GOBIERNO ONTOLOGÍAS SOFIA2

DICIEMBRE 2014

Versión 2

indra

1 INDICE

1 INDICE 2
2 INTRODUCCIÓN 3
3 ENTIDADES SOFIA2 4
3.1 ASSETS 4
3.2 ONTOLOGÍAS 5
4 NOMENCLATURA 6
5 TIPADO Y FORMATOS 7
6 PLANTILLAS PREDEFINIDAS 9
6.1 FEEDS (MEDIDAS) 9
6.2 COMMANDS (COMANDOS) 12
6.3 ALERTAS 13
6.4 EVENTOS 14
6.5 AUDIT 15

2 INTRODUCCIÓN

La Plataforma Sofia2 ofrece todos los mecanismos necesarios para interconectar cualquier tipo de red de sensores ofreciendo diversos mecanismos para ello.

El presente documento, en aras de establecer una base común de trabajo para todas las aplicaciones y verticales, establece una serie de **recomendaciones** sobre reglas y procedimientos para la utilización de los mismos.

El objetivo es que **a priori** los desarrolladores y sistemas para explotar la información conozcan las estructuras de datos y tipos que maneja la Plataforma y **puedan hacer un uso homogéneo de los mismos independiente de la implementación concreta y particular de los desarrollos de integración**.

Se recomienda encarecidamente la lectura previa de la documentación de referencia para entender todos los conceptos manejados en el presente documento.

Plataforma obtiene, registra y gestiona toda la información de carácter sensorial (medidas, comandos, alertas, eventos, etc…) usando **ontologías json** para la representación de los datos.

Dado que los mecanismos ofrecidos por Sofia2 para la gestión de ontologías son muy flexibles y adaptables, para **garantizar que toda la información en la Plataforma tiene una estructura homogénea entre los diferentes aplicaciones y verticales**, y que permite **hacer un uso horizontal de toda la información generada**, se han establecido las reglas de gobernanza fijadas en el presente documento.

3 ENTIDADES SOFIA2

La Plataforma sofia2 gestiona dos tipos de entidades:

- **Assets**. Un asset es todo elemento (físico o virtual) capaz de generar o consumir información de carácter sensorial y gestionarla a través de la Plataforma SOFIA2
- **Ontologías**. Los assets generan información y dicha información se modela por medio de ontologías (JSON) en la Plataforma.

3.1 Assets

Dependiendo de su naturaleza y de cómo se conectan dichos assets a la Plataforma se establece una clasificación de los mismos en:

- **Assets del Inventario** de la Plataforma. Son Assets directamente gestionados por la Plataforma por lo cual se debe de conocer a priori toda la información sobre los mismos que permita conectarse a ellos directamente. **Todos los assets de inventario tienen que estar dados de alta en el Inventario de Assets de la Plataforma**. Se dividen a su vez en dos tipos:
    - **Managed.** Se trata de los Assets conectados a la Plataforma es decir, gestionados por los KPs. Su característica principal es que los KPs que los gestionan necesitan conocer toda la información de acceso físico a los mismos.
    - **Unmanaged.** Se trata de Assets conocidos en el inventario de la Plataforma, que generan y consumen información de la misma, pero para los cuales no es responsabilidad del módulo de interconexión su gestión final, es decir, son gestionados por otros sistemas de información que se integran en la Plataforma.
- **Assets Virtuales**. Los assets virtuales son a priori no conocidos, generan información y esta llega a la Plataforma a través de integraciones, normalmente contra otros servicios de información en la nube, como por ejemplo las redes sociales.

3.2 Ontologías

Los assets conectados generan y consumen información sensorial en forma de ontologías json.

A la hora de crear una ontología se pueden usar los catálogos (plantillas) siguientes:

- **Feed** (medidas). Se trata de la información de medida emitida por los assets conectados. Pueden ser medidas instantáneas, agregaciones por períodos de tiempo, cálculos, etc… Un feed puede recoger varias medidas simultáneamente y **todos los feeds deben de estar georreferenciados** (puntos, líneas o áreas) pudiendo ser éstos móviles.
- **Command** (comandos u órdenes). Se trata de las **órdenes enviadas a los assets** con capacidades de actuación. Dichas órdenes son asíncronas (no existe respuesta inmediata) por lo que existen dos tipos de comandos:
    - **Comandos request.** Las instrucciones enviadas a un Asset.
    - **Comandos response**. Las respuestas o ACKs enviadas por los Assets a dichos comandos. Las respuestas a los comandos únicamente indican la confirmación de la recepción de dicha instrucción.
- **Alert** (alertas). Las alertas son mensajes de notificación de situaciones (alertas, incidencias, mensajes, notificaciones, etc…) y tienen la particularidad de que **su estado así como su nivel de criticidad cambia a lo largo del tiempo** desde un mensaje inicial que genera la alerta hasta el cierre de la misma. Las alertas no se conocen a priori, es decir, se generan en el momento en el que suceden.
- **Schedule** (eventos). Los eventos reflejan situaciones conocidas en el tiempo y en el espacio, es decir, ocurren durante un periodo de tiempo en un lugar concreto y, normalmente, se conocen a priori.
- **Audit** (log). Los mensajes de auditoría o log son internos a la Plataforma y se utilizan para hacer explícitas y dejar gestionadas situaciones concretas particulares de los Assets. Por ejemplo, el encendido de una farola, el mal funcionamiento de un sensor, etc...
- **KPI** (indicador). Los indicadores son un tipo especial de medida y su característica principal es que no son generados por un único Asset. Se trata de cálculos realizados sobre múltiples datos, series históricas e incluso capas de datos espaciales que se almacenan como medidas en Plataforma.

Reglas especiales a tener en cuenta:

- Cada vez que se conecta un nuevo conjunto de assets (del mismo tipo) a la Plataforma deben de darse de alta al menos las ontologías **feed** y **audit** y si corresponde **command** para el mismo.
- Inmediatamente a la recepción de un comando y su correcta ejecución el Asset (el KP que lo gestiona) debe de generar una nueva medida para reflejar en la Plataforma su nuevo estado.

4 NOMENCLATURA

A continuación se establecen las reglas básicas de nomenclatura:

- Se utilizarán nombres en inglés y se seguirá el estándar Java (aka “camel”).
- Para la definición de plantillas:
  - Nombres cortos, autoexplicativos. Primera letra en mayúsculas. Las primeras letras identifican el tipo de ontología:
    - Feed: **Feed**
    - Command: **Cmd**
    - Alert: **Alrt**
    - Schedule: **Schdl**
    - Audit: **Adt**
    - KPI: **Kpi**
- Para la definición de **ontologías**:
  - Primera letra en minúscula, empiezan siempre por el tipo de ontología. Ejemplo para una Espira: feedEspira, cmdEspira, adtEspira …
- Para la definición de **atributos** de las ontologías:
  - Primera letra en minúsculas, sin espacios, sin caracteres especiales.
- Para la definición de **constantes** utilizadas como valores posibles para los atributos de las ontologías: todas las letras en mayúsculas. Por ejemplo: MOBILE, FIXED, VIRTUAL.

5 TIPADO Y FORMATOS

Para la definición de ontologías se utilizarán cadenas de texto UTF-8 siguiendo el esquema json establecido por la correspondiente plantilla (actualmente siguiendo JSON Schema 0.4 http://json-schema.org/draft-04/schema#).

A continuación se establecen las reglas de tipado y formato para los diferentes tipos soportados:

- **UUIDs**:
  - Cadena de texto. Standard Universally Unique Identifier.
- **Números enteros**:
  - Entero Largo de 64 bits
  - Ejemplo: {'contador' : 10}
- **Números flotantes**:
  - Notación simple. Decimal con punto. 64 bits
  - Ejemplo: {'valor' : 10.5}
- **Cadenas de texto**:
  - Cadena de texto. UTF-8. Caracteres especiales escapados
  - Ejemplo: {'comment' : 'next station'}
- **URLs y URIs**:
  - Cadena de texto. Codificadas siguiendo estándar RFC-1738
  - Ejemplo: {'url' : 'http%3A%2F%2Fwww.coruna.es%2Fmedioambiente%2F'}
- **Timestamps**:
  - Fecha. Cadena de texto siguiendo formato ISO-8601. RFC 3339
  - Objeto conteniendo atributo “$date”
  - Ejemplo: {"timestamp":{"$date":"2014-01-27T11:14:00Z"}}
- **Fechas e intervalos de fechas**:
  - Cadena de texto siguiendo formato ISO-8601.RFC 3339
  - Objeto con atributo "$date"
  - Ejemplo de fecha: {"created":{"$date":"2014-01-27T11:14:00Z"}}
  - Ejemplo de intervalo entre dos fechas: {“period”:{“$date” : “2010-07-02T11:44:09Z/2010-07-02T11:47:00Z”}}
- **Direcciones**:
  - Notificación simplificada para facilitar las tareas de integración:

```
{"address": {
            "location": "cadena de texto",
             "number": "cadena de texto"
            }
}
```

- **Unidades de medidas**
  - Cadena de texto string siguendo notación **JScience library** (http://jscience.org/api/javax/measure/unit/SI.html)(http://jscience.org/api/javax/measure/unit/NonSI.html)
  - Ejemplo {'unit': 'A'} # Amperios
- **Coordenadas geográficas**:
  - Siguen la definición **OGC GeoJson**. Esquema de coordenadas **WGS84**. No hay coordenada Z. Orden [longitud, latitud]
  - **Puntos**:
    - GeoJson Point
    - Ejemplo:

```
{"geometry": {
              "type": "Point",
             "coordinates": [-8.410161625142807, 43.360463863501934]
        }
}
```

  - **Líneas**:
    - GeoJson LineString
    - Ejemplo:

```
{"geometry": {
              "type": "LineString",
             "coordinates":[
                            [-8.410161625142807, 43.360463863501934],
                            [-8.410161625142807, 43.360463863501978]
                            ]
        }
}
```

  - **Áreas**:
    - GeoJson Polygon
    - Ejemplo:

```
{"geometry": {
              "type": "Polygon",
             "coordinates":[
                           [[-8.410161625142807, 43.360463863501934],
                            [-8.410161625142807, 43.360463863501978],
                            [-8.41016162514290, 43.360463863501978],
                            [-8.410161625142807, 43.3604638635019341]
                           ]
        }
}
```

[NOTA]: Para cerrar el polígono el primer y el último valor de cada anillo deben de ser idénticos.

[NOTA]: Un polígono puede tener 2 anillos (el exterior y el interior).

6 PLANTILLAS PREDEFINIDAS

6.1 Feeds (Medidas)

Para la definición de la plantilla de medidas se utiliza una simplificación del estándar de datos AMON (**http://amee.github.io/AMON/**):

```
{"Feed":
       {
       "asset":
              {
               "assetId" : string, (required)
               "assetType": string, (required)
               "assetSource": string, (required)
               "assetName": string (optional)
              },
       "type": string, (required) [FIXED, MOBILE, VIRTUAL]
       "timestamp": (required)
                    {
                       "$date": "RFC 3339 DATETIME"
                     },
       "attribs": (optional)
                   [
                    { "name": "value" }
                   ],
       "geometry":  geojson [Point, LineString, Polygon], (optional)
       "measures": (required)
                   {
                      "timestamp" : (required)
                                    {
                                     "$date": "RFC 3339 DATETIME"
                                    },
                      "type"      :  string, (required) [INSTANT, CUMULATIVE, PULSE]
                      "period"    :  number, (optional)
                      "periodUnit":  string, (optional) [m, s, h, d]
                      "values"    : (required)
                                    [
                                      {
                                           "name": string, (optional)
                                           "desc": string, (optional)
                                           "unit": string, (required)
                                           "measure": string, (required)
                                           "method": string,(required)
                                           "modifications": (optional)
                                                  [ {
                                                    "oldMeasure": string, (required)
                                                    "changeTimestamp": (required)
                                                            {
                                                             "$date": "RFC 3339 DATETIME"
                                                            }
                                                    "changeDesc": string, (optional)
                                                    }
                                                  ]
                                      }
                                    ]
                   }

       }
}
```

El objeto **asset** hace referencia al activo que emite la medida:

- **assetId**: identificador del activo en el sistema de referencia que lo gestiona (establecido en el campo assetSource).
- **assetType**: tipo de asset (farola, sensor de humedad, etc…)
- **assetSource**: sistema de información que gestiona el activo.
- **assetName**: atributo opcional para asociar un nombre al activo si se considera necesario.

El tipo de sensor (**type**) hace referencia a su naturaleza, los tipos validos son:

- **FIXED**. Sensores a priori conocidos (gestionados por un inventario conocido) posicionados geograficamente en una posición fija conocida.
- **MOBILE**. Sensores a priori conocidos que se mueven y su posición se actualiza en cada medida.
- **VIRTUAL**. Sensores a priori no conocidos (por ejemplo redes sociales).

El **timestamp** referencia la **fecha y hora de captura del feed**.

> **NOTA1**: no confundir con el timestamp que se genera automáticamente al enviar la ontología al módulo de interconexión
>
> **NOTA2**: no confundir con el timestamp de las medidas que referencia el momento de recogida de las mismas).

El objeto de **atributos** (attribs) tiene por objeto recoger una lista arbitraria de atributos modelados en forma clave:valor. Su utilidad puede ir desde recoger claves secundarias hasta almacenar cualquier atributo adicional necesario.

El objeto **geometry** recoge la posicion (punto, linea o poligono) a la que referencia el feed. **En todo caso, siempre que la posicion del feed sea conocida debe de figurar en el feed** independientmente de que la misma se encuentre dada de alta en el inventario. En caso de no conocerse la posición el atributo no debe de figurar en el feed.

El objeto **measures** hace referencia a las características comunes de referencia de todas las medidas capturadas y lista todas las medidas realizadas:

- **timestamp**: fecha de referencia de realización de las medidas
- **type**: tipo de medida realizada: medidad instantánea, acumulado, pulso
- **period**: si procede, período de tiempo utilizado para el cálculo de las medidas.
- **periodUnit**: unidad de tiempo (‘s’, ’m’, ‘h’, ‘d’) utilizada para definir el período de tiempo.
- **values**: lista de medidas realizadas
  - **name**: si procede, nombre representativo de la medida
  - **desc**: si procede, descripción de la medida realizada

- **unit**: unidad de medida
- **measure**: valor de la medida en su versión más actualizada. Es decir, el atributo measure contendrá siempre la medida válida. En caso de realizarse modificaciones de la medida los valores históricos serán almacenados en la lista del atributo modifications.
- **method**: método utilizado para obtener la medida (media, min, max, etc…)
- **modifications**: lista de modificaciones realizadas sobre la medida originalmente capturada

6.2 Commands (Comandos)

Para la definición de los comandos se utiliza:

```
{"Command":
            {
            "commandId":    string, (required)
             "asset":
                {
                 "assetId" :    string, (required)
                 "assetType":   string, (required)
                 "assetSource": string, (required)
                 "assetName":   string  (optional)
                },
             "timestamp": (required)
                          {
                            "$date": "ISO 3339 DATETIME"
                          },
             "desc": string, (optional),
             "type": string, (required) [REQUEST, RESPONSE],
             "command":
                      {
                       "type": string, (required) [SWITCH, DIM, SET, EXECUTE, SEND]
                       "value1": string, (optional)
                       "value2": string, (optional)
                       "value3": string, (optional)
                       "msg":    string  (optional)
                       }
              "rule":
                     {
                     "type":       string, (required) [ASAP, DATE]
                     "date":       (optional)
                            {
                             "$date": "ISO 3339 DATETIME"
                            }
                     }
               }
}
```

6.3 Alertas

Para la definición de las alertas se utiliza una simplificación del estándar **CAP 1.2** (**http://docs.oasis-open.org/emergency/cap/v1.2/CAP-v1.2-os.html**)

```
{"Alert":
       {
        "id" : {
                "alertId": string, (required)
                "alertSource":  string  (required)
               },
        "timestamp":{
                     "$date": "ISO 3339", (required)
                    },
        "asset":
                {
                 "assetId" : string, (required)
                 "assetType": string, (required)
                 "assetSource": string, (required)
                 "assetName": string (optional)
                },
        "alert":
               {
                "sourceAlertId": identifier,
                "subject":       string required,
                "description":   string optional,
                "source":        string required,
                "type":          [ALARM, WARNING, MESSAGE, NOTIFICATION, INFO],
                "status":        [OPEN, CLOSED, UNKNOWN],
                "affectedLocations": (optional)
                         [
                          {
                          "desc":        string, (optional)
                          "geometry":    geojson,(optional) [Point, Line, Polygon],
                          "locationUri": string, (optional)
                          }
                         ]
                }
         "info":
               {
                "action":          [CREATE, CLOSE, UPDATE, ACK,
                                    FOLLOW, SCALATION, REMINDER, CANCEL],
                "sender":         string, (required)
                "contact":        string, (optional)
                "description":    string, (optional)
                "parameters":     string, (optional)
                "urgency":        [EXPECTED, FUTURE, INMEDIATE, PAST, UNKNOWN],
                "severity":       [EXTREME, MINOR, MODERATE, SEVERE, UNKNOWN],
                "certainty":      [LIKELY, OBSERVED, POSSIBLE, UNLIKELY, UNKNOWN],
                "resources": optional
                           [
                            {
                             "name":        string, (required)
                             "description": string, (optional)
                             "uri":         string, (required)
                             "mimeType":    string  (optional)
                            }
                           ]
               }
       }
}
```

6.4 Eventos

La definición de eventos sigue el siguiente esquema.

```
{"Event":
       {
        "id":
              {
               "eventId": string, (required)
               "eventSource": string (required)
              },
       "timestamp":{
                    "$date": "ISO 3339", (required)
                   },
        "asset":
                 {
                  "assetId" : string, (required)
                  "assetType": string, (required)
                  "assetSource": string, (required)
                  "assetName": string (optional)
                 },
       "eventInfo":
              {
              "subject":          string, (required)
              "description":      string, (optional)
              "type":             string,[INFO, PROGRAM, EVENT]
              "affectedLocations": (optional)
                     [
                            {
                            "desc":         string, (required)
                            "geometry":     geojson optional [Point, Line, Polygon],
                           "locationURI":   string optional
                            }
                     ],
              "resources": (optional)
                     [
                            {
                            "name":              string, (required)
                            "description":      string, (optional)
                            "uri":              string, (required)
                            "mimeType":          string  (optional)
                            }
                     ]
              }
       "eventRule":
              {
              "type":             [SINGLE, PERIOD, RULE],
              "period": (required)
                       {
                        $date: "RFC 3339 INTERVAL"
                       },
              "repeatEach":       entero, (opcional)
              "repeatUnit":       string (opcional) [s, m, h, d, w, m]

       }
}
```

6.5 Audit

La definición de mensajes de auditoria sigue el siguiente esquema.

```
{"Adt":
       {
       "id" : {
              "auditId": string, (required)
              "auditSource":  string  (required)
              },
       "timestamp": (required)
                    {
                     "$date": "ISO 3339 DATETIME"
                    },
       "asset":
             {
             "assetId" :        string, (required)
             "assetType":       string, (required)
             "assetProvider":   string, (required)
             "assetName":       string  (optional)
             },
      "message": {
             "source":  string required,
             "sender":  string optional,
             "subject": string required,
             "body":    string optional,
             "level":   [INFO, WARNING, ERROR, DEBUG]
      }
}
```