What's Inside

HOMARD PLATFORM: SPECIFICATIONS

This document provides an overview of the Homard platform and the ecosystem of services around it. For this, the document will guide us from an abstract view of the platform to the specific details of the implementation and end user use cases. HOMARD is a device management platform for the OMA LwM2M protocol. The platform offers functionalities for device management, i.e., remote maintenance, firmware upgrade and open/standard APIs for information reporting. OMA LwM2M is the evolution of OMA Device Management (OMA DM) in order to attend the new requirements and constraints of the Internet of Things, OMA DM is the most extended protocol for remote device management worldwide being used in over 1,4 billions of devices, making OMA LwM2M a very relevant standard based on the experience and knowledge from the most validated and extended protocol for device management (firmware upgrade over the air, remote monitoring, remote reboot, maintenance etc.).

In details, the operations offered by the device management platform are:

Software Management:
Enabling the installation, removal of applications, and retrieval of the inventory of software components already installed on the device and the most relevant firmware upgrade over the air.

Diagnostics and Monitoring:
Enabling remote diagnostic and standardized object for the collection of the memory status, battery status, radio measures, QoS parameters, peripheral status and other relevant parameters for remote monitoring.

Connectivity and security:
Allowing the configuration of bearers (WiFi, Bluetooth, cellular connectivity), proxies, list of authorized servers for remote firmware upgrade and also all the relevant parameters for enabling secure communication.

Device Capabilities:
Allowing to the Management Authority to remotely enable and disable device peripherals like cameras, Bluetooth, USB, sensors (ultrasound, temperature, humidity, etc.) and other relevant peripherals from the nodes.

Lock and Wipe:
Allowing to remotely lock and/or wipe the device, for instance when the device is lost (relevant for devices in open ocean, air etc.), or when the devices are stolen or sold. It enables the remote erase of personal / enterprise data when they are compromised.

Management Policy:
Allowing the deployment on the device of policies which the client (node, device, sensor) can execute and enforce independently under some specific conditions, i.e., if some events happen, then perform some operations.

In addition to the functionalities, OMA LwM2M defines the semantics for the management objects. These objects have been defined with other standards organizations such as oneM2M and IPSO Alliance, which cooperate with OMA to avoid fragmentation and duplication that enables the semantic integration with the Management Objects. OMA LWM2M provides service providers with a secure, scalable, application-independent IoT control platform that provides control and security across multiple industries.
Thereby, this extension will also enable the integration into other initiatives such as oneM2M , which is the major initiative being led by ETSI and all the members from 3GPP to enable a worldwide architecture for Internet of Things. It has a special focus on Semantic Web and interoperability.

1. LwM2M protocol

OMA Lightweight M2M protocol is a protocol for device management designed for sensor networks for covering demand for machine-to-machine (M2M) environments.

OMA has responded to market demand for a common standard for managing low-power devices on different networks to take advantage of the potential of IoT (Internet of Things). OMA LwM2M protocol, which is designed for remote management of M2M devices and related services, has a modern architecture design based on REST, defines an extensible model of resources and data based on a secure data transfer standard called Constrained Application Protocol (CoAP). LwM2M has been designed by a group of industry experts through the Open Mobile Alliance's Device Management Working Group and is based on IETF (Internet Engineering Task Force) protocol and security standards.

OMA has responded to market demand in the M2M area, understanding that, a common set of standards for managing lightweight devices on different types of networks is not only a good choice, It is a mandatory approach for deploying the potential Internet of Things (IoT). Taking into account the billions of devices that are expected to connect to networks in homes, likewise smart telemetry and medical devices among others, providing, monitoring and managing these billions of connections is an absolutely essential task.

1.1 Communication Architecture

LwM2M communicates through a client-server architecture implemented in the application layer. The LwM2M Client resides on the device and the LwM2M Server performs the maintenance and enablement of service tasks provided by LwM2M. LwM2M devices are primarily devices that provide resources by accessing them in a compressed way, thus LwM2M uses a lightweight and compressed protocol to efficiently access these resources and their data model.

4 interfaces have been defined between the LwM2M Client and the LwM2M Server:

- Bootstrap.
- Client Registration.
- Device management and service enablement.
- Information Reporting.

The architecture is shown in Figure 2, where LwM2M uses the CoAP protocol with UDP and / or SMS communications. Data Transport Layer Security (DTLS) provides security for the UDP transport protocol.

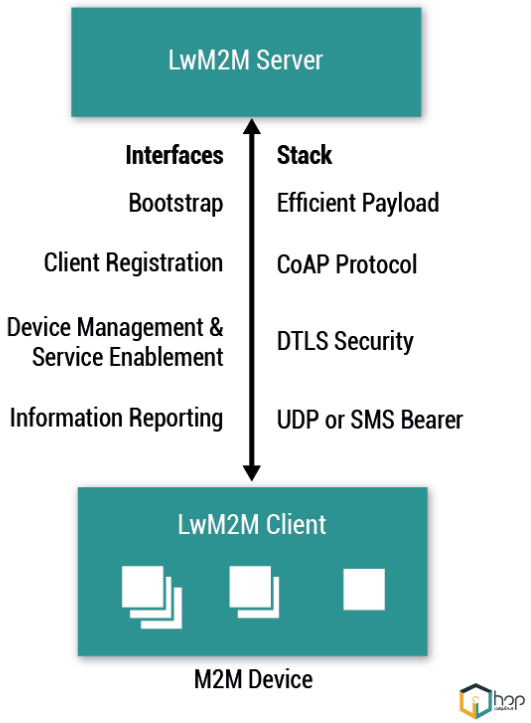

Figure 1. LwM2M Architecture

The LwM2M stack is shown below.

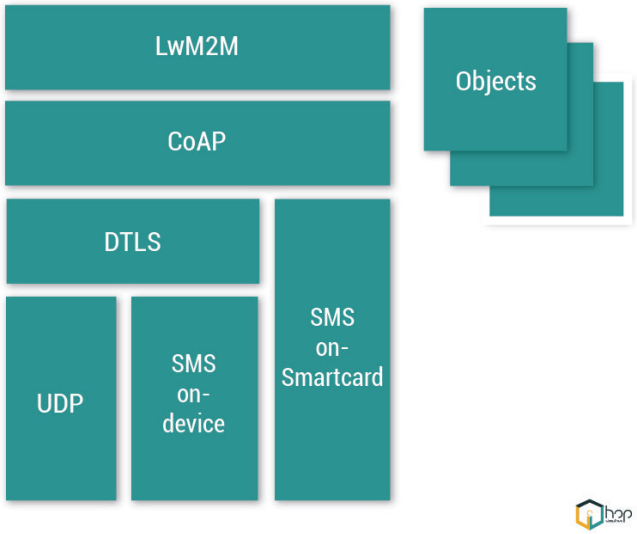

Figure 2. LwM2M Stack

1. 2 LwM2M objects

LwM2M comes with a series of defined basic objects that are needed for the correct operation of the protocol, the following objects are part of the core.

0. Security
1. Server
2. Access Control
3. Device
4. Connectivity Monitoring
5. Firmware Update
6. Location
7. Connectivity Statistics

2. Platform Architecture

The platform architecture consists of a system of distributed and independent services in order to be able to supply them to the users when they request it automatically, the platform also allows the development of third party services so that our users can create their own solutions adjusted to their use cases.
Figure 3 shows a schematic of the platform, which can be divided into 4 categories, Security, where you will find security related services, **Core**, which is in charge of managing LwM2M devices, **Services**, where are housed the services of HOP that will enrich the functionality of the system and **User Interface**, which will allow users to interact with services in a simple and intuitive way.
Core and key components, such as **Data Storage** and **Logic Builder**, are Open Source, released under Apache 2.0 license.

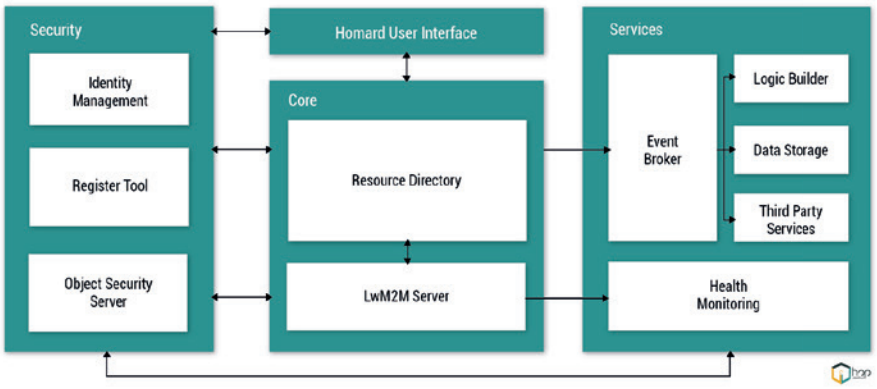

Figure 3. Platform Architecture

3. Components Description

3.1 BootStrap Device Service

The bootstrapping service is the first LwM2M server to which a device is connected. This server, although a priori may seem dispensable, largely simplifies the management of devices. This service is responsible for adding the necessary configuration to the device to get a proper connection with the desired servers. Therefore, it implies the first checkpoint in the system. When a device initializes its registration in the system it is registered in the bootstrap server, storing the server's configuration (address, lifetime, etc.) to which it should be redirected. That allows us to change the servers to which the device will be connected and the objects and instances it will present during the registration process, all of this remotely and without reprogramming the device.

If we follow the flow of processes that will perform the device and its interaction with the bootstrap server we can enumerate the following steps: first the device will be programmed by establishing the BootStrap Device Service as the first LwM2M server with which the device will try to connect. When the device attempts to register on the service, the service will force its connection to the LwM2M servers that are registered in its database, at same time it will write the appropriate configuration if necessary. At this point the device will be able to connect to one or multiple LwM2M servers. One of them will be the Core.

Once the communication between the device and the service is complete, the device will store in flash memory the servers to which it must connect. Thus, it will only re-communicate with this service if it is forced on the device. This operation can be performed if a new server has been added or a modification has been made to one of the existing servers.

3.2 Core

The Homard name comes from Hop Ubiquitous OMA resource directory. Its name is unequivocal since the main functionality is to provide the devices with a common server where they register their resources and capabilities. From this registry, other services can query the device, execute operations on it or subscribe to a change in the information recorded. To give semantics and to provide a framework of information shared between the different institutions around the world the protocol OMA LwM2M is used.

As explained in the previous section, this protocol in the application layer provides a structure to the information that must be represented. In this way, we would have a first level represented by objects and within each object the set of resources that compose it. In addition an intermediate layer is added, the interfaces, which allow multiple representations of the same object in a single device. The existence of the instances may respond, for example, to the existence of multiple sensors connected to the same device.

In this way, the core of the Homard ecosystem, the one who performs the tasks of resource directory, is this service. This will be responsible for managing events, registration or updating devices on the LwM2M server, besides serving as a management, monitoring and control tool for devices created, registered or connected to the platform.

From the point of view of the architecture of the system the core occupies the central part. It is a service composed of two servers that enables end-user communication with the devices, besides collecting event information for the rest of services. This makes it possible to monitor the status of the network or store the information changes that are recorded on any connected device. To do this, the core has two servers: a LwM2M server in the southbound border that implements a constrained stack to be able to make a formal exchange of information with the devices through CoAP and, ultimately, with the LwM2M protocol; and an HTTP server that allows all other services to communicate with the device.

Following the example that begins in previous section. When a device connects to the Core it sends a REGISTRATION request where it will indicate the objects and instances that it implements. The LwM2M server will respond affirmatively and, in the Core, the information presented by the device will be stored so that it can be consultable by any service. After this, the device will continue to perform UPDATE operations to remain connected to the server or indicates changes to the objects or instances it contains. For its part, the Core will allow the query, creation, modification or elimination of resources and instances on the device. Finally, at the moment it is necessary, the device will send a DEREGISTRATION event by closing its connection to the LwM2M server. The Core will therefore maintain the state of the device, understanding it as the queries that are in process, even if the device loses the connection or performs a forced restart.

3.3 Services

The services provide extra functionality to the platform, such as connection monitoring, data storage or logical rule construction, the platform core communicates with the services through a broker (Event Broker) in order to relieve load to the LwM2M server. The broker will be in charge of communicating to the active services, through their respective interfaces. This architecture allow us to activate and deactivate services in a simple and scalable way. These services are designed to be able to deploy individually without depend on each other.

3.3.1 Event Broker

This service consists of a bridge that translate the LwM2M events sended by a REST Interface to OMA NGSI in a representation device-entity, and a broker that communicates with the services.

The bridge receives all the events that generate the devices in the LwM2M server, the read, write and observation operations, as well as the registration, de-registration and update events. When a registration event is received by the bridge, the bridge translates the resource information provided by the LwM2M server into an OMA NGSI entity, this entity will be sent to the broker and will be updated every time any operation on the device occurs.
The broker is in charge of sending the information received by the bridge to the services subscribed to it, the services can be subscribed to complete entities or simply to a field, the broker will keep updated to each service every time that the entity or the field which they are subscribed change.
Thanks to this we managed to have a platform with a high cohesion, low coupling and highly scalable. In which we could easily adopt third party services that cover a specific need without needing to compromise the integrity of the system.

3.3.2 Health Monitoring

The Device and Network Health Monitor service provides an insight into the quality of the link between the device and the LwM2M server, besides other parameters that allows us to know the device load in compute and memory. This service is deployed as an external Homard module that will be notified of any LwM2M messages arriving or sending to the LwM2M server. Through this information we can store the messages exchanged, infer problems in the device and calculate interesting values of the link such as round trip time, number of messages exchanged, number of failures against requests made or the real time that occurs between device updates.
We understand the Device and Network Health Monitor service as a piece that supports the detection of problems between device and server, besides a tool that allows us to monitor the conditions of our network.
In this service are reflected each of the different operations that are performed on the device and its control messages. This information is exposed to the end user either through the graphical interface that we provide and which will be discussed later or through the REST API for requesting at the time it is required.

3.3.3 Health Monitoring

Data Storage is a mass storage service for data and events from the Core. Its purpose is to record each of the operations that are performed in the system. The deployment of the service is adaptable to the needs and can be done from one deployment per instance to a deployment by device.

This service, through the exploitation of the big data, will provide the user with the ability to analyze the data collected by the devices.

3.3.4 Logic Builder

Logic Builder is a logical constructor service that allows users to define complex rules. These rules will have as input the values provided by the devices and as output complex customizable actions. We have consider different actions as alerts, new logical rules within the service that deal with new scenarios or actions in internal or external services through a REST interface.

3.4 Security

3.4.1 Indentity Management

The identity manager allows the user to provide access to the various services that compose Homard through a single sign on, achieving fine-grained access control as well as facilitating the end-user interaction with the system. By using the OAuth protocol the security can be ensured against possible attacks.

3.4.2 Object Security Server

This service is responsible for registering all the proprietary information on the platform, serving as: a database for identity verification and permissions, a recovering service of user devices and, at the same time, as a backup in case of potential attack or loss of information.

3.5 User Interface

The user interface allows users to interact with services in an easy and intuitive way. The goal is to provide the end user with the ability to manage and monitor their devices, implementing the logic of all their devices and analyze the information they collect.
The user interface is therefore a service that can be deployed to simplify the management of the rest of the services that make up the platform. The purpose of this document is not to show the full functionality of the platform. For this reason we will simply provide, through screenshots, the vision of some of the modules developed.

In Figure 4 we can simply see the platform access window. After accessing, we access the dashboard view as shown in Figure 5. Figure 6 corresponds to the Device Management view implemented in the Core through which we can perform operations on the instances and resources of the devices. In this one we can appreciate the structure of the information that imposes the protocol LwM2M. Finally in Figure 7 we can see the result of a query to the Health Monitoring module. In the image we can see statistics of sending and receiving packets between the device and the LwM2M server.

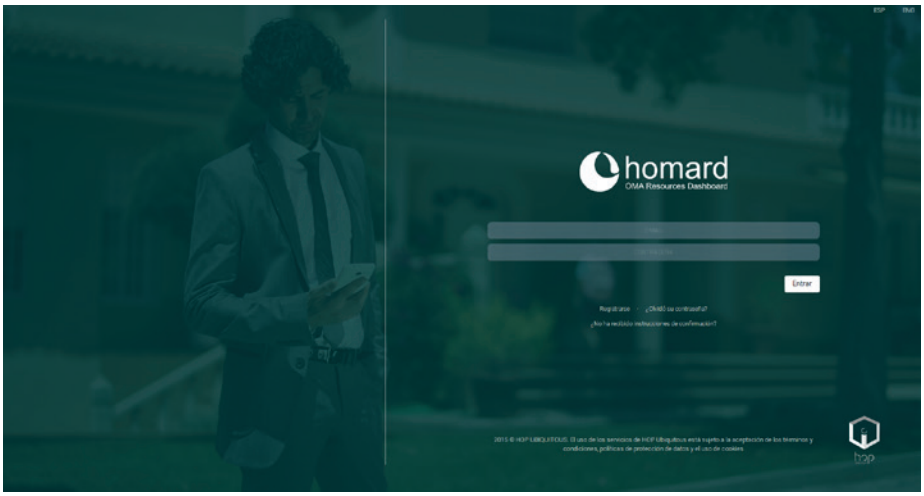

Figure 4. Login view

Figure 5. Dashboard Homard

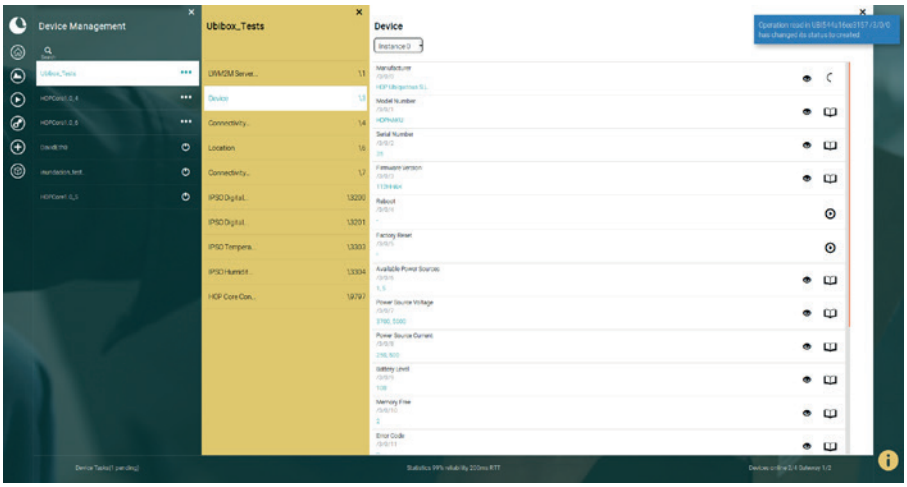

Figure 6. Device Management View.

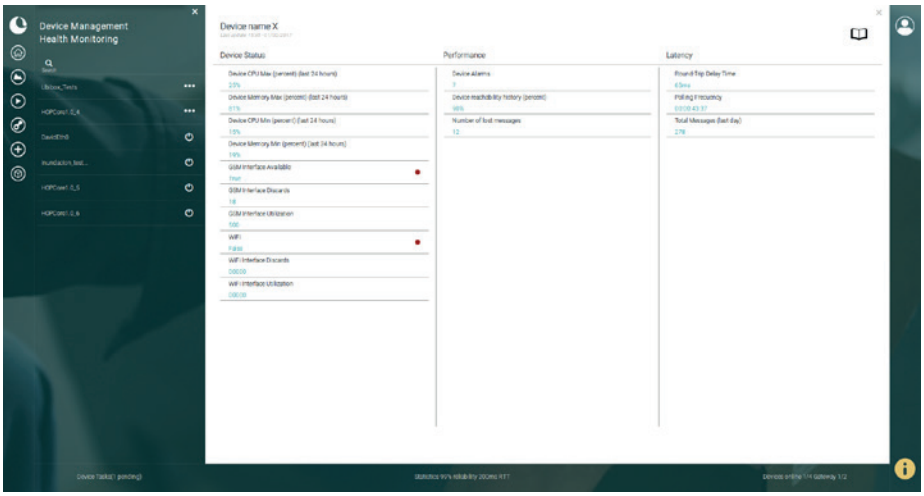

Figure 7. Device and Network Health Monitoring View.