

DCF

Data Collection Framework

User Manual



Contents

INTRODUCTION	3
DCF ARCHITECTURE	4
DOWNLOADS ADDITIONAL FILES AND INSTALLATION	5
USER INTERFACE	6
Information/historical data dashboard	6
Event processing interface	7
Creating new operation	8
Modifying existing operation	11
Deleting existing operation	13
Input parameters interface	14
Defining new input variables.....	14
Modify existing input	16
Delete existing input.....	17
Legacy data interface.....	18
Database entry section	18
MongoDB database interface	25
Note on Legacy and Mongo DB database interface:	28

INTRODUCTION

The purpose of this manual is to introduce the architecture of DCF as well as steps to install DCF and test the functionalities, the manual also guides the users on how to navigate the web GUI and correctly define the input accordingly to the data formats.

Requirement:

- PC
- Docker
- DCF image
- Web browser (Edge, Firefox, etc)

For testing (optional):

- Either one of: Postman or programming IDE for performing CRUD operations
- Scripts for importing data from csv files to MongoDB for testing

DCF ARCHITECTURE

DCF role is to collect data from shopfloor through data adapter for example MQTT, OPC-UA, Fiware/orion or legacy system and data stored in legacy systems and database for example mongodb, after that the data is transmitted through Fiware/orion.

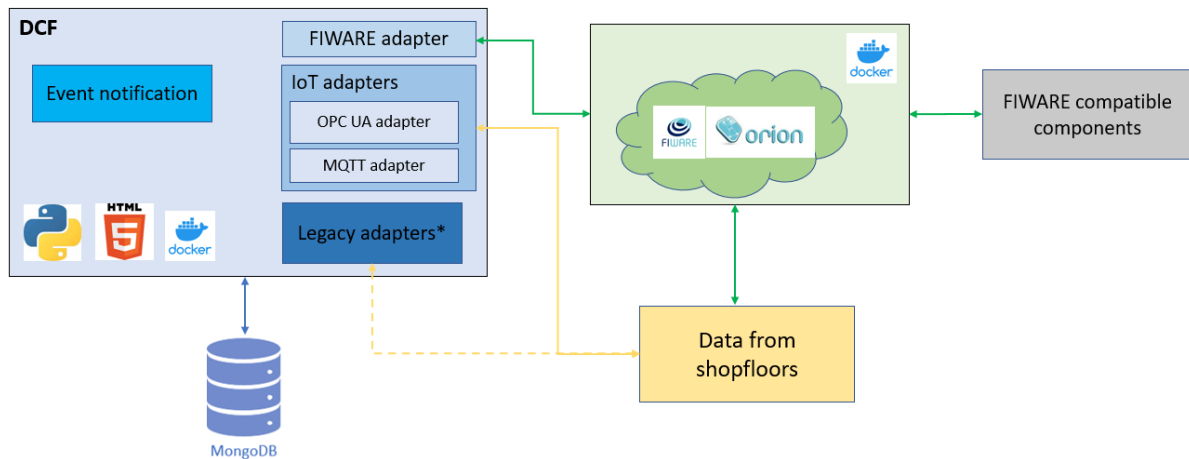


Figure 1: DCF architecture

To interact with other components, FIWARE is needed, although, DCF can be configured to communicated by using other brokers/adapters for example OPC-UA and MQTT as well.

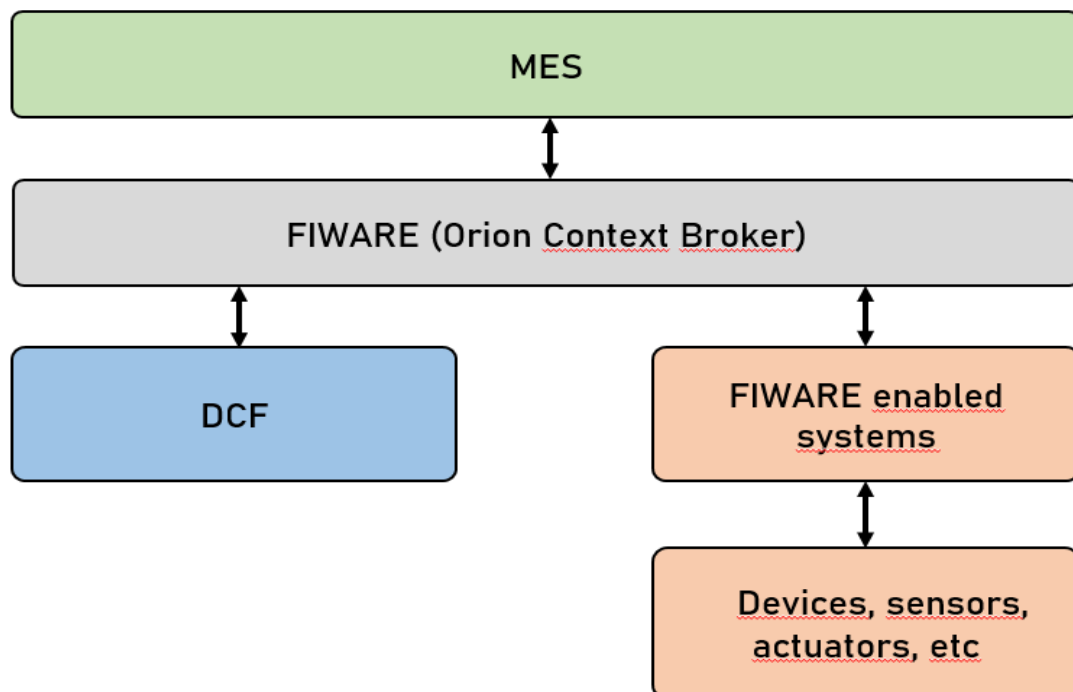
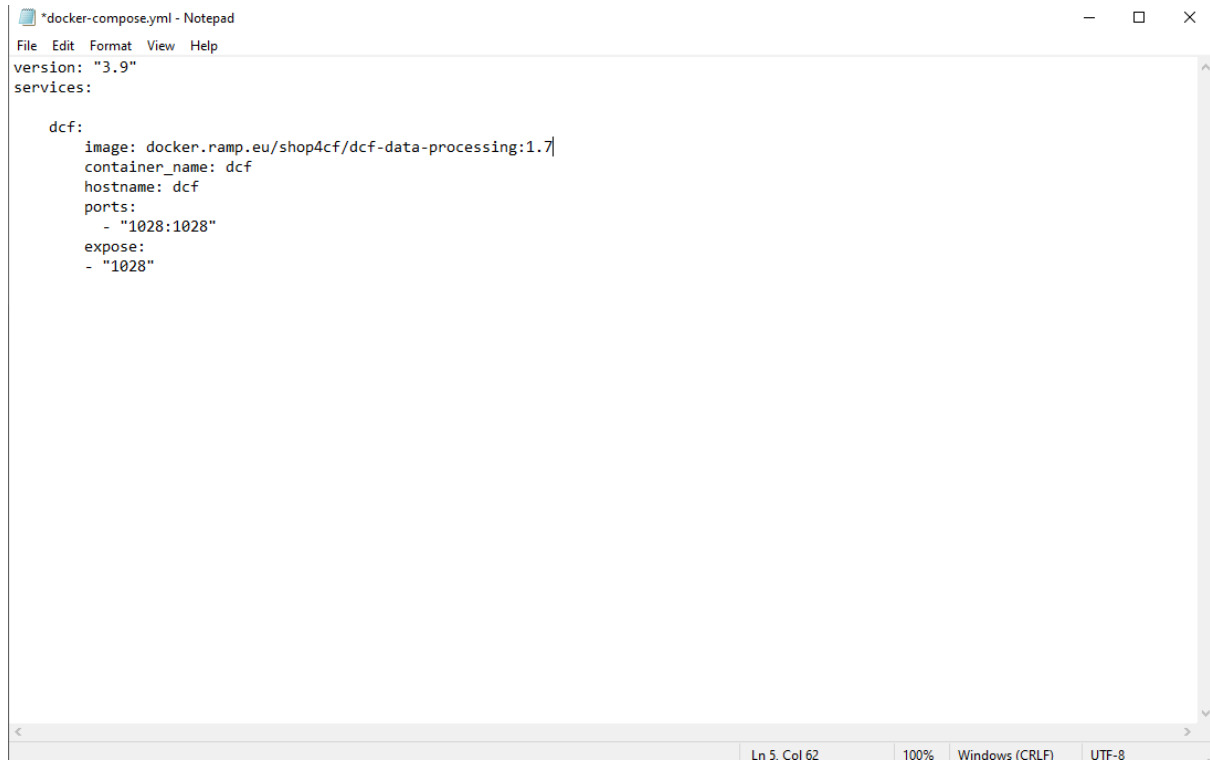


Figure 2: DCF communication with other components

DOWNLOADS ADDITIONAL FILES AND INSTALLATION

Beside DCF image, only docker-compose file is needed to compose the image into container. To get the docker-compose.yml file, download the file from <https://github.com/TAU-FASTLab/DCF>. Version of DCF can be changed to compose the suitable version. The latest version is 1.7.

A screenshot of a Notepad window titled "*docker-compose.yml - Notepad". The window contains the following YAML configuration for a Docker Compose service named 'dcf':

```
version: "3.9"
services:
  dcf:
    image: docker.ramp.eu/shop4cf/dcf-data-processing:1.7
    container_name: dcf
    hostname: dcf
    ports:
      - "1028:1028"
    expose:
      - "1028"
```

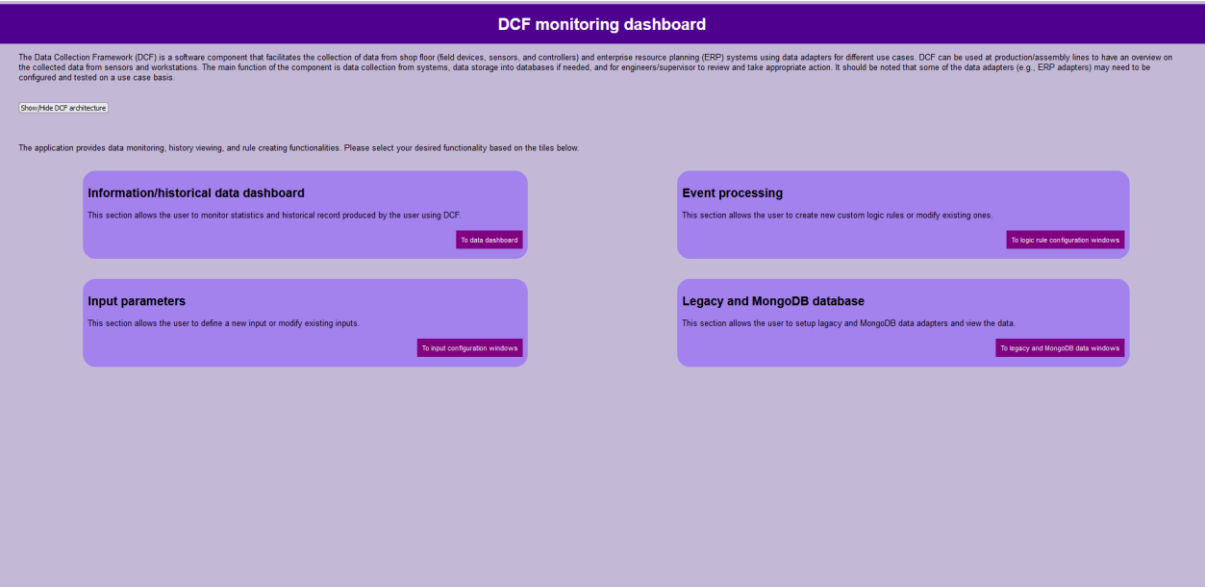
The status bar at the bottom indicates "Ln 5, Col 62", "100%", "Windows (CRLF)", and "UTF-8".

If users use dockerized fiware and MongoDB server or other applications, when referring to these servers for data retrieving and publishing, the host name will be "host.docker.internal" instead of "localhost", the ports will be the same ports of these applications.

USER INTERFACE

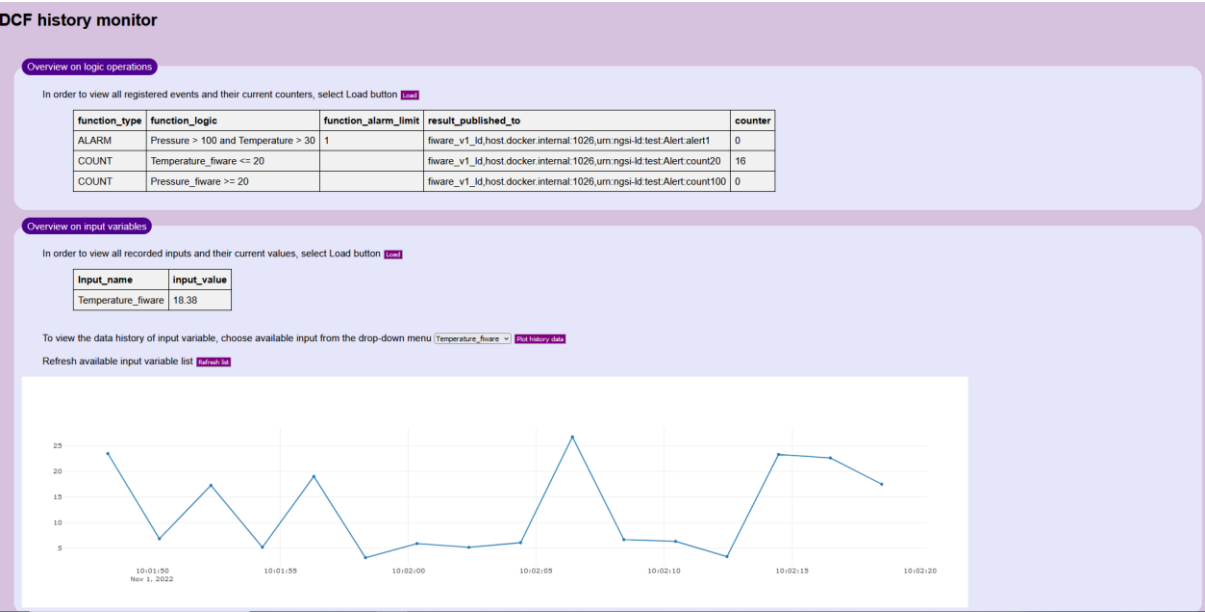
DCF also provides a GUI where users can monitor events and define new operations.

First, the dashboard is used as navigation panel to different functionalities of DCF



Information/historical data dashboard

This dashboard allows users to monitor output counters of defined logic operations and their parameters such as logic conditions and output publish endpoint, as well as recorded input variable data which can be turned into a graph. The UI also has experimental feature to transfer the data from OPC-UA and MQTT to Fiware v1 NGSI-LD, documentation for the function will be completed once the feature is fully developed, but for now, the feature can be seen on the UI



Event processing interface

This interface allows users to define new logic operations or modify existing operations.

Back to DCF monitoring dashboard

Event definition and processing

This section allows the creation of custom logical operations for event notification. The "Overview on logic operations" section provides a more detailed overview on the saved operations. The "Operation definition" section provides the ability to create new functions or edit existing ones. Please note that: New variables for the use in operations definition, should be defined in the "variable parameters" window.

Overview on logic operations

Predefined operations

In order to view the predefined operations and specific parameters click the load button. [Load](#)

Id	Function_type	Logic_condition	Alarm_limit_count	Result_published_to	Close
1	ALARM	Pressure > 100 and Temperature > 30	1	fware_v1_id,host,docker:internal:1026,urn:ngsi-ld:test:Alert:alert1	
2	COUNT	Temperature_fware <= 20		fware_v1_id,host,docker:internal:1026,urn:ngsi-ld:test:Alert:count20	
3	COUNT	Pressure_fware >= 20		fware_v1_id,host,docker:internal:1026,urn:ngsi-ld:test:Alert:count100	

To assist users in these tasks, list of defined variables is also provided in the interface.

Available inputs

In order to view the list of available inputs click the load button. [Load](#)

Id	Name	Topic to data cluster/ Fware id/ Opc-ua data node	Path to variable data	Server	Close
1	Temperature	/tuanvutest/Temperature	Temperature	broker.hivemq.com	
2	Pressure	/tuanvutest/Pressure	Pressure	broker.hivemq.com	
3	Temperature_emx	/tuanvutest/Temperature	Temperature_emx	broker.emqx.io	
4	Pressure_emx	/tuanvutest/Pressure	Pressure_emx	broker.emqx.io	
5	Moisture	/tuanvutest/Moisture	Moisture	broker.hivemq.com	
6	Temperature_opc	ns=2;i=3		opc.tcp://127.0.0.1:12345	
7	Pressure_opc	ns=2;i=2		opc.tcp://127.0.0.1:12345	
8	Temperature_fware	urn:ngsi-ld:test:Device:Temperature	value/value	host.docker:internal:1026	
9	Pressure_fware	urn:ngsi-ld:test:Device:Pressure	value/value	host.docker:internal:1026	
10	Moisture_emx	/tuanvutest/Moisture	Moisture_emx	broker.emqx.io	
11	Moisture_opc	ns=2;i=4		opc.tcp://127.0.0.1:12345	
12	datapoint1	test_entity	value/value	host.docker:internal:1026	

To create/modify/delete operation, select the dropdown menu and select one of the functionalities

5	ALARM	Temperature < 1000	mqtt.broker.hivemq.com/tuanvutest/alarmlu
6	COUNT	Pressure_emx > 10	mqtt.broker.hivemq.com/tuanvutest/count12
7	COUNT	Moisture_emx > 10 and Moisture > 10	mqtt.broker.hivemq.com/tuanvutest/count372
8	COUNT	Temperature_fiware <= 20	fiware.orion:1026 urn:ngsi-id:test:Alert:count20
9	COUNT	Pressure_fiware >= 20	fiware.orion:1026 urn:ngsi-id:test:Alert:count100
10	COUNT	Pressure > 100	mqtt.broker.hivemq.com/tuanvutest/counter10000

Available inputs

In order to view the list of available inputs click the load button. [Load](#)

Operation definition

In order to create new functions or edit existing ones, please select one of the following options:

CHOOSE AN OPTION

CHOOSE AN OPTION

NEW_FUNCTION

CHANGE_EXISTING_FUNCTION

DELETE_FUNCTION

Creating new operation

Operation definition

In order to create new functions or edit existing ones, please select one of the following options:

NEW_FUNCTION

All operation parameters can be defined here. Users can either manually create your desired functions directly in the input area below or use the following input methods to define their functions.

- Choose an operation type from the list **COUNT**
- Endpoints where results are published to:

Add new endpoint [Add endpoint](#)

Endpoint NO.	Endpoint type	Endpoint server	Endpoint topicid
1	Fiware/Orion-id v1		

- Choose a desired variable from the list of defined variables **Temperature** [Insert variable](#)
- Users can input numeric data, use dot to separate decimals [Insert number](#)
- Select the desired logic operators from the list below.

() < > >= <= != and or ** * + / not

True

[Save](#) [Cancel all changes](#)

In this section, users can choose which type of operation, ALARM type generate an output only when the counter of number of times the logic condition has been triggered exceeds or reaches the Alarm count limit parameter, while COUNT type constantly presents how many times the logic condition has been triggered.

Section 2 is used to define the endpoints where processed results are published to

Section 5 is the place where logic arguments are created, the default condition for new operation is always “True”.

There are list of tools are created to assist users to create the logic condition, the list of defined variables dropdown menu contains all defined variables by users. After users has chosen a variable or given a numeric input (with “.” as decimal delimiter and no thousand separators), “insert variable”/ “insert number” need to be clicked respectively to insert the inputs. Users can also choose logic operators from the logic operator list. It is highly recommended to use these features to ensure

the data format is followed so the program can parse the data. To finalize, the “confirm logic change” button is also needed to be chosen.

For example, the logic condition “Temperature_fiware > 5 and Pressure_fiware <= 30” is defined.

To publish the data to MQTT or Fiware/orion-ld, users need to use section 2 to define the endpoints, topic/id is recommended to be unique, endpoint type is one of “Fiware/Orion-ld v1” or “mqtt” or “Fiware v2”. In this case, the result needs to be publish to Fiware/Orion-ld v1 with the entity id “urn:ngsi-ld:test:Alert:count5”, on my containerized fiware/orion-ld server so the server should be host.docker.internal:1026

Operation definition

In order to create new functions or edit existing ones, please select one of the following options:

NEW_FUNCTION

All operation parameters can be defined here. Users can either manually create your desired functions directly in the input area below or use the following input methods to define their functions.

1. Choose an operation type from the list

COUNT

2. Endpoints where results are published to:

Add new endpoint

Endpoint NO.	Endpoint type	Endpoint server	Endpoint topicId
1	Fiware/Orion-ld v1	host.docker.internal:1026	urn:ngsi-ld:test:Alert:count5

3. Choose a desired variable from the list of defined variables

Temperature

Insert variable

4. Users can input numeric data, use dot to separate decimals

Insert number

5. Select the desired logic operators from the list below.

() < > >= <= != and or ** * - + / not

Temperature_fiware > 5 and Pressure_fiware <= 30

Confirm

Save

Cancel all changes

Push “Save” button to update the data.

Id	Function_type	Logic_condition	Alarm_limit_count	Result_published_to	
1	ALARM	Pressure > 100 and Temperature > 30	1	fiware_v1_ld,host.docker.internal:1026,urn:ngsi-ld:test:Alert:alert1	
2	COUNT	Temperature_fiware <= 20		fiware_v1_ld,host.docker.internal:1026,urn:ngsi-ld:test:Alert:count20	
3	COUNT	Pressure_fiware >= 20		fiware_v1_ld,host.docker.internal:1026,urn:ngsi-ld:test:Alert:count100	
4	COUNT	Temperature_fiware > 5 and Pressure_fiware <= 30		fiware_v1_ld,host.docker.internal:1026,urn:ngsi-ld:test:Alert:count5	

To check whether the operation is operational, head to either the monitor dashboard or entity of the Alert in fiware/orion server using internet browsers or postman

Overview on logic operations

In order to view all registered events and their current counters, select Load button

Load

function_type	function_logic	function_alarm_limit	result_published_to	counter
ALARM	Pressure > 100 and Temperature > 30	1	fiware_v1_ld,host.docker.internal:1026,urn:ngsi-ld:test:Alert:alert1	0
COUNT	Temperature_fiware <= 20		fiware_v1_ld,host.docker.internal:1026,urn:ngsi-ld:test:Alert:count20	310
COUNT	Pressure_fiware >= 20		fiware_v1_ld,host.docker.internal:1026,urn:ngsi-ld:test:Alert:count100	351
COUNT	Temperature_fiware > 5 and Pressure_fiware <= 30		fiware_v1_ld,host.docker.internal:1026,urn:ngsi-ld:test:Alert:count5	2

DCF manual

JSON	Raw Data	Headers
Save	Copy	Collapse All Expand All Filter JSON
id:		"urn:ngsi-ld:test:Alert:count5"
type:		"https://url.fiware.org/ns/data-models#Alert"
▼ https://smart-data-models.github.io/data-models/terms.jsonld#/definitions/category:		
type:		"Property"
value:		"Alert"
▼ https://smart-data-models.github.io/data-models/terms.jsonld#/definitions/validTo:		
type:		"Property"
▼ value:		
@type:		"DateTime"
@value:		"2022-11-01T10:18:25.67Z"
▼ value:		
type:		"Property"
value:		4
observedAt:		"2022-11-01T10:18:25.670Z"
▼ description:		
type:		"Property"
▼ value:		
function_type:		"COUNT"
operation logic argument:		"COUNT;Temperature_fiware > 5 and Pressure_fiware <= 30;;fiware_v1_ld,host.docker.internal:1026,urn:ngsi-ld:test:Alert:count5"
▼ https://smart-data-models.github.io/data-models/terms.jsonld#/definitions/dateIssued:		
type:		"Property"
▼ value:		
@type:		"DateTime"
@value:		"2022-11-01T10:18:25.67Z"
▼ https://smart-data-models.github.io/data-models/terms.jsonld#/definitions/alertSource:		
type:		"Relationship"
object:		"urn:ngsi-ld:dcf-logic-engine"
▼ https://smart-data-models.github.io/data-models/terms.jsonld#/definitions/validFrom:		
type:		"Property"
▼ value:		
@type:		"DateTime"
@value:		"2022-11-01T10:18:25.67Z"
▼ https://smart-data-models.github.io/data-models/terms.jsonld#/definitions/severity:		
type:		"Property"
value:		"high"
▼ humanVerified:		
type:		"Property"
value:		"false"

The operation is registered and output is published. Note: data shown on the 2 screenshots are made in different time mark, thus the different in the data.

Modifying existing operation

To modifying existing operation, select “CHANGE_EXISTING_FUNCTION” option

Operation definition

In order to create new functions or edit existing ones, please select one of the following options:

CHANGE_EXISTING_FUNCTION ▾

Id of function needs changing:

Fetch the operation parameters

From here, users need to provide the id of the operation that needs to be modified, for example, the operation last created with id 4 needs to add new logic condition, after providing the id, “Fetch the operation parameters” button need to be clicked to retrieve the operation parameter

Operation definition

In order to create new functions or edit existing ones, please select one of the following options:

CHANGE_EXISTING_FUNCTION ▾

Id of function needs changing: 4

Fetch the operation parameters

All operation parameters can be defined here. Users can either manually create your desired functions directly in the input area below or use the following input methods to define their functions.

1. Choose an operation type from the list
2. Endpoints where results are published to:

Add new endpoint **Add endpoint**

Endpoint NO.	Endpoint type	Endpoint server	Endpoint topic/id
1	<input type="text" value="Fiware/Orion-Id v1"/>	<input type="text" value="host.docker.internal:1026"/>	<input type="text" value="urn:mgsa-Id:test:Alert:coul"/> <input type="button" value="X"/>

3. Choose a desired variable from the list of defined variables
4. Users can input numeric data, use dot to separate decimals
5. Select the desired logic operators from the list below.

() < > == >= <= != and or ** * - + / not

Save **Cancel all changes**

New condition is added

In order to create new functions or edit existing ones, please select one of the following options:

CHANGE_EXISTING_FUNCTION ▾

Id of function needs changing:

Fetch the operation parameters

All operation parameters can be defined here. Users can either manually create your desired functions directly in the input area below or use the following input methods to define their functions.

1. Choose an operation type from the list COUNT ▾

2. Endpoints where results are published to:

Add new endpoint **Add endpoint**

Endpoint NO.	Endpoint type	Endpoint server	Endpoint topic/id
1	Fiware/Orion-Io v1 ▾	host.docker.internal:1026	urn:ngsi-ld:test:Alert:count ✕

3. Choose a desired variable from the list of defined variables Temperature ▾ Insert variable

4. Users can input numeric data, use dot to separate decimals Insert number

5. Select the desired logic operators from the list below.

() < > == >= <= != and or ** * - + / not

Temperature_fiware > 5 and Pressure_fiware <= 30 and Pressure_fiware > 0

✕

Save

Cancel all changes

After modifying the logic condition, the change can be saved or cancelled. The new operation saved will inherit the counter data from the original function.

Overview on logic operations

In order to view all registered events and their current counters, select Load button Load

function_type	function_logic	function_alarm_limit	result_published_to	counter
ALARM	Pressure > 100 and Temperature > 30	1	fiware_v1_id,host.docker.internal: 1026,urn:ngsi-ld:test:Alert:alert1	0
COUNT	Temperature_fiware <= 20		fiware_v1_id,host.docker.internal: 1026,urn:ngsi-ld:test:Alert:count20	443
COUNT	Pressure_fiware >= 20		fiware_v1_id,host.docker.internal: 1026,urn:ngsi-ld:test:Alert:count100	529
COUNT	Temperature_fiware > 5 and Pressure_fiware <= 30 and Pressure_fiware > 0		fiware_v1_id,host.docker.internal: 1026,urn:ngsi-ld:test:Alert:count5	29

Deleting existing operation

To delete existing function, choose “DELETE_FUNCTION” option

In this section, users need to provide the id of the operation that needs to be deleted, for example in this case operation with id 1 with logic condition “Pressure > 100 and Temperature > 30”

Operation definition

In order to create new functions or edit existing ones, please select one of the following options:

DELETE_FUNCTION

Id of function needs deleting: 1

Confirm delete the function **Cancel**

Click “confirm delete the function” to delete the selection operation, once committed, the count record of the function is deleted as well, so this needs to be done carefully

Predefined operations

In order to view the predefined operations and specific parameters click the load button. **Load**

id	Function_type	Logic_condition	Alarm_limit_count	Result_published_to	Code
1	COUNT	Temperature_firmware <= 20		fiware_v1_id,host.docker.internal:1026,urn:ngsi-ld:test.Alert.count20	
2	COUNT	Pressure_firmware >= 20		fiware_v1_id,host.docker.internal:1026,urn:ngsi-ld:test.Alert.count100	
3	COUNT	Temperature_firmware > 5 and Pressure_firmware <= 30 and Pressure_firmware > 0		fiware_v1_id,host.docker.internal:1026,urn:ngsi-ld:test.Alert.count5	

Input parameters interface

Input parameters interface is used to define, delete or modify the input variables that are used for logic operations

Input overview

In order to view all defined inputs, select load button **Load**

Id	Name	Topic to data cluster/ Fiware id/ Opc-ua data node	Path to variable data	Server	Adapter_type	Close
1	Temperature	/tuanvutest/Temperature	Temperature	broker.hivemq.com	mqtt	
2	Pressure	/tuanvutest/Pressure	Pressure	broker.hivemq.com	mqtt	
3	Temperature_emx	/tuanvutest/Temperature	Temperature_emx	broker.emqx.io	mqtt	
4	Pressure_emx	/tuanvutest/Pressure	Pressure_emx	broker.emqx.io	mqtt	
5	Moisture	/tuanvutest/Moisture	Moisture	broker.hivemq.com	mqtt	
6	Temperature_opc	ns=2;i=3		opc.tcp://127.0.0.1:12345	opc_ua	
7	Pressure_opc	ns=2;i=2		opc.tcp://127.0.0.1:12345	opc_ua	
8	Temperature_fiware	urn:ngsi-Id:test:Device:Temperature	value/value	host.docker.internal:1026	fiware_v1_Id	
9	Pressure_fiware	urn:ngsi-Id:test:Device:Pressure	value/value	host.docker.internal:1026	fiware_v1_Id	
10	Moisture_emx	/tuanvutest/Moisture	Moisture_emx	broker.emqx.io	mqtt	
11	Moisture_opc	ns=2;i=4		opc.tcp://127.0.0.1:12345	opc_ua	
12	datapoint1	test_entity	value/value	host.docker.internal:1026	fiware_v2	

Defining new input variables

Choose option “NEW_INPUT” to bring up new section

Input definition

In order to define new input or modify defined input, choose one of options below:

NEW_INPUT ▼

1. Input name (users' choice)
2. Choose your adapter **No adapter** ▼
3. Input topic to data cluster(MQTT)/ Entity ID/ OPC-UA node cluster location
4. Path to data for input assignment inside data cluster
5. Server where input data is stored (OPC-UA, MQTT) or url (Fiware)

Save **Cancel change**

List of parameters:

- Input name: the name of input variable to be assigned value to
- Adapter choice: the protocol that carries the input
- Input topic to data cluster (MQTT)/ Entity ID/ OPC-UA node cluster location: depends on the adapter type, this can be either topic (MQTT), entity id (fiware), node id (OPC-UA), where the data cluster (json file or data) is stored.
- Path to data for input assignment inside data cluster: path within the json file where relevant data can be extracted (if json is used), can be empty if data can be used directly
- Server where input data is stored (OPC-UA, MQTT) or url (Fiware): the server/link where the data is located.

For example, the value for Moisture_fiware variable is located in dockerized fiware orion-Id server with entity id urn:ngsi-Id:test:Device:Moisture, the json file has format:

DCF manual

JSON	Raw Data	Headers
Save Copy Collapse All Expand All Filter JSON		
id:		"urn:ngsi-ld:test:Device:Moisture"
type:		"https://uri.fiware.org/ns/data-models#Device"
▼ https://smart-data-models.github.io/data-models/terms.jsonld#/definitions/source:		
type:		"Relationship"
object:		"urn:ngsi-ld:Device:company-xyz:busbar-789"
▼ https://smart-data-models.github.io/data-models/terms.jsonld#/definitions/category:		
type:		"Property"
value:		"sensor"
▼ https://smart-data-models.github.io/data-models/terms.jsonld#/definitions/serialNumber:		
type:		"Property"
value:		"9845A"
▼ value:		
type:		"Property"
value:		9.05
observedAt:		"2020-12-01T11:23:19.000Z"
▼ https://smart-data-models.github.io/data-models/terms.jsonld#/definitions/deviceState:		
type:		"Property"
value:		"ok"
▼ isSpecifiedBy:		
type:		"Property"
▼ value:		
type:		"Relationship"
▼ object:		"urn:ngsi-ld:ResourceSpecification:company-xyz:sensor"

And users want to extract only the numeric value of the entity, so the path to the value is "value/value". Thus, the inputs for the variable is

Input name: Moisture_fiware

Choose your adapter: Fiware v1 ld

Input topic to data cluster(MQTT)/ Entity ID/ OPC-UA node cluster location: urn:ngsi-ld:test:Device:Moisture

Path to data for input assignment inside data cluster: value/value

Server where input data is stored (OPC-UA, MQTT) or url (Fiware):
host.docker.internal:1026

To save the variable, choose "Save", or else, choose "Cancel change"

1. Input name (users' choice)

Moisture_fiware

2. Choose your adapter

Fiware v1 ld

3. Input topic to data cluster(MQTT)/ Entity ID/ OPC-UA node cluster location

urn:ngsi-ld:test:Device:Moisture

4. Path to data for input assignment inside data cluster

value/value

5. Server where input data is stored (OPC-UA, MQTT) or url (Fiware)

host.docker.internal:1026

Save

Cancel change

Id	Name	Topic to data cluster/ Fiware id/ Opc-ua data node	Path to variable data	Server	Adapter_type	Close
1	Temperature	/tuanvutest/Temperature	Temperature	broker.hivemq.com	mqtt	
2	Pressure	/tuanvutest/Pressure	Pressure	broker.hivemq.com	mqtt	
3	Temperature_emx	/tuanvutest/Temperature	Temperature_emx	broker.emqx.io	mqtt	
4	Pressure_emx	/tuanvutest/Pressure	Pressure_emx	broker.emqx.io	mqtt	
5	Moisture	/tuanvutest/Moisture	Moisture	broker.hivemq.com	mqtt	
6	Temperature_opc	ns=2;i=3		opc.tcp://127.0.0.1:12345	opc_ua	
7	Pressure_opc	ns=2;i=2		opc.tcp://127.0.0.1:12345	opc_ua	
8	Temperature_fiware	urn:ngsi-Id:test:Device:Temperature	value/value	host.docker.internal:1026	fiware_v1_Id	
9	Pressure_fiware	urn:ngsi-Id:test:Device:Pressure	value/value	host.docker.internal:1026	fiware_v1_Id	
10	Moisture_emx	/tuanvutest/Moisture	Moisture_emx	broker.emqx.io	mqtt	
11	Moisture_opc	ns=2;i=4		opc.tcp://127.0.0.1:12345	opc_ua	
12	datapoint1	test_entity	value/value	host.docker.internal:1026	fiware_v2	
13	Moisture_fiware	urn:ngsi-Id:test:Device:Moisture	value/value	host.docker.internal:1026	fiware_v1_Id	

Modify existing input

To modify existing variable, choose “CHANGE_EXISTING_INPUT”

Input definition

In order to define new input or modify defined input, choose one of options below:

CHANGE_EXISTING_INPUT

Id of defined input needs changing:

Get the input parameters

Cancel change

The id of the variable need to be provided, after confirming the id, the input variable detail is loaded in the windows

In order to define new input or modify defined input, choose one of options below:

CHANGE_EXISTING_INPUT

Id of defined input needs changing:

Get the input parameters

Cancel change

1. Input name (users' choice)

2. Choose your adapter

Fiware v1 Id

3. Input topic to data cluster(MQTT)/ Entity ID/ OPC-UA node cluster location

4. Path to data for input assignment inside data cluster

5. Server where input data is stored (OPC-UA, MQTT) or url (Fiware)

Save

Cancel change

Delete existing input

To delete existing input variable, select “DELETE_VARIABLE” and provide the id of the input needing to be deleted

Input definition

In order to define new input or modify defined input, choose one of options below:

DELETE_INPUT ▾

Id of defined input needs deleting:

Confirm delete the input Cancel change

Legacy data interface

The interface allows users to

- Define database entry with parameters for saving the data within DCF (applies for oracle and sap hana)
- Define fiware entities for data extraction from saved database (the same functionality is shared with mongoDB interface)



Database entry section

Displaying db table info and content

Allows users to monitor/add new/ delete existing database information and view table content if data is retrieved, if the data is not retrieved, the old entry needs to be deleted and defined again.

For example, saved data with table_id ORACLE_IRISDATA1 is displayed

legacy GUI

load all saved db info

Entry_NO.	table_id	type	server	port	table	target_column	Close
1	ORACLE_TIMEDATA1	Oracle	localhost	1521	TIMEDATA	*	get this db
2	ORACLE_IRISDATA1	Oracle	localhost	1521	IRISDATA	*	get this db

ORACLE_IRISDATA1

SEPAL_LENGTH	SEPAL_WIDTH	PETAL_LENGTH	PETAL_WIDTH	SPECIES	Close
0.01	0.01	1.01	2.01	dawawd	
1.01	1.01	2.01	3.01	dads	
2.01	2.01	3.01	4.01	dsdaw	
3.01	3.01	4.01	5.01	dwafs	
4.01	4.01	5.01	6.01	caw	

choose an action ▼

Adding new database entry and deleting existing entry

Adding new entry

To add new database entry, select “new database entry” to bring up new input section

load all saved db info

Entry_NO.	table_id	type	server	port	table	target_column	Close
1	ORACLE_TIMEDATA1	Oracle	localhost	1521	TIMEDATA	*	get this db
2	ORACLE_IRISDATA1	Oracle	localhost	1521	IRISDATA	*	get this db

new database entry

Oracle

server address

server port

Username

Password

Table name

Get column, put "*" if users want to get entire table

Assign table id for data extraction

save database info

cancel

For example, data from column SPECIES and SEPAL_WIDTH of IRISDATA table from oracle database needs to be stored, to log in and retrieve data, username and password are also needed, the retrieved data is assigned to unique table_id ORACLE_IRISDATA2, the columns to be retrieved need to be separated by “,” (comma).

load all saved db info

Entry_NO.	table_id	type	server	port	table	target_column	Close
1	ORACLE_TIMEDATA1	Oracle	localhost	1521	TIMEDATA	*	get this db
2	ORACLE_IRISDATA1	Oracle	localhost	1521	IRISDATA	*	get this db

new database entry

Oracle

server address localhost

server port 1521

Username system

Password Test_372

Table name IRISDATA

Get column, put "*" if users want to get entire table SPECIES,SEPAL_WIDTH

Assign table id for data extraction ORACLE_IRISDATA2

save database info cancel

Click "save database info" and retrieve the data

load all saved db info

Entry_NO.	table_id	type	server	port	table	target_column	Close
1	ORACLE_TIMEDATA1	Oracle	localhost	1521	TIMEDATA	*	get this db
2	ORACLE_IRISDATA1	Oracle	localhost	1521	IRISDATA	*	get this db
3	ORACLE_IRISDATA2	Oracle	localhost	1521	IRISDATA	SPECIES,SEPAL_WIDTH	get this db

ORACLE_IRISDATA2

SPECIES	SEPAL_WIDTH	Close
dawawd	0.01	
dads	1.01	
dsdaw	2.01	
dwafs	3.01	
caw	4.01	

choose an action

Deleting existing entry

To delete entry, choose “delete database entry” and provide the ENTRY_NO. of the needed entry

legacy GUI

load all saved db info

Entry_NO.	table_id	type	server	port	table	target_column	Close
1	ORACLE_TIMEDATA1	Oracle	localhost	1521	TIMEDATA	*	get this db
2	ORACLE_IRISDATA1	Oracle	localhost	1521	IRISDATA	*	get this db
3	ORACLE_IRISDATA2	Oracle	localhost	1521	IRISDATA	SPECIES,SEPAL_WIDTH	get this db

delete database entry ▾

choose ENTRY_NO. to be deleted

save database info

cancel

Adding new parameters for retrieving data from saved database entries or deleting existing entries

load all saved variables info

id	table_id	reference_column	reference_value	target_column	fiware_id	fiware_server	Close
1	ORACLE_IRISDATA1	SPECIES	dawawd	PETAL_LENGTH,PETAL_WIDTH	urn:ngsi-Id:Device:flower	orion:1026	
2	MONGODB_PRODUCT1	product_name	lascannon	price	urn:ngsi-Id:Device:lascannon	orion:1026	

choose an action ▾

Adding new entry for retrieving data

Data from saved database table can be extracted and sent through fiware as entity, this functionality is shared and can be shown in both MongoDB interface and legacy interface, to add new parameters entry, choose “new variable entry”

load all saved variables info

id	table_id	reference_column	reference_value	target_column	fiware_id	fiware_server	Close
1	ORACLE_IRISDATA1	SPECIES	dawawd	PETAL_LENGTH,PETAL_WIDTH	urn:ngsi-Id:Device:flower	orion:1026	
2	MONGODB_PRODUCT1	product_name	lascannon	price	urn:ngsi-Id:Device:lascannon	orion:1026	

new variable entry ▾

Data from table (use table_id) Column used for reference Value for reference column

Target columns for data extraction, use "," to separate columns, enter "" if users want to get entire rows

Fiware entity id, format urn:ngsi-Id:Device:{your designated id} Orion-Id/fiware server domain name and port

save variable info

cancel

The table_id must be the table_id of one of defined database table (either legacy or mongodb table).

For example, defining a new entity with id “urn:ngsi-lid:Device:flower1” that retrieves the SEPAL_LENGTH, SEPAL_WIDTH of SPECIES with name “caw” from table ORACLE_IRISDATA1

new variable entry

Data from table (use table_id)

ORACLE_IRISDATA1

Column used for reference

SPECIES

Value for reference column

caw

Target columns for data extraction, use "," to separate columns, enter "*" if users want to get entire rows

_LENGTH,SEPAL_WIDTH

Fiware entity id, format urn:ngsi-lid:Device:{your designated id}

n:ngsi-lid:Device:flower1

Orion-lid/fiware server domain name and port

orion:1026

save variable info

cancel

load all saved variables info

id	table_id	reference_column	reference_value	target_column	fiware_id	fiware_server	Close
1	ORACLE_IRISDATA1	SPECIES	dawawd	PETAL_LENGTH,PETAL_WIDTH	urn:ngsi-lid:Device:flower	orion:1026	
2	MONGODB_PRODUCT1	product_name	lascannon	price	urn:ngsi-lid:Device:lascannon	orion:1026	
3	ORACLE_IRISDATA1	SPECIES	caw	SEPAL_LENGTH,SEPAL_WIDTH	urn:ngsi-lid:Device:flower1	orion:1026	

choose an action

Result from orion server

←

→

↺

🏠

localhost:1026/ngsi-lid/v1/entities/urn:ngsi-lid:Device:flower1

JSONRaw DataHeaders

SaveCopyCollapse AllExpand AllFilter JSON

id:

type:

▼ https://smart-data-models.github.io/data-models/terms.jsonld#/definitions/source:

type:

object:

▼ value:

type:

▼ value:

▼ value:

▼ 0:

SPECIES:

SEPAL_LENGTH:

SEPAL_WIDTH:

type:

observedAt:

▼ isSpecifiedBy:

type:

▼ value:

type:

object:

"urn:ngsi-lid:Device:flower1"

"https://uri.fiware.org/ns/data-models#Device"

"Relationship"

"urn:ngsi-lid:Device:company-xyz:database"

"Property"

"caw"

4.01

4.01

"Property"

"2022-07-29T17:26:55.450Z"

"Property"

"Relationship"

"urn:ngsi-lid:ResourceSpecification:legacyDatabase"

Delete entry

To delete entry, choose “delete variable entry” and provide the id of the entry that need deleting

load all saved variables info

id	table_id	reference_column	reference_value	target_column	fiware_id	fiware_server	Close
1	ORACLE_IRISDATA1	SPECIES	dawawd	PETAL_LENGTH,PETAL_WIDTH	urn:ngsi-lid:Device:flower	orion:1026	
2	MONGODB_PRODUCT1	product_name	lascannon	price	urn:ngsi-lid:Device:lascannon	orion:1026	
3	ORACLE_IRISDATA1	SPECIES	caw	SEPAL_LENGTH,SEPAL_WIDTH	urn:ngsi-lid:Device:flower1	orion:1026	

delete variable entry ▼

choose db id to be deleted

save database infocancel

MongoDB database interface

MongoDB database interface serves two purposes:

- Define database entry with parameters for saving the data within DCF (only apply for mongoDB)
- Define fiware entities for data extraction from saved database (this functionality is the same as the one in legacy interface)

back to main dashboard page

MongoDB GUI

load all saved mongodb table info

choose an action ▼

load all saved variables info

choose an action ▼

Adding new mongoDB info entry for saving data or deleting existing entry

Adding new entry

To add new entry, select “new database entry”

MongoDB GUI

load all saved mongodb table info

Entry_NO.	table_id	server_address	db_name	collection_name	Close
1	MONGODB_PRODUCT1	mongodb://mongo-db/	test_db	test_col	get this db

new database entry ▾

server uri

Database name

Collection name

Assign table id for data extraction

save database info

cancel

Users need to provide needed parameters and unique table id, the table id is any of users’ choosing and must be unique.

After the entry is defined and data is retrieved, the data can be viewed and used for data extraction.

For example, the content of table_id MONGODB_PRODUCT1

load all saved mongodb table info

Entry_NO.	table_id	server_address	db_name	collection_name	Close
1	MONGODB_PRODUCT1	mongodb://mongo-db/	test_db	test_col	get this db

MONGODB_PRODUCT1

product_id	product_name	price	Close
1	lascannon	1200	
2	cogitator	200	
3	lance	350	
4	battle barge	8000	
5	cheese	400	

Deleting existing entry

To delete existing entry, select “delete database entry” and provide the ENTRY_NO. of the entry that needs deleting

MongoDB GUI

load all saved mongodb table info

Entry_NO.	table_id	server_address	db_name	collection_name	Close
1	MONGODB_PRODUCT1	mongodb://mongo-db/	test_db	test_col	get this db

delete database entry ▼

choose ENTRY_NO. to be deleted

save database info

cancel

Adding new parameters entry for data retrieval or deleting existing entry

The procedures are the same as in legacy interface

Update note on Legacy and Mongo DB database interface

Since version 1.6, both Legacy and Mongo DB interfaces are merged into the same interface.

Legacy and MongoDB data

Legacy and MongoDB database entry overview

All saved legacy database entries

In order to view all saved legacy database entries, select load button [Load](#)

All saved MongoDB entries

In order to view all saved MongoDB database entries, select load button [Load](#)

Database entry definition (legacy and MongoDB)

Adding new database entry or deleting existing entry

In order to define new entry or delete existing entry, choose one of the options below:

Choose an action

Data extraction entry overview

Existing data extraction entries from existing legacy database and MongoDB

Currently all data extracted from databases are used to create fiware/orion-Id as output

In order to view all defined data extraction entries [Load](#)

Adding new data extraction entry or deleting existing entry

In order to define new data extraction entry or delete existing entry, select an option below:

Choose an action

While procedures remain the same for all entries, there is a change to MongoDB domain for data extraction. Due to DCF container is no longer placed within the same network as dockerized MongoDB and Fiware orion-Id container, the domain for MongoDB is “mongodb://host.docker.internal/” and fiware orion-Id domain “host.docker.internal:1026”

All saved MongoDB entries

In order to view all saved MongoDB database entries, select load button [Load](#)

Entry_NO.	Table_id	Server_address	DB_name	Collection_name	Retrieval_status	Close
1	MONGODB_PRODUCT2	mongodb://host.docker.internal/	test_db	test_col	data retrieved	Get this db

Menu selection for database entries manipulation

Database entry definition (legacy and MongoDB)

Adding new database entry or deleting existing entry

In order to define new entry or delete existing entry, choose one of the options below:

Choose an action

Choose an action

New legacy database entry

Delete legacy database entry

New MongoDB database entry

Delete MongoDB database entry

Data extraction entry overview

Domain name for entity publishing.

Existing data extraction entries from existing legacy database and MongoDB

Currently all data extracted from databases are used to create fiware/orion-Id as output

In order to view all defined data extraction entries [Load](#)

id	table_id	reference_column	reference_value	target_column	fiware_id	fiware_server	Close
1	ORACLE_IRISDATA1	SPECIES	dawawd	PETAL_LENGTH,PETAL_WIDTH	urn:ngsi-Id:Device:flower	host.docker.internal:1026	
2	MONGODB_PRODUCT1	product_name	lascannon	price	urn:ngsi-Id:Device:lascannon	host.docker.internal:1026	