Middleware SOMIOD

Service Oriented Middleware for Interoperability and Open Data

Developers

Gonçalo Francisco
ESTG
Institute of Polytechnic of Leiria
Leiria, Portugal
2201758@my.ipleiria.pt

Joel Bastos
ESTG
Institute of Polytechnic of Leiria
Leiria, Portugal
2191618@my.ipleiria.pt
joelsb2001@gmail.com

Pedro Alfaiate
ESTG
Institute of Polytechnic of Leiria
Leiria, Portugal
2202759@my.ipleiria.pt

Tiago Batista
ESTG
Institute of Polytechnic of Leiria
Leiria, Portugal
2191826@my.ipleiria.pt

Tiago Cabecinhas
Escola Superior de Tecnologia e Gestão
Instituto Politécnico de Leiria
Leiria, Portugal
2191187@my.ipleiria.pt

Advisers

Marisa Maximiano
ESTG
Instituto Politécnico de Leiria
Leiria, Portugal
marisa.maximiano@ipleiria.pt

Nuno Costa
ESTG
Instituto Politécnico de Leiria
Leiria, Portugal
nuno.costa@ipleiria.pt

Humberto Ferreira
ESTG
Instituto Politécnico de Leiria
Leiria, Portugal
humberto.ferreira@ipleiria.pt

Abstract

The Internet of Things (IoT) presents significant potential to address social challenges, but the emergence of the "SILO OF THINGS" highlights the difficulty of devices in sharing data. This work proposes a solution through the SOMIOD middleware, which standardizes the use of data, regardless of the application. The system architecture includes the SOMIOD REST API, applications 'A' and 'B', and the MQTT Message Broker to facilitate communication. The applications play distinct roles in the creation and manipulation of resources, while the SOMIOD REST API ensures consistency in data access. Efficient communication between applications, mediated by MQTT, contributes to overcoming the challenge of the "SILO OF THINGS". The standardization of SOMIOD simplifies IoT development, promoting more direct collaboration between systems. The success achieved suggests future expansion to support more devices and protocols, as well as exploring advanced security measures.

Keywords—XML, XSD, Middleware, RESTful API, IoT

I. Introduction

Despite the potential of the Internet of Things (IoT) to address certain social issues, this solution brings forth a challenge known as the "SILO OF THINGS," which entails different devices facing significant difficulties in sharing data. The objective of this work is to tackle this problem by employing middleware, which ensures that data is always utilized uniformly, regardless of the application. This work is being developed due to the way IoT systems operate, with each one having a distinct way of communicating with devices. Through this initiative, we aim to simplify IoT systems by utilizing common services and standards on the web, making data easily accessible.

.

II. System Architecture

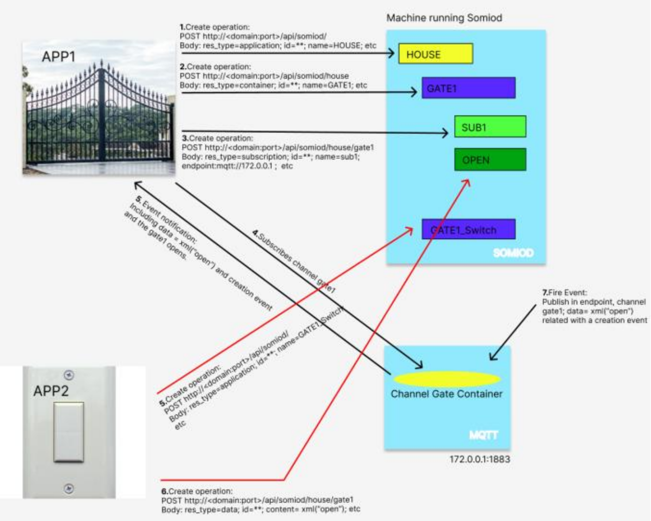
Machine running Somiod
APP1
1.Create operation:
POST http://<domain:port>/api/somiod/
Body: res_type=application; id=**; name=HOUSE; etc
HOUSE
2.Create operation:
POST http://<domain:port>/api/somiod/house
Body: res_type=container; id=**; name=GATE1; etc
GATE1
SUB1
OPEN
3.Create operation:
POST http://<domain:port>/api/somiod/house/gate1
Body: res_type=subscription; id=**; name=sub1;
endpoint:mqtt://172.0.0.1 ; etc
GATE1_Switch
SOMIOD
4.Subscribes channel gate1
5. Event notification:
Including data = xml("open") and creation event
and the gate1 opens.
7.Fire Event:
Publish in endpoint, channel
gate1; data= xml("open")
related with a creation event
APP2
5.Create operation:
POST http://<domain:port>/api/somiod/
Body: res_type=application; id=**; name=GATE1_Switch
etc
Channel Gate Container
MQTT
172.0.0.1:1883
6.Create operation:
POST http://<domain:port>/api/somiod/house/gate1
Body: res_type=data; id=**; content= xml("open"); etc

Fig. 1. Generic architecture of the sample testing application

A. *SOMIOD*

The SOMIOD REST API is the main element of this middleware. The purpose of this API is to respond to requests made by applications. SOMIOD supports GET, POST, PUT, and DELETE requests for the 'Application' and 'Container' resources, and it supports GET, POST, and DELETE requests for the 'Data' and 'Subscription' resources. As an API, SOMIOD is responsible for connecting the applications to the database.

B. *App 'A'*

This application, leveraging the SOMIOD API, is responsible for creating the resources 'application', 'container', and 'subscription'. Firstly, an 'application' resource is created; subsequently, a 'container' resource is created that is contained within the 'application', along with a 'subscription' resource that is associated with the 'container'. This subscription connects to a communication channel and is responsible for sending notifications whenever there is a change in the 'Data' resource. The resources are created according to the defined hierarchy, which is as follows:

Application (A):
        Container(1)
                Data (0..N)
                Subscriptions (0..N)

C. *App 'B'*

The app 'B' is responsible for the creation of the 'data' resource, which also involves the creation of the 'container' resource. The 'data' resource is the entity that stores the data records of the application. After each modification, the 'subscription' resource triggers a notification.

D. *Message Broker (MQTT)*

This component establishes communication between applications and is responsible for delivering the respective data to each application. It allows one application to subscribe to another, thus enabling communication between the two through shared data.

III. Integration/App Development

A. *Application A*

This test application has features that include the creation of resources such as applications, containers, and subscriptions. It also allows access to the list of available applications and their respective containers, as well as the ability to receive data through subscriptions in the channels related to an application's containers. Additionally, it presents a test case that simulates interaction between applications, acting as a gate that can be opened or closed upon receiving data of type 'open', which opens the gate, or 'close', which closes the gate, through the subscribed channel. This application represents only one of the many possible interactions between applications utilizing the SOMIOD service.

B. *Application B*

The functionalities of this application include the creation of containers, also allowing access to the list of applications and their respective modules. Furthermore, it presents a test case that simulates the interaction between applications, assuming the role of a command that controls the opening or closing of a gate by sending data of the type "open" (opens the gate) or "close" (closes the gate) through the subscribed channel. This application represents only one of the various possible interactions between applications that utilize the SOMIOD service.

IV. Conclusions and Future Work

The main objective of this work was to address the challenge of the "SILO OF THINGS" in the Internet of Things (IoT) through the implementation of the SOMIOD middleware. The specific objectives to achieve this goal were the creation of the SOMIOD REST API, the implementation of applications 'A' and 'B', and the use of the MQTT Message Broker to establish communication between the applications. Each application played a fundamental role in the creation and manipulation of resources such as applications, containers, subscriptions, and data. We concluded that the proposed objectives were successfully achieved. The SOMIOD REST API provides a consistent interface for the applications, ensuring uniformity in data access and manipulation, regardless of the application in question. Communication between the applications, mediated by the MQTT Message Broker, allows for an efficient exchange of information, contributing to the resolution of the "SILO OF THINGS" problem.

The standardization provided by SOMIOD simplifies the development of IoT applications, reducing the complexity in the integration of different devices. Moreover, effective communication between applications fosters more direct collaboration between systems. For future developments, we suggest expanding the middleware to include support for additional types of devices and communication protocols, further enhancing the system's scope and flexibility. Additionally, we consider it pertinent to explore advanced security mechanisms to protect the integrity and confidentiality of data transmitted between applications.

Appendix

Appendix A

Application resource

GET http://<domain:port>/api/somiod

POST http://<domain:port>/api/somiod

- Example of the body content (xml):

```
<application>
   <name>gate</name>
  </application>
```

PUT http://<domain:port>/api/lighting

- Example of the body content (xml):

```
<application>
   <name>switch</name>
  </application>
```

DELETE http://<domain:port>/api/somiod/gate

<u>Container resource</u>

GET http://<domain:port>/api/somiod/gate

POST http://<domain:port>/api/somiod/gate

- Example of the body content (xml):

```
<container>
   <name>door</name>
</container>
```

PUT http://<domain:port>/api/somiod/gate/door

- Example of the body content (xml):

```
<container>
   <name>garage_door</name>
</container>
```

DELETE http://<domain:port>/api/somiod/gate/door

<u>Data resource</u>

GET http://<domain:port>/api/somiod/gate/door/data

POST http://<domain:port>/api/somiod/gate/door

- Example of the body content (xml):

```
<data>
   <content>Open</ content >
</data>
```

DELETE http://<domain:port>/api/somiod/gate/door/on

<u>Subscription resource</u>

GET http://<domain:port>/api/somiod/gate/door/sub1

POST http://<domain:port>/api/somiod/gate/door

- Exemplo do conteúdo do body (xml):

```
<subscription>
      <name>Sub1</name>
      <endpoint>mqtt://127.0.0.1</endpoint>
      <eventType>creation</eventType>
</subscription>
```

DELETE http://<domain:port>/api/somiod/gate/door/sub1

Appendix B

All members of the group contributed to the development of some part of the API. The project was initiated entirely by member **Joel Bastos**, who laid the foundation and developed the initial core of the system on his own. Building upon this starting point, member **Gonçalo Francisco** developed most of the code for one of the applications, which was then improved by **Tiago Batista**, who fixed bugs from the first application and developed the second one. There was also some input from the other members of the group. The report was prepared by student **Pedro Alfaiate**, with assistance from the rest of the group in clarifying doubts and creating the architectural diagram.

This project was developed within the scope of the **Systems Integration** course.

As part of the **bachelor's degree in computer engineering** at Polytechnic of Leiria in Portugal.

During the **2023/2024** academic year.