Deferred retrieval of IoT information using QLM messaging interface

Sylvain Kubler, Manik Madhikermi, and Kary Främling

Aalto University, School of Science, Espoo, Finland.
P.O. Box 15400, FI-00076 Aalto, Finland.
corresponding author: sylvain.kubler@aalto.fi

Abstract. Internet of Things (IoT) is intended to provide a network where information flows could easily be retrieved and set up between any kinds of products, devices, users, *etc.* Quantum Lifecycle Management messaging interface (QLM-MI) has been designed accordingly, providing generic and standardized application-level interfaces. One cornerstone property of QLM-MI provides the opportunity to subscribe IoT information from any "Thing" so as to retrieve it in a deferred way. Subscriptions can be performed to receive a "continuous" data flow or can be based on "client-initiated" communications. All these features are described in this paper, and then a technical proof-of-concept is provided based on a scenario in the framework of smart home.

Keywords: Internet of things; Quantum Lifecycle Management; Intelligent product; Smart house

1 Introduction

IoT refers to a world where physical objects and beings, as well as virtual data and environments, all interact with each other in the same space and time [1]. Connections are not just people to people or people to computers, but people to things and most strikingly, things to things. IoT is becoming increasingly popular in a wide range of sectors (manufacturing, transportation, smart home, healthcare, and so on) [2,3]. In the context of closed-loop product lifecycle management [4], IoT is used as a generic information system for accessing and synchronizing any kind of product-related information over the Internet. In this context, products can have varying degrees of "intelligence", which can range from simple barcodes or RFID tags to vehicles and other products that have advanced sensing, actuating, memory and communication capabilities [5,6].

Quantum Lifecycle Management messaging interface (QLM-MI) was created[1] to enable any kind of intelligent product to exchange IoT information in ad hoc, loosely coupled ways. The lifecycle concept of QLM-MI signifies that information exchange standards have to be able to provide interoperability between products and with all other information systems that consume or provide relevant information in the product lifecycle. QLM-MI specifications respond to

[1] http://www.opengroup.org/getinvolved/workgroups/qlm

requirements of several real-life industrial applications of the PROMISE (Product Lifecycle Management and Information Tracking Using Smart Embedded Systems: `www.promise.no`) EU FP6 project [4], the main ones are:

– possible to implement for any kind of information systems, including embedded and mobile systems,
– not restricted to one communication protocol only, it must be possible to send messages using protocols such as plain HTTP, SOAP, SMTP,
– support for "synchronous" messaging as immediate read/write operations, including "client-poll" subscriptions,
– handling mobility and intermittent network connectivity, i.e. support for asynchronous messaging capabilities, message consistence, time-to-live functionality,
– historical queries, i.e. retrieving values between two points in time.

This paper provides an overview of the QLM-MI standards proposal that fulfill these requirements. QLM-MI combines the main features of asynchronous, enterprise messaging protocols with those of instant messaging protocols in a way that allows for peer-to-peer type communication. Section 2 first gives insight into the QLM-MI properties. Among them, a fundamental one enables a *Thing A*[2] to subscribe any information from another *Thing B*. The subscription can be performed to receive a "continuous" data flow from *Thing B*, but also provides the opportunity to perform a "client-initiated" communication (most of the solutions do not provide this second opportunity). Section 3 presents these two types of subscription. The main contribution of this paper is to provide a technical proof-of-concept of the subscription property, which is provided by the case study in section 4 (defined in the framework of smart home).

2 QLM messaging interface

In practice, no suitable standards were found to fulfill the different requirements mentioned in section 1. In addition to EPC-related standards, such as STEP, MIMOSA, PLCS and DPWS were considered [7]. In practice there seemed to be "too many" standards already but they all tended to be too domain-, technology- or lifecycle phase-specific [8]. Accordingly, PROMISE created two main specifications that fulfilled the necessary requirements: the PROMISE Messaging Interface (PMI) and the PROMISE System Object Model (SOM). At the end of the PROMISE project, the work on these standards proposals was moved to the QLM workgroup of The Open Group (http://www.opengroup.org/qlm/). This paper presents the standard proposal derived from PMI, i.e. QML-MI.

In the QLM-MI world, communication between the participants, e.g. products and backend systems, is done by passing messages between nodes using interfaces defined in QLM-MI. The QLM-MI "cloud" in Fig. 1 is intentionally drawn in the same way as for the Internet cloud. Where the Internet uses

[2] A "Thing" is also referred to as a "QLM node" in this paper.

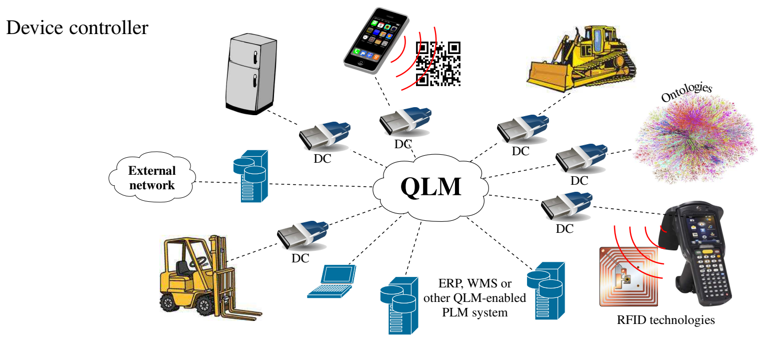


Fig. 1. QLM "Cloud"

the HTTP protocol for transmitting HTML-coded information mainly intended for human users, QLM-MI is used for transmitting lifecycle-related information mainly intended for automated processing by information systems[3]. In the same way as HTTP can be used for transporting payloads also in other formats than HTML, QLM-MI can be used for transporting payloads in nearly any format. XML might currently be the most common text-based payload format but others such as JSON, CSV can be used. In total, 7 major properties of QLM-MI can be listed:

1. *QLM-MI messages are **self-contained***: messages can be transported using most "lower-level" protocols such as HTTP, SOAP, FTP or similar protocols. It can also be transported using files on USB sticks or other memory devices. This non-dependency on specific communication protocols makes QLM-MI different from many other potential IoT messaging standards,
2. *three possible operations*:
   - **QLM-MI write:** used for sending information updates to QLM nodes. This involves a QLM-MI response to inform the message originator about the success or failure of the operation,
   - **QLM-MI read**: used for
     - immediate retrieval of information. This involves a QLM-MI response from the targeted QLM node to return the requested information,
     - placing subscriptions for deferred retrieval of information from a QLM node. This is done with a QLM-MI read query if the interval parameter has been set. If:
       - **a callback address** is provided, then the data is sent using a QLM-MI response at the requested interval,
       - **no callback address** is provided, then the data can be retrieved (polled) by issuing a new QLM-MI read query with the ID of the subscription,

[3] QLM-MI resides in the Application layer of the Open Systems Interconnection (OSI) model.

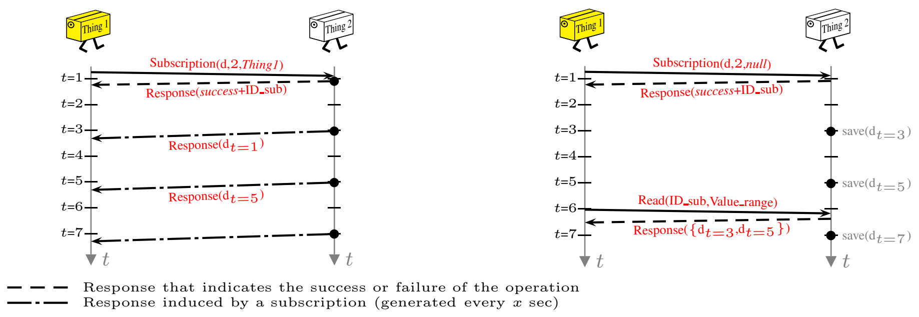
   - **QLM-MI cancel:** used to cancel subscriptions before they expire. This involves a QLM-MI response to inform the message originator about the success or failure of the operation.
3. *all queries and responses can specify a* ***time-to-live***: if the message has not been delivered to the "next" node before time-to-live expires, then the message should be removed and an error message returned to the message originator, if possible,
4. *allowing different* ***payload*** *formats*: a QLM-MI message (whether query or response) can transport actual information using any text-based format that can be embedded into an XML message. It is even possible to use different payload formats in different return elements of a QLM-MI response,
5. *allowing* ***synchronous*** *("real-time") communication between nodes*: any QLM-MI response can include a new query, which is useful for instance in control applications. It also provides a possibility to perform "client-initiated" communication with nodes that are located behind a firewall,
6. ***publication and discovery*** *of data sources, services and meta-data*: publication of new data sources, services and meta-data can be done with QLM-MI write operation. "RESTful" URL-based queries allow the discovery of them, including discovery by search engines,
7. *all queries can specify a list of* ***target QLM nodes***: the receiving node(s) are then responsible of re-routing the query to the target QLM nodes, or sending back an error message to the requesting QLM node in case of failure.

QLM-MI can be applied to virtually any kind of information, i.e. not only physical products but also to document repositories, *etc.*. Querying for available design documents (e.g. CAD documents); subscribing to the addition/deletion/-modification of documents; subscribing to particular change events in design documents is conceptually similar to queries and subscriptions for physical products. As mentioned, the "subscription" property is a cornerstone of QLM-MI. The conceptual framework used here is the Observer Design Pattern presented by [9], which signifies that a QLM node can add itself as an observer of events that occur at another QLM node. In this sense, QLM-MI differs from other messaging models such as JMS, which is based on the Publish-Subscribe model. For many applications, the Observer and the Publish-Subscribe models can be used in quite similar ways. However, the Publish-Subscribe model usually assumes the usage of a "high-availability server", which the Observer pattern does not [10]. For this reason, the Observer model is more suitable for IoT applications where products might communicate with each other directly.

In this paper, a particular focus is given to the property 2, and especially on the two options for placing subscriptions for deferred retrieval of IoT information (i.e. with and without callback).

3 Deferred retrieval of information using "subscriptions"

In IoT, new objects frequently join and start communicating and existing ones disappear. Currently, there are millions of sensors deployed around the world


Thing 1
Thing 2
Subscription(d,2,*Thing1*)
Response(*success*+ID_sub)
Response($d_{t=1}$)
Response($d_{t=5}$)
t=1
t=2
t=3
t=4
t=5
t=6
t=7
t
Subscription(d,2,*null*)
Read(ID_sub,Value_range)
Response($\{d_{t=3}, d_{t=5}\}$)
save($d_{t=3}$)
save($d_{t=5}$)
save($d_{t=7}$)
Response that indicates the success or failure of the operation
Response induced by a subscription (generated every x sec)

(a) With callback address (b) Without callback adress

Fig. 2. Deferred retrieval of information: callback *vs.* no callback

and it is predicted that there will be $50 - 100$ billion devices connected to the Internet by 2020 [11]. Similarly to the number of objects, the number of communications will also increase significantly, thus requiring new types of IoT interactions. QLM-MI aims - in the long term - at covering the vast majority of these interactions and, in this effect, new ones are introduced with QLM-MI through the concept of "subscription".

Fig. 2(a) and 2(b) respectively detail the two types of subscriptions considering two QLM nodes: *Thing 1* and *Thing 2*. In Fig. 2(a), *Thing 1* subscribes a data d on *Thing 2* at $t = 1$, providing its address as callback with an interval of 2 units of time[4]. A response is first sent by *Thing 2* to *Thing 1* both to confirm the correct reception of its QLM-MI read and to provide it with the ID of the subscription (see the response at $t = 1$). Then, every 2 units of time, *Thing 2* sends to *Thing 1* the current data d value (again through a QLM response as depicted at $t = 3, 5, 7, \ldots$). A distinction is made between a "response that indicates the success or failure of the operation" and a "response induced by the subscription (i.e. automatically generated every x units of time)". This distinction is of particular importance in presence of a firewall. Let us consider a firewall that does not permit *Thing 2* to reach *Thing 1* but does the opposite; the QLM-MI response at $t = 1$ could successfully reach *Thing 1* because the firewall always authorizes a response subsequent to a query. However, the QLM-MI responses sent by *Thing 2* at $t = 3, 5, 7, \ldots$ will be stopped by the firewall because they are not "directly" induced by a query, but automatically generated after x units of time.

In Fig. 2(b), *Thing 1* subscribes a data d on *Thing 2* at $t = 1$, also with an interval of 2 units of time, but this time without providing its address as callback (see the value "*null*" in the Subscription performed at $t = 1$). Accordingly, *Thing 2* saves the current data d value every 2 units of time (see *save(*$d_{t=3}$*)*, *save(*$d_{t=5}$*)*, *etc.*). *Thing 1* may therefore request one or several historical values

[4] The corresponding function in Fig. 2(a) is noted "Subscription(d,x,`add`)"; d the data to subscribe, x the interval value (in units of time), `add` the callback address.

from *Thing 2* at any time by performing a new QLM-MI read with the correct *Subscription_ID* and the range of requested values. Such a query is performed at $t = 6$, where the requested values are the ones saved at $t = 3$ and $t = 5$ (see the QLM-MI response at $t = 6$). Considering a subscription without callback and returning to the previous example (i.e. with the presence of a firewall), the QLM-MI response could this time successfully reach *Thing 1* because it is a response "directly" induced by a query (i.e. a new QLM-MI read), which was not the case in Fig. 2(a).

4 Case study

The case study presented in this section aims at validating the feasibility of placing subscriptions between different "Things". Two "Things" are considered in this scenario as illustrated in Fig. 3:

1. *Thing 1*: a node located in a house that monitors the indoor temperature. In our scenario, this node is a *Raspberry PI* connected to a temperature sensor,
2. *Thing 2*: a node that displays the indoor temperature evolution over a time period. This node may be a laptop, a smartphone and could be located anywhere; it is only necessary to have a continuous access to the Internet.

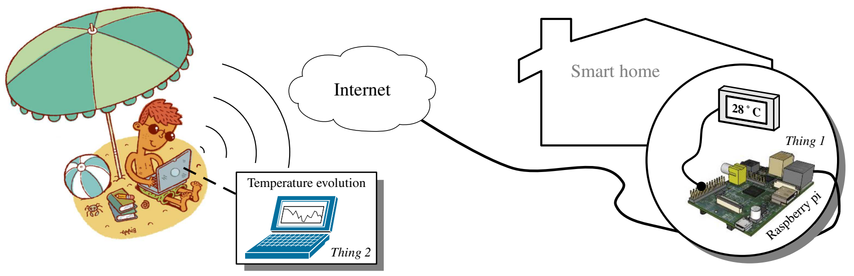


Fig. 3. Illustration of the applicative scenario

In this scenario, the house owner wants to continuously receive the indoor temperature during his/her trip. For this to happen, the house owner (i.e. *Thing 2*) sends a QLM-MI read query for subscribing the temperature data on *Thing 1*. This query is detailed in Fig. 4. The payload format defined in QLM-MI, named "`QLM_mf.xsd`" in line 4, has been presented in [8]. To briefly summarize it, it is defined as a simple ontology, specified using XML Schema, which is generic enough for representing "any" object and information that is needed for information exchange in the IoT. It is structured as a hierarchy with an "Objects" element as its top element (*cf.* line 4), which can contain any number of "Object" sub-elements (*cf.* line 5). The most important attribute of an

```
1  <qlmEnvelope xmlns="QLM_mi.xsd" version="1.0" ttl="3600">
2    <read msgformat="QLM_mf.xsd" interval="1800" callback="http:
         //207.46.130.1">
3      <msg xmlns:xs="http://www.w3.org/2001/XMLSchema" xmlns:xsi="http:
           //www.w3.org/2001/XMLSchema-instance" xsi:type="xs:string">
4        <Objects xmlns="QLM_mf.xsd">
5          <Object class="Thing1_Node">
6            <id>Temp_value</id>
7            <InfoItem class="Temperature"></InfoItem>
8            <id>Time_value</id>
9            <InfoItem class="Time"></InfoItem>
10         </Object>
11       </Objects>
12     </msg>
13   </read>
14 </qlmEnvelope>
```

Fig. 4. QLM-MI read (subscription with callback) sent by *Thing 2* to *Thing 1*

```
<qlmEnvelope xmlns="QLM_mi.xsd" version="1.0" ttl="3600">
  <response>
    <result>
      <return returnCode="200" /></result>
      <requestId>REQ654534</requestId>
      <msg xmlns:xs="http://www.w3.org/2001/XMLSchema" xmlns:xsi="
          http://www.w3.org/2001/XMLSchema-instance" xsi:type="
          xs:string">
        <Objects>
          <Object class="Thing1_Node">
            <id>Temp\_value</id>
            <InfoItem class="Temperature">
              <value>27</value>
            </InfoItem>
            <InfoItem class="Time">
              <value>14:00:04</value>
            </InfoItem>
          </Object>
        </Objects>
      </msg>
    </result>
  </response>
</qlmEnvelope>
```

Fig. 5. QLM-MI response induced by the subscription (generated every 1800 sec)

Object is "class", which specifies what kind of object it is[5] (line 5 indicates it is "`Thing1_Node`"). "Object" elements can have any number of properties, referred to as InfoItem (line 7 and 9 respectively indicate that this message deals with a property named "Temperature" and another named "Time"). Line 2 indicates that this message is a QLM-MI read query, where the interval parameter is set to "1800" (i.e. 30 min) with a callback address of `http://207.46.130.1`, meaning that *Thing 1* has to send the indoor temperature at this address every 30 min. Accordingly, this information is sent via QLM-MI responses every 30 min, one of

[5] An optional attribute called "udef" may be used for specifying the object class using the *Universal Data Element Framework* (UDEF; www.udef.com) taxonomy.

which is detailed in Fig. 5. This response indicates in line 5 that the subscription ID is `RED654534` and that, at this specific moment (2 p.m. as indicated line 14), the indoor temperature is of 27 $^\circ$ C. This scenario was tested throughout a whole day and a generic software application has been designed to allow users to display such information (or similar) on their web browser (e.g. on their mobile devices).

Despite its simplicity, this scenario validates the feasibility of placing subscription for receiving a "continuous" data flow from a distant QLM node. More complex scenarios and applications could further be imagined, e.g. to update CAD documents/simulations in manufacturing applications. This scenario did not present a validation for "client-initiated" communications, but QLM-MI messages used for such communications are similar to those presented in this case study.

5 Conclusion

This paper provides an overview of the QLM Messaging Interface (QLM-MI) standards proposal, which combines the main features of asynchronous, enterprise messaging protocols with those of instant messaging protocols. The main focus of the paper is on a fundamental property that provides the possibility to subscribe (with or without callback) IoT information from a "Things" so as to retrieve it in a deferred way. Subscriptions with callback are particularly useful in applications where it is necessary to receive a "continuous" data flow (e.g. for continuous monitoring, diagnostics, *etc.*), while subscriptions without callback are particularly useful in cases where "Things" are located behind a firewall.

A technical proof-of-concept of this property is provided by the case study in Section 4. The case study has been specified in an overly simple way because the main purpose here is to provide a validation of such a functionality. However, such a property is essential in most real-life IoT applications, especially due to the expansion of the IoT that requires new types of interactions between "Things". The possibility of placing subscription over a long or a short period of time offers a clear advantage compared to the current state-of-the-art.

References

1. Sundmaeker, H., Guillemin, P., Friess, P., Woelfflé, S.: Vision and challenges for realising the Internet of Things. Cluster of European Research Projects on the Internet of Things, European Commision (2010)
2. Atzori, L., Iera, A., Morabito, G.: The internet of things: A survey. Computer Networks **54**(15) (2010) 2787–2805
3. Barnaghi, P., Wang, W., Henson, C., Taylor, K.: Semantics for the Internet of Things: early progress and back to the future. International Journal on Semantic Web and Information Systems **8**(1) (2012) 1–21
4. Kiritsis, D.: Closed-loop PLM for intelligent products in the era of the internet of things. Computer-Aided Design **43**(5) (2011) 479–501

5. Meyer, G., Främling, K., Holmström, J.: Intelligent products: A survey. Computers in Industry **60**(3) (2009) 137–148
6. McFarlane, D., Giannikas, V., Wong, A.C., Harrison, M.: Product intelligence in industrial control: Theory and practice. Annual Reviews in Control (2013)
7. Villa, M., Rotondi, D., Comolli, M., Seccia, C., Huth, H. P., Kloukinas, C., Trsek, H., Claessens, J.: WP 1 – Plug&Work IoT Requirement assessment and architecture. IoT@ Work (2010)
8. Främling, K., Maharjan, M.: Standardized communication between intelligent products for the IoT. In: 11th IFAC Workshop on Intelligent Manufacturing Systems, São Paulo (Brazil). (2013)
9. Gamma, E., Helm, R., Johnson, R., Vlissides, J.: Design patterns: elements of reusable object-oriented software. Reading: Addison Wesley Publishing Company (1995)
10. Eugster, P.T., Felber, P.A., Guerraoui, R., Kermarrec, A.M.: The many faces of publish/subscribe. ACM Computing Surveys **35**(2) (2003) 114–131
11. Perera, C., Zaslavsky, A., Christen, P., Georgakopoulos, D.: Context aware computing for the internet of things: A survey. IEEE Communications surveys & Tutorials (2013)