How oneM2M is Enabling More Sustainable IoT Deployments

oneM2M White Paper

2nd edition - September 2022

oneM2M is the global standards initiative that covers requirements, architecture, API specifications, security solutions and interoperability for Machine-to-Machine and IoT technologies. oneM2M was formed in 2012 and consists of eight of the world's preeminent standards development organizations: ARIB (Japan), ATIS (U.S.), CCSA (China), ETSI (Europe), TIA (U.S.), TSDSI (India), TTA (Korea), and TTC (Japan), together with industry fora and consortia (GlobalPlatform) and over 200 member organizations. oneM2M specifications provide a framework to support applications and services such as the smart grid, connected car, home automation, public safety, and health. oneM2M actively encourages industry associations and forums with specific application requirements to participate in oneM2M, in order to ensure that the solutions developed support their specific needs. For more information, including how to join and participate in oneM2M, see: www.onem2m.org.

TABLE OF CONTENTS

1. Introduction 3
2. IoT Sustainability Challenges 3
   - 2.1 IoT Devices 3
   - *2.2* IoT Networks 3
   - 2.3 IoT Systems 4
3. oneM2M Sustainable IoT Features 4
   - 3.1 IoT Digital Twins 4
   - 3.2 IoT Message Profiles 5
   - 3.3 IoT Device Triggering 6
   - 3.4 IoT Event Processing 7
   - 3.5 IoT Time Synchronization & Compensation 8
   - 3.6 IoT Group Management 9
   - 3.7 IoT Interworking 10
   - 3.8 IoT Scheduling and Throttling 11
4. Conclusion 12

1. Introduction

Today, IoT devices already outnumber smartphones 2 to 1 and this gap is expected to widen to 4 to 1 by 2025. Due to their growing numbers, optimizing power consumption and the carbon footprint of IoT devices and networks has been a top priority for oneM2M. oneM2M, a global IoT standards body specifically targeting the standardization of a common service layer for IoT networks and devices, has a dedicated sustainability committee focusing on how IoT networks and devices can be deployed and maintained in a more sustainable manner. As part of its charter, the oneM2M Sustainability Committee has studied sustainability challenges for IoT deployments and has identified several supported features of the oneM2M standard which can be used to address these challenges. The findings of this study are captured within this white paper which further delves into the topic of IoT sustainability and builds upon the first edition of this white paper series from the oneM2M Sustainability Committee.

2. IoT Sustainability Challenges

2.1 IoT Devices

IoT devices differ from smart phones in a few key ways. IoT devices typically handle comparatively small amounts of data and are instead optimized to prioritize battery life. Battery-powered IoT devices are required to remain operational for long durations (e.g. a decade or more) without recharging or replacing their batteries. These devices often have a much smaller battery or no battery at all. Some IoT devices rely on harvesting ambient energy from sources such as solar, thermal, RF, and piezoelectric vibrations. As such, to conserve their energy, many IoT devices need to remain disconnected and powered off for long durations until they are awoken to perform an operation. For instance, a sensor that wakes-up monthly to report a reading and then returns to sleep. Other devices must leave a minimal amount of circuitry powered so they can be remotely triggered to fully power-up if/when needed. For example, an actuator that must be sent a remote command to close a critical valve in a timely manner. Unlike smartphones which can rely on users for daily recharging, and/or transitioning in and out of low power modes when needed (e.g., switch to airplane mode, or dim the device display), IoT devices typically cannot rely on user interaction. Instead, connected IoT devices must rely on more autonomous forms of management and control (within the devices themselves or in the networks they connect to), in order to help them conserve energy.

2.2 IoT Networks

Networks supporting IoT devices typically consist of a diverse collection of IoT devices as well as non-IoT devices. These devices often have different and even conflicting requirements such as network bandwidth, latency, and reliability. Balancing and meeting the requirements of these different types of devices can be challenging for networks let alone doing so in a sustainable manner. To deploy networks more sustainably, network protocols and the usage of the network by devices must be further optimized. For example, protocols must be optimized to minimize unnecessary overhead within individual messages as well as reduce the overall number of messages exchanged over the network. In addition, scheduling of communication to reduce the peak load on the network is also very important. These optimizations can smooth out peaks in demand as well as reduce the overall load on network resources (e.g., servers, switches, routers). In turn, networks can be deployed with less overall resources which reduces their energy and carbon footprints.

2.3 IoT Systems

IoT systems consist of a network of networks with each network typically consisting of different types of devices, protocols, services, and applications. These differences pose sustainability challenges for IoT systems. For example, the lifetime of an IoT system is often dictated by how well the system can adapt and interwork diverse networks and devices together with one another. When older networks and devices are unable to be easily interworked with newer ones, then this typically requires swapping out older technologies and replacing them with newer ones. This not only can be costly from a financial perspective but also from a carbon footprint perspective since it results in large amounts of e-waste. Enabling systems with the capability to efficiently interwork older networks and devices with newer ones can prolong the lifetime and significantly reduce the carbon footprint of IoT systems.

3. oneM2M Sustainable IoT Features

oneM2M has defined several key features within the oneM2M standard to minimize the power consumption and carbon footprints of IoT deployments. Some examples are shown in Figure 1 and include IoT digital twins, message profiles, device triggering, event processing, time synchronization and compensation, group management, interworking, and scheduling and throttling. These features are supported within the individual services defined within the oneM2M service layer. The oneM2M service layer is typically deployed on a cloud server and depending on deployment requirements may also be deployed on edge gateways or servers and even on IoT devices themselves if they have sufficient resources (e.g., compute, memory, and energy).

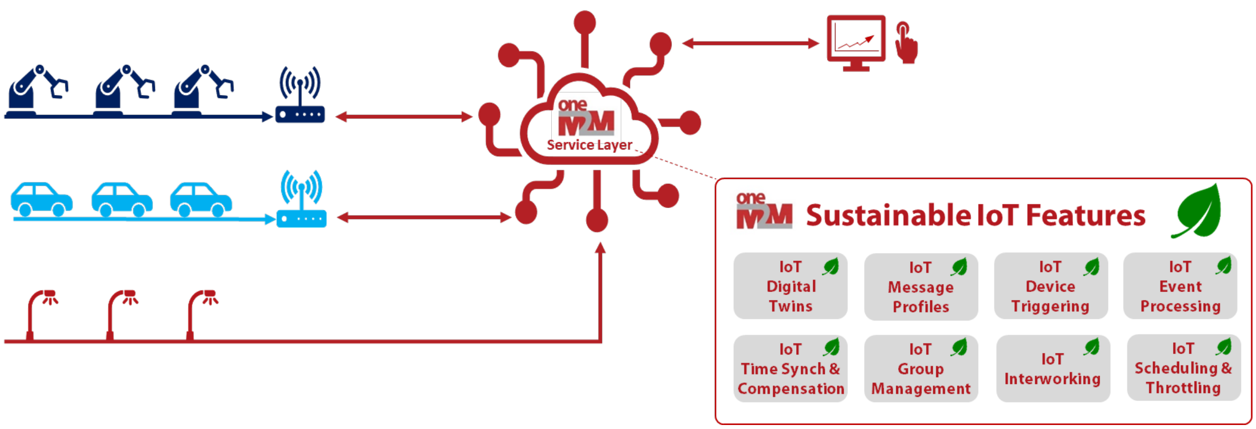

Figure 1 - oneM2M Sustainable IoT Features

3.1 IoT Digital Twins

Digital twin technology enables physical IoT devices deployed in the field to be represented by digital counterparts that reside within services hosted on a cloud or edge server. These digital counterparts, or twins, are then used to exchange information between the devices themselves and the applications and services that wish to interact with them. For example, a sensor can publish readings to its digital twin which are then consumed by applications and services. Use of digital twin technology can provide a huge power consumption benefit for IoT devices because digital twins allow IoT devices to disconnect, power down and sleep for longer durations of time without impacting the functionality of applications and services that require data from these IoT devices. Even while IoT devices are sleeping, their data can be accessed from their digital twins.

Digital twin technology also reduces the number of requests IoT devices must send and receive since devices only need to interact directly with their digital twins instead of with all the applications that may be interested in interacting with the device. An IoT device is only required to wake-up long enough to send an update to its digital twin, and then can immediately go back to sleep. Even an actuator-type device, which requires receiving commands, can benefit from digital twins because actuator commands can be stored within the digital twin by applications. The IoT device can then wake-up based on its own schedule, check its digital twin for any new commands, perform these commands, update its digital twin with the results, and then return to sleep.

As shown in Figure 2, the oneM2M standard defines several standardized digital twin resources providing developers with flexible options for representing their devices and applications as digital counterparts in the oneM2M service layer. Based on use case requirements, developers can choose different types of digital twin resources. Using standardized (rather than proprietary) digital twins enables increased levels of interoperability and streamlined communication and sharing of data between devices, applications, and services. For example, use of standardized digital twins minimizes costly copying and translating of data often required to interwork different proprietary digital twin technologies with one another.

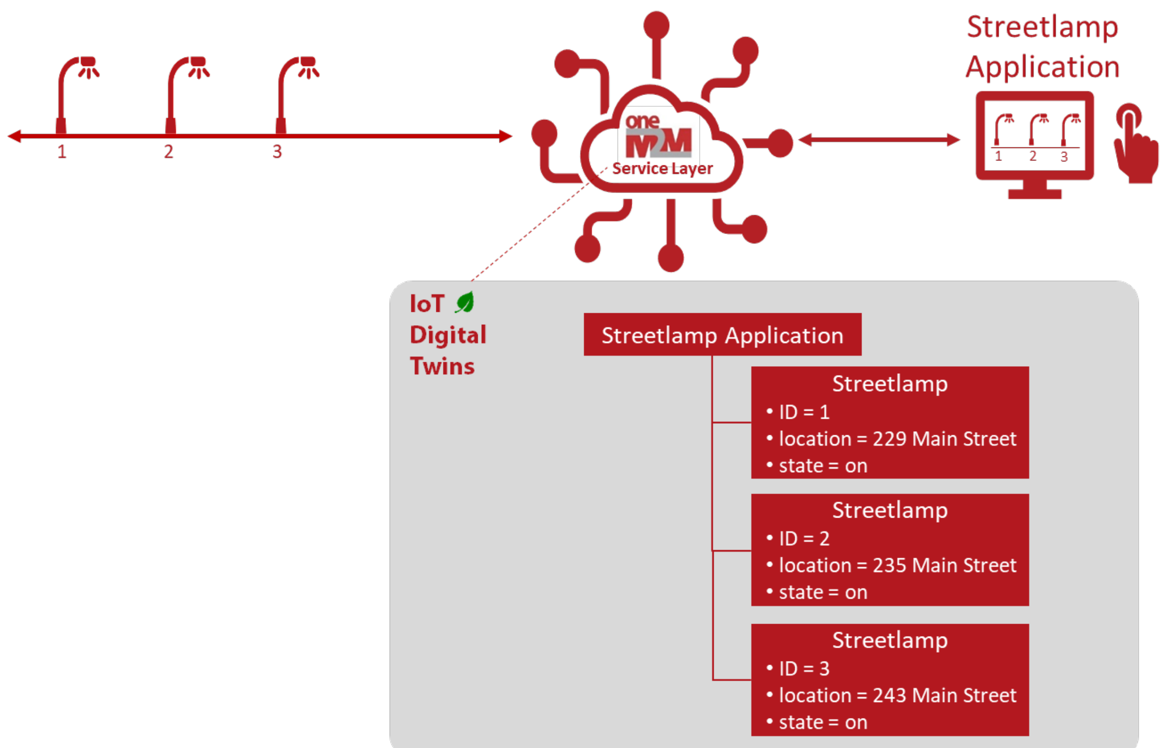

Figure 2 - oneM2M Digital Twins Feature

3.2 IoT Message Profiles

Many IoT devices generate a series of sensor readings on a periodic or event basis. The number of bytes of data included within each message the device sends or receives can have a major impact on the battery life of the device and the amount of congestion and overhead on the network. The larger the message sent or received over the network, the longer the device must remain powered to send or receive the message. Over time, this overhead can add up and dramatically reduce the lifetime of a device's battery.

In many IoT use cases, there is a certain amount of information contained within the messages a device sends that is static and repetitive in nature. For example, certain fields within application protocol headers or within in the data payload of messages such as the name, description, or location of a device which may seldom or never change. This static information is critical to the services and applications (e.g., analytics) consuming and processing the messages. However, this static information can result in significant overhead on the device and the network if included within each and every message the device sends or receives. To alleviate this, oneM2M message profiles can be used as shown in Figure 3. Within a oneM2M message profile, static informational elements can be defined along with criteria defining which devices and/or types of messages a profile is applicable to. Message profiles can be configured within the oneM2M service layer which an IoT device sends it messages to. When the oneM2M service layer receives messages, it compares the messages against the profiles and if a match is found, the messages are enriched with the static information defined in the profiles and further processed by the oneM2M service layer and applications. This allows devices to send only the bare minimum amount of information that has dynamically changed per message (e.g., a sensor reading) and streamline the size of the message.

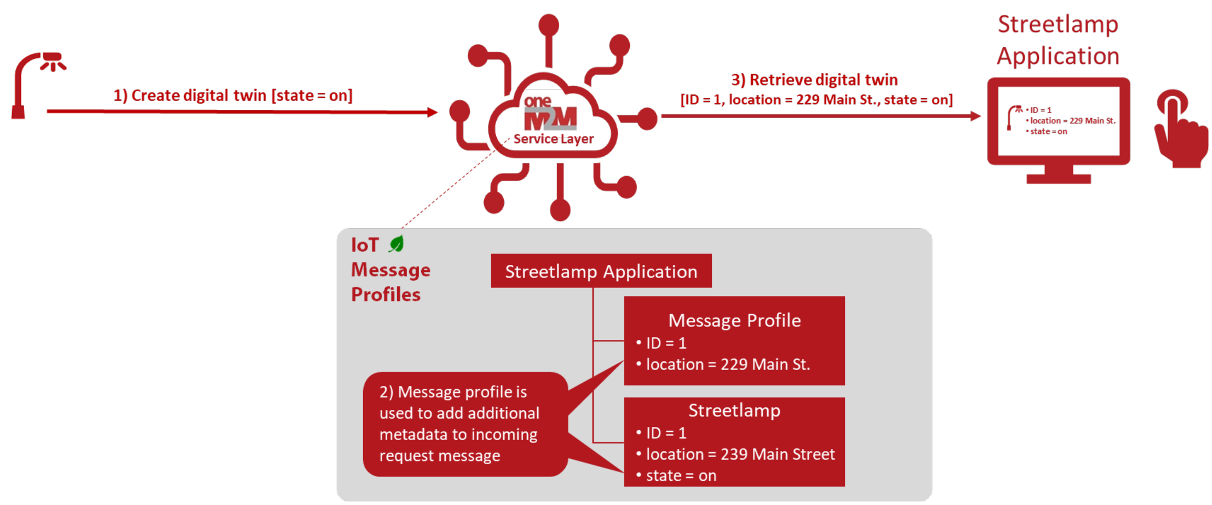

Figure 3 - oneM2M Message Profile Feature

3.3 IoT Device Triggering

oneM2M device triggering enables IoT devices to "sleep" for long periods of time when not in use, and intelligently "wake up" when needed. IoT devices register to the oneM2M service layer and provide instructions for triggering the device. The instructions include a device's contact information as well as its sleep schedule. Once registered, the IoT device can power down to conserve power. Based on its sleep schedule, the device can periodically wakeup for a short duration of time to listen for triggers. When listening for triggers, the device can optimally enable only a small, select portion of its circuitry required to receive a trigger. This allows the device to consume a relatively small amount of power. If the oneM2M service layer needs to communicate with the device (e.g., send the device a command to perform an operation), it schedules a trigger to be sent to the device during the scheduled wake up time of the device. When the device receives the trigger, it fully powers-up and connects to the network.

Once connected, the oneM2M service layer can communicate with the device to send it a command to perform. After performing the command, the device powers back down and goes back to sleep to continue conserving its power. Hence device triggering functionality allows devices to significantly extend their battery lives while at the same time still remain reachable to IoT applications. An example is shown in Figure 4 involving triggering a device to the connect to the oneM2M service layer to perform a software update.

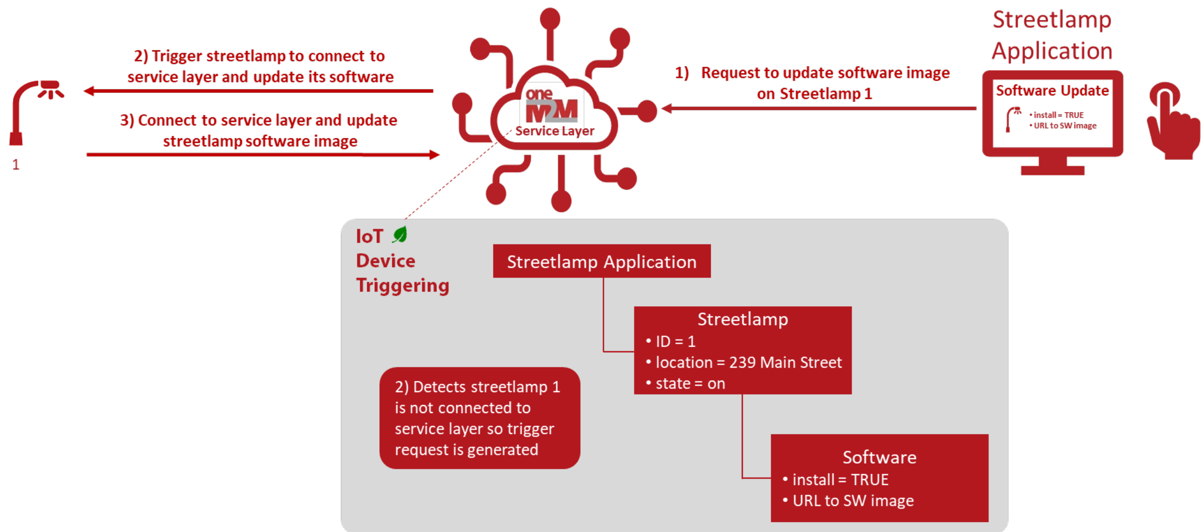

Figure 4 - oneM2M Device Triggering Feature

3.4 IoT Event Processing

Offloading event detection and processing from user applications to the oneM2M service layer reduces the overall number of messages exchanged over the network and the amount of data that needs to be exchanged and stored within an IoT system. Figure 5 shows an example of oneM2M event processing. Rather than an application retrieving sensor readings published by the devices and checking whether the sensor readings have crossed a certain threshold value of interest, the application instead offloads an event detection process to the oneM2M service layer. Within the process, the application specifies the applicable devices, the data of interest from these devices, conditions of interest for this data, and actions that are to be performed if/when these conditions are detected. For example, an application can define and offload a process to the oneM2M service layer which is used to turn-on cameras, alarms, and lights when an unknown intruder is detected.

Offloading event detection and processing to the oneM2M service layer eliminates the need to send individual copies of device data to an application to process. This greatly reduces the amount of messaging and data exchanged in the system. It also enables the pooling and reuse of compute and storage resources of the oneM2M service layer by the IoT devices and applications in the system. For example, the same data offloaded by IoT devices to the oneM2M service layer can be used for monitoring conditions defined not just in a single process from a single application, but from multiple processes from different applications. As result, offloading IoT event detection and processing can significantly optimize resource and energy consumption in the overall system.

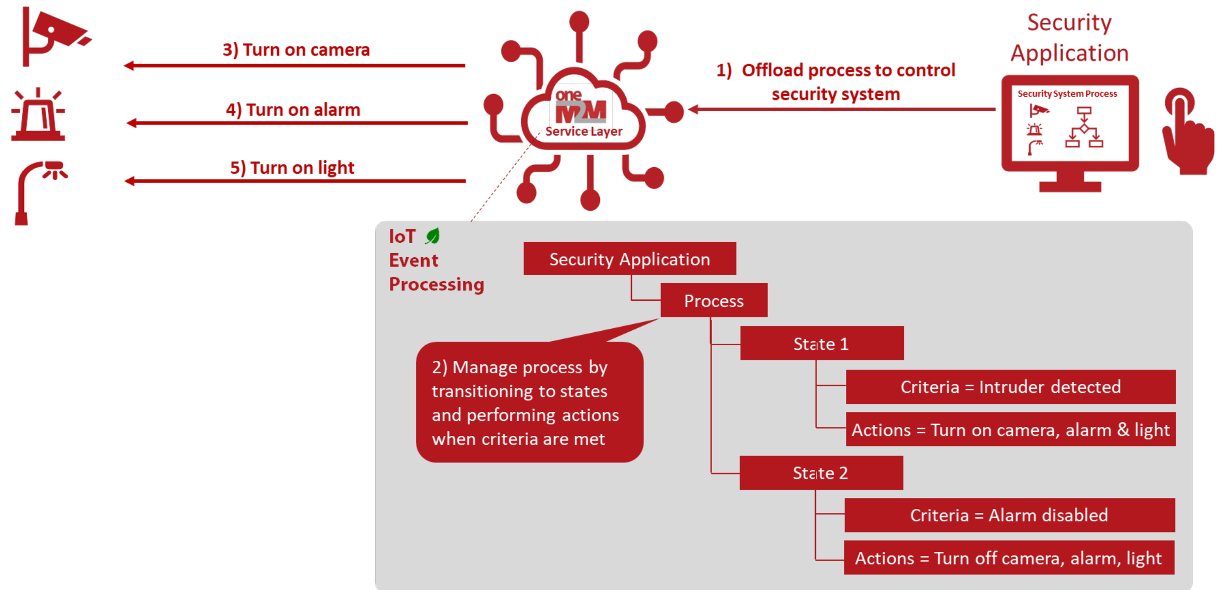

Figure 5 - oneM2M Event Processing Feature

3.5 IoT Time Synchronization & Compensation

Unlike non-resource constrained devices which have the capability to keep their local time synchronized with other devices in the network using technologies such as GPS and Network Time Protocol (NTP), many IoT devices lack this capability due to their limited resources and capabilities. The inability for many IoT devices to keep their local time synchronized with other devices, services and applications in the network can have catastrophic effects. For example, if a patient's medical device timestamps a patient's sensor reading with a date and time which is not synchronized with the medical staff and equipment that is analyzing the readings, then this can lead to a patient being mis-diagnosed and treated.

The oneM2M time management service supports a set of time management capabilities to minimize time synchronization overhead on IoT devices allowing these devices to remain synchronized with the rest of the system without introducing a lot of extra overhead and complexity to devices. As shown in Figure 6, the oneM2M service layer supports the capability to perform time compensation on behalf of IoT devices by adjusting time related metadata contained in the messages sent by a device to the oneM2M service layer. The oneM2M service layer is able to monitor and detect time synchronization offsets of IoT devices and the rest of the system. Based on the detected offsets, the oneM2M service layer is able to compensate for the time offsets on behalf of the devices. For example, when messages are received from devices which contain timestamp information, the oneM2M service layer can adjust the timestamps to compensate for any detected time offsets. This simplifies devices by alleviating them from the burden and overhead of having to maintain time synchronization with the rest of the system. Hence devices are kept simple and consume less compute and energy.

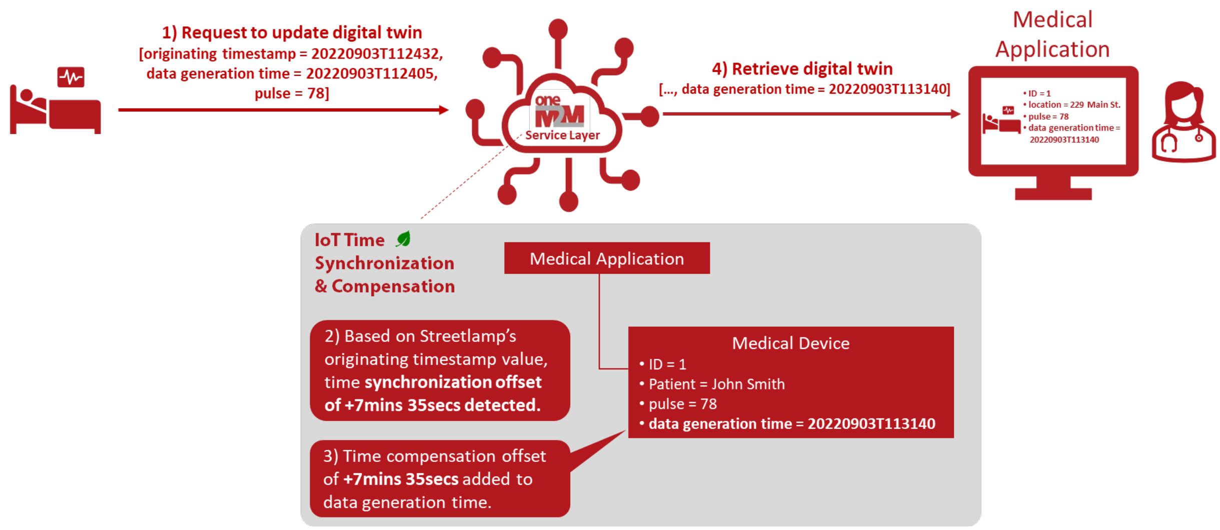

Figure 6 - oneM2M Time Synchronization and Compensation Feature

3.6 IoT Group Management

For certain use cases, deploying the same type of IoT sensor device in groups can have significant sustainability benefits. For example, deploying groups of IoT sensors to monitor soil moisture in a field of crops. This not only adds redundancy in case a sensor device fails but also can extend the lifetime of individual devices. For example, rather than a single device having to wake-up and report a sensor reading once a day, a group of 7 sensors deployed in proximity to one another can allow individual sensors to wake-up just once a week and collectively still provide sensor readings once a day. Not only can this extend the overall lifetime of an IoT deployment, but it can also reduce its carbon footprint. This is because there are also significant environmental costs of having to dispatch technicians which requires costly truck rolls involving fuel usage and emissions which need to be considered (especially for IoT networks deployed in remote locations).

oneM2M supports several features to manage the grouping of IoT devices. For example, oneM2M supports formation of groups and fanning our requests to the group members. When fanning out requests, oneM2M supports load balancing requests across a group of devices as shown in Figure 7. When the oneM2M service layer receives a request targeting a load balancing group, a device is selected from the group to receive the request. The selection of the device can be made based on a configured algorithm such as the least loaded device (e.g., device with highest battery level), round-robin, weighted round-robin, etc. The oneM2M service layer then sends the request to the selected device (but not the other devices in the group). In doing so, the oneM2M service layer intelligently load balances requests across the group of devices and prolongs the deployment lifetime of the group of devices such that the group can monitor a given region for a longer period of time than a single device is able to (with the same measurement sampling rate).

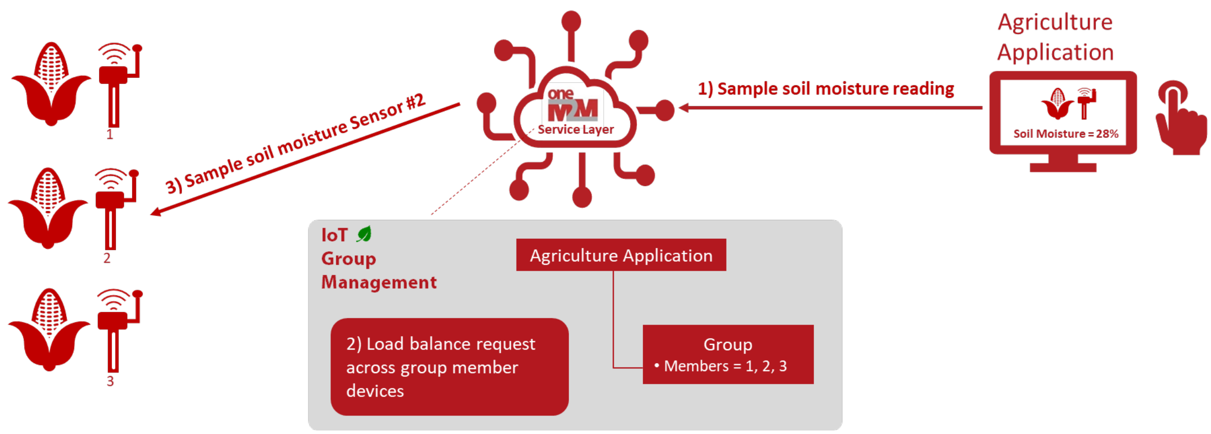

Figure 7 - oneM2M Group Management Feature

3.7 IoT Interworking

Many large IoT deployments require interconnecting different IoT platforms and networks together to enable communication between devices and sharing of data. Smart Cities is a great example. The day-to-day operations of most cities is split-up and run by a number of different departments (e.g., police, fire, transportation, trash, water, etc.). These departments typically operate with independent budgets, staff and IoT service platforms. For example, the water department may deploy a separate platform of sensors to monitor the health of the water infrastructure and collection of water meter readings from consumers. Likewise, the transportation department may deploy a separate platform to manage traffic lights, electronic signs, and street lights throughout the city. Interworking and sharing information between these different platforms can be useful. For example, notifying the transportation department when there is a water main leak so traffic can be diverted.

Interworking of platforms can be challenging and typically requires developing complex and costly custom over-the-top application code to glue these platforms together. This is the case since these platforms tend to be proprietary and do not use standardized communication protocols and data models. This complexity often increases when interworking newer and older platforms together due to the increased technology disparity between them. Therefore, many cities find themselves in a never-ending refresh cycle, replacing older proprietary platforms with newer ones in hopes of reducing their costs and frustration. This is problematic from a sustainability perspective since frequent refreshing of older platforms with newer ones generates a significant amount of e-waste.

oneM2M can help extend the life of IoT deployments. Deploying platforms based on the oneM2M standard ensures standards-based interoperability and also avoids vendor lock-in. As shown in Figure 8, oneM2M also supports capabilities to ease the complexity of interworking different IoT technologies which helps extend the deployment lifetimes of IoT platforms. For example, the oneM2M standard has defined standard interworking proxies that interconnect many of the common types of local IoT networks and devices such as LWM2M and OPC-UA as well as several others. In addition, oneM2M defines a standardized framework for developers to build their own interworking proxies for connecting 3rd party proprietary platforms and services together if/when needed. This standardized framework reduces the complexity and maintenance of interworking by providing developers with a reusable and standardized method for interworking that has been simplified and vetted by industry experts.

Since oneM2M is a standard and not proprietary, there is a level of assurance that its functionality is complete, well-defined and will continue to maintain backwards compatibility support across releases. This can put minds at ease that the functionality used to interwork IoT platforms together with one another will be cost effective and non-complex to manage. Therefore, the lifetime of older technologies can be extended, and their sustainability footprints reduced.

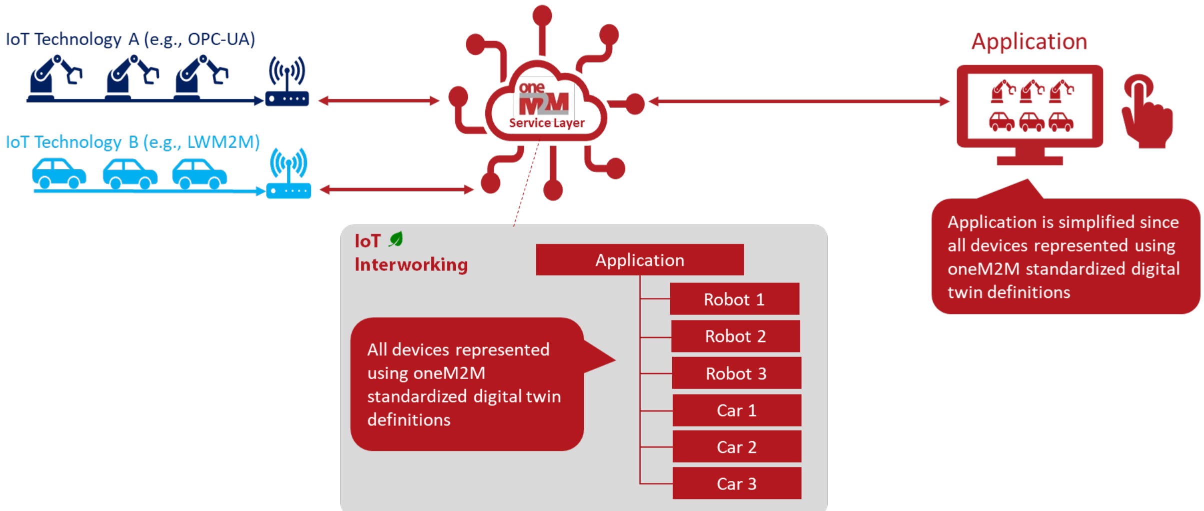

Figure 8 - oneM2M Interworking Feature

3.8 IoT Scheduling and Throttling

By managing peak demands on communication networks, network operators can optimize the amount of network equipment (e.g., switches, routers, servers, cell towers and base stations) required to meet their customer demands. This can significantly reduce the energy consumption and carbon footprints of their networks.

As shown in Figure 9, the oneM2M standard supports several features to help network operators minimize the peak demand on their communication networks. Since the oneM2M service layer is deployed in between IoT devices and IoT applications and also layered over top of underlying communication networks, it is well positioned to help manage IoT communication and peak demands on communication networks. Devices and applications exchange messages with one another via the oneM2M service layer. The oneM2M service layer supports the capability to schedule the message exchanges occurring between IoT devices and applications. Within oneM2M service layer, configurable communication schedules for each IoT device and each application are supported. Using these schedules, the oneM2M service layer schedules the exchange of messages based on availability of the devices and applications as well as the priority of the messages being exchanged.

oneM2M also supports interworking with underlying communication networks to monitor and detect congestion and overloading conditions in these networks. If congestion occurs in a communication network, the oneM2M service layer can be notified and can take action to help alleviate the network congestion. One supported action includes adjusting the communication schedules of IoT devices and applications such that communication takes place in off-peak non-congested periods.

oneM2M also supports throttling of messages by buffering them during periods when communication networks are congested and then forwarding the messages when congestion has subsided. When buffering IoT messages, oneM2M also supports batching multiple individual messages into a single message such that batch message can be more efficiently delivered across a communication network (rather than sending many small individual IoT messages).

oneM2M also supports mechanisms to notify devices and applications if/when communication networks become congested such that the devices and applications themselves can take action such as reducing the size and/or frequency of their messages.

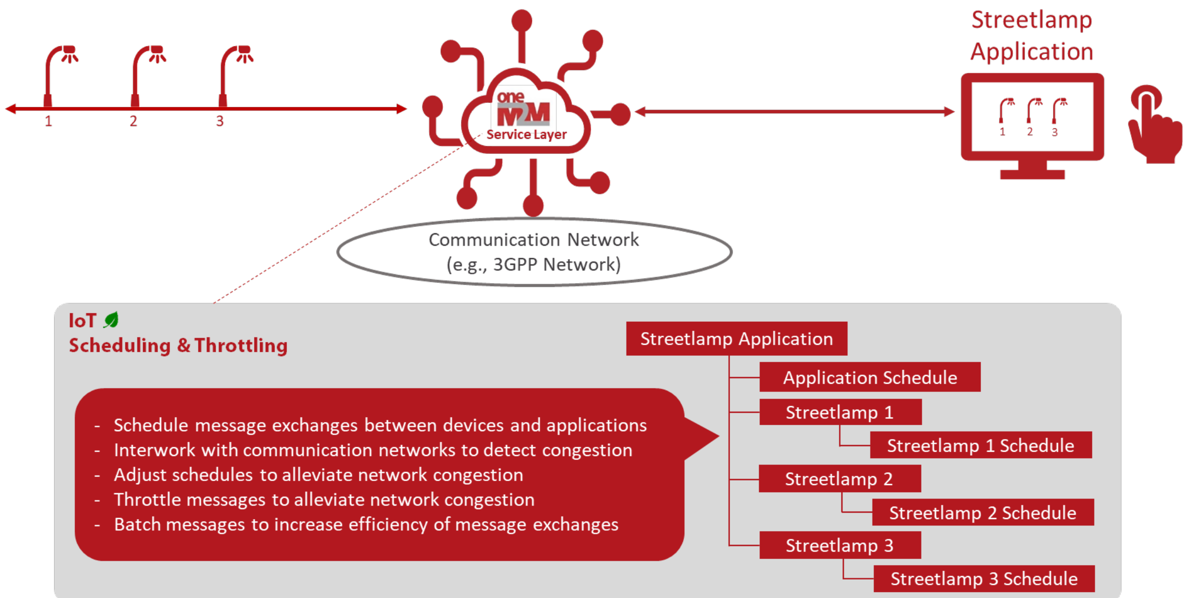

Figure 9 - oneM2M Scheduling and Throttling Feature

4. Conclusion

As the mass scale out of connected IoT devices, IoT networks and IoT systems continues, the importance of responsibly deploying IoT technologies and minimizing the carbon footprint of IoT deployments is becoming more critical. The standardized oneM2M service layer supports several features to enhance the sustainability of IoT. These features can help significantly reduce the energy consumption, increase the longevity, and minimize the amount of e-waste of IoT deployments. For more information about the standardized oneM2M service layer and how it can enable more sustainable IoT deployments, please refer to the oneM2M specifications which are publicly available at www.oneM2M.org.