O-MI/O-DF Standards as Interoperability Enablers for Industrial Internet: a Performance Analysis

Jeremy Robert*, Sylvain Kubler*, Yves Le Traon*, Kary Främling†
*University of Luxembourg – Interdisciplinary Centre for Security, Reliability and Trust, Luxembourg
{*firstname.lastname*}@uni.lu
†Aalto School of Science and Technology, Finland
{*firstname.lastname*}@aalto.fi

***Abstract*—The Industrial Internet should provide means to create ad hoc and loosely coupled information flows between objects, users, services, and business domain systems. However, today's technologies and products often feed 'vertical silos' (e.g., vertical/siloed apps), which inevitably result in multiple and non-interoperable systems. Standardization will play an ever-increasing part in enabling information to flow between such vertically-oriented closed systems. This paper presents recent IoT messaging standards, notably O-MI (Open Messaging Interface) and O-DF (Open Data Format), whose initial requirements were defined for enhanced collaboration and interoperability in product lifecycle management. The performance of those standards is evaluated in terms of efficiency ratio, defined as the percentage of payload over traffic load. A first analytical model of the efficiency ratio based on the required/basic standard specifications is then proposed. A smart maintenance use case relying on the first version of the standard reference implementation is developed, based on which our analytical model is applied to evaluate the degree of deviation (w.r.t. the standard specifications) of this reference implementation.**

***Index Terms*—Industrial Internet; Industry 4.0; Interoperability; Networking; Product Lifecycle Management.**

I. Introduction

OVER the past decade, a flourishing number of concepts and architectural shifts appeared such as the Internet of Things (IoT), Cyber-Physical Systems (CPS), or Industrial Internet. As a rapidly growing area, the Industrial Internet has become a technological focus area for academia, industry, and governmental organizations [1]. It is not a new technology, it is simply a 'catch-all' term for existing technologies and disciplines applied in an industrial setting, such as Machine-to-Machine (M2M) protocols, IoT/CPS systems[1], cognitive science, Big Data technologies, *etc.* [2]. Those disciplines interact and cooperate together, from the collection of human- and machine-generated data, to its storage and analysis, leading to decision making and the resultant system behavior. Currently, the Industrial Internet consortium is essentially driven by US enterprises, meanwhile in Europe similar initiatives have different names: Industry 4.0 in Germany, 'Smart Factory' in the Netherlands, 'Usine du Futur' in France, *etc.* For consistency purposes, the term "Industrial Internet" is used.

Industrial Internet, considered either as an extension or subpart of the IoT, envisions a world of heterogeneous objects uniquely identifiable and accessible through the Internet [3], the whole forming a dynamic global network infrastructure with self configuring capabilities. Ideally, the Industrial Internet should provide means to create ad hoc and loosely coupled information flows between any kinds of objects, users, systems, when and as needed. However, while new smart and connected products hit the market every day, they mostly feed 'vertical silos' (e.g., vertical apps, siloed apps...), often resulting in non-interoperable and proprietary systems, which are expensive to train on and maintain [4]. In lack of standardized solutions, it is likely that a proliferation of such vertically-oriented closed systems will develop side by side, each one dedicated to a particular or separate use [5]. To date, numerous organizations and/or consortiums such as IEEE, W3C, ETSI, ITU, AIOTI, The Open Group, understood this problem and have thus undertaken standardization efforts and programs [6].

The vertical silo problem is discussed further in section II, along with existing IoT communication models and messaging standards. Recent IoT standards, named O-MI (Open-Messaging Interface) and O-DF (Open-Data Format), are introduced in section III, for which the performance is evaluated in terms of efficiency ratio. To this end, an analytical model based on the required/basic standard specifications is developed. A smart maintenance use case relying on the first version of the standard reference implementation is developed in section IV, based on which our analytical model is applied for evaluating the deviation of that implementation with respect to the standard specifications; the conclusion follows.

II. Vertical Silo & IoT communication models

Industrial environments are complex ecosystems, with a wide range of interacting and cooperating actors such as manufacturers, suppliers, machine and infrastructure providers, as well as a heterogeneity of digital services oriented to the well-functioning of the company [7]. Sections II-A and II-B respectively discusses the vertical silo problem and the existing IoT communication models.

A. The "Vertical Silo" problem

Current M2M manufacturers have been integrating Internet-connected systems for high-value asset tracking, product lifecycle management (PLM), fleet management, *etc.*, for more than 15 years [8]. These M2M systems are challenging to

[1]In this paper, IoT and CPS are used interchangeably.

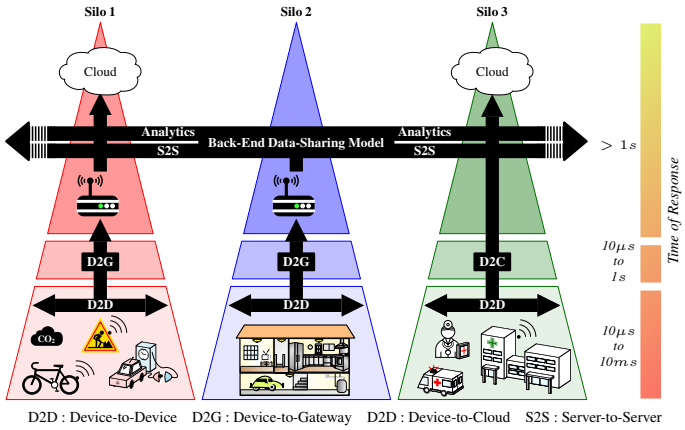


Fig. 1. Vertical Silo problem & IoT communication models (*cf.* RFC 7452)

build even though some are based on standard industrial protocols such as OPC UA (Open Platform Communications Unified Architecture) or SCADA (Supervisory Control and Data Acquisition). Systems may use the exact same protocols but the communication layers are still inconsistent. Although some systems use application programming interfaces (APIs), they tend to be proprietary, thus hindering cross-application and service integration in industrial settings. In fact, today's reality is that the evolution of the Industrial Internet is tightly dependent on the evolution of the IoT sector. Unfortunately, there are still challenges ahead, and particularly the vertical silos that are a serious impediment for developers to produce new added value across multiple platforms due to the lack of interoperability and openness. This issue is highlighted in Fig. 1 through the different pyramids, where data is pushed and "siloed" in a unique system, which is closed to the rest of the IoT. Although the most striking examples of 'vertical silos' are the major providers of Cloud-based IoT services such as Google Cloud Platform, Amazon Web Services, *etc.*, the problem is not limited to cloud-only applications, but also to domain-specific systems. For example, home automation systems are often unable to communicate/cooperate with healthcare or manufacturing systems, and *vice-versa* [4].

Moving towards more collaborative, open and ecosystem-based service models in the Industrial Internet is of the utmost importance and should mark a new turning point for radical transformations in business dynamics. However, to make this a reality, it is necessary to unlock the potential of the IoT paradigm by enabling horizontal interoperability across vertically-oriented closed systems, as illustrated in Fig. 1 (see *"Back-End Data-Sharing Model"*).

B. IoT communication models

From an operational perspective, there are different IoT communication models to enable devices to connect and communicate with each other and backend systems. In March 2015, the Internet Architecture Board (IAB) released a guiding architectural document [9] for networking of smart objects, which outlines four common IoT communication models. Those models are depicted in Fig. 1, namely:

- *Device-To-Device (D2D):* two or more devices directly connect and communicate between one another (*cf.* Silos 1, 2 and 3 in Fig. 1), rather than through an intermediary application server.
- *Device-To-Gateway (D2G):* the IoT device connects to a local gateway device that may either (i) be connected to a Cloud service provider (*cf.* Silo 1 in Fig. 1) or (ii) store and process device-related data at the edge (*cf.* Silo 2);
- *Device-To-Cloud (D2C):* the IoT device connects directly to an Internet Cloud provider to exchange data and services (*cf.* Silo 3 in Fig. 1). Frequently, the device and Cloud service are from the same vendor (commonly referred to as "vendor lock-in");
- *Back-End Data-Sharing (S2S):* this model plays a key role in improving horizontal interoperability across sectors and platforms, thus breaking down traditional data silo barriers (as shown in Fig. 1). More concretely, this model shall facilitate *Server-To-Server* (S2S) information exchange based upon open and standardized IoT interfaces (open APIs, standardized semantic vocabularies...), but shall also provide provisions for Analytics services, e.g. to filter, aggregate and analyze cross-domain and cross-platform information (*cf.* Fig. 1).

Several IoT messaging standards have been designed to address one or more of these communication models. The most well known IoT standards to date are [2]: (i) CoAP, developed by IETF, which addresses D2D-D2G; (ii) MQTT, developed by IBM, which is well suited for use within D2G-D2C applications; (iii) AMQP, developed by OASIS, which supports D2C-D2G-S2S models; (iv) Data Distribution Service (DDS), developed by the Object Management Group, which addresses D2D-D2C-D2G-S2S. Recently, two messaging standards named Open Messaging Interface (O-MI) [10] and Open Data Format (O-DF) [11] were published by The Open Group, whose primarily aim is to improve horizontal interoperability across vertical silos (S2S). Although this paper is not intended to carry out a technical comparison between O-MI/O-DF and the above-mentioned standards, a few striking differences can nonetheless be pointed out: O-MI uses text-based representations (XML, JSON...) instead of binary formats, and can use any of the 'Communication' and 'Transport' level standards as its underlying protocol; O-MI provides a "RESTful" URL-based query mechanism and, like DDS, is "Data-centric" meaning that middleware can understand the data (e.g., object identity, hierarchy...). O-MI/O-DF standards are presented in further detail in the next section.

III. O-MI & O-DF: An efficiency ratio model

O-MI and O-DF standards emerged out of past EU FP6 and FP7 projects (e.g., PROMISE FP6, LinkedDesign FP7...), where real-life industrial applications required the collection and management of product instance-level information for many domains involving heavy and personal vehicles, household equipment, phone switches, *etc.* [12]. Information such as sensor readings, alarms, assembly, disassembly, shipping events, and other information related to the entire product life-cycle needed to be exchanged between products and systems

of different organizations [13]. Based on the needs of those real-life applications, and as no existing standards could be identified that would fulfil those requirements without extensive modification or extensions, the partner consortia started the specification of new messaging interfaces [14]. Those specifications have since then been further developed and published by the IoT WG of The Open Group. Section III-A gives more details about both standards, while section III-B focuses on the related efficiency ratio analytical model.

A. O-MI & O-DF: a high-level introduction

O-MI and O-DF are independent entities that reside in the OSI Application layer, respectively specified at the 'communication' and 'format' levels [14]. O-MI provides a generic Open API for any RESTful IoT information system, meaning that in the same way that HTTP can be used for transporting payloads in formats other than HTML, O-MI can be used for transporting payloads in nearly any format. The complementary – *but not compulsory* – standard (O-DF) partly fulfils the same role in the IoT as HTML does for the Internet, meaning that O-DF is a generic content description model for Things in the IoT that can be extended with more specific vocabularies (e.g., using domain-specific ontology vocabularies).

O-DF is defined as a simple ontology, specified using XML Schema, that is generic enough for representing "any" object and information that is needed for information exchange in the IoT. It is intentionally defined in a similar manner as data structures in object-oriented programming. O-DF is structured as a hierarchy with an "Objects" element as its top element, which can contain any number of "Object" sub-elements. "Object" elements can have any number of properties, referred to as InfoItems, as well as "Object" sub-elements. The resulting Object tree can contain any number of levels. Every Object has a compulsory sub-element called "id" that identifies the Object. The "id" should preferably be globally unique, or at least unique for the application of the involved organizations.

A defining characteristic of O-MI is that nodes may act both as "servers" and as "clients", and therefore communicate directly with each other or with back-end servers in a peer-to-peer manner. One of the fundamental properties of O-MI is that O-MI/O-DF messages are "protocol agnostic" so they can be exchanged using HTTP, SOAP, SMTP, or similar protocols. Four key operations (as summarized in TABLE I) as well as a "RESTful" URL-based query mechanisms (for information publication and discovery) are supported. Another important feature is that messages are "self-contained" in the sense that all the necessary information to enable the recipient to handle the message is contained in the message itself (e.g., operation to be performed, callback address, subscription interval…).

The use case developed in section IV will provide an overview of an O-MI/O-DF message, which will facilitate the understanding of the above introduced properties and features.

B. Efficiency ratio analytical model

The efficiency ratio (hereafter denoted by ER) is defined as the percentage of payload over the amount of data being carried by the network. The analytical model developed in this paper is intended to pre-determine the length of one or more O-MI/O-DF request/response messages depending on the type of operations (*cf.* TABLE I). This model is developed based upon the official O-MI/O-DF standard specification documents [10], [11], and only takes into account the 'compulsory' fields (i.e., specified as SHALL in the standard).

TABLE I
MAIN MESSAGING INTERFACES SPECIFIED IN THE O-MI STANDARD

Operation	Description
Write	Used to send information updates to O-MI nodes.
Read	Used for immediate retrieval of information from an O-MI node.
Subscript °	Used to perform subscriptions, either: • ***with*** *callback address:* the subscribed data is sent to the callback address at the requested interval. Two types of intervals are supported: ***interval-based*** and ***event-based***; • ***without*** *callback address:* data is memorized on the subscribed node as long as the subscription is valid. Historical data can be retrieved (i.e., polled) by issuing a new O-MI read request (by specifying the subscription ID).
Cancel	Used to cancel a subscription before it expires.

As previously mentioned, O-MI is independent of the lower layers. As a first approximation, the size of a request or a response can be formulated as in Eq. 1, where $\ell_{\text{low-layer}}$ and $\ell_{\text{app-layer}}$ are respectively the length of the lower layers and the application layer (O-MI included). Both variables are detailed in the following.

$$S_{\text{req}} = \ell_{\text{low-layer}} + \ell_{\text{app-layer}} \tag{1}$$

1) Application layer: O-MI standard specifications recommend to implement HTTP as underlying communication protocol, whose length ($\ell_{\text{app-layer}}$) corresponds to the sum of the HTTP and O-MI protocols, as well as the message payload (which may be O-DF, JSON…), as given in Eq. 2. These three variables can be determined using a parametric model, as proposed in Fig. 2. The following discusses each layer and associated variables of this figure/model.

$$\ell_{\text{app-layer}} = \ell_{\text{HTTP}} + \ell_{\text{O-MI}} + \ell_{\text{payload}} \tag{2}$$

According to the standard specifications, O-MI messages can be sent using either HTTP POST or HTTP GET (URL-based) when a RESTful interface is more appropriate. The size of the request message therefore depends on the method used (POST or GET) as well as the URL length denoted by ℓ_{url} in Fig. 2. The HTTP response contains a status-code referred to as "reason-phrase", whose size is denoted by ℓ_{reason}. In practice, HTTP can also embed a general header that contains e.g. the user-agent (name of the software producing the request) and/or an entity-header that provides additional information on the payload (e.g., encoding, length and type of the content). However, as mentioned above, our analytic model only takes into account 'compulsory' fields.

The length of an O-MI request message depends both on the type of operation/interface used (e.g. write, read, cancel…) and associated fields (e.g. TTL, ID, callback…). The O-MI frame in Fig. 2 emphasises which fields is required by each operation, e.g. a 'Write' message interface requires a TTL[2], whose size is denoted by ℓ_{ttl}. Similarly, other fields

[2]When TTL expires, O-MI nodes SHOULD answer with an error response.

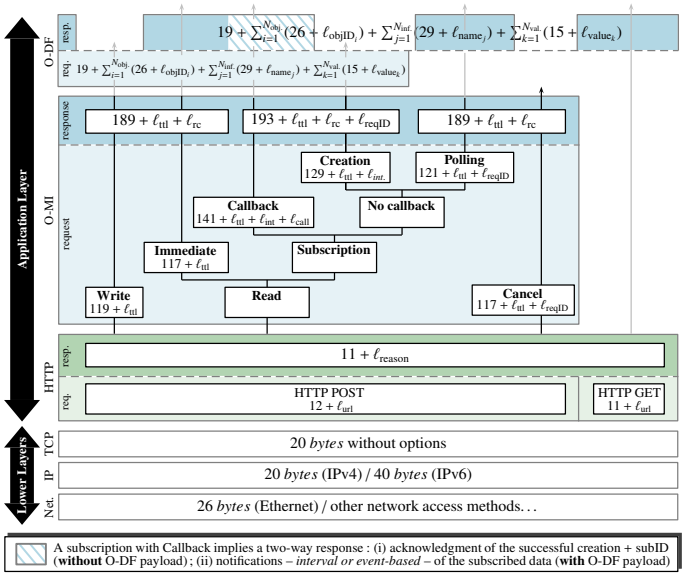


Fig. 2. O-MI/O-DF request/response message size

and associated lengths are introduced with regard to the other operations, such as *Interval*, *Callback* and *Request_ID* when performing subscription requests (lengths respectively denoted by ℓ_{int}, ℓ_{call} and ℓ_{reqID}). Regarding O-MI responses, they always include a return-code indicating the success or failure of the operation (associated length denoted by ℓ_{rc}), and the overall message size depends on whether the response includes a subscription ID, as shown in Fig. 2.

Considering now the O-DF layer in Fig. 2, it must be noted that the length to any O-DF structure depends on the number of 'Object' elements (denoted by N_{obj}), 'InfoItem' elements (N_{inf}), 'Value' elements[3] (N_{val}) and, as a result, depends on the number of digits composing each Object's ID (denoted by ℓ_{objID}), InfoItem's name (ℓ_{name}) and the Value itself (ℓ_{value}). Although the scientific contribution of our study is primarily on the O-MI/O-DF message size model, it is also worth focusing on the overall traffic load, which requires to take into account the lower layers as discussed below.

2) Lower layers: Based on the HTTP specifications, HTTP communications usually take place over TCP/IP connections. Having said that, the length of the (network) IP header depends on whether IPv4 or IPv6 is in use, as emphasized in Fig. 2. In our model, we consider Ethernet as underlying network access protocol, but other protocols coud be considered in future models (e.g., IEEE 802.15.4). As a result, $\ell_{\text{low-layer}}$ is either equal to 66 $bytes$ $(26 + 20 + 20)$ or 86 $bytes$ $(26 + 40 + 20)$.

3) Efficiency Ratio: It should be noted that S_{req} (*cf.* Eq.1) does not consider the lower layer constraints (especially the network access method) in terms of Maximum Transmission Unit (MTU – 1500 $bytes$ in Ethernet). Indeed, more than one frame will be sent if $\ell_{\text{app-layer}} > MSS$ (Maximum Segment Size – 1460 $bytes$ with IPv4/TCP). The number of frames can be expressed as $n = \left\lceil \frac{\ell_{\text{app-layer}}}{MSS} \right\rceil$. The total length of data carried by the network for a request, denoted by L_{req}, can therefore be defined as in Eq. 3 (ℓ_{net} being the length of the network access layer). The length of data for a response, denoted by L_{req}, can be computed based upon the same equation.

$$L_{\text{req}} = (n-1) \cdot (\text{MTU} + \ell_{net}) + S_{\text{req}} - (n-1) \cdot \text{MSS} \quad (3)$$

As TCP is used as transport protocol, the overall traffic load cannot be expressed directly from the request and response length of data (i.e., L_{req} and L_{resp}). Indeed, opening and closing TCP connections and segment acknowledgments should be taken into consideration. It is important to note that one or more O-MI/O-DF requests/responses can be transmitted over a same TCP connection; as a result, the transient states of the TCP opening and closing operations are not considered in this study. The overall traffic load, denoted by TL, can finally be defined as in Eq. 4, where L_{ack} is the length of all acknowledgments[4]. The acknowledgment can be achieved either immediately a segment is received, or after several segments are received, or inside a new data transmission (piggybacking). However, as in practice the TCP behavior changes according to the operating system and the TCP configuration, we assume that an acknowledgement is sent after m received segments. The number of acknowledgments for a request and/or response comprising n Ethernet frames is equal to $\frac{n}{m}$, and the overall length of those acknowledgements can be expressed as in Eq. 5. Finally, Eq. 6 provides the efficiency ratio.

$$TL = L_{\text{req}} + L_{\text{resp}} + L_{\text{ack}} \quad (4)$$

$$L_{\text{ack}} = \left(\left\lceil \frac{n_{\text{req}}}{m} \right\rceil + \left\lceil \frac{n_{\text{resp}}}{m} \right\rceil \right) \cdot \ell_{\text{low-layer}} \quad (5)$$

$$ER = \frac{\ell_{\text{payload}}}{TL} \quad (6)$$

In the next section, a smart maintenance use case relying on the first version of the standard reference implementation is presented, based on which our model is applied to.

IV. Smart Maintenance Use Case

The overall use case is depicted in Fig. 3, which involves a 'Company X' that produces goods via its company branches (see *Branch company Xa, Xb, Xc*). Different types of industrial robots – *from various robot manufacturers* – are used in those branches, as illustrated in Fig. 3 with *Robot A* and *Robot B*. *Company X*'s head office has a maintenance service department that relies on/implements the O-MI/O-DF messaging standards to (i) monitor whether a malfunction occurs in a specific robot; (ii) investigate the possible causes of the malfunction (potentially by exchanging information with the robot's manufacturer); and (iii) take appropriate actions to fix the problem (e.g., send specific repair procedures or a repairman on site...). Given this, the objective of this use case is twofold:

- Section IV-A: providing concrete insights into how the Industrial Internet can benefit from O-MI/O-DF to improve interoperability among various industrial stakeholders and systems, and to achieve the above-mentioned services;

[3]Note that N_{obj}, N_{inf}, $N_{\text{val}} = \emptyset$ if no Object, InfoItem, Value is embedded in the O-DF message, making the associated sum term equal to zero.

[4]The segment retransmission phase when messages are lost – *which inevitably affects the traffic load* – is not taken into account in our model.

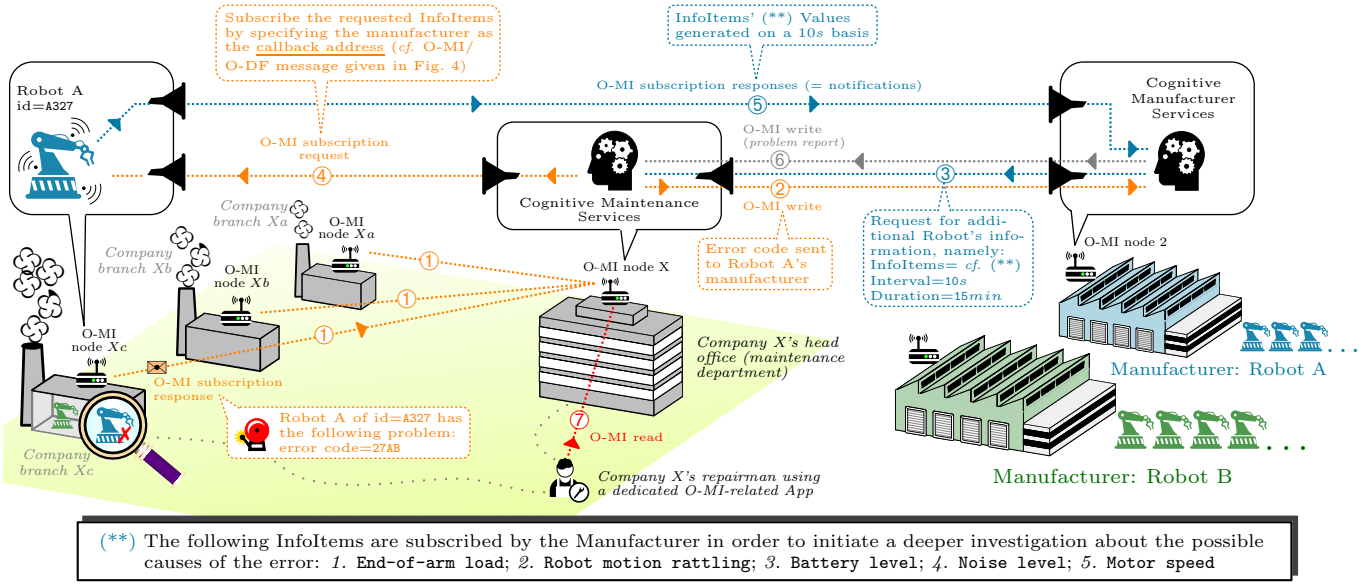


Fig. 3. PLM scenario highlighting how the SoS integration framework opens up business opportunities for creating more collaborative & open service models

- Section IV-B: evaluating – *in terms of traffic load deviation w.r.t the standards* – the first version of the standard reference implementation[5] that has been used to set up this smart maintenance use case.

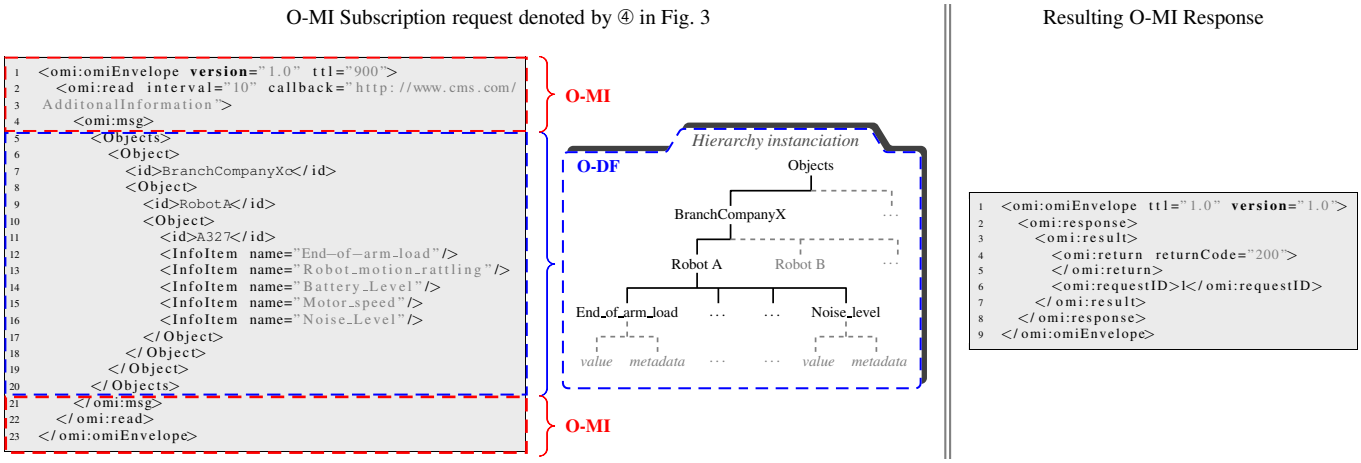
A. O-MI/O-DF-based smart maintenance services

As depicted in Fig. 3, the different actors implement an O-MI node, namely the three company branches (see *O-MI node Xa, Xb, Xc*), the company X's maintenance department (*O-MI node X*) and the robot manufacturers. Each node publishes a set of (hierarchical) information that can be accessed by authorized peer systems. The company branch's node publishes, among other things, an information about robot units that raise one or more error messages. As illustrated through arrows denoted by ① in Fig. 3, the maintenance department subscribes – *'forever' and using an 'event-based' subscription* – to this information item on each company branch's node. An error code occurs on a *Robot A* in *Company branch Xc*, resulting in a notification being sent to the maintenance department (*cf.* Fig. 3). Given the error code and the type of robot, the smart maintenance system takes the decision to send those information to the robot's manufacturer system (using an O-MI write request, see arrow denoted by ② in Fig. 3) in order receive relevant feedback/support and potential repair procedures. The manufacturer system, after having processed the error code, would like to access specific sensors embedded in the robot in order to investigate in more depth the problem. To this end, the manufacturer system sends a request for subscribing, during $15min$, with an interval of $10sec$, to five distinct robot's sensors (see arrow denoted by ③ and **). Based on this request, the maintenance department's node generates an O-MI subscription request being sent to O-MI node *Xc* (see arrow denoted by ④).

[5]see e.g.: https://otaniemi3d.cs.hut.fi/omi/node/html/webclient/index.html

The corresponding O-MI subscription request message is given in Fig. 4. Rows 1 to 4 correspond to the message interface (i.e., O-MI-related fields) where the operation is set to `read` (a subscription being a specific read operation, *cf.* Fig. 2), the interval to `10sec`, the subscription duration (TTL) to `900sec`, and the callback address to `http://www.cms.com:...` (manufacturer's service/servlet). Rows 5 to 20 detail the message payload built on the generic O-DF information hierarchy. This hierarchy instanciation highlights that `BranchCompanyXc` and `RobotA` are defined as O-DF 'Object' with specific IDs (see rows 7 and 9). The InfoItems (i.e., Object properties) that need to be subscribed to are specified at rows 12 to 16. Following this request, a response is sent to the subscription initiator node (i.e., to *O-MI node X*), as shown in Fig. 4, including the success `returnCode` (see row 4) and the subscription `requestID` (see row 6). It can be noted that the information hierarchy presented in this scenario is very basic in an effort to simplify the understanding of O-DF, but more complex hierarchies can be designed, e.g. respecting complex BOMs (bill of materials), while preserving a basic compatibility between all hierarchies/extensions. The IoT WG of The Open Group has created one such extension, called *Physical Product Extension*, which provides specifications for representing PLM information [15].

Following this subscription, notification messages containing the "Values" of the five subscribed sensor data are pushed to the manufacturer system (see arrow ⑤). Based on those sensor values, the manufacturer system identifies the probable cause of the problem and, as a result, sends a report – *including e.g. repair procedures* – to the Company X's maintenance department (see arrow ⑥). The scenario could potentially be extended by proposing maintenance O-MI/O-DF-related Apps that would enable any repairman to access the report when and as needed, or still discover, when arriving on site, new information sources and/or historical robot-related data that could prove extremely valuable during a repair process.




```
<omi:omiEnvelope version="1.0" ttl="900">
  <omi:read interval="10" callback="http://www.cms.com/
  AditonalInformation">
    <omi:msg>
      <Objects>
        <Object>
          <id>BranchCompanyXc</id>
          <Object>
            <id>RobotA</id>
            <Object>
              <id>A327</id>
              <InfoItem name="End-of-arm_load"/>
              <InfoItem name="Robot_motion_rattling"/>
              <InfoItem name="Battery_Level"/>
              <InfoItem name="Motor_speed"/>
              <InfoItem name="Noise_Level"/>
            </Object>
          </Object>
        </Object>
      </Objects>
    </omi:msg>
  </omi:read>
</omi:omiEnvelope>
```

```
<omi:omiEnvelope ttl="1.0" version="1.0">
  <omi:response>
    <omi:result>
      <omi:return returnCode="200">
      </omi:return>
      <omi:requestID>1</omi:requestID>
    </omi:result>
  </omi:response>
</omi:omiEnvelope>
```

Fig. 4. O-MI/ODF subscription request message (denoted by ③ in Fig. 3) and corresponding response (acknowledgment of the successful creation + subID)

B. Efficiency ratio analysis of the reference implementation

The efficiency ratio analytical model developed in section III-B is applied to the smart maintenance use case considering one of the first version of the O-MI/ODF reference implementation[6] (version 0.2.2). To this end, a traffic sniffer (Wireshark) was used to capture the traffic load generated by the different O-MI nodes. In the following, each communication exchange (i.e., arrows that have been denoted by ① to ⑦ in Fig. 3) are represented on the x-axis of Fig. 5.

Fig. 5(a) gives insight into the traffic load (TL) computed by our analytical model (referred to as "std.") as well as generated by the reference implementation (referred to as "ref."). First, it can be noted that the traffic load related to "ref." is around twice higher than the (minimal) traffic load. Nonetheless, from an efficiency perspective, it can be observed that the efficiency ratio (computed via Eq. 6) is being approximately equal between "ref." and "std.", which gives a first indication on the fact that the implementation is compliant with the O-DF standard. However, the difference/deviation of the traffic load between "ref." and "std." that has been previously discussed can be studied further. In this respect, let us remind ourselves that a deviation is not necessarily a negative outcome; in fact, it depends on either (i) the developer added, for specific reasons, optional fields/features in one or more underlying protocols (lower layers, HTTP, O-MI/O-DF), or (ii) the developer did not fully comply with the standard specifications.

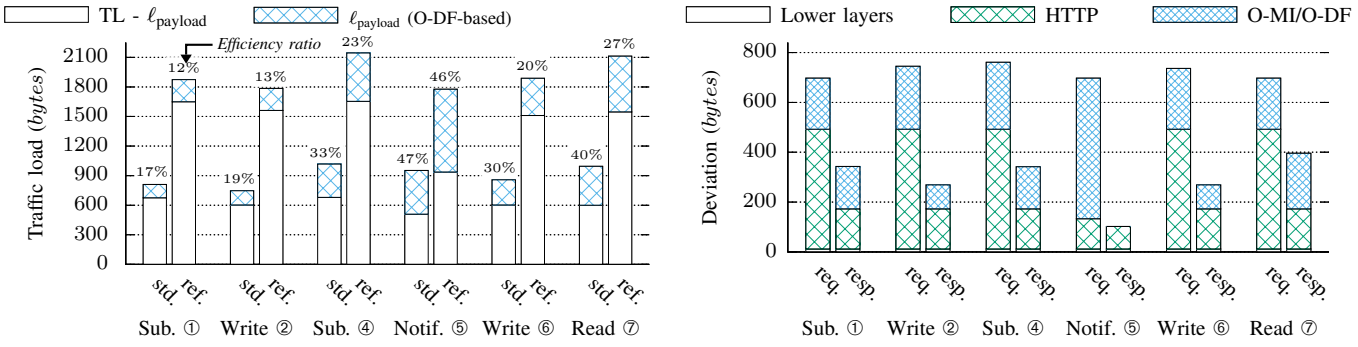
Fig. 5(b) gives insight into the traffic load deviation (in $bytes$) of the O-MI/O-DF reference implementation with respect to the standards. Looking at the "Lower layers", it can be noted that they do not deviate much from the basic functionalities required by the lower layer protocols (the reference implementation always adding 12 $bytes$ in the lower layers). The reason of this addition is that TCP options have been included, namely: timestamps (10 $bytes$) and (no-op) padding (2 $bytes$). Although timestamps can potentially be used to determine the Round Trip Time (RTTM mechanism), one may wonder whether this is really necessary in the context of a S2S communication model.

[6]https://github.com/AaltoAsia/O-MI

Looking at the HTTP layer, it can be observed that 480 $bytes$ for all the requests and 161 $bytes$ for all the responses (except for notifications ⑤) are added. This stems from the fact that the web browser adds 'optional' HTTP fields (general-header, request/response-header, entity-header) in the request. Indeed, when a web browser makes a request, it sends information to the server about what it is looking for. Similarly, optional fields are added by the O-MI node web server in the response, corresponding to HTML `<meta>` tags. It should be noted that such metadata is mainly dependent on the web browser and operating system used in the application; to put it another way, this deviation w.r.t the analytical model was somehow expected. Finally, notification request/response messages are handled by a specific service (e.g., Java Servlet) that uses a specific header, which is why a slight deviation is observed compared with the other communication operations. All in all, the HTTP layer implementation is by no means a bad implementation of the standards.

Finally, looking at the O-MI/O-DF layer, the traffic load deviation varies according to the communication operations (between 0 and 564 $bytes$). This deviation can be explained by different factors, namely the provision for: (i) optional fields defined in the O-MI/O-DF standards (e.g., the `Description` field into response messages) or recommended fields (e.g., the message format field in the O-MI message interface to provide indications on the payload format), (ii) specific domain-specific data model information via namespace attributes, e.g. using the *Physical Product Extension* model as exemplified in the standards; (iii) human-readable formats, e.g. using spaces and carriage return/line feed (choice made in the O-MI/O-DF reference implementation web interface).

Finally, let us point out that none of the O-MI message exchanges from our scenario was transmitted in several TCP segments, which is due to the fact that the payload is inferior to the MTU. However, this all depends on the O-DF structure. In our scenario, in an effort to simplify the understanding of O-DF, a few 'Object'(s), InfoItems and Values were defined, but this payload may significantly increase in real-life industrial situations (e.g., considering BOM information, or semantic vocabularies that are of the utmost importance for



(a) Traffic Load (TL) analysis

(b) Traffic Load (TL) deviation analysis

Fig. 5. O-MI/O-DF reference implementation analysis with respect to the standard specifications

S2S communication purposes [16]. However, the traffic load and network efficiency criteria can be used in further studies to (i) define and refine O-DF structures for minimizing the traffic load (impacting on the response time), while maximizing the network efficiency, which may prove relevant when developing new reference implementations (e.g., for resource-constrained devices), or – *as used in this paper* – (ii) to evaluate and compare one or more standard reference implementations with respect to the standards, or among themselves.

V. Conclusion

Billions of devices are connected to the Internet and it is predicted that there will be 50 to 100 billions by 2020. On the way towards platforms for connected smart objects, the biggest challenge to overcome is the fragmentation of vertically-oriented closed systems towards open, integrated and collaborative systems-of-systems. In this context, this paper discusses the importance of standardization, while positioning and presenting two recent IoT messaging standards published by The Open Group, notably O-MI and O-DF. Although those standards are a result of over 10 years of research work jointly with many academic and industrial partners, creating such standards and getting them widely used tends to be a long and challenging task.

In addition to presenting O-MI and O-DF, this paper develops and presents an analytical model of the efficiency ratio based on the required/basic standard specification. This model is a applied to a smart maintenance use case, which is built on the first version of the standard reference implementation, thus helping us to assess the deviation – *in terms of efficiency ratio and traffic load* – of that implementation with respect to the standards. Such a deviation is not necessarily a negative outcome since it can be due to the introduction of optional fields/features (whether in the lower layers, HTTP, or O-MI/O-DF), but at least the proposed model helps to be aware of this, and potentially to take decisions out of it. From an implementation viewpoint, it should noted that O-DF is verbose as XML is used for structuring IoT information (mainly due to its flexibility for complex data structures), but O-MI is independent of O-DF and could potentially transport other formats, and *vice-versa* (O-DF can be used as payload in MQTT, AMQP…).

Acknowledgments

The research leading to this publication is supported by the EU's H2020 Programme (grant 688203), as well as the National Research Fund Luxembourg (grant 9095399).

References

[1] P. C. Evans and M. Annunziata, "Industrial Internet: Pushing the Boundaries of Minds and Machines," Tech. report, General Electric, 2012.

[2] A. Al-Fuqaha, M. Guizani, M. Mohammadi, M. Aledhari, and M. Ayyash, "Internet of Things: A Survey on Enabling Technologies, Protocols and Applications," *IEEE Communications Surveys & Tutorials*, DOI: 10.1109/COMST.2015.2444095, 2015.

[3] K. Ashton, "Internet things – MIT, embedded technology and the next internet revolution," *Baltic Conventions, The Commonwealth Conference and Events Centre*, London, UK, May 2000.

[4] L. Xu, W. He, and S. Li, "Internet of Things in industries: a survey," *IEEE Transactions on Industrial Informatics*, no. 99, 2014.

[5] W. He and L. Da Xu, "Integration of distributed enterprise applications: a survey," *IEEE Transactions on Industrial Informatics*, vol. 10, no. 1, pp. 35–42, 2014.

[6] R. Schneiderman, *Modern standardization: case studies at the crossroads of technology, economics, and politics*. John Wiley & Sons, 2015.

[7] A. W. Colombo, S. Karnouskos, and T. Bangemann, "A system of systems view on collaborative industrial automation," in *IEEE International Conference on Industrial Technology*, Cape Town, South Africa, 2013.

[8] J. Kim, J. Lee, J. Kim, and J. Yun, "M2M service platforms: survey, issues, and enabling technologies," *IEEE Communications Surveys & Tutorials*, vol. 16, no. 1, pp. 61–76, 2013.

[9] H. Tschofenig, J. Arkko, and D. McPherson, "Architectural Considerations in Smart Object Networking", *Internet Engineering Task Force, Fremont, CA, USA, Tech. Rep. RFC-7452*, March 2015.

[10] The Open Group, "*Open Messaging Interface Technical Standard (O-MI)*," *https://www2.opengroup.org/ogsys/catalog/C14B*, October 2014.

[11] The Open Group, "*Open Data Format Technical Standard (O-DF)*," *https://www2.opengroup.org/ogsys/catalog/C14A*, October 2014.

[12] D. Kirtsis, "Closed-loop PLM for intelligent products in the era of the Internet of Things," *Computer-Aided Design*, vol. 43, no. 5, pp. 479–501, 2011.

[13] K. Främling, J. Holmström, J. Loukkola, J. Nyman, and A. Kaustell, "Sustainable PLM through Intelligent Products," *Engineering Applications of Artificial Intelligence*, vol. 26, no. 2, pp. 789–799, 2013.

[14] K. Främling, S. Kubler, and A. Buda, "Universal messaging standards for the IoT from a lifecycle management perspective," *IEEE Internet of Things Journal*, vol. 1, no. 4, pp. 319–327, 2014.

[15] S. Parrotta, J. Cassina, S. Terzi, M. Taisch, D. Potter, and K. Främling, "Proposal of an Interoperability Standard Supporting PLM and Knowledge Sharing," in *Advances in Production Management Systems. Sustainable Production and Service Supply Chains*, pp. 286–293, 2013.

[16] S. N. Han, G. M. Lee, and N. Crespi, "Semantic context-aware service composition for building automation system," *IEEE Transactions on Industrial Informatics*, vol. 37, no. 1, pp. 69–88, 2013.