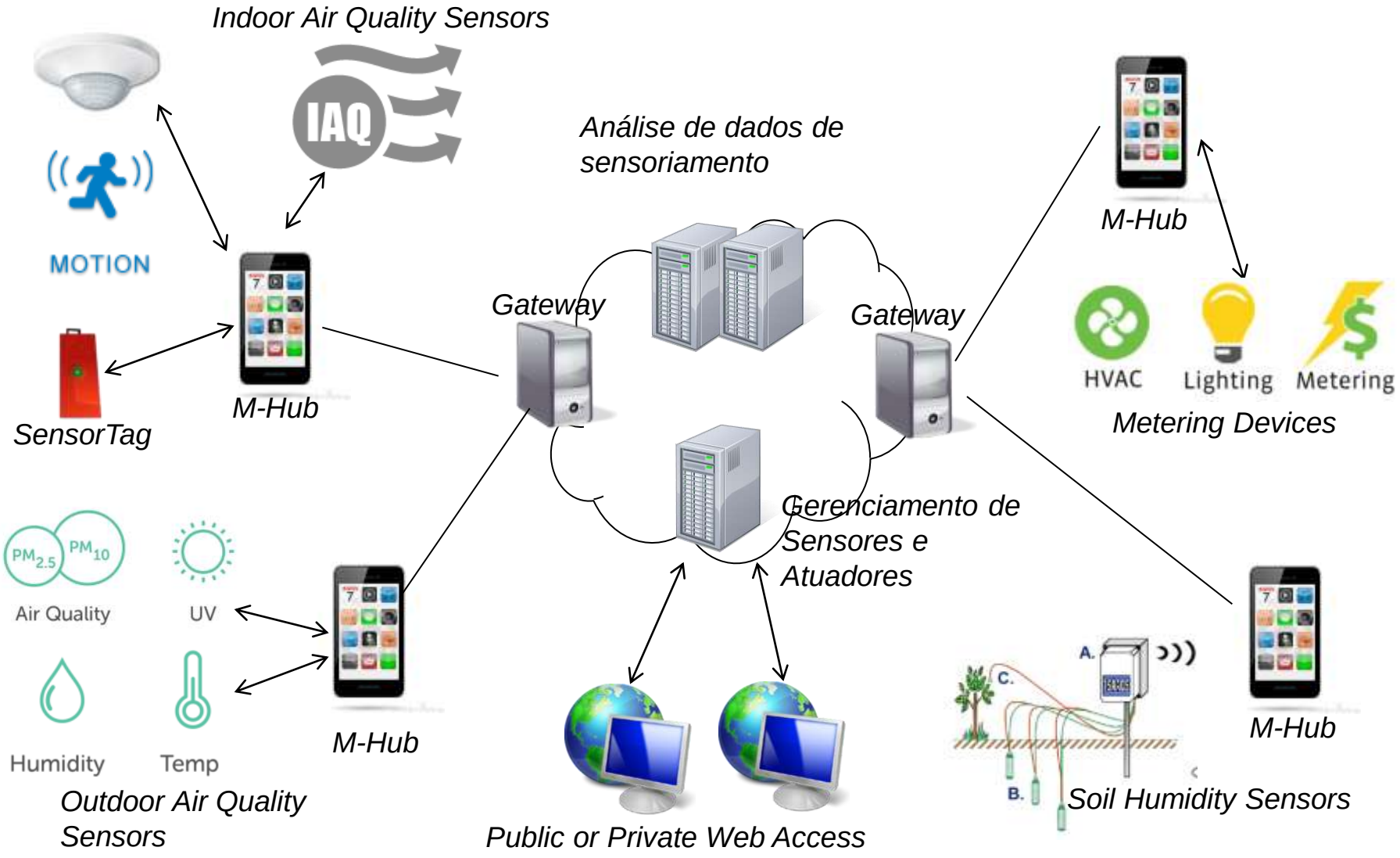
Exemplo de Aplicação IoT/IoMT

- Detecção de focos de incêndio florestal.
- Incêndios elevam ligeiramente a temperatura e reduzem significantemente a umidade relativa do ar.
- Sensores de temperatura e umidade (com BLE) podem ser fixados em placas de sinalização nas estradas que passam pela floresta.
- Carros que passam pela região poderiam enviar os dados coletados para uma central de monitoramento.
- Assim, seria possível detectar o foco do incêndio e por onde está se alastrando.

Aplicações de IoMT

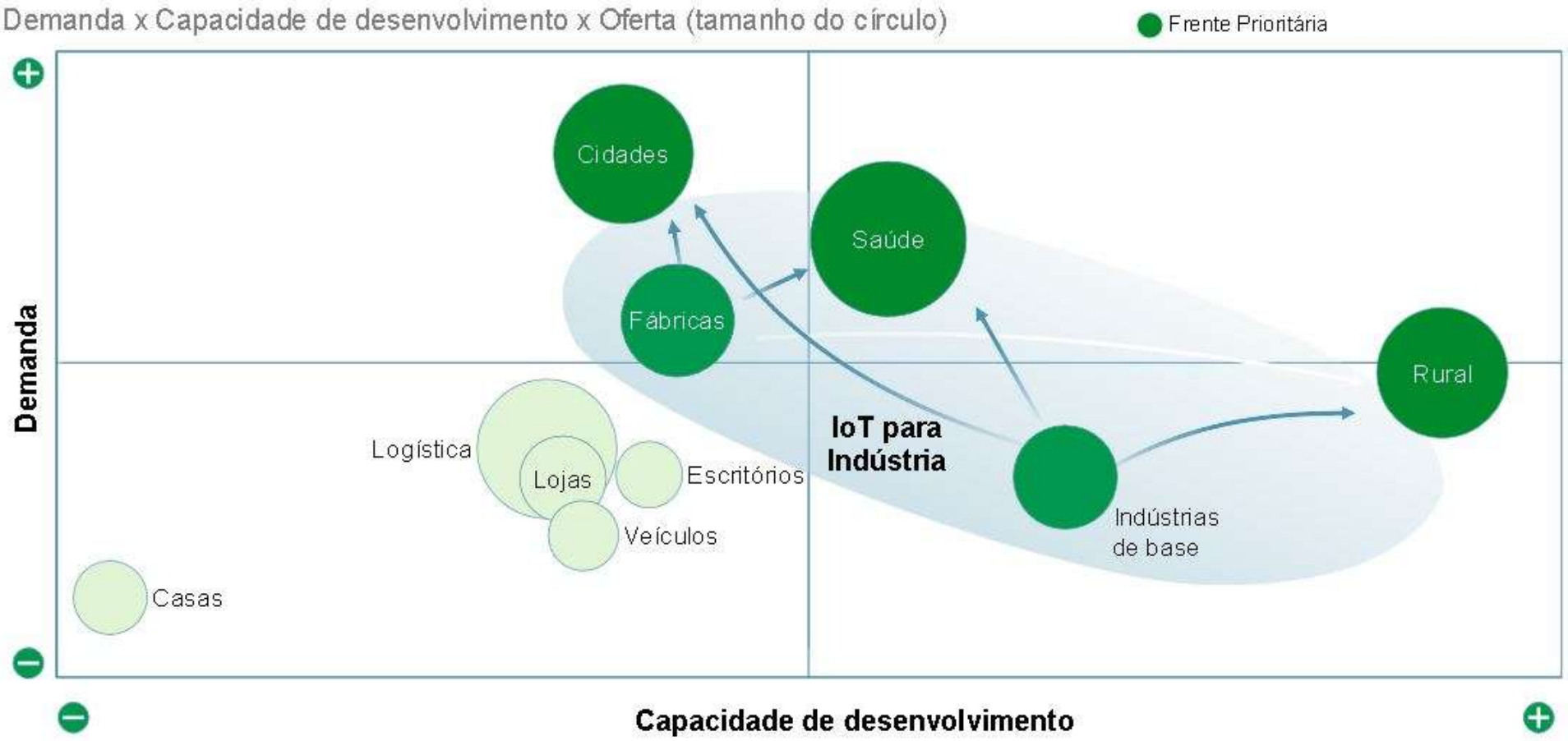
- Sensoriamento ambiental participativo (*crowd sensing*).
- Smart Home / Smart Office.
- Cidades Inteligentes: Intelligent Transportation Systems, Smart Parking, etc.
- Healthcare.
- Logística.
- Varejo.
- Lazer.
- Smart Manufacturing com robôs móveis.

IoMT : Sensoriamento Participativo e Oportunístico



Plano Nacional de IoT para o Brasil (2018-2020)

A matriz de priorização destacou quatro Frentes Prioritárias de IoT para o país

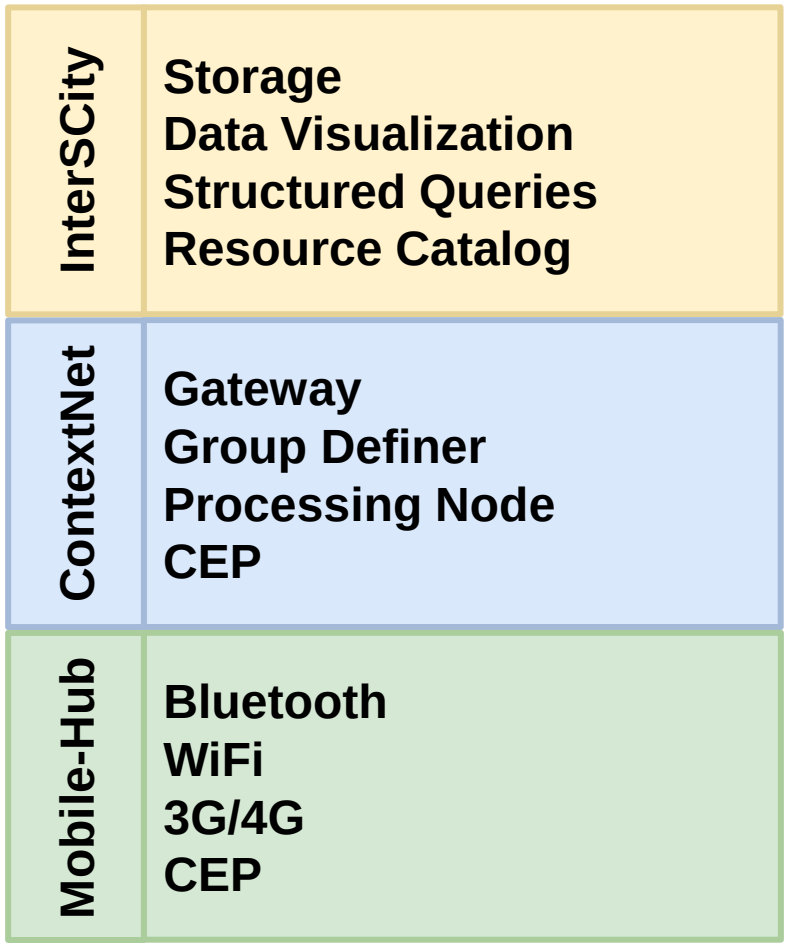


FONTE: MGI; PINTEC; Avaliação de especialistas independentes; Avaliação de participantes da Câmara IoT; Análise do consórcio

MUSANet Layers

MUSANet – Mobile Urban Sensor and Actuator Network

Three-layer architecture



Lower Layer: Mobile Hub

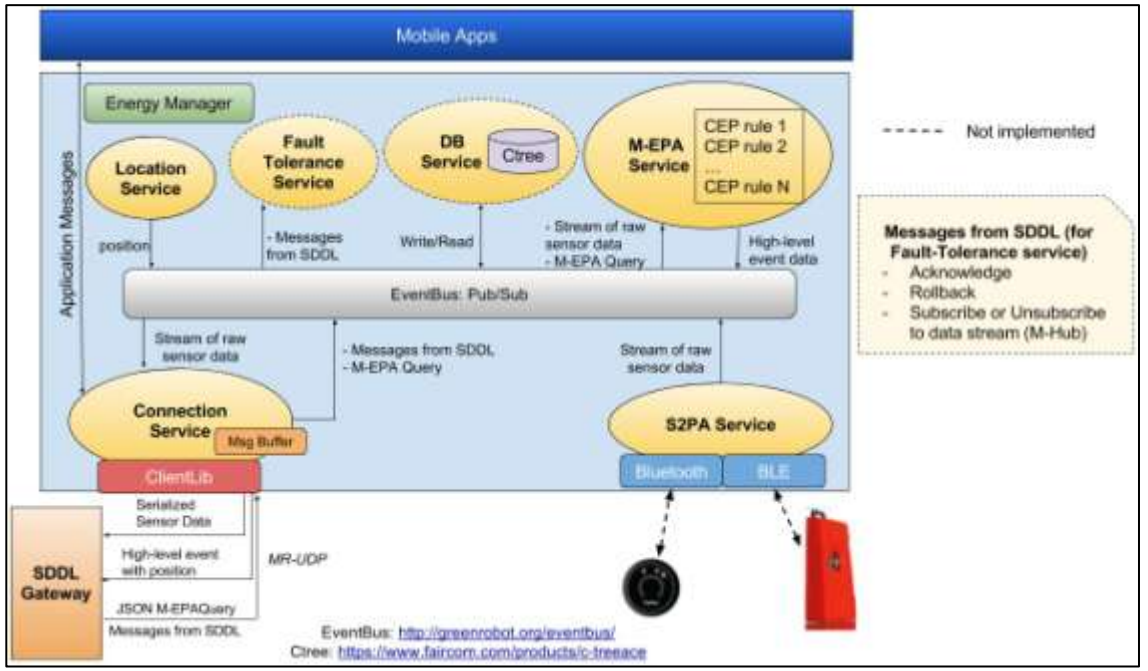
Componente do middleware que executa em Android e provê:

- **Comunicação** com Gateways do SDDL e outros M-Hubs.
- **Descoberta de smart objects** nas proximidades (usando Bluetooth Classic e BLE).
- Serviço de **data stream processing** (para filtragem & sumarização, agregação de dados de sensores) e tomada local de decisão.
- Possibilita adição de **infomação contextual** aos dados coletados(tempo, posição, etc.) provida pelos sensores embutidos no smartphone.
- Serviço de **gerenciamento de energia** parametriza a execução dos demais serviços (frequência de *scan* para descoberta, comunicação com Internet, etc.).
- Serviço de persistência de dados (DBS) e serviço de gerenciamento de atuação (M-ACT) sobre smart objects (em desenvolvimento).

Lower Layer: Mobile Hub

Sensor discovery and data collection

- Sensor gateway
  - Bluetooth
  - BLE
- CEP (Complex Event Processing)
- Context-aware

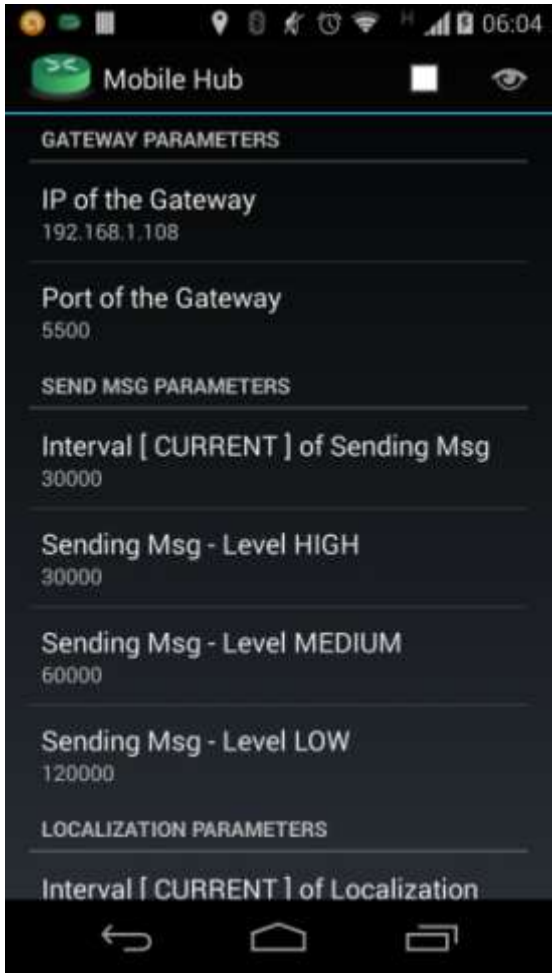


Mobile Hub

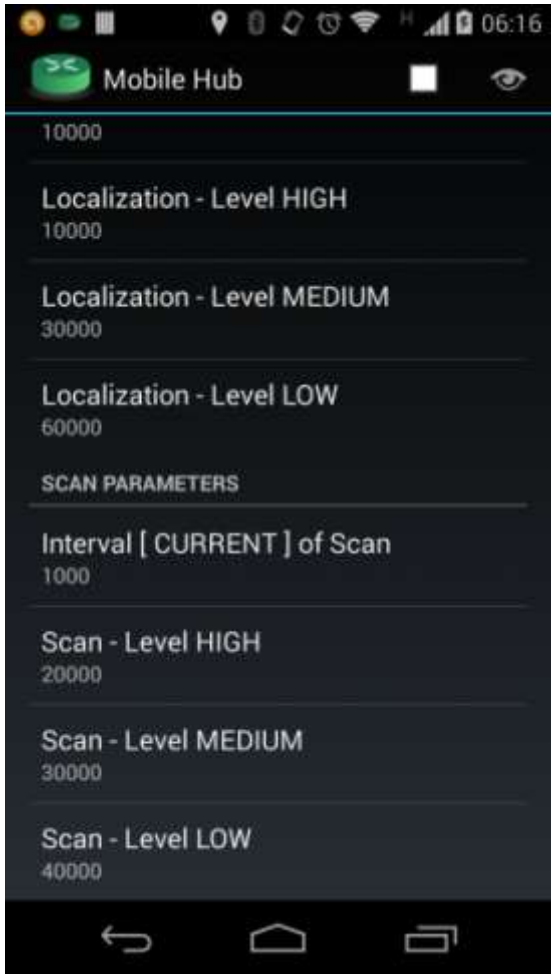
CC2541 SensorTags (Textas Instruments)

M-Hub conectado ao Zephyr BioHarness-3

Interface do Mobile Hub



Parâmetros de Conexão



Parâmetros de Scan

Objetos Conectados

Exemplo de Aplicação do Mobile-Hub

3 M-OBJs próximos ao M-Hub-1

1 M-OBJ próximo ao M-Hub-2

Middle Layer: ContextNet

Objetivos:

- Arquitetura de software escalável para comunicação móvel e processamento com baixa latência.
- Processamento de mobile data streams.
- Complex Event Processing (CEP) distribuído.
- Ciência de Contexto e adaptação.
- Suporte a colaboração e coordenação mobile-mobile.

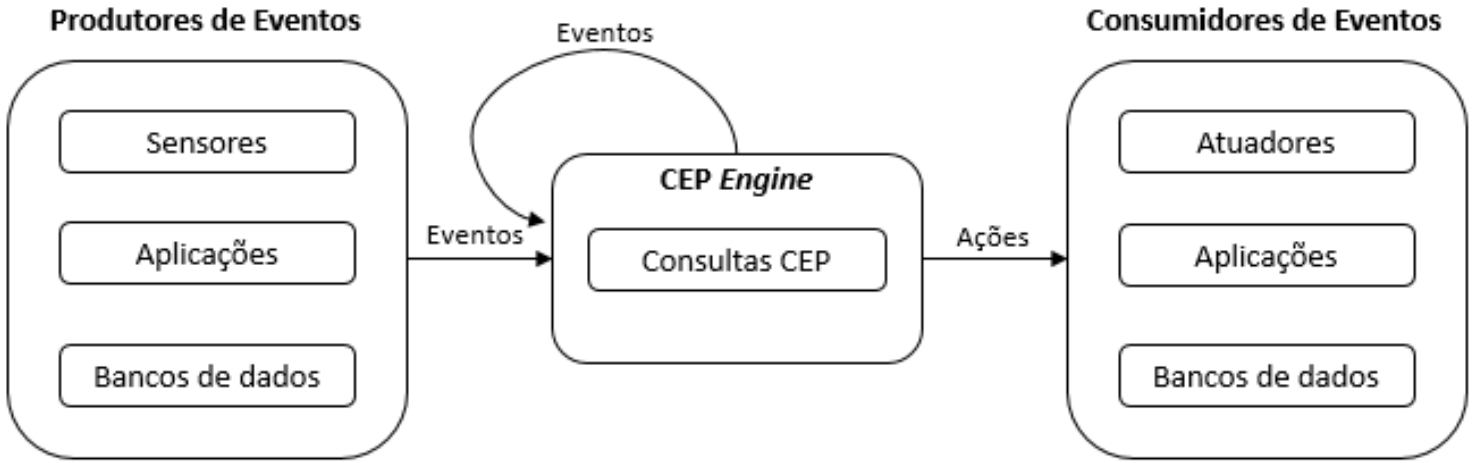
O middleware mobile-nuvem (Scalable Data Distribution Layer – SDDL) provê:

- comunicação escalável mobile-mobile, multi-ponto e com confiabilidade.
- balanceamento de conexões e data streams, handover transparente.

Usado em vários protótipos de pesquisa.

Complex Event Processing

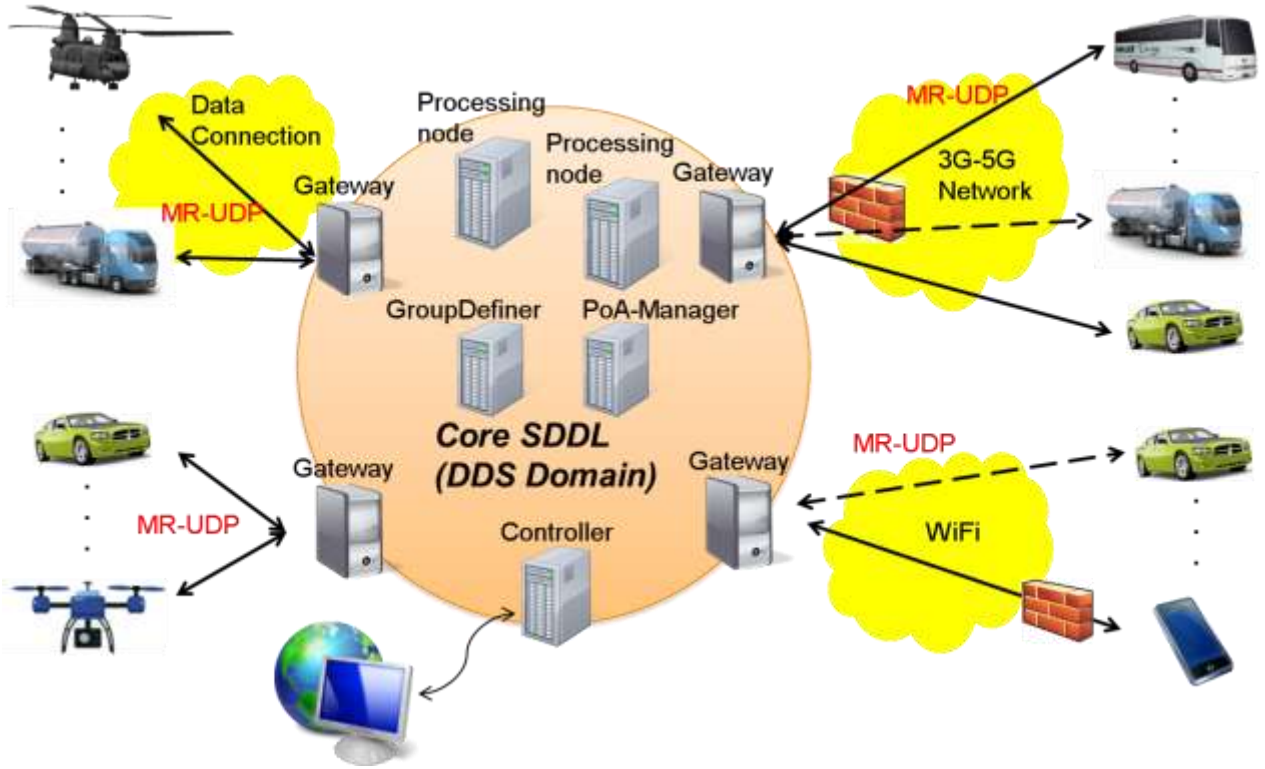
- Paradigma (e Tecnologia) para analisar e manipular **fluxos de eventos** utilizando consultas contínuas (on-line).
- Linguagem CQL (*Continuous Query Language*):
  - SELECT * FROM **SensorData**.win:time(5 s) WHERE **sensorName** LIKE '%temperature%' AND **latitude** > -22 AND **latitude** < -21 AND **longitude** > -43 AND **longitude** < -42
  - Expressividade para elaboração de consultas.



ContextNet

Reliable wireless communication between mobile and cloud

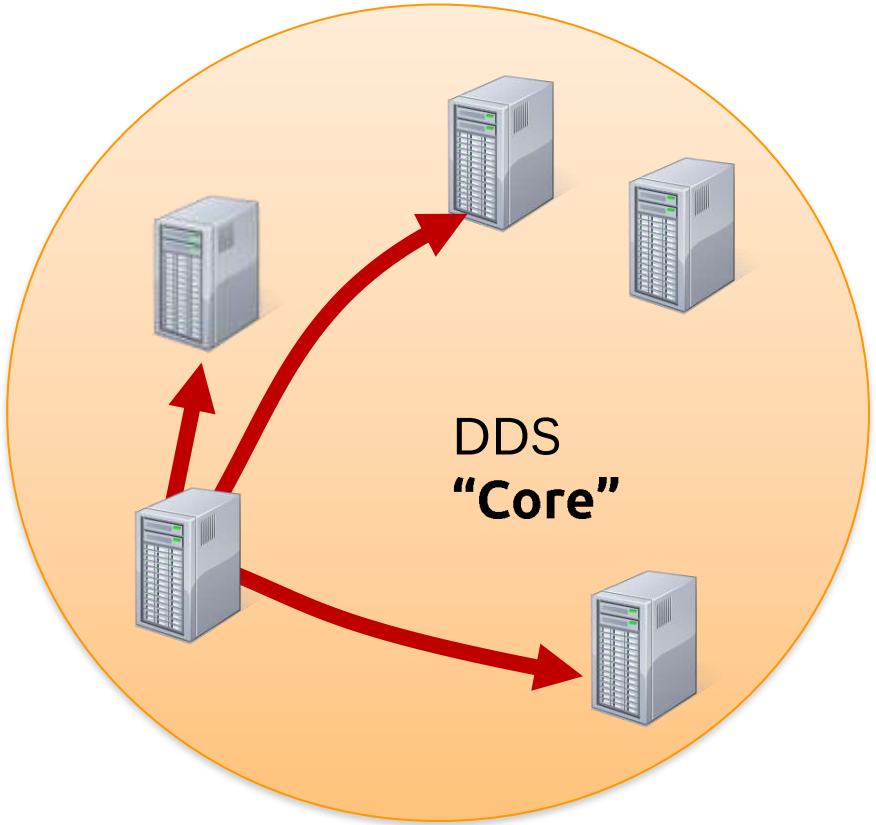
- SDDL Core
- Mobile nodes
  - MR-UDP over IP



M. Endler et al, “ContextNet: context reasoning and sharing middleware for large-scale pervasive collaboration and social networking,” in Proceedings of the Workshop on Posters and Demos Track. ACM, 2011, p. 2.

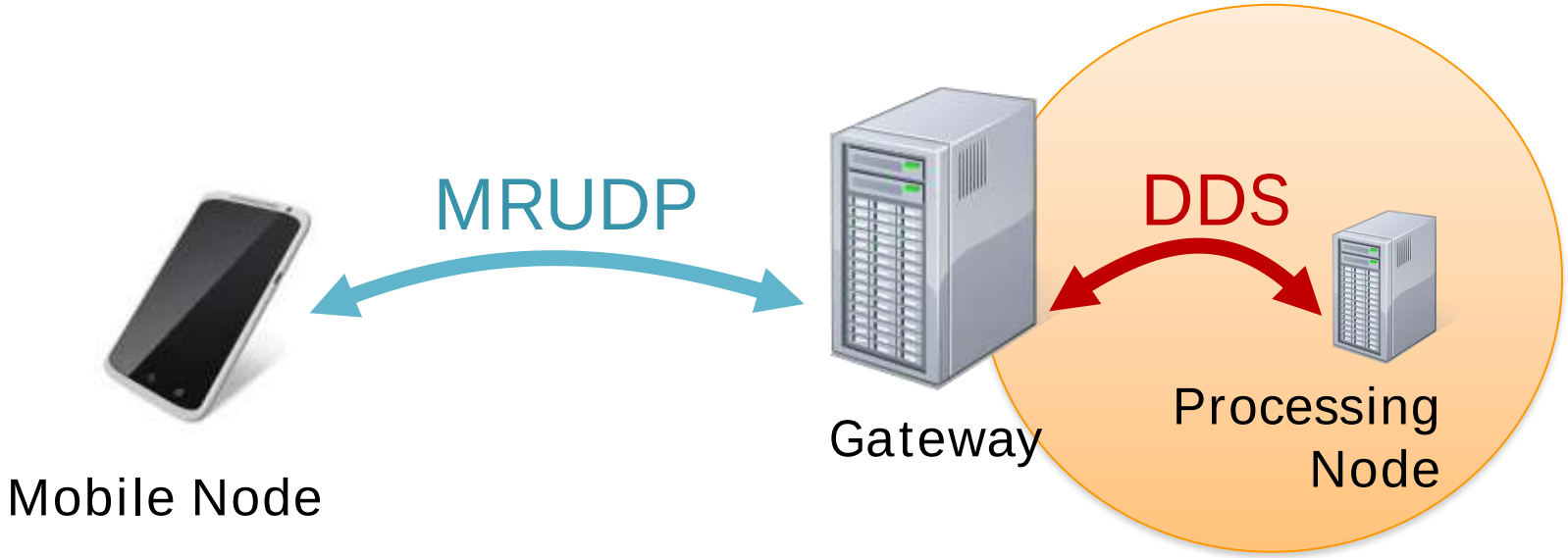
Data Distribution Service™

- Especificação OMG de um middleware Publish-Subscribe para comunicação em tempo real e de alta vazão em sistemas de larga escala.
- DDS define uma arquitetura P2P completamente descentralizada.

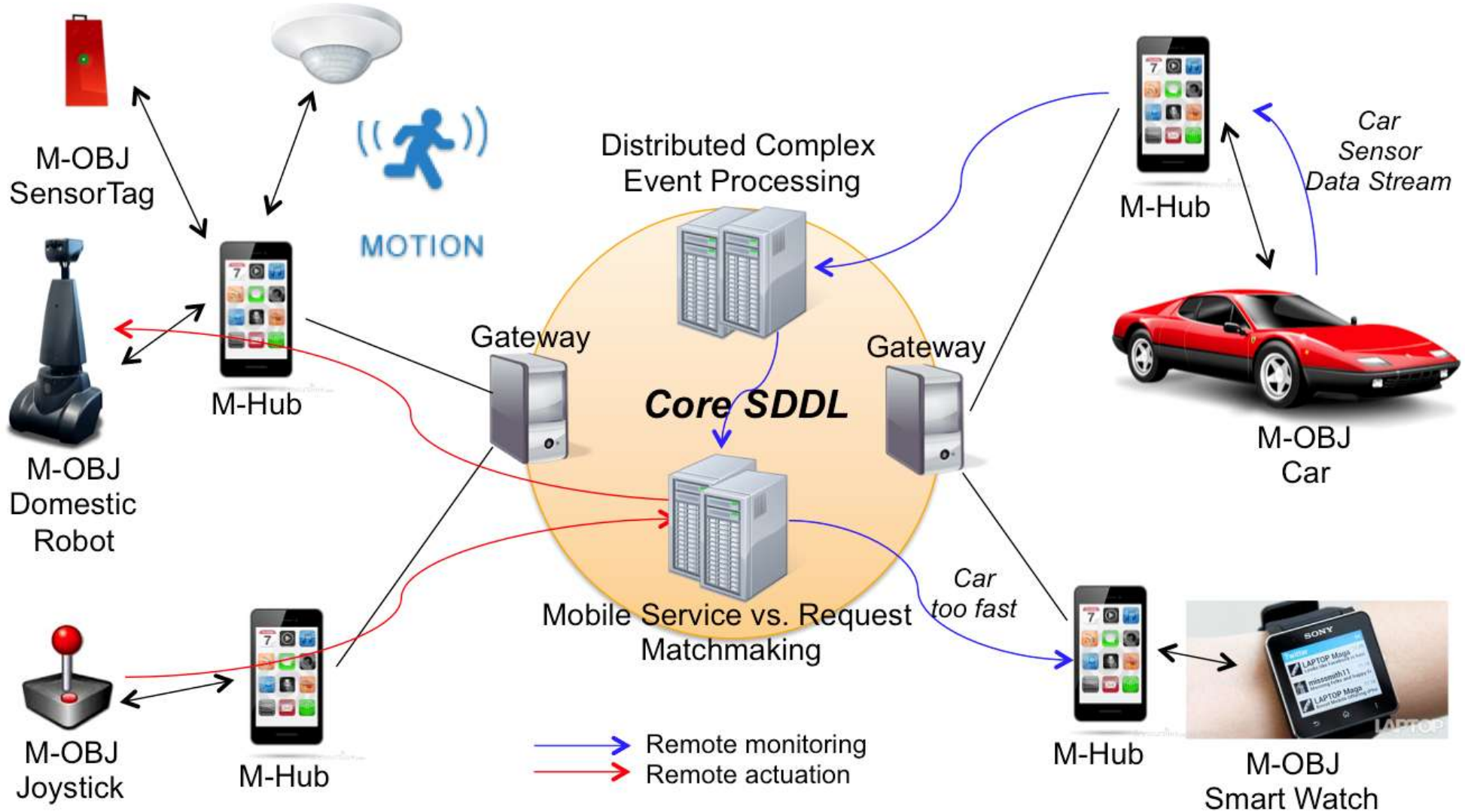


Protocolos

- MRUDP – Mobile Reliable UDP.
- DDS – Data Distribution Service.



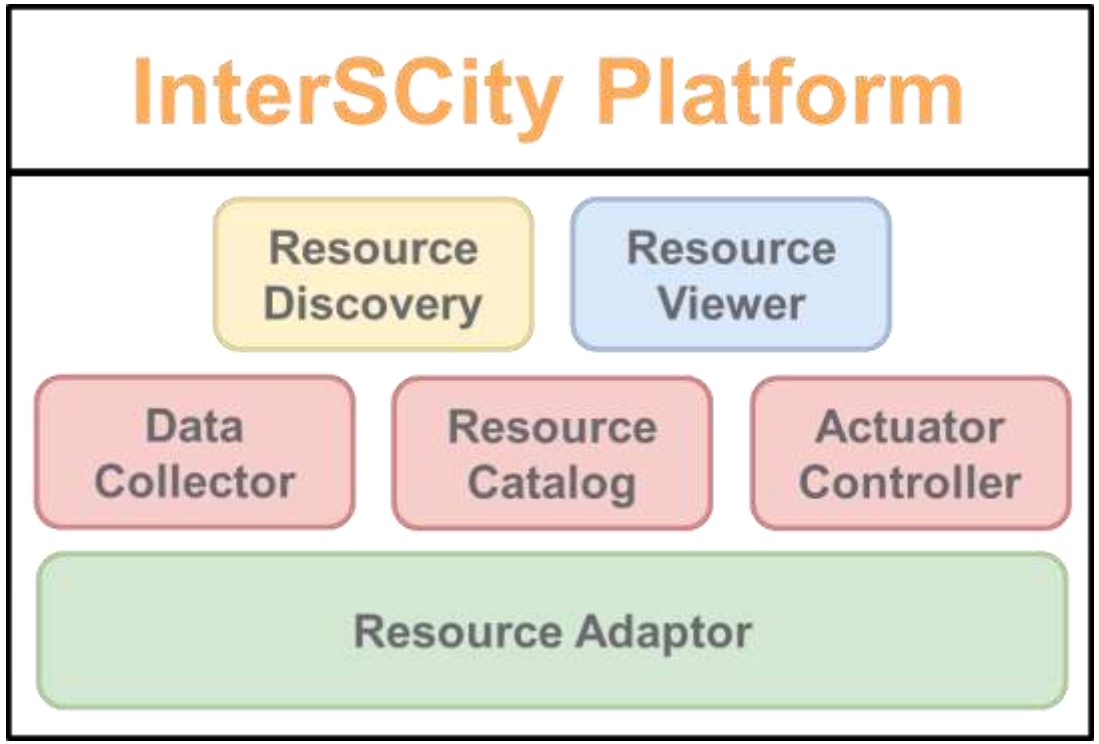
ContextNet: Arquitetura



Upper Layer: InterSCity

Model based storage and retrieval of sensor data

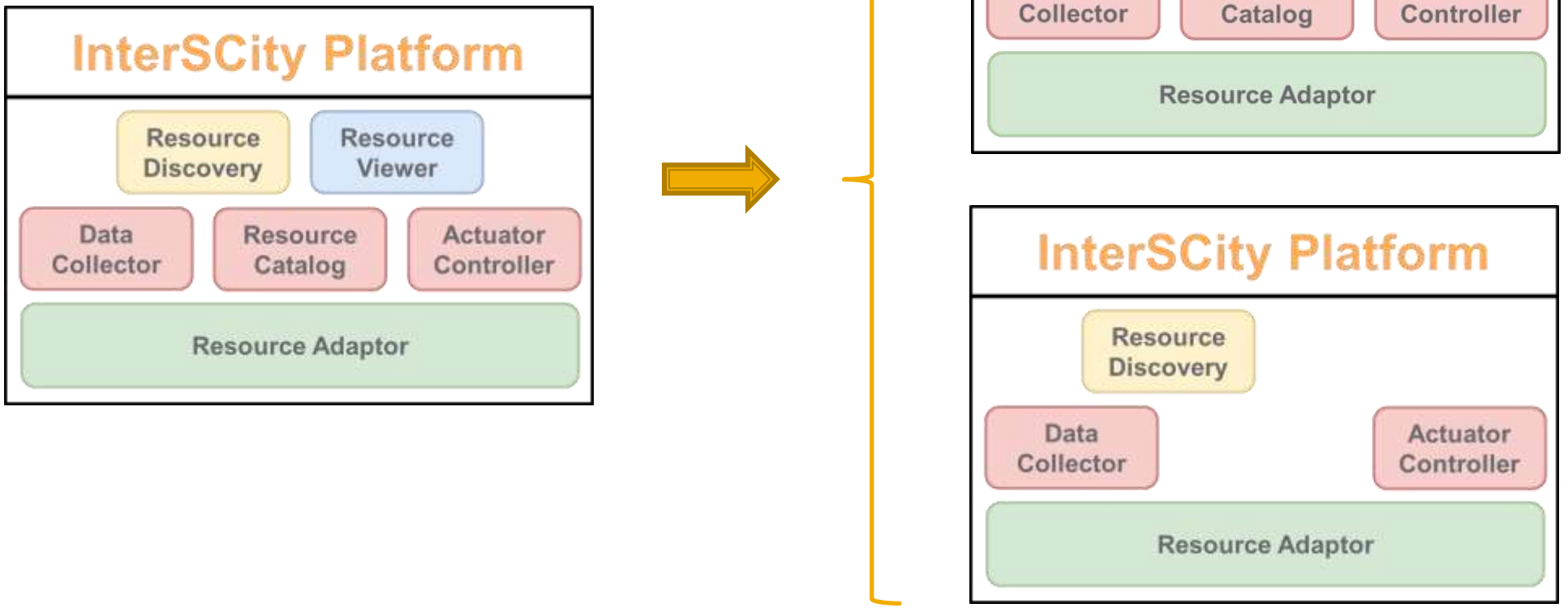
- Microservices
- Access over HTTP (Intranet)
- Structured queries
- Parallel execution of the platform in clusters



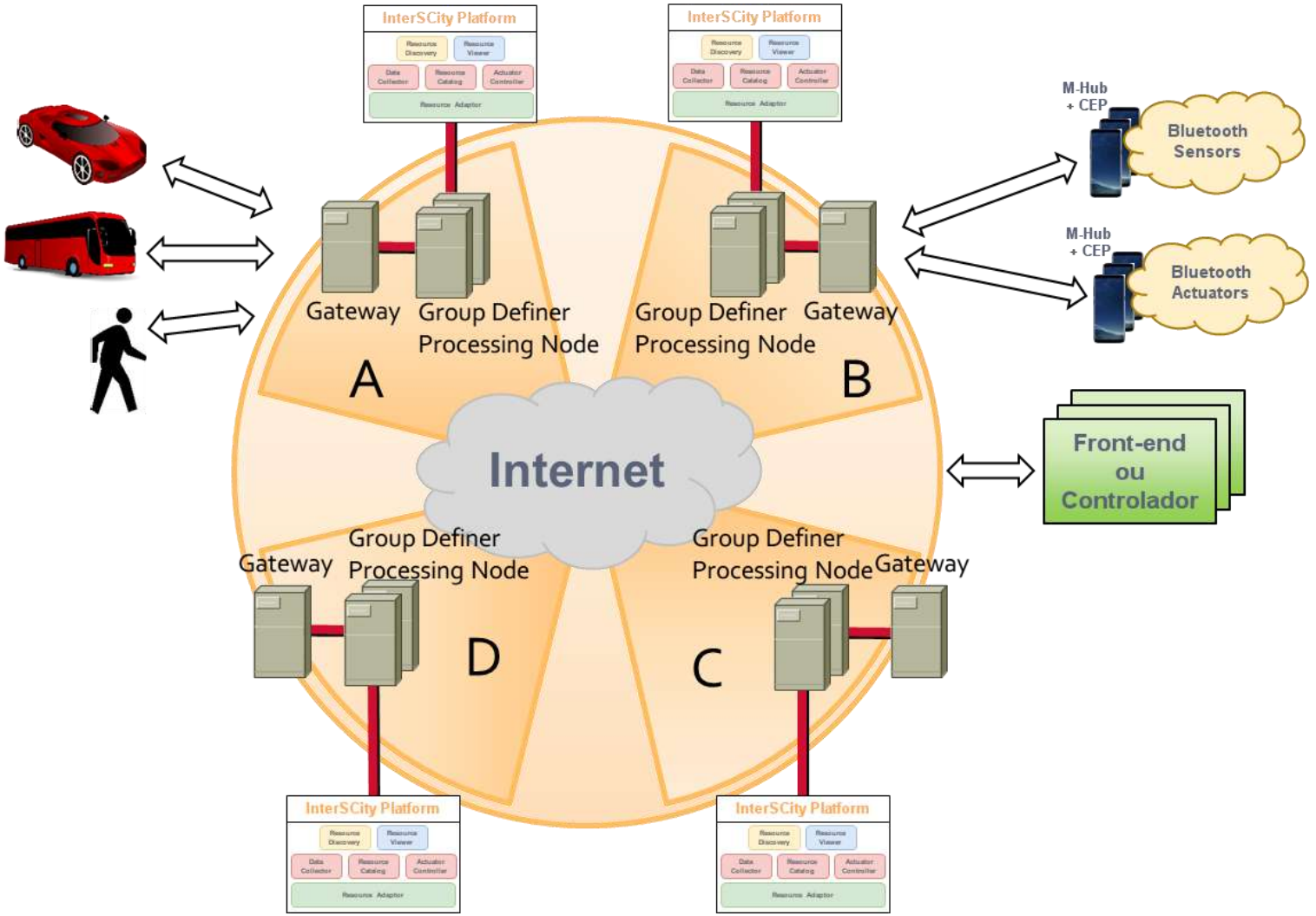
D. M. Batista et al., “InterSCity: Addressing Future Internet Research Challenges for Smart Cities,” in Network of the Future (NOF), 2016 7th International Conference on the. IEEE, 2016, pp. 1–6.

InterSCity

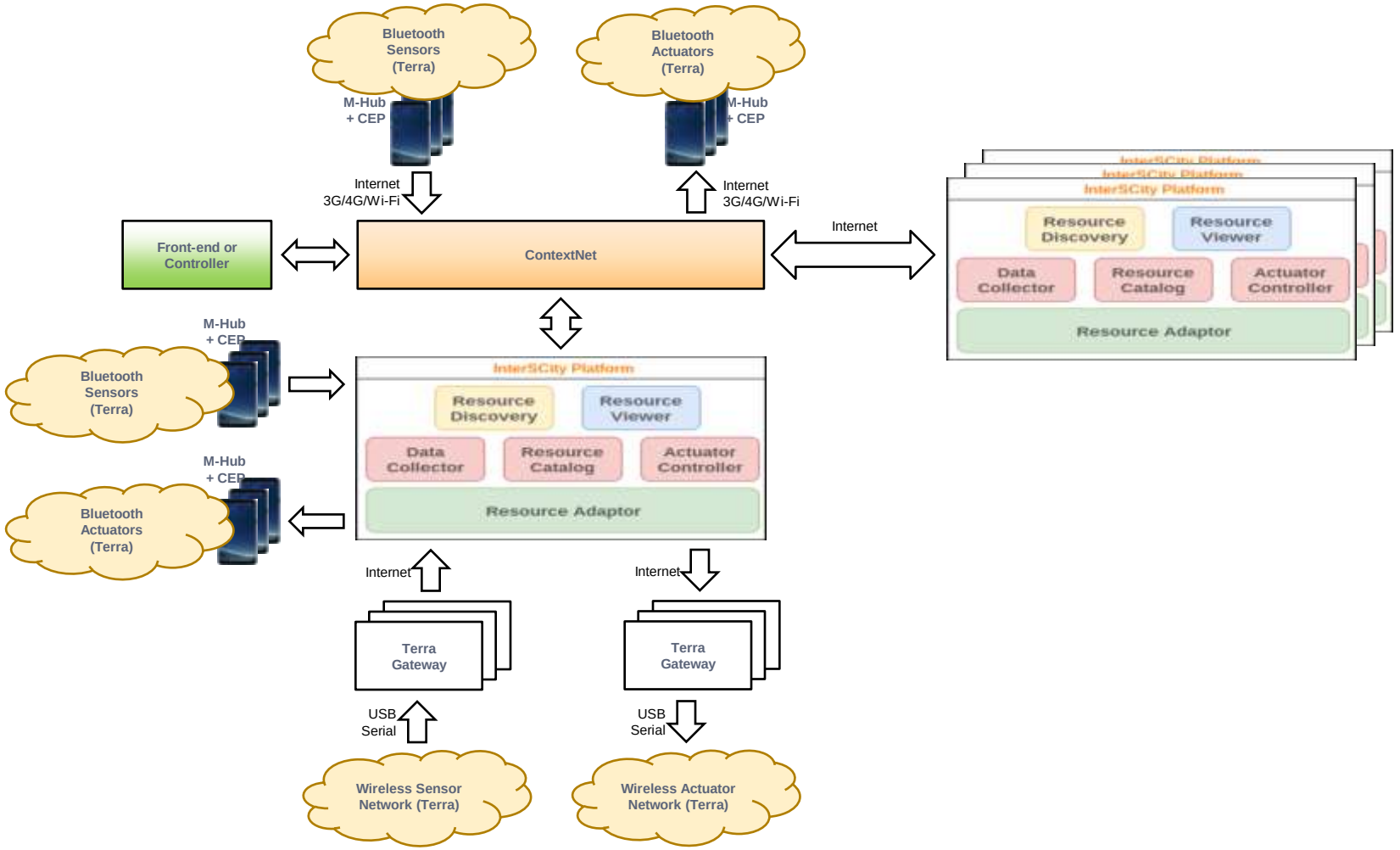
- Microservice replication



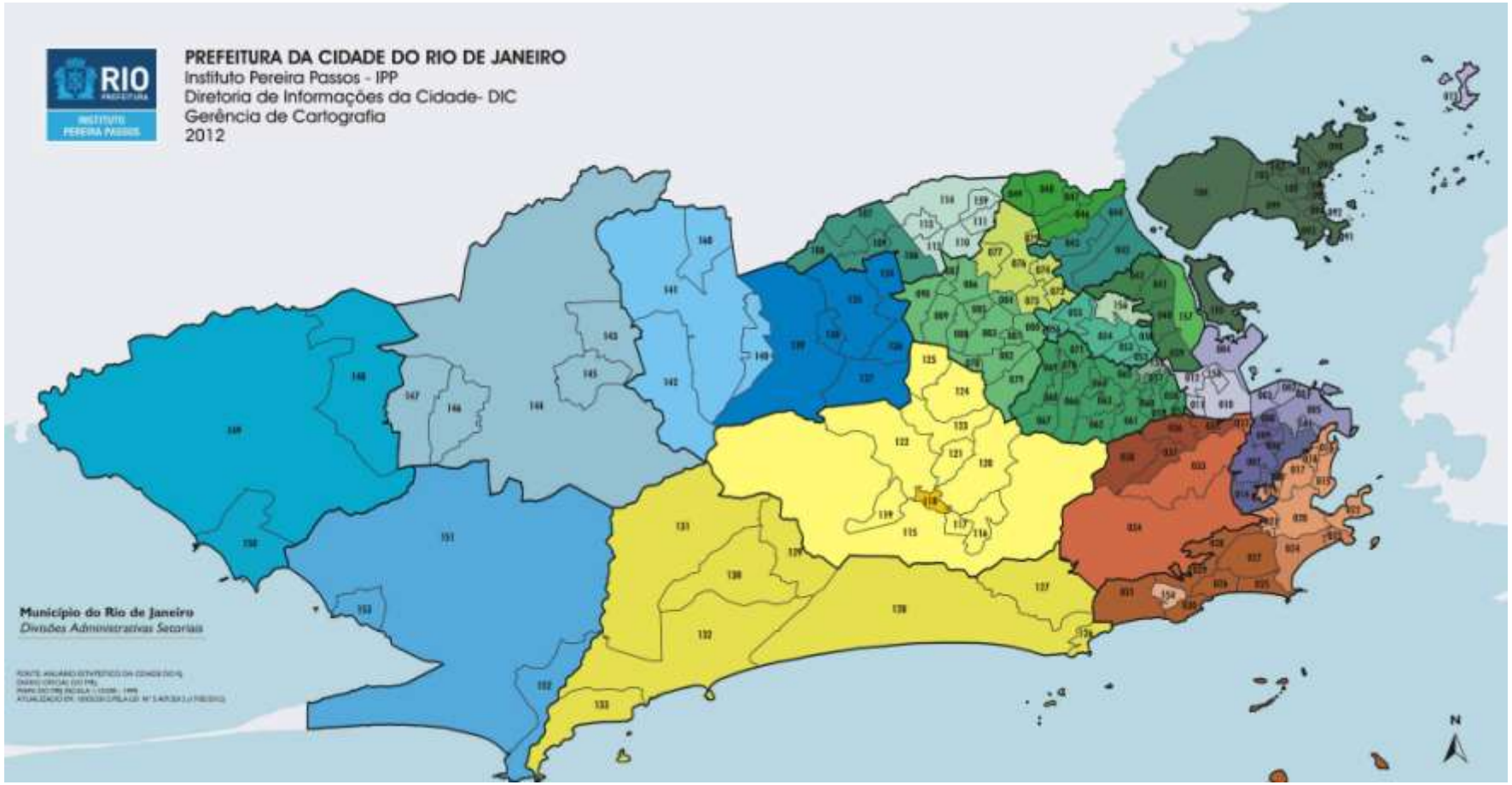
MUSANet


InterSCity Platform
InterSCity Platform
M-Hub
+ CEP
Bluetooth
Sensors
M-Hub
+ CEP
Bluetooth
Actuators
Gateway
Group Definer
Processing Node
Group Definer
Processing Node
Gateway
A
B
Internet
Front-end
ou
Controlador
Gateway
Group Definer
Processing Node
Group Definer
Processing Node
Gateway
D
C
InterSCity Platform
InterSCity Platform

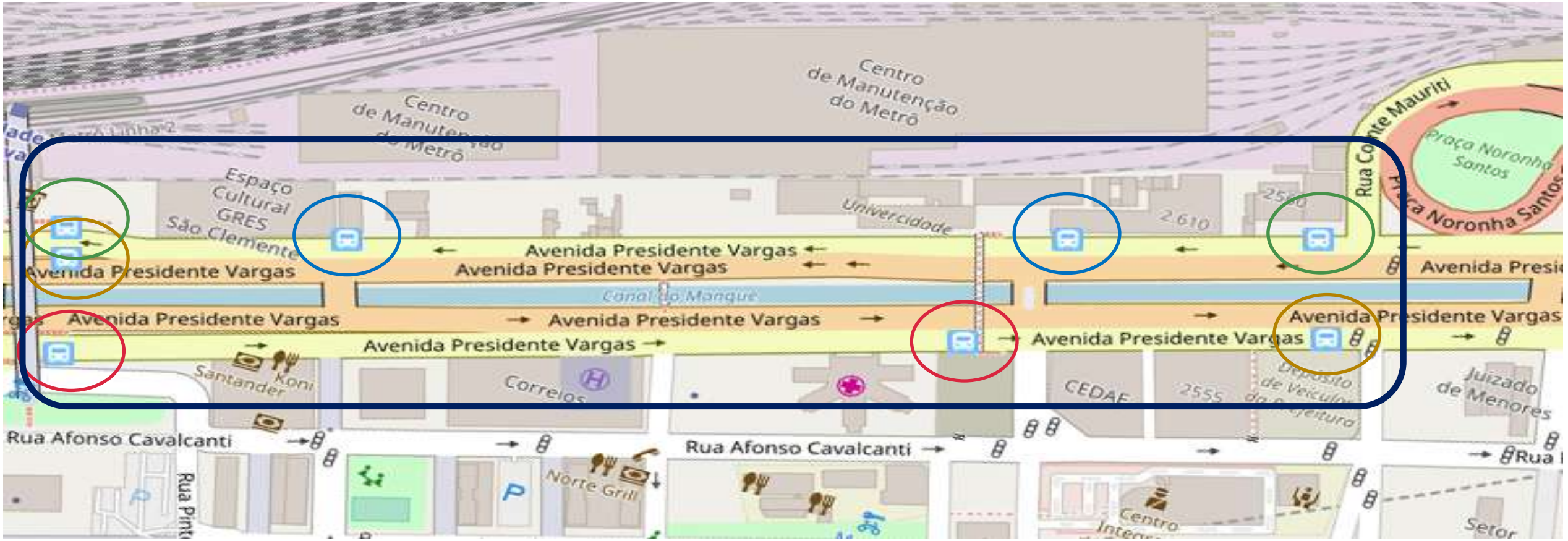
MUSANet


Bluetooth Sensors (Terra)
Bluetooth Actuators (Terra)
M-Hub + CEP
M-Hub + CEP
Internet 3G/4G/Wi-Fi
Internet 3G/4G/Wi-Fi
Front-end or Controller
ContextNet
Internet
InterSCity Platform
Resource Discovery
Resource Viewer
Data Collector
Resource Catalog
Actuator Controller
Resource Adaptor
M-Hub + CEP
Bluetooth Sensors (Terra)
M-Hub + CEP
Bluetooth Actuators (Terra)
InterSCity Platform
Resource Discovery
Resource Viewer
Data Collector
Resource Catalog
Actuator Controller
Resource Adaptor
Internet
Internet
Terra Gateway
Terra Gateway
USB Serial
USB Serial
Wireless Sensor Network (Terra)
Wireless Actuator Network (Terra)

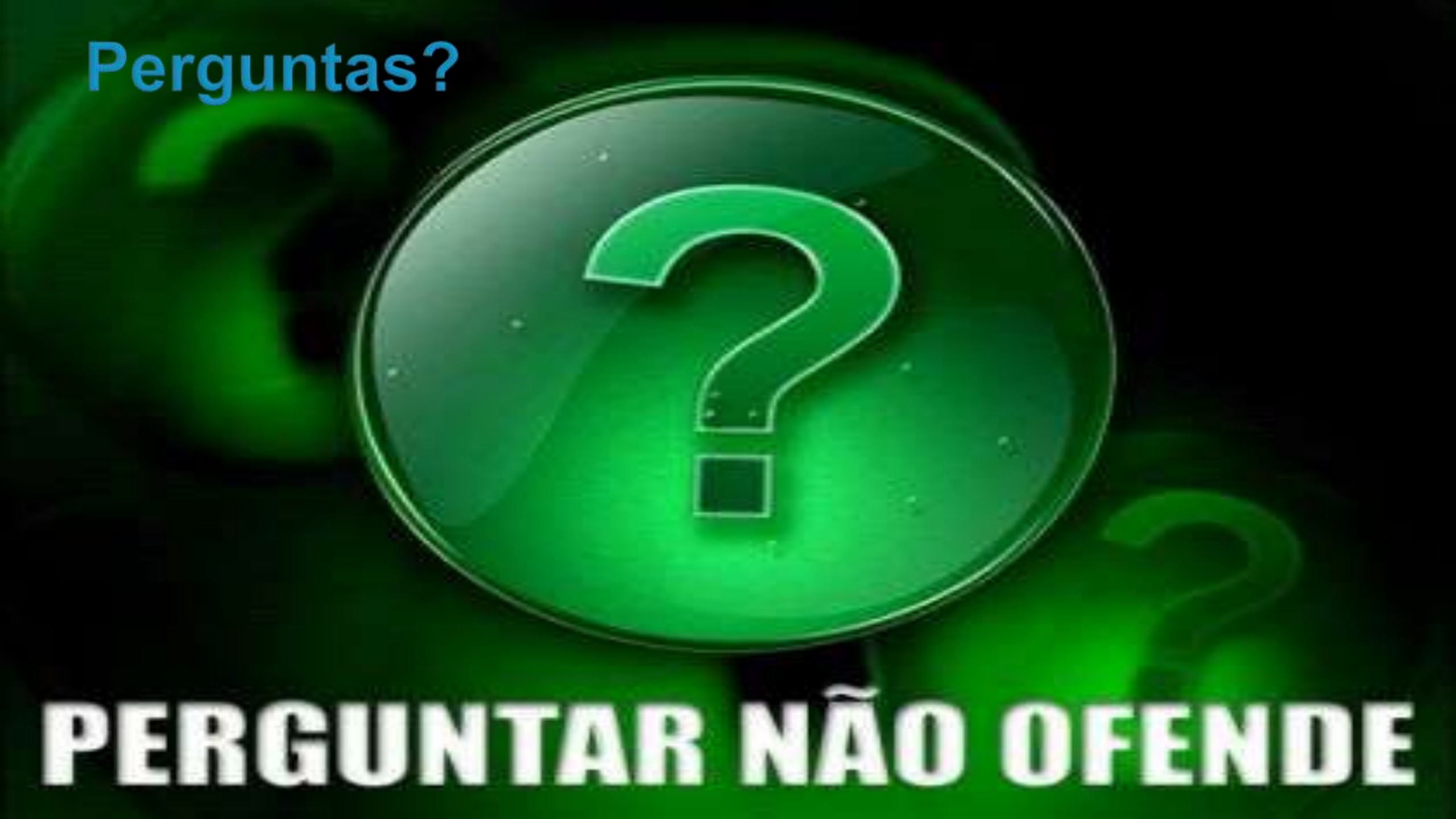
Rio de Janeiro e seus bairros



Use Case: Controlling Bus Schedule



Perguntas?



Instalando e conhecendo a plataforma InterSCity

http://interscity.org/

Slides InterSCity

- Os slides a seguir foram extraídos da apresentação de Arthur Del Esposte, desenvolvedor do InterSCity
  - https://www.youtube.com/watch?v=kLxIUbVFyOA&t=420s
  - https://docs.google.com/presentation/d/1QGjurOAoCO0vdEc0ZuBJoAy8Vi8TkpmczvzoKFmWMPc/edit#slide=id.g228574380c_0_0

Horizontal Solutions

Transportation Systems

Health Care Systems

Traffic Management Systems

City Management Systems

Smart City Platform

* Icons from Flaticon

Definition of Smart City Platform

"An integrated middleware environment that supports software developers in designing, implementing, deploying, and managing applications for Smart Cities." - (Santana et al, 2016)

Opportunities

Proprietary Solutions

Lack of Validations

Poor Reusability

Ad-hoc Solutions

Reusable Open Software & Data

Practical and Scientific Validations

Flexible and Evolvable Architecture

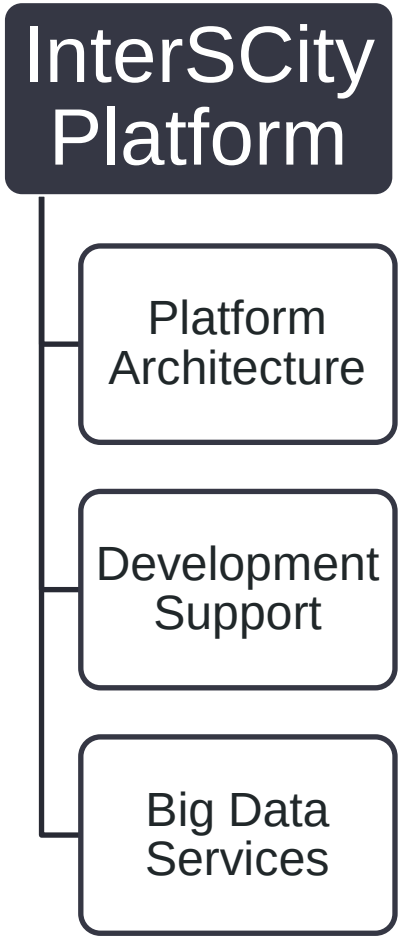
Multidisciplinary Research

The InterSCity Platform

- The InterSCity platform is an **open-source microservice-based** project to enable novel smart city research, development, and deployment initiatives
- It provides a set of high-level **cloud-based services** that mediate the communication between the applications and the city devices

https://gitlab.com/groups/smart-city-software-platform

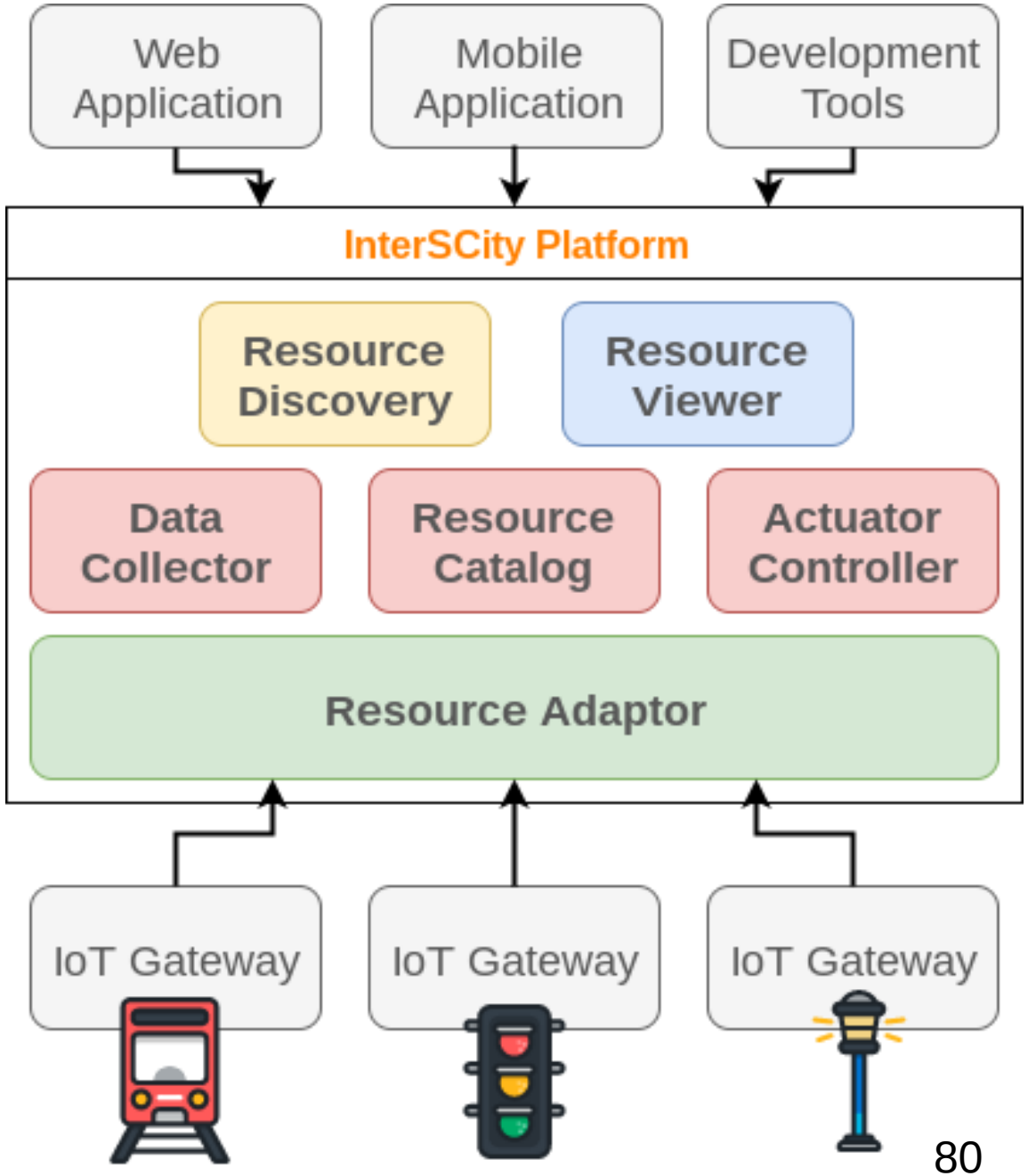
Research Lines



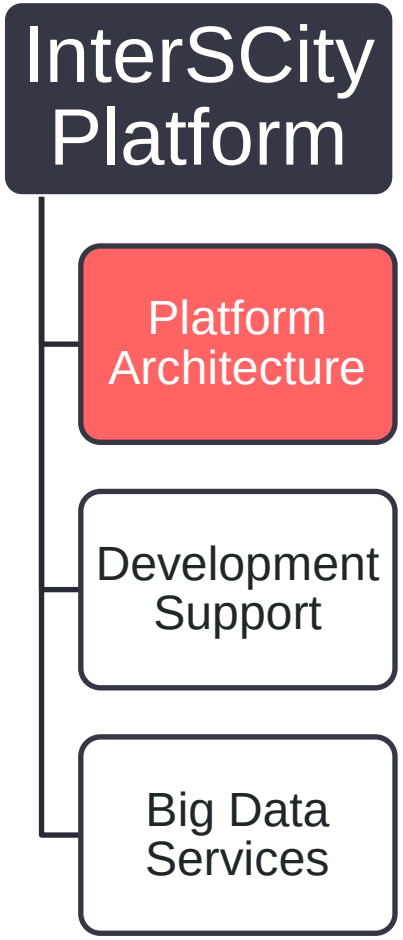
InterSCity Platform

- Platform Architecture
- Development Support
- Big Data Services

InterSCity Platform overview, boundaries, and microservices



Why Microservices for Smart Cities?



- Microservices is a software architecture style composed of a collection of loosely coupled, fine-grained services
- Refining traditional SOA guidelines
- Microservices communicate with lightweight mechanisms
- **Support several non-functional requirements (mainly scalability and evolvability)**
- **Horizontal scalability through functional decomposition**

InterSCity Platform

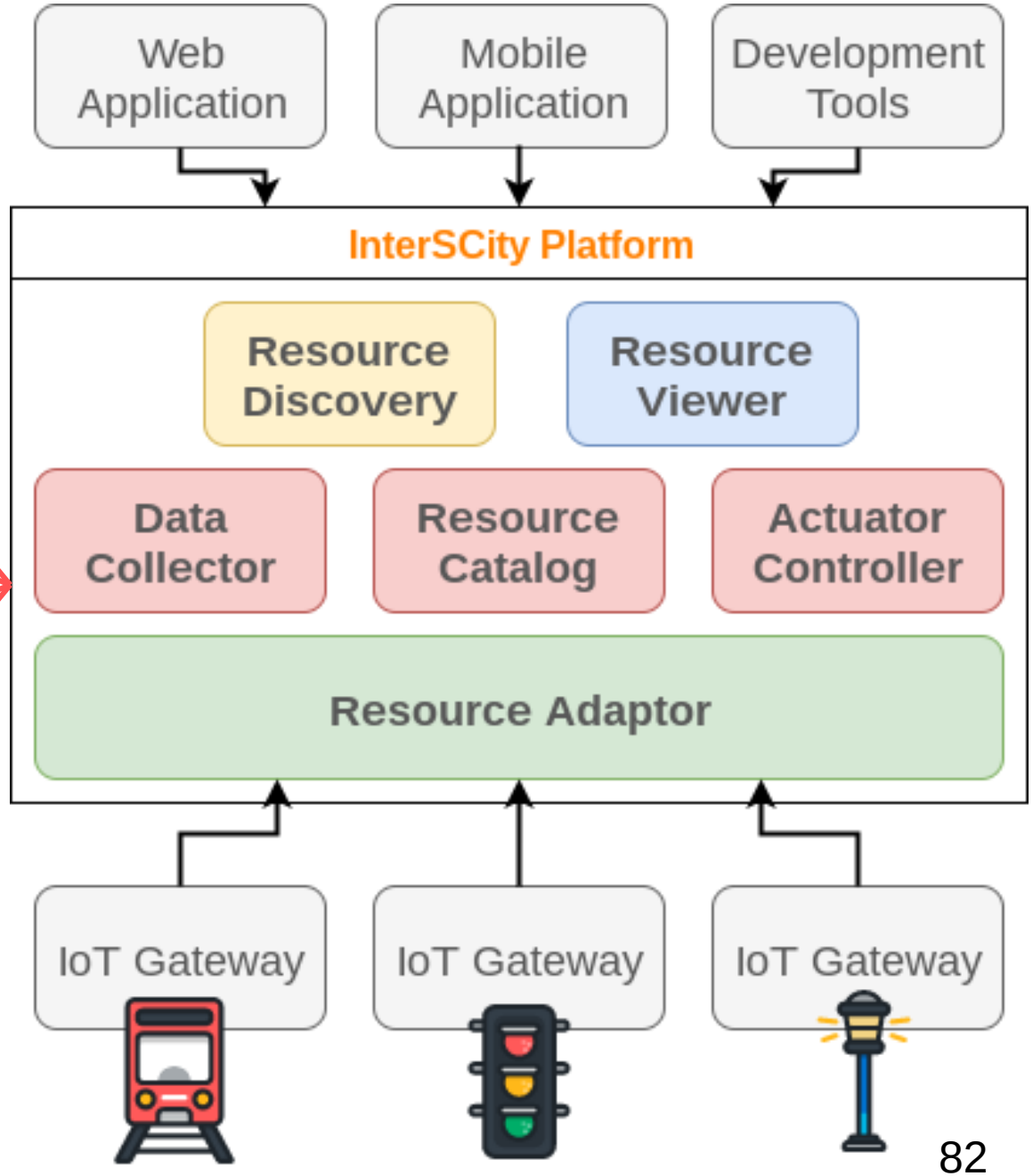
- Platform Architecture
- Development Support
- Big Data Services

Manage heterogeneous city resources

Integrate with IoT Systems

Data storage and processing

Context-aware resource discovery

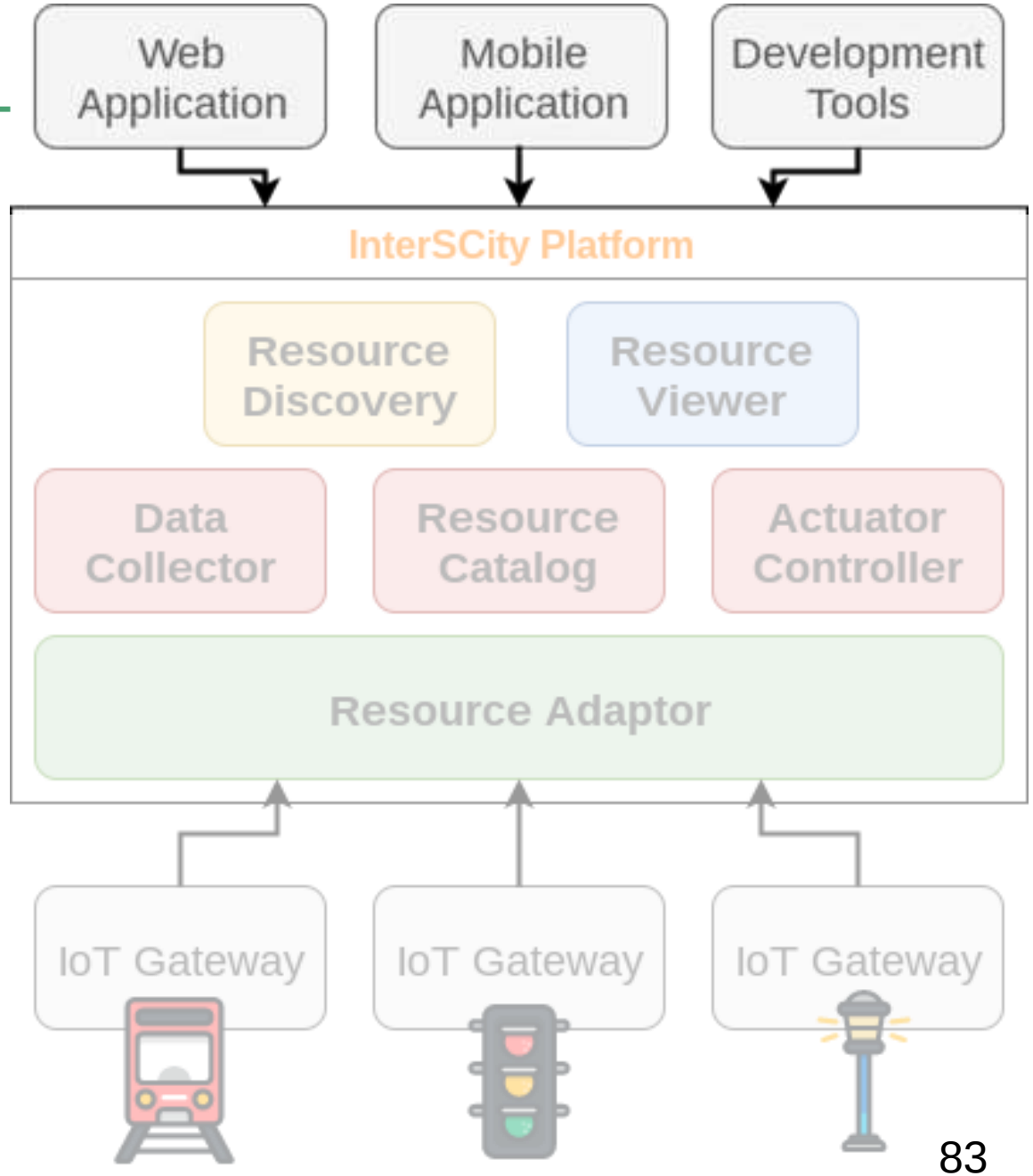


InterSCity Platform

- Platform Architecture
- Development Support
- Big Data Services

Use InterSCity REST APIs

Access InterSCity facilities



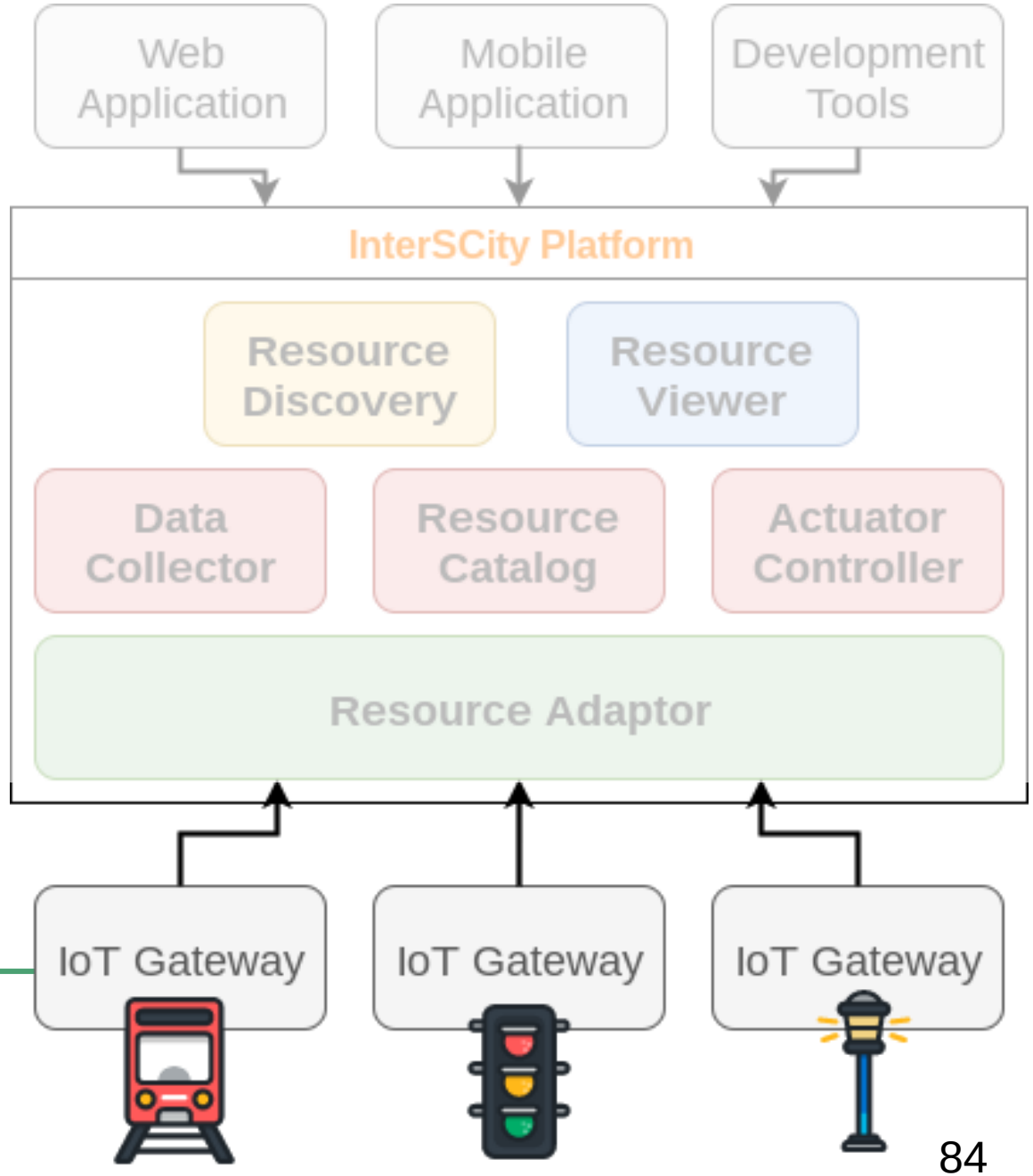
InterSCity Platform

- Platform Architecture
- Development Support
- Big Data Services

Register new resources

Post observed data from resources

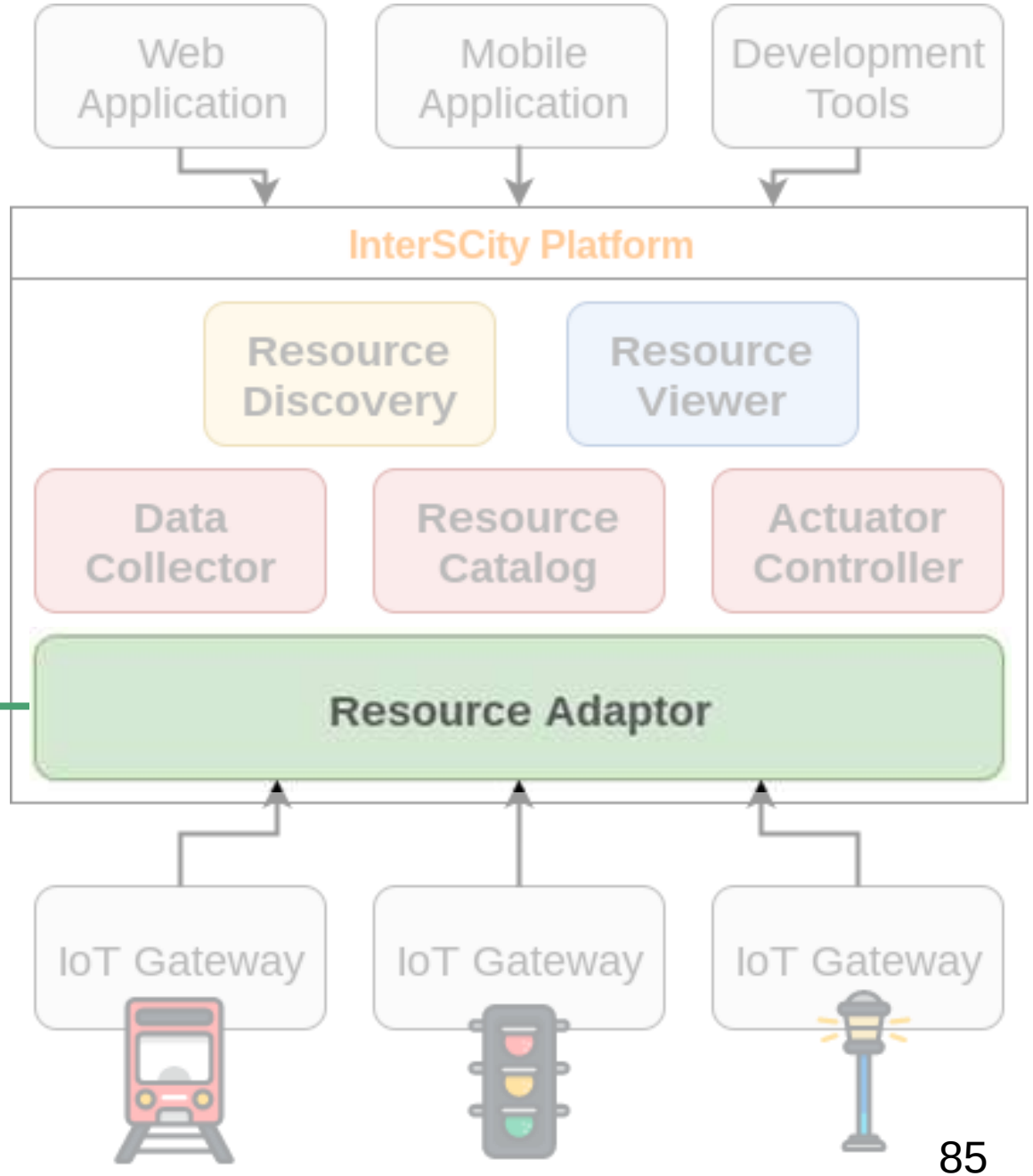
Subscribe to receive actuator commands



InterSCity Platform

- Platform Architecture
- Development Support
- Big Data Services

Proxy Service

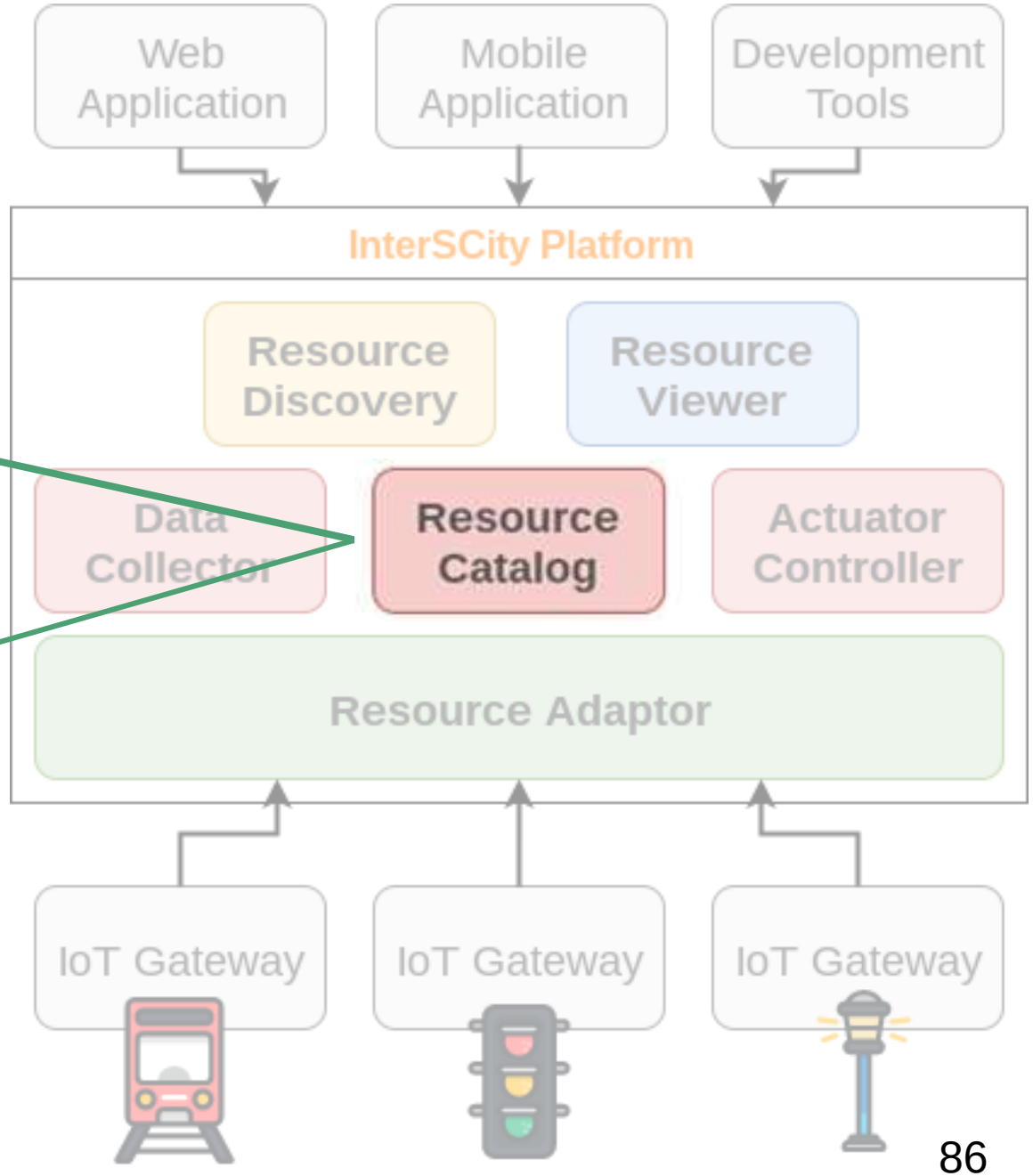


InterSCity Platform

- Platform Architecture
- Development Support
- Big Data Services

Store city resources static data

Notify services on resource creation and update

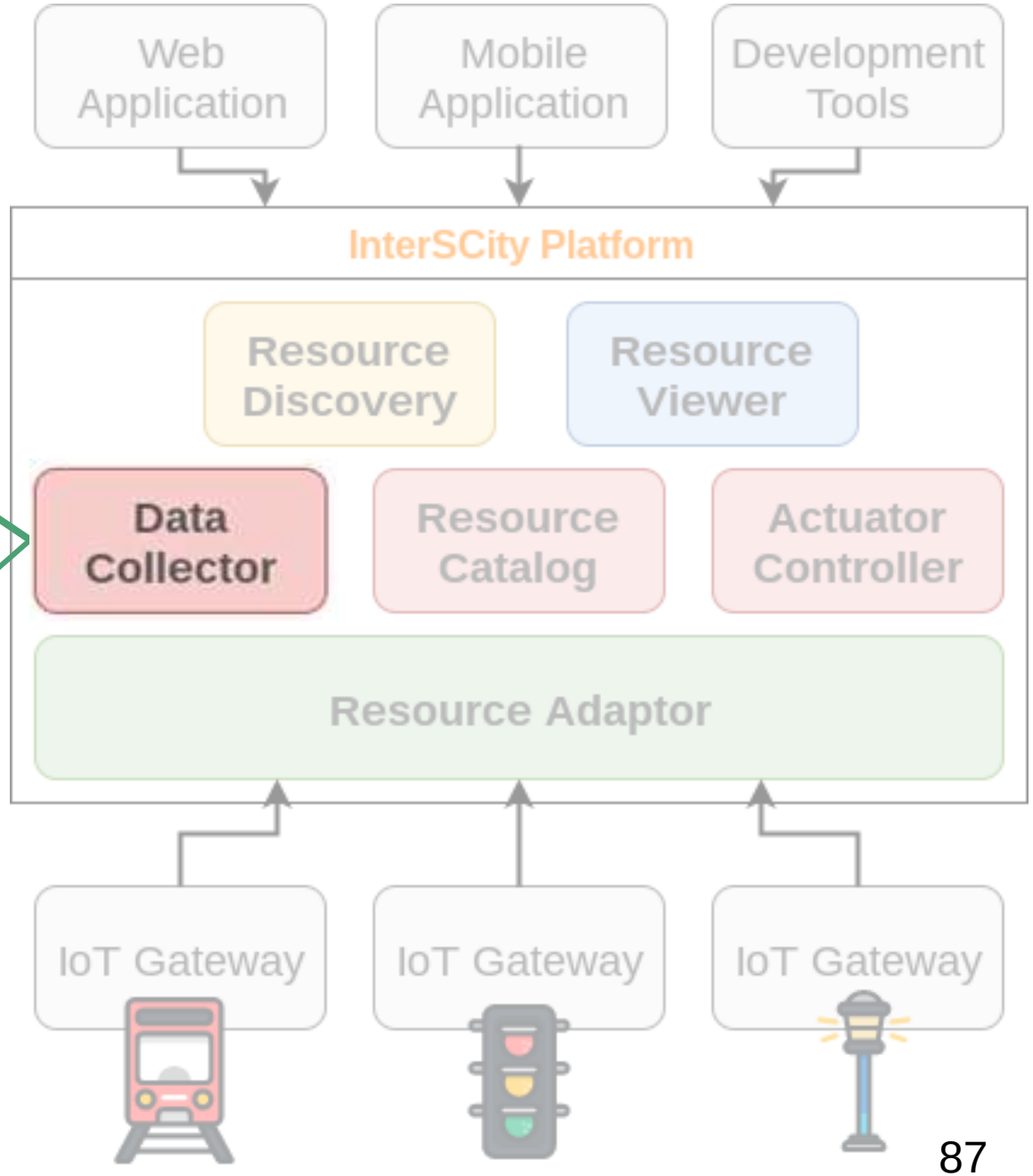


InterSCity Platform

- Platform Architecture
- Development Support
- Big Data Services

Store sensor data

API to access historical and real-time data

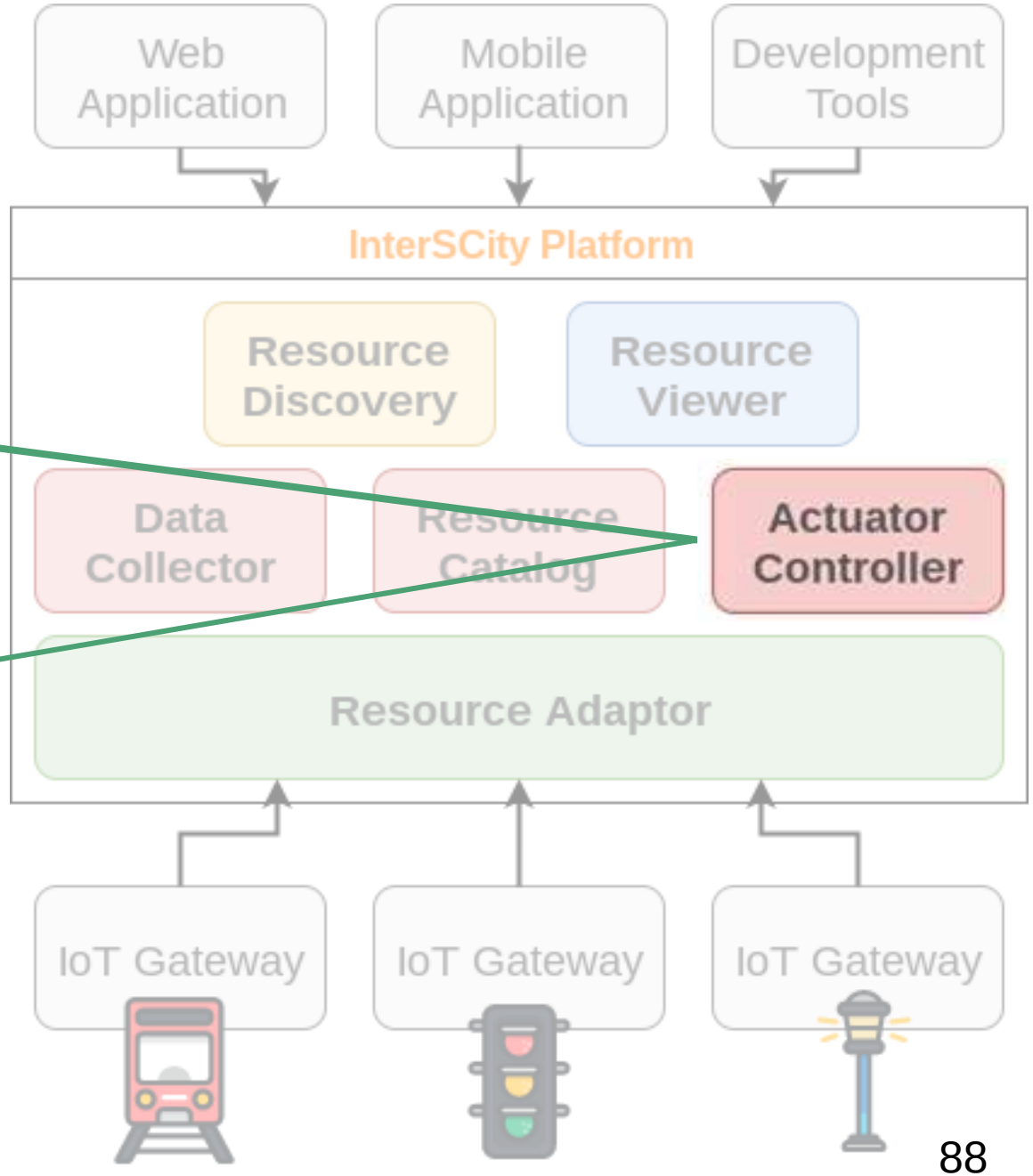


InterSCity Platform

- Platform Architecture
- Development Support
- Big Data Services

Receive actuation commands from applications

Notify on actuation requests

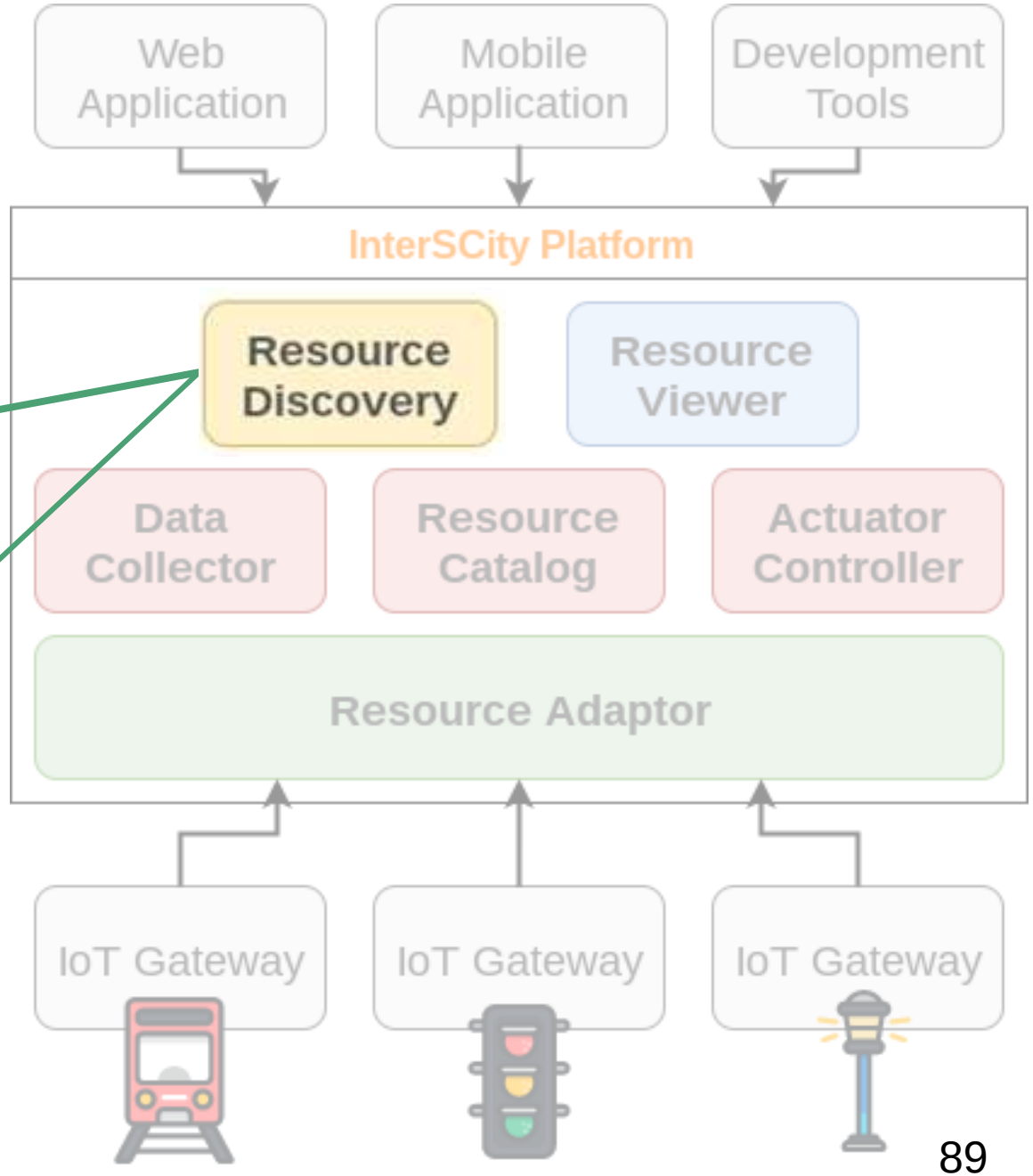


InterSCity Platform

- Platform Architecture
- Development Support
- Big Data Services

Service to discovery city resources

Search filters based on location, range sensor values, static data

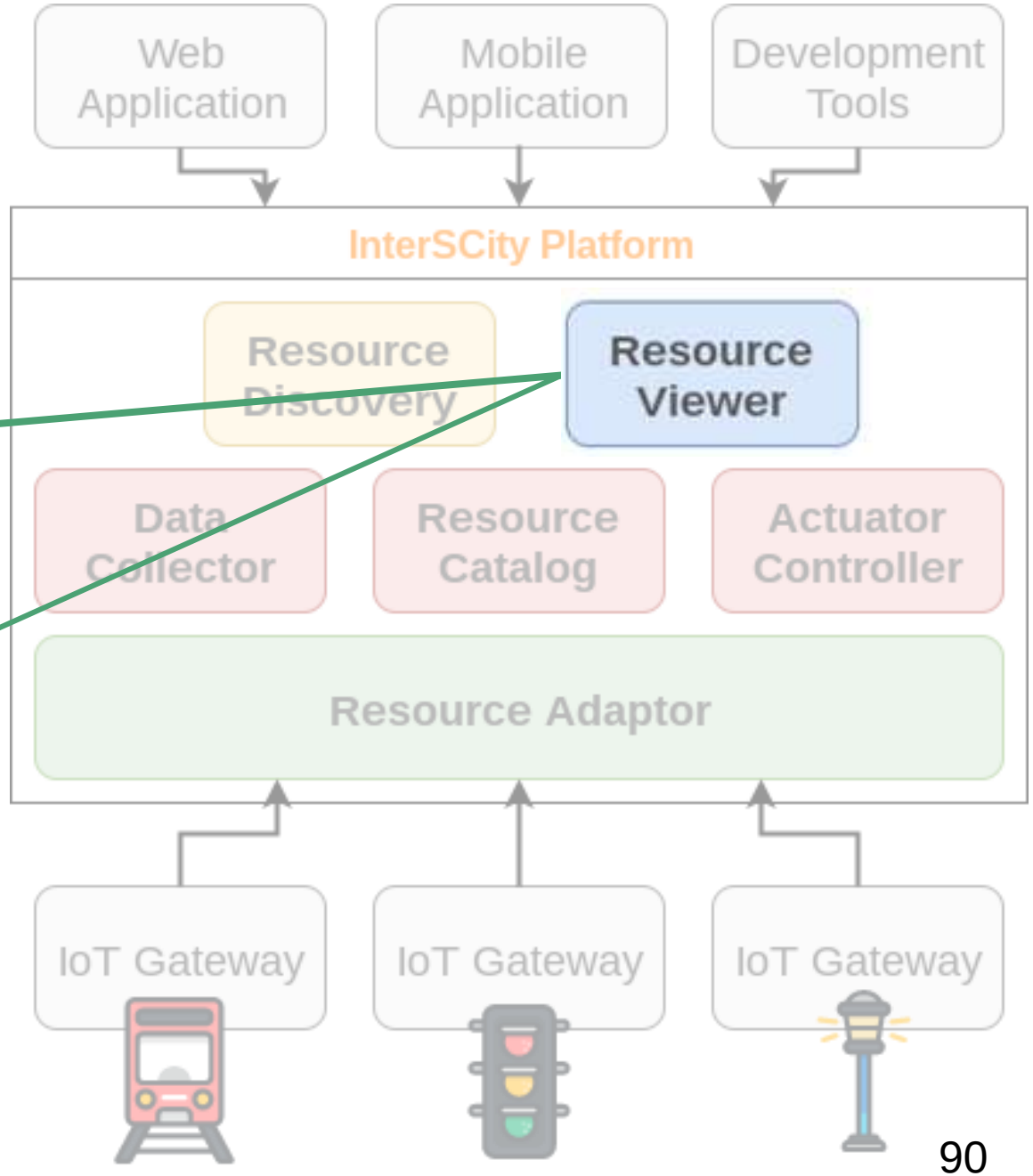


InterSCity Platform

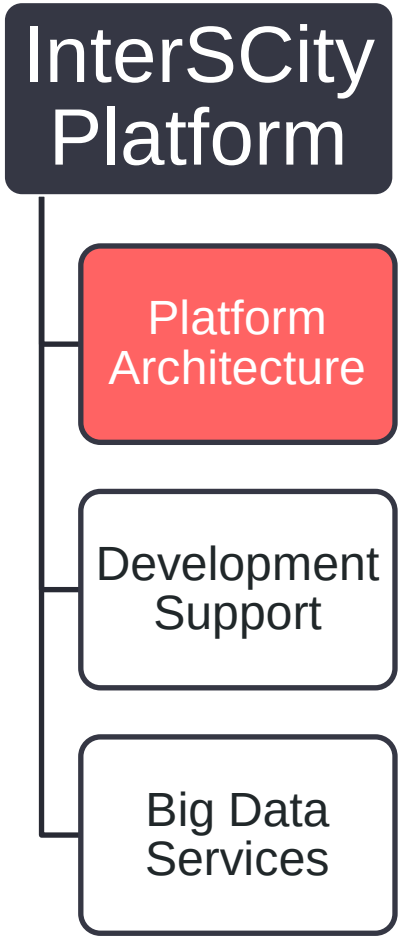
- Platform Architecture
- Development Support
- Big Data Services

Front-end service

City resources exploratory visualization

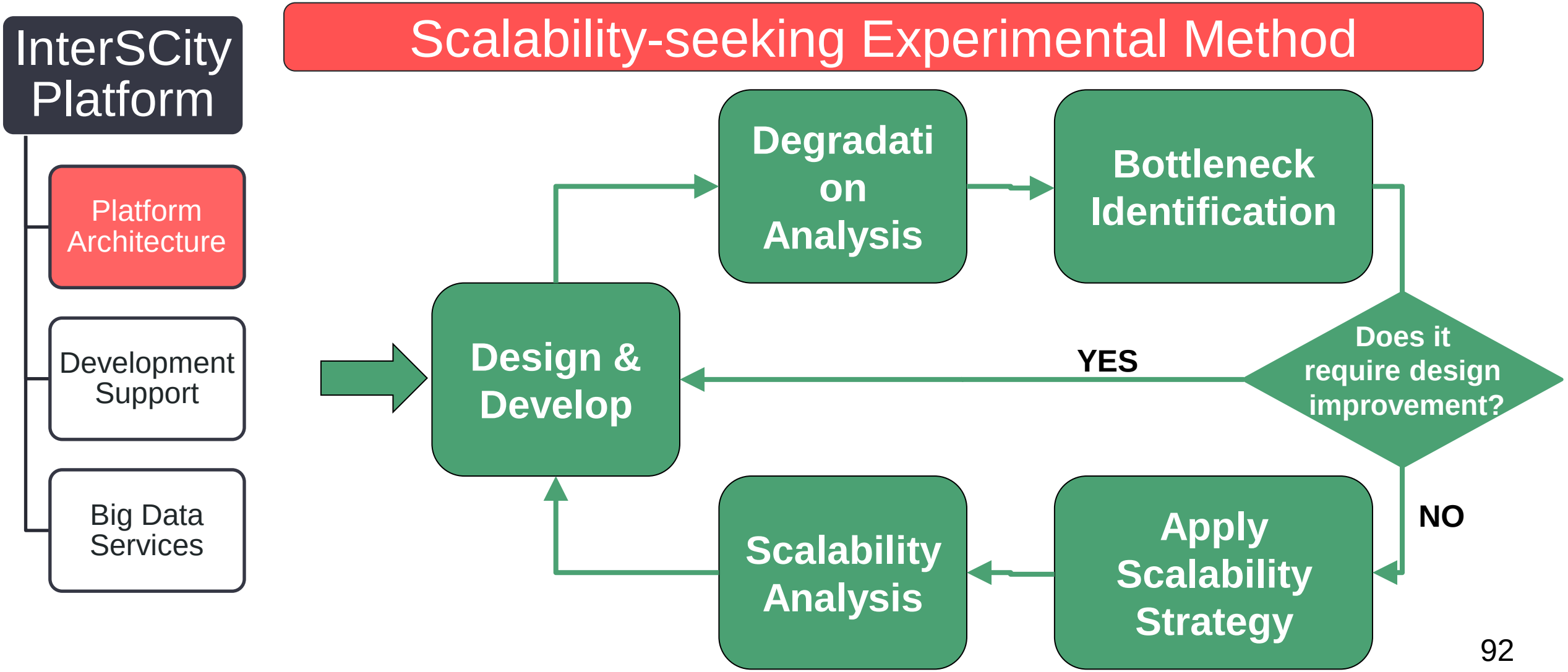


Design Principles



- Modularity via Microservices
- Distributed Models and Data
- Decentralized Evolution
  - Heterogeneity of technologies
- Reuse of Open Source Projects
- Asynchronous vs Synchronous
  - AMQP and HTTP
- Stateless Services

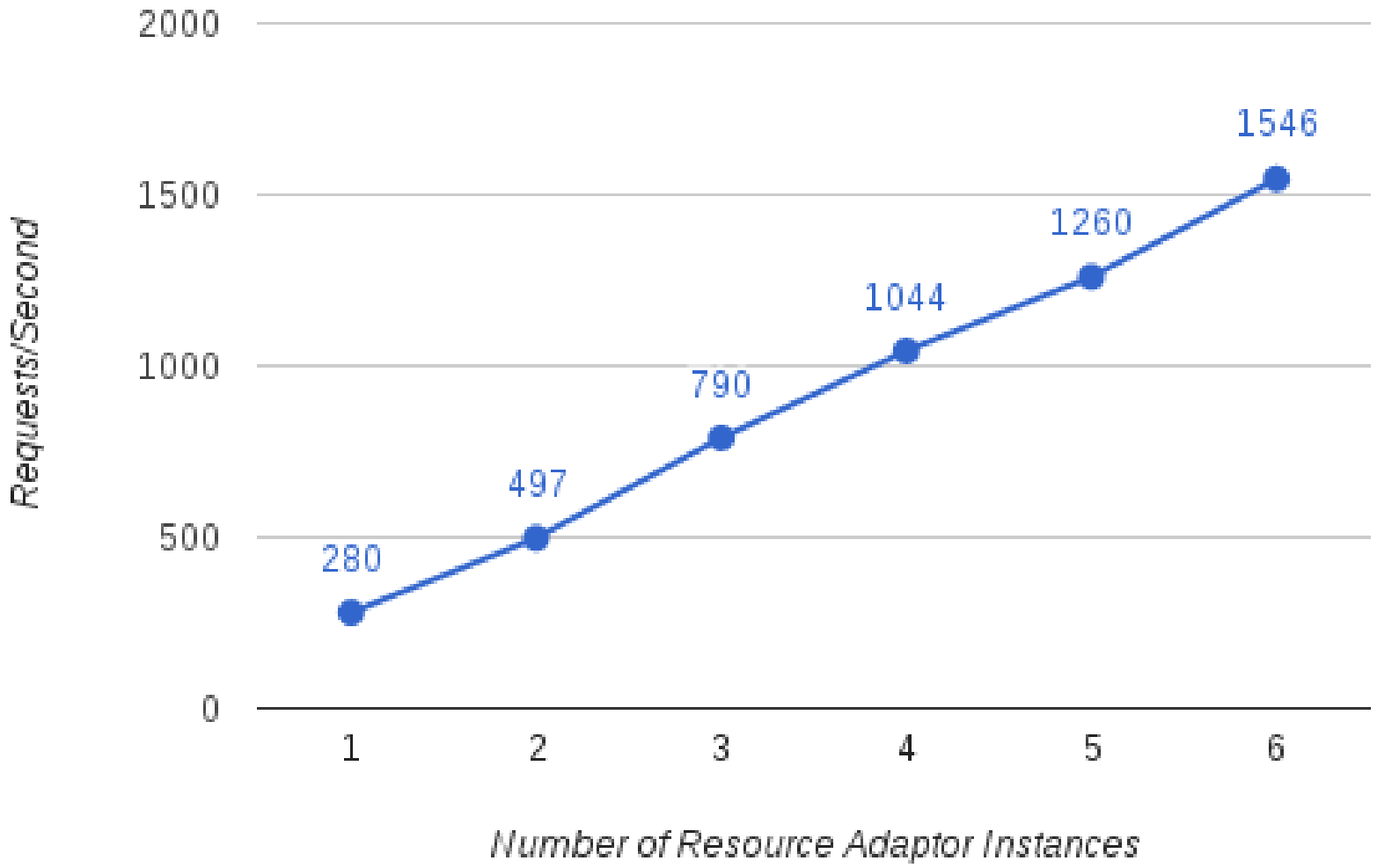
How to achieve scalability?



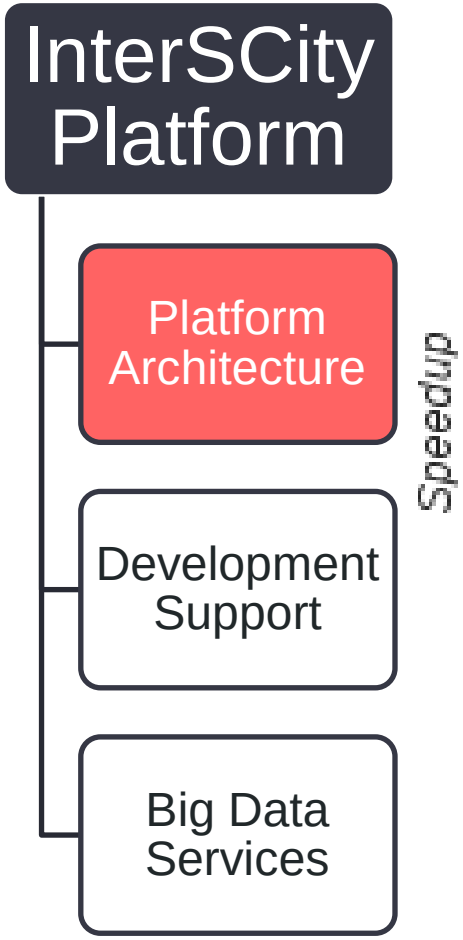
Scalability Analysis - Scale-up

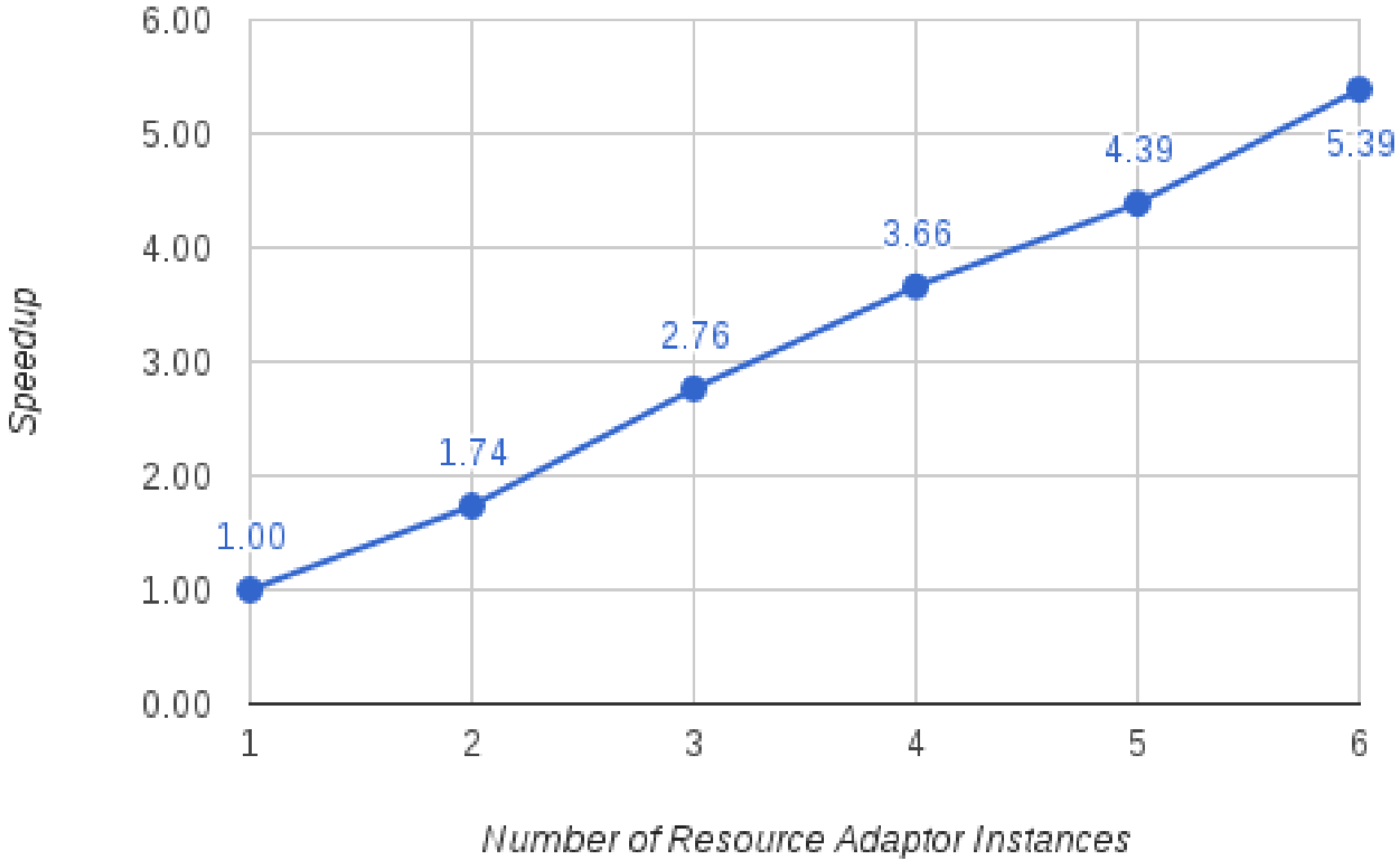
InterSCity Platform

- Platform Architecture
- Development Support
- Big Data Services

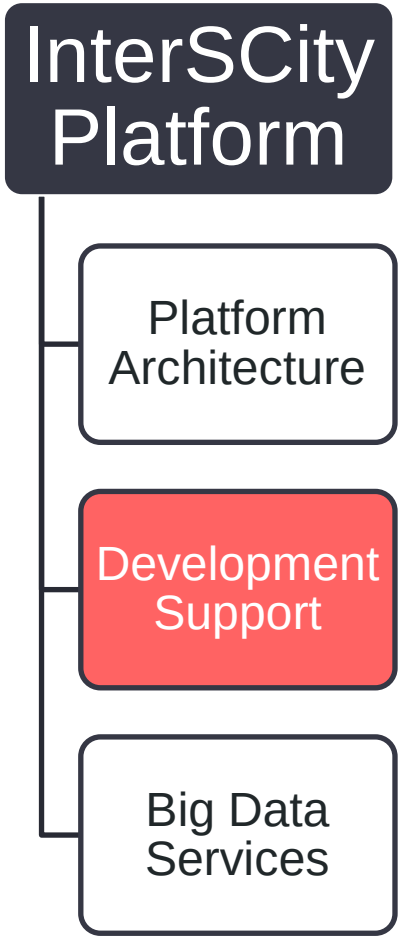


Scalability Analysis - Speedup





Development Support

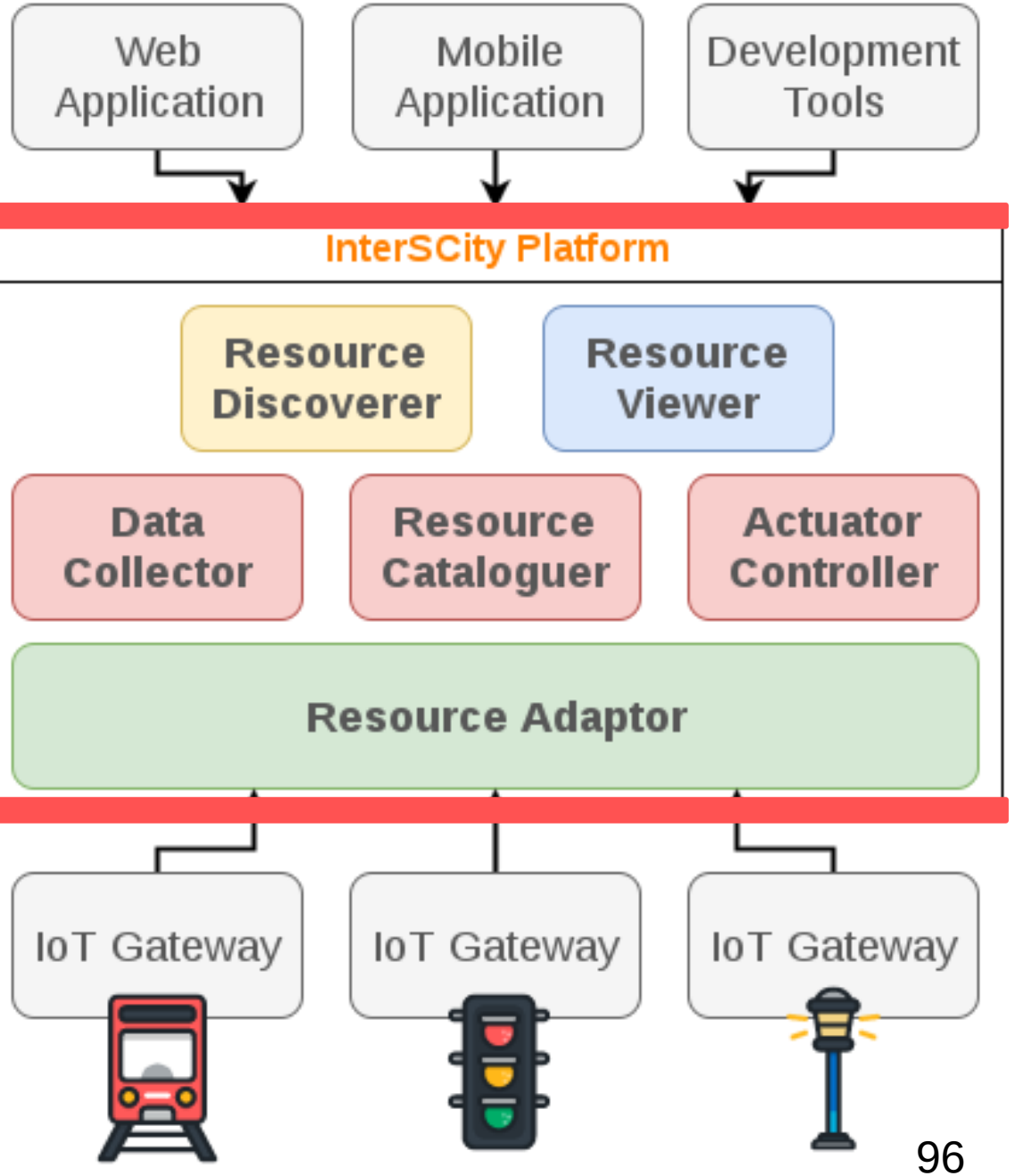


Development Support

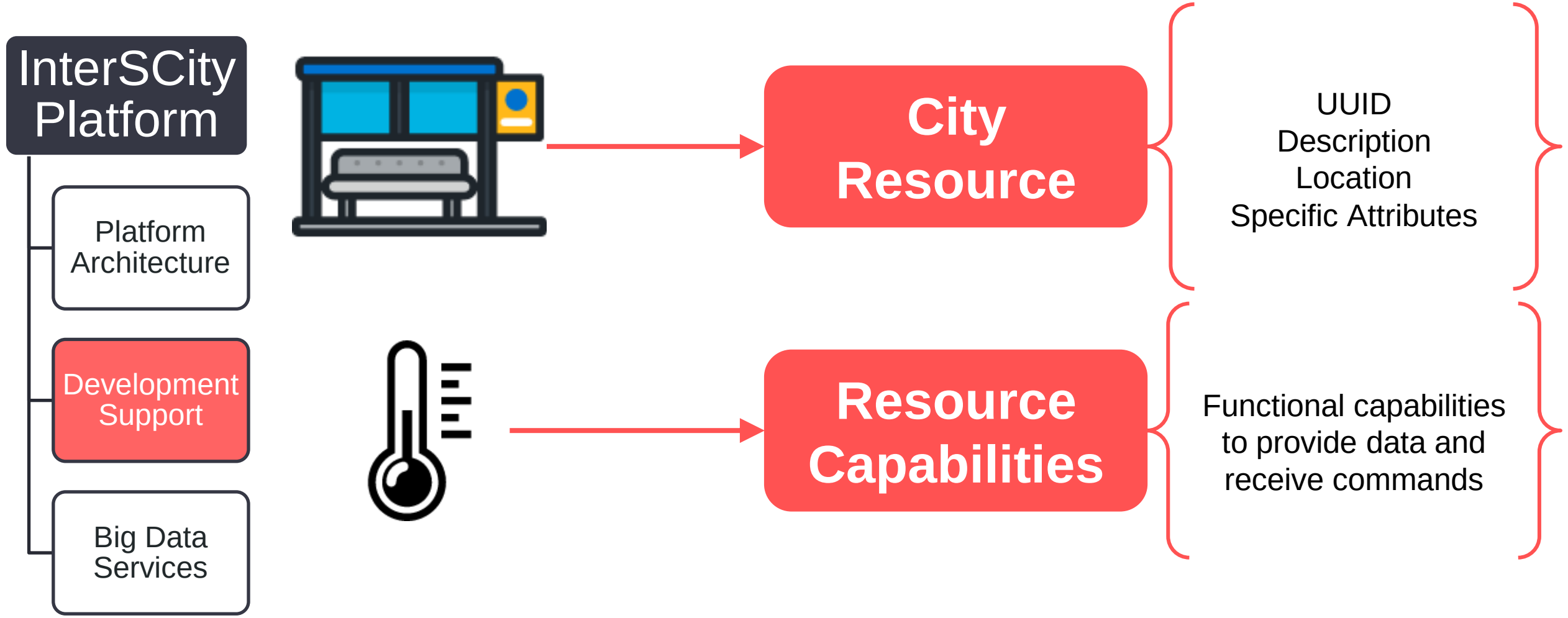
InterSCity Platform

- Platform Architecture
- Development Support
- Big Data Services

All external interfaces of the platform are **REST** APIs based on the **HTTP** protocol and **JSON** messages

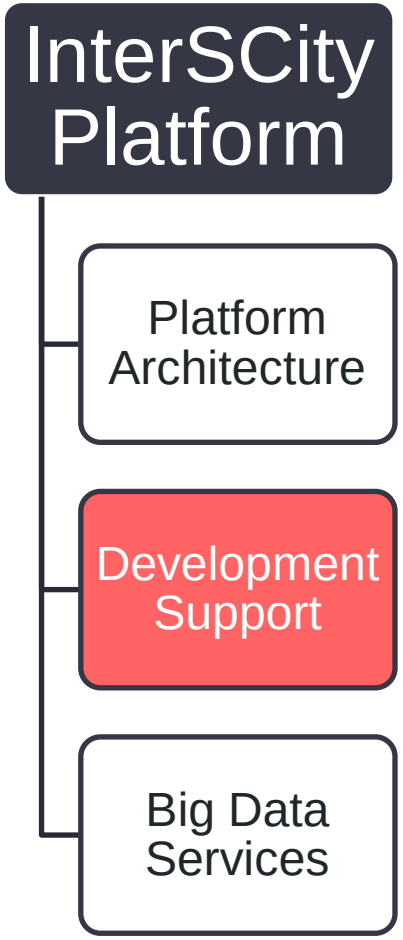


Main Abstractions



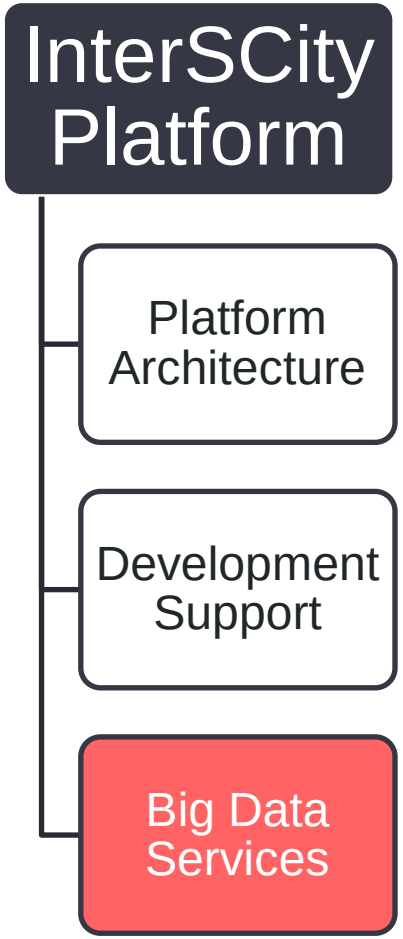
API Documentation: https://gitlab.com/smart-city-software-platform/docs/blob/master/api/API.md

What else could be done?

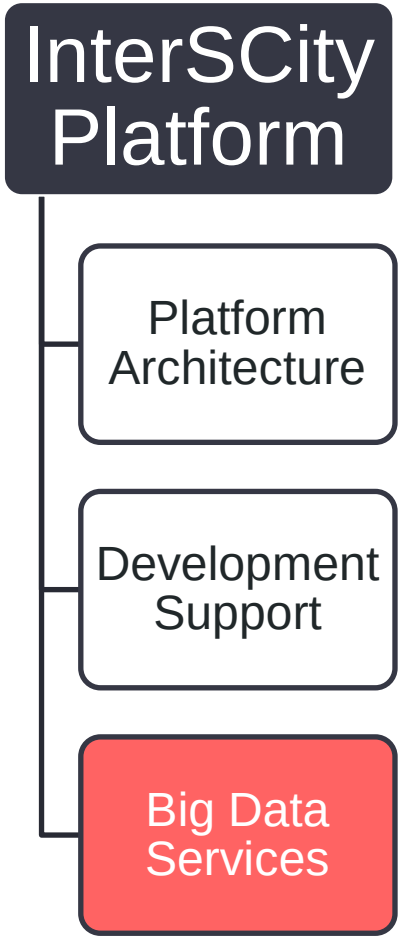


- Interactive API documentation
- Native libraries to abstract API calls
- DSL/Framework to facilitate the integration of low-level IoT Platforms

Big Data Services

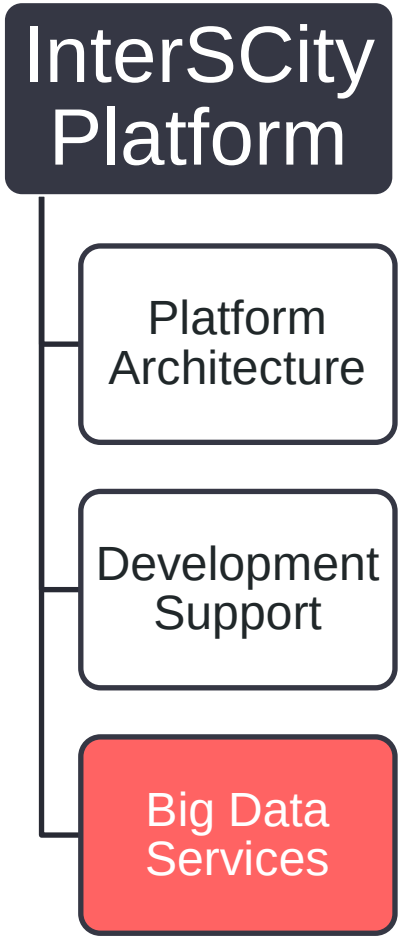


Big Data Research Topics



- Big Data generic services to improve InterSCity's development support
- Stream processing
- Big Data Visualization and Analytics
- Complex Event Processing - CEP
- Historical data processing
- Machine Learning
- Applications

Big Data Research Topics



- Big Data generic services to improve InterSCity's development support
- Stream processing
- Big Data Visualization and Analytics
- Complex Event Processing - CEP
- Historical data processing
- Machine Learning
- Applications

Ongoing Effort

Mapped or Starting

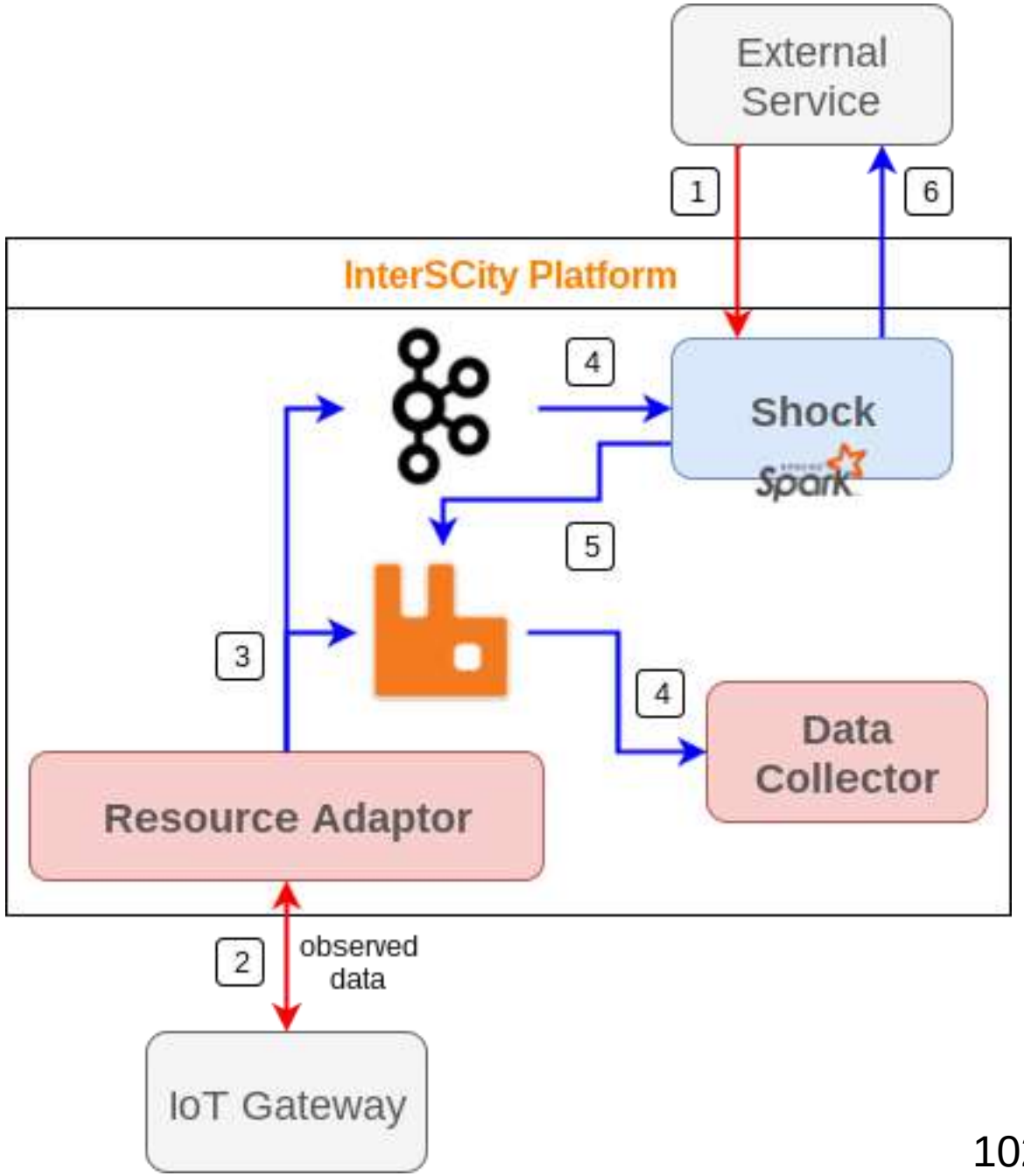
Not initialized

Big Data Service

InterSCity Platform

- Platform Architecture
- Development Support
- Big Data Services

- Creation of Composite Resources
- Aggregation of Resource Observations
- Resource Observations Filtering
- Automatic actuation



Perguntas?

Instalando o InterSCity

Instalando o InterSCity

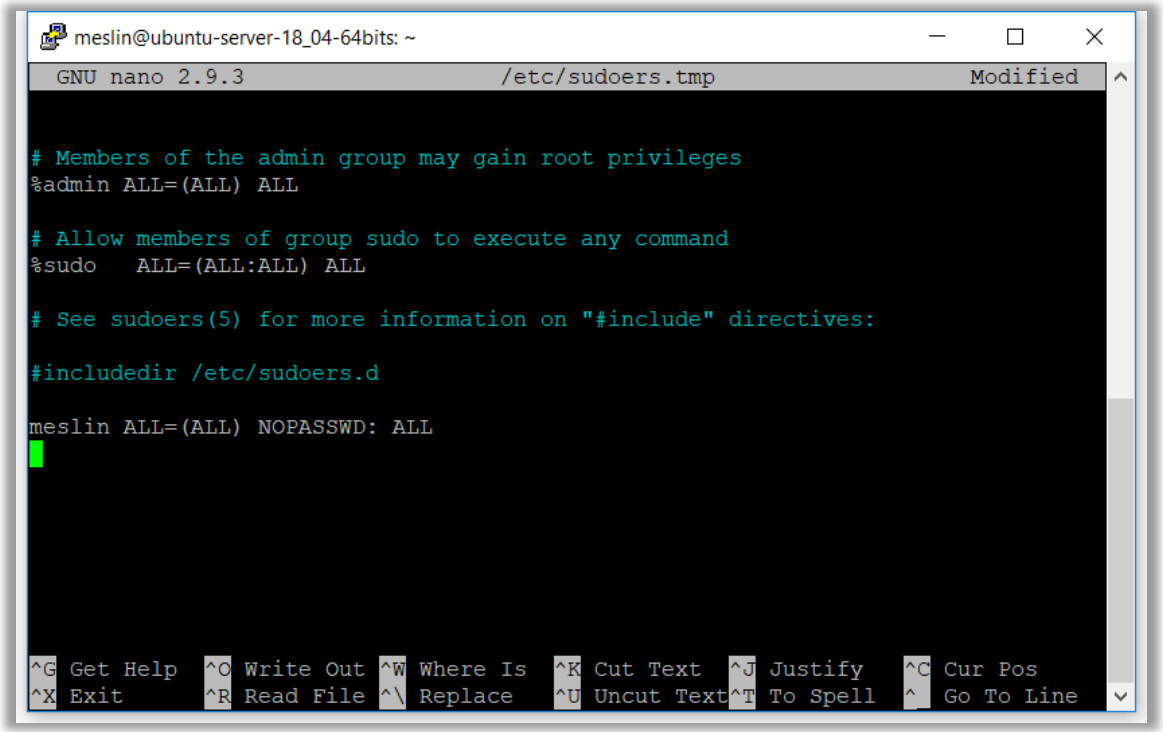
- Antes de começar:
  - Para facilitar a vida, habilitar o uso de sudo sem precisar informar a senha
  - Edite o arquivo de configuração do sudo com o comando:

```
$ sudo visudo
```

  - Acrescente a seguinte linha no final do arquivo:

```
usuário ALL=(ALL) NOPASSWD: ALL
```

  - Substituindo `usuário` pelo seu username



Instalando o InterSCity

- Antes de começar:
  - Atualize o sistema

```
$ sudo apt update
$ sudo apt -y upgrade
$ sudo apt -y autoremove
```

- Para o seu conforto
  - Instale um servidor SSH
  - Para instalar o OpenSSH Server, use o seguinte comando:

```
$ sudo apt -y install openssh-server
```

Instalando o InterSCity

Se for necessário IP fixo

UBUNTU 18.04

- Editar o arquivo /etc/netplan/50-cloud-init.yaml

```
network:
  ethernets:
    ens3:
      addresses: [172.16.3.202/24]
      gateway4: 172.16.3.1
      dhcp4: no
      nameservers:
        addresses: [139.82.16.3, 10.8.0.1]
  version: 2
```

  - E depois usar o comando:

```
$ sudo netplan apply
```

UBUNTU 16.04

- Editar o arquivo /etc/network/interfaces

```
auto ens160
iface ens160 inet static
        address 172.16.1.202
        netmask 255.255.255.0
        network 172.16.1.0
        broadcast 172.16.1.255
        gateway 172.16.1.1
        dns-nameservers 139.82.16.3
        dns-search inf.puc-rio.br
```

Instalando o InterSCity

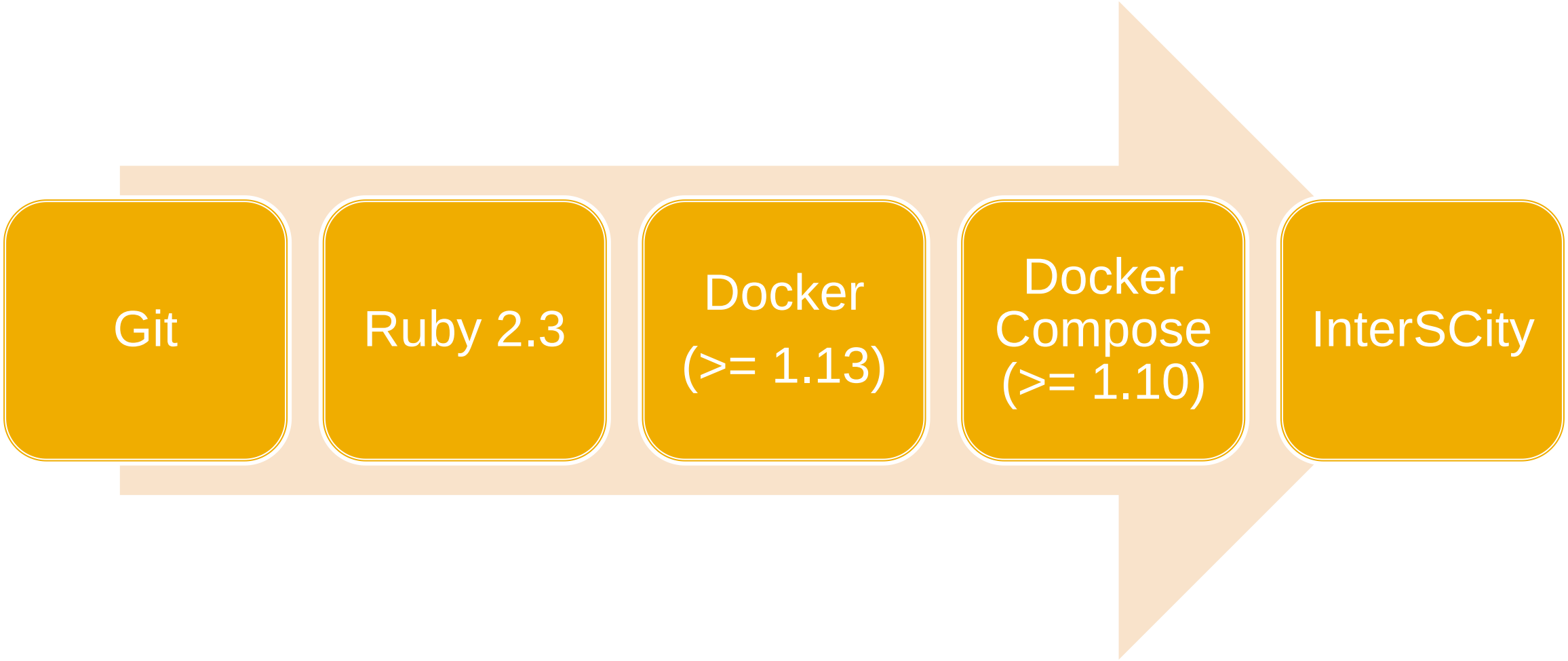
- Na falta de DNS, você pode usar o arquivo /etc/hosts para resolver nomes:

```
# Endereços do MUSANet
172.16.1.202      ContextNet1
172.16.2.202      ContextNet2
172.16.3.202      ContextNet3
172.16.4.202      ContextNet4

172.16.0.211      cliente1
172.16.0.212      cliente2
172.16.0.213      cliente3
172.16.0.214      cliente4
```

Instalando o InterSCity

- Instalação descrita em: `https://gitlab.com/interscity/interscity-platform/interscity-platform/tree/master/src`

Instalando o InterSCity



Instalando o InterSCity

Preliminares

git

- Normalmente pré-instalado na maior parte das distribuições Linux
- $ sudo apt install git

gpg2

- Você vai precisar para instalar o Ruby
- $ sudo apt install gnupg2

Instalando o InterSCity

Ruby 2.3

- Descrito em https://rvm.io/
- Instalar chave GPG:
- `$ gpg2 --recv-keys 409B6B1796C275462A1703113804BB82D39DC0E3 7D2BAF1CF37B13E2069D6956105BD0E739499BDB`

- RVM:
- `$ curl -sSL https://get.rvm.io | bash -s stable --rails`

Instalando o InterSCity
Docker

- Descrito em https://docs.docker.com/install/linux/docker-ce/ubuntu/

```
$ sudo apt -y install apt-transport-https ca-certificates curl gnupg-agent
software-properties-common
$ curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add -
$ sudo apt-key fingerprint 0EBFCD88
$ sudo add-apt-repository "deb [arch=amd64]
https://download.docker.com/linux/ubuntu $(lsb_release -cs) stable"
$ sudo apt update
$ sudo apt -y install docker-ce docker-ce-cli containerd.io
```

Instalando o InterSCity
Docker

- Para testar a instalação do Docker:

```
$ sudo docker run hello-world
Unable to find image 'hello-world:latest' locally
latest: Pulling from library/hello-world
ca4f61b1923c: Pull complete
Digest:
sha256:083de497cff944f969d8499ab94f07134c50bcf5e6b9559b27182d3fa80ce3f7
Status: Downloaded newer image for hello-world:latest

Hello from Docker!
This message shows that your installation appears to be working correctly.

To generate this message, Docker took the following steps:
 1. The Docker client contacted the Docker daemon.
 2. The Docker daemon pulled the "hello-world" image from the Docker Hub.
    (amd64)
```

Instalando o InterSCity

Docker Compose

- Descrito em https://docs.docker.com/compose/install/

```
$ sudo curl -L
"https://github.com/docker/compose/releases/download/1.24.0/docker-compose-
$(uname -s)-$(uname -m)" -o /usr/local/bin/docker-compose
$ sudo chmod +x /usr/local/bin/docker-compose
```

Instalando o InterSCity
Docker

- Para testar a instalação do Docker Compose:

```
$ docker-compose --version
docker-compose version 1.16.1, build 1719ceb
```

Instalando o InterSCity (finalmente!!!)

- Descrito em https://gitlab.com/smart-city-software-platform/dev-env

```
$ sudo mkdir /opt/InterSCity
$ cd /opt/InterSCity
$ sudo git clone https://gitlab.com/smart-city-software-platform/dev-env
$ cd dev-env
$ sudo git submodule init
$ sudo git submodule update
$ ./project setup
$ sudo docker network create platform
```

- Iniciar o projeto, dentro do diretório /opt/InterSCity/dev-env:

```
$ ./project start
```

Instalando o InterSCity
Testando...

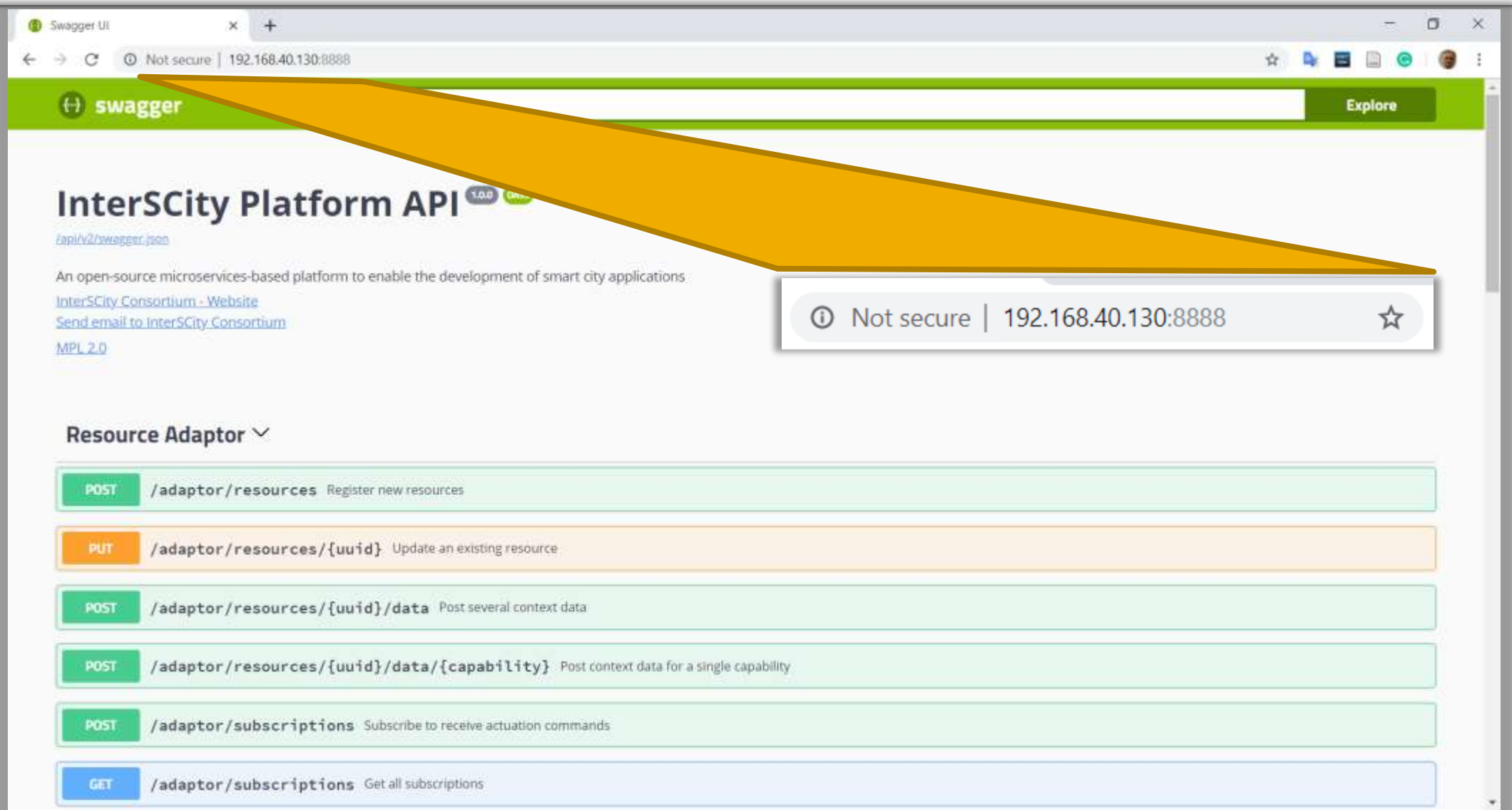
- Para testar o projeto, execute o comando a seguir. A resposta deverá ser uma lista de capabilities previamente configurada no InterSCity:

```
$ curl -X GET "http://localhost:8000/catalog/capabilities"
{"capabilities":[{"id":1,"name":"air-
quality","function":0,"description":null},{"id":2,"name":"air-
humidity","function":0,"description":null},{"id":3,"name":"parking-
occupancy","function":0,"description":null},{"id":4,"name":"video","function":
0,"description":null},{"id":5,"name":"queue-
length","function":0,"description":null},{"id":6,"name":"semaphore","function"
:1,"description":null},{"id":7,"name":"illuminate","function":1,"description":
null},{"id":8,"name":"parking-
type","function":2,"description":null},{"id":9,"name":"current_users","functio
n":0,"description":null},{"id":10,"name":"info_facility_type","function":2,"de
scription":null},{"id":11,"name":"info_total_capacity","function":2,"descripti
```

Perguntas?

Conhecendo o InterSCity

Conhecendo o InterSCity

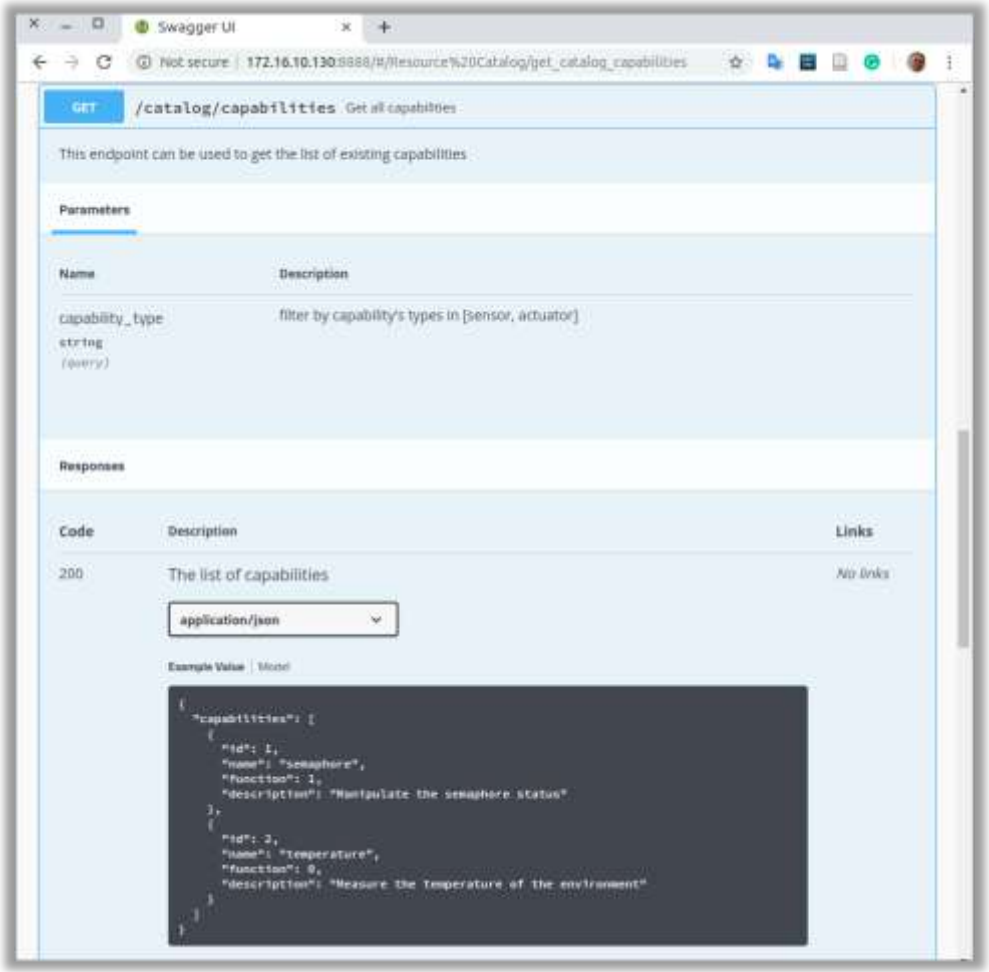


InterSCity
Resource Catalog

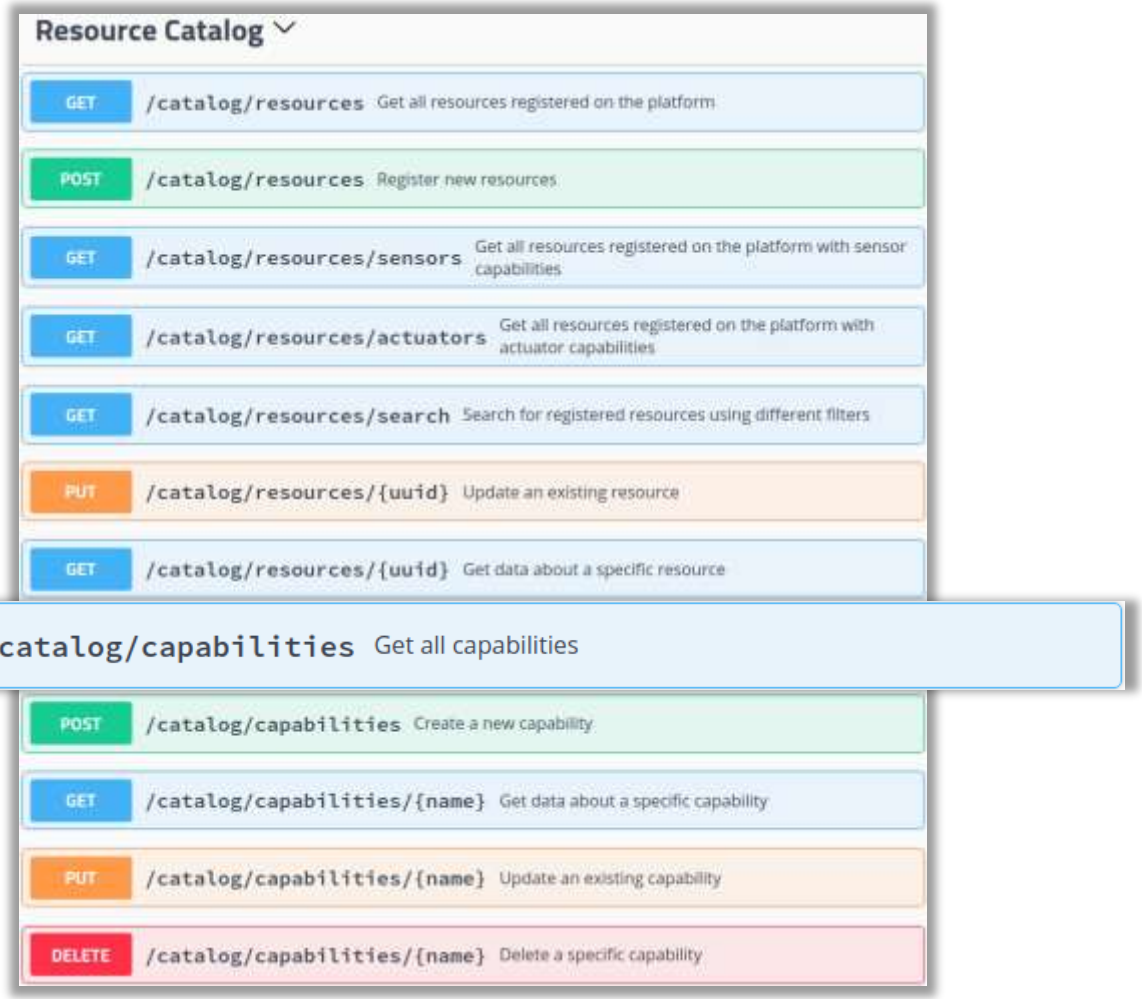
InterSCity
Resource Catalog

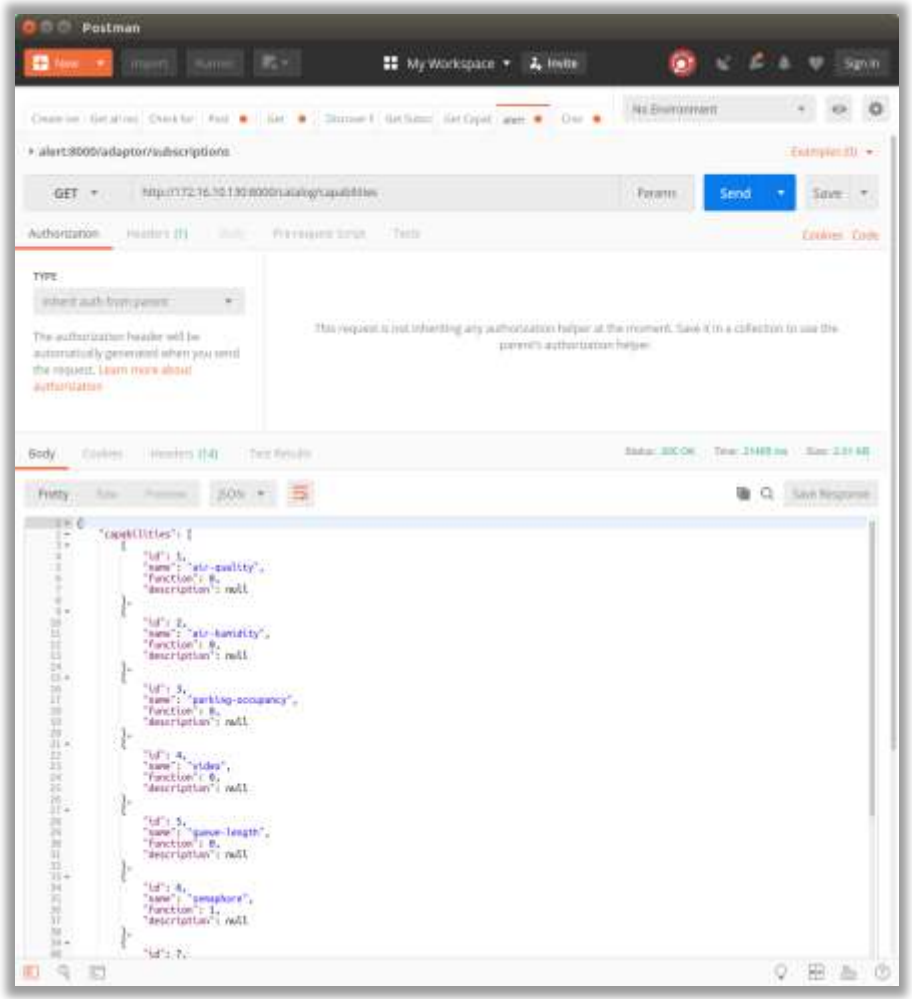
Resource Catalog

GET	/catalog/resources	Get all resources registered on the platform
POST	/catalog/resources	Register new resources
GET	/catalog/resources/sensors	Get all resources registered on the platform with sensor capabilities
GET	/catalog/resources/actuators	Get all resources registered on the platform with actuator capabilities
GET	/catalog/resources/search	Search for registered resources using different filters
PUT	/catalog/resources/{uuid}	Update an existing resource
GET	/catalog/resources/{uuid}	Get data about a specific resource
GET	/catalog/capabilities	Get all capabilities
POST	/catalog/capabilities	Create a new capability
GET	/catalog/capabilities/{name}	Get data about a specific capability
PUT	/catalog/capabilities/{name}	Update an existing capability
DELETE	/catalog/capabilities/{name}	Delete a specific capability

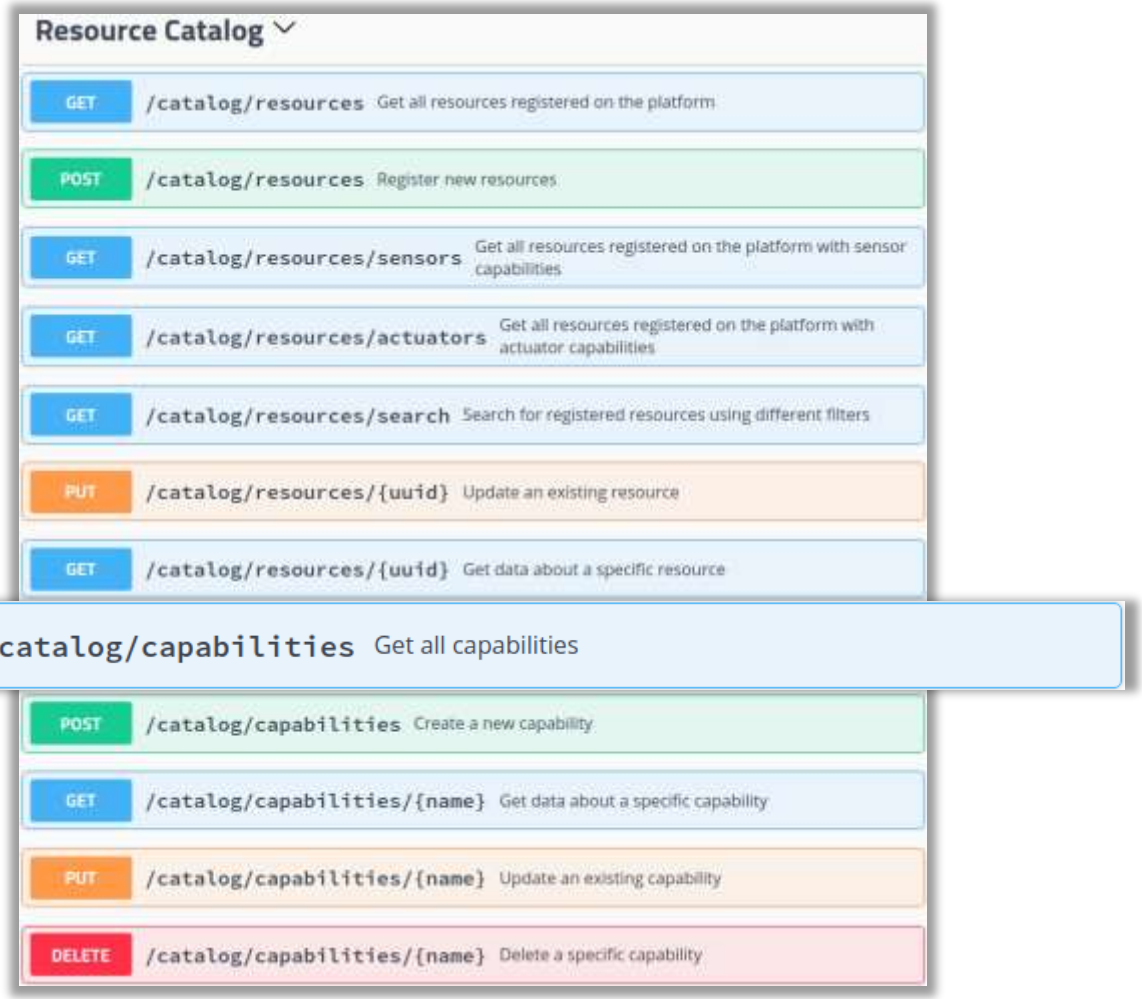


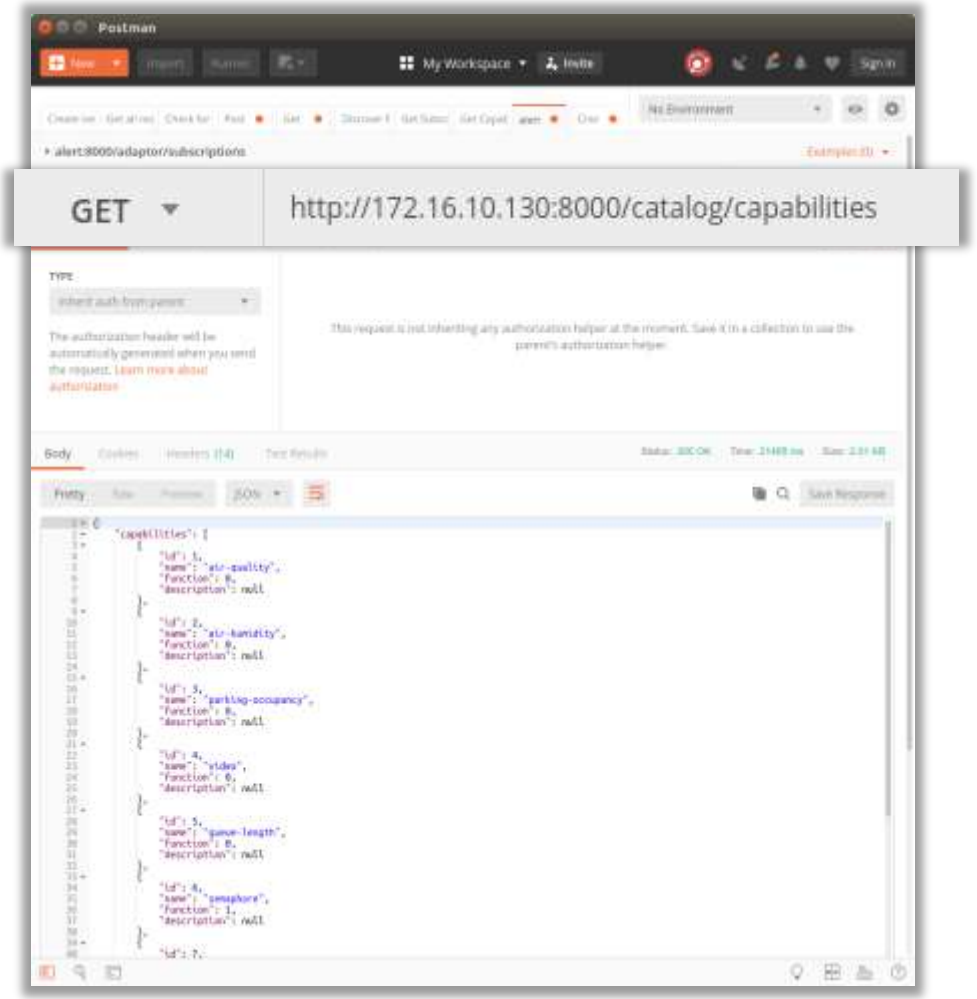
InterSCity
Resource Catalog



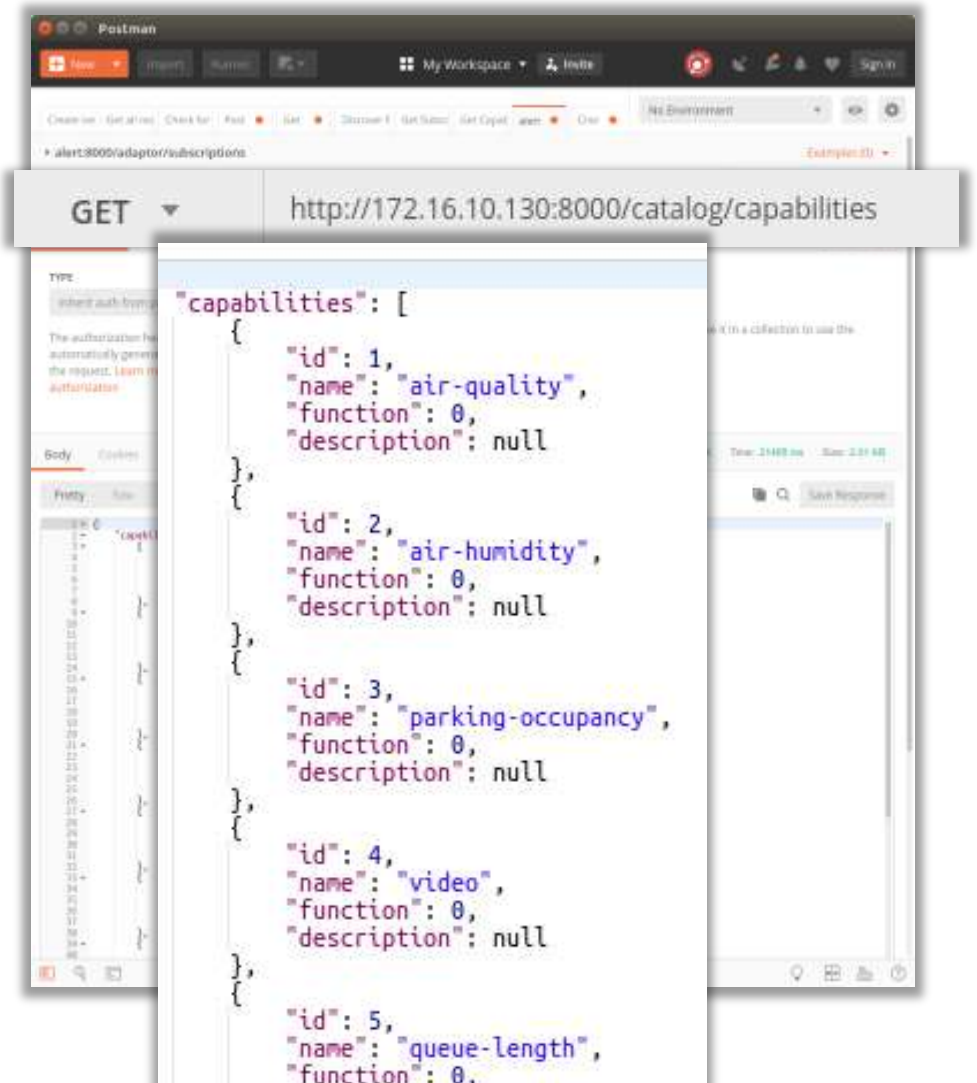


InterSCity
Resource Catalog



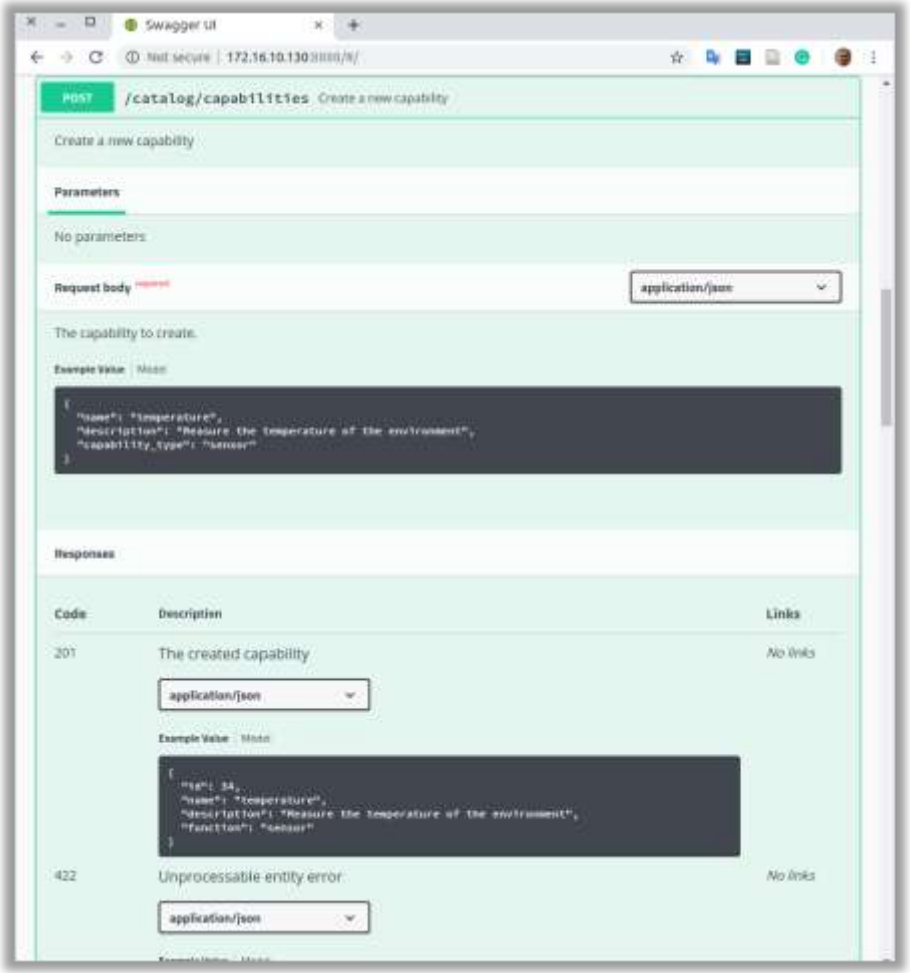


InterSCity
Resource Catalog



InterSCity
Resource Catalog

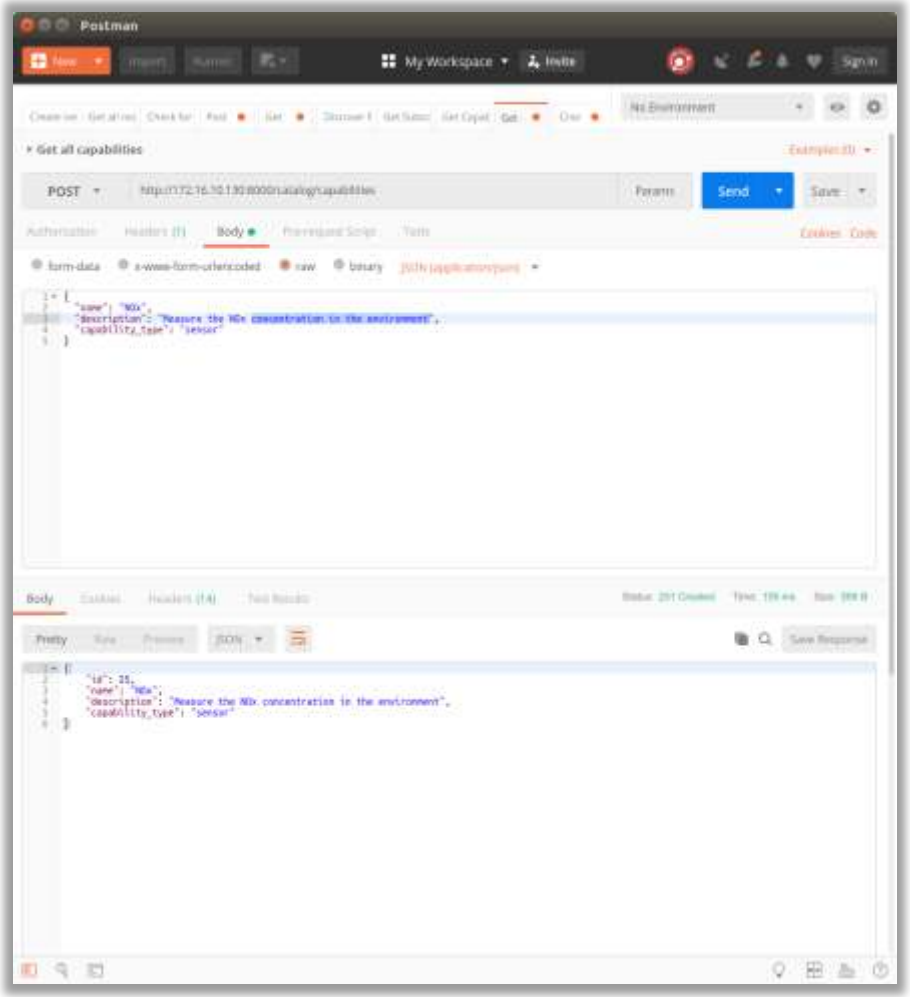
InterSCity
Resource Catalog



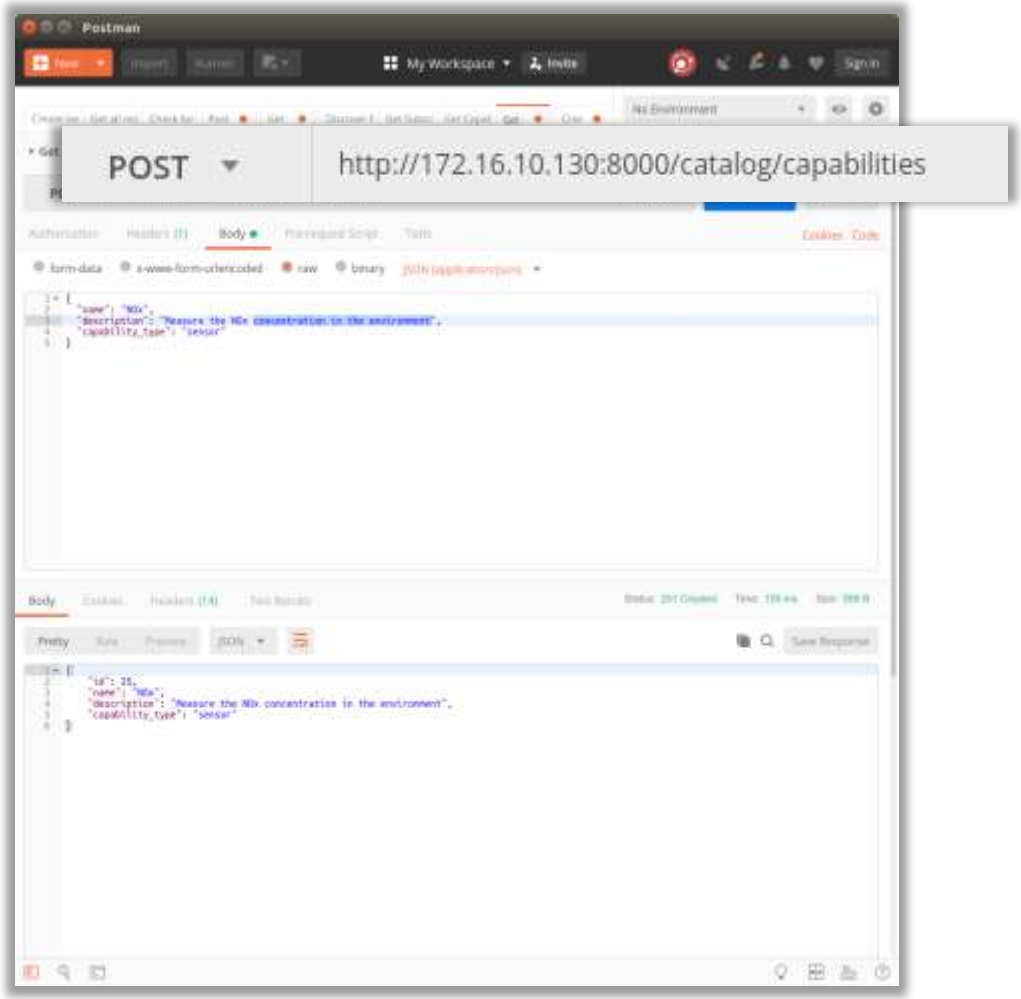
InterSCity
Resource Catalog

Resource Catalog

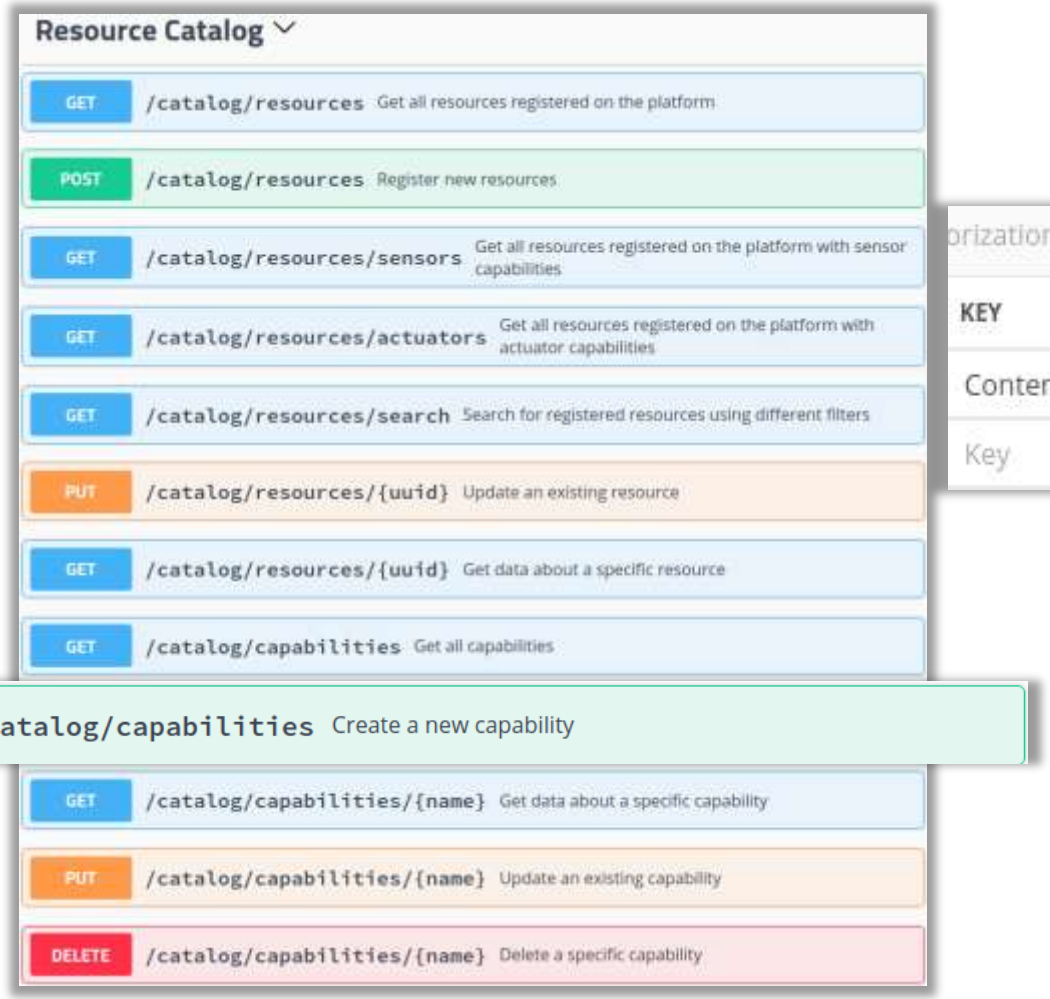
GET	/catalog/resources	Get all resources registered on the platform
POST	/catalog/resources	Register new resources
GET	/catalog/resources/sensors	Get all resources registered on the platform with sensor capabilities
GET	/catalog/resources/actuators	Get all resources registered on the platform with actuator capabilities
GET	/catalog/resources/search	Search for registered resources using different filters
PUT	/catalog/resources/{uuid}	Update an existing resource
GET	/catalog/resources/{uuid}	Get data about a specific resource
GET	/catalog/capabilities	Get all capabilities
POST	/catalog/capabilities	Create a new capability
GET	/catalog/capabilities/{name}	Get data about a specific capability
PUT	/catalog/capabilities/{name}	Update an existing capability
DELETE	/catalog/capabilities/{name}	Delete a specific capability

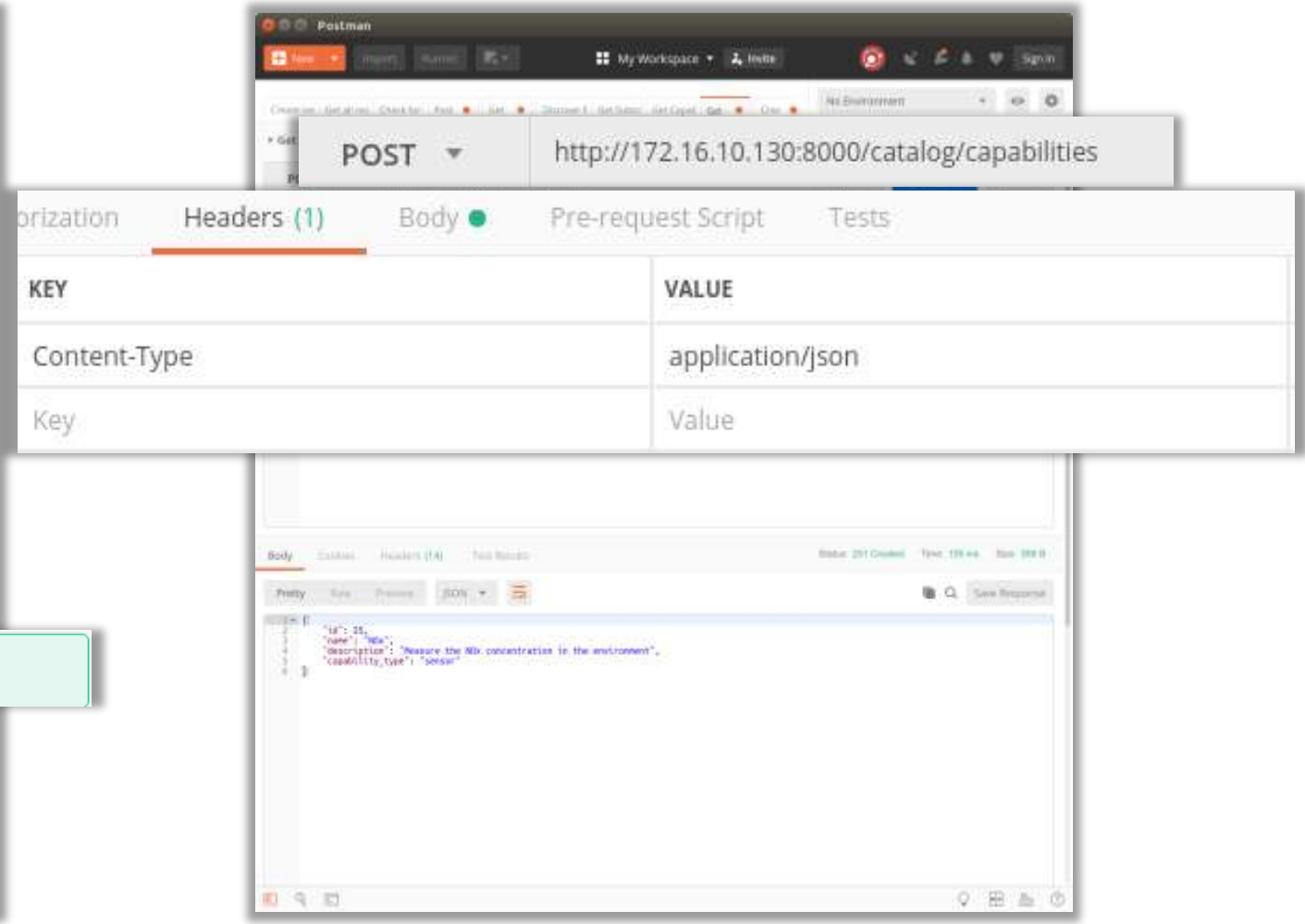


InterSCity
Resource Catalog

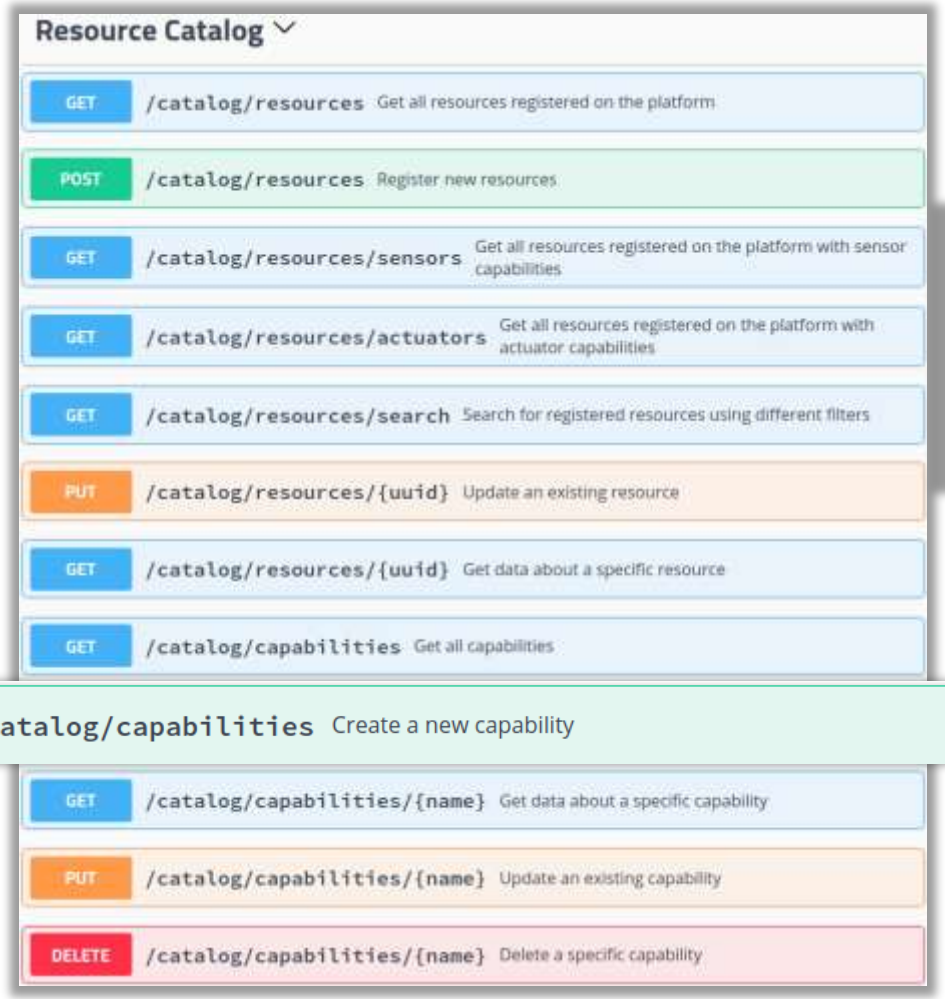


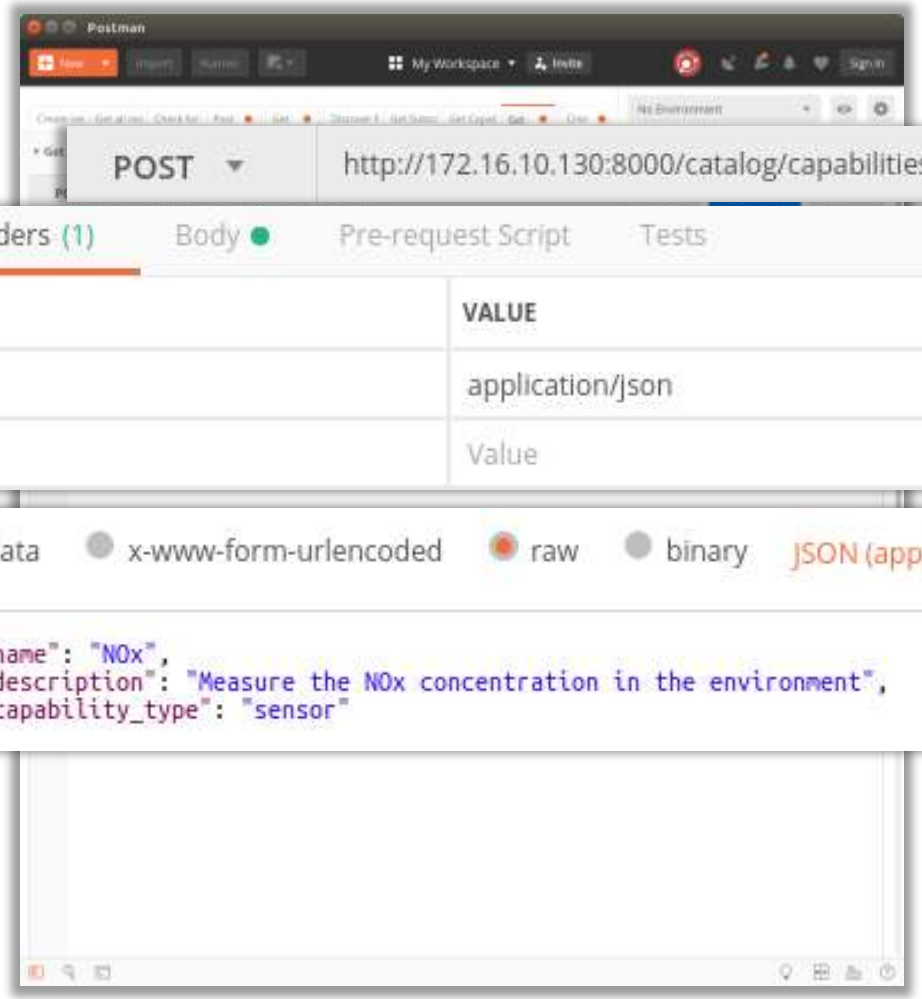
InterSCity
Resource Catalog



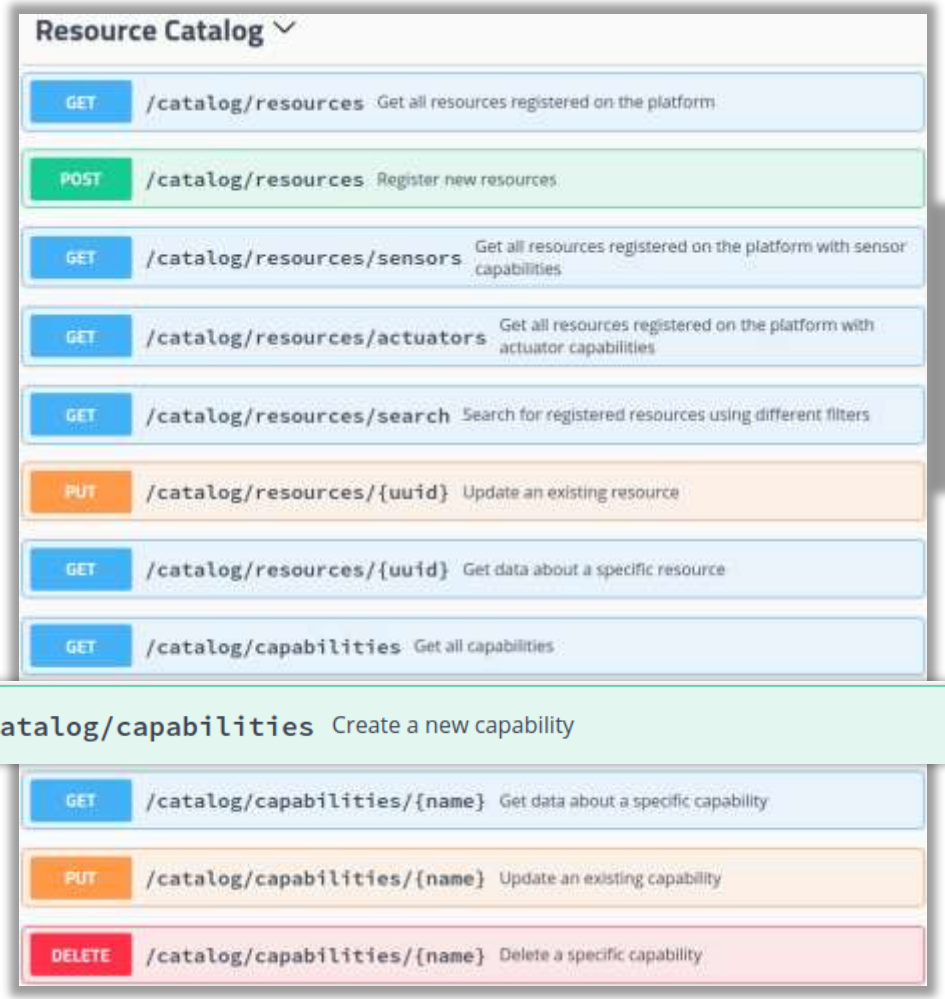


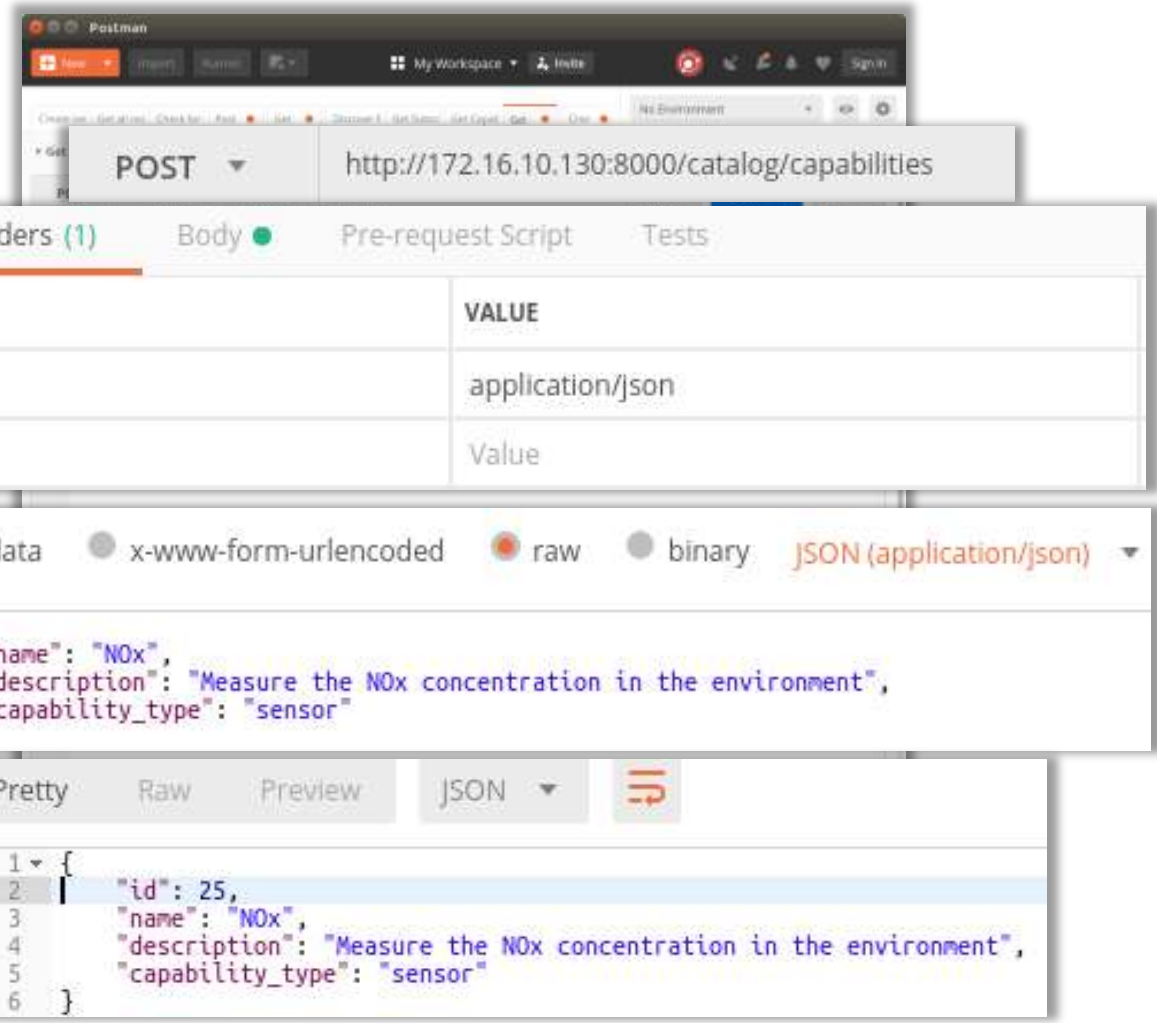
InterSCity
Resource Catalog





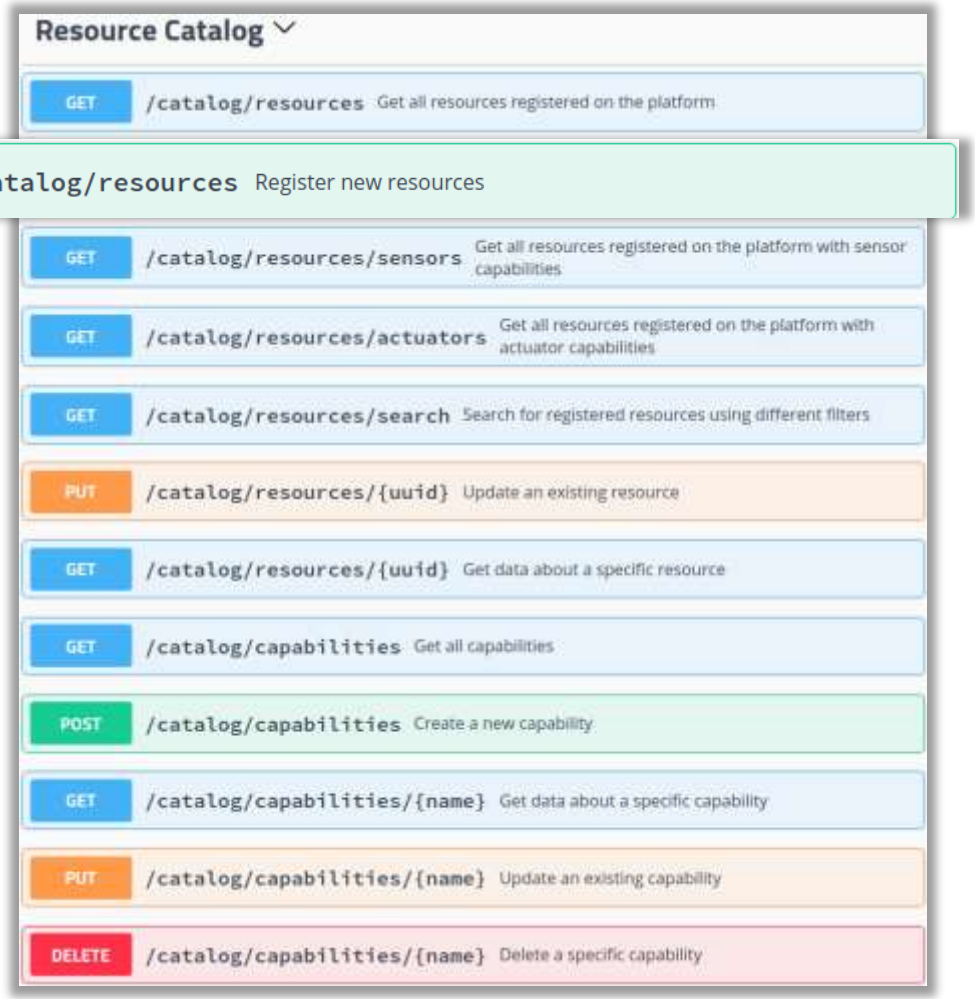
InterSCity
Resource Catalog

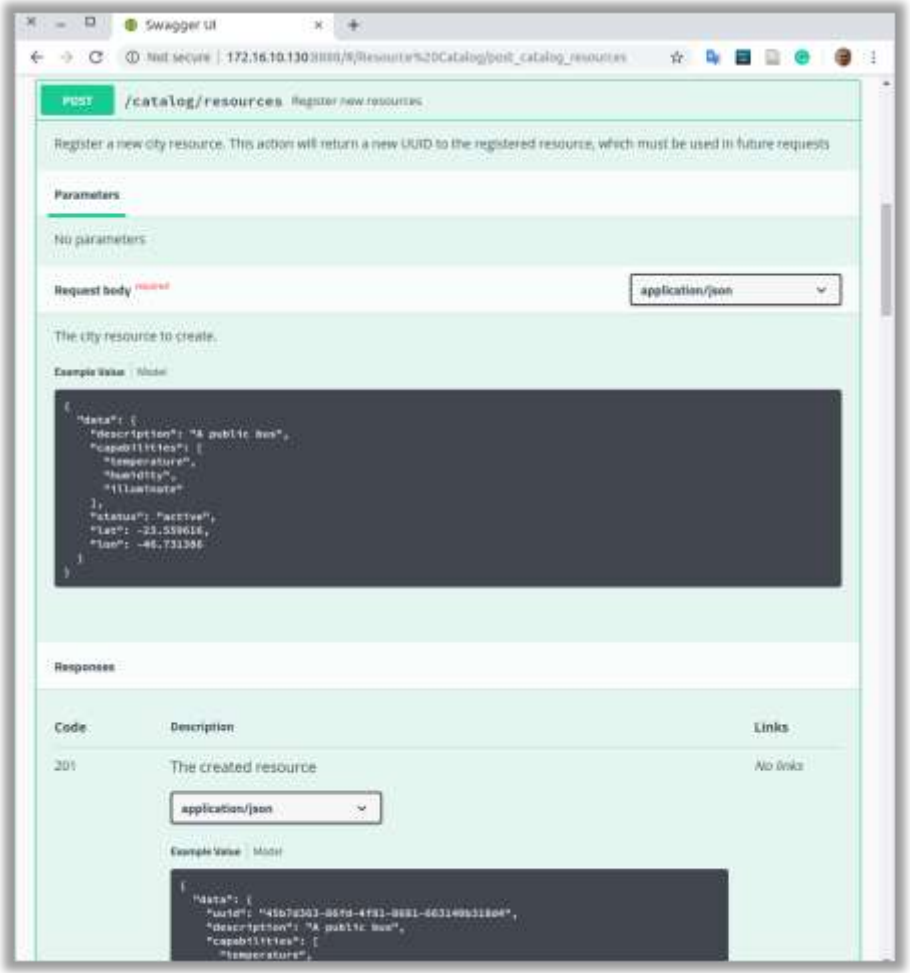




InterSCity
Resource Catalog

InterSCity
Resource Catalog

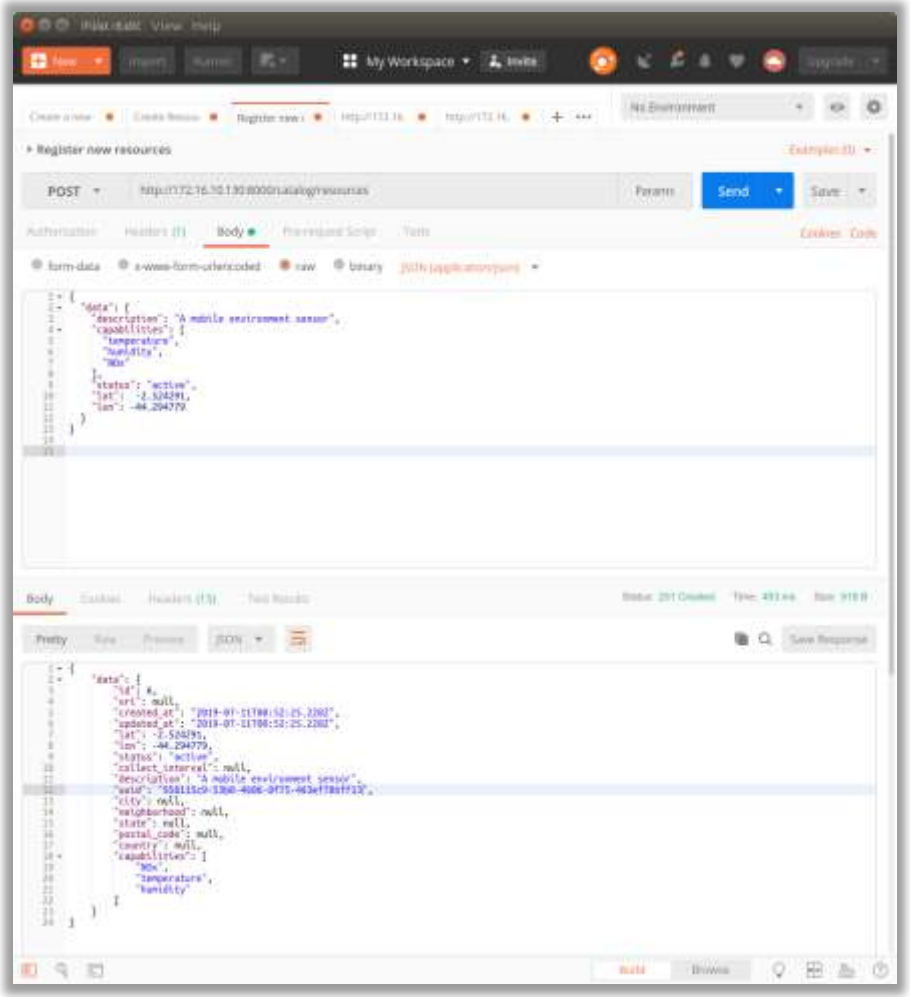




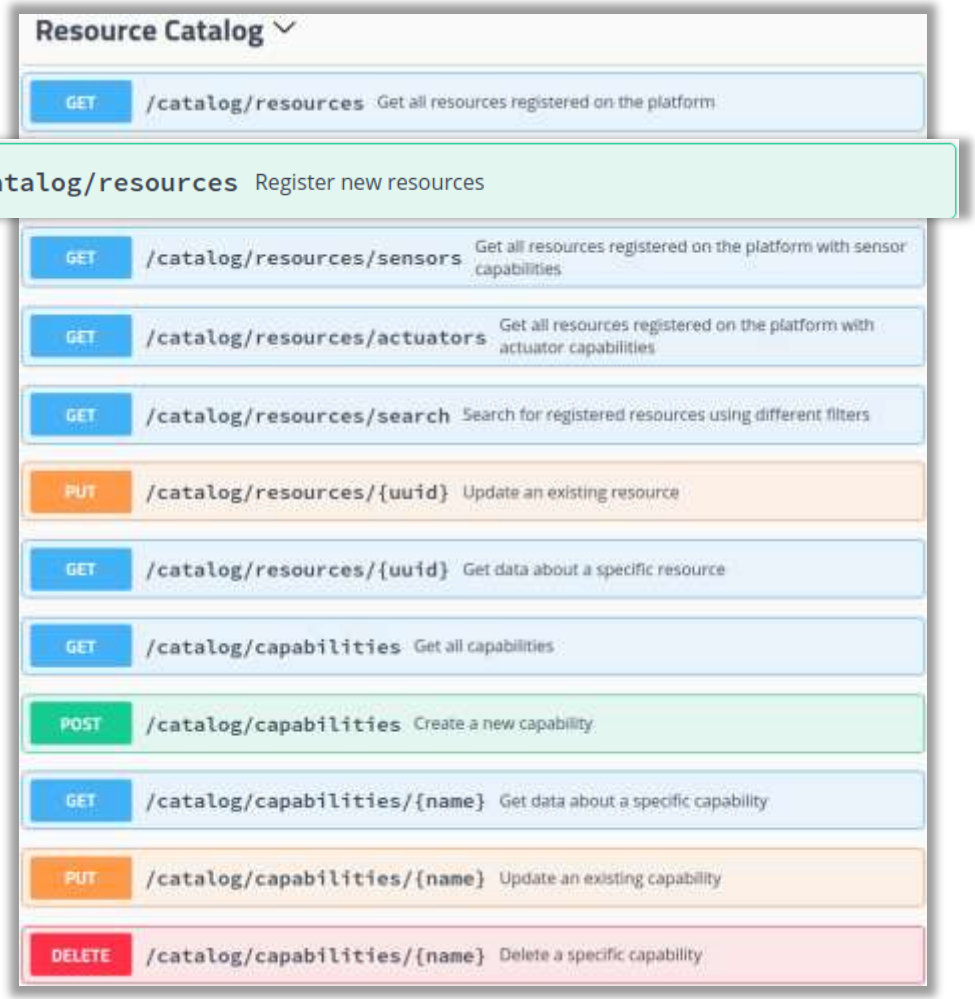
InterSCity
Resource Catalog

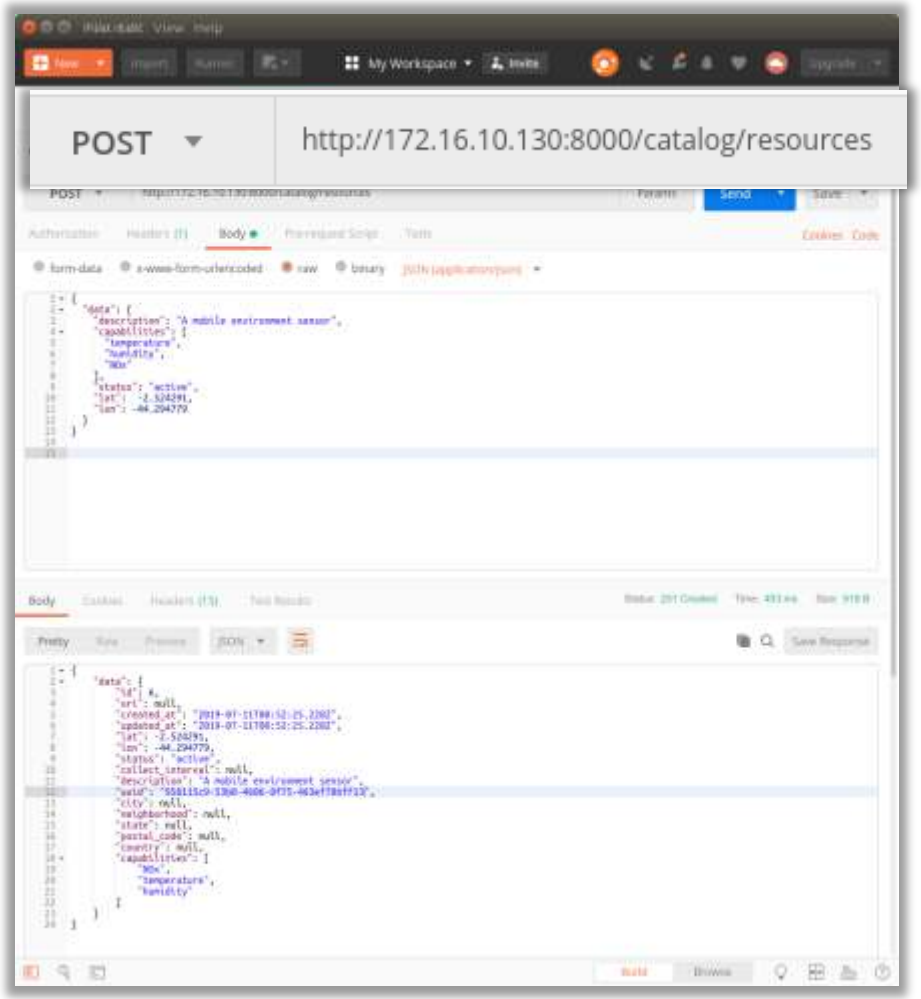
Resource Catalog

GET	/catalog/resources	Get all resources registered on the platform
POST	/catalog/resources	Register new resources
GET	/catalog/resources/sensors	Get all resources registered on the platform with sensor capabilities
GET	/catalog/resources/actuators	Get all resources registered on the platform with actuator capabilities
GET	/catalog/resources/search	Search for registered resources using different filters
PUT	/catalog/resources/{uuid}	Update an existing resource
GET	/catalog/resources/{uuid}	Get data about a specific resource
GET	/catalog/capabilities	Get all capabilities
POST	/catalog/capabilities	Create a new capability
GET	/catalog/capabilities/{name}	Get data about a specific capability
PUT	/catalog/capabilities/{name}	Update an existing capability
DELETE	/catalog/capabilities/{name}	Delete a specific capability

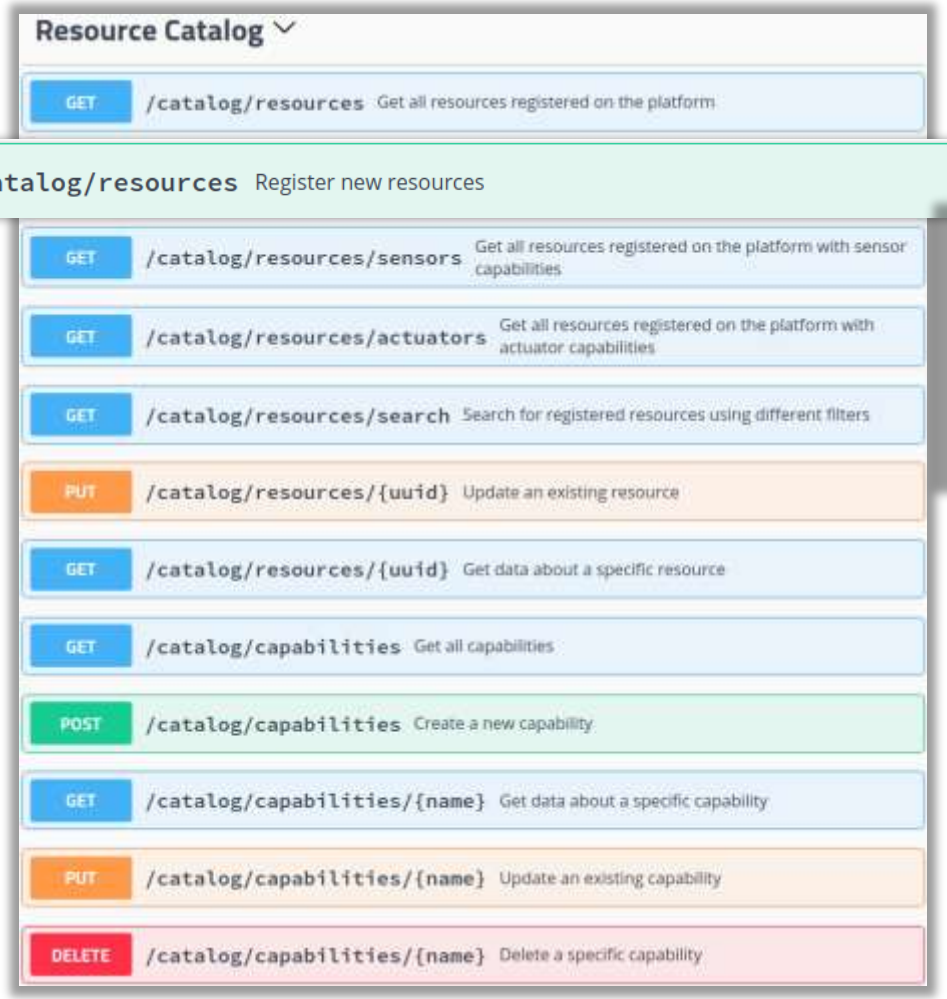


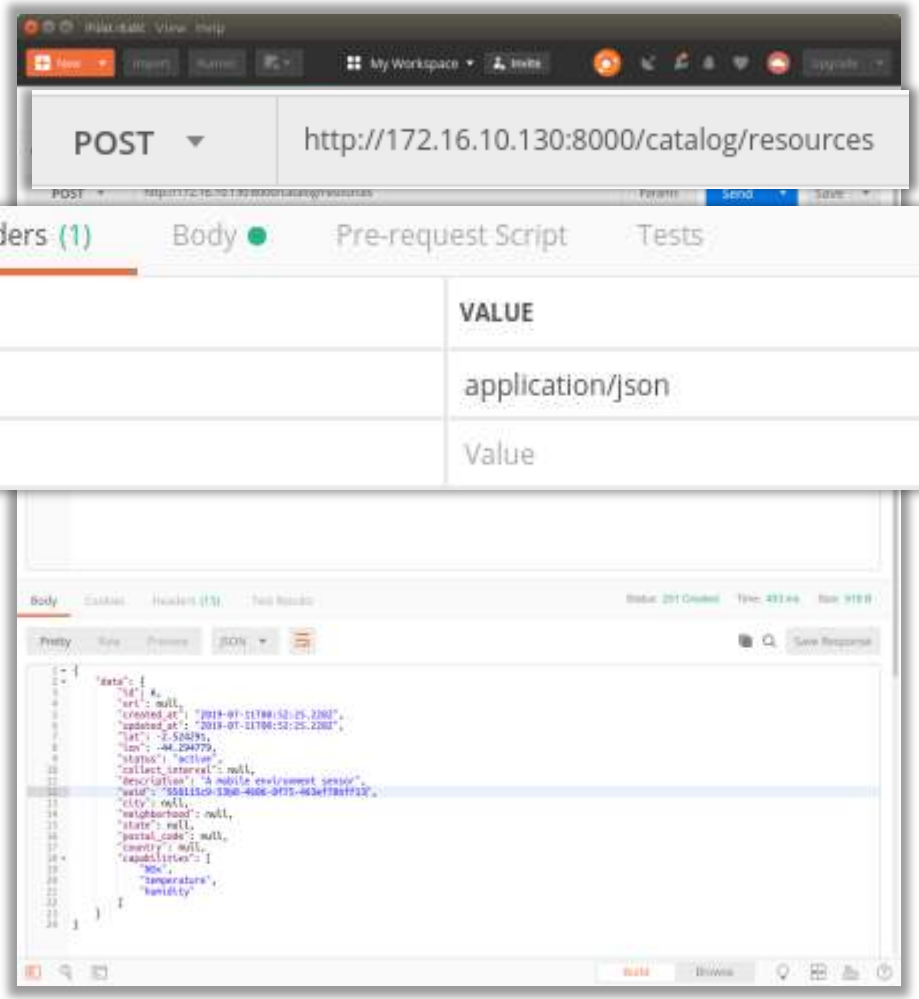
InterSCity
Resource Catalog



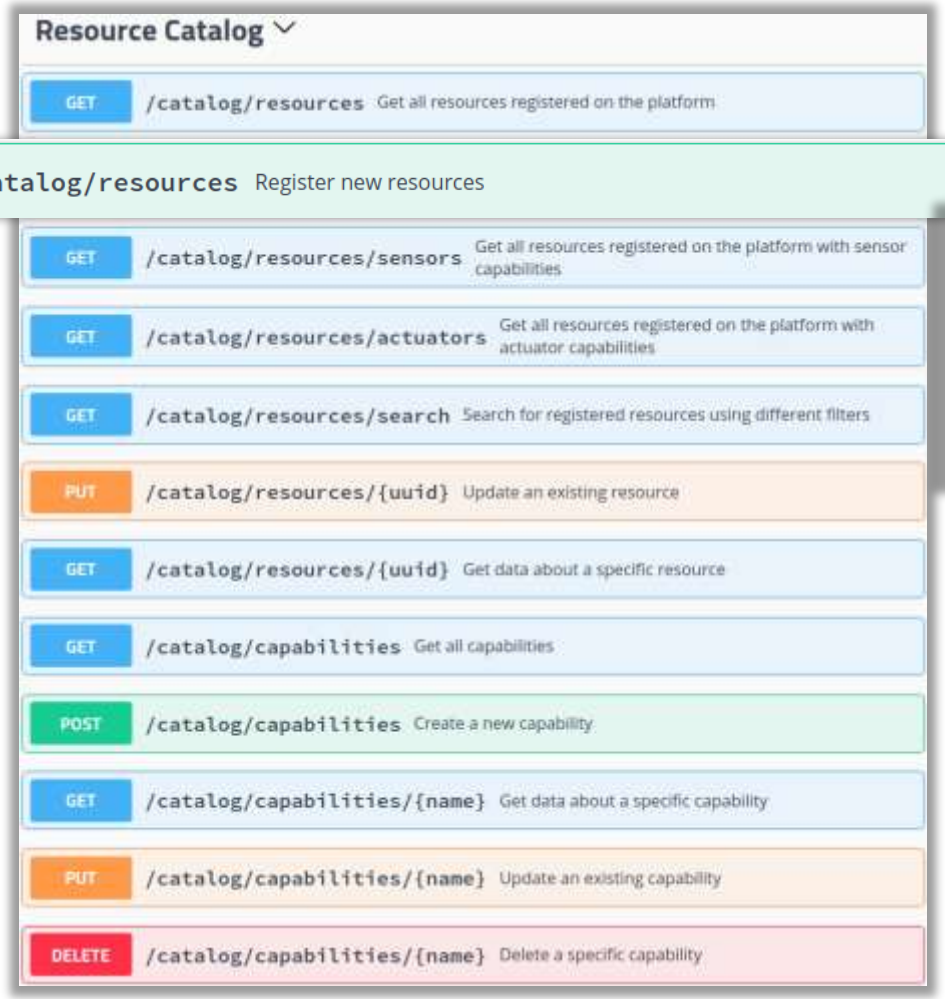


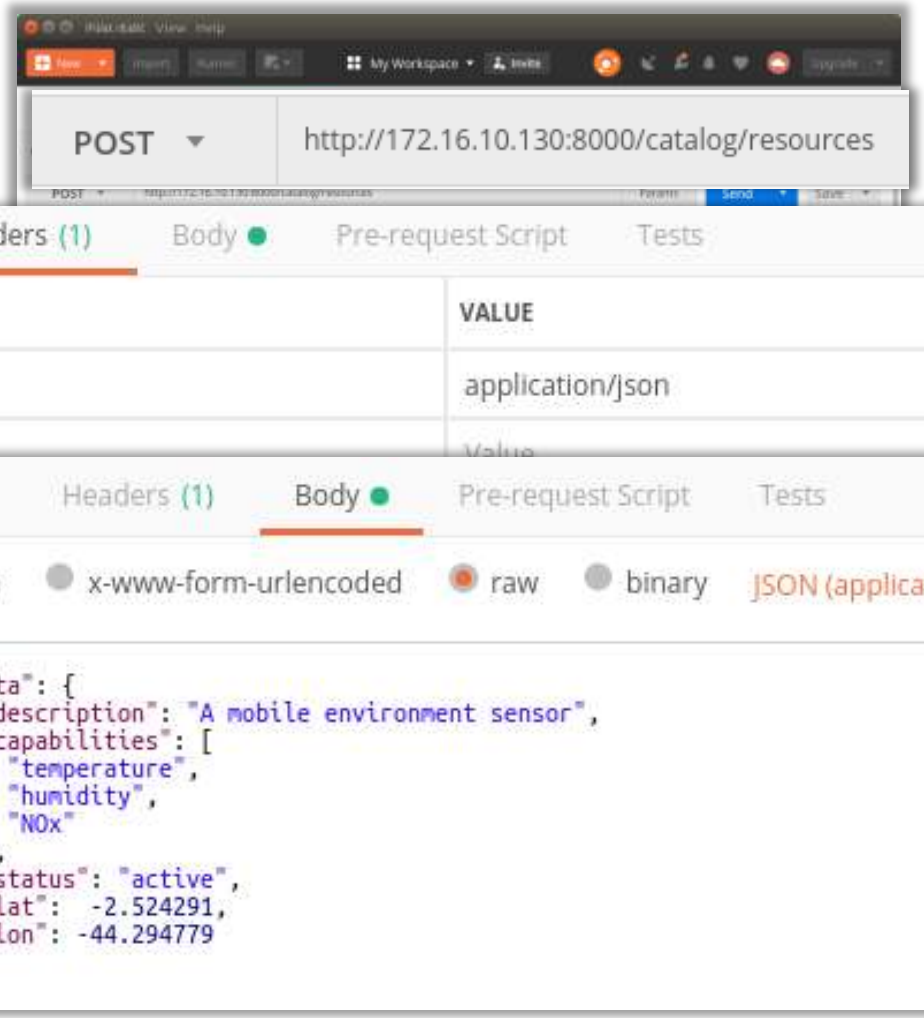
InterSCity
Resource Catalog





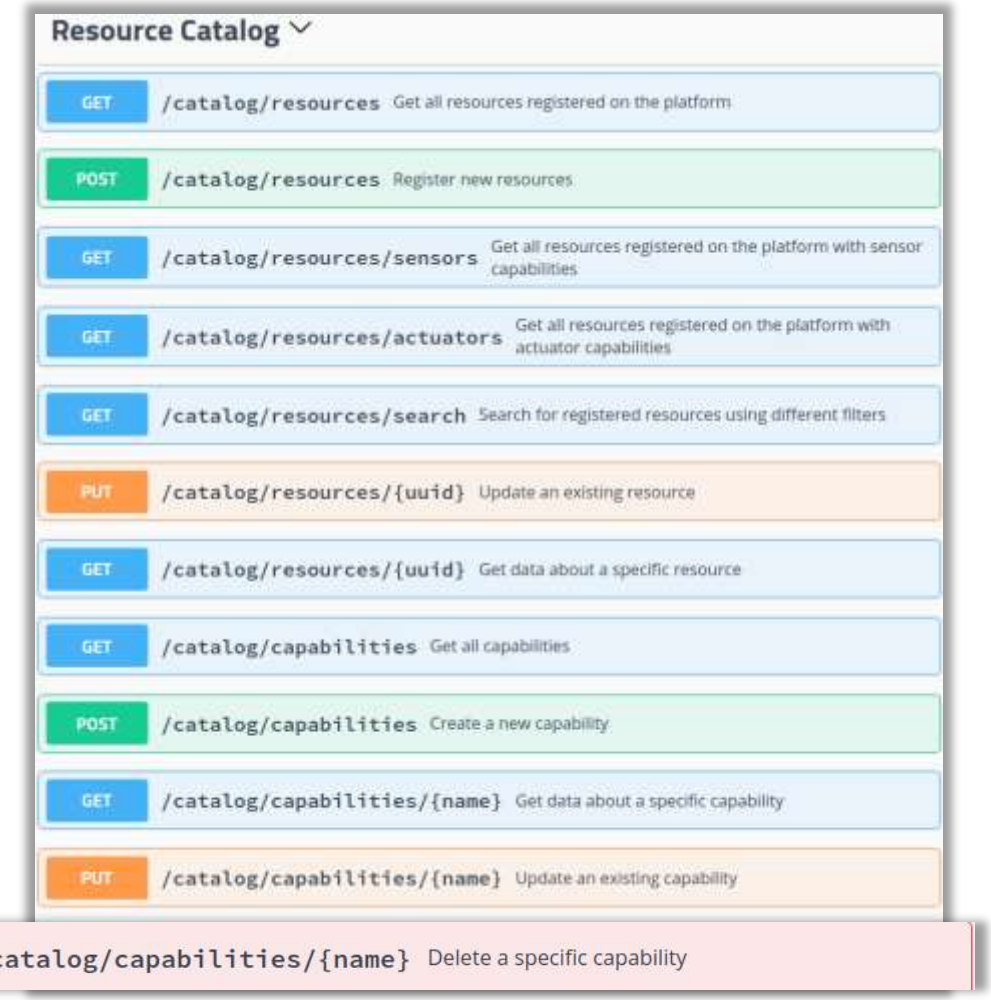
InterSCity
Resource Catalog

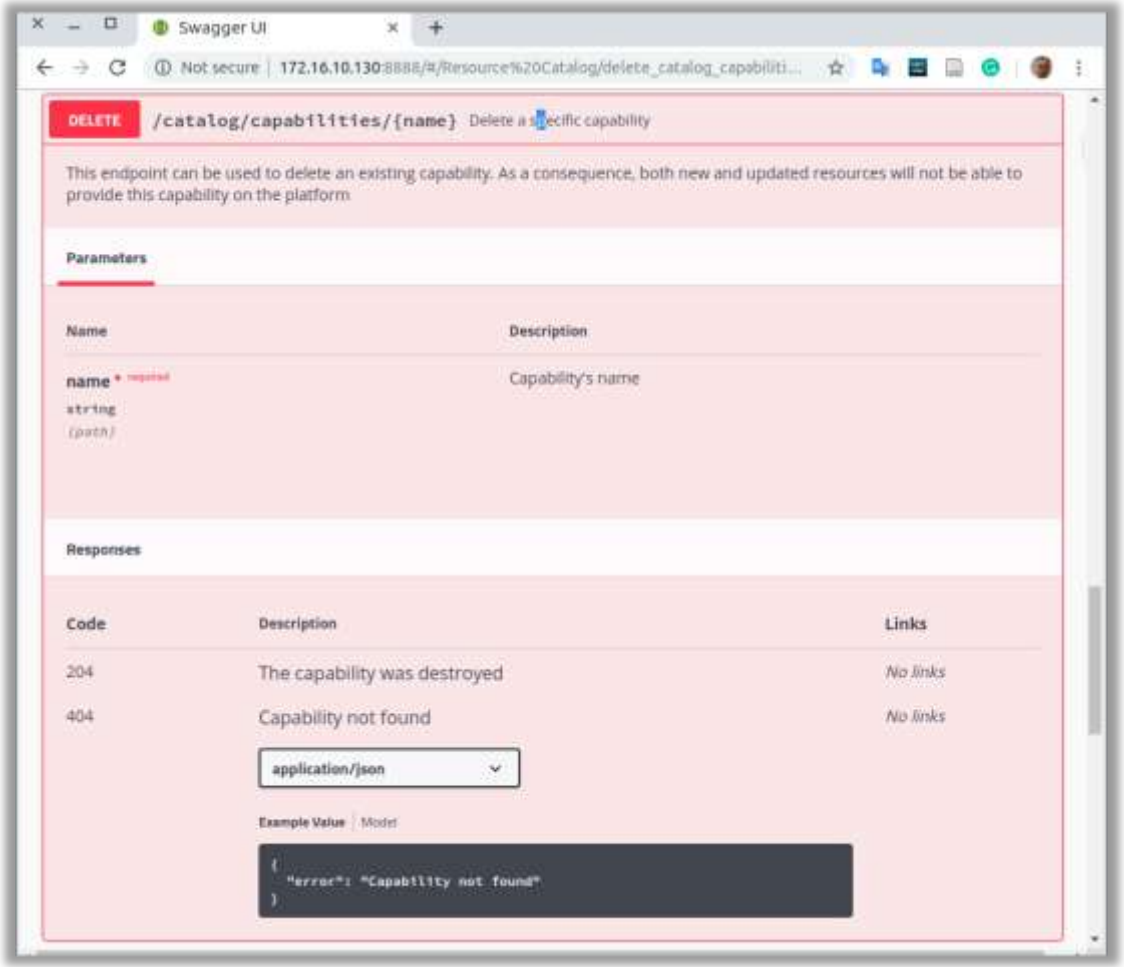




InterSCity
Resource Catalog

InterSCity
Resource Catalog

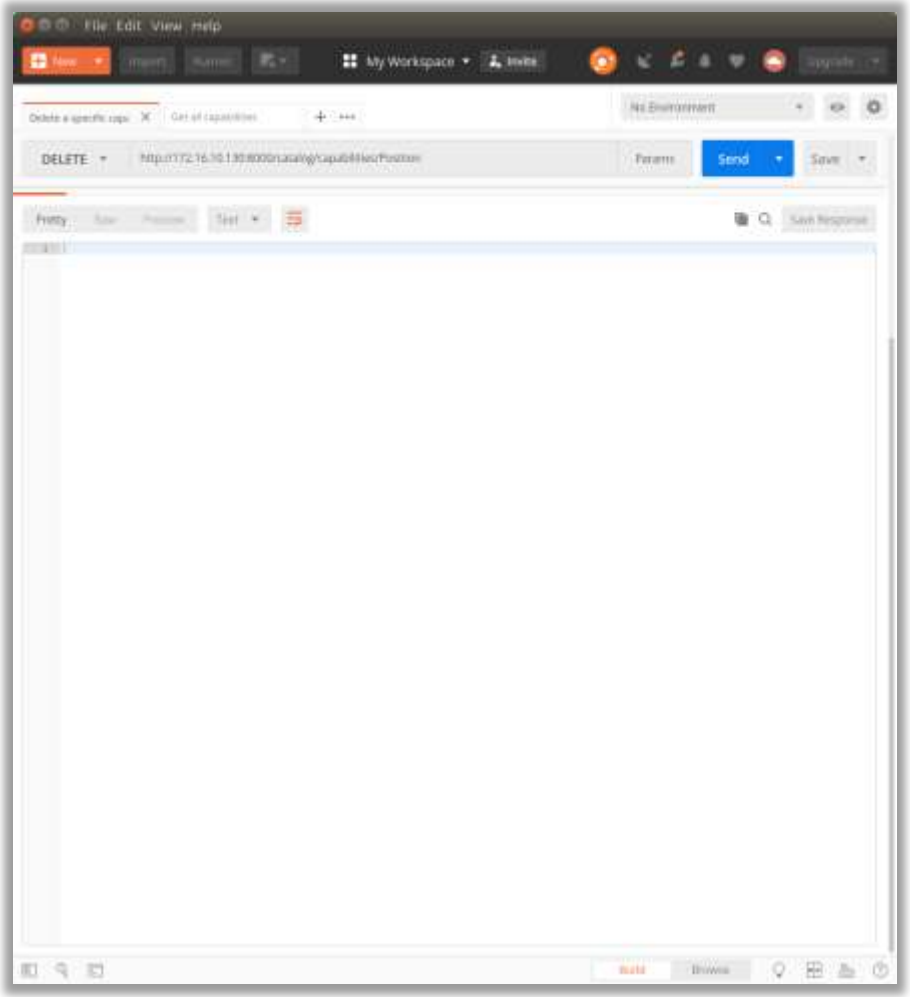




InterSCity
Resource Catalog

Resource Catalog

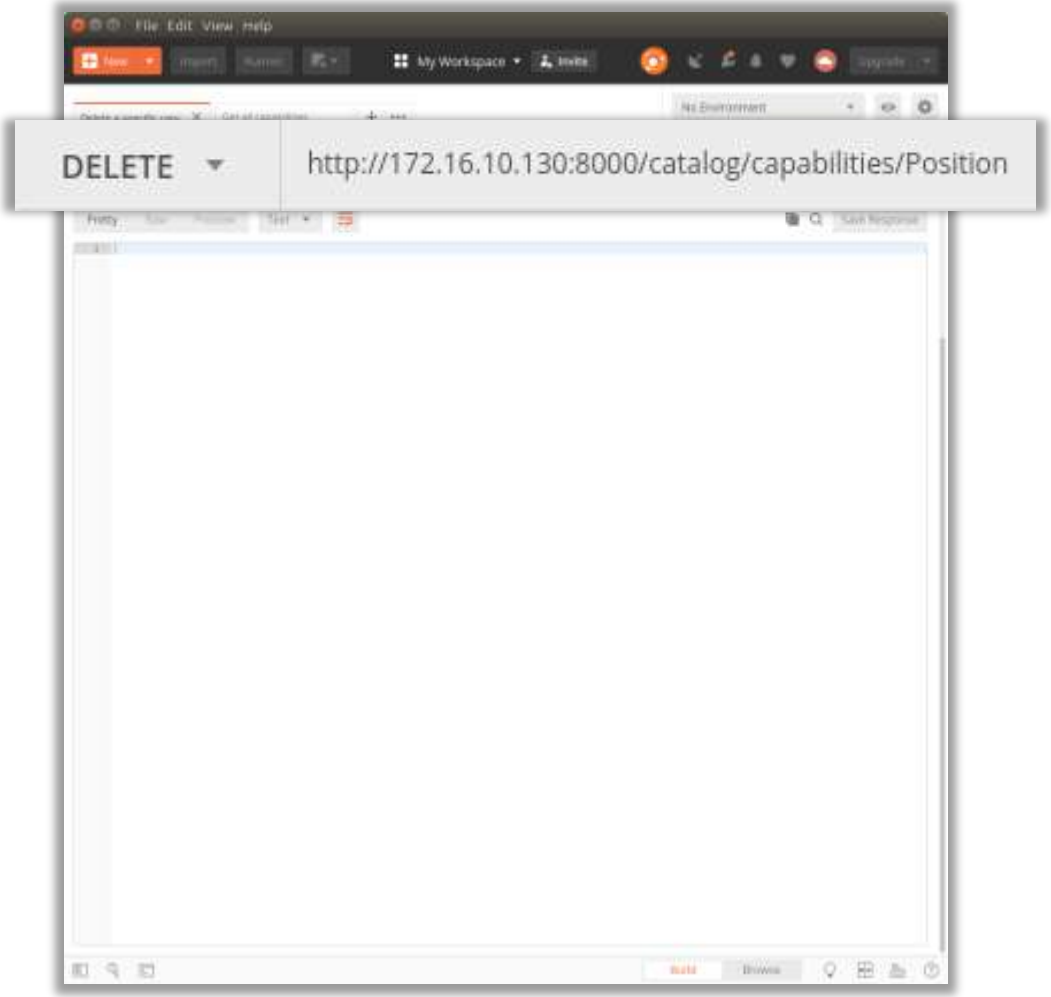
GET	/catalog/resources	Get all resources registered on the platform
POST	/catalog/resources	Register new resources
GET	/catalog/resources/sensors	Get all resources registered on the platform with sensor capabilities
GET	/catalog/resources/actuators	Get all resources registered on the platform with actuator capabilities
GET	/catalog/resources/search	Search for registered resources using different filters
PUT	/catalog/resources/{uuid}	Update an existing resource
GET	/catalog/resources/{uuid}	Get data about a specific resource
GET	/catalog/capabilities	Get all capabilities
POST	/catalog/capabilities	Create a new capability
GET	/catalog/capabilities/{name}	Get data about a specific capability
PUT	/catalog/capabilities/{name}	Update an existing capability
DELETE	/catalog/capabilities/{name}	Delete a specific capability



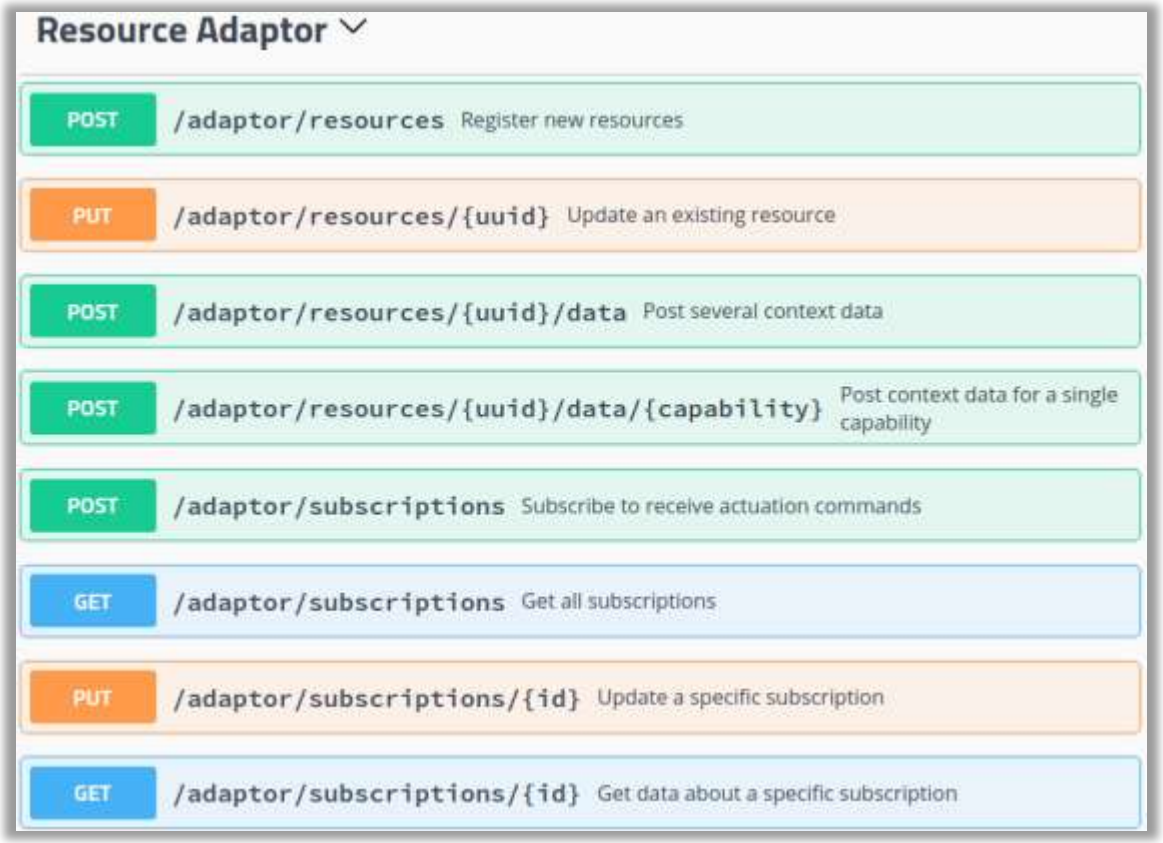
InterSCity
Resource Catalog

Resource Catalog

GET	/catalog/resources	Get all resources registered on the platform
POST	/catalog/resources	Register new resources
GET	/catalog/resources/sensors	Get all resources registered on the platform with sensor capabilities
GET	/catalog/resources/actuators	Get all resources registered on the platform with actuator capabilities
GET	/catalog/resources/search	Search for registered resources using different filters
PUT	/catalog/resources/{uuid}	Update an existing resource
GET	/catalog/resources/{uuid}	Get data about a specific resource
GET	/catalog/capabilities	Get all capabilities
POST	/catalog/capabilities	Create a new capability
GET	/catalog/capabilities/{name}	Get data about a specific capability
PUT	/catalog/capabilities/{name}	Update an existing capability
DELETE	/catalog/capabilities/{name}	Delete a specific capability



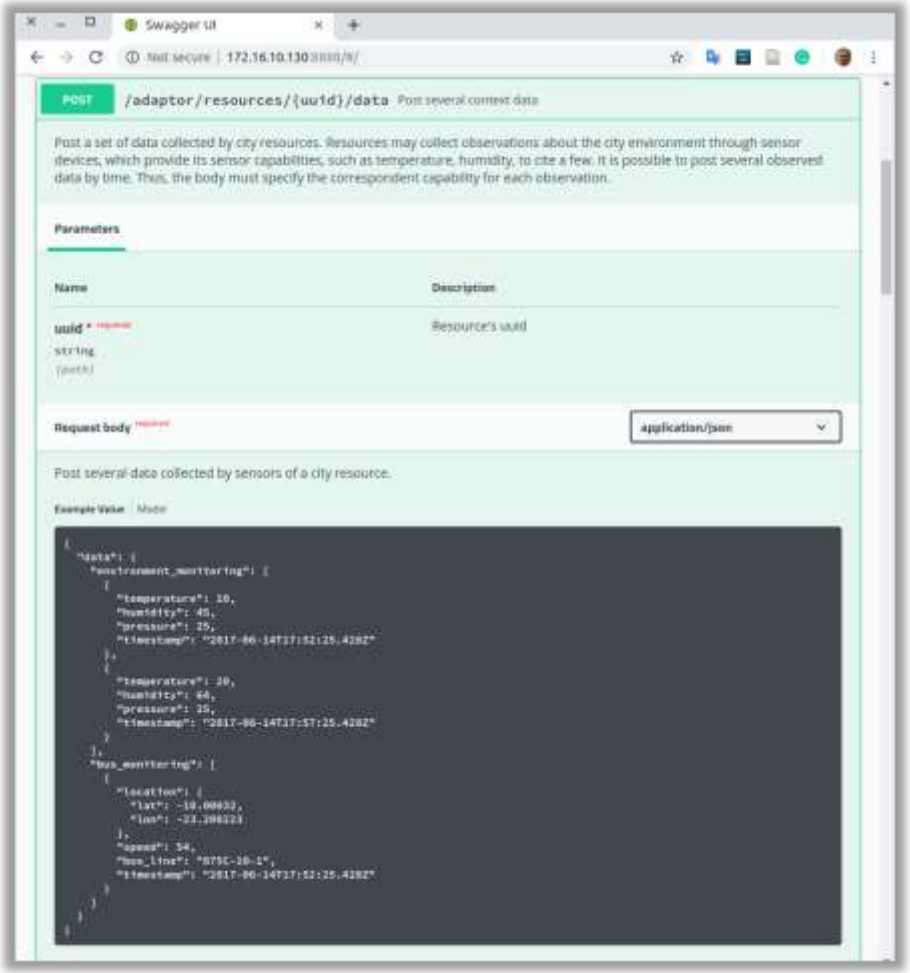
InterSCity
Resource Adaptor



InterSCity
Resource Adaptor

Resource Adaptor

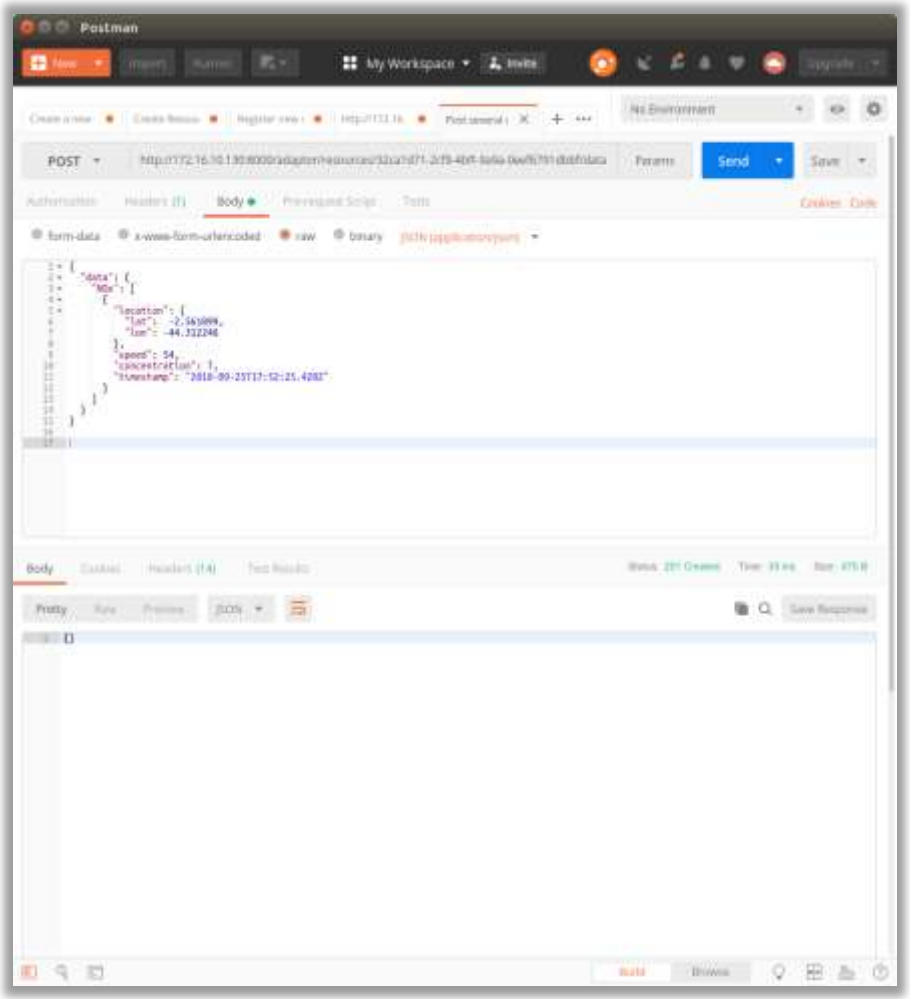
POST	/adaptor/resources	Register new resources
PUT	/adaptor/resources/{uuid}	Update an existing resource
POST	/adaptor/resources/{uuid}/data	Post several context data
POST	/adaptor/resources/{uuid}/data/{capability}	Post context data for a single capability
POST	/adaptor/subscriptions	Subscribe to receive actuation commands
GET	/adaptor/subscriptions	Get all subscriptions
PUT	/adaptor/subscriptions/{id}	Update a specific subscription
GET	/adaptor/subscriptions/{id}	Get data about a specific subscription



InterSCity
Resource Adaptor

Resource Adaptor

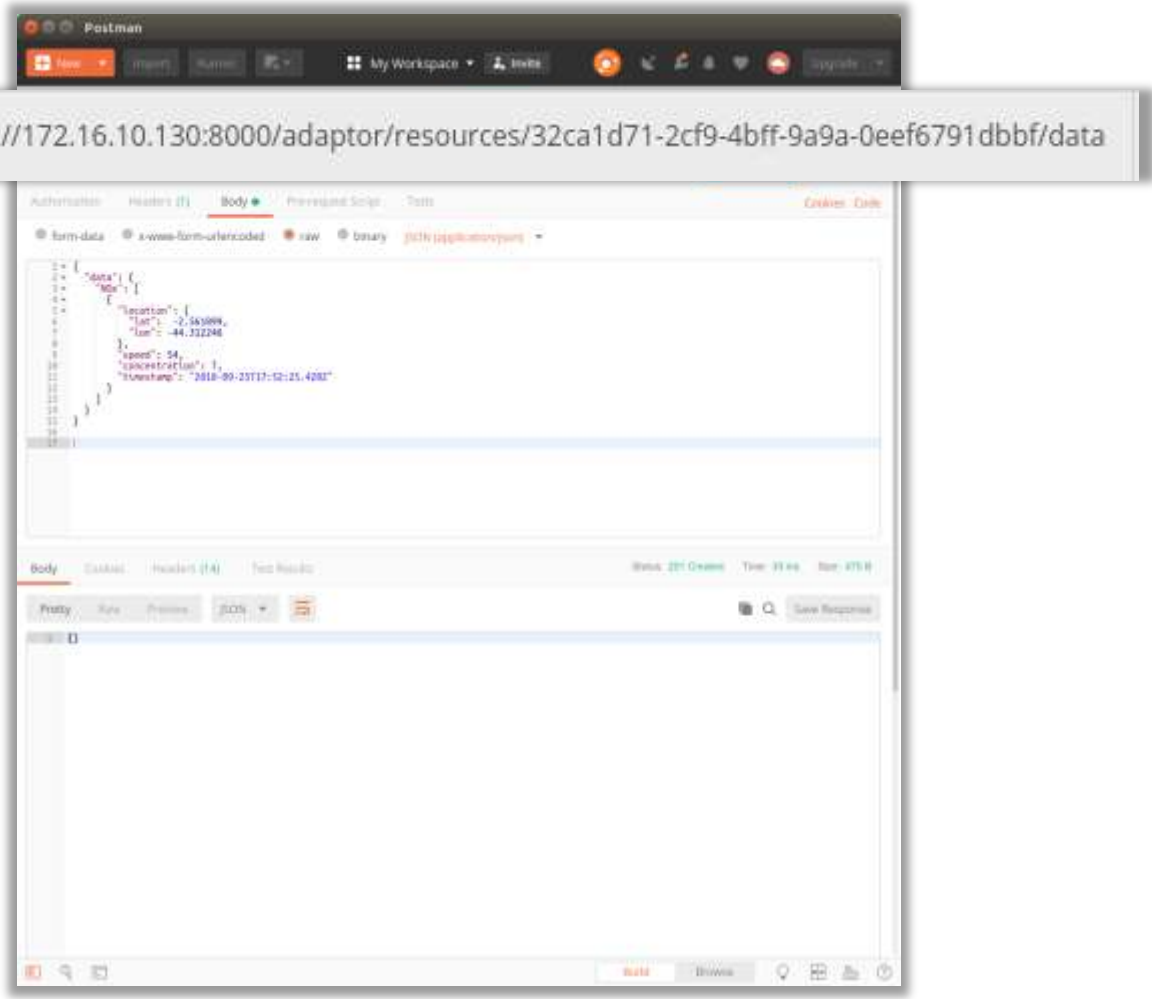
POST	/adaptor/resources	Register new resources
PUT	/adaptor/resources/{uuid}	Update an existing resource
POST	/adaptor/resources/{uuid}/data	Post several context data
POST	/adaptor/resources/{uuid}/data/{capability}	Post context data for a single capability
POST	/adaptor/subscriptions	Subscribe to receive actuation commands
GET	/adaptor/subscriptions	Get all subscriptions
PUT	/adaptor/subscriptions/{id}	Update a specific subscription
GET	/adaptor/subscriptions/{id}	Get data about a specific subscription



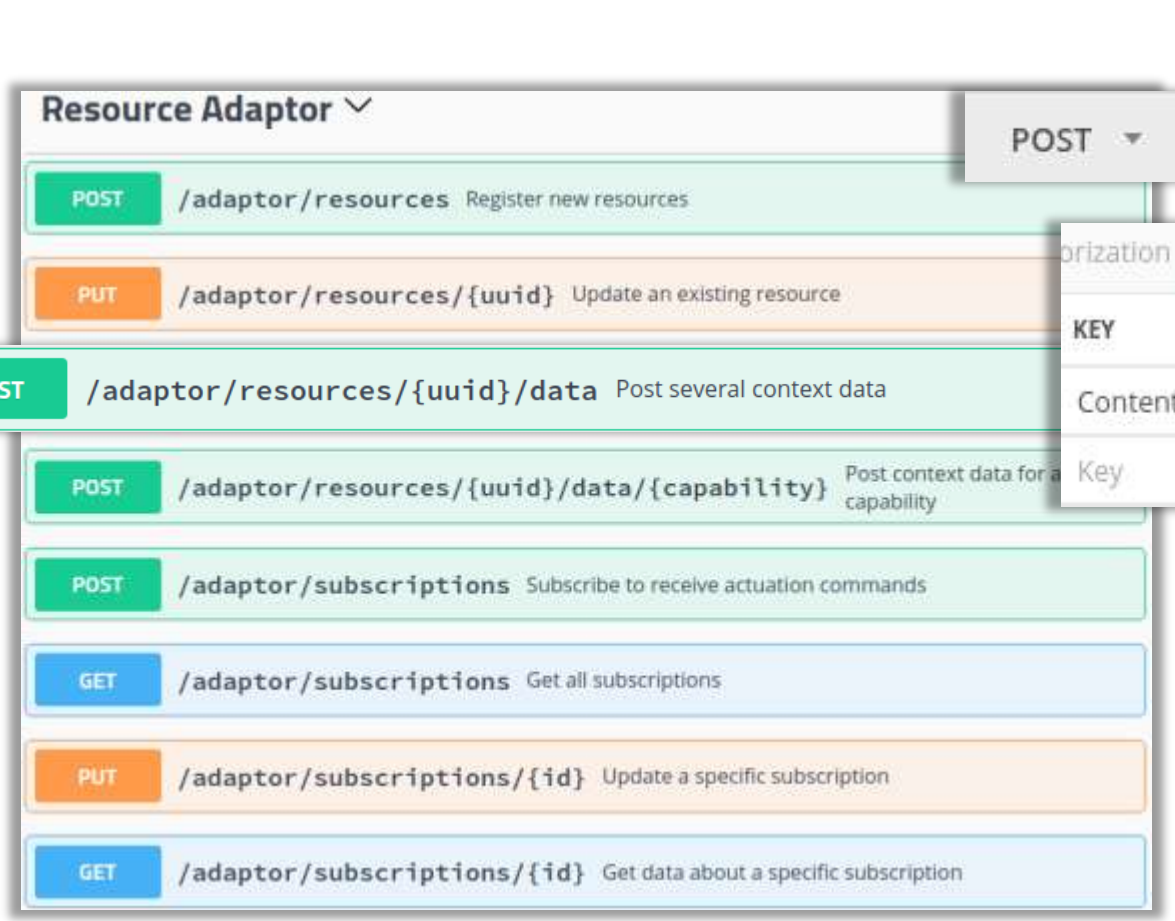
InterSCity
Resource Adaptor

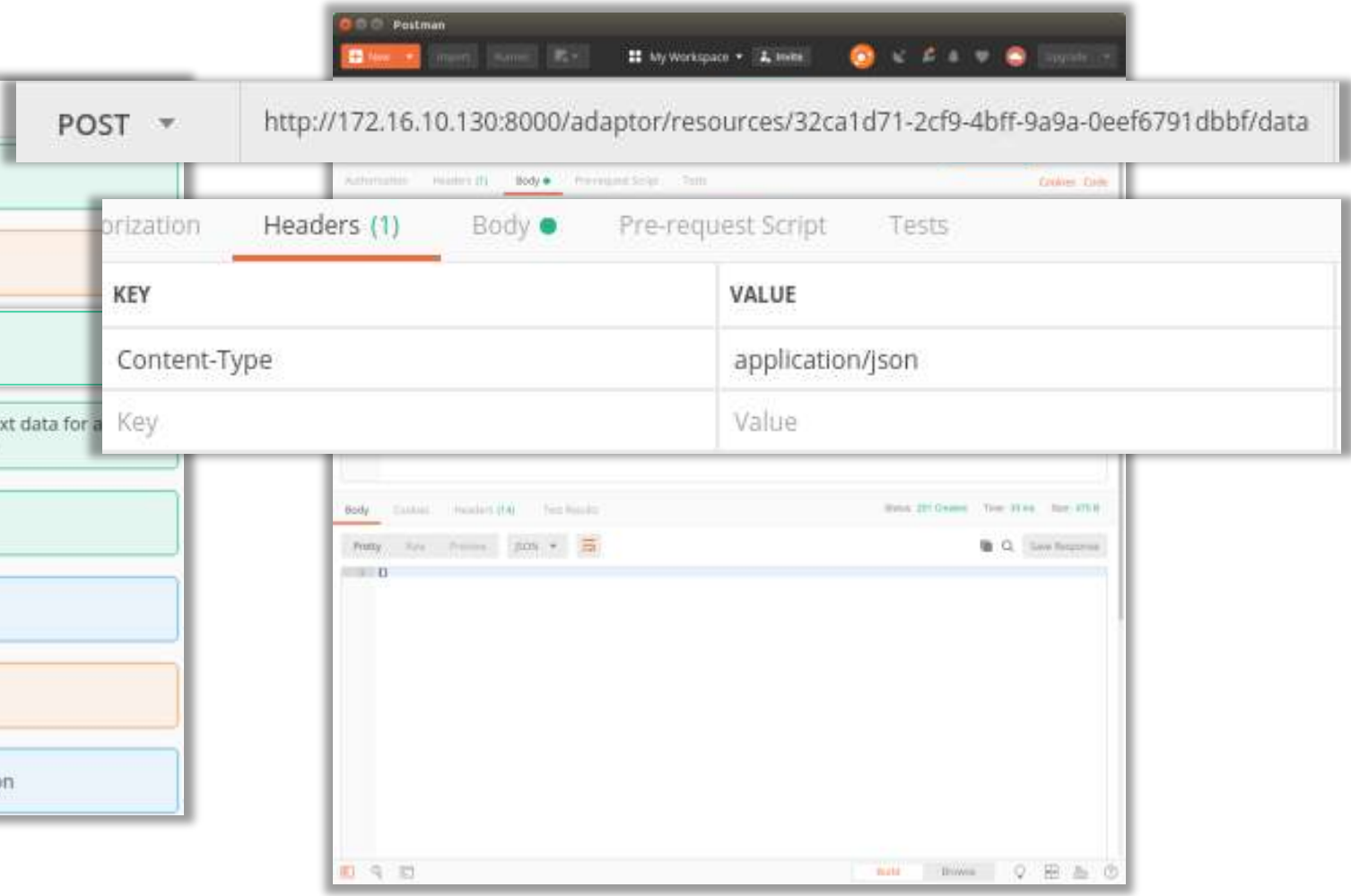
Resource Adaptor

POST	/adaptor/resources	Register new resources
PUT	/adaptor/resources/{uuid}	Update an existing resource
POST	/adaptor/resources/{uuid}/data	Post several context data
POST	/adaptor/resources/{uuid}/data/{capability}	Post context data for a single capability
POST	/adaptor/subscriptions	Subscribe to receive actuation commands
GET	/adaptor/subscriptions	Get all subscriptions
PUT	/adaptor/subscriptions/{id}	Update a specific subscription
GET	/adaptor/subscriptions/{id}	Get data about a specific subscription

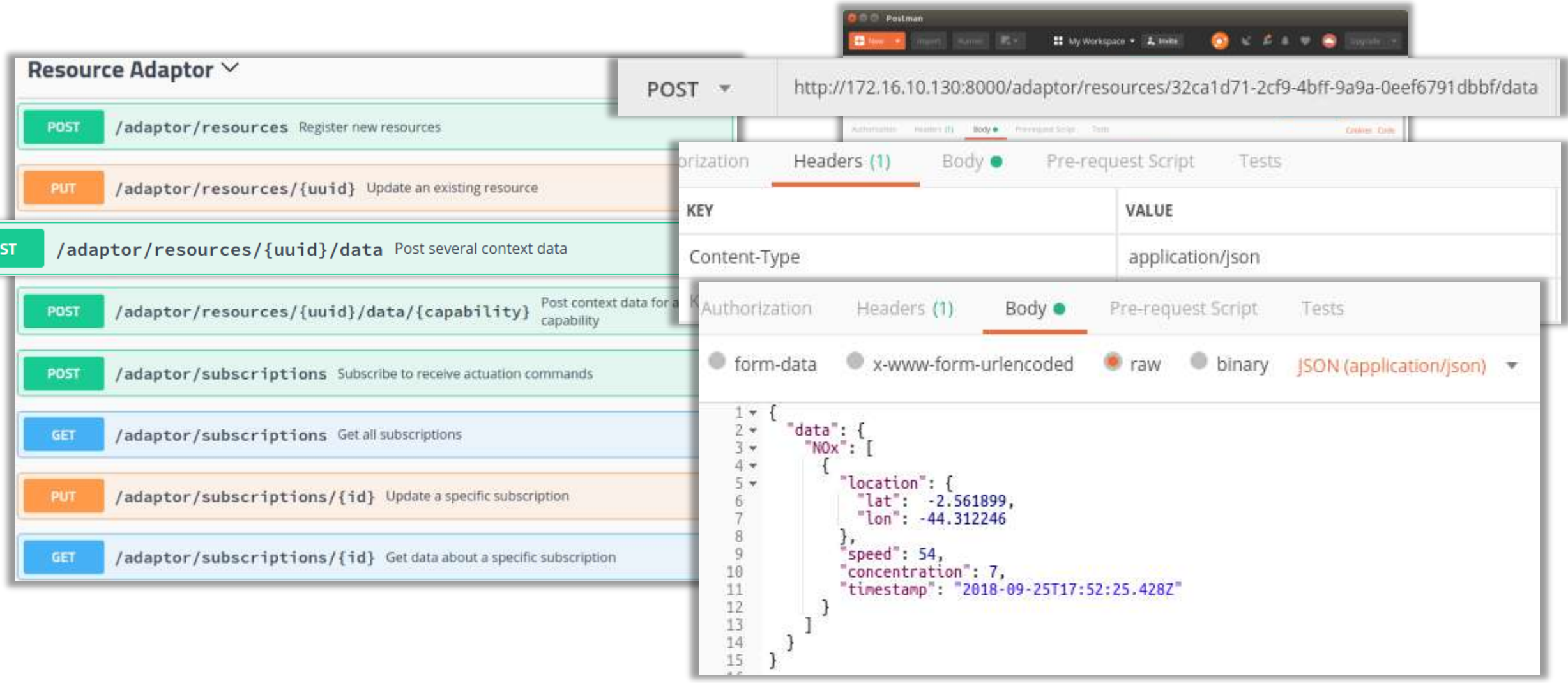


InterSCity
Resource Adaptor

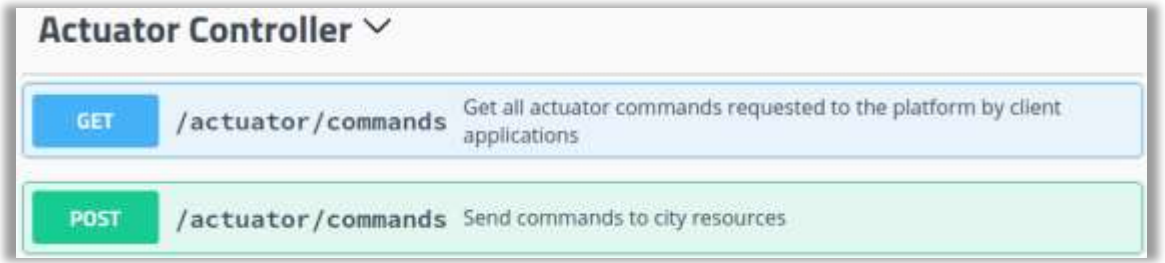




InterSCity
Resource Adaptor

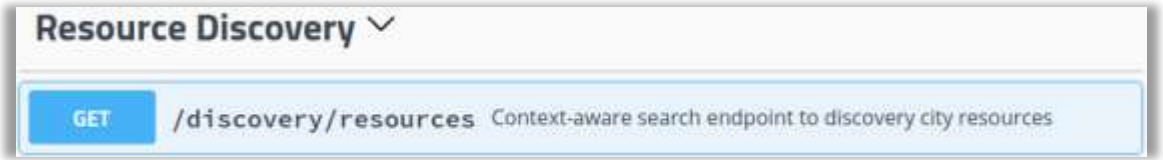


InterSCity
Actuator Controller



InterSCity
Data Collector

InterSCity
Resource Discovery



InterSCity
Resource Discovery

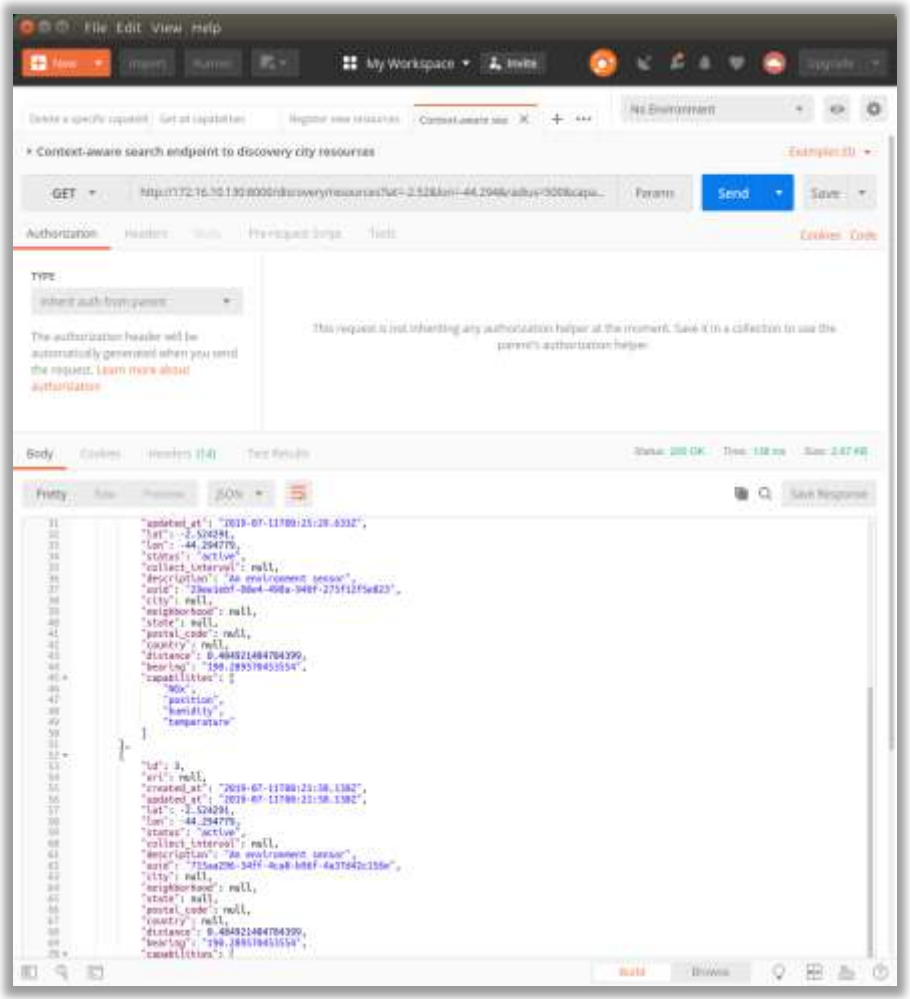
Resource Discovery

GET `/discovery/resources` Context-aware search endpoint to discovery city resources

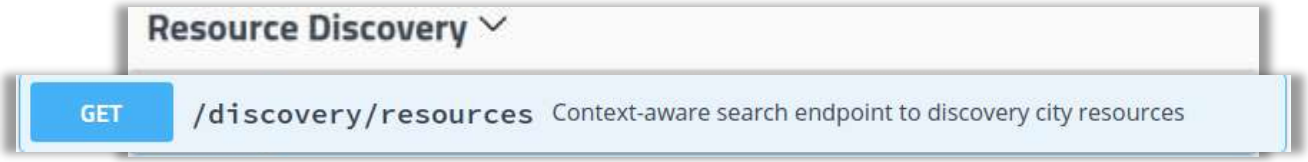
InterSCity
Resource Discovery

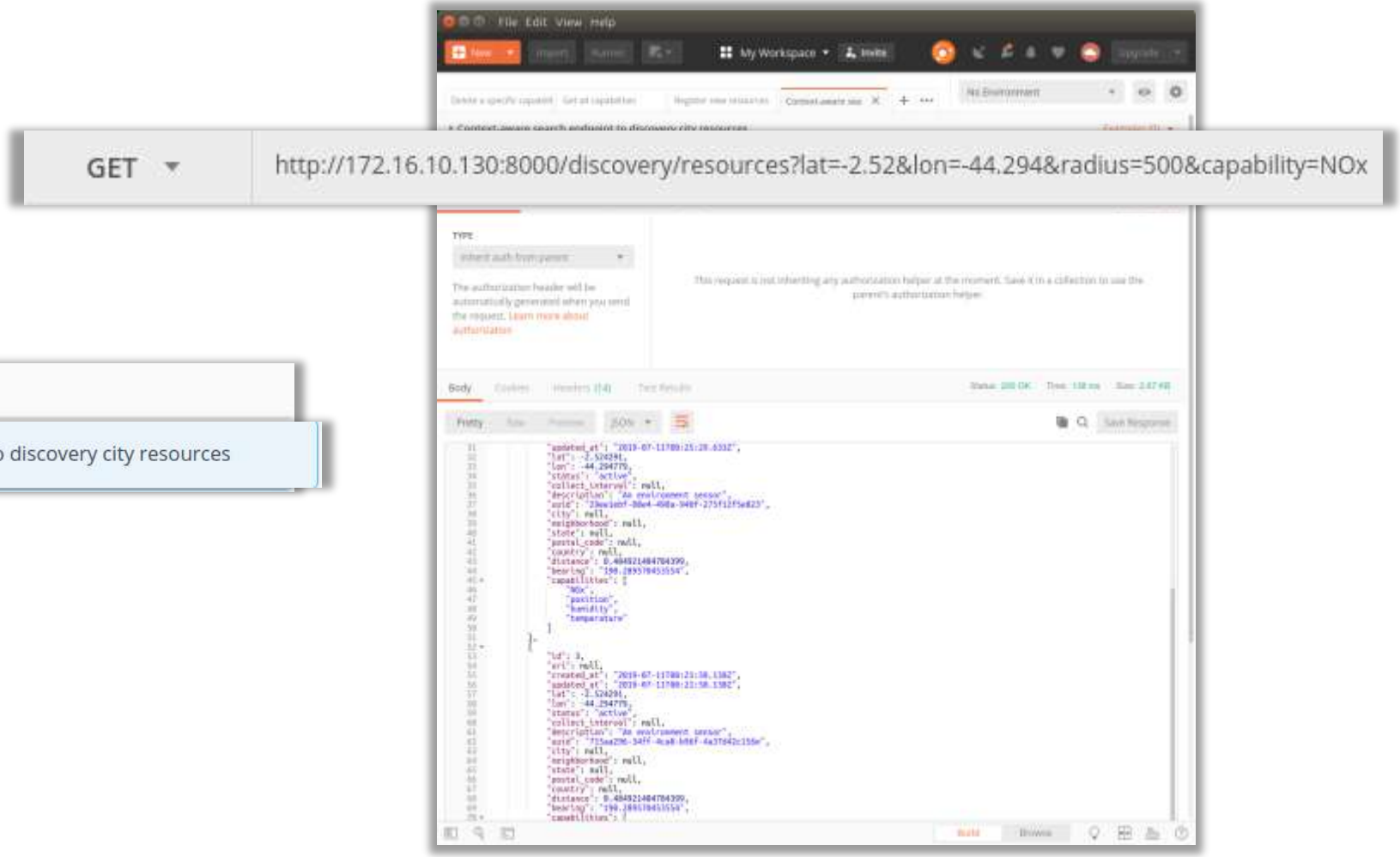
Resource Discovery

GET `/discovery/resources` Context-aware search endpoint to discovery city resources



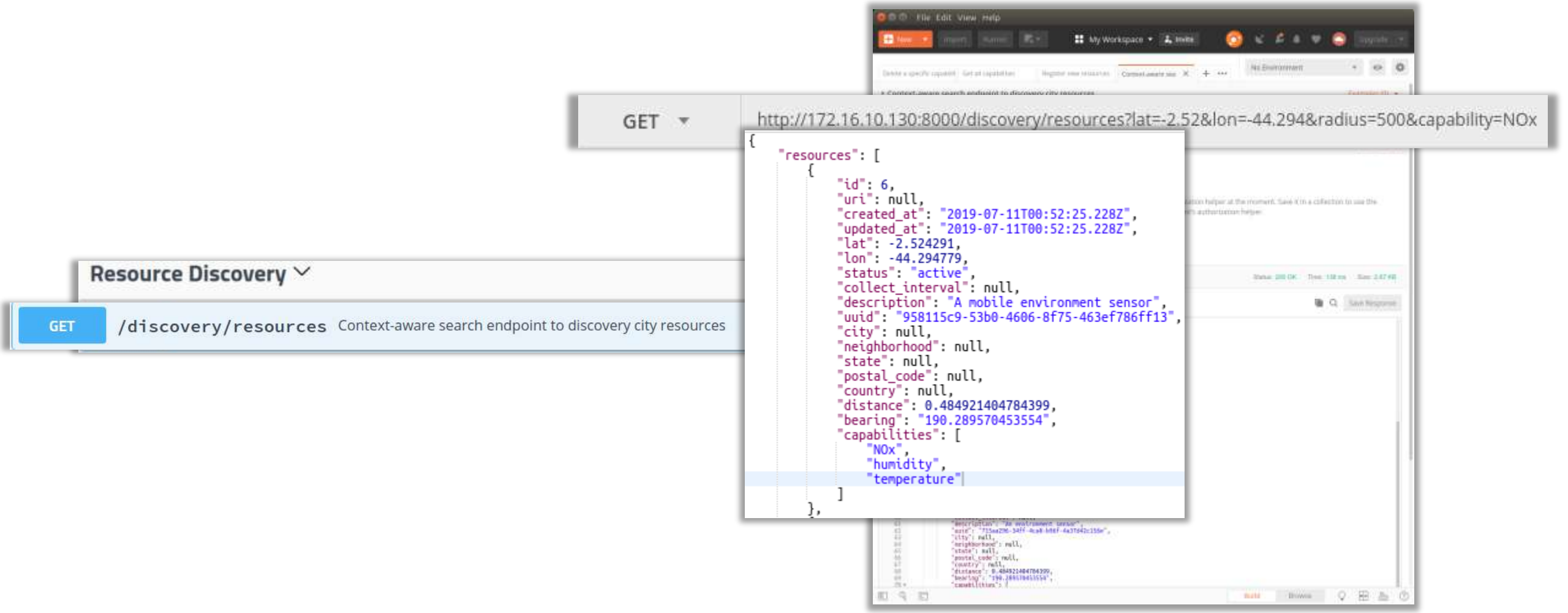
InterSCity
Resource Discovery





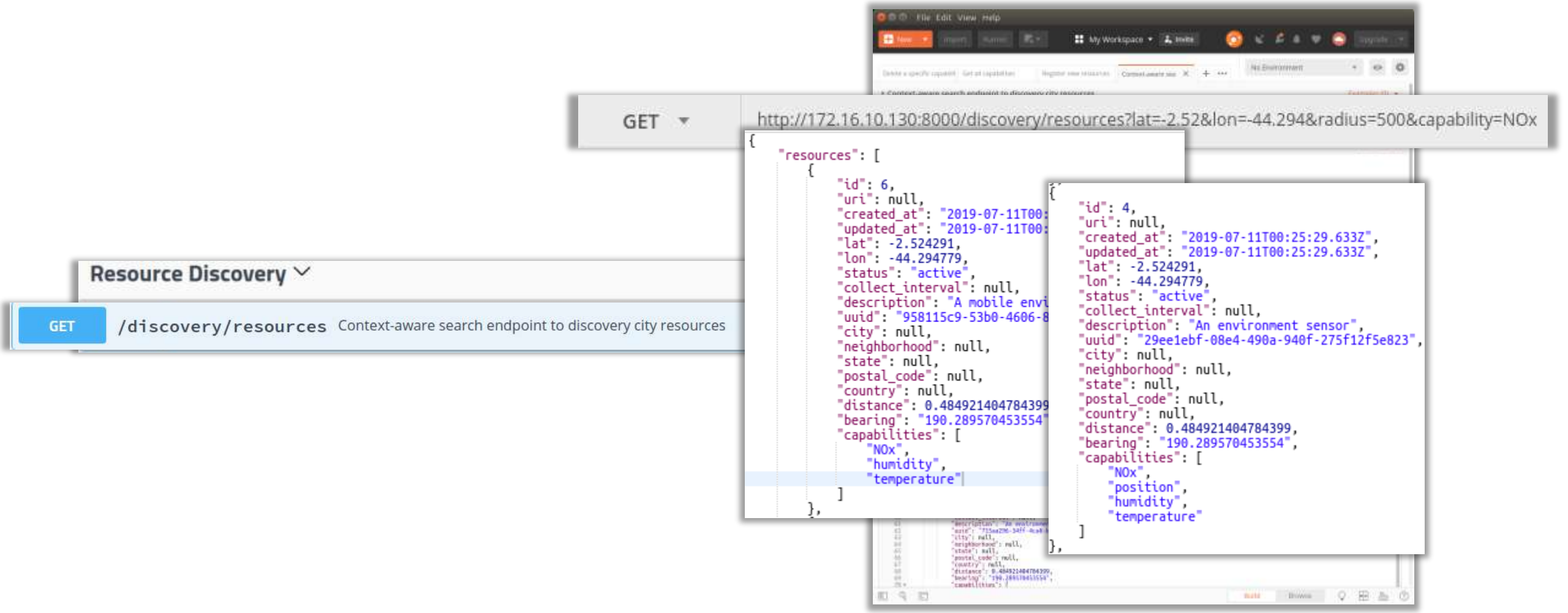
InterSCity

Resource Discovery



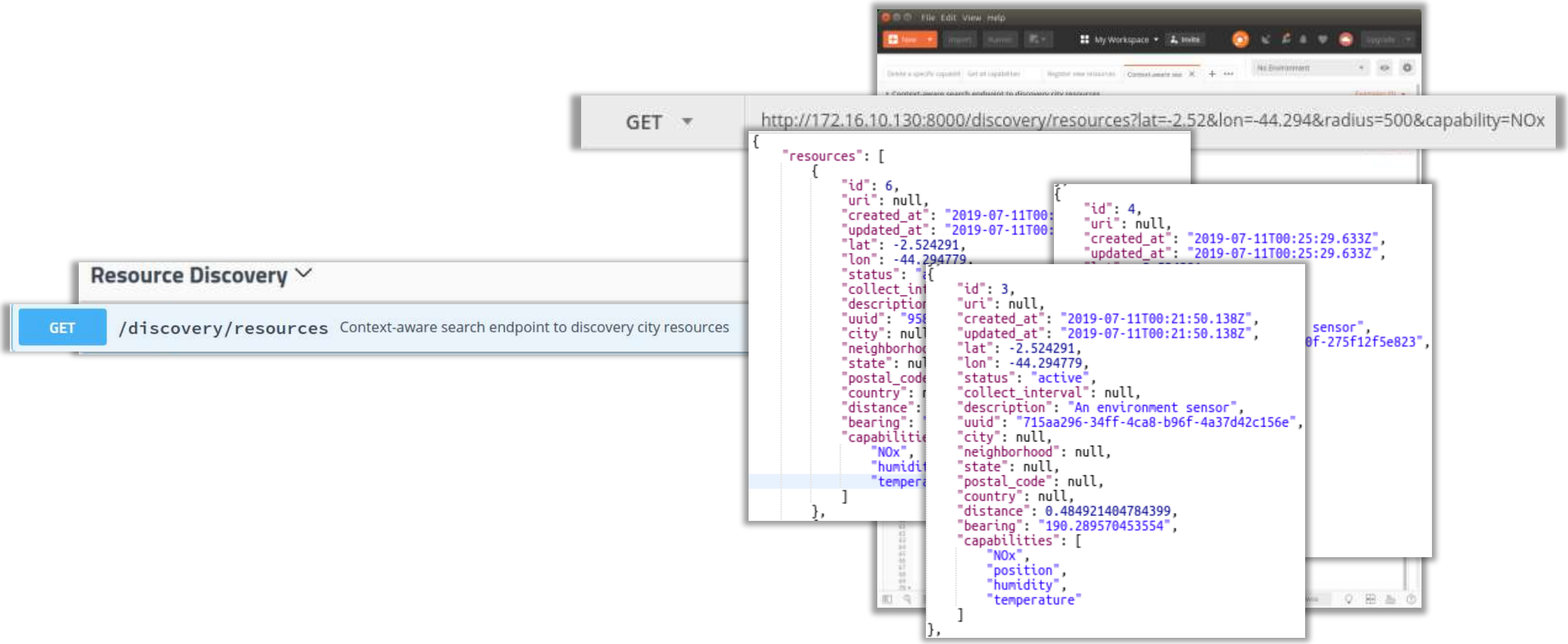
InterSCity

Resource Discovery



InterSCity

Resource Discovery



Perguntas?

Exercícios

Exercício

- Crie uma “capacidade” para monitorar a qualidade do ar
- Crie 3 recursos que possuam capacidade de monitorar a qualidade do ar
- Entre com os dados nos recursos
  - Suponha que os recursos são móveis

InterSCity

Via Java

Classe MyHTTPConnection

Pacotes

```
package br.com.meslin.alert.connection;

import java.io.BufferedReader;
import java.io.DataOutputStream;
import java.io.IOException;
import java.io.InputStreamReader;
import java.net.HttpURLConnection;
import java.net.MalformedURLException;
import java.net.URL;

import br.com.meslin.alert.util.Debug;
```

Classe MyHTTPConnection

Construtor e método auxiliar

```
public class MyHTTPConnection {
  private static final String USER_AGENT = "Mozilla/5.0";
  private String interSCityURI;

  public MyHTTPConnection() {
    interSCityURI = Constants.INTERSCITY_URL;
  }

  public MyHTTPConnection(String interSCityIPAddress) {
    if(interSCityIPAddress == null) {
      interSCityURI = Constants.INTERSCITY_URL;
    } else {
      interSCityURI = "http://" + interSCityIPAddress + ":8000";
    }
  }
  ...
}

public String getIpAddress() {
  return interSCityURI;
}
```

Classe MyHTTPConnection

Delete

```
public String sendDelete(String directory, String data) throws Exception {
  final String url = interSCityURI + "/" + directory + "/" + data;
  final String method = "DELETE";

  HttpURLConnection con = (HttpURLConnection) (new URL(url)).openConnection();
  con.setRequestMethod(method);
  con.setRequestProperty("User-agent", USER_AGENT);
  con.setDoOutput(true);

  int responseCode = con.getResponseCode();
  if(responseCode /100 != 2) {   throw new HTTPException(responseCode); }
  BufferedReader in = new BufferedReader(new InputStreamReader(con.getInputStream()));
  String inputLine;
  StringBuffer answer = new StringBuffer();
  while((inputLine = in.readLine()) != null) {
    answer.append(inputLine) ;
  }
  in.close();
  con.disconnect();
  return answer.toString();
}
```

Classe MyHTTPConnection

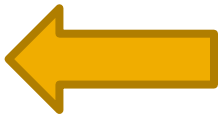
Get

```
public String sendGet(String directory, String data) throws Exception {
  final String url = interSCityURI + "/" + directory + "?" + data;
  final String method = "GET";
  HttpURLConnection con = (HttpURLConnection) (new URL(url)).openConnection();
  con.setRequestMethod(method);
  con.setRequestProperty("User-agent", USER_AGENT);
  int responseCode;
  try {
    responseCode = con.getResponseCode();
  } catch (Exception e) {
    throw new Exception(e.getMessage());
  }
  if(responseCode /100 != 2) { throw new HTTPException(responseCode); }
  BufferedReader in = new BufferedReader(new InputStreamReader(con.getInputStream()));
  String inputLine;
  StringBuffer answer = new StringBuffer();
  while ((inputLine = in.readLine()) != null) { answer.append(inputLine) ; }
  in.close();
  con.disconnect();
  return answer.toString();
}
```

Classe MyHTTPConnection

Put

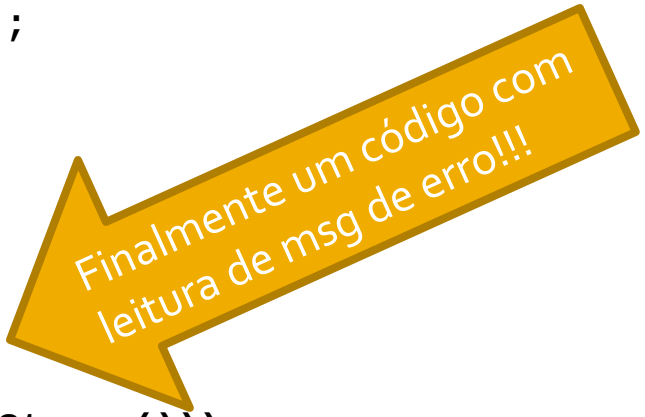
```
public String sendPut(String directory, String data) throws Exception {
  final String url = interSCityURI + "/" + directory;
  final String method = "PUT";
  HttpURLConnection con = (HttpURLConnection) (new URL(url)).openConnection();
  con.setRequestMethod(method);
  con.setRequestProperty("User-agent", USER_AGENT);
  con.setRequestProperty("Content-Type", "application/json");
  con.setDoOutput(true);
  DataOutputStream wr = new DataOutputStream(con.getOutputStream());
  wr.writeBytes(data);
  wr.flush();
  wr.close();
  int responseCode = con.getResponseCode();
  if(responseCode /100 != 2) { throw new HTTPException(responseCode); }
  BufferedReader in = new BufferedReader(new InputStreamReader(con.getInputStream()));
  String inputLine;
  StringBuffer answer = new StringBuffer();
  while ((inputLine = in.readLine()) != null) { answer.append(inputLine) ; }
  in.close();
  con.disconnect();
  return answer.toString();
}
```



Classe MyHTTPConnection

Post

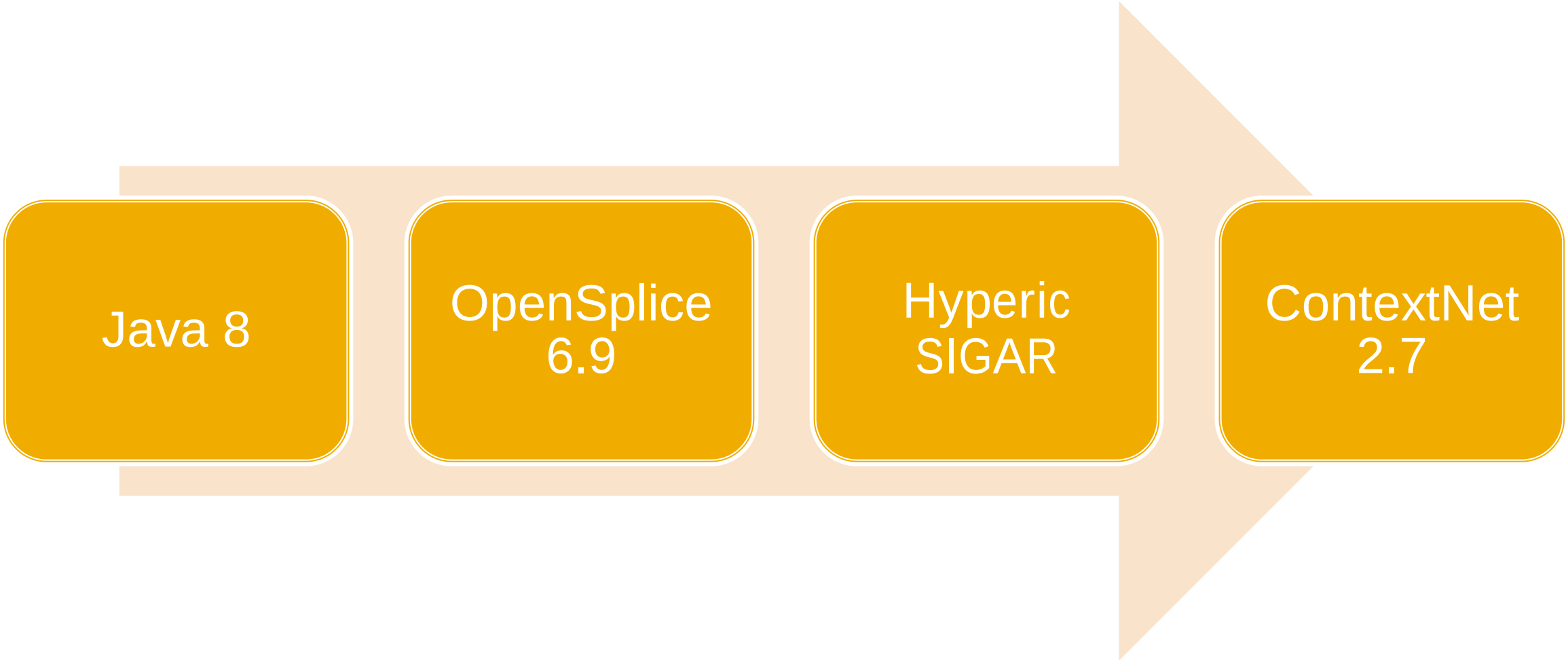
```
public String sendPost(String directory, String jsonString) throws Exception {
  final String url = interSCityURI+"/"+directory;  final String method = "POST";  String inputLine;
  HttpURLConnection con = (HttpURLConnection) (new URL(url)).openConnection();
  con.setRequestMethod(method);
  con.setRequestProperty("User-agent", USER_AGENT);
  con.setRequestProperty("Content-Type", "application/json");
  con.setDoOutput(true);
  DataOutputStream wr = new DataOutputStream(con.getOutputStream());
  wr.writeBytes(jsonString);   wr.flush();   wr.close();
  int responseCode = con.getResponseCode();
  if(responseCode /100 != 2) {
    BufferedReader in = new BufferedReader(new InputStreamReader(con.getErrorStream()));
    StringBuffer answer = new StringBuffer();
    while ((inputLine = in.readLine()) != null) { answer.append(inputLine) ; }
    in.close();   con.disconnect();
    throw new HTTPException("Code: " + responseCode + ": " + answer.toString());
  }
  BufferedReader in = new BufferedReader(new InputStreamReader(con.getInputStream()));
  StringBuffer answer = new StringBuffer();
  while ((inputLine = in.readLine()) != null) { answer.append(inputLine) ; }
  in.close();   con.disconnect();
  return answer.toString();
}
```



Perguntas

Instalação do ContextNet

Instalação do ContextNet



Instalação do ContextNet
Java

- Java
  - No momento que esse documento foi gerado, o ContextNet somente era compatível com Java 8.
  - Para instalar o Java 8, utilize os seguintes comandos:

```
$ sudo apt install openjdk-8-jdk-headless
```

Instalação do ContextNet OpenSplice

- Instale o OpenSplice.
  - Instruções mais detalhadas em `http://www.lac.inf.puc-rio.br/dokuwiki/doku.php?id=installingdds`
- Faça download (ou upload) do arquivo `OpenSpliceDDSV6.4.140407OSS-HDE-x86_64.linux-gcc4.6-glibc2.15-installer.tar.gz`
- Descompacte o arquivo, crie o diretório /opt/OpenSplice e copie o diretório HDE para /opt/OpenSplice

```
$ tar -xvf OpenSpliceDDSV6.4.140407OSS-HDE-x86_64.linux-gcc4.6-glibc2.15-installer.tar.gz
$ sudo mkdir /opt/OpenSplice
$ sudo mv HDE /opt/OpenSplice/
```

- O resultado será:

```
$ ls -l /opt/OpenSplice/
total 4
drwxrwxr-x 3 contextnet contextnet 4096 Apr 14  2014 HDE
```

Instalação do ContextNet OpenSplice

Para facilitar, podemos fazer download a partir do Google Drive ou do GitHub.

OpenSplice v6.9:

- https://drive.google.com/file/d/1iwcjNxzdPK5zoqvTpq7w9mCmiN8b_Lbv/view?usp=sharing
- https://github.com/ADLINK-IST/opensplice/releases

Instalação do ContextNet OpenSplice

Crie um arquivo vazio chamado opensplice.sh e torne o arquivo executável:

- `$ sudo vi /etc/profile.d/opensplice.sh`
- `$ sudo chmod +x /etc/profile.d/opensplice.sh`

Copie o seguinte conteúdo para dentro do arquivo:

- `OSPL_HOME=/opt/OpenSplice/HDE/x86_64.linux`
- `PATH=$OSPL_HOME/bin:$PATH`
- `LD_LIBRARY_PATH=$OSPL_HOME/lib:$LD_LIBRARY_PATH`
- `CPATH=$OSPL_HOME/include:$OSPL_HOME/include/sys:${CPATH:=}`
- `OSPL_URI=file://$OSPL_HOME/etc/config/ospl.xml`
- `export OSPL_HOME PATH LD_LIBRARY_PATH CPATH OSPL_TMPL_PATH OSPL_URI`

Instalação do ContextNet OpenSplice

Edite o arquivo /opt/OpenSplice/HDE/x86_64.linux/etc/config/ospl.xml:

- $ sudo vi /opt/OpenSplice/HDE/x86_64.linux/etc/config/ospl.xml

Substitua o seu conteúdo pelo texto disponível em

- http://wiki.lac.inf.puc-rio.br/doku.php?do=export_code&id=installingdds&codeblock=1

Instalação do ContextNet Hyperic SIGAR

Opcionalmente, para evitar a mensagem de erro ao iniciar o Gateway do ContextNet, instale a biblioteca SIGAR.

Faça upload (ou download) da biblioteca hyperic-sigar-1.6.4.zip ou equivalente para o host a partir de

- `https://sourceforge.net/projects/sigar/files/`

Descompacte, mova e muda as permissões com os comandos a seguir (pode ser necessário instalar o unzip):

- `$ unzip hyperic-sigar-1.6.4.zip`
- `$ sudo mv hyperic-sigar-1.6.4/sigar-bin/lib/*.so /opt/OpenSplice/HDE/x86_64.linux/lib/`
- `$ sudo chmod 775 /opt/OpenSplice/HDE/x86_64.linux/lib/libsigar-*`

Instalação do ContextNet (Finalmente!!!)

Download do ContextNet:

- `http://wiki.lac.inf.puc-rio.br/doku.php?id=download`

Javadoc

- `http://wiki.lac.inf.puc-rio.br/api/`

Crie o diretório /opt/ContextNet e copie o ContextNet (a ClientLib e a UDILib) para lá.

- `$ sudo mkdir /opt/ContextNet`
- `$ sudo cp contextnet-2.7.jar /opt/ContextNet`

Perguntas?

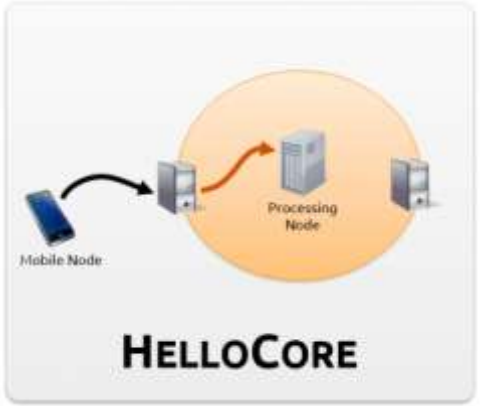
Desenvolvendo primeiras aplicações

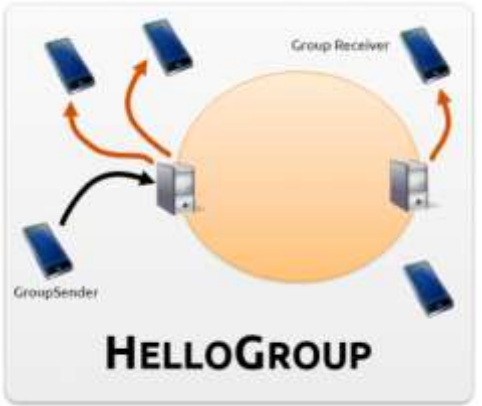
ContextNet

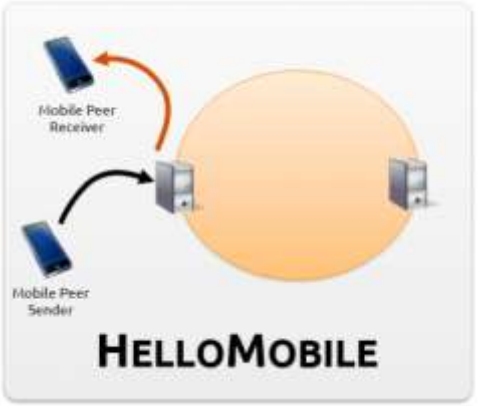
Desenvolvendo aplicações com o ContextNet

- Tutorial
  - HelloCore
  - HelloMobile
  - HelloGroup
  - HelloAndroid

- Tutorial em `http://wiki.lac.inf.puc-rio.br/doku.php?id=tutorial`







Desenvolvendo aplicações com o ContextNet

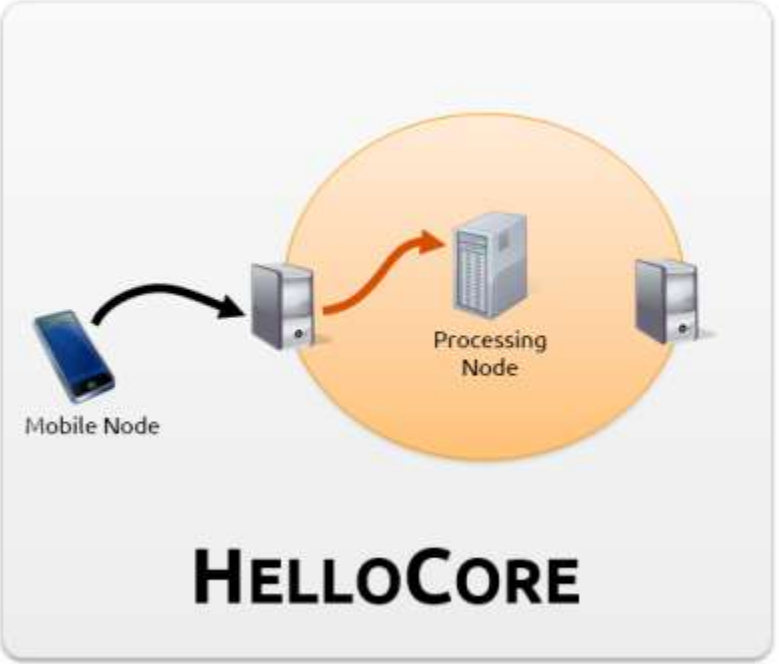
- Tutorial
  - HelloCore
  - HelloMobile
  - HelloGroup
  - HelloAndroid

Dois componentes:

- Cliente móvel
- Nó fixo

Objetivo

- Enviar uma mensagem do cliente móvel para o nó fixo



Desenvolvendo aplicações com o ContextNet

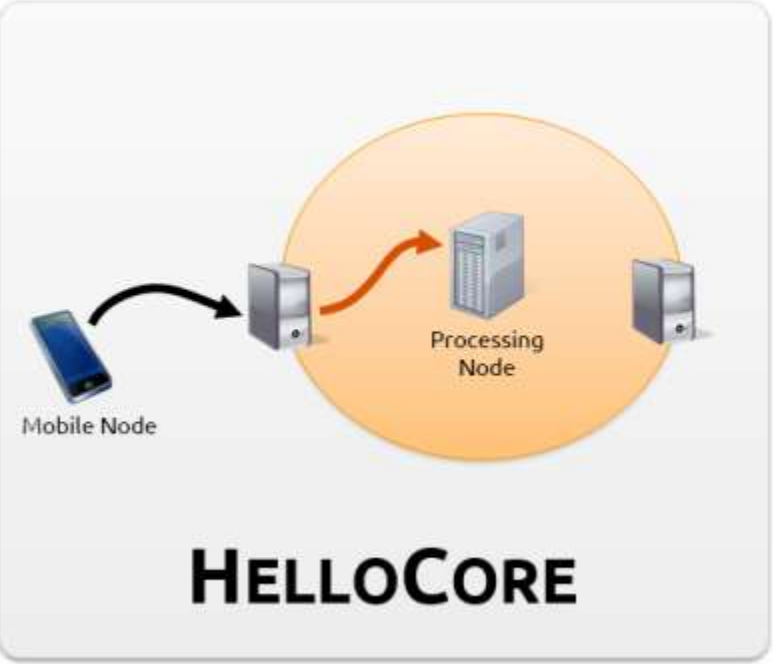
- Tutorial
  - HelloCore
  - HelloMobile
  - HelloGroup
  - HelloAndroid

Dois componentes:

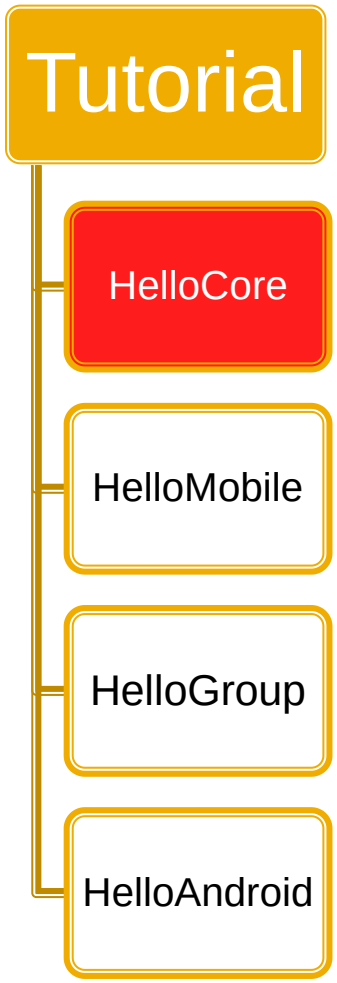
- Cliente móvel
- Nó fixo

Objetivo

- Enviar uma mensagem do cliente móvel para o nó fixo

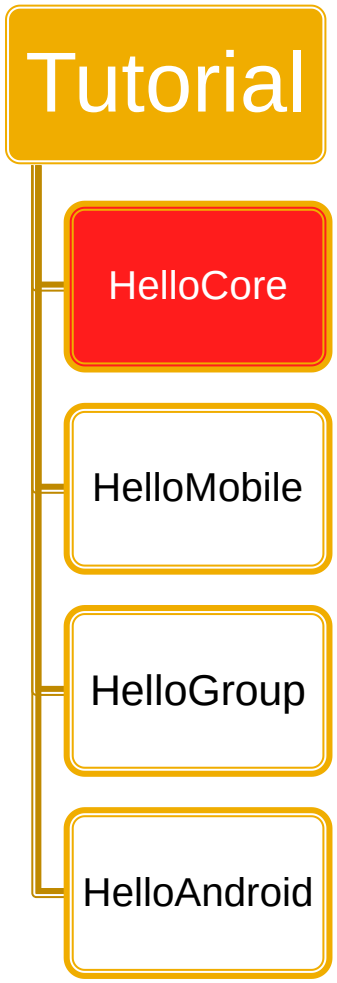


Desenvolvendo aplicações com o ContextNet
HelloCoreClient (1/4) – pacotes



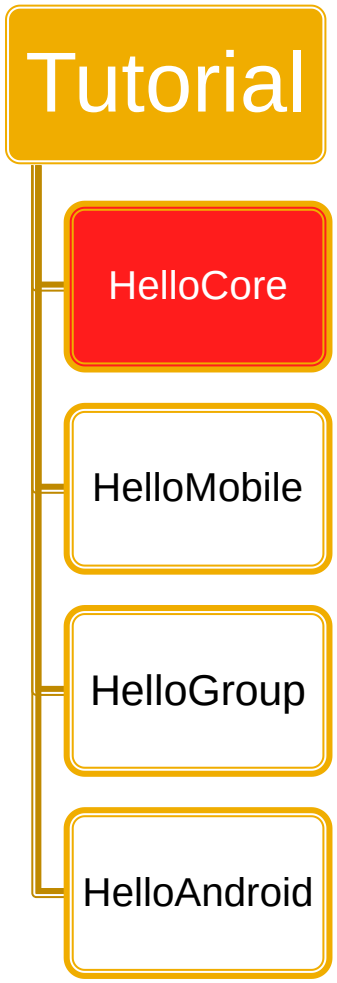

```
package br.com.meslin.lac;

import java.io.IOException;
import java.net.InetSocketAddress;
import java.net.SocketAddress;
import java.util.List;
import java.util.logging.Level;
import java.util.logging.Logger;

import lac.cnclib.net.NodeConnection;
import lac.cnclib.net.NodeConnectionListener;
import lac.cnclib.net.mrudp.MrUdpNodeConnection;
import lac.cnclib.sddl.message.ApplicationMessage;
import lac.cnclib.sddl.message.Message;
```

Desenvolvendo aplicações com o ContextNet
HelloCoreClient (2/4) – construtor e main




```
public class HelloCoreClient implements NodeConnectionListener {
  private static StringgatewayIP   = "127.0.0.1";
  private static intgatewayPort = 5500;
  private MrUdpNodeConnectionconnection;

  public static void main(String[] args) {
      Logger.getLogger("").setLevel(Level.OFF);
      new HelloCoreClient();
  }

  public HelloCoreClient() {
      InetSocketAddress address = new InetSocketAddress(gatewayIP, gatewayPort);
      try {
          connection = new MrUdpNodeConnection();
          connection.addNodeConnectionListener(this);
          connection.connect(address);
      } catch (IOException e) {
          e.printStackTrace();
      }
  }
```

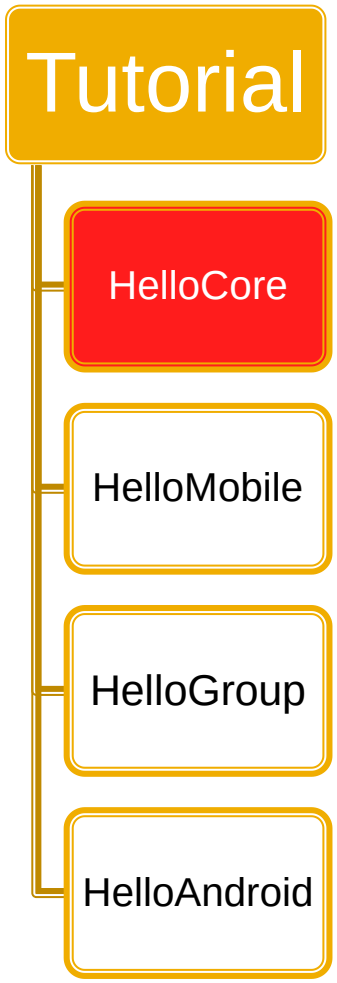
Desenvolvendo aplicações com o ContextNet
HelloCoreClient (3/4) – envio da mensagem




```
@Override
 public void connected(NodeConnection remoteCon) {
     ApplicationMessage message = new ApplicationMessage();
     String serializableContent = "Hello World";
     message.setContentObject(serializableContent);

     try {
         connection.sendMessage(message);
     } catch (IOException e) {
         e.printStackTrace();
     }
 }

 @Override
 public void newMessageReceived(NodeConnection remoteCon, Message message) {
     System.out.println(message.getContentObject());
 }
```

Envio de mensagem para o núcleo (Processing Node)

Desenvolvendo aplicações com o ContextNet
HelloCoreClient (4/4) – métdos auxiliares




```
 // other methods

  @Override
  public void reconnected(NodeConnection remoteCon, SocketAddress endPoint, boolean wasHandover,
boolean wasMandatory) {}

  @Override
  public void disconnected(NodeConnection remoteCon) {}

  @Override
  public void unsentMessages(NodeConnection remoteCon, List<Message> unsentMessages) {}

  @Override
  public void internalException(NodeConnection remoteCon, Exception e) {}
}
```

Desenvolvendo aplicações com o ContextNet

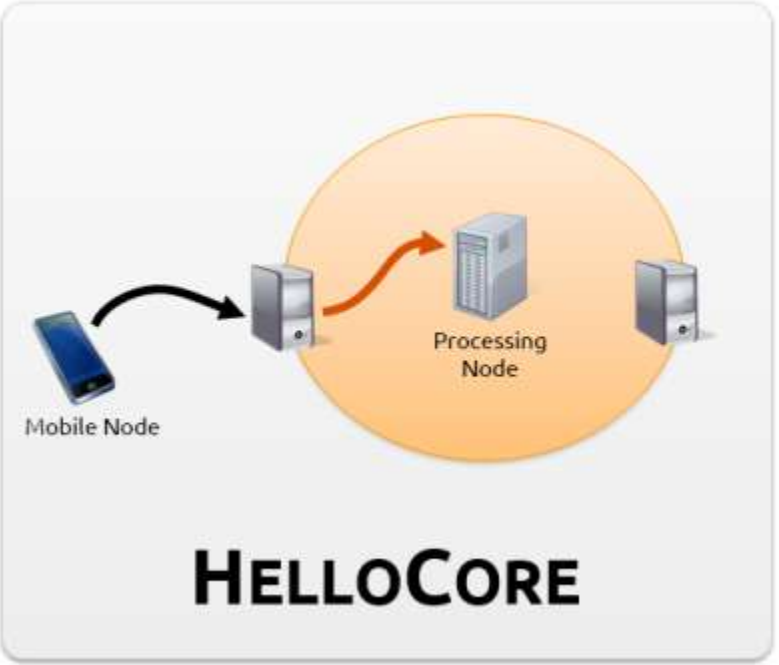
- Tutorial
  - HelloCore
  - HelloMobile
  - HelloGroup
  - HelloAndroid

Dois componentes:

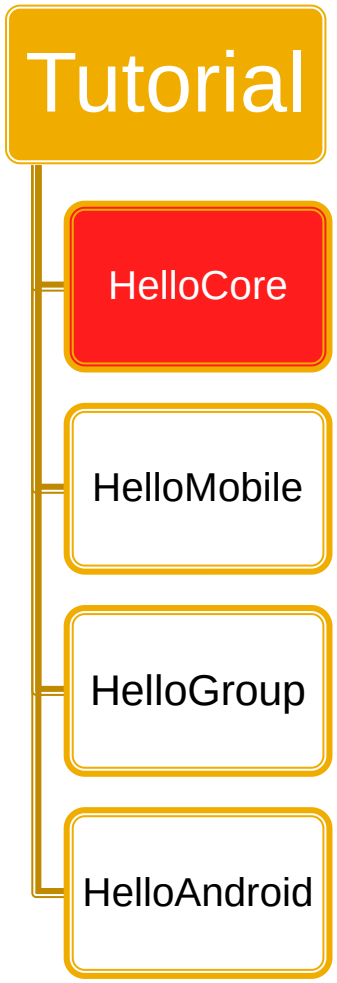
- Cliente móvel
- Nó fixo

Objetivo

- Enviar uma mensagem do cliente móvel para o nó fixo

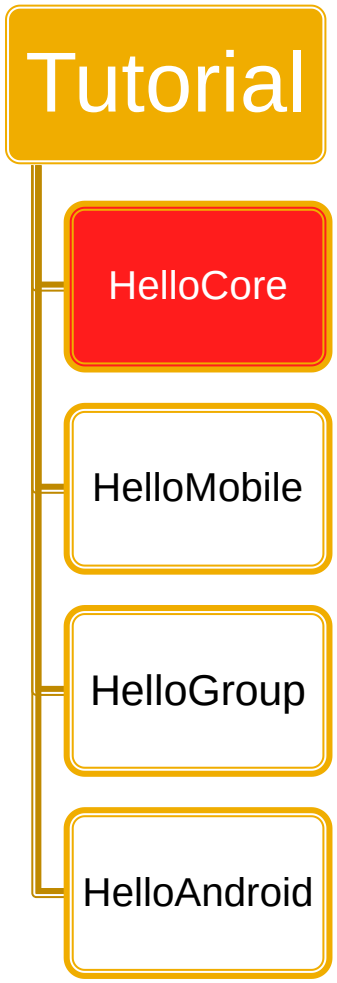


Desenvolvendo aplicações com o ContextNet
HelloCoreServer (1/5) – pacotes



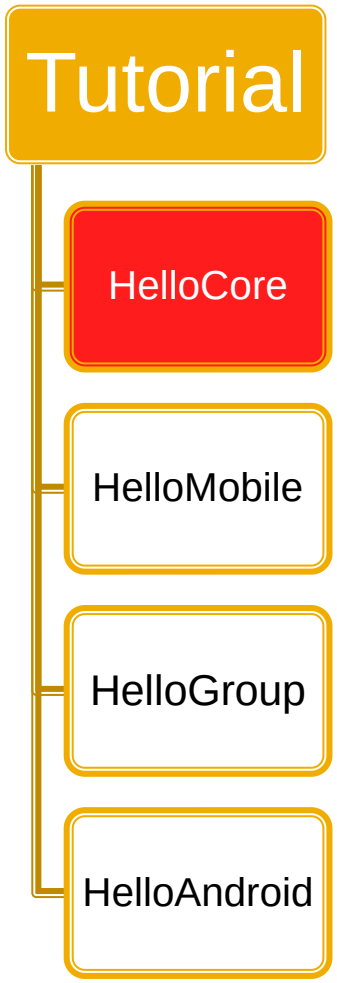

```
package br.com.meslin.lac;

import java.util.logging.Level;
import java.util.logging.Logger;

import lac.cnclib.sddl.message.ApplicationMessage;
import lac.cnclib.sddl.serialization.Serialization;
import lac.cnet.sddl.objects.ApplicationObject;
import lac.cnet.sddl.objects.Message;
import lac.cnet.sddl.objects.PrivateMessage;
import lac.cnet.sddl.udi.core.SddlLayer;
import lac.cnet.sddl.udi.core.UniversalDDSLayerFactory;
import lac.cnet.sddl.udi.core.listener.UDIDataReaderListener;
```

Desenvolvendo aplicações com o ContextNet
HelloCoreServer (2/5) – main



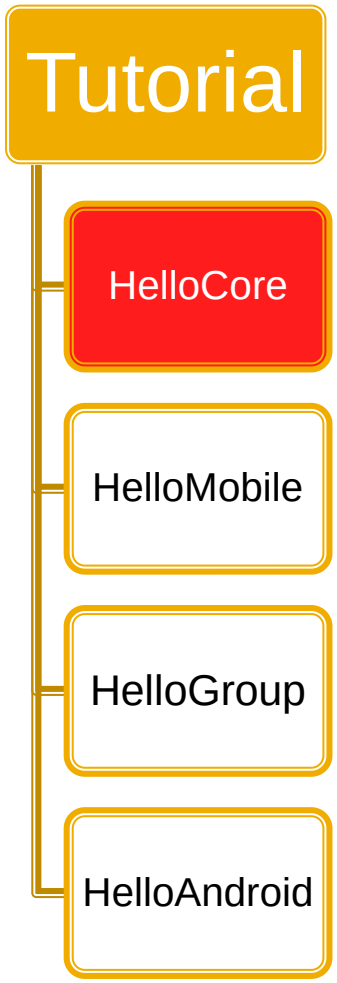

```
public class HelloCoreServer implements UDIDataReaderListener<ApplicationObject> {
  SddlLayer  core;
  int        counter;

  public static void main(String[] args) {
    Logger.getLogger("").setLevel(Level.OFF);

    new HelloCoreServer();
  }
```

Desenvolvendo aplicações com o ContextNet
HelloCoreServer (3/5) – registro



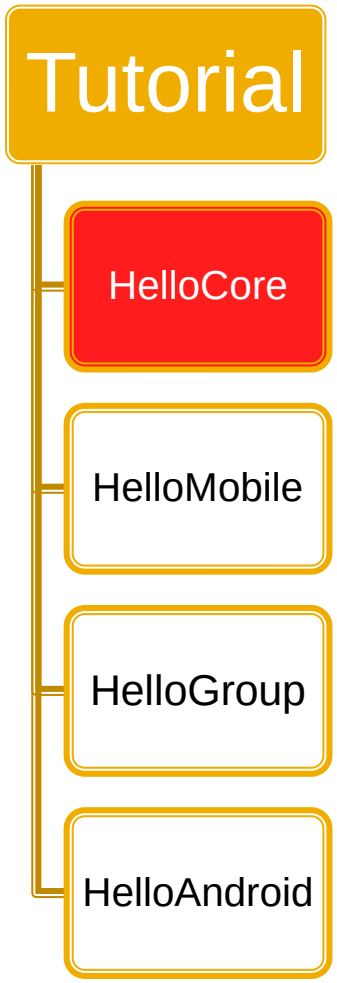

```
public class HelloCoreServer implements UDIDataReaderListener<ApplicationObject> {
...
 public HelloCoreServer() {
    core = UniversalDDSLayerFactory.getInstance();
    core.createParticipant(UniversalDDSLayerFactory.CNET_DOMAIN);

    core.createPublisher();
    core.createSubscriber();

    Object receiveMessageTopic = core.createTopic(Message.class, Message.class.getSimpleName());
    core.createDataReader(this, receiveMessageTopic);

    Object toMobileNodeTopic = core.createTopic(
        PrivateMessage.class,
        PrivateMessage.class.getSimpleName());
    core.createDataWriter(toMobileNodeTopic);
 ...
```

Desenvolvendo aplicações com o ContextNet
HelloCoreServer (4/5) – finalizações




```
public class HelloCoreServer implements UDIDataReaderListener<ApplicationObject> {
...
 public HelloCoreServer() {
 ...
 counter = 0;
    System.out.println("=== Server Started (Listening) ===");
    synchronized (this) {
      try {
        this.wait();
      } catch (InterruptedException e) {
        e.printStackTrace();
      }
    }
  }
 ...
}
```

Desenvolvendo aplicações com o ContextNet
HelloCoreServer (5/5) – recebendo dados




```
public class HelloCoreServer implements UDIDataReaderListener<ApplicationObject> {
...
 @Override
  public void onNewData(ApplicationObject topicSample) {
    Message message = (Message) topicSample;
    System.out.println(Serialization.fromJavaByteStream(message.getContent()));

    PrivateMessage privateMessage = new PrivateMessage();
    privateMessage.setGatewayId(message.getGatewayId());
    privateMessage.setNodeId(message.getSenderId());

    synchronized (core) {
      counter++;
    }

    ApplicationMessage appMsg = new ApplicationMessage();
    appMsg.setContentObject(counter);
    privateMessage.setMessage(Serialization.toProtocolMessage(appMsg));

    core.writeTopic(PrivateMessage.class.getSimpleName(), privateMessage);
  }
}
```

Alerta de spoiler: cenas dos próximos capítulos
Decodificando tipos de mensagens

```
public void onNewData(ApplicationObject topicSample) {
  if(topicSample instanceof Message) {
    Message msg = (Message) topicSample;
    String content = new String(msg.getContent());
    JSONParser parser = new JSONParser();
    try {
      JSONObject object = (JSONObject) parser.parse(content);
      String tag = (String) object.get("tag");
      switch (tag) {
        case "SensorData":
          handleSensorData(object);
        break;
        case "EventData":
          final String label = (String) object.get("label");
          final String data = (String) object.get("data");
          handleEvent(label, data);
        break;
        case "ReplyData":
        case "ErrorData":
          handleMessage(tag, object);
        break;
      }
    } catch (Exception e) {
      System.out.println("Mensagem sem tag (pode ter sido um keep-alive): " + e.getMessage());
    }
  }
}
```

Desenvolvendo aplicações com o ContextNet Executando

- Tutorial
  - HelloCore
  - HelloMobile
  - HelloGroup
  - HelloAndroid

Precisaremos de três "máquinas" (consoles):

Gateway

- `$ java -jar /opt/ContextNet/contextnet-2.7.jar 127.0.0.1 5500 OpenSplice`

Nó fixo

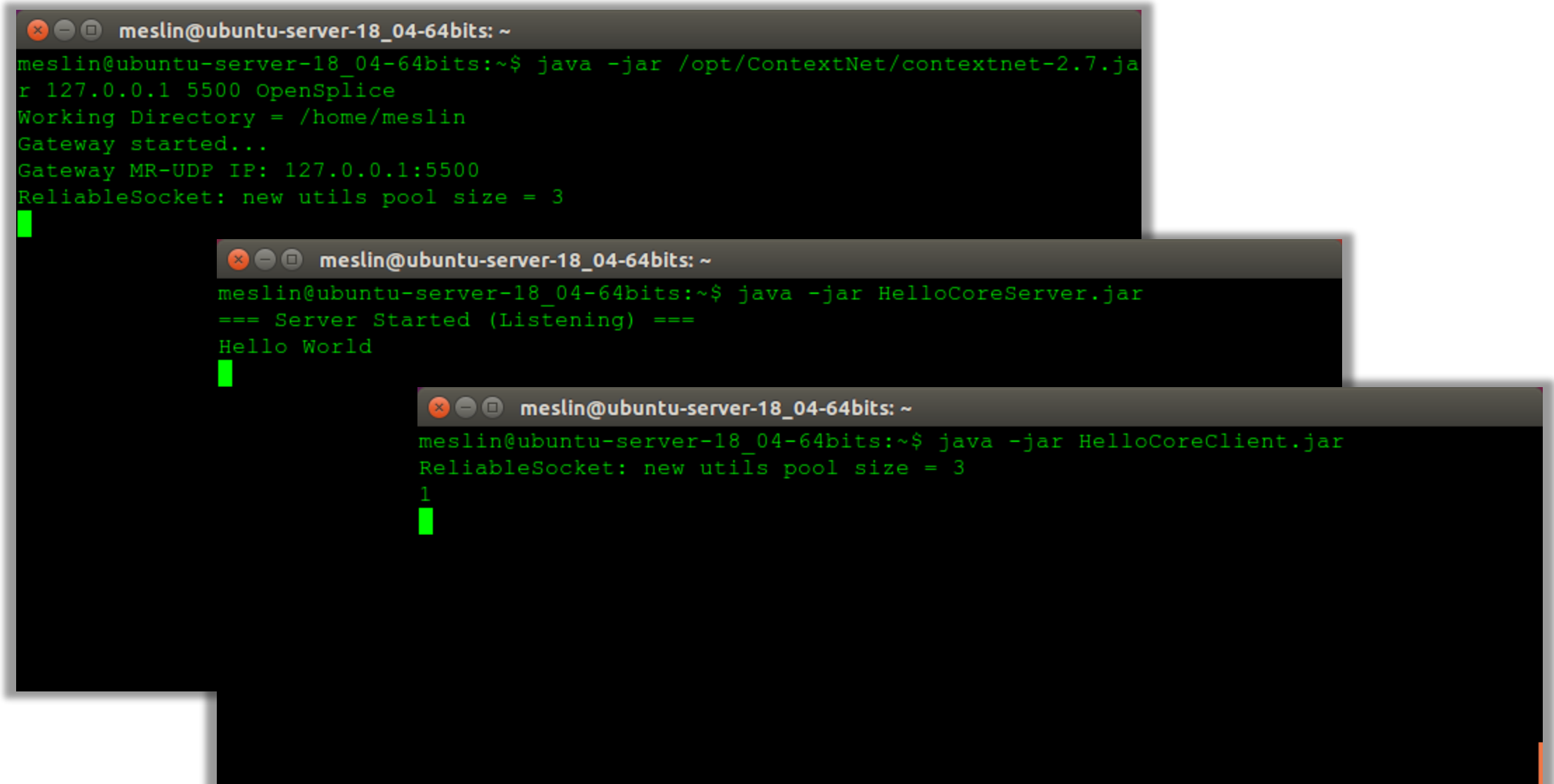
- `$ java -jar HelloCoreServer.jar`

Cliente móvel

- `$ java -jar HelloCoreClient.jar`

Desenvolvendo aplicações com o ContextNet
Resultado

- Tutorial
  - HelloCore
  - HelloMobile
  - HelloGroup
  - HelloAndroid



Desenvolvendo aplicações com o ContextNet

- Tutorial
  - HelloCore
  - HelloMobile
  - HelloGroup
  - HelloAndroid

Dois componentes:

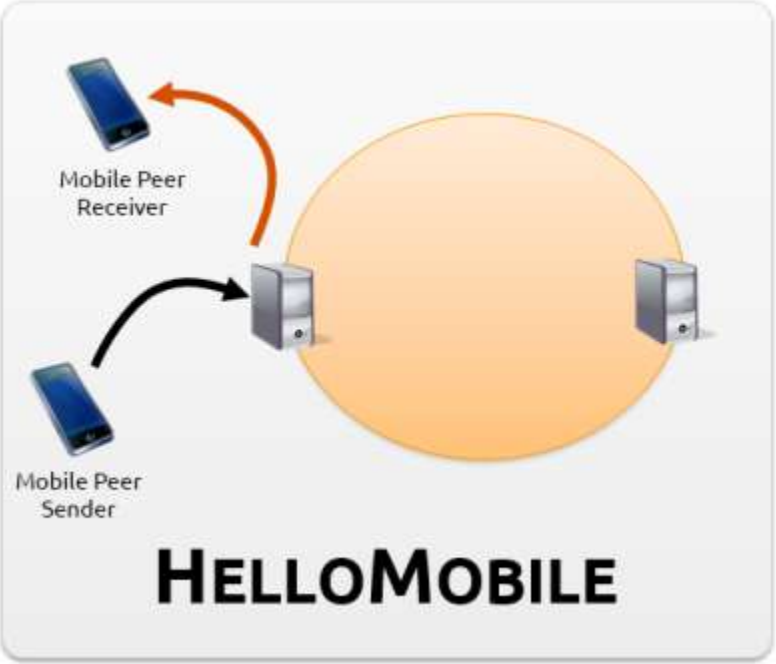
- Cliente móvel transmissor
- Cliente móvel receptor

Classe auxiliar

- CustomData

Objetivo

- Enviar uma mensagem de um cliente móvel para outro cliente móvel



Desenvolvendo aplicações com o ContextNet

Tutorial

- HelloCore
- HelloMobile
- HelloGroup
- HelloAndroid

Dois componentes:

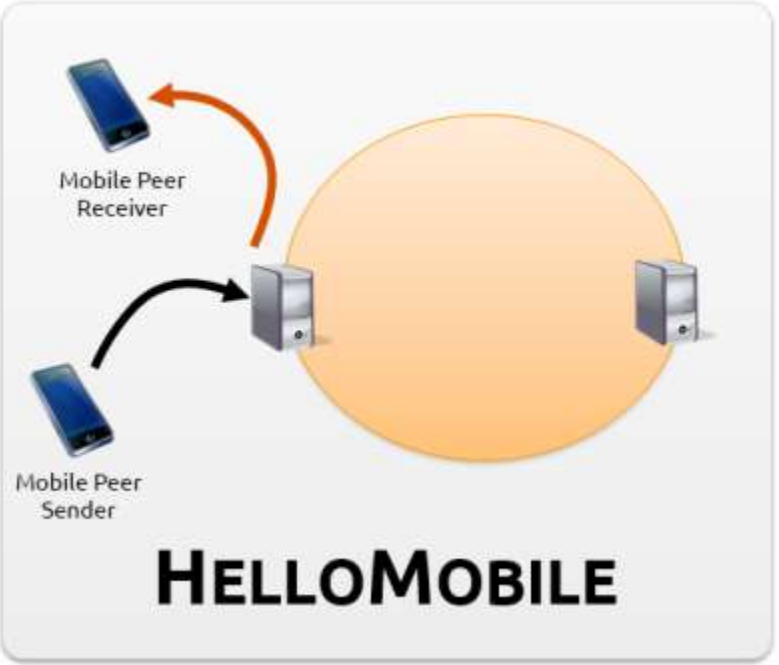
- Cliente móvel transmissor
- Cliente móvel receptor

Classe auxiliar

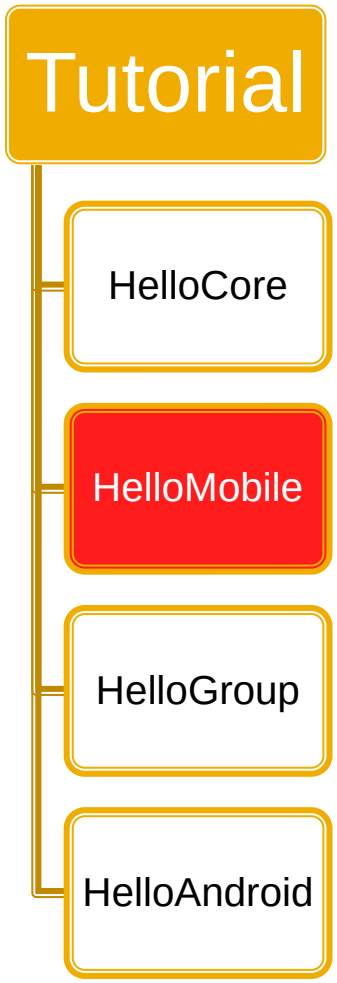
- CustomData

Objetivo

- Enviar uma mensagem de um cliente móvel para outro cliente móvel

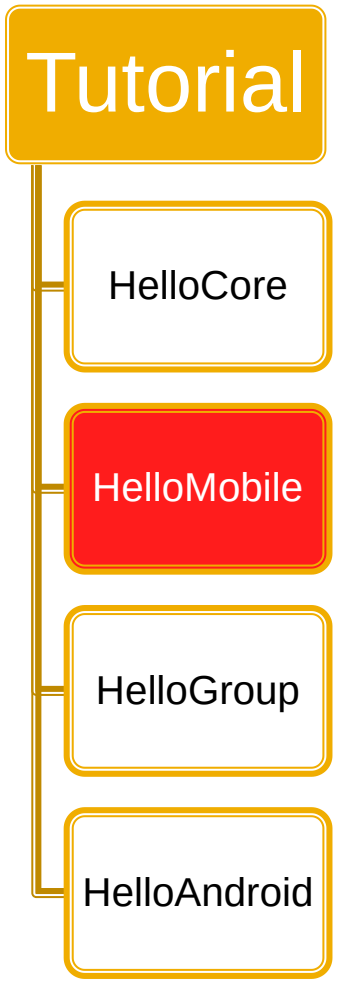


Desenvolvendo aplicações com o ContextNet
CustomData (1/2) – pacotes e construtor




```
package br.com.meslin.lac;

import java.io.Serializable;
import javax.swing.ImageIcon;

public class CustomData implements Serializable {
  private static final long serialVersionUID = 6093226637618022646L;
  private String caption;
  private ImageIcon icon;

  public CustomData() {
  }

  public CustomData(String caption, String icon) {
    this.caption = caption;

    this.icon = new ImageIcon(Thread.currentThread().getContextClassLoader().getResource(icon));
  }
```

Desenvolvendo aplicações com o ContextNet
CustomData (2/2) – métodos de acesso




```
    public String getCaption() { return caption;  }

    public ImageIcon getIcon() { return icon; }

    public void setCaption(String caption) { this.caption = caption; }

    public void setIcon(ImageIcon icon) { this.icon = icon; }

    @Override
    public String toString() {
      String firstPart  = "Caption: " + caption;
      String secondPart = "\nIcon Height: " + icon.getIconHeight();
      String thirdPart  = "\nIcon Width: " + icon.getIconWidth();

      return firstPart + secondPart + thirdPart;
    }
}
```

Desenvolvendo aplicações com o ContextNet

- Tutorial
  - HelloCore
  - HelloMobile
  - HelloGroup
  - HelloAndroid

Dois componentes:

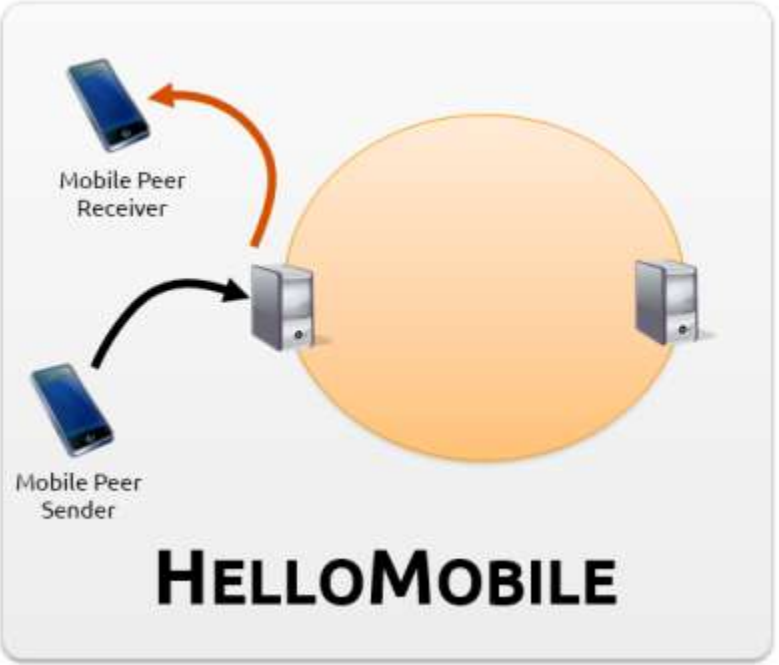
- Cliente móvel transmissor
- Cliente móvel receptor

Classe auxiliar

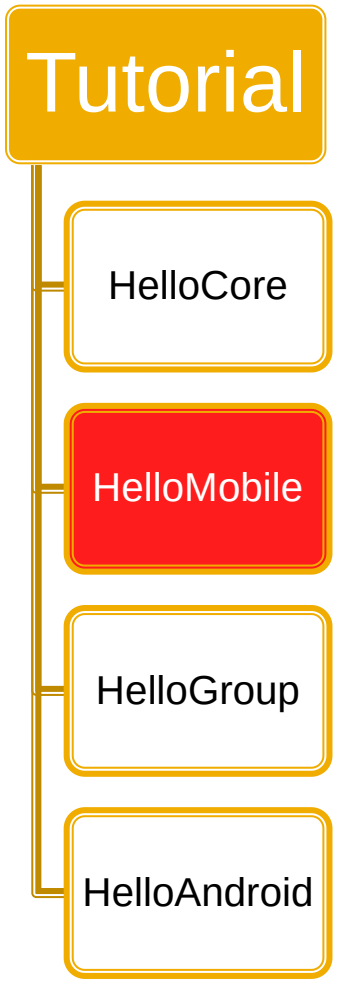
- CustomData

Objetivo

- Enviar uma mensagem de um cliente móvel para outro cliente móvel

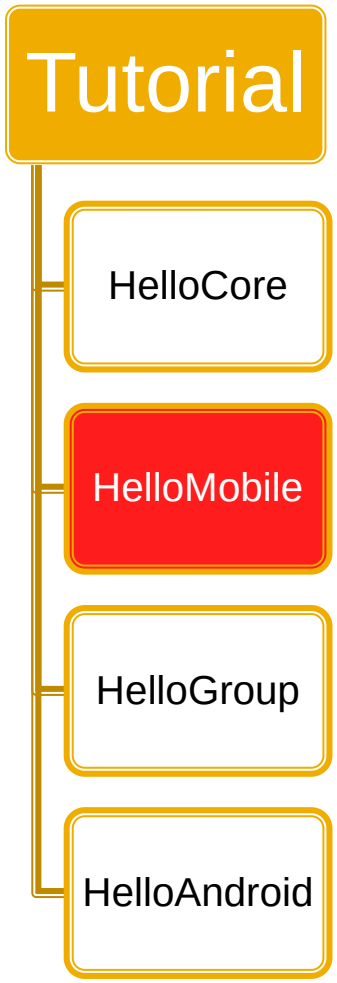


Desenvolvendo aplicações com o ContextNet
HelloMobileSender (1/6) – pacotes



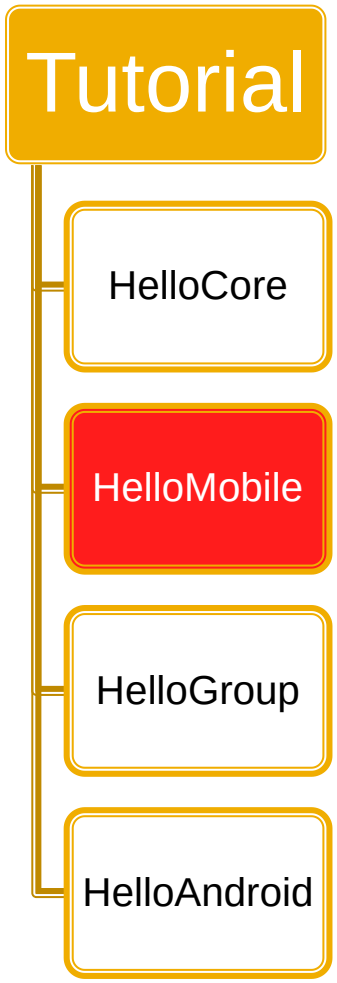

```
package br.com.meslin.lac;

import java.io.IOException;
import java.net.InetSocketAddress;
import java.net.SocketAddress;
import java.util.List;
import java.util.UUID;
import java.util.logging.Level;
import java.util.logging.Logger;

import lac.cnclib.net.NodeConnection;
import lac.cnclib.net.NodeConnectionListener;
import lac.cnclib.net.mrudp.MrUdpNodeConnection;
import lac.cnclib.sddl.message.ApplicationMessage;
import lac.cnclib.sddl.message.Message;
```

Desenvolvendo aplicações com o ContextNet
HelloMobileSender (2/6) – construtor



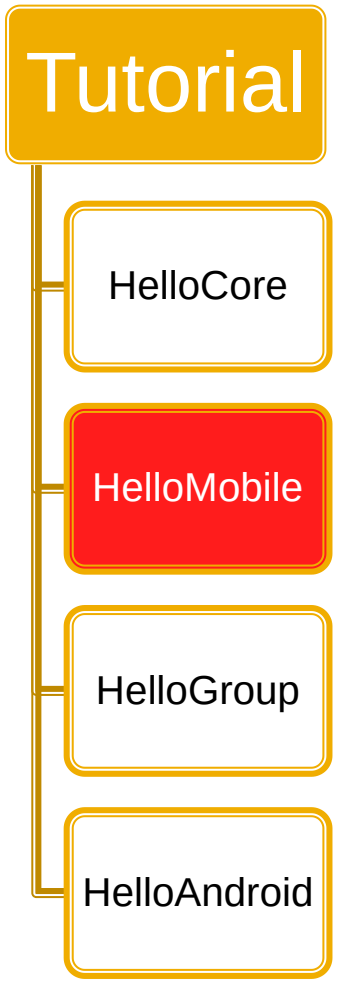

```
public class HelloMobileSender implements NodeConnectionListener {
  private static String gatewayIP = "127.0.0.1";
  private static int gatewayPort = 5500;
  private MrUdpNodeConnection connection;
  private UUID myUUID;

  public HelloMobileSender() {
    myUUID = UUID.fromString("bb103877-8335-444a-be5f-db8d916f6754");
    InetSocketAddress address = new InetSocketAddress(gatewayIP, gatewayPort);
    try {
      connection = new MrUdpNodeConnection(myUUID);
      connection.addNodeConnectionListener(this);
      connection.connect(address);
    } catch (IOException e) {
      e.printStackTrace();
    }
  }
```

Desenvolvendo aplicações com o ContextNet
HelloMobileSender (3/6) – main



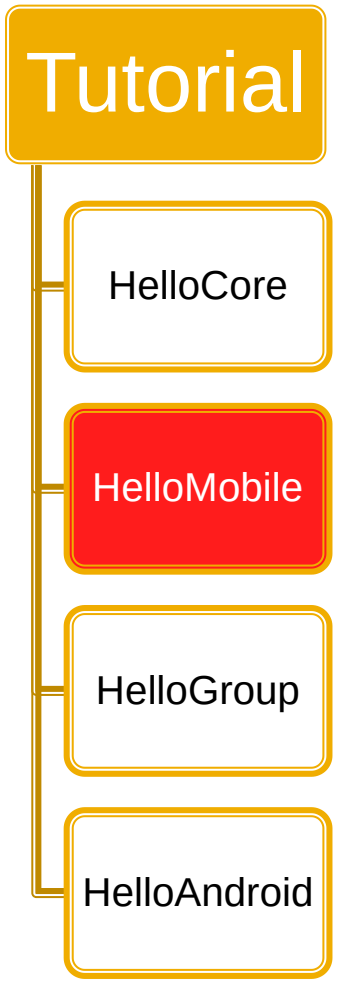

```
public class HelloMobileSender implements NodeConnectionListener {
…
  public static void main(String[] args) {
    Logger.getLogger("").setLevel(Level.ALL);

    HelloMobileSender sender = new HelloMobileSender();
    sender.sendPicture("Rio de Janeiro", "/home/meslin/rio.jpg");
  }
 ...
}
```

Desenvolvendo aplicações com o ContextNet
HelloMobileSender (4/6) – método auxiliar



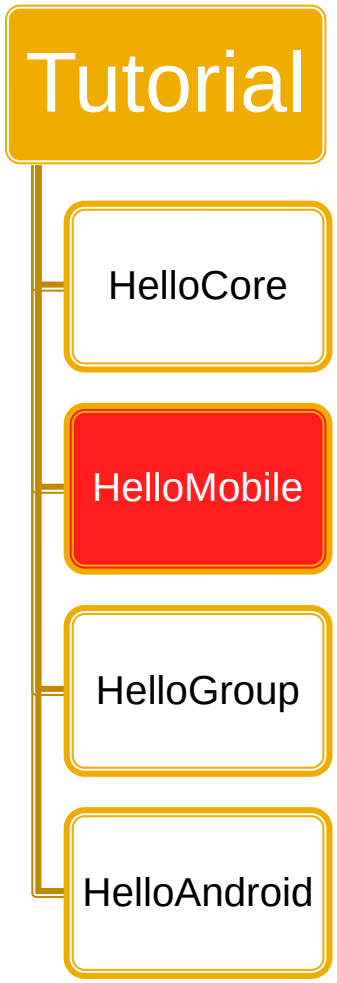

```
public class HelloMobileSender implements NodeConnectionListener {
…
 public void sendPicture(String caption, String imageName) {
      CustomData serializableContent = new CustomData(caption, imageName);
      ApplicationMessage message = new ApplicationMessage();
      message.setContentObject(serializableContent);
      message.setRecipientID(UUID.fromString("788b2b22-baa6-4c61-b1bb-01cff1f5f878"));

      try {
        System.out.println("Sending image + caption");
        connection.sendMessage(message);
      } catch (IOException e) {
        e.printStackTrace();
      }
   }
...
}
```

Desenvolvendo aplicações com o ContextNet
HelloMobileSender (5/6) – mensagens




```
public class HelloMobileSender implements NodeConnectionListener {
…
 public void connected(NodeConnection remoteCon) {
    ApplicationMessage message = new ApplicationMessage();
    message.setContentObject("Registering");

    try {
      connection.sendMessage(message);
    } catch (IOException e) {
      e.printStackTrace();
    }
  }

  public void newMessageReceived(NodeConnection remoteCon, Message message) {
    System.out.println("Sender received the message!!");
    System.out.println(message.getContentObject());
  }
 ...
}
```

Desenvolvendo aplicações com o ContextNet
HelloMobileSender (6/6) – outros métodos




```
public class HelloMobileSender implements NodeConnectionListener {
…
 public void reconnected(
    NodeConnection remoteCon,
    SocketAddress endPoint,
    boolean wasHandover,
    boolean wasMandatory) {}
  public void disconnected(NodeConnection remoteCon) {}
  public void unsentMessages(NodeConnection remoteCon, List<Message> unsentMessages) {}
  public void internalException(NodeConnection remoteCon, Exception e) {}
}
...
}
```

Desenvolvendo aplicações com o ContextNet

- Tutorial
  - HelloCore
  - HelloMobile
  - HelloGroup
  - HelloAndroid

Dois componentes:

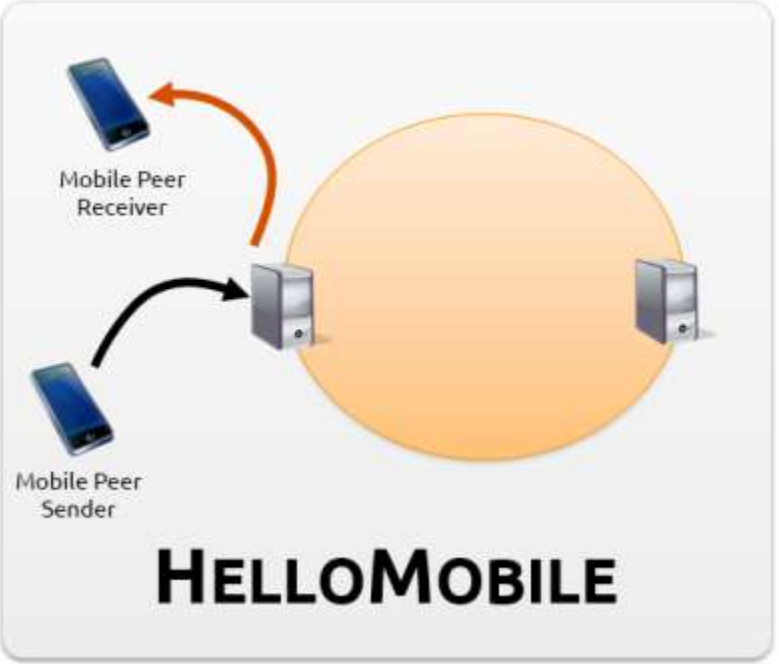
- Cliente móvel transmissor
- Cliente móvel receptor

Classe auxiliar

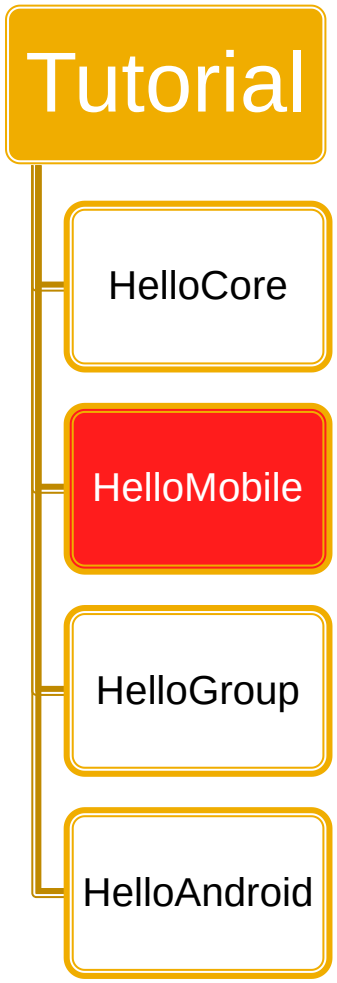
- CustomData

Objetivo

- Enviar uma mensagem de um cliente móvel para outro cliente móvel

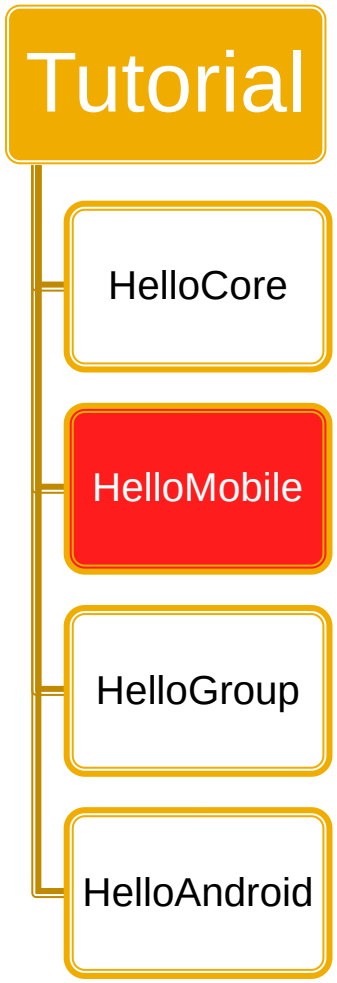


Desenvolvendo aplicações com o ContextNet
HelloMobileReceiver (1/4) – pacotes



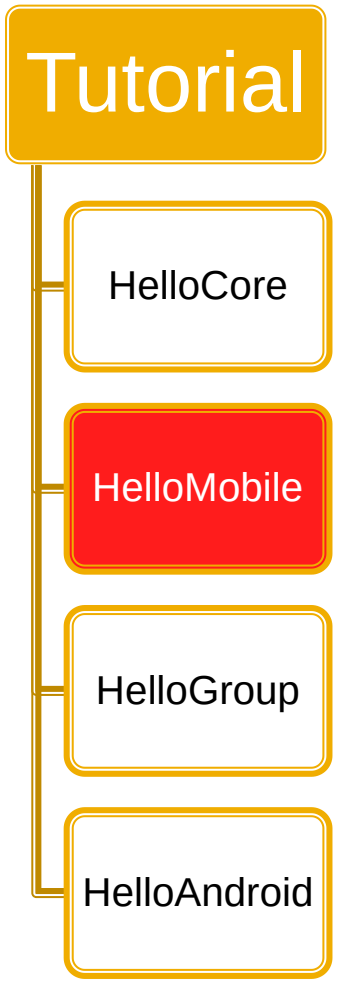

```
package br.com.meslin.lac;

import java.io.IOException;
import java.net.InetSocketAddress;
import java.net.SocketAddress;
import java.util.List;
import java.util.UUID;
import java.util.logging.Level;
import java.util.logging.Logger;

import lac.cnclib.net.NodeConnection;
import lac.cnclib.net.NodeConnectionListener;
import lac.cnclib.net.mrudp.MrUdpNodeConnection;
import lac.cnclib.sddl.message.ApplicationMessage;
import lac.cnclib.sddl.message.Message;
```

Desenvolvendo aplicações com o ContextNet
HelloMobileReceiver (2/4) – construtor e main



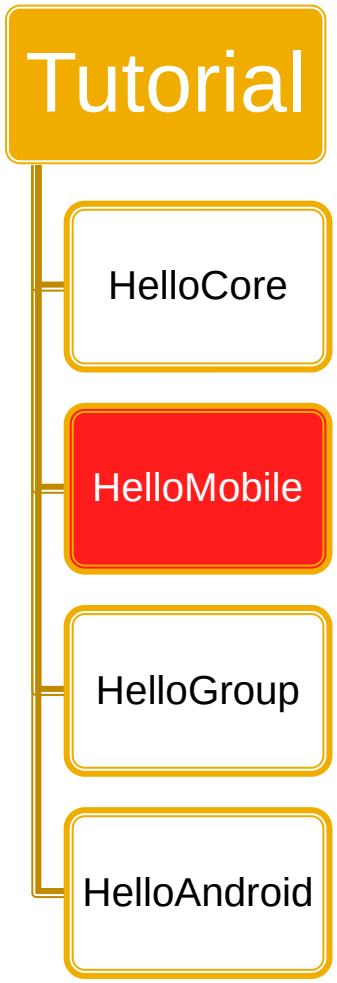

```
public class HelloMobileReceiver implements NodeConnectionListener {
  private static String       gatewayIP  = "127.0.0.1";
  private static int          gatewayPort  = 5500;
  private MrUdpNodeConnection connection;
  private UUID                myUUID;

  public HelloMobileReceiver() {
    myUUID = UUID.fromString("788b2b22-baa6-4c61-b1bb-01cff1f5f878");
    InetSocketAddress address = new InetSocketAddress(gatewayIP, gatewayPort);
    try {
      connection = new MrUdpNodeConnection(myUUID);
      connection.addNodeConnectionListener(this);
      connection.connect(address);
    } catch (IOException e) {
      e.printStackTrace();
    }
  }

  public static void main(String[] args) {
    Logger.getLogger("").setLevel(Level.INFO);
    new HelloMobileReceiver();
  }
```

Desenvolvendo aplicações com o ContextNet
HelloMobileReceiver (3/4) – métodos de conexão




```
public class HelloMobileReceiver implements NodeConnectionListener {
…
  public void connected(NodeConnection remoteCon) {
    ApplicationMessage message = new ApplicationMessage();
    message.setContentObject("Registering");

    try {
      connection.sendMessage(message);
    } catch (IOException e) {
      e.printStackTrace();
    }
  }

  public void reconnected(NodeConnection remoteCon, SocketAddress endPoint, boolean wasHandover,
boolean wasMandatory) {}
  public void disconnected(NodeConnection remoteCon) {}
  public void unsentMessages(NodeConnection remoteCon, List<Message> unsentMessages) {}
  public void internalException(NodeConnection remoteCon, Exception e) {}
```

Desenvolvendo aplicações com o ContextNet
HelloMobileReceiver (4/4) – recebe mensagem




```
public class HelloMobileReceiver implements NodeConnectionListener {
…
  public void newMessageReceived(NodeConnection remoteCon, Message message) {
    System.out.println("Receiver Node!");
    System.out.println("Message: " + message.getContentObject());

    ApplicationMessage appMessage = new ApplicationMessage();
    appMessage.setContentObject("Rio de Janeiro continua lindo!");
    appMessage.setRecipientID(message.getSenderID());

    try {
      connection.sendMessage(appMessage);
    } catch (IOException e) {
      e.printStackTrace();
    }
  }
}
```

Desenvolvendo aplicações com o ContextNet Executando

- Tutorial
  - HelloCore
  - HelloMobile
  - HelloGroup
  - HelloAndroid

Precisaremos de três "máquinas" (consoles):

Gateway

- `$ java -jar /opt/ContextNet/contextnet-2.7.jar 127.0.0.1 5500 OpenSplice`

Cliente quer irá receber a mensagem

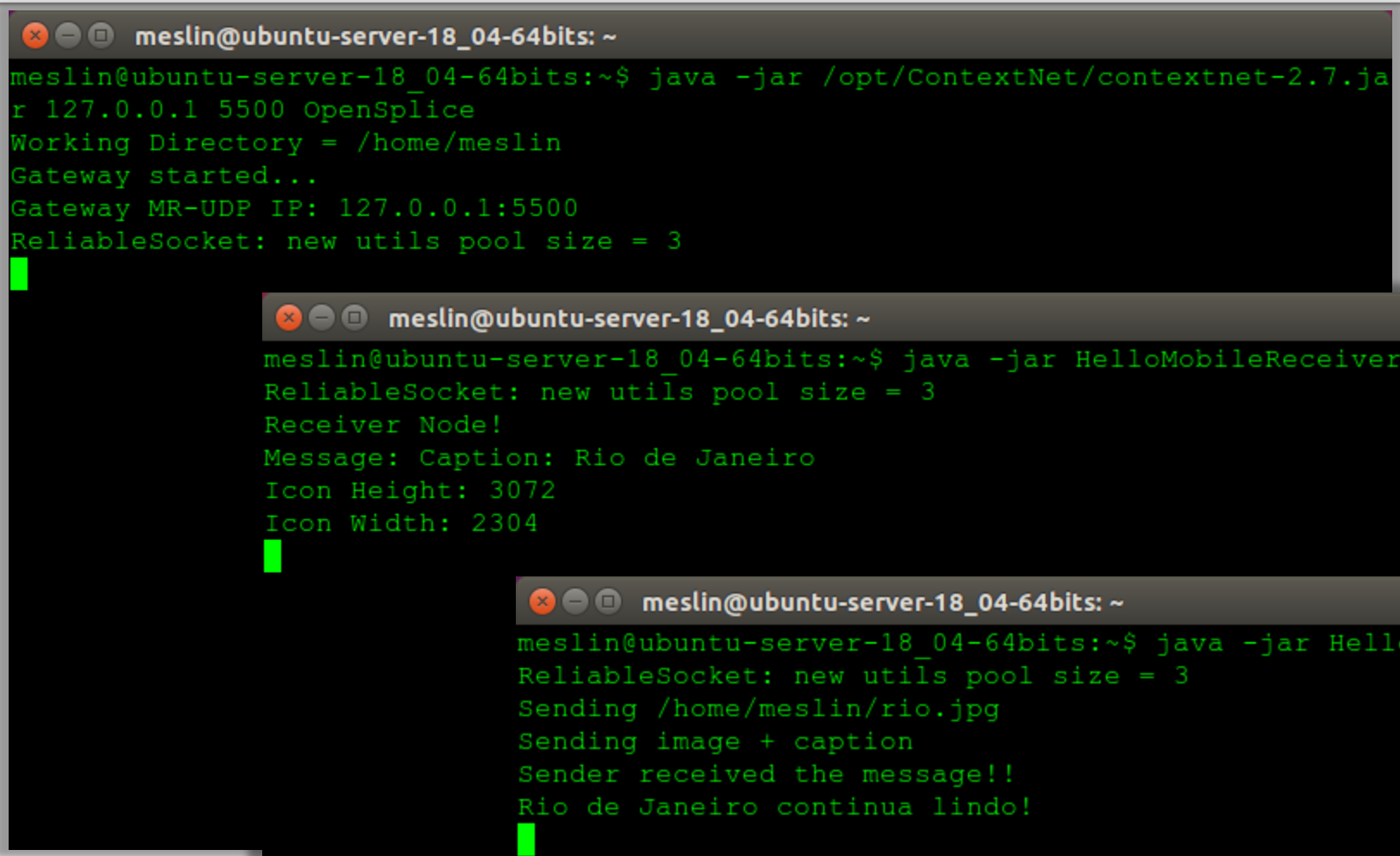
- `$ java -jar HelloMobileReceiver.jar`

Cliente que irá enviar a mensagem

- `$ java -jar HelloMobileSender.jar`

Desenvolvendo aplicações com o ContextNet
Resultado

- Tutorial
  - HelloCore
  - HelloMobile
  - HelloGroup
  - HelloAndroid



Desenvolvendo aplicações com o ContextNet

Tutorial

- HelloCore
- HelloMobile
- **HelloGroup**
- HelloAndroid

Dois componentes:

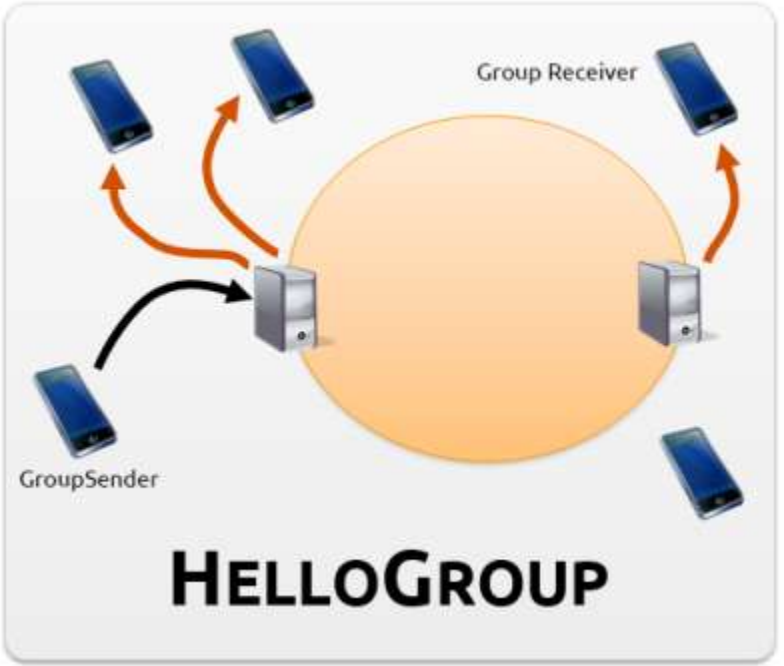
- Cliente móvel transmissor
- Cliente móvel receptor (2)

Classe auxiliar

- GroupDefiner

Objetivo

- Enviar uma mensagem do cliente móvel transmissor para os clientes móveis receptores



Desenvolvendo aplicações com o ContextNet

- Tutorial
  - HelloCore
  - HelloMobile
  - HelloGroup
  - HelloAndroid

Dois componentes:

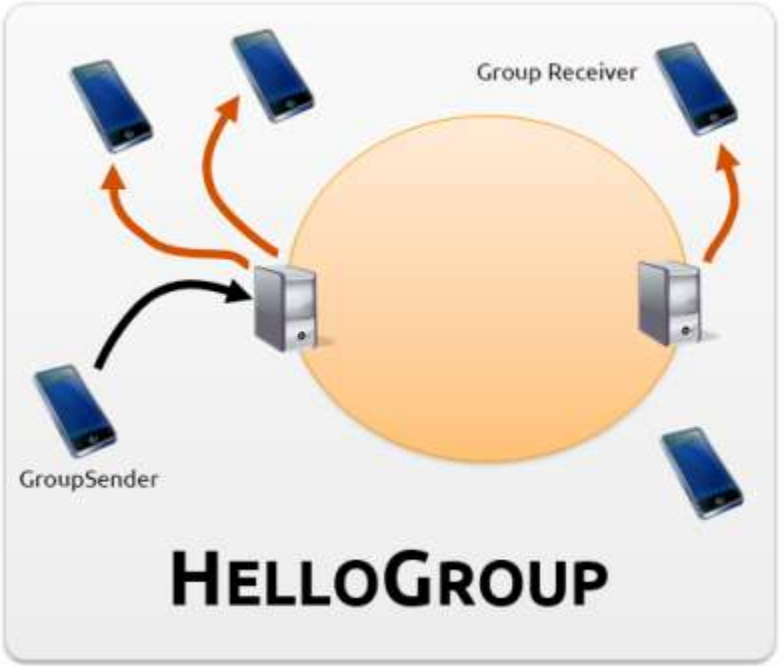
- Cliente móvel transmissor
- Cliente móvel receptor (2)

Classe auxiliar

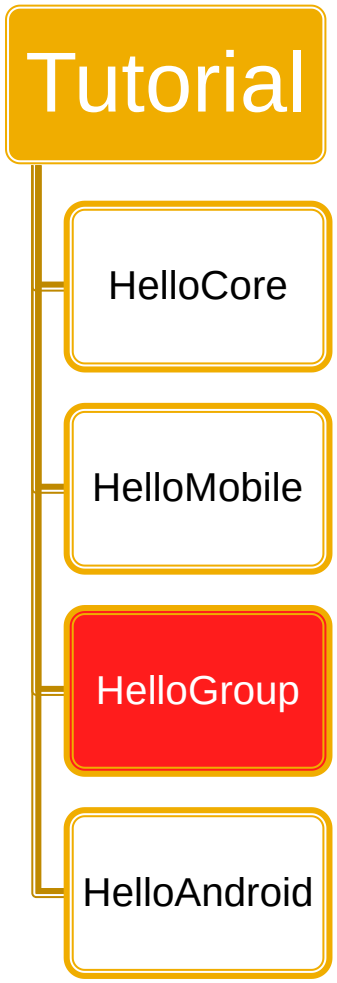
- **GroupDefiner**

Objetivo

- Enviar uma mensagem do cliente móvel transmissor para os clientes móveis receptores

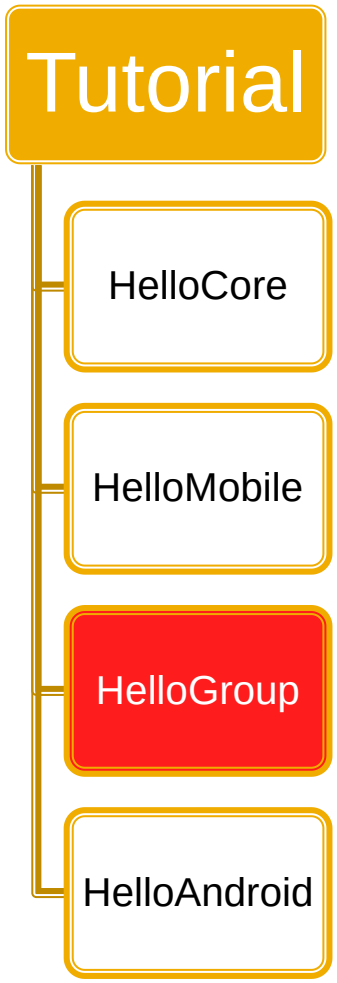


Desenvolvendo aplicações com o ContextNet
HelloGroupDefiner (1/1) – pacotes e main



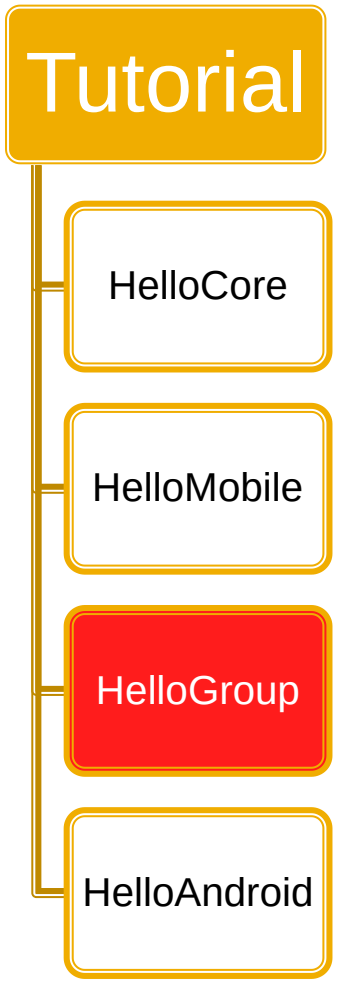

```java
package br.com.meslin.lac;

import lac.cnet.groupdefiner.components.GroupDefiner;
import lac.cnet.groupdefiner.components.groupselector.GroupSelector;

public class HelloGroupDefiner {

  public static void main(String[] args) {

    GroupSelector groupSelector = new HelloGroupSelector();
    new GroupDefiner(groupSelector);

    try {
      Thread.sleep(Long.MAX_VALUE);
    } catch (InterruptedException e) {
      e.printStackTrace();
    }
  }
}
```

Desenvolvendo aplicações com o ContextNet
HelloGroupSelector (1/2) - pacotes




```
package br.com.meslin.lac;

import java.util.HashSet;
import java.util.Set;

import lac.cnet.groupdefiner.components.groupselector.GroupSelector;
import lac.cnet.sddl.objects.GroupRegion;
import lac.cnet.sddl.objects.Message;
```

Desenvolvendo aplicações com o ContextNet
HelloGroupSelector (2/2) – definição de grupos




```
public class HelloGroupSelector implements GroupSelector {
  public int getGroupType() { return 3; }
  public Set<Integer> processGroups(Message nodeMessage) {
    System.out.println("STARTED CLASSIFYING GROUP MESSAGE");
    // Add the node to two groups
    HashSet<Integer> groupCollection = new HashSet<Integer>(2, 1);
    // Some default group (ID = 1)
    groupCollection.add(1);
    // Even (2) or Odd Group (3)
    if (nodeMessage.getSenderId().getLeastSignificantBits() % 2 == 0) {
      groupCollection.add(2);
    } else {
      groupCollection.add(3);
    }
    System.out.println("ENDED CLASSIFYING GROUP MESSAGE\n");
    return groupCollection;
  }
}
```

Desenvolvendo aplicações com o ContextNet

Tutorial

- HelloCore
- HelloMobile
- HelloGroup
- HelloAndroid

Dois componentes:

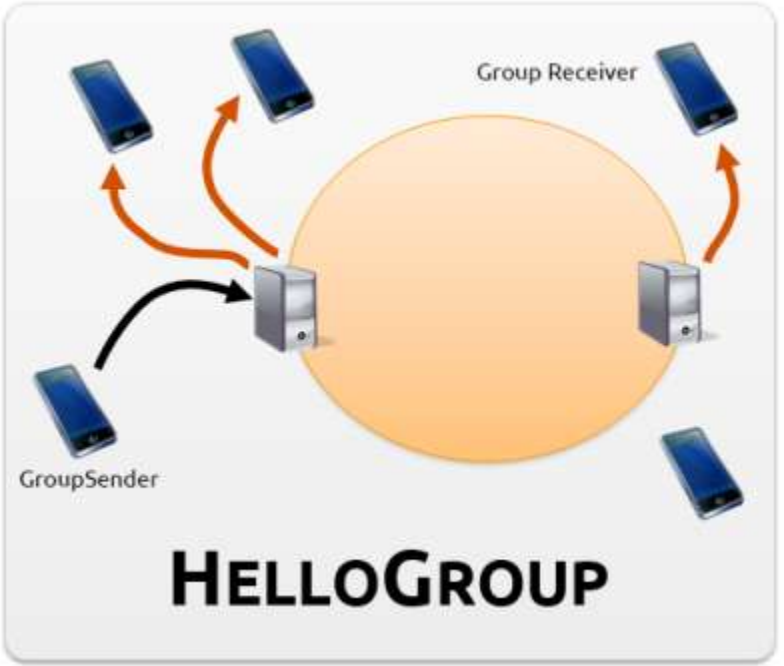
- **Cliente móvel transmissor**
- Cliente móvel receptor (2)

Classe auxiliar

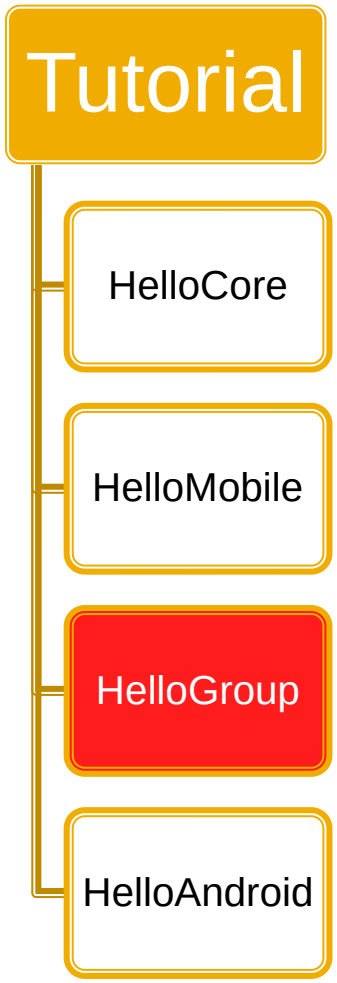
- GroupDefiner

Objetivo

- Enviar uma mensagem do cliente móvel transmissor para os clientes móveis receptores



Desenvolvendo aplicações com o ContextNet
HelloGroupClient – pacotes




```
package br.com.meslin.lac;

import java.io.IOException;
import java.net.InetSocketAddress;
import java.net.SocketAddress;
import java.util.List;
import java.util.logging.Level;
import java.util.logging.Logger;

import lac.cnclib.net.NodeConnection;
import lac.cnclib.net.NodeConnectionListener;
import lac.cnclib.net.groups.Group;
import lac.cnclib.net.groups.GroupCommunicationManager;
import lac.cnclib.net.groups.GroupMembershipListener;
import lac.cnclib.net.mrudp.MrUdpNodeConnection;
import lac.cnclib.sddl.message.ApplicationMessage;
import lac.cnclib.sddl.message.Message;
```

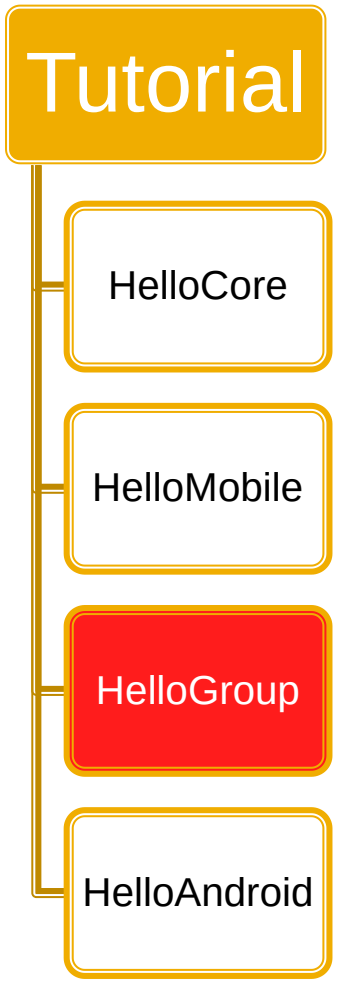
Desenvolvendo aplicações com o ContextNet
HelloGroupClient – construtor

Tutorial

- HelloCore
- HelloMobile
- HelloGroup
- HelloAndroid

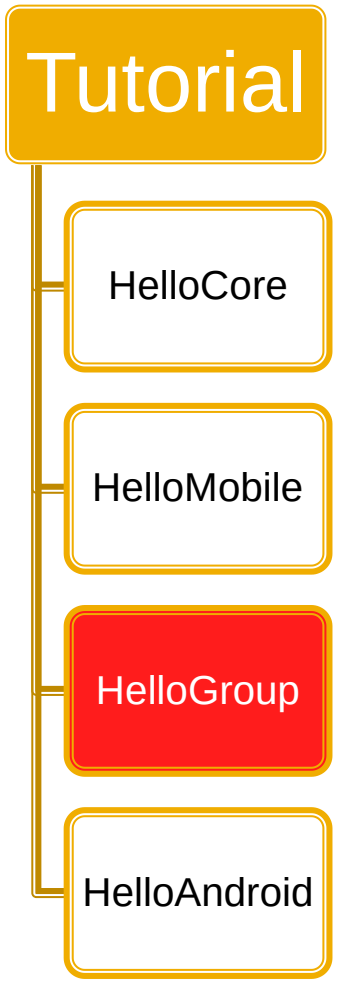
```
public class HelloGroupClient implements NodeConnectionListener, GroupMembershipListener {
  private final static String gatewayIP = "172.0.0.1";
  private final static int gatewayPort  = 5500;
  private GroupCommunicationManager groupManager;

  public HelloGroupClient() {
    InetSocketAddress address = new InetSocketAddress(gatewayIP, gatewayPort);
    try {
      MrUdpNodeConnection connection = new MrUdpNodeConnection();
      connection.connect(address);
      connection.addNodeConnectionListener(this);
      try {
        Thread.sleep(30000);
      } catch (InterruptedException e) {
        e.printStackTrace();
      }
    } catch (IOException e) {
      e.printStackTrace();
    }
  }
```

Desenvolvendo aplicações com o ContextNet
HelloGroupClient – main



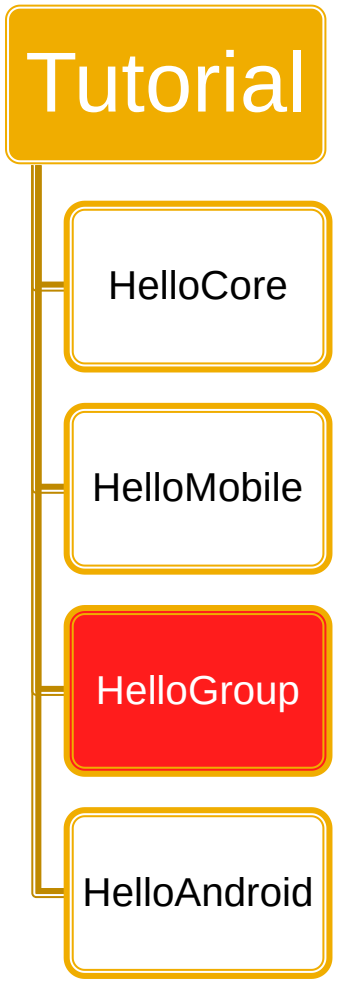

```
public class HelloGroupClient implements NodeConnectionListener, GroupMembershipListener {
…
 public static void main(String[] args) {
  System.err.println("main");
    Logger.getLogger("").setLevel(Level.OFF);

    new HelloGroupClient();
  }
...
}
```

Desenvolvendo aplicações com o ContextNet
HelloGroupClient – envio de mensagem




```
public class HelloGroupClient implements NodeConnectionListener, GroupMembershipListener {
…
 @Override
  public void connected(NodeConnection remoteCon) {
    groupManager = new GroupCommunicationManager(remoteCon);
    groupManager.addMembershipListener(this);
    try {
      // i = 1 Default
      // i = 2 EVEN
      // i = 3 ODD
      for (int i = 1; i < 4; i++) {
        Group group = new Group(3, i);
        ApplicationMessage appMsg = new ApplicationMessage();
        System.err.println("Message GroupID: " + i);
        appMsg.setContentObject("Message GroupID: " + i);
        groupManager.sendGroupcastMessage(appMsg, group);
      }
    } catch (IOException e) {
      e.printStackTrace();
    }
  }
```

Desenvolvendo aplicações com o ContextNet
HelloGroupClient – outros métodos




```
public class HelloGroupClient implements NodeConnectionListener, GroupMembershipListener {
…
 @Override
  public void reconnected(NodeConnection remoteCon, SocketAddress endPoint, boolean
wasHandover, boolean wasMandatory) {}
  @Override
  public void disconnected(NodeConnection remoteCon) {}
  @Override
  public void newMessageReceived(NodeConnection remoteCon, Message message) {
    System.out.println(message.getContentObject());
  }
  @Override
  public void unsentMessages(NodeConnection remoteCon, List<Message> unsentMessages) {}
  @Override
  public void internalException(NodeConnection remoteCon, Exception e) {}
  @Override
  public void enteringGroups(List<Group> groups) {}
  @Override
  public void leavingGroups(List<Group> groups) {}
}
```

Desenvolvendo aplicações com o ContextNet

- Tutorial
  - HelloCore
  - HelloMobile
  - HelloGroup
  - HelloAndroid

Dois componentes:

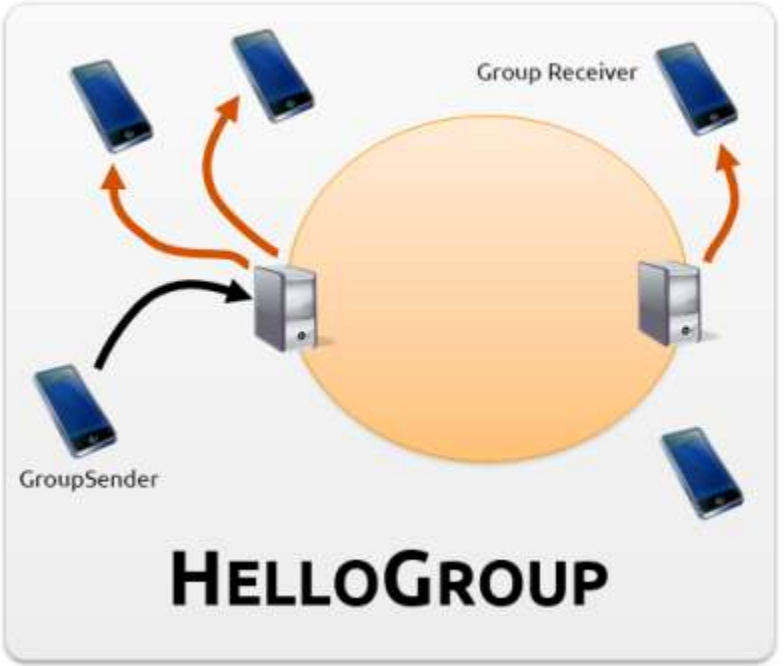
- Cliente móvel transmissor
- **Cliente móvel receptor (2)**

Classe auxiliar

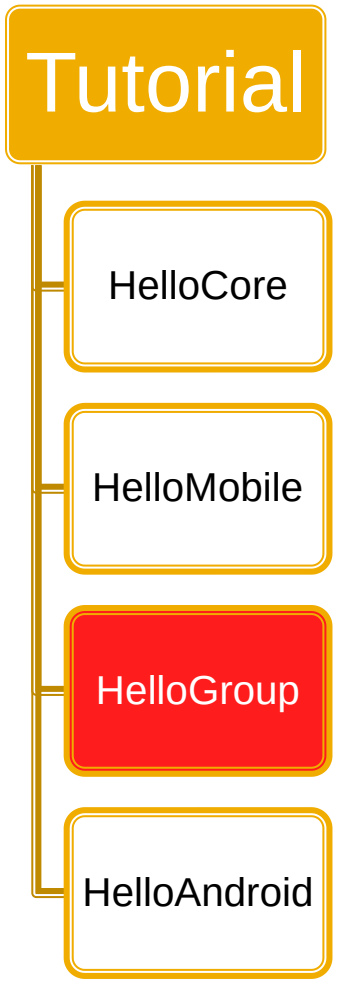
- GroupDefiner

Objetivo

- Enviar uma mensagem do cliente móvel transmissor para os clientes móveis receptores

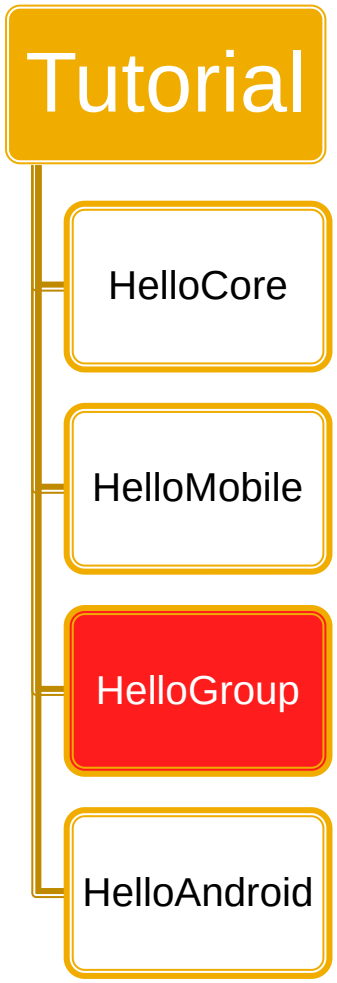


Desenvolvendo aplicações com o ContextNet
HelloGroupDefinerEvenReceiver – pacotes



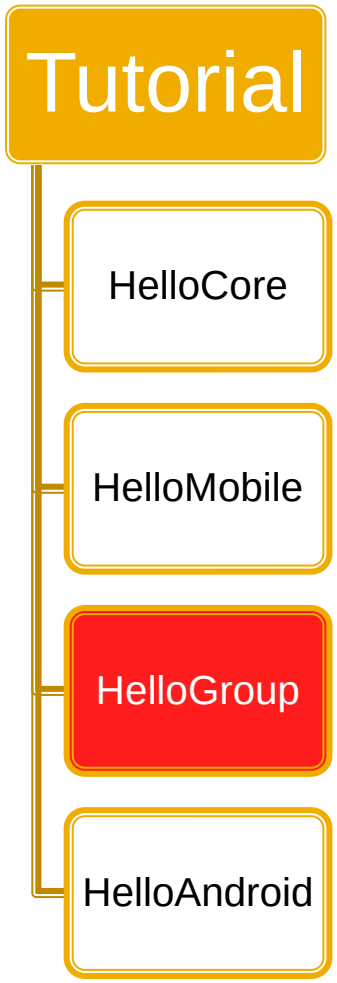

```
package br.com.meslin.lac;

import java.io.IOException;
import java.net.InetSocketAddress;
import java.net.SocketAddress;
import java.util.List;
import java.util.UUID;
import java.util.logging.Level;
import java.util.logging.Logger;

import lac.cnclib.net.NodeConnection;
import lac.cnclib.net.NodeConnectionListener;
import lac.cnclib.net.groups.Group;
import lac.cnclib.net.groups.GroupCommunicationManager;
import lac.cnclib.net.groups.GroupMembershipListener;
import lac.cnclib.net.mrudp.MrUdpNodeConnection;
import lac.cnclib.sddl.message.ApplicationMessage;
import lac.cnclib.sddl.message.Message;
```

Desenvolvendo aplicações com o ContextNet
HelloGroupDefinerEvenReceiver – construtor



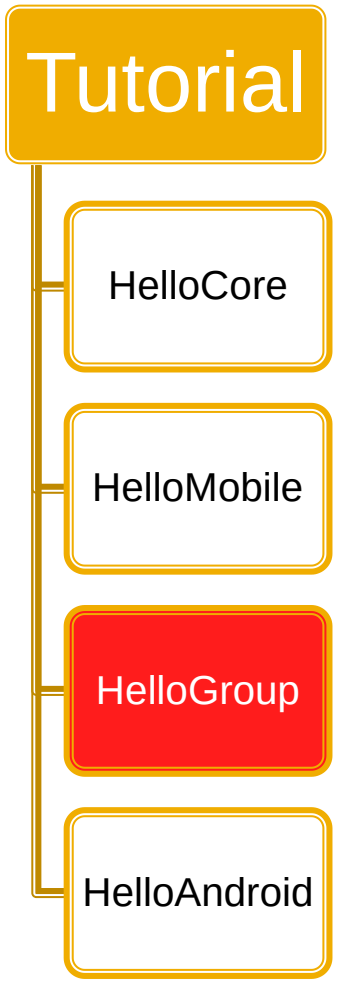

```
public class HelloGroupDefinerEvenReceiver implements NodeConnectionListener,
GroupMembershipListener {
  private final static String         gatewayIP    = "127.0.0.1";
  private final static int            gatewayPort  = 5500;
  private GroupCommunicationManager   groupManager;

  public HelloGroupDefinerEvenReceiver() {
    UUID uuid = UUID.randomUUID();
    while (uuid.getLeastSignificantBits() % 2 != 0) {
      uuid = UUID.randomUUID();
    }
    InetSocketAddress address = new InetSocketAddress(gatewayIP, gatewayPort);
    try {
      MrUdpNodeConnection connection = new MrUdpNodeConnection(uuid);
      connection.connect(address);
      connection.addNodeConnectionListener(this);
    } catch (IOException e) {
      e.printStackTrace();
    }
  }
```

Desenvolvendo aplicações com o ContextNet
HelloGroupDefinerEvenReceiver – main



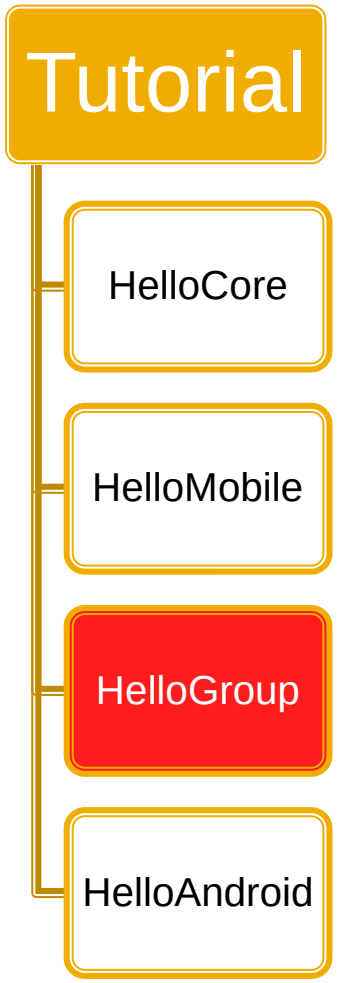

```
public class HelloGroupDefinerEvenReceiver implements NodeConnectionListener,
GroupMembershipListener {
…
 public static void main(String[] args) {
    Logger.getLogger("").setLevel(Level.OFF);
    new HelloGroupDefinerEvenReceiver();
  }

  @Override
  public void connected(NodeConnection remoteCon) {
    groupManager = new GroupCommunicationManager(remoteCon);
    groupManager.addMembershipListener(this);

    try {
      ApplicationMessage message = new ApplicationMessage();
      String serializableContent = "Bogus Message for Group Definer";
      message.setContentObject(serializableContent);

      remoteCon.sendMessage(message);
    } catch (Exception e) {
      e.printStackTrace();
    }
  }
```

Desenvolvendo aplicações com o ContextNet
HelloGroupDefinerEvenReceiver – outros...




```
 @Override
  public void reconnected(NodeConnection remoteCon, SocketAddress endPoint, boolean
wasHandover, boolean wasMandatory) {}
  @Override
  public void disconnected(NodeConnection remoteCon) {}
  @Override
  public void newMessageReceived(NodeConnection remoteCon, Message message) {
    System.out.println(message.getContentObject());
  }
  @Override
  public void unsentMessages(NodeConnection remoteCon, List<Message> unsentMessages) {}
  @Override
  public void internalException(NodeConnection remoteCon, Exception e) {}
  @Override
  public void enteringGroups(List<Group> groups) {}
  @Override
  public void leavingGroups(List<Group> groups) {}
}
```

Desenvolvendo aplicações com o ContextNet
HelloGroupDefinerOddReceiver



- Identico ao HelloGroupDefinerEvenReceiver
- Modificar a linha:

```
while (uuid.getLeastSignificantBits() % 2 != 0) {
```

- Para:

```
while (uuid.getLeastSignificantBits() % 2 == 0) {
```

Desenvolvendo aplicações com o ContextNet Executando

- Tutorial
  - HelloCore
  - HelloMobile
  - HelloGroup
  - HelloAndroid

Precisaremos de cinco "máquinas" (consoles):

Gateway

- `$ java -jar /opt/ContextNet/contextnet-2.7.jar 127.0.0.1 5500 OpenSplice`

Group Definer

- `$ java -jar HelloGroupDefiner.jar`

Clientes quer irão receber a mensagem

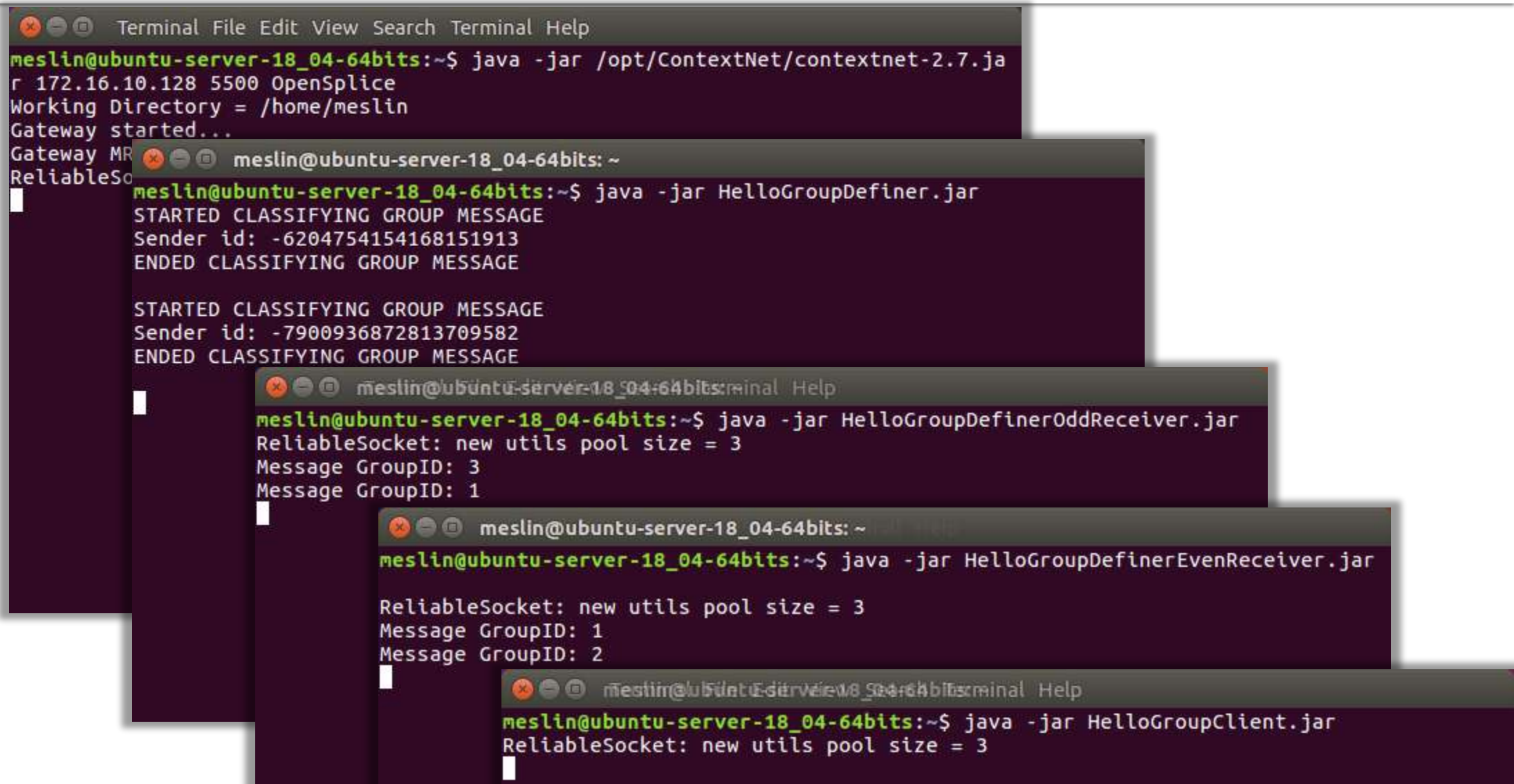
- `$ java -jar HelloGroupDefinerEvenReceiver.jar`
- `$ java -jar HelloGroupDefinerOddReceiver.jar`

Cliente que irá enviar a mensagem

- `$ java -jar HelloGroupClient.jar`

Desenvolvendo aplicações com o ContextNet
Resultado

- Tutorial
  - HelloCore
  - HelloMobile
  - HelloGroup
  - HelloAndroid



Desenvolvendo aplicações com o ContextNet Versão Android

- Tutorial
  - HelloCore
  - HelloMobile
  - HelloGroup
  - HelloAndroid

- Não dessa vez! Programação Android não é a minha praia e não deu tempo de preparar o material, mas tem um excelente tutorial em:
  http://wiki.lac.inf.puc-rio.br/doku.php?id=helloandroid
- Dispositivos móveis reais em:
  https://aws.amazon.com/device-farm/

Desenvolvendo aplicações com o ContextNet
O que é Complex Event Processing?

- Um Paradigma e tecnologia para analisar e manipular **fluxos de eventos** (data stream) em consultas contínuas (on-line)
- CEP inclui:
  - Padrões que verificam relações causais e temporais entre eventos de diferentes tipos
  - Manipulação e geração de eventos complexos, derivados de eventos mais simples
  - Vários tipos de janelas deslizantes para análise contínua e batch de eventos (Ex. win(5 events) ou win(1 sec))
  - Event Processing Network (EPN) define a lógica completa do processamento em vários passos

Desenvolvendo aplicações com o ContextNet Esper Engine

- **Esper** é uma engine CEP open-source:
  - Regras definidas em linguagem SQL-like
  - Regra consiste de Filtros, Maps, Padrões, Janelas e Listeners
  - Para Java e C#

Desenvolvendo aplicações com o ContextNet
Implementando CEP – pacotes

```
package br.com.meslin.cep;

import java.util.HashSet;

import br.com.meslin.onibus.aux.event.EventBusArrivingListener;
import br.com.meslin.onibus.aux.event.EventRegion;
import br.com.meslin.onibus.aux.model.Bus;

import com.espertech.esper.client.Configuration;
import com.espertech.esper.client.EPServiceProvider;
import com.espertech.esper.client.EPServiceProviderManager;
import com.espertech.esper.client.EPStatement;
```

Desenvolvendo aplicações com o ContextNet
Implementando CEP – métodos

```
public class MyCEP {
  ...
  public MyCEP() {
    // configuration
    Configuration config = new Configuration();
    config.addEventTypeAutoName("br.com.meslin.onibus.aux.event");
    config.addEventType("EventRegion", EventRegion.class.getName());
    // creating a statement
    epService = EPServiceProviderManager.getProvider("myCEPEngine", config);
    String expression = "select * from EventRegion having EventRegion.region = 1";
    EPStatement statement = epService.getEPAdministrator().createEPL(expression);
    statement.addListener(new EventBusArrivingListener());
  }

  public void createEvent(Bus bus, HashSet<Integer> newGroups) {
    for(int group : newGroups) {
      if(!bus.getGroups().contains(group)) {
        EventRegion eventRegion = new EventRegion(bus, group, System.currentTimeMillis());
        epService.getEPRuntime().sendEvent(eventRegion);
      }
    }
  }
}
```

Desenvolvendo aplicações com o ContextNet
Implementando CEP – classe de evento

```
package br.com.meslin.onibus.aux.event;

import java.util.Date;

import br.com.meslin.onibus.aux.model.Bus;

public class EventRegion {
  private Bus bus;
  private int region;
  private Date timeStamp;

  public EventRegion(Bus bus, int region, long time) {
    this.bus = bus;
    this.region = region;
    this.timeStamp = new Date(time);
  }
}
```

Desenvolvendo aplicações com o ContextNet
Implementando CEP – listener

```
package br.com.meslin.onibus.aux.event;

import java.util.Date;
import java.util.HashMap;
import java.util.Map;

import org.json.JSONException;
import org.json.JSONObject;

import br.com.meslin.onibus.aux.Debug;
import br.com.meslin.onibus.aux.StaticLibrary;
import br.com.meslin.onibus.aux.contextnet.MessageSender;
import br.com.meslin.onibus.aux.model.Bus;

import com.espertech.esper.client.EventBean;
import com.espertech.esper.client.PropertyAccessException;
import com.espertech.esper.client.UpdateListener;
```

Desenvolvendo aplicações com o ContextNet
Implementando CEP – listener

```
public class EventBusArrivingListener implements UpdateListener {
  private Map<Integer, Integer> nextRegion;

  public EventBusArrivingListener() {
    nextRegion = new HashMap<Integer, Integer>();
    nextRegion.put(1, 10002);
    nextRegion.put(2, 10001);
  }

  @Override
  public void update(EventBean[] newEvents, EventBean[] oldEvents) {
    try {
      MessageSender messageSender = MessageSender.getInstance();
      String linha = ((Bus)newEvents[0].get("bus")).getLinha();
      String ordem = ((Bus)newEvents[0].get("bus")).getOrdem();
      JSONObject jsonObject = new JSONObject();
      jsonObject.put("message", "ATENÇÃO!!! Seu ônibus " + linha + "(" + ordem + ") está chegando");
      messageSender.sendMessageToGroup(targetRegion, jsonObject);
    } catch (PropertyAccessException | JSONException e) {
      e.printStackTrace();
    }
  }
}
```

Perguntas?

Exercícios

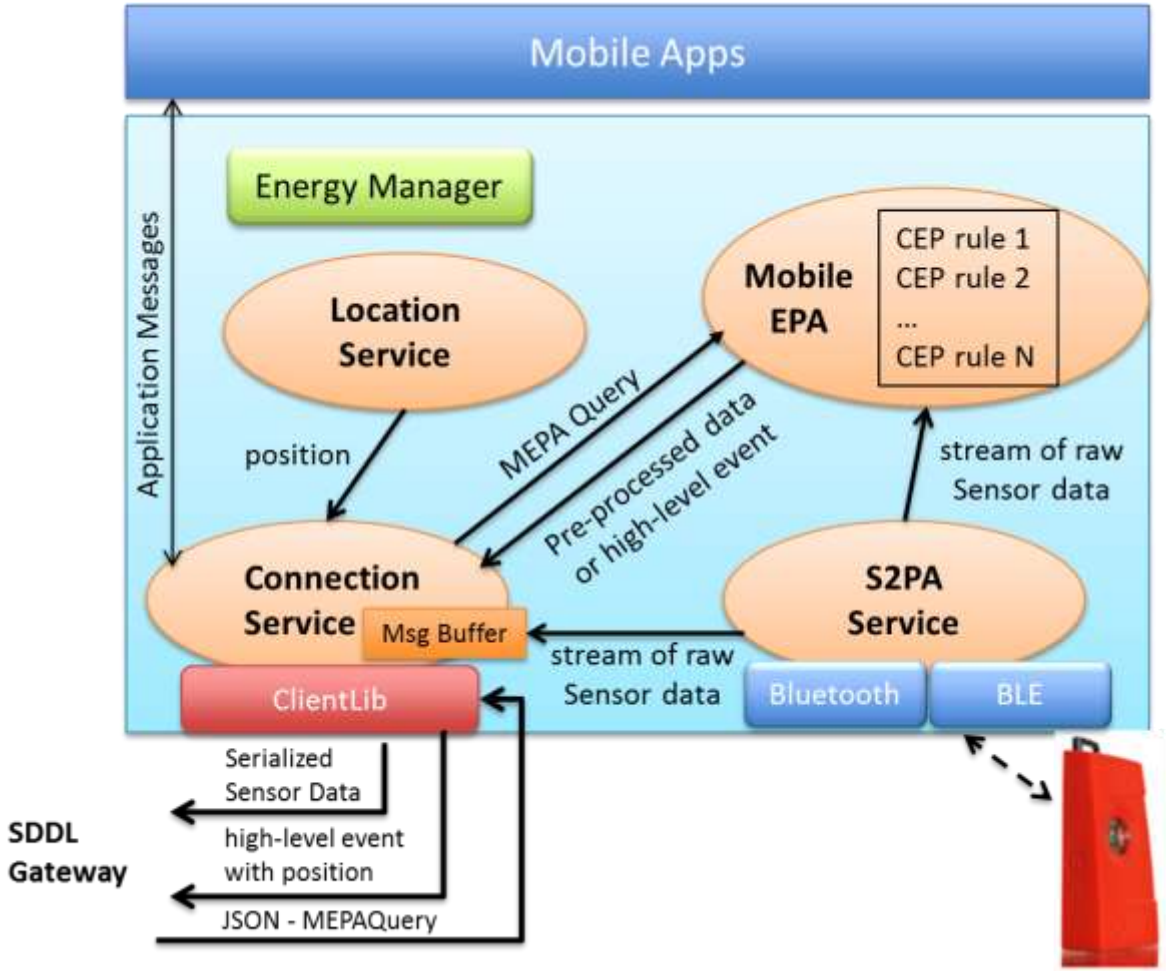
Exercício

- Faça um sistema de mensagens para grupos
  - Crie os grupos baseados em tipos de ocupação funcional do usuário

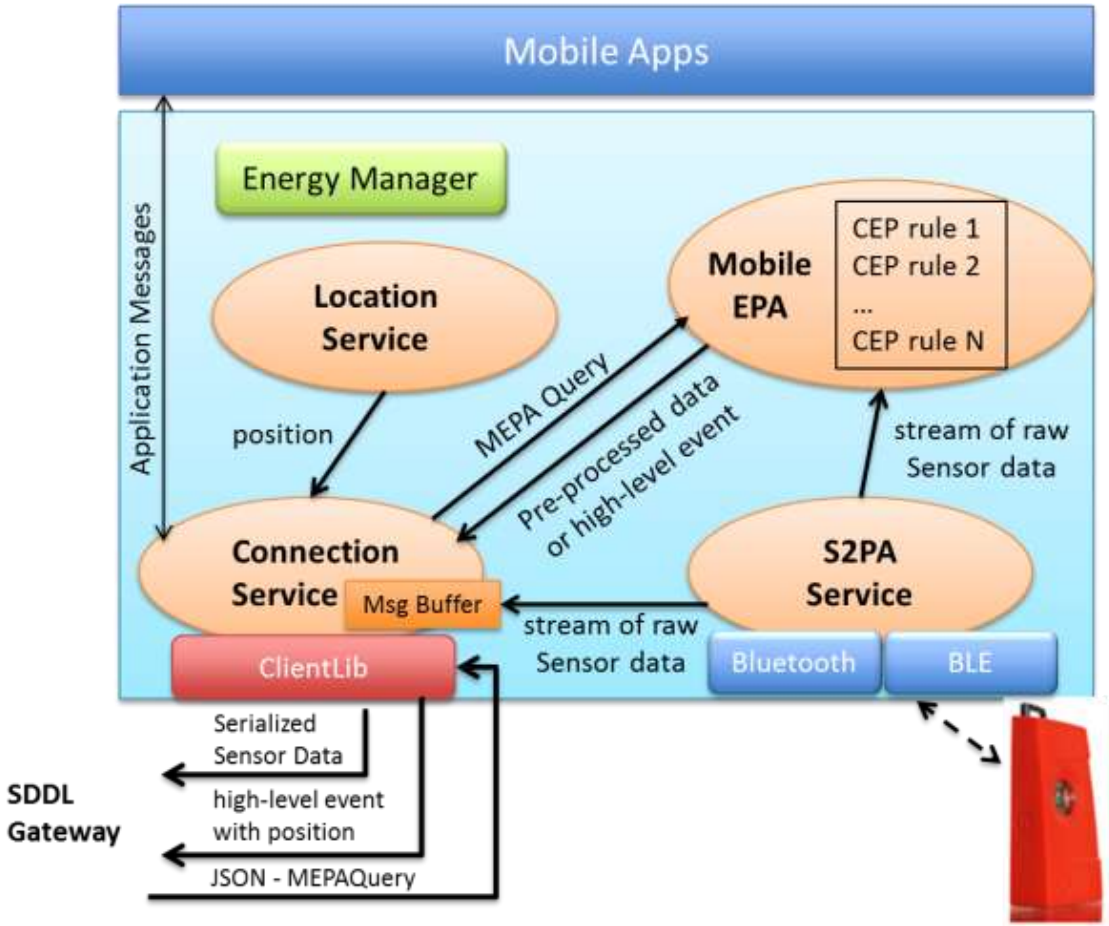
Utilizando o Mobile-Hub para captura de dados

Transmitindo e recebendo dados para/de o ContextNet

Mobile-Hub
Arquitetura



Mobile-Hub
Arquitetura



- **S2PA** (Short-range Sensor, Presence, and Actuation): descoberta e monitoramento de M-OBJs
- Connection Service: gerência de mensagens
- Location Service: obtenção da localização através de GPS, rede ou manual
- Energy Manager: monitor de bateria
- MEPA (Mobile Event Processing Agente) Service: CEP

Mobile-Hub
Requerimentos

The M-Hub application (apk) requires:

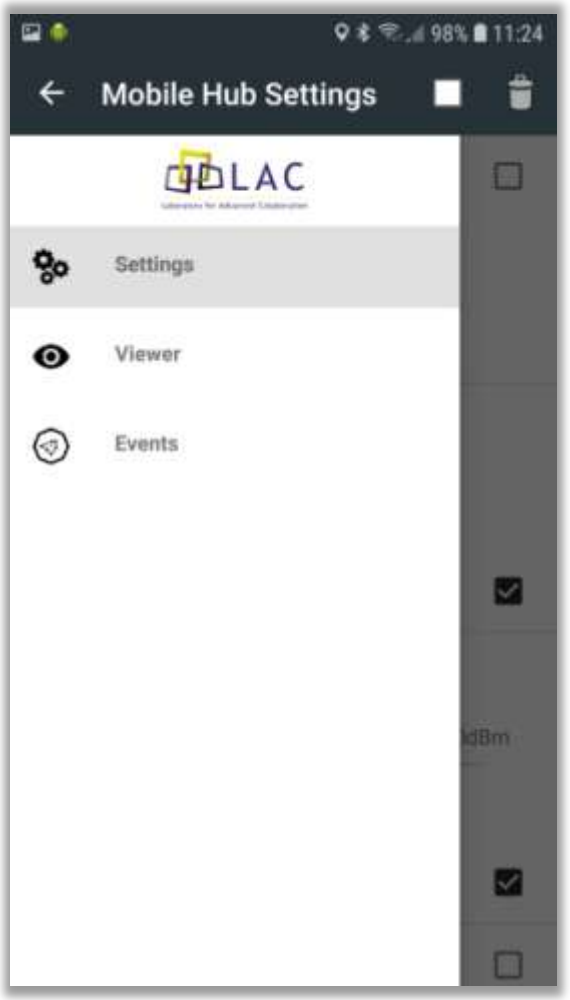
- Android >=4.3
- a Gateway.

The current implementation has support for Bluetooth Low Energy as a WPAN technology

On the settings activity of the M-Hub, you can uncheck some services/manager

Mobile-Hub

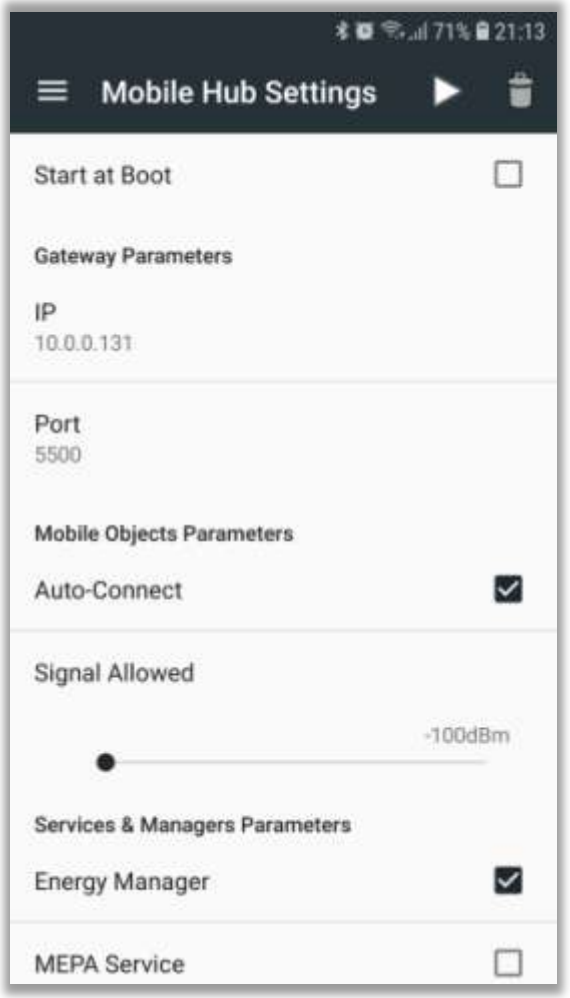
Configurações



Mobile-Hub
Configurações

Gateway

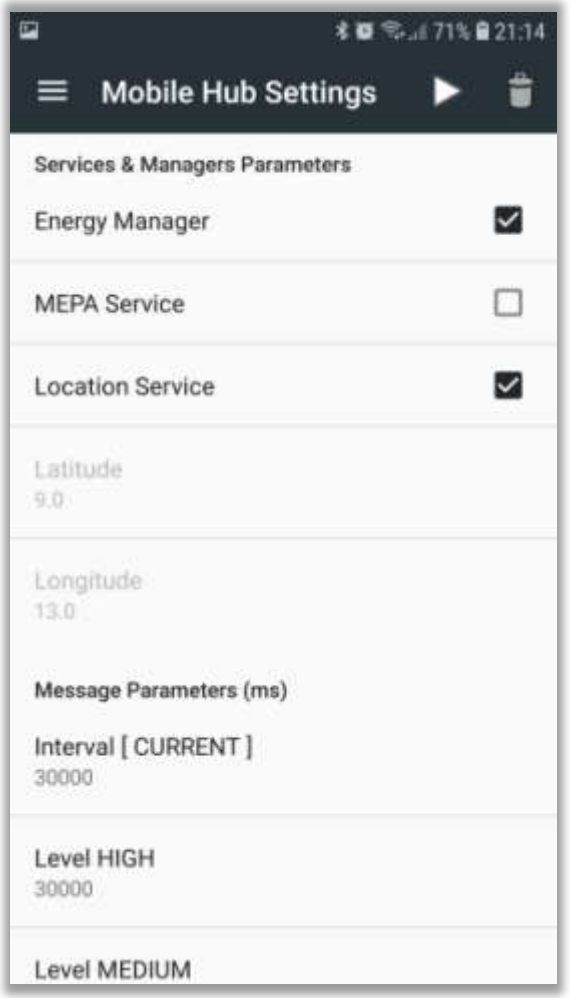
- Endereço IP
- Porta



Mobile-Hub
Configurações

Parâmetros

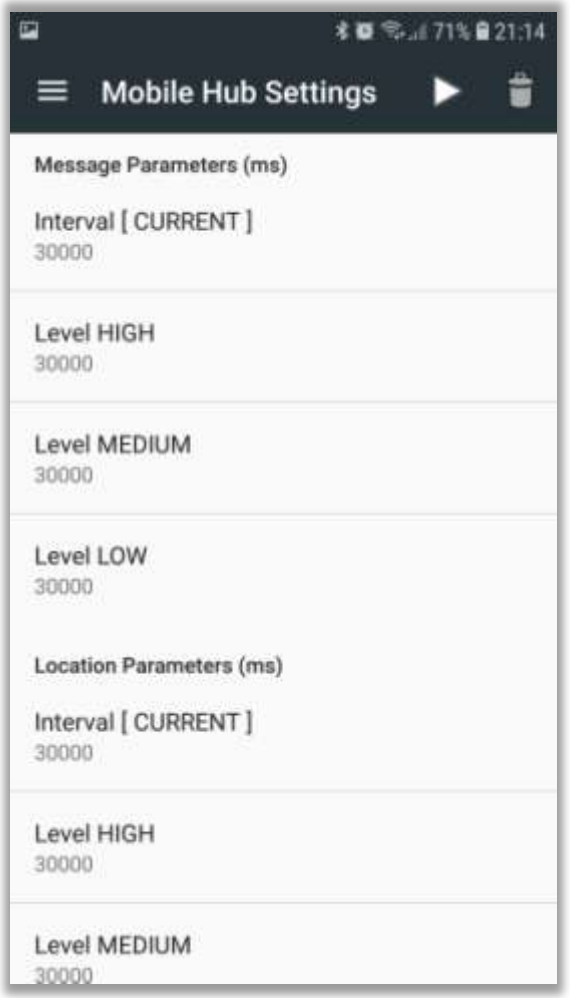
- Gerenciamento
- Serviços



Mobile-Hub
Configurações

Intervalo de mensagens de acordo com o nível de bateria

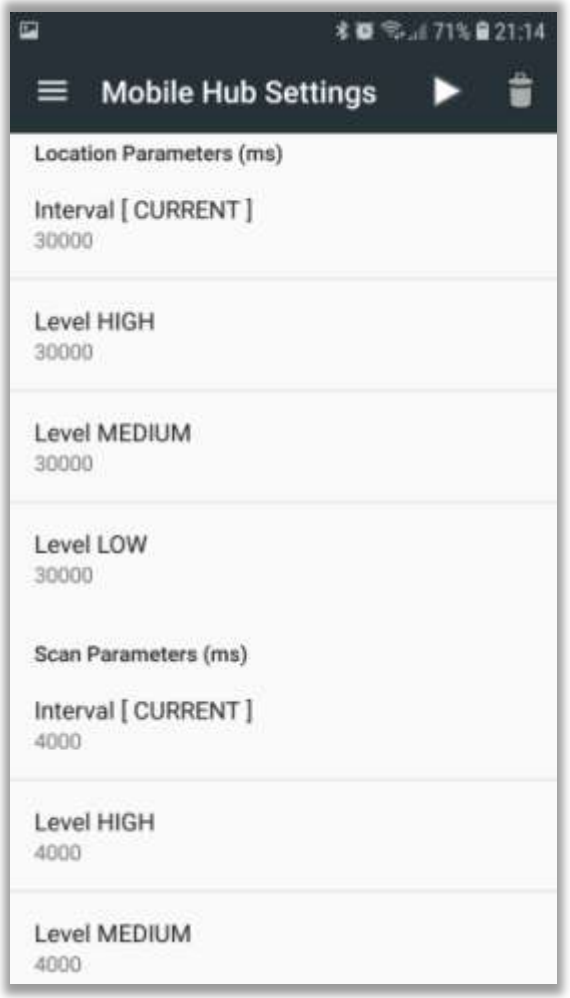
- Atual
- Nível alto
- Nível médio
- Nível baixo



Mobile-Hub
Configurações

Intervalo de mensagens de localização de acordo com o nível de bateria

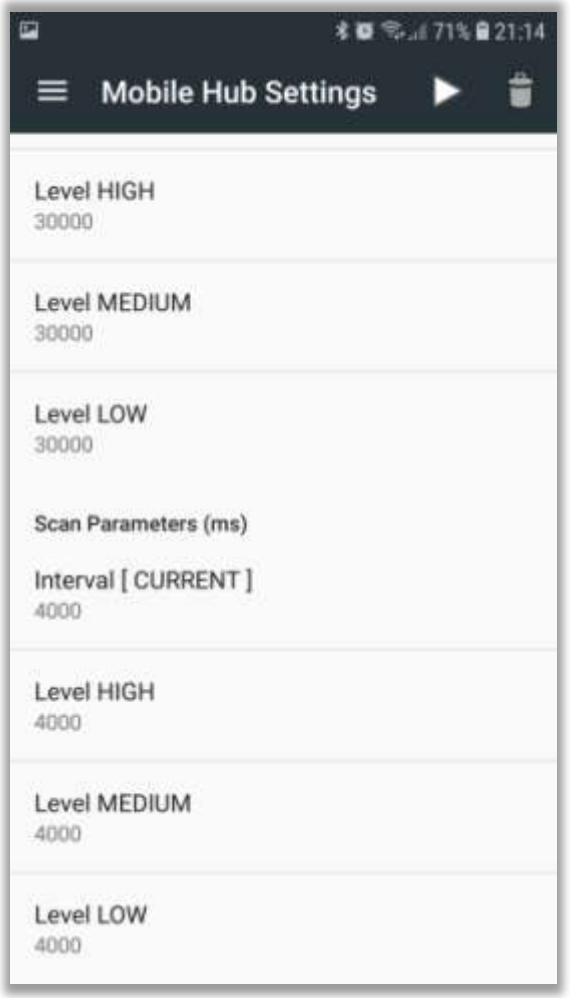
- Atual
- Nível alto
- Nível médio
- Nível baixo



Mobile-Hub
Configurações

Intervalo de procura por dispositivos de acordo com o nível de bateria

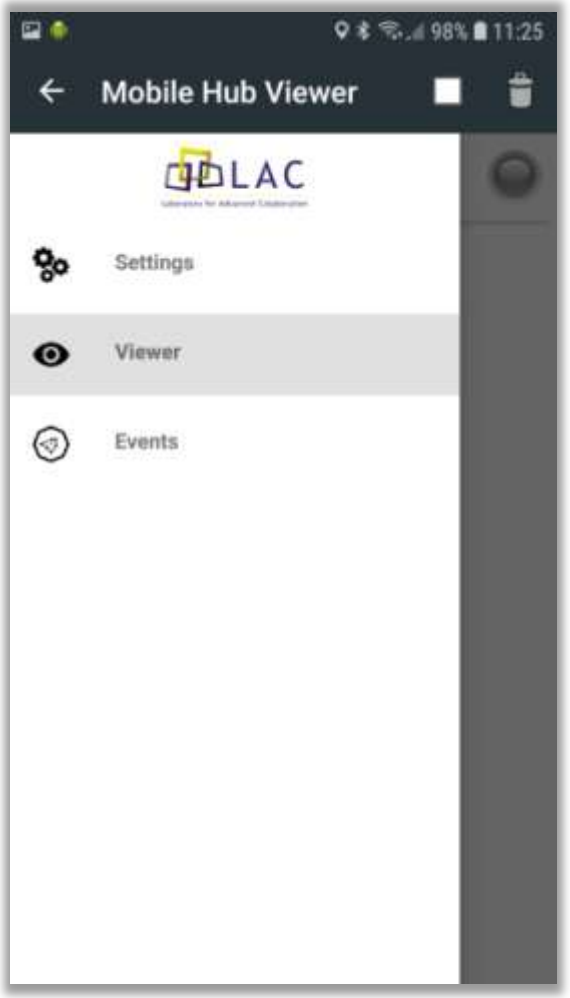
- Atual
- Nível alto
- Nível médio
- Nível baixo



Mobile-Hub

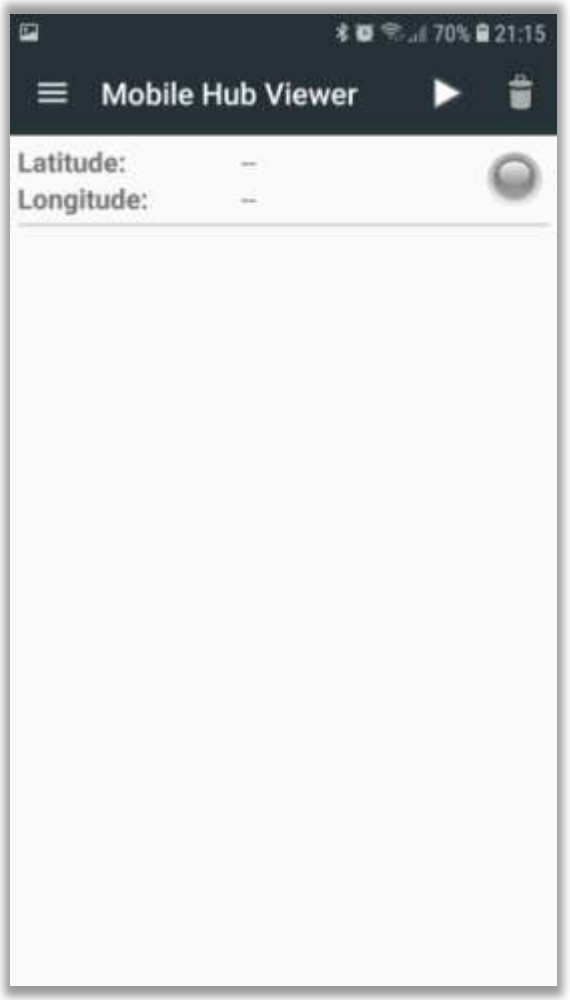
Visualização

- Dispositivos conectados



Mobile-Hub
Visualização

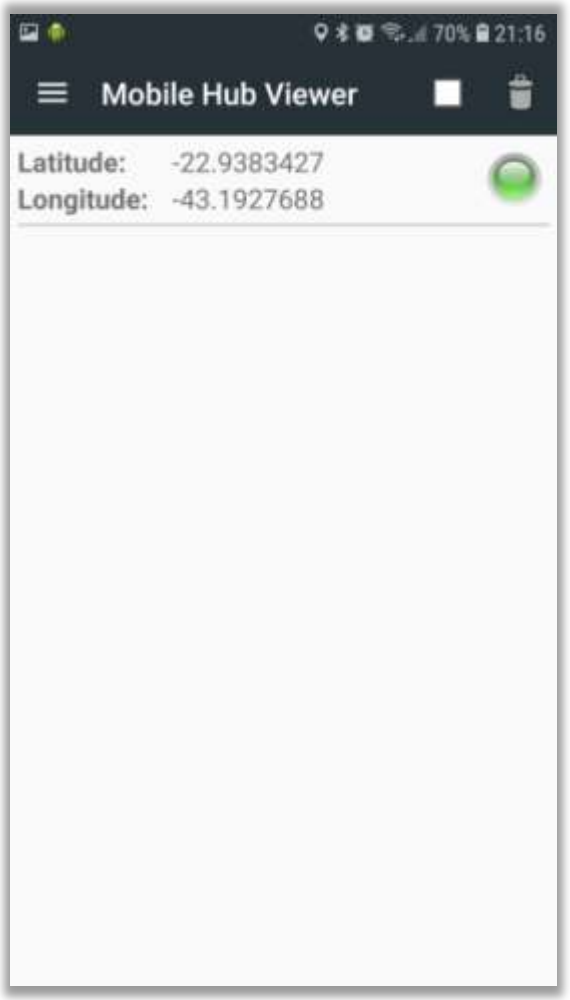
Nenhum dispositivo conectado



Mobile-Hub
Visualização

Transmitindo posição

- Latutude: -22,9383427
- Longitude: -43,1927688



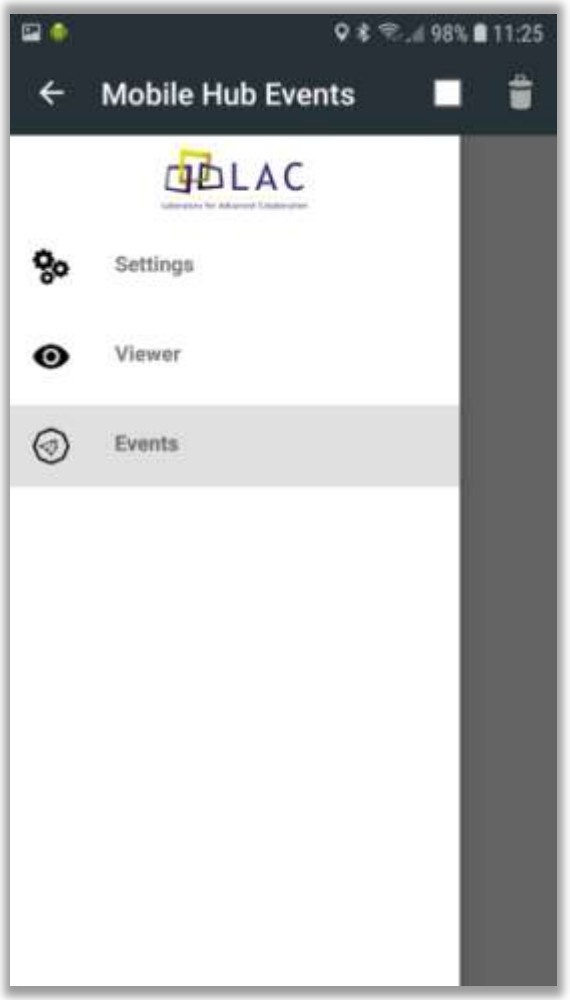
Mobile-Hub
Visualização

Conectado a um sensor tag

- Acelerômetro
- Barômetro
- Giroscópio
- Umidade
- Magnetômetro
- Temperatura

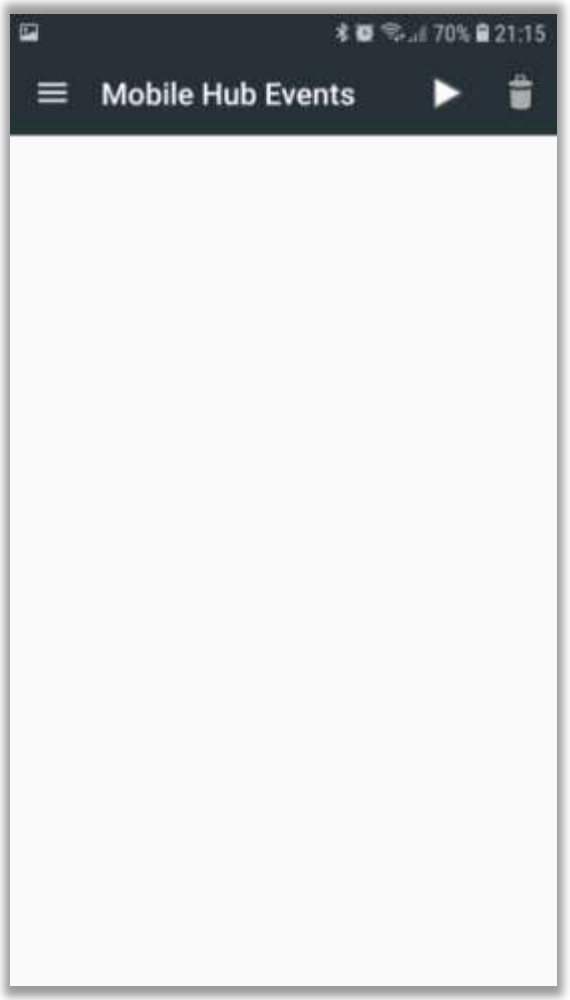
Mobile-Hub

Visualização de eventos



Mobile-Hub
Eventos

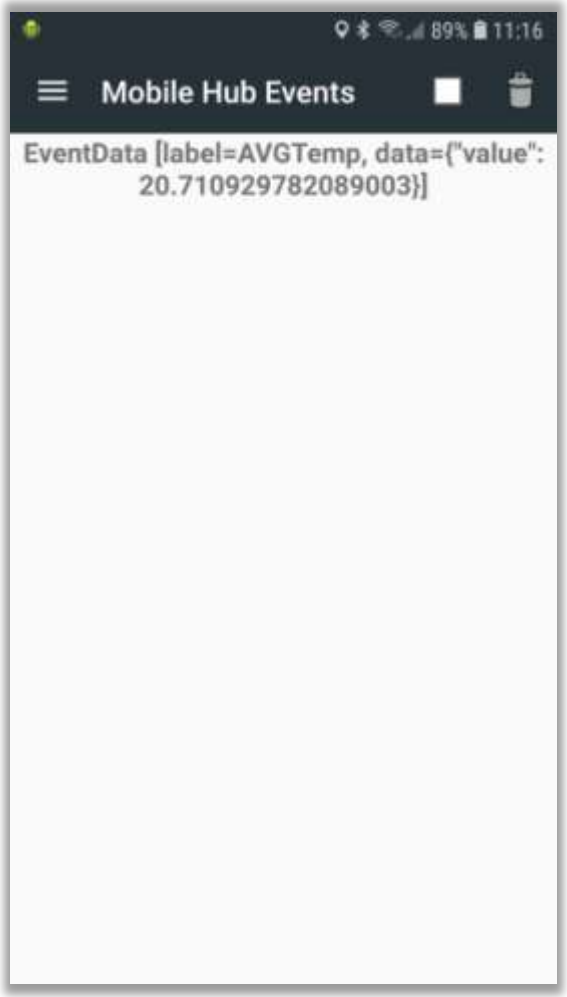
Nenhum evento a ser exibido



Mobile-Hub
Eventos

Evento de temperatura

- Nome: AVGTemp
- Dado: 20,710929782089003



Mobile-Hub

Mensagens

CONEXÃO


```
{
   "uuid":"9509494b-b270-4cd7-a5a2-08cc6bb998d1",
   "source":"1-2CB6E5B2DF07",
   "action":"found",
   "signal":-80,
   "latitude":-22.9383307,
   "longitude":-43.1927734,
   "tag":"SensorData",
   "timestamp":1563288348
}
```


LOCALIZAÇÃO


```
{
   "uuid":"9509494b-b270-4cd7-a5a2-08cc6bb998d1",
   "date":"Tue Jul 16 11:45:23 GMT-03:00 2019",
   "accuracy":19.6560001373291,
   "provider":"network",
   "speed":0,
   "bearing":0,
   "altitude":46.89999771118164,
   "connection":"java.lang.InfoConnectivityWifi",
   "battery":96,
   "charging":false,
   "tag":"LocationData",
   "latitude":-22.9383307,
   "longitude":-43.1927734,
   "timestamp":1563288323
}
```

Mobile-Hub

Mensagens

CONEXÃO


```
{
   "uuid":"9509494b-b270-4cd7-a5a2-08cc6bb998d1",
   "source":"1-2CB6E5B2DF07",
   "action":"found",
   "signal":-80,
   "latitude":-22.9383307,
   "longitude":-43.1927734,
   "tag":"SensorData",
   "timestamp":1563288348
}
```


M-Hub UUID

M-Obj UUID

LOCALIZAÇÃO


```
{
   "uuid":"9509494b-b270-4cd7-a5a2-08cc6bb998d1",
   "date":"Tue Jul 16 11:45:23 GMT-03:00 2019",
   "accuracy":19.6560001373291,
   "provider":"network",
   "speed":0,
   "bearing":0,
   "altitude":46.89999771118164,
   "connection":"java.lang.InfoConnectivityWifi",
   "battery":96,
   "charging":false,
   "tag":"LocationData",
   "latitude":-22.9383307,
   "longitude":-43.1927734,
   "timestamp":1563288323
}
```

Mobile-Hub

Mensagens

BARÔMETRO


```
{
   "uuid":"9509494b-b270-4cd7-a5a2-08cc6bb998d1",
   "source":"1-78A5048CEA0A",
   "action":"read",
   "signal":-53,
   "sensor_name":"Barometer",
   "sensor_value":[101508.01809210484],
   "latitude":-22.9383307,
   "longitude":-43.1927734,
   "tag":"SensorData",
   "timestamp":1563288350
}
```


UMIDADE


```
{
   "uuid":"9509494b-b270-4cd7-a5a2-08cc6bb998d1",
   "source":"1-78A5048CEA0A",
   "action":"read",
   "signal":-53,
   "sensor_name":"Humidity",
   "sensor_value":[69.40245056152344],
   "latitude":-22.9383307,
   "longitude":-43.1927734,
   "tag":"SensorData",
   "timestamp":1563288351
}
```

Mobile-Hub
Mensagens

GIROSCÓPIO


```
{
   "uuid":"9509494b-b270-4cd7-a5a2-08cc6bb998d1",
   "source":"1-78A5048CEA0A",
   "action":"read",
   "signal":-53,
   "sensor_name":"Gyroscope",
   "sensor_value":[1.29699707,-2.357482910,0.0915527],
   "latitude":-22.9383307,
   "longitude":-43.1927734,
   "tag":"SensorData",
   "timestamp":1563288351
}
```


TEMPERATURA


```
{
   "uuid":"9509494b-b270-4cd7-a5a2-08cc6bb998d1",
   "source":"1-78A5048CEA0A",
   "action":"read",
   "signal":-53,
   "sensor_name":"Temperature",
   "sensor_value":[26.34375,22.941261956570088],
   "latitude":-22.9383307,
   "longitude":-43.1927734,
   "tag":"SensorData",
   "timestamp":1563288351
}
```

Mobile-Hub

Mensagens

ACELERÔMETRO


```
{
   "uuid":"9509494b-b270-4cd7-a5a2-08cc6bb998d1",
   "source":"1-78A5048CEA0A",
   "action":"read",
   "signal":-53,
   "sensor_name":"Accelerometer",
   "sensor_value":[0,-0.984375,-0.09375],
   "latitude":-22.9383307,
   "longitude":-43.1927734,
   "tag":"SensorData",
   "timestamp":1563288351
}
```


MAGNETÔMETRO


```
{
   "uuid":"9509494b-b270-4cd7-a5a2-08cc6bb998d1",
   "source":"1-78A5048CEA0A",
   "action":"read",
   "signal":-53,
   "sensor_name":"Magnetometer",
   "sensor_value":[-8.39233398,-8.1176757,29.4799804],
   "latitude":-22.9383307,
   "longitude":-43.1927734,
   "tag":"SensorData",
   "timestamp":1563288350
}
```

Mobile-Hub

Mensagens

EVENTO

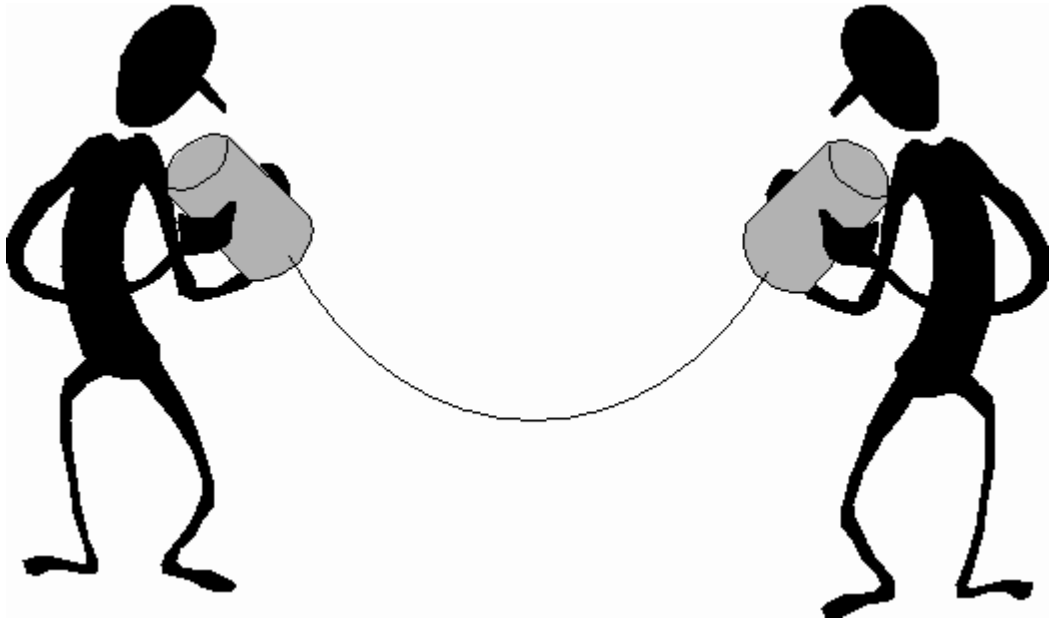

```
{
  "tag": "EventData",
  "uuid": "b06de58d-6a20-44f9-8cd4-83f074c2edd6",
  "label": "AVGTemp",
  "data": {"value": 22.30},
  "latitude": -22.98137128,
  "longitude": -43.23421961,
  "timestamp": 1442169467
}
```


STATUS


```
{
  "tag": "ErrorData|ReplyData",
  "uuid": "b06de58d-6a20-44f9-8cd4-83f074c2edd6",
  "component": "MEPAService|S2PAService",
  "message": "Action not supported.",
  "latitude": -22.98137128,
  "longitude": -43.23421961,
  "timestamp": 1442169467
}
```

Mobile-Hub

Recebendo mensagens



Tipo de Mensagem	Nome da Tag
Dados	SensorData
Eventos	EventData
Respostas	ReplyData
Erros	ErrorData

Mobile-Hub

Recebendo mensagens (1/6)

```
package br.com.meslin.lac;

import org.json.simple.JSONObject;
import org.json.simple.parser.JSONParser;

import lac.cnet.sddl.objects.ApplicationObject;
import lac.cnet.sddl.objects.Message;
import lac.cnet.sddl.objects.PrivateMessage;
import lac.cnet.sddl.udi.core.SddlLayer;
import lac.cnet.sddl.udi.core.UniversalDDSLayerFactory;
import lac.cnet.sddl.udi.core.listener.UDIDataReaderListener;
```

Mobile-Hub

Recebendo mensagens (2/6)

```
public class MyProcessingNode implements UDIDataReaderListener<ApplicationObject> {
  private SddlLayer core;
  public MyProcessingNode() {
    core = UniversalDDSLayerFactory.getInstance();
    core.createParticipant(UniversalDDSLayerFactory.CNET_DOMAIN);
    core.createPublisher();
    core.createSubscriber();
    Object receiveMessageTopic = core.createTopic(Message.class, Message.class.getSimpleName());
    core.createDataReader(this, receiveMessageTopic);
    Object toMobileNodeTopic = core.createTopic(PrivateMessage.class,PrivateMessage.class.getSimpleName());
    core.createDataWriter(toMobileNodeTopic);
    System.out.println("=== Server Started (Listening) ===");
    synchronized (this) {
      try {
        this.wait();
      } catch (InterruptedException e) {
        e.printStackTrace();
      }
    }
  }
  ...
```

Mobile-Hub

Recebendo mensagens (3/6)

```
public class MyProcessingNode implements UDIDataReaderListener<ApplicationObject> {
  …
  public static void main(String[] args) {
    new MyProcessingNode();
    while (true);
  }
```

Mobile-Hub
Recebendo mensagens (4/6)

```
public class MyProcessingNode implements UDIDataReaderListener<ApplicationObject> {
  …
  @Override
  public void onNewData(ApplicationObject topicSample) {
    if(topicSample instanceof Message) {
      Message msg = (Message) topicSample;
      String content = new String(msg.getContent());
      JSONParser parser = new JSONParser();

      try {
        JSONObject object = (JSONObject) parser.parse(content);
        String tag = (String) object.get("tag");

        switch (tag) {
          ...
        }
      } catch (Exception e) {
        System.out.println("Mensagem sem tag (pode ter sido um keep-alive): " + e.getMessage());
      }
    }
  }
```

Mobile-Hub

Recebendo mensagens (5/6)

```
public class MyProcessingNode implements UDIDataReaderListener<ApplicationObject> {
   ...
        switch (tag) {
          case "SensorData":
            handleSensorData(object);
          break;

          case "EventData":
            final String label = (String) object.get("label");
            final String data = (String) object.get("data");
            handleEvent(label, data);
          break;

          case "ReplyData":
          case "ErrorData":
            handleMessage(tag, object);
          break;
        }
  ...
```

Mobile-Hub

Recebendo mensagens (6/6)

```
public class MyProcessingNode implements UDIDataReaderListener<ApplicationObject> {
  …
  private void handleMessage(String tag, JSONObject object) {
    System.out.println("handleMessage: " + tag);
  }

  private void handleEvent(String label, String data) {
    System.out.println("handleEvent: " + data);
  }

  private void handleSensorData(JSONObject object) {
    System.out.println("handleSensorData: " + object.toString());
  }
}
```

Sugestão: implemente usando alguma interface gráfica!

Mobile-Hub
Enviando regras CEP

- Linguagem EPS da EsperTech
  http://esper.espertech.com/release-5.0.0/esper-reference/html_single/index.html

- Formato:

```
[
  {
    "MEPAQuery": {
      "type":"add|remove|start|stop|clear|get",
      "label":"AVGTemp",
      "object":"rule|event",
      "rule":"SELECT avg(sensorValue[1]) as value FROM
SensorData(sensorName='Temperature').win:time_batch(10 sec)",
      "target":"local|global"
    }
  }
]
```

Mobile-Hub

Enviando regras CEP – pacotes (1/5)

```
package br.com.meslin.lac;

import java.io.IOException;
import java.net.InetSocketAddress;
import java.net.SocketAddress;
import java.util.List;
import java.util.UUID;

import lac.cnclib.net.NodeConnection;
import lac.cnclib.net.NodeConnectionListener;
import lac.cnclib.net.mrudp.MrUdpNodeConnection;
import lac.cnclib.sddl.message.ApplicationMessage;
import lac.cnclib.sddl.message.ClientLibProtocol.PayloadSerialization;
import lac.cnclib.sddl.message.Message;

import org.json.simple.JSONArray;
import org.json.simple.JSONObject;
```

Mobile-Hub

Enviando regras CEP - construtor (2/5)

```
public class SendMEPAQuery implements NodeConnectionListener {
  private MrUdpNodeConnection connection;
  private String gatewayIP = "127.0.0.1";
  private int gatewayPort = 5500;

  public SendMEPAQuery() {
    InetSocketAddress address = new InetSocketAddress(gatewayIP, gatewayPort);
    UUID myUUID = UUID.randomUUID();
    try {
      connection = new MrUdpNodeConnection(myUUID);
      connection.addNodeConnectionListener(this);
      connection.connect(address);
    } catch (IOException e) {
      e.printStackTrace();
    }
}
```

Mobile-Hub

Enviando regras CEP – main e métodos auxiliares (3/5)

```
public class SendMEPAQuery implements NodeConnectionListener {
…
  public static void main(String[] args) {
    SendMEPAQuery mepa = new SendMEPAQuery();
    mepa.doAll();
  }
…
  @Override
  public void connected(NodeConnection arg0) { }
  @Override
  public void disconnected(NodeConnection arg0) { }
  @Override
  public void internalException(NodeConnection arg0, Exception arg1) { }
  @Override
  public void newMessageReceived(NodeConnection arg0, Message topicSample) { }
  @Override
  public void reconnected(NodeConnection arg0, SocketAddress arg1, boolean arg2, boolean arg3) { }
  @Override
  public void unsentMessages(NodeConnection arg0, List<Message> arg1) { }
}
```

Mobile-Hub

Enviando regras CEP – criando a mensagem (4/5)

```
public class SendMEPAQuery implements NodeConnectionListener {
  ...
  private void doAll() {
    String eplString = "select avg(sensorValue[1]) as value "
      + "from SensorData(sensorName='Temperature').win:time_batch(10 sec) "
      + "where sensorValue[1] > 20";
    JSONObject mepaQuery =  new JSONObject();
    mepaQuery.put("type", "add");           // add|remove|start|stop|clear|get
    mepaQuery.put("label", "AVGTemp");              // rule|event
    mepaQuery.put("object", "rule");
    mepaQuery.put("rule", eplString);
    mepaQuery.put("target", "local");

    JSONArray jsonMsg = new JSONArray();
    JSONObject oneQuery = new JSONObject();
    oneQuery.put("MEPAQuery", mepaQuery);
    jsonMsg.add(oneQuery);
  …
}
```

Mobile-Hub

Enviando regras CEP – enviando a mensagem (5/5)

```
public class SendMEPAQuery implements NodeConnectionListener {
  ...
  private void doAll() {
    ...
    ApplicationMessage msg = new ApplicationMessage();
    msg.setRecipientID(UUID.fromString("63c38558-8f1e-4096-bd43-876872072f50"));
    msg.setContentObject(jsonMsg.toString());
    msg.setPayloadType(PayloadSerialization.JSON);
    try {
      connection.sendMessage(msg);
    } catch (IOException e) {
      e.printStackTrace();
    }
  }
  ...
}
```

Perguntas?

Exercícios

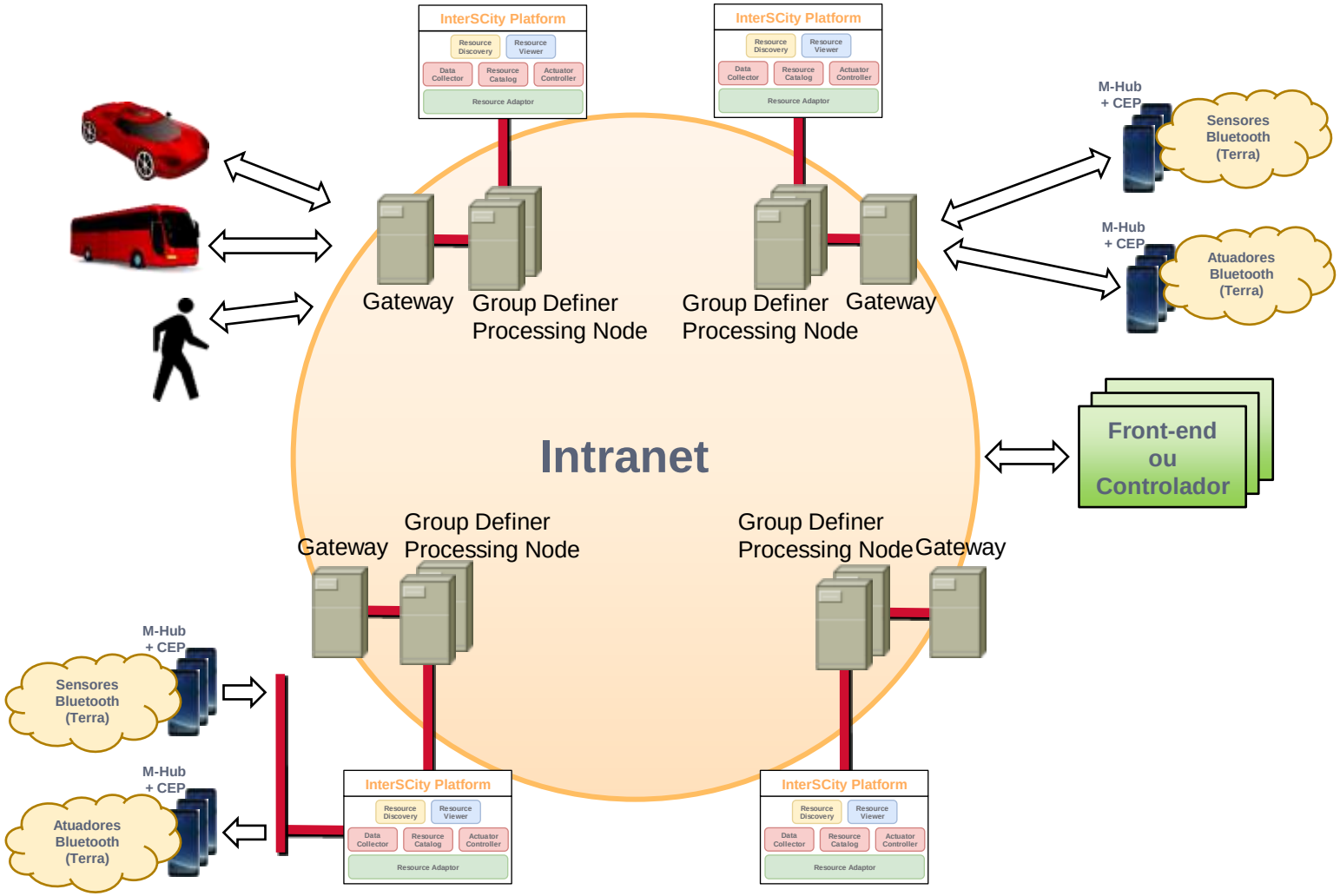
Exercício

- Exiba as informações do sensor tag.
- DESAFIO!!! Crie uma bússola.

ContextNet Distribuído

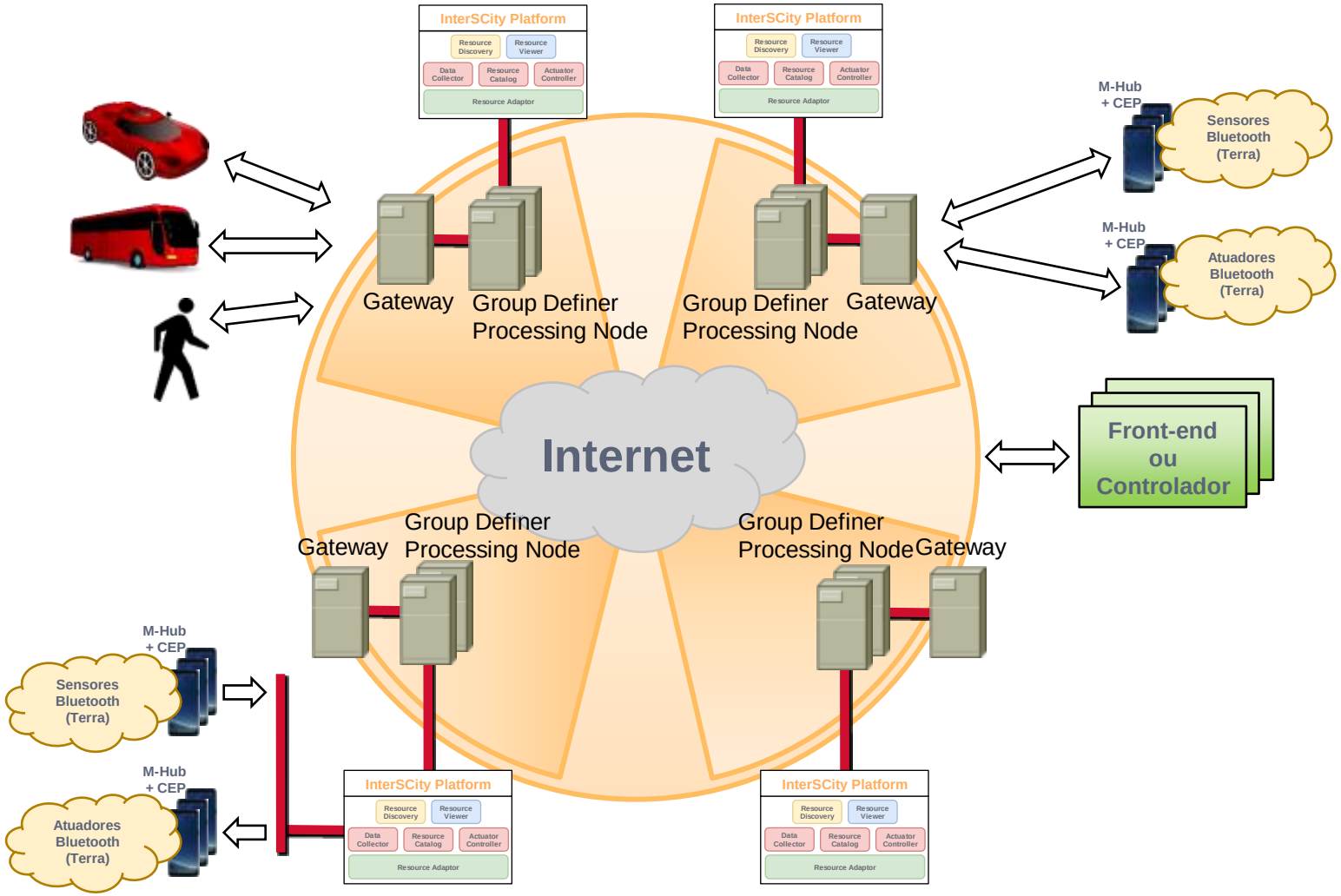
Até agora foi assim...

- Um único domínio de broadcast
- Mesma rede local (LAN)
- Descoberta automática de peers



Agora vai ser assim...

- Vários domínios de broadcast
- Rede de longa distância (WAN)
- Peers pré-configurados



O que muda para o programador?

- Nada
- Precisamos apenas reconfigurar o OpenSplice
  - Editar o arquivo ospl.xml

Editar o arquivo:

/opt/OpenSplice/HDE/x86_64.linux/etc/config/ospl.xml

```
<OpenSplice>
  . . .
  <DDSI2Service name="ddsi2">
    <General>
      <NetworkInterfaceAddress>
        172.16.1.202
      </NetworkInterfaceAddress>
      <AllowMulticast>false</AllowMulticast>
      <EnableMulticastLoopback>
        false
      </EnableMulticastLoopback>
      <CoexistWithNativeNetworking>
        false
      </CoexistWithNativeNetworking>
    </General>
    . . .
```

Endereço IP da interface local

```
<OpenSplice>
  . . .
  <DDSI2Service name="ddsi2">
    . . .
    <Discovery>
      <Peers>
        <Peer address="172.16.1.202"/>
        <Peer address="172.16.2.202"/>
        <Peer address="172.16.3.202"/>
        <Peer address="172.16.4.202"/>
      </Peers>
    </Discovery>
  </DDSI2Service>
</OpenSplice>
```

Endereço dos peers

Perguntas?

Projetos usando o MUSANet

Projetos

- Mobilidade
- Comunicação
- Sensores – Atuadores
- Usuários
- Grupos

Algumas sugestões

Comunicação por grupos

- Médicos/enfermeiros/técnicos/administrativos
- Professores/alunos graduação/alunos pós-graduação/por turma

Gerência de viajantes

- http://wiki.lac.inf.puc-rio.br/doku.php?id=travelersgroup

Ônibus chegando

- Aviso de que há algum ônibus chegando no ponto
- http://dadosabertos.rio.rj.gov.br/apiTransporte/apresentacao/rest/index.cfm/obterTodasPosicoes
- http://www.sptrans.com.br/desenvolvedores/api-do-olho-vivo-guia-de-referencia/documentacao-api/

Realidade aumentada (mas não muito...)

- Informações de contexto de acordo com a posição

Mais exemplos em:

- http://wiki.lac.inf.puc-rio.br/doku.php?id=classes

Perguntas?

Primeiros testes dos protótipos

Continuação dos testes

Validação e avaliação das aplicações desenvolvidas

Alexandre Meslin (Bolsista CAPES/BRASIL)
Noemi Rodriguez
Markus Endler (meslin, noemi, endler)@inf.puc-rio.br

MUSANet – Mobile Urban Sensor and Actuator Network