IoT Semantic Interoperability: a Pragmatic Approach

Submitted to: IAB IoT Semantic Interoperability Workshop 2016

Author: Milan Milenkovic, milan@intel.com

1 Introduction

McKinsey report estimates that achieving interoperability in IoT would unlock additional 40% of the total available market (TAM). Conversely, failure to interoperate shrinks IoT TAM by 40%. This paper discusses requirements and implementation principles for data and meta-data interoperability in IoT systems. The target is to achieve semantic or, as IIC refers to it, conceptual interoperability [IIC IOT ref arch, p68], i.e. represent information in a form whose meaning is independent of the application generating or using it. When supported, semantic interoperability achieves two important objectives (1) enables service-level integration of IoT end-to-end systems constructed using components from different vendors, such as a variety gateways running different middleware, with a third-party cloud for data storage, processed by analytics from independent vendors, and (2) it allows aggregation of data from different domains, such as disparate systems in smart cities, to allow for holistic management and – more importantly – to enable big data through aggregation of large data sets that are understandable, and thus usable, to analytics and other services.

Semantic data, as originally defined by Tim Berners-Lee, is the information that allows machines to understand the meaning of data. In this paper, we take a more pragmatic interpretation of semantic data as the well-structured meta-data that annotates the context and meaning of associated sensor data for the purposes of enabling portable analytics and applications.

Interoperability does not mandate the use of common data and meta-data formats in different systems, but it does require sufficient specification and adherence to some basic design principles. There are many levels and interpretations of interoperability, including transport and protocol layers, data and meta-data definition layer, device-device level, device-cloud level, service level, run-time or design/compile time. In this paper, we are referring to service-level semantic interoperability, which is arguably the most common IoT cross-domain requirement and comparatively the easiest form of interoperability to achieve. It can be implemented at the data-exchange level between systems, thus enabling aggregations of useful big data sets. Semantic interoperability here means a common data and meta-data format that applications and services obtain as responses to their sensor database queries/APIs – format that they can understand and use, regardless of differences in encoding at the data-capture and transport protocol levels. In operational terms, interoperability can be embodied as the tool to perform (automated) machine translation of data and meta-data formats across systems with different data models.

2 Metadata: the often neglected one

Meta-data or tags annotate data reported by sensors to provide context, such as sensor type, function, location, data format and capture time. Its primary function is to provide contextual semantics to create "rich data" (basically data made useful) for a variety of post-processing services and applications, such as analytics and device/asset management. It also enables searches of annotated data by attribute such as type, owner, location, reading value(s), time period. As a minor

illustration of its diversity and range, below is a partial list of IoT sensor data and meta-data (bulleted items) structured into expandable categories (*italicized*) from a project.

Identification (unique end-to-end reference ID/key for reporting, retrieval, meta-data association)

- Name, identifier (URI or something mappable to it)

Description, capability (for discovery and optional run-time binding)

- Type, Capability, Configuration - sensor connectivity
- Network connection, Communication protocol, Data encoding

Sensed values reading and writing (actuation)

- Reading, value
- Time stamp
- (reading) Data type, Units of measurement
- Range - Min, max values
- Resolution, Sampling Rate
- Frequency of reporting
- Scaling factor
- Writing, value (for actuators)

Location

- Location – static or mobile (if sensor mobile report location with each reading)

Maintenance, asset management

- Status = {active, inactive, connected, fault}
- Battery status
- Accuracy, Calibration
- Manufacturer, part#, serial#
- Configuration settings
- Firmware, Software version installed

Access rights, privacy, security

- Owner/domain, Associations
- Access rights, Privacy restrictions
- Reputation

Other, extensions,…

Basic sensor data-reporting format typically includes:
<sensor name/identifier to (to associate report with the originator)>, <sensor data value reading>, <reading time (stamp)>, <(changed) meta-data>

Meta-data tends to be static or to change much less frequently than the associated data. This fortunately reduces the size of end-point reports as only the meta-data that changed since the last reported value needs to be included. In practice, we found it useful to establish a shared contextual reference of meta-data entries and values between the edge (gateway or smart sensor) and the cloud. This can be established at the system configuration and provisioning time, and communicated only when changes occur or repeated in its entirety when recovering from catastrophic failures or connecting to a different back-end system or cloud.

Great and ever growing variety of sensor types and their associated data and meta-data, even within individual vertical domains, makes it infeasible to design an exhaustive set of data and meta-data formats. The pragmatic approach is to define a limited subset of the most used data and meta-data

fields, and then provide a method for future and/or domain-specific extensions that are architecturally compliant.

3 Towards Interoperable Solution - Design Principles and Guidelines

At least one standards group will focus on semantic interoperability, an excerpt from its charter states "The group will focus on accomplishing interoperability across different information models developed by IPSO and other organizations. It will attempt to do so by developing and promoting a "meta" information model that represents key inherent properties and relationships of the physical entities that are being modelled. The expectation is that this "meta" model will serve as a universal data and meta-data exchange layer that can be mapped to specific bindings and representations of smart objects.

The group will perform this work in three mostly consecutive phases:

(1) Definition of architectural principles and design guidelines for constructing interoperable semantic data models
(2) Definition of meta-model for describing semantic properties for interoperability
(3) Mapping of meta-model to specific bindings, such as LWM2M (IPSO gen 1) and OIC."

Based on several implementations of IoT systems and lessons from the Internet, it appears that there are some basic architectural principles and design guidelines for data and meta-data format definition that facilitate system-level semantic interoperability. They include:

- Identify minimal specification necessary to achieve interoperability (expand later if needed in a manner that preserves it)
- Persistent URIs for naming, or a naming system that can be correlated to those
- Use machine/human readable formats, such as JSON, XML (to binary for small devices, links)
- Represent attributes (data and meta-data) and their values as name, value pairs
    - popular key, value and common encodings like JSON are a very good fit for that
- Flat - do not impose or assume a hierarchy or specific object (grouping) implementation, structural relationships can be represented as links, e.g. in JSON LD notation
- Allow flexibility as to what is used in a given system, i.e. do not prescribe specific data and meta-data types to be used, *but*
    - ensure consistency across implementations, i.e. when a specific (m) data type is used it follows a consistent documented format (within a system and is thus machine-convertible across systems)
- (desirable) helps if key/attribute fields are consistently named within a domain – this essentially encodes semantic meaning and is conducive to machine-parsing in portable apps
    - Haystack provides a good example on how this can be done in (buildings) domain.

Note that these principles are for achieving interoperability when exchanging data between systems and components. One of the key insights about interoperability is that – due to different design decisions in individual systems - it is almost impossible to achieve at the data-structuring level, such as hierarchical object definitions. Smart objects are a powerful abstraction within a compliant system, but interoperability is much easier by communicating their individual attributes and values through a flat non-hierarchical manner and communicated through a mechanism, such as linked data, thus allowing the receiving system to map those to its own internal object representation.