# Flex4Apps

## Platform for Application and Infrastructure Flexibility in Cyber-Physical Systems

## DELIVERABLE D3.5

## APIs and programming guide for Flex4Apps platform

| | |
|---|---|
| Document identifier: | **Flex4Apps-WP3-D3.5** |
| Work Package: | **WP3** |
| Lead Partner: | **Fraunhofer** |
| Document Status: | **Final** |
| Dissemination Level: | **Public** |

# TABLE OF CONTENTS

# 1 INTRODUCTION

This document briefly introduces various components of the Flex4Apps toolbox and includes links to more detailed information.

# 2 CLOUD-BASED OPEN SOURCE INFRASTRUCTURE

The Flex4Apps Github repository (https://github.com/Flex4Apps/flex4apps) contains instructions and code on how to quickly set up a cloud-based open source infreastructure for your company. Containerization with Docker plays a key role here.

Documentation, including best practices on various Flex4Apps-related topics, can be found at https://f4a.readthedocs.io.

# 3 LOG ANALYSIS LIBRARY

In today's software industry, monolithic architectures are more and more replaced by smaller, distributed services. Paradigms such as cloud computing and microservices are central to this development.

While small programs providing one or few well defined services are easy to understand in isolation, real-life systems frequently comprise a multitude of such programs, resulting in a high level of complexity and possibly unforeseen problems. A hard part of fixing issues in such a system consists of reconstructing the exact interactions and conditions under which the issue occurred. Logging often is the only way by which this information can be made available. But log files do not only provide crucial information for problem-solving in distributed systems; they additionally are a data source from which valuable business insights can be obtained.

Modern large-scale web services produce vast amounts of logs, which makes manual inspection and analysis unfeasible. Automated analysis techniques such as machine laerning can be applied to solve this problem. Cheap storage to save the logs and cheap computing power to analyse them are both available.

Within the scope of Flex4Apps, Fraunhofer SCAI developed a software library for analysis of and anomaly detection in unstructured log files. An essential feature of this library is the ability to convert unstructured into structured log files. For this purpose, filters for common parameter types and a component to detect remaining parameters by machine learning are included. The anomaly detection component of the library uses structured log files as an input to calculate an anomaly score on a per line or per file basis.

For a detailed description of the log analysis library, see https://github.com/Flex4Apps/flex4apps/blob/master/docs/chapter02_stateOfTheArt/LogAno_manual.pdf (you may need to open the PDF file in an external viewer to be properly displayed). Best practices on how to format logs are available at https://f4a.readthedocs.io/en/latest/chapter02_stateOfTheArt/_structure.html#log-file-analysis.

# 4 MONITORING OF IOT DEVICES: MOSQUITTO

One main aspect Flex4Apps fulfills is its applicability to existing and future IoT solutions and comparable cyber-physical systems. According to its toolbox-philosophy, Flex4Apps doesn't use a proprietary software and/or hardware to reach this goal, but rather uses and adapts existing tools.

When taking a closer look into the world of IoT one will very soon get in touch with MQTT. MQTT is a lightweight protocol that has been widely adopted to IoT applications in the recent years. This way it has become a quasi-standard for the internet of things. MQTT is easy to understand for developers as well as for customers. It implements a simple message distribution algorithm that can evolve while an IoT system grows and – one of the most important aspects – it doesn't expect all network elements to be connected and fully operational all the time.

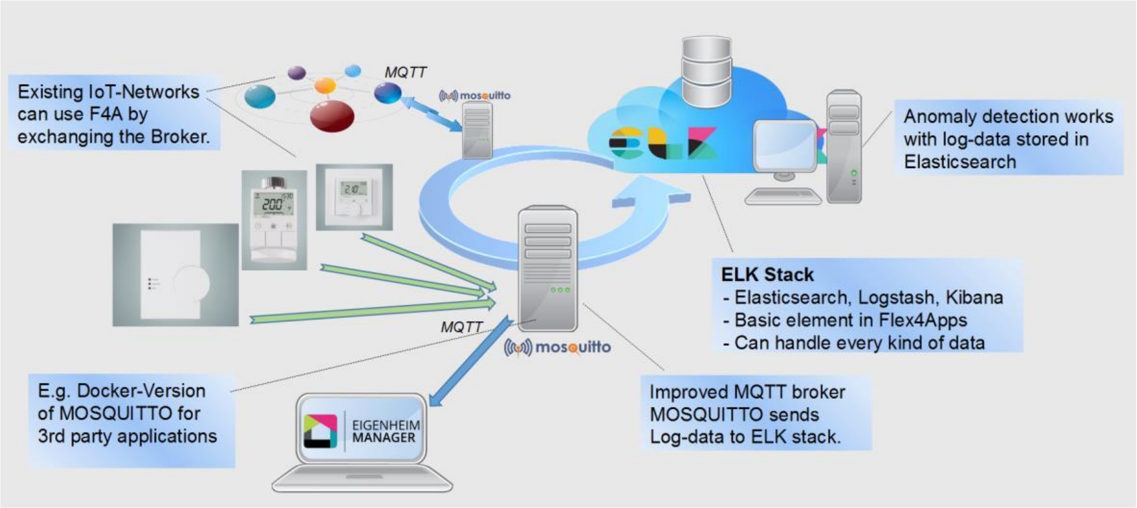
However, this separation of connection-related, lower-layer functionality from application based, upper-layer information distribution makes it challenging to survey the whole network behavior in terms of failure resiliency and run-time stability. Furthermore, MQTT is not very well equipped with security features, which makes it susceptible to intrusion attacks and misconfiguration.

Using anomaly-detection methods offered with Flex4Apps can significantly improve existing and new installed IoT systems based on MQTT. In order to adopt Flex4Apps tools with that kind of IoT networks we need to build a bridge which allows log data to flow from an IoT system to the cloud-based Flex4Apps environment. Here MOSQUITTO comes in.

MOSQUITTO is an open-source MQTT broker software that serves as the central network entity in MQTT networks. It handles connection establishment and connection releases, receives messages from publishers and distributes it over the network to subscribers. This makes it an ideal “probe” to extract log information.

To let MOSQUITTO deliver log information which can be fed to Elasticsearch we modified an existing release of MOSQUITTO and implemented a method to push logs to a network-based receiver.

The figure below shows a possible infrastructure that can be built with the modified broker software:


Existing IoT-Networks can use F4A by exchanging the Broker.
MQTT
mosquitto
Anomaly detection works with log-data stored in Elasticsearch
ELK
ELK Stack
- Elasticsearch, Logstash, Kibana
- Basic element in Flex4Apps
- Can handle every kind of data
MQTT
mosquitto
E.g. Docker-Version of MOSQUITTO for 3rd party applications
EIGENHEIM MANAGER
Improved MQTT broker MOSQUITTO sends Log-data to ELK stack.

To insert log data into ElasticStack (ELK Stack) there exists an interface utility named Logstash. Logstash can be accessed via TCP and it best accepts JSON formatted information. There is no restriction on structure and size of this information. That's why MOSQUITTO converts its log data into JSON format.

Here is one example of a log message:

```
{"mqtt":
        {"topic":"topic",
        "clientIP": "192.168.23.45",
        "port": 1883,
        "clientID":"mosqpub|5704-t550",
        "tsUTC": "2019-04-29T09:30:00Z (GMT)",
        "tsLocal": "2019-04-29T09:30:00+0000 (UTC)",
        "payload": "bXNn"}
}
```

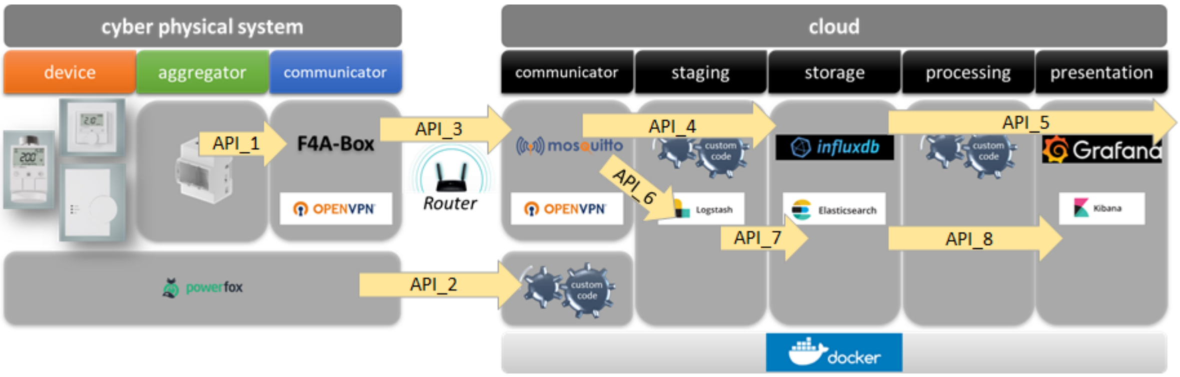
MOSQUITTO can run in a cloud-based environment like Docker or Kubernetes. It also can be installed on local servers or even barebone systems like IPC or dedicated gateways. This makes it a well-suited entry point for IoT systems to Flex4Apps.

For more technical information on how to install and configure MOSQUITTO see https://github.com/Flex4Apps/mosquitto.

# 5 API DOCUMENTATION FOR HOME AUTOMATION USE CASE

For the home-automation use case, at least one home automation system as input and one third party platform as output is needed. Tested input systems are Homematic, provedo, and powerfox. Eigenheim-Manager was tested as one output platform. The software stack provided by the Flex4Apps toolbox is used to transport sensor data and meta data.

The following APIs are covered by the home-automation use case (evermind GmbH):



## 5.1 API_1

- XML RPC for HomeMatic:
  - https://github.com/hobbyquaker/XML-API/releases/tag/1.16
  - https://www.eq-3.de/Downloads/eq3/download%20bereich/hm_web_ui_doku/HM_XmlRpc_API.pdf

- F4A Gateway collects meter data from various sensors via vendor specific interface.
- Gateway pulls data in intervals.

## 5.2 API_2

- Non public REST-API:
  - http://www.powerfox.energy/
  - https://development.powerfox.energy/

- If data is available online, no gateway is needed. A small piece of vendor-specific code can connect various data sources directly to the data storage of the Flex4Apps toolbox.
- Powerfox API stands as an example for various online data sources.

## 5.3 API_3

- https://www.thethingsnetwork.org/docs/applications/mqtt/api.html

- For physical devices data transfer via MQTT protocol is perferred. In conjunction with a message broker (Mosquitto) a simple, lightweight and resilient solution can be established.
- This protocol is vendor agnostic so various devices can be connected and used with the toolbox.
- Messages are organized in topics.
- The toolbox expects one topic per device.

## 5.4 API_4

- https://www.thethingsnetwork.org/docs/applications/mqtt/api.html
- https://docs.influxdata.com/influxdb/v1.7/tools/api/

### Store Payload

- Connector code from toolbox normalizes data from message broker and stores it.
- Connector code uses standard APIs: MQTT for receiving and Influx for sending.

## 5.5 API_5

- https://docs.influxdata.com/influxdb/v1.7/tools/api/
- Grafana has access to influx data via a standard data source driver. This allows inspection of data independent from a specific domain model.
- For third-party application a specific API exists to aggregate and visualize data series.
- Moreover some administrative functions covered by API like register users for data access, authenticate and connect users to time series and so one.

## 5.6 API_6

- Watch and analyse transport meta data.
    - Mosquitto enhanced by Flex4Apps members sends some trace messages to network port
        - Direct network connection leads to small runtime footprint
    - Logstash gathers these messages from TCP
        - https://www.elastic.co/guide/en/logstash/current/plugins-inputs-tcp.html

## 5.7 API_7

- Logstash pushes data to elastic search. Due both products are from same vendor, no custom code is needed. Logstash provides functionality to push these data direct into Elasticsearch storage.

## 5.8 API_8

- https://www.elastic.co/guide/en/elasticsearch/reference/current/docs.html
- With Kibana user can inspect data stored in Elasticsearch. Both products are well integrated.

- User has to use Elasticsearch query API for discovering and visualize data. Kibana helps more or less.
- https://www.elastic.co/guide/en/elasticsearch/reference/current/docs-get.html
- Machine learning APIs