package main.java.com.wfdai.weatherforecastdai.main;

import main.java.com.wfdai.weatherforecastdai.main.KPI.KPI; import java.io.IOException; import javax.xml.bind.JAXBException; import main.java.com.wfdai.weatherforecastdai.main.KPI.RegistoKPI; import main.java.com.wfdai.weatherforecastdai.main.weather.Weather; import main.java.com.wfdai.weatherforecastdai.main.weather.WeatherFactory;

/** * Classe responsavel por instanciar o ETL @author daniel */ public class App {

```
String[] localidades = {"Vila Velha de Ródão,Castelo Branco,Portugal", "Mação ,Santarém,Port
Weather weather = new Weather();
WeatherFactory weatherFactory = new WeatherFactory();
Parser parser = new Parser();
Historico historico = new Historico();
Publisher publisher = new Publisher();
Alerta alerta = new Alerta();
KPI kpi = new KPI();
RegistoKPI registoKPI = new RegistoKPI();

public App() {

}

/**
 * Corre todas as localidades, recolhe os dados metereologicos, verifica se
 * existem alertas, e envia a mensagem para o Broker
 *
 * @throws javax.xml.bind.JAXBException
 * @throws java.io.IOException
 */
public void cicloLocalidades() throws JAXBException, IOException {
    for (String localidade : localidades) {
        weatherFactory.setWeather(localidade, weather);
        alerta.checkAlerta(weather, localidade);
        parser.setParser(weather);
        String mensagem = parser.getParsedMessage();
        publisher.publish(localidade, mensagem);
    }
}

/**
 * Corre todas as localidades, recolhe os dados metereologicos, guarda os
 * dados na BD, recolhe os dados da BD, e envia o Histórico atualizado para
 * o Broker
 *
```

```
 * @throws javax.xml.bind.JAXBException
 * @throws java.io.IOException
 */
public void cicloHistorico() throws JAXBException, IOException {
    for (String localidade : localidades) {
        weatherFactory.setWeather(localidade, weather);
        historico.putHistorico(weather, localidade);
        historico.getHistorico(localidade);
        parser.setParser(historico);
        String mensagem = parser.getParsedMessage();
        publisher.publish(localidade + "/historico", mensagem);
    }
}

/**
 * Recebe os KPI´s, regista-os na BD, recolhe os KPI´s da BD, e envia os
 * KPI's atualizados para o Broker
 *
 * @throws java.lang.InterruptedException
 */
public void cicloKpi() throws InterruptedException {
    kpi.getKPI();
    registoKPI.putKPI(kpi);
    registoKPI.getRegistoKPI();
    parser.setParser(registoKPI);
    String mensagem = parser.getParsedMessage();
    publisher.publish("/kpi", mensagem);
}
}
```