Progress Report

Title of the Bachelor's Thesis

Service meshes like Istio

submitted by

Akbota Aitbayeva

in partial fulfilment of the requirements for the degree of

Bachelor of Science (BSc)

Vienna, 08.08.2022

Degree programme code:	UA 033 521
Degree programme:	Bachelor's degree programme Computer Science
Supervisor:	Univ.-Prof. Dr.Uwe Zdun
Co-Supervisor:	Amine El Malki, BSc. MBA MSc.

Contents

1 Bachelor's Thesis Project: Simulated Internet of Things Cloud

1.1 System Description

Since the focus of the bachelor's thesis is on service meshes that enable service-to-service communication between microservices, the bachelor's thesis project is based on a set of **polyglot microservices** that are developed using different programming languages to gain additional functionality and efficiencies not available in a single language. Altogether, the microservices constitute a **distributed Internet of Things system**.

The Internet of things (IoT) is the term used to describe physical objects (”things”) that are equipped with sensors and other technology to connect them to other devices and systems over the Internet so that data can be exchanged between the objects.

As the use of a single IoT device provides only a limited view of a situation and therefore limited results, the project deploys multiple IoT devices, shares available sensor data, derives valuable insights by aggregating sensor values, applies predictive techniques, warns of unexpected behavior and evaluates data.

1.2 Technology Stack

Technologies used in developing the project:

- Java 11,
- Python 3.10,
- Apache Maven,
- Spring Boot,
- Flask,
- Project Lombok,
- Anomaly Detection Toolkit (ADTK),
- Darts,
- Prophet,
- Scikit-learn,
- PostgreSQL,
- Docker,
- Kubernetes,
- Istio.

1.3 System Architecture

Figure 1 illustrates the conceptual model of the system.

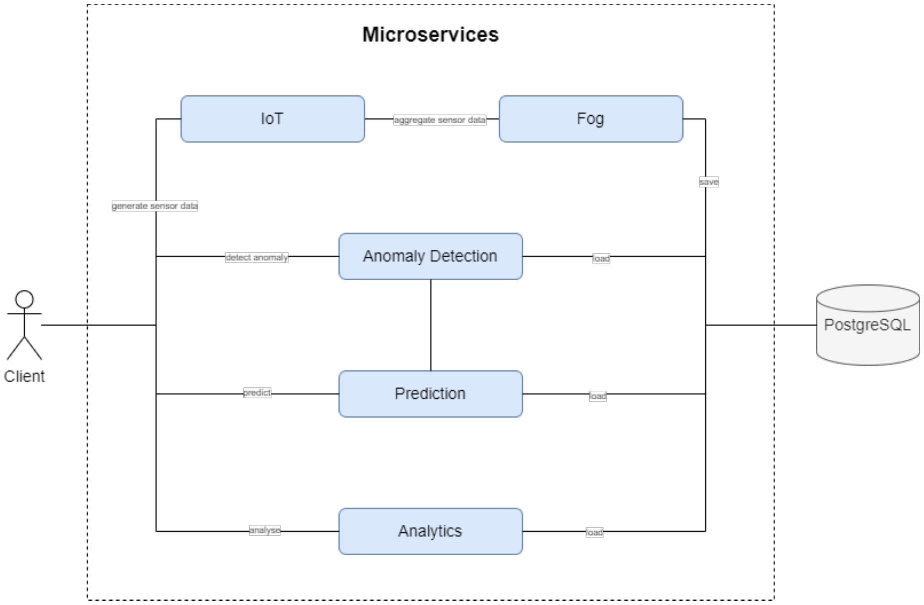

Figure 1: Simulated Internet of Things Cloud. System Architecture

1.4 Bird's-Eye View of the System

The project includes the following components.

1.4.1 Microservice IoT

The IoT microservice simulates two different IoT devices – temperature and pressure – by publishing sensor data. This is achieved by providing a single implementation and multiple instances (SIMI). The differences in the devices are configured inside the `application.properties` files and using REST API. According to a configured data type, an IoT device simulation strategy is injected to generate and publish a stream of simulated sensor data on an hourly basis. Standard sensor values, deviation from them, anomaly sensor values and their frequency are defined to simulate reasonable sensor data. The generated data has to be sent to the Fog.

1.4.2 Microservice Fog

The Fog microservice is used to avoid the high network load caused by the deployment of multiple IoT devices. Fog computing performs an aggregation of sensor values received from IoT devices before analyzing them. Each unique data type is handled by a different Fog instance. This is achieved by providing a single implementation and multiple instances (SIMI). The differences between the instances are specified inside the `application.properties` file. The configurable properties are aggregation mode, sensor data type and aggregation interval. Once the Fog starts collecting sensor data, Fog's aggregation strategy is injected, which is determined by the configured aggregation mode. The introduced aggregation operations are the calculation of average, maximum and minimum sensor values. After obtaining the sensor values over the entire aggregation interval, the sensor data is aggregated and stored in the database.

1.4.3 Microservice Anomaly Detection

The Anomaly Detection microservice warns of unexpected behavior and allows system monitoring to reduce machine downtimes. It can be used to detect anomalies in stored data (based on generalized ESD test or on user-given thresholds) or to clean outliers in data to make a better forecast.

1.4.4 Microservice Prediction

The Prediction microservice analyzes the data aggregated by the Fog and stored in the database and foresees upcoming behavior. Predictions can be made using one of the time series forecasting methods: Freedman Diaconis Estimator, Exponential Smoothing or Prophet. The last two methods are only available in the advanced version of Prediction.

1.4.5 Microservice Analytics

The Analytics microservice has an access to the database and provided analysis options on made predictions. It can measure the mean squared error (MSE) of the predicted values, report the most accurate prediction and examine predicted values against the expected thresholds.

1.4.6 Application Runner

The Runner is an application that runs data generation, simulates common client activity and handles request/response communication between microservices while serving the client.