IoT Big Data NGSIv2 API Profile

Version 1.0

25 October 2016

This is a Non-binding Permanent Reference Document of the GSMA

Security Classification: Non-confidential

Access to and distribution of this document is restricted to the persons permitted by the security classification. This document is confidential to the Association and is subject to copyright protection. This document is to be used only for the purposes for which it has been supplied and information contained in it must not be disclosed or in any other way made available, in whole or in part, to persons other than those permitted under the security classification without the prior written approval of the Association.

Copyright Notice

Disclaimer

The GSM Association ("Association") makes no representation, warranty or undertaking (express or implied) with respect to and does not accept any responsibility for, and hereby disclaims liability for the accuracy or completeness or timeliness of the information contained in this document. The information contained in this document may be subject to change without prior notice.

Antitrust Notice

The information contain herein is in full compliance with the GSM Association's antitrust compliance policy.

Table of Contents

1 Introduction

1.1 Overview

This document has been developed to identify the set of capabilities that are required of the 'Data & Control API' to support storage, retrieval, querying, update, deletion and subscriptions for 'harmonised entities' used in the IoT Big Data ecosystem.

The project has identified the use of the FIWARE NGSIv2 API for such purposes and this document identifies the parts of the FIWARE standard that should be supported by implementers in delivering the IoT Big Data ecosystem.

This document quotes extensively and should be read in conjunction with the FIWARE specification which is published at the following location.

http://fiware.github.io/specifications/ngsiv2/stable/

1.2 Scope

This document 'profiles' the NGSIv2 specification to identify the capabilities of the NGSIv2 API (Application Programming Interface) that are required for implementation in order to support the IoT Big Data ecosystem.

1.3 Abbreviations

Term	Description
API	Application Programming Interface
ASCII	American Standard Code for Information Interchange
FIWARE	Refers to https://www.fiware.org
IoT	Internet of Things
IoTBD	Internet of Things Big Data (project)
JSON	JavaScript Object Notation
JSON-LD	JSON for Linked Data
NGSIv2	Next Generation Service Interface version 2

1.4 References

Ref	Doc Number	Title
[1]	PRD CLP.26	IoT Big Data Harmonised Data Model
[2]	RFC 2119	"Key words for use in RFCs to Indicate Requirement Levels", S. Bradner, March 1997. Available at http://www.ietf.org/rfc/rfc2119.txt
[3]	FIWARE NGSIv2	FIWARE-NGSIv2 Specification available at http://fiware.github.io/specifications/ngsiv2/stable/
[4]	GeoJSON	Specification is available from http://geojson.org
[5]	JSON	Introduction available at http://json.org

Ref	Doc Number	Title
[6]	RFC 4627	JSON specification/ media type https://tools.ietf.org/html/rfc4627
[7]	JSON-LD	JSON for Linking Data http://json-ld.org
[8]	FIWARE Governance	https://www.fiware.org/fiware-governance/
[9]	FIWARE roadmap item NGSI-9	https://docs.google.com/document/d/12yC3BK2JnxzAuulUZyehvYrBu2_kDYpylpD1TY6v7GM/edit
[10]	FIWARE roadmap item JSON-LD	https://docs.google.com/document/d/1lWH_a-JkwfSR9uE03KQbKe09uajpJqsPXk7nHFJPYQs/edit
[11]	PRD CLP.25	IoT Big Data Framework Architecture

1.5 Conventions

"The key words "must", "must not", "required", "shall", "shall not", "should", "should not", "recommended", "may", and "optional" in this document are to be interpreted as described in RFC2119 **Error! Reference source not found.**."

2 Introduction

The NGISv2 API supports the generic capabilities to allow arbitrary data items ('entities') to be stored within a generalised data repository. Once stored they can be retrieved, updated, deleted and searched (queried).

NGSIv2 also supports the ability to create 'subscriptions' whereby a subscribing application can receive updates from the NGSIv2 compliant repository when information is updated relevant to the subscription criteria. Subscriptions can of course be deleted.

This API is seen as of general purpose use for the IoT Big Data ecosystem as there is a requirement to be able to handle many types of IoT and related data (context data). The NGSIv2 API therefore supports the various operations required on IoT devices.

This profile of NGSIv2 only differs from the original specification in those areas marked as "Not Required" and greyed out. In all other respects the NGSIv2 protocol specification remains unchanged.

The entity data types and examples have been updated in line with the IoT Big Data Harmonised Data Entities specification [1] and together with the “IoT Big Data Framework Architecture” [11], aim to define a framework of how mobile operators can approach the delivery of IoT Big Data services.

2.1 Overview of NGSIv2

At the highest level the NGSIv2 API supports the following capabilities

- Create an entity including one or more attributes;
- Retrieve entity (/specified attributes) by entity identifier;
- Update one or more attributes of an existing identity;
- Remove an entity;
- Get an attribute value;
- Update an attribute value;
- Remove an attribute of an entity;
- List entities matching specified criteria;
- List known entity types;
- Retrieve entity type information;
- Create a subscription;
- List subscriptions;
- Retrieve details of a subscription;
- Update details of a subscription;
- Delete a subscription;
- Batch update;
- Batch query.

This document reviews all of the defined capabilities of the NGSIv2 specification and identifies the support required in IoT Big Data Ecosystem implementations according to the following criteria:

- "Must" - all implementations are required to implement the defined capability;
- "Recommended" - it is recommended that the implementation supports the defined capability;
- "Optional" - implementers may choose not to support the defined capability;
- "Not Required" - it is not required that the respective capability is supported/ available for IoT Big Data Ecosystem deployments.

3 NGSIv2 Profile General Specification

In the following description "application" is used generically to refer to either an application reading entity data, or storing/ updating entity/ attribute data.

3.1 API Entry Points

IoT Big Data Ecosystem implementations shall use the following guidance in the API services offered

Capability	**Usage**	**Compliance**
Retrieve API Resources	Allows an application to retrieve the base URL for services (entities, types, subscriptions)	Not required
List entities	Allows an application to list entities matching defined criteria	Must
Create entity	Allows an application to create a new entity instance	Must

Retrieve entity by ID	Allows an application to retrieve the entity information referenced by a known identifier	Must
Retrieve entity attributes (referenced by entity ID)	Allows an application to retrieve specified attributes of an entity referenced by a known identifier	Must
Update or append entity attributes (referenced by entity ID)	Allows an application to update attributes or add to the attributes of an entity referenced by a known identifier	Must
Update existing entity attributes (referenced by entity ID)	Allows an application to update existing attributes of an entity referenced by a known identifier	Not Required
Replace all entity attributes	Allows an application to replace the existing attribute set for an entity (referenced by known identifier) with a new set of attributes	Not Required
Remove entity	Allows an application to remove an entity and all related attributes	Must
Get attribute data (for entity referenced by ID)	Allows an application to retrieve the value and related information of a named attribute for an entity referenced by known identifier. This includes attribute value, type and metadata.	Must
Update attribute data (for entity referenced by ID)	Allows an application to update the value stored for a named attribute for an entity referenced by known identifier. This includes attribute value, type and metadata.	Must
Remove a single attribute (for entity referenced by ID)	Allows an application to remove a named attribute value for an entity referenced by known identifier	Must
Get attribute value (for entity referenced by ID)	Allows an application to retrieve the value of a named attribute for an entity referenced by known identifier. This includes attribute value only.	Must
Update attribute value (for entity referenced by ID)	Allows an application to update the value of a named attribute for an entity referenced by known identifier. This includes attribute value only.	Must
Retrieve entity types	Allows an application to retrieve information (meta data) about the entity types stored	Must
Retrieve entity type	Allows an application to retrieve a list of the attribute types available for a specified entity type	Must
Retrieve subscription list	Allows an application to retrieve a list of active subscriptions it has created	Must
Create a new subscription	Allows an application to create a new subscription based on specified matching criteria	Must
Retrieve subscription (by subscription ID)	Allows an application to retrieve details of the specified subscription	Must

Update subscription (by subscription ID)	Allows an application to update details of a specified subscription	Must
Delete subscription (by subscription ID)	Allows an application to cancel and remove the specified subscription	Must
Batch update	Allows an application to perform a batch update to a number of entities	Must
Batch query	Allows an application to perform a batch query (retrieval) based on specified selection criteria	Must

3.2 Identifier naming

Identifiers shall follow the NGSIv2 naming restrictions i.e.

- Restricted to using the plain ASCII character set excluding control characters, whitespace, &, ?, / and #
- Minimum identifier length is 1 character
- Maximum identifier length is 256 characters
- The names 'id', 'type', 'dateCreated', 'dateModified' and 'geo:distance' are reserved. Applications cannot set identifiers with these names

3.3 Attribute types

Attribute types shall follow the NGSIv2 specification as follows

Type name	**Usage**	**Compliance**
`Array`	An ordered list of values that are dereferenced by numerical index. This is a core JSON type.	Must
`Boolean`	Value of **`true`** or **`false`**. This is a core JSON type.	Must
`Date`	A sequence of characters using ISO 8601 encoding to represent a Date. (https://schema.org/Date)	Must
`DateTime`	A string value using ISO 8601 encoding to represent a Date Time field	Must
`ExtQuantitativeValue`	An extended collection of key value pairs describing a point value characteristic of an entity Specifically adding a **`timestamp`** (the date and time or the observation) to the existing Quantitative Value as defined by schema.org (https://schema.org/QuantitiativeValue)	Must
`geo:box`	Defines a bounding box described by two pairs of WGS-84 geo-coordinates	Must
`geo:json`	Defines a location specified using GeoJSON encoding	Must
`geo:line`	Defines an (open) polygonal line described by a list of pairs of WGS-84 geo-coordinates	Must

`geo:point`	Defines a point on the Earth surface described using a pair of WGS-84 geo-coordinates	Must
`geo:polygon`	Defines a closed polygon described by at least four pairs of WGS-84 geo-coordinates. The first and final geo-coordinate must be identical to form a closed polygon.	Must
`Number`	An integer or floating point number. This is a core JSON type.	Must
`Offer`	An offer definition for goods or services as defined by schema.org (https://schema.org/Offer)	Must
`Organization`	An organisation definition as defined by schema.org (https://schema.org/Organization)	Must
`Person`	A person definition as defined by schema.org (https://schema.org/Person)	Must
`Place`	A place definition as defined by schema.org (https://schema.org/Place)	Must
`PostalAddress`	A Postal Address of an item as defined by schema.org (https://schema.org/PostalAddress)	Must
`Product`	A product definition as defined by schema.org (https://schema.org/Product)	Must
`QuantitativeValue`	A collection of key value pairs describing a point value characteristic of an entity or attribute as defined by schema.org (https://schema.org/QuantitiativeValue)	Must
`Reference`	A sequence of characters which represents a reference to another entity.	Must
`StructuredValue`	A collection of key value pairs. Values may themselves be a Text, Number, Boolean, Array, StructuredValue or DateTime as defined by schema.org (https://schema.org/StructuredValue)	Must
`Text`	A sequence of characters. This is a core JSON type.	Must
`Time`	A sequence of characters using ISO 8601 encoding to represent a Time (https://schema.org/Time)	Must
`URL`	A sequence of characters. Defining a URL. (https://schema.org/URL)	Must

3.4 Query syntax

IoT Big Data Ecosystem implementations shall use the following guidance in the entity query operations supported. This is relevant for example to retrieving entities using a query specification or when creating/ updating a subscription.

The query syntax allows multiple query criteria to be specified which are applied as a logical AND.

Operation	Usage	Compliance

`==`
Alternative specification
`:`
	Check for equality. Includes
	• Simple entity field and specified value equality
	• Entity field value found within a list of specified values
	• Entity field value matches a specified range (numbers, strings and dates)
		Must

`!=`	Check for inequality. Includes • Simple entity field and specified value inequality • Entity field value not present within a list of specified values • Entity field value is not matched in a specified range (numbers, strings and dates)	Must
`>`	Check that the entity field value is greater than a specified value. Applies only to numbers, strings and date fields.	Must
`<`	Check that the entity field value is less than a specified value. Applies only to numbers, strings and date fields.	Must
`>=`	Check that the entity field value is greater than or equal to a specified value. Applies only to numbers, strings and date fields.	Must
`<=`	Check that the entity field value is less than or equal to a specified value. Applies only to numbers, strings and date fields.	Must
`~=`	Check that the entity field value matches the specified regular expression. Applies only to string fields.	Must
`'`*`<value>`*`'`	Allows a string literal **`<value>`** to be specified which may include other reserved characters (such as commas or other query operations)	Must
`<attributeName>`	Checks for the existence of the defined **`<attributeName>`** within the entity. Matches even if the attribute value is null.	Must
`!`*`<attributeName>`*	Checks for the non-existence of the defined **`<attributeName>`** within the entity.	Must

	Matches provided the entity does not possess the specified attribute.	
`georel=near`	Checks a geographical position field of the entity is within a defined minimum or maximum range of a specified reference point	Must
`georel=coveredBy`	Checks that the geographical bounds of the entity are within a defined reference geometry	Must
`georel=intersects`	Checks that the geographical bounds of the entity intersects with a defined reference geometry	Must
`georel=equals`	Checks that the geography of the entity matches exactly the specified geometry	Must
`georel=disjoint`	Checks that the geographic bounds of the entity does not intersect the defined reference geometry	Must

3.5 Geometries

The following implementation guidance is provided for geometries supported for the IoT Big Data Ecosystem

Examples:

Operation	Usage	Compliance
`geometry=point`	Defines a point on the Earth surface described using a pair of WGS-84 geo-coordinates	Must
`geometry=line`	Defines an (open) polygonal line described by a list of pairs of WGS-84 geo-coordinates	Must
`geometry=polygon`	Defines a polygon described by at least four pairs of WGS-84 geo-coordinates. The first and final geo-coordinate must be identical to form a closed polygon.	Must
`geometry=box`	Defines a bounding box described by two pairs of WGS-84 geo-coordinates	Must

Examples

```
georel=near;maxDistance:1000&geometry=point&coords=-40.4,-3.5
```

Matching entities must be located (at most) 1000 meters from the reference point.

```
georel=near;minDistance:5000&geometry=point&coords=-40.4,-3.5
```

Matching entities must be (at least) 5000 meters from the reference point.

```
georel=coveredBy&geometry=polygon&coords=25.774,-80.190;18.466,-
66.118;32.321,-64.757;25.774,-80.190
```

Matching entities are those located within the referenced polygon.

3.6 Subscriptions

As defined in the NGSIv2 specification a subscription is represented by a JSON object with the following fields

Field name	Usage	Type
`id`	Subscription unique identifier. Automatically created at creation time.	String
`description`	An (optional) free text used by the client to describe the subscription.	String
`subject`	An object that describes the subject of the subscription.	Object
`notification`	An object that describes the notification to send when the subscription is triggered.	Object
`expires`	Subscription expiration date in ISO8601 format. Permanent subscriptions must omit this field.	String
`status`	Either **`active`** (for active subscriptions) or **`inactive`** (for inactive subscriptions). If this field is not provided at subscription creation time, new subscriptions are created with the **`active`** status, which can be changed by clients afterwards. For expired subscriptions, this attribute is set to **`expired`** (no matter if the client updates it to **`active`**/**`inactive`**).	String
`throttling`	Minimal period of time in seconds which must elapse between two consecutive notifications. It is optional.	Number

The subject object can contain the following fields

Field name	Usage	Type
`entities`	A list of objects, each one composed of the following subfields (**`id`**/**`idPattern`** or **`type`** must be present): • **`id`** or **`idPattern`**: Id or pattern of the affected entities. Both cannot be used at the same time, but at least one of them must be present. • **`type`**: Type of the affected entities (optional).	Object
`condition`	Condition to trigger notifications. It has two optional properties: • **`attrs`**: array of attribute names • **`expression`**: an expression composed of q, georel, geometry and coords (see "List entities" operation above about this field).	Object

The notification object can contain the following fields

Field name	Usage	Type
attrs or exceptAttrs	Either **attrs** or **exceptAttrs** (not both) • **attrs**: List of attributes to be included in notification messages. It also defines the order in which attributes must appear in notifications when attrsFormat value is used (see "Notification Messages" section). An empty list means that all attributes are to be included in notifications. • **exceptAttrs**: List of attributes to be excluded from the notification message, i.e. a notification message includes all entity attributes except the ones listed in this field. • If neither **attrs** nor **exceptAttrs** is specified, all attributes are included in notifications.	Object
http	It is used to convey parameters for notifications delivered through the HTTP protocol.	Object
attrsFormat	Optionally specifies how the entities are represented in notifications. Accepted values are **normalized** (default), **keyValues** or **values**. If attrsFormat takes any value different than those, an error is raised. See detail in "Notification Messages" section.	String
timesSent	Number of notifications sent due to this subscription. (not editable, only present in GET operations)	Number
lastNotification	Last notification date in ISO8601 format. (not editable, only present in GET operations)	String

An **http** object contains the following subfields

Field name	Usage	Type
url	URL referencing the service to be invoked when a notification is generated. An NGSIv2 compliant server must support the http URL schema. Other schemas could also be supported.	String

3.6.1 Notification Messages

Notifications include two fields

Field name	**Usage**	**Type**
`subscriptionId`	Identifies the subscription that originates the notification	String
`data`	Is an array with the notification data itself which includes the entity and all concerned attributes. Each element in the array corresponds to a different entity. By default, the entities are represented in **`normalized`** mode. However, using the **`attrsFormat`** modifier, a simplified representation mode can be requested.	Array of entity objects

If **`attrsFormat`** is **`normalized`** (or if **`attrsFormat`** is omitted) then default entity representation is used:

```
{
  "subscriptionId": "0335b13d-00f4-4e5f-9c6f-101c274c42a1",
  "data": [
    {
      "id": "7b014918-7ccc-4d2f-af65-6cdfe3690be4",
      "type": "AgriSoil",
      "dateCreated": {
        "value": "2016-08-08T10:18:16Z",
        "type": "DateTime",
        "metadata": {}
      },
      "dateModified": {
        "value": "2016-08-08T10:18:16Z",
        "type": "DateTime",
        "metadata": {}
      },
      "source": {
        "value": "www.samplefarmproduct.com",
        "type": "URL",
        "metadata": {}
      },
      "dataProvider": {
        "value": "OperatorA",
        "type": "Text",
        "metadata": {}
      },
      "name": {
        "value": "Clay",
        "type": "Text",
        "metadata": {}
      }
    },
    {
      "id": "35ac1609-6b31-4cad-8f4c-ad8be828c8a2",
      "type": "AgriSoil",
      "dateCreated": {
        "value": "2016-08-08T10:18:16Z",
        "type": "DateTime",
        "metadata": {}
      },
      "dateModified": {
        "value": "2016-08-08T10:18:16Z",
        "type": "DateTime",
        "metadata": {}
      },
      "source": {
        "value": "www.samplefarmproduct.com",
        "type": "URL",
        "metadata": {}
      },
      "dataProvider": {
        "value": "OperatorA",
        "type": "Text",
        "metadata": {}
      },
      "name": {
        "value": "Silt",
        "type": "Text",
        "metadata": {}
      }
    }
  ]
}
```

If **`attrsFormat`** is **`keyValues`** then keyValues partial entity representation mode is used:

```
{
  "subscriptionId": "0335b13d-00f4-4e5f-9c6f-101c274c42a1",
  "data": [
    {
      "id": "7b014918-7ccc-4d2f-af65-6cdfe3690be4",
      "type": "AgriSoil",
      "dateCreated": "2016-08-08T10:18:16Z",
      "dateModified": "2016-08-08T10:18:16Z",
      "source": "www.samplefarmproduct.com",
      "dataProvider": "OperatorA",
      "name": "Clay"
    },
    {
      "id": "35ac1609-6b31-4cad-8f4c-ad8be828c8a2",
      "type": "AgriSoil",
      "dateCreated": "2016-08-08T10:18:16Z",
      "dateModified": "2016-08-08T10:18:16Z",
      "source": "www.samplefarmproduct.com",
      "dataProvider": "OperatorA",
      "name": "Silt"
    }
  ]
}
```

If **`attrsFormat`** is **`values`** then values partial entity representation mode is used:

```
{
  "subscriptionId": "0335b13d-00f4-4e5f-9c6f-101c274c42a1",
  "data": [
    [
      "2016-08-08T10:18:16Z",
      "2016-08-08T10:18:16Z",
      "www.samplefarmproduct.com",
      [],
      "Silt"
    ],
    [
      "2016-08-08T10:18:16Z",
      "2016-08-08T10:18:16Z",
      "www.samplefarmproduct.com",
      [],
      "Clay"
    ]
  ]
}
```

Notifications must include the **`Ngsiv2-AttrsFormat`** HTTP header with the value of the format of the associated subscription, so that notification receivers are aware of the format without needing to process the notification payload.

4 APIs

4.1 Retrieve API Resources

HTTP Verb: `GET`

Example Resource URL: `/v2`

Request Parameters

None

Response Parameters

Parameter	**Type/ Meaning**	**Required/ Optional**
`entities_url`	String URL which points to the entities resource `/v2/entities`	required
`types_url`	String URL which points to the types resource `/v2/types`	required
`subscriptions_url`	String URL which points to the subscriptions	required

	resource /v2/subscriptions	

HTTP Response Code: 200 (Success)

Response example

```
{
  "entities_url": "/v2/entities",
  "types_url": "/v2/types",
  "subscriptions_url": "/v2/subscriptions"
}
```

4.2 List entities

HTTP Verb: GET

Example Resource URL:
/v2/entities?idPattern=*idPattern*&q=statement&options=options

Request Parameters

Parameter	**Meaning**	**Type**
id	A comma-separated list of elements. Retrieve entities whose ID matches one of the elements in the list. Incompatible with **idPattern**. Example: **Boe_Idearium**.	String
type	comma-separated list of elements. Retrieve entities whose type matches one of the elements in the list. Example: **Soil**.	String
idPattern	A correctly formatted regular expression. Retrieve entities whose ID matches the regular expression. Incompatible with **id**. Example: **AgriSoil**	String
q	A query expression, composed of a list of statements separated by; i.e., **q**=statement; statements; statement. See Query syntax specification. Example: **temperature>40**.	String
georel	Spatial relationship between matching entities and a reference shape. See Geographical Queries. Example: **near**.	String
geometry	Geographical area to which the query is restricted. See Geographical Queries. Example: **point**.	String
coords	List of latitude-longitude pairs of coordinates separated by ';'. See Geographical Queries. Example: **41.390205,2.154007;48.8566,2.3522.**	String
limit	Limits the number of entities to be retrieved Example: **20**.	Number

offset	Establishes the offset from where entities are retrieved Example: **20**.	Number
attrs	Comma-separated list of attribute names whose data are to be included in the response. The attributes are retrieved in the order specified by this parameter. If this parameter is not included, the attributes are retrieved in arbitrary order. Example: **seatNumber**.	String
orderBy	Criteria for ordering results. See "Ordering Results" section for details. Example: **temperature,!speed**.	String
options	Options dictionary with possible values • **count** • **keyValues** • **values** • **unique**	String

Response Parameters

Parameter	Type/ Meaning	Required/ Optional
<entityArray>	Array of objects Array of entities matching the respective selection criteria and ordered according to the ordering criteria	required

Example URL:

```
v2/entities?idPattern=*&limit=2&options=dateCreated,dateModified
```

HTTP Response Code: 200 (Success)

Response example

```
[
  {
    "id": "7b014918-7ccc-4d2f-af65-6cdfe3690be4",
    "type": "AgriSoil",
    "dateCreated": "2016-08-08T10:18:16Z",
    "dateModified": "2016-08-08T10:18:16Z",
    "source": "www.samplefarmproduct.com",
    "dataProvider": "OperatorA",
    "name": "Clay"
  },
  {
    "id": "35ac1609-6b31-4cad-8f4c-ad8be828c8a2",
    "type": "AgriSoil",
    "dateCreated": "2016-08-08T10:18:16Z",
    "dateModified": "2016-08-08T10:18:16Z",
    "source": "www.samplefarmproduct.com",
    "dataProvider": "OperatorA",
    "name": "Silt"
  }
]
```

4.3 Create entity

HTTP Verb: `POST`

Resource URL: `/v2/entities?options=options`

Request Parameters

Parameter	Meaning	Type
`options`	Options dictionary with possible value • **`keyValues`**	String

Example URL:

```
v2/entities?options=keyValues
```

Request body example

```
{
  "id": "a01b4849-2eca-4518-b5a3-230f9749147e",
  "type": "AgriSoil",
  "source": "www.samplefarmproduct.com",
  "dataProvider": "OperatorA",
  "name": "Sand"
}
```

NOTE: The FIWARE Context Broker defines a set of standard attributes for its entities. dateCreated and dateModified are amongst those standard attributes. These 2 attributes are set automatically be the Context Broker.

Response Parameters

Parameter	Type/ Meaning	Required/ Optional
`location` (HTTP header)	String URL which points to the created entity	required

HTTP Response Code: `201` (Created)

Response example

```
Location:/v2/entities/a01b4849-2eca-4518-b5a3-230f9749147e
```

4.4 Retrieve entity

HTTP Verb: `GET`

Example Resource URL: `/v2/entities/entityId?type=`*`type`*`&options=options`

Request Parameters

Parameter	Meaning	Type
`entityId`	Id of the entity to be retrieved	String
`type`	Entity type, to avoid ambiguity in case there are several entities with the same **`entityId`**.	String
`attrs`	Comma-separated list of attribute names whose data must be included in the response. The attributes are retrieved in the order specified by this parameter. If this parameter is not included, the attributes are retrieved in arbitrary order, and all the attributes of the entity are included in the response. Example: **`temperature, humidity`**.	String
`options`	Options dictionary with possible values • **`keyValues`** • **`values`** • **`unique`**	String

Response Parameters

Parameter	Type/ bv	Required/ Optional
`<entity>`	Returns the entity data matching the specified ID and type.	Required

Example URL:

```
/v2/entities/a01b4849-2eca-4518-b5a3-
230f9749147e?type=AgriSoil&options=keyValues,dateCreated,dateModified
```

HTTP Response Code: `200` (Success)

Response example

```
{
  "id": "a01b4849-2eca-4518-b5a3-230f9749147e",
  "type": "AgriSoil",
  "dateCreated": "2016-08-08T10:18:16Z",
  "dateModified": "2016-08-08T10:18:16Z",
  "source": "www.samplefarmproduct.com",
  "dataProvider": "OperatorA",
  "name": "Sand"
}
```

4.5 Retrieve entity attributes

HTTP Verb: `GET`

Example Resource URL:
/v2/entities/entityId/attrs?type=*type*&options=options

Request Parameters

Parameter	Meaning	Type
`entityId`	Id of the entity to be retrieved	String
`type`	Entity type, to avoid ambiguity in case there are several entities with the same **`entityId`**.	String
`attrs`	Comma-separated list of attribute names whose data are to be included in the response. The attributes are retrieved in the order specified by this parameter. If this parameter is not included, the attributes are retrieved in arbitrary order, and all the attributes of the entity are included in the response. Example: **`temperature,humidity`**.	String
`options`	Options dictionary with possible values • **`keyValues`** • **`values`** • **`unique`**	String

Response Parameters

Parameter	Type/ Meaning	Required/ Optional
<entity>	Returns the entity data matching the specified ID and type.	Required

HTTP Response Code: `200` (Success)

Example URL:

```
/v2/entities/a01b4849-2eca-4518-b5a3-
230f9749147e/attrs?type=AgriSoil&attrs=dateCreated,dateModified,name
&options=keyValues
```

Response example

```
{
  "dateCreated": "2016-08-08T10:18:16Z",
  "dateModified": "2016-08-08T10:18:16Z",
  "name": "Sand"
}
```

4.6 Update or append entity attributes

HTTP Verb: `POST`

Resource URL: `/v2/entities/entityId/attrs?type=type&options=options`

Request Parameters

Parameter	**Meaning**	**Type**
`entityId`	Entity id to be updated	String
`type`	Entity type, to avoid ambiguity in case there are several entities with the same **`entityId`**.	String
`options`	Options dictionary with possible values • **`append`** • **`keyValues`**	String

Example URL:

```
/v2/entities/a01b4849-2eca-4518-b5a3-230f9749147e/attrs?options=keyValues
```

Request body example

```
{
  "description": "light, dry, warm, low in nutrients and often acidic"
}
```

Response Parameters

None

HTTP Response Code: `204` (No content)

4.7 Update existing entity attributes

HTTP Verb: `PATCH`

Resource URL: `/v2/entities/entityId/attrs?type=type&options=options`

Request Parameters

Parameter	Meaning	Type
`entityId`	Entity id to be updated	String
`type`	Entity type, to avoid ambiguity in case there are several entities with the same **`entityId.`**	String
`options`	Options dictionary with possible value • **`keyValues`**	String

Example URL:

```
/v2/entities/a01b4849-2eca-4518-b5a3-230f9749147e/attrs?options=keyValues
```

Request body example

```
{
  "source" : "www.samplefarmproductshop.com"
}
```

Response Parameters

None

HTTP Response Code: `204` (No content)

4.8 Replace all entity attributes

HTTP Verb: `PUT`

Resource URL: `/v2/entities/entityId/attrs?type=type&options=options`

Request Parameters

Parameter	Meaning	Type
`entityId`	Entity id to be updated	String
`type`	Entity type, to avoid ambiguity in case there are several entities with the same **`entityId`**.	String
`options`	Options dictionary with possible values • **`keyValues`**	String

Example URL:

```
/v2/entities/a01b4849-2eca-4518-b5a3-230f9749147e/attrs?options=keyValues
```

Request body example

```
{
  "source": "www.samplefarmproductshop.com",
  "dataProvider": "OperatorA",
  "name": "Stones"
}
```

Response Parameters

None

HTTP Response Code: `204` (No content)

4.9 Remove entity

HTTP Verb: `DELETE`

Resource URL: `/v2/entities/entityId?type=type`

Request Parameters

Parameter	Meaning	Type
`entityId`	Entity id to be updated	String
`type`	Entity type, to avoid ambiguity in case there are several entities with the same **`entityId`**.	String

Example URL:

```
/v2/entities/a01b4849-2eca-4518-b5a3-230f9749147e
```

Response Parameters

None

HTTP Response Code: `204` (No content)

4.10 Get attribute data

HTTP Verb: `GET`

Example Resource URL: `/v2/entities/entityId/attrs/attrName?type=type`

Request Parameters

Parameter	Meaning	Type

`entityId`	Id of the entity	String
`type`	Entity type, to avoid ambiguity in case there are several entities with the same **`entityId`**	String
`attrName`	Name of the attribute to be retrieved. Example: **`temperature`**.	String

Response Parameters

Parameter	**Type/ Meaning**	**Required/ Optional**
`<entity>`	Returns the entity data matching the specified ID and type.	Required

Example URL:

`/v2/entities/a01b4849-2eca-4518-b5a3-230f9749147e/attrs/source`

HTTP Response Code: `200` (Success)

Response example

```
{
  "value": "www.samplefarmproductshop.com",
  "type": "URL",
  "metadata": {}
}
```

4.11 Update attribute data

HTTP Verb: `PUT`

Resource URL: `/v2/entities/entityId/attrs/attrName?type=type`

Request Parameters

Parameter	**Meaning**	**Type**
`entityId`	Entity id to be updated	String
`type`	Entity type, to avoid ambiguity in case there are several entities with the same **`entityId`**.	String
`attrName`	Attribute name Example: **`temperature`**.	String

Example URL:

```
/v2/entities/a01b4849-2eca-4518-b5a3-230f9749147e/attrs/source
```

Request body example

```
{
  "value": "www.farmproductshop.com",
  "type": "URL",
  "metadata": {}
}
```

Response Parameters

None

HTTP Response Code: `204` (No content)

4.12 Remove a single attribute

HTTP Verb: `DELETE`

Example Resource URL: `/v2/entities/entityId/attrs/attrName?type=type`

Request Parameters

Parameter	Meaning	Type
`entityId`	Id of the entity	String
`type`	Entity type, to avoid ambiguity in case there are several entities with the same **`entityId`**.	String
`attrName`	Name of the attribute to be deleted. Example: **`temperature`**.	String

Example URL: /v2/entities/a01b4849-2eca-4518-b5a3-230f9749147e/attrs/source

Response Parameters

None

HTTP Response Code: `204` (No Content)

4.13 Get attribute value

HTTP Verb: `GET`

Example Resource URL:
`/v2/entities/entityId/attrs/attrName/value?type=type`

Request Parameters

Parameter	Meaning	Type
`entityId`	Id of the entity	String
`type`	Entity type, to avoid ambiguity in case there are several entities with the same **`entityId`**.	String
`attrName`	Name of the attribute to be deleted. Example: **`temperature`**.	String

Response Parameters

Parameter	**Type/ Meaning**	**Required/ Optional**
`<attributeValue>`	Returns the attribute value matching the specified ID and type and attrName.	Required

Example URL:

`/v2/entities/0d6c64ef-4df9-49ce-8e8e-d73901559df1/attrs/postalAddress/value`

HTTP Response Code: `200` (Success)

Response example

```
{
  "streetAddress": "The Walbrook Building, 25 Walbrook",
  "postalCode": "EC4N 8AF",
  "addressLocality": "London",
  "addressCountry": "United Kingdom"
}
```

4.14 Update attribute value

HTTP Verb: `PUT`

Resource URL: `/v2/entities/entityId/attrs/attrName/value`

Request Parameters

Parameter	Meaning	Type
`entityId`	Entity id to be updated	String
`type`	Entity type, to avoid ambiguity in case there are several entities with the same **`entityId`**.	String
`attrName`	Name of the attribute to be deleted. Example: **`temperature`**.	String

Example URL:

```
/v2/entities/0d6c64ef-4df9-49ce-8e8e-d73901559df1/attrs/postalAddress/value
```

Request body example

```
{
  "streetAddress": "The Walbrook Building, 25 Walbrook",
  "postalCode": "EC4N 8AF",
  "addressLocality": "London",
  "addressCountry": "United Kingdom"
}
```

Response Parameters

None

HTTP Response Code: `204` (No content)

4.15 Retrieve entity types

HTTP Verb: `GET`

Example Resource URL: `/v2/types/?limit=10&offset=20&options=options`

Request Parameters

Parameter	Meaning	Type
`limit`	Limits the number of entities to be retrieved Example: **`20`**.	Number
`offset`	Establishes the offset from where entities are retrieved Example: **`20`**.	Number

`options`	Options dictionary with possible values • **`count`** • **`values`**	String

Response Parameters

Parameter	**Type/ Meaning**	**Required/ Optional**
`<typeArray>`	JSON Array If option=values harmonised entity types are identified by name only otherwise the full entity type definition is returned. Types are sorted by alphabetical name. For each type there is the return of • Type name • Type definition (attributes) • Count of entities of that type	required

Example URL:

```
/v2/types?limit=2&options=count
```

HTTP Response Code: `200` (Success)

Response example

```
[
  {
    "type": "AgriSoil",
    "attrs": {
      "dateCreated": {
        "types": [
          "DateTime"
        ]
      },
      "dateModified": {
        "types": [
          "DateTime"
        ]
      },
      "source": {
        "types": [
          "URL"
        ]
      },
      "dataProvider": {
        "types": [
          "Text"
        ]
      },
      "name": {
        "types": [
          "Text"
        ]
      }
    },
    "count": 12
  },
  {
    "type": "PointOfInterest",
    "attrs": {
      "dateCreated": {
        "types": [
          "DateTime"
        ]
      },
      "dateModified": {
        "types": [
          "DateTime"
        ]
      },
      "source": {
        "types": [
          "URL"
        ]
      },
      "dataProvider": {
        "types": [
          "Text"
        ]
      },
      "location": {
        "types": [
          "geo:json"
        ]
      },
      "category": {
        "types": [
          "Text"
        ]
      }
    },
    "count": 7
  }
]
```

4.16 Retrieve entity type

HTTP Verb: `GET`

Example Resource URL: `/v2/types/entityType`

Request Parameters

Parameter	Meaning	Type
`entityType`	Entity type (name).	String

Response Parameters

Parameter	Type/ Meaning	Required/ Optional
`attrs`	Object map the set of attribute names along with all the entities of such type, represented in a JSON object whose keys are the attribute names and whose values contain information of such attributes (in particular a list of the types used by attributes with that name along with all the entities).	required
`count`	Number the number of entities belonging to that type.	required

Example URL:

`/v2/types/agrisoil?options=dateCreated,dateModified`

HTTP Response Code: `200` (Success)

Response example

```
{
  "attrs": {
    "dateCreated": {
      "types": [
        "DateTime"
      ]
    },
    "dateModified": {
      "types": [
        "DateTime"
      ]
    },
    "source": {
      "types": [
        "URL"
      ]
    },
    "dataProvider": {
      "types": [
        "Text"
      ]
    },
    "name": {
      "types": [
        "Text"
      ]
    }
  },
  "count": 7
}
```

4.17 Retrieve subscriptions

HTTP Verb: `GET`

Example Resource URL: `/v2/subscriptions`

Request Parameters

Parameter	Meaning	Type
`limit`	Limit the number of types to be retrieved Example: **`10`**.	Number
`offset`	Skip a number of records (from the start of the results) Example: **`20`**.	Number
`options`	Options dictionary with possible values - **`count`**	String

Response Parameters

Parameter	Type/ Meaning	Required/ Optional
`<subscriptionsArray>`	Array of objects Array of all the subscriptions present in the system	required

HTTP Response Code: `200` (Success)

Response example

```
{
  "id": "0b06aff5-b476-48e5-9465-7cd8a2671166",
  "description": "One subscription to rule them all",
  "subject": {
    "entities": [
      {
        "id": "0d6c64ef-4df9-49ce-8e8e-d73901559df1",
        "type": "Building"
      }
    ],
    "condition": {
      "attrs": [
        "buildingType"
      ],
      "expression": {
        "q": "buildingType=house"
      }
    }
  },
  "notification": {
    "http": {
      "url": "http://www.gsma.com/iotbd"
    }
  },
  "attrsFormat": "keyValues",
  "attrs": [
    "buildingType",
    "owner",
    "occupier"
  ],
  "timesSent": 12,
  "lastNotification": "2015-10-05T10:18:16Z",
  "expires": "2016-04-05T14:00:00.00Z",
  "status": "active",
  "throttling": 5
}
```

4.18 Create a new subscription

HTTP Verb: `POST`

Resource URL: `/v2/subscriptions`

Request Parameters

None

Request body example

```
{
  "description": "One subscription to rule them all",
  "subject": {
    "entities": [
      {
        "id": "0d6c64ef-4df9-49ce-8e8e-d73901559df1",
        "type": "Building"
      }
    ],
    "condition": {
      "attrs": [
        "buildingType"
      ],
      "expression": {
        "q": "buildingType=house"
      }
    }
  },
  "notification": {
    "http": {
      "url": "http://www.gsma.com/iotbd"
    }
  },
  "attrsFormat": "keyValues",
  "attrs": [
    "buildingType",
    "owner",
    "occupier"
  ],
  "timesSent": 12,
  "lastNotification": "2015-10-05T16:00:00.00Z",
  "expires": "2016-04-05T14:00:00.00Z",
  "status": "active",
  "throttling": 5
}
```

Response Parameters

Parameter	**Type/ Meaning**	**Required/ Optional**
`Location` (HTTP header)	string URL which points to the created subscription	required

HTTP Response Code: `201` (Created)

Response example

```
Location:/v2/subscriptions/0b06aff5-b476-48e5-9465-7cd8a2671166
```

4.19 Retrieve subscription

HTTP Verb: `GET`

Example Resource URL: `/v2/subscriptions/subscriptionId`

Request Parameters

Parameter	**Meaning**	**Type**

`subscriptionId`	Id of the subscription to be retrieved	String

Response Parameters

Parameter	**Type/ Meaning**	**Required/ Optional**
`<subscription>`	Returns the subscription information matching the specified ID.	Required

Example URL:

```
/v2/subscriptions/0b06aff5-b476-48e5-9465-7cd8a2671166
```

HTTP Response Code: `200` (Success)

Response example

```
{
  "id": "0b06aff5-b476-48e5-9465-7cd8a2671166",
  "description": "One subscription to rule them all",
  "subject": {
    "entities": [
      {
        "id": "0d6c64ef-4df9-49ce-8e8e-d73901559df1",
        "type": "Building"
      }
    ],
    "condition": {
      "attrs": [
        "buildingType"
      ],
      "expression": {
        "q": "buildingType=house"
      }
    }
  },
  "notification": {
    "http": {
      "url": "http://www.gsma.com/iotbd"
    }
  },
  "attrsFormat": "keyValues",
  "attrs": [
    "buildingType",
    "owner",
    "occupier"
  ],
  "timesSent": 12,
  "lastNotification": "2015-10-05T16:00:00.00Z",
  "expires": "2016-04-05T14:00:00.00Z",
  "status": "active",
  "throttling": 5
}
```

4.20 Update subscription

HTTP Verb: `PATCH`

Resource URL: `/v2/subscriptions/subscriptionId`

Request Parameters

Parameter	Meaning	Type
`subscriptionId`	Subscription id to be updated	String

Example URL:

```
/v2/subscriptions/0b06aff5-b476-48e5-9465-7cd8a2671166
```

Request body example

```
{
  "expires": "2016-04-05T10:18:16Z",
}
```

Response Parameters

None

HTTP Response Code: `204` (No content)

4.21 Delete subscription

HTTP Verb: `DELETE`

Resource URL: `/v2/subscriptions/subscriptionId`

Request Parameters

Parameter	Meaning	Type
`subscriptionId`	Subscription to be updated	String

Example URL:

```
/v2/subscriptions/0b06aff5-b476-48e5-9465-7cd8a2671166
```

Response Parameters

None

HTTP Response Code: `204` (No content)

4.22 Update entities (batch)

HTTP Verb: `POST`

Resource URL: `/v2/op/update?options=options`

Request Parameters

Parameter	Meaning	Type
`actionType`	Specifies the kind of update action to do: either **`APPEND`**, **`APPEND_STRICT`**, **`UPDATE`**, **`DELETE`**.	String
`entities`	An array of entities, each one specified using the JSON entity representation format	Array of objects
`options`	Options dictionary with possible values • **`keyValues`**	String

Example URL:

`/v2/op/update?options=keyValues`

Request body example

```
{
  "actionType": "APPEND",
  "entities": [
    {
      "id": "7b014918-7ccc-4d2f-af65-6cdfe3690be4",
      "type": "AgriSoil",
      "dateCreated": "2016-08-08T10:18:16Z",
      "dateModified": "2016-08-08T10:18:16Z",
      "source": "www.samplefarmproduct.com",
      "dataProvider": "Text",
      "name": "Clay"
    },
    {
      "id": "35ac1609-6b31-4cad-8f4c-ad8be828c8a2",
      "type": "AgriSoil",
      "dateCreated": "2016-08-08T10:18:16Z",
      "dateModified": "2016-08-08T10:18:16Z",
      "source": "www.samplefarmproduct.com",
      "dataProvider": "Text",
      "name": "Silt"
    }
  ]
}
```

Response Parameters

None

HTTP Response Code: `204` (No content)

4.23 Query entities (batch)

HTTP Verb: `POST`

Resource URL: `/v2/op/query?limit=10&offset=20&options=options`

Request Parameters

Parameter	Meaning	Type
`limit`	Limit the number of entities to be retrieved. Example: 10.	Number
`offset`	Skip a number of records. Example: 20.	Number
`orderBy`	Criteria for ordering results. See "Query Specification" section for details. Example: temperature,!speed.	String
`options`	Options dictionary with possible values • **`count`** • **`keyValues`** • **`values`** • **`unique`**	String

Example URL:

`/v2/op/query?limit=2&options=keyValues`

Request body example

```
[
  {
    "idPattern": "agriSoil.*",
    "type": "AgriSoil"
  }
]
```

Response Parameters

Parameter	Type/ Meaning	Required/ Optional
<entityArray>	Array of objects Array of the matched entities present in the system	required

HTTP Response Code: `200` (Success)

Response example

```
[
  {
    "id": "7b014918-7ccc-4d2f-af65-6cdfe3690be4",
    "type": "AgriSoil",
    "name": "Clay"
  },
  {
    "id": "35ac1609-6b31-4cad-8f4c-ad8be828c8a2",
    "type": "AgriSoil",
    "name": "Silt"
  }
]
```

Annex A NGSIv2 Roadmap

There are three major work items on the short term roadmap of FIWARE NGSI version 2:

- NGSI-9 interfaces to enable context provider registration and discovery. An initial discussion has been launched by some community members [9].
- Support JSON-LD to improve how linked data is represented and to have improved interoperability with RDF and other graph formats. This work item has been discussed deeply [10] during 2016 and a consensus will be reached very soon.
- Extend the API to support short term historic queries.

Annex B Document Management

B.1 Document History

Version	Date	Brief Description of Change	Approval Authority	Editor / Company
0.10	8 Sept 16	New PRD - first draft	PSMC	Allan Bartlett / GSMA
1.0	11 Oct 16	Approved first version	PSMC	Allan Bartlett / GSMA

B.2 Other Information

Type	Description
Document Owner	IoT Big Data Ecosystem Project
Editor / Company	Allan Bartlett / GSMA

It is our intention to provide a quality product for your use. If you find any errors or omissions, please contact us with your comments. You may notify us at prd@gsma.com

Your comments or suggestions & questions are always welcome.