FIWARE-CEP v5.4.1 Specification

DATE: 30 July 2016

This version:

http://sample.com/api/v5.4.1

Previous version:

http://sample.com/api/v4.4.1

Latest version:

http://sample.com/api/latest

Copyright

License

Table of Contents

API Summary

specification

CEP API Cookbook

As described in the CEP GE open specification document, CEP has three main interfaces:

- Receiving raw events from event producers using a RESTful service
- Sending output events to event consumers using an output REST client adapter
- Administrating the CEP engine state, and its definition repository

Please check the following FI-WARE Open Specification Legal Notice to understand the rights to use this specification.

API Specification

Default

CEP Instance API Root **[/{instance_name}/rest]**

Parameters

instance_name (required, string)
the name of the CEP instance. The default name is ProtonOnWebServer. Used to support multiple CEP instances running on the same server.

Receiving input events [/events]

Note:

- Name is a built-in attribute used to represent the event type being reported. Please consult the user guide for event representation and built-in attributes.
- The data in the tag format should be given with no blanks.
- In the JSON format, all the attributes values are given as strings, the CEP processes each attribute value according to its defined type (in the event definition).

Receive a new input event in json, tag, NGSI xml or NGSI json formats

POST /events

Sending output events

POST /application-name/consumer

The CEP GE activates a REST client for sending output events (in a push mode) to an external application REST service

The following is what the REST consumer will generate as a request to an external REST service called /application-name/consumer. This external REST service is expected to be able to interpret either the tag-delimited, JSON, XML/NGSI or JSON/NGSI data formats sent via the POST method. Note: 'Name' is a built-in attribute used to represent the event type being reported. Please consult the user guide for event representation and built-in attributes.

Note: that this is an external REST service of another application that is activated by the CEP consumer, according to the defined consumer in the CEP application. The URI of the service, the format, and the event types to be sent to this service are defined as part of the Consumer definition as part of the CEP application definition. Although this is not a CEP api, it is described here to show what is the data the CEP posts to an external service, when such a consumer is defined.

CEP Administration API Root [/{CEP_Admin}/resources/]

Parameters

CEP_Admin (required, string)
the name of the CEP administration service. The default name is ProtonOnWebServerAdmin.

Definitions [/definitions]

This service allows to manage the definitions repository. The repository is a file directory. Adding or deleting a definition will add or remove a file from the directory respectively. Each definition represents a CEP application definition in json format. Each definition is identified by a unique name (prefixed by the repository location) and a URI associated with it. The URI is used to retrieve the file by the applications that make use of the definition.

Retrieve all the existing definitions in the repository

GET /definitions

Creating a new definition

POST /definitions

The application definition itself, in json format, can be generated by the CEP UI, and can be generated programmatically by any other application. The format of this definition file is described by a [JSON schema for a CEP application] (https://forge.fi-ware.eu/docman/view.php/9/2732/CEP_EPN_Schema_FI_WARE.json) while the semantics of the various elements in this schema are described in a user guide. Examples for definitions can be found in the CEP test plan. The “name” property (containing the name for the definition) added alongside the “epn” property (containing the full definition).

Retrieve an application definition in JSON format

GET /{definition_name}

Parameters

definition_name (required, string)
the name of the CEP definition e.g., DoSAttack

Replace content of an existing definition with new content

PUT /{definition_name}

Parameters

definition_name (required, string)
the name of the CEP definition

Delete a definition

DELETE /{definition_name}

Parameters

definition_name (required, string)
the name of the CEP definition

Administrating run-time instances

[/instances/{instance_name}]

There are two administration actions that can be performed on a run-time instance. The first is changing the definition (epn) for the instance to work with on its next activation. This will define the types of events the instance will accept for processing and the type of patterns it will be computing. The second action is to start or stop the run time engine.

Note: When you change the definition file of the engine, it has no affect on the current run. You need to stop and start the engine to make it run with the updated definition.

Parameters

instance_name (required, string)
the name of the CEP instance. The default value is ProtonOnWebServerAdmin

Get instance status

GET /instances/{instance_name}

Configuring/Changing a definition for an instance

PUT /instances/{instance_name}

Note that there are two actions. The first action tells the engine to run with a different definition set. You must stop and start the engine if you want it to read the updated definition. The second action is to change the run status of the engine. To start a run or to stop a current run.

Examples

Default

Receiving input events

[/events]

Receive a new input event in json, tag, NGSI xml or NGSI json formats

POST /events

Request /events (application/json)

Headers

```
Content-Type: application/json
```

Body

```
{
   "Name":"TrafficReport",
   "volume":"1000"
}
```

Response 201

Request /events (text/plain)

Headers

```
Content-Type: text/plain
```

Body

```
Name=TrafficReport;volume=1000;
```

Response 201

Request /events (application/xml)

Headers

```
Content-Type: application/xml
```

Body

```
<notifyContextRequest>
<subscriptionId>51a60c7a286043f73ce9606c</subscriptionId>
<originator>localhost</originator>
<contextResponseList>
   <contextElementResponse>
      <contextElement>
         <entityId type="Node" isPattern="false">
            <id>OUTSMART.NODE_3505</id>
         </entityId>
```

```
        <contextAttributeList>
          <contextAttribute>
            <name>TimeInstant</name>
            <type>urn:x-ogc:def:trs:IDAS:1.0:ISO8601</type>
            <contextValue>2013-05-31T18:59:08+0300</contextValue>
          </contextAttribute>
          <contextAttribute>
            <name>presence</name>
            <type>urn:x-ogc:def:phenomenon:IDAS:1.0:presence</type>
            <contextValue></contextValue>
          </contextAttribute>
          <contextAttribute>
            <name>batteryCharge</name>
            <type>urn:x-ogc:def:phenomenon:IDAS:1.0:batteryCharge</type>
            <contextValue>2</contextValue>
          </contextAttribute>
          <contextAttribute>
            <name>illuminance</name>
            <type>urn:x-ogc:def:phenomenon:IDAS:1.0:illuminance</type>
            <contextValue></contextValue>
          </contextAttribute>
          <contextAttribute>
            <name>Latitud</name>
            <type>urn:x-ogc:def:phenomenon:IDAS:1.0:latitude</type>
            <contextValue></contextValue>
          </contextAttribute>
          <contextAttribute>
            <name>Longitud</name>
            <type>urn:x-ogc:def:phenomenon:IDAS:1.0:longitude</type>
            <contextValue></contextValue>
          </contextAttribute>
        </contextAttributeList>
      </contextElement>
      <statusCode>
        <code>200</code>
        <reasonPhrase>OK</reasonPhrase>
      </statusCode>
    </contextElementResponse>
</contextResponseList>
</notifyContextRequest>
```

Response 201

Sending output events

POST /application-name/consumer

Request /application-name/consumer (application/json)

Headers

```
Content-Type: application/json
```

Body


```
{
   "Cost":"0.0",
   "Certainty":"0.0",
   "Name":"TrafficReport",
   "EventSource":"",
   "Duration":"0.0",
   "Annotation":"",
   "volume":"1000",
   "EventId":"e206b5e8-9f3a-4711-9f46-d0e9431fe215",
   "DetectionTime":"1350311378034"
}
```


Response 201

Request /application-name/consumer (text/plain)

Headers

Content-Type: **text/plain**

Body


```
Name=TrafficReport;Certainty=0.0;Cost=0.0;EventSource=;OccurrenceTime=null;Annot
ation=;Duration=0.0;volume=1000;EventId=40f68052-3c7c-4245ae5a-6e20def2e618;E
xpirationTime=null;Chronon=null;DetectionTime=1349181899221;
```


Response 201

Request /application-name/consumer (application/xml)

Headers

Content-Type: **application/xml**

Body


```
<updateContextRequest>
   <contextElementList>
      <contextElement>
         <entityId type="CEPEventReporter" isPattern="false">
            <id>CEPEventReporter_Singleton</id>
         </entityId>
         <contextAttributeList>
            <contextAttribute>
               <name>EventId</name>
               <contextValue>4be0ab1c-ec30-4525-b278-78222f3ce081</contextValu
e>
            </contextAttribute>
            <contextAttribute>
```

```
            <name>EventType</name>
            <contextValue>LowBatteryAlert</contextValue>
          </contextAttribute>
          <contextAttribute>
            <name>DetectionTime</name>
            <contextValue>2013-06-05T08:25:15.804000CEST</contextValue>
          </contextAttribute>
          <contextAttribute>
            <name>EventSeverity</name>
            <contextValue>Critical</contextValue>
          </contextAttribute>
          <contextAttribute>
            <name>Cost</name>
            <contextValue>0.0</contextValue>
          </contextAttribute>
          <contextAttribute>
            <name>Certainty</name>
            <contextValue>1</contextValue>
          </contextAttribute>
          <contextAttribute>
            <name>Name</name>
            <contextValue>LowBatteryAlert</contextValue>
          </contextAttribute>
          <contextAttribute>
            <name>OccurrenceTime</name>
            <contextValue>2013-06-05T08:25:15.804000CEST</contextValue>
          </contextAttribute>
          <contextAttribute>
            <name>TimeInstant</name>
            <contextValue>2013-06-05T08:24:45.581000CEST</contextValue>
          </contextAttribute>
          <contextAttribute>
            <name>Duration</name>
            <contextValue>0</contextValue>
          </contextAttribute>
          <contextAttribute>
            <name>AffectedEntityType</name>
            <contextValue>Node</contextValue>
          </contextAttribute>
          <contextAttribute>
            <name>AffectedEntity</name>
            <contextValue>OUTSMART.NODE_3505</contextValue>
          </contextAttribute>
        </contextAttributeList>
      </contextElement>
    </contextElementList>
    <updateAction>UPDATE</updateAction>
</updateContextRequest>
```

Response 201

Definitions

[/definitions]

Creating a new definition

POST /definitions

Request /definitions

Body

```
{
   "name":"MyDefinition",
   "epn":{...}
}
```

Response 201

Body

```
/ProtonOnWebServerAdmin/resources/definitions/MyDefinition
```

Replace content of an existing definition with new content

PUT /{definition_name}

Request /{definition_name}

Body

```
{
   "epn": {...}
}
```

Response 200

Body

```
/ProtonOnWebServerAdmin/resources/definitions/MyDefinition
```

Administrating run-time instances

[/instances/{instance_name}]

Configuring/Changing a definition for an instance

PUT /instances/{instance_name}

Request /instances/{instance_name}

Body


```
{
   "action":"ChangeDefinitions",
   "definitions-url":"/ProtonOnWebServerAdmin/resources/definitions/DoSAttac
k"
}
```


Response 200

Request /instances/{instance_name}

Body


```
{
   "action":"ChangeState",
   "state":"start"
}
```


Response 200

Request /instances/{instance_name}

Body


```
{
   "action":"ChangeState",
   "state”:”stop”
}
```


Response 200

References

- Apiary project (http://docs.na.apiary.io/#reference)
- Github source (http://github.com/ishkin/Proton.git)
- Apache License, Version 2.0 (https://opensource.org/licenses/Apache-2.0)
- CEP GE open specification document (http://forge.fiware.org/plugins/mediawiki/wiki/fiware/index.php?title=FIWARE.OpenSpecification.Data.CEP)
- FI-WARE Open Specification Legal Notice (http://forge.fiware.org/plugins/mediawiki/wiki/fiware/index.php/FI-WARE_Open_Specification_Legal_Notice_(implicit_patents_license))
- https://forge.fi-ware.eu/docman/view.php/9/2732/CEP_EPN_Schema_FI_WARE.json
- CEP test plan (https://forge.fiware.org/plugins/mediawiki/wiki/fiware/index.php/CEP_GE_-_IBM_Proactive_Technology_Online_Unit_Testing_Plan)