Adaptable Protocol Selection for Reliable Smart City Services

Asma Elmangoush[1,*] and Thomas Magedanz[2]

[1]*The College of Industrial Technology – Misurata, Libya*
[2]*Technische Universität Berlin, Germany*
Email: asma.elmangoush@alumni.tu-berlin.de; Thomas.magedanz@tu-berlin.de
[*]*Corresponding Author*

Received 27 October 2016; Accepted 27 September 2017;
Publication 20 October 2017

Abstract

The Machine-to-Machine (M2M) communication intends to promote seamless interaction between connecting devices to enable the automation of decision making based on aggregated data. This forms the new Internet of Things (IoT) paradigm, which aims to increase the level of system automation by enabling devices and systems to exchange and share data, facilitating the communication for other industry branches. This article addresses the requirements of implementing reliable Smart services, with consideration to the heterogeneous traffic patterns rises from the huge variant devices. The main challenges for M2M communication are driven from integrating low-power devices, each produce a different pattern of traffic, and low bandwidth networks. The heterogeneity of integrated communications technologies, targeted service domain and data representation, highlights the need to study the communication requirements of various services and the traffic patterns in the system.

Keywords: Smart City, Internet of Things, Adaptability, Interoperability.

Journal of Cyber Security, Vol. 6_1, 57–76.
doi: 10.13052/jcsm2245-1439.613

1 Introduction

The aim of the Machine-to-Machine (M2M) communication paradigm is to connect any connectable device to create Smart Cities. Promoting Smart City platforms is challenged by the existing Internet technologies and infrastructure, which were designed for less complex usage. The latest practice of the Internet, which integrates rich real-time applications to the legacy services, motivates the ongoing research work toward the Internet of Things (IoT). The vision of the IoT is to enable objects to be connected any-time, any-place, with anything and any-one ideally using any-path/network and any service. From a technical point of view, this vision could not be accomplished by implementing one technology; instead, several complementary technical developments shall provide functionalities and capabilities to assist in bridging the gap between the virtual and physical world.

Smart City applications are driven from enterprise and government needs. While the enterprise focus is on efficiency and cost reduction, the government focus is on sustainability, safety and socio-economic impacts. Thus, features and requirements of M2M applications differ in multiple aspects, and therefore the associated data flows will demand the assignment of different levels of priorities. It might be the case that these requirements differ based on the desire of the user or project owner and the available budget. For the deployment of an M2M system in any domain, such as health care, Smart Home or Industry, it's important to start by specifying the environment conditions and requirements. The understating of operating specifications including the integrated devices, traffic pattern, samples rate, and delay tolerance level, have a great impact on designing the final solution. Many options are available when selecting the components, platform capabilities and access technologies to be used in the system. The work presented here is motivated by the evolution of the M2M ecosystem towards Smart City realization which presumes a high number of heterogeneous devices communicating anywhere and anytime and interacting with various services.

In reliability engineering, reliability is the probability that a system can provide uninterrupted delivery of acceptable service for a given mission time [1]. The principle of limiting human interaction in M2M systems as well as integrating a massive number of devices with heterogeneous computational capabilities into it, demands a fully interconnected and application-agnostic system. The challenge is to build a connectivity aware M2M node that performs reliable and secure communication interweaving.

This article presents an adaptable framework to support the dynamically transportation of heterogeneous traffic within Smart City environment and mediation interfaces with other M2M platforms. The objective of the framework is to increase the adaptability of M2M nodes in handling flows of requests from different connected objects, being resource-constrained or resource-rich, and different applications demanding heterogeneous Quality of Service (QoS) requirements. Generally, the framework is compatible with latest international standards for M2M communication. It aims to enhance the data transmission process between connected objects by dynamically plugging in and out transport protocols to match the network conditions in order to enable a reliable end-to-end data delivery. This work can be considered as a foundation for an M2M-based Smart City platform that addresses the requirements of heterogeneous vertical domains. In this article, the characteristics of the desired M2M systems are identified based on the features and requirements of common Smart City applications. These characteristic were considered in the design of the proposed system. Some applications in the Smart City context have critical requirements in terms of data latency and demanded throughput, such as eHealth and Smart Grid. However, current networks treat traffic generated by different applications in the same way regardless of the content or its source.

The rest of the paper is structured as follows: Section 2 provides background information on M2M communication; in Section 3 presents the system architecture and design considerations. Section 4 presents the proof of concepts evaluation results. Finally, we conclude the paper in Section 5.

2 Background and Related Work

2.1 M2M Communication Overview

Developing a detailed understanding of the traffic characteristics of the network is very important to achieve the goal of delivering reliable services. The current cellular mobile networks are designed for human communication, and therefore are optimized for the traffic characteristics of human-based communication applications, i.e. communication with a certain session period, data volume, interaction frequency and patterns. A study in an operational 3G network with the Busplus application has shown that even a few tens of M2M terminals can lead to a significant network accessibility degradation (up to 60%) [2]. To overcome the shortcomings, improve the network performance and further guarantee the application's Quality of Service (QoS), reliable traffic models and reference scenarios are necessary. They could help in better understanding the M2M traffic pattern and characteristics.

In our previous work, a conceptual classification of M2M traffic classes based on the generated payload size and sampling rate was presented [3]. Generally, the following characteristics of the desired M2M systems are detected based on the understanding of the key features and requirements of common M2M applications:

- Dynamically changing topology: A massive number of devices are foreseen to be existing in the service coverage of a single wireless base-station, and concurrent network access attempt from these devices. Some of these connected devices are establishing connections in a frequent manner, operating in an Adhoc mode, or requiring mobility support.
- Diverse traffic patterns: The heterogeneity of aggregated data from M2M nodes, in terms of size and frequency results on new traffic characteristic dissimilar to conventional human-to-human (H2H) traffic. It is worth to mention that the current communication platforms are optimized for H2H communications, and require essential improvements to support the heterogenetic nature of connected objects and their capabilities, as well as the different trend of traffic patterns [4] associated with M2M communication.
- Heterogeneous QoS requirements: The wide range of services imposes various levels of QoS requirements, which may require priority-based protocols at different layers.
- Heterogeneous device's capabilities: The magnitude number of connected nodes consists of devices with various capabilities; ranging from resource-constrained devices (e.g., simple tags) to resource-rich devices with high computation/storage capabilities.
- Autonomous information exchange: High level of system automation in which the devices and systems can exchange data and make decisions without human intervention.

These characteristics introduce a number of challenges towards the successful deployment of M2M services. Focusing on the connectivity aspects, it is noticed the highly fragmented situation of the current M2M protocol stack needed to be supported due to the variation in connected devices and application's requirements. In the Vehicle Communications experiments done in [5], the results show that vehicles traveling at normal speed (e.g. 60 mph) can exchange at least 10 messages per second, with 3K bits in each message. Other types of services generate different levels of traffic [6, 7].

2.2 Standardized M2M Protocols

Many standard organizations have promoted several activities in the M2M communication domain. In 2012, the oneM2M consortium was established with the aim of consolidating the standardization work in M2M communication. The oneM2M partnership [8] is a consortium of seven international standards development bodies working in the M2M communication standardization. In its first release, oneM2M referred to the REST-full application-level transport protocols: HTTP [9] and the Constrained Application Protocol (CoAP) [10]. Recently, oneM2M specified the binding of the Message Queuing Telemetry Transport (MQTT) protocol [11] within its architecture [12].

While HTTP fits the requirements for web transfers, it requires a TCP session established consuming valuable energy and computation capacity. CoAP is proposed by the CORE (Constrained RESTful Environments) IETF working group to accommodate M2M devices with constrained resources in the integration with existing web services. It uses UDP for transportation and has a small overhead. The disadvantage of CoAP is that the requests might not be processed without retransmission. Other protocols adapting the Publish/Subscribe model include MQTT [11] and Advanced Message Queuing Protocol (AMQP) [13]. Although this model lacks the end-to-end delivery reliability due to the existence of intermediary entities (i.e. broker) between publishers and subscribers, it does provide a sufficient method to forward infrequent status changes to multiple subscribers. While the polling mechanism used by HTTP and CoAP lacks the scalability to handle the status update to a huge number of subscribers.

The second major version of the HTTP protocol, known as HTTPv2 or H2, is focusing on improving performance; i.e. end-user perceived latency, network and server resource usage. One major goal is to allow multiplexing of requests and responses to avoid the head-of-line blocking problem in HTTPv1. A server pushing functionality is also specified to enable pushing resources to clients in a Publish/Subscribe model, this feature is useful when sending notifications on M2M systems. Additionally, an optimized binary encoding for the header is introduced in order to reduce the needed network bandwidth. Table 1 shows a comparison of the M2M transport protocols commonly used in IoT systems.

In addition, new communication technologies have been introduced for M2M communication with the objective of simplifying the deployments and gaining wider network coverage without compromising cost or power consumption. New technical solutions have been implemented that present

Table 1 Comparison of M2M transport protocols

Protocol	HTTP/1.1	HTTP/2	CoAP	MQTT	AMQP
Standards	IETF RFC2616	IETF RFC 7540	IETF RFC 7252	Proposed OASIS standard MQTT	OASIS AMQP
Architecture Style	Client/ server model RESTful	Client/ servers model	Client/ server model RESTful	Brokered style	Brokered style
Transport	TCP/IP	TCP/IP	UDP/IP	TCP/IP	TCP/IP
Messaging	Request/ Response	Supports multiplexing of request/ response	Request/ Response	Publish/ Subscribe	Publish/ Subscribe (P2P or Brokered)
Service levels	All messages get the same level of service	Priority mechanism of streams	Confirmable or non-confirmable messages	Three quality of service settings	Different 3 QoS levels
Data distribution	One-to-one	One-to-one, and one-to-many	One-to-one	One-to-one, and one-to-many	One-to-one, and one-to-many
Security	Typically based on SSL or TLS	Requires TLS version 1.2 or higher	Typically based on SSL or TLS	Simple Username/ Password Authentication, SSL for data encryption	SASL authentication, TLS for data encryption
Header	Text-based	Binary (header compression)	4 Byte binary-based	Fixed-length header of 2 bytes	Header 8 byte
Message size	Configurable by server, recommended to be less than 8KB	Configurable by server	Small to fit in single IP datagram (1500Byte).	Up to 256 MB	Unlimited

an IP-based backhaul where the end-devices are provided with direct connectivity to a base station implementation a long-range wireless technology such as SIGFOX [14] or short-range wireless technology such as Zigbee [15].

Table 2 Comparison of M2M transport protocols

Protocol	Max Range	Max Data Rate (Kbps)	Life of Batteries	Max Num of Devices/ Base Station	Topology
Sigfox	50 km	1	20 years	$\sim$1Mil	star
Neul	10 km	100	10–15	n/a	star
LoRaWAN	15 km	37.5	10	$\sim$10 K	Tree
Z-Wave	100 m	100	Few years	232	Mesh
Zigbee	70 m	20/40/250	Few years	2–6 K	star
Bluetooth low-Power (BLE)	100 m	1M	10–15	7	star
6LoWPAN	200 m	250 K	Few months	100	Mesh

The use of the long- or short-rang wireless technology depends on the application domain and the operational requirements. Table 2 summarized the main characterises of some over-IP Protocols widely used in M2M projects. Many of these technologies are used in a variety of domains including home automation, sensor applications, and Smart Grid. Interested readers are referred to [16] for further discussion and insights of listed technologies for IoT/M2M.

3 The Adaptable M2M Framework

Adding Billions of physical and virtual "things" to the communication system and allowing them to seamlessly integrate into a global IoT infrastructure, requires establishing an effective management platform to provide reliable services to existing customers based on the M2M technology. The M2M communication imposes a number of challenges, such as: managing devices with high level of heterogeneity, optimizing communications and energy usage, and infrastructure scalability. In order to realize the Smart City approach, content from connected devices should be handled with intelligent methods to earn enriched content from aggregated information. M2M middleware platforms are required as a key enabler of reliable Smart City infrastructure, which facilitates the collaboration of various stakeholders to increase the efficiency of administrative services, and developing environment friendly applications.

In this section, the specification of an adaptable framework for handling heterogeneous traffic patterns within a Smart City system is presented. By definition, the proposed Adaptable M2M Transport Framework (AdM2M) is considered to be located in a Middle Node (MN) e.g., a gateway, or in an

Infrastructure Node (IN) e.g., backend server, according to the terminology adopted by OneM2M specification.

3.1 Design Considerations

The standard conformance of the proposed solution is an important prerequisite in order to support the interoperation between different systems and vendors. The architecture of oneM2M partnership is adopted as the basis of the proposed framework. In addition, further extensibility is allowed by means of adding standardized extensions. For example, more protocol binding functionality could be added to the framework.

It is expected that variant applications will be deployed on M2M entities and generate heterogeneous traffic patterns. The key assumptions to be considered in the design of the AdM2M framework related to the involved M2M environment are:

- The connected M2M nodes will include both resource-constrained and resource-rich devices. Consequently, their capabilities of computation power and memory size are heterogeneous.
- The connected M2M nodes are autonomously involved in the system. Some might be located in locations that are hard to access, others are supposed to work with limited human intervention.
- One M2M device/gateway could integrate several sensors and/or run more than one application at the same time.
- One M2M application might be registered to more than one platform.

The proposed framework was exemplified on top of the OpenMTC toolkit [17]. The OpenMTC platform is a prototype implementation of the oneM2M standard to provide a horizontal convergence layer supporting multiple vertical application domains, such as transport, utilities, and EHealth [18]. The OpenMTC features are aligned with the OneM2M Rel.1 specifications [8], providing an implementation of Common Service Entities (CSE) at the Frontend (Gateway/Device) and Backend (Network Server) M2M architecture. The platform supports a client/server based RESTful architecture with a hierarchical resource tree defined by oneM2M, and communication over all interfaces is independent of the transport protocol.

3.2 The Framework Overview

Figure 1 depicts the High-level architecture of the proposed AdM2M framework within a connectivity management unit of a Middle Node (MN) e.g.,

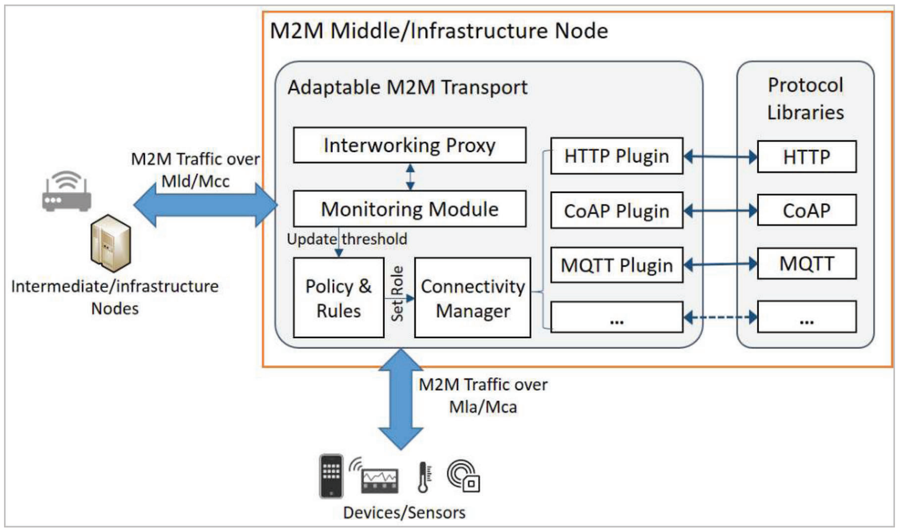

Figure 1 Architecture of Adaptable M2M Transport Framework.

a gateway, or in an Infrastructure Node (IN) e.g., backend server, within an ETSI/oneM2M compatible system to enhance the communication facility of the ETSI SCL/oneM2M CSE. The prime functionalities of an infrastructure/middle M2M node, as specified by the oneM2M CSE, are provided by the core middleware. An internal bus supports the inter-communication between the components by sharing an inner-API based on events, where an internal subscribe/notify mechanism enables internal components and plug-ins to be notified about updated status.

The Adaptable M2M Transport (AdM2M) provides different transport protocol stacks, such as HTTP and CoAP as plugins to support the communication with M2M devices using the proper stack to each use case. The framework is extendable to add more protocol libraries and plugins. From the performed experiments, it was observed the improvement in the performance when using CoAP to push data flows with high rate/small payload, and HTTP for data flows with low rate/big payload. This motivates the further development to support dynamic adaptability on M2M nodes to interaction statues. Furthermore, MQTT protocol provides better support for group communications by adapting the Publish/Subscribe scheme, this improves the forwarding of notification to multiple subscribers.

An application, i.e. Application Service Nodes (ASN), could interact with the OpenMTC, deployed in an IN or MN, in different ways such as sending data/measurements in a telemetry manner, or receiving notifications in response to events that was previously subscribed to. The ASN in the field domain could send requests to the IN directly or via an MN (a gateway that the ASN is registered with). In the latter case, the MN will retarget the requests to the IN in behalf of the ASN. The request destination address (represented by the resource URL) is analysed and if it is not matched to any address on the machine IP address list, the message will be forwarded to the destination IP. This mechanism is implemented as a method to forward requests on a multi-hop route. Enabling the dynamic selection of transport protocol translation and communication channel based on different criteria, such as the cost or reliability, brings a value-add service to M2M communication. These channels can be assimilated to physical interfaces, as Wi-Fi or Ethernet. Furthermore, the Store and Forward (SAF) policy, which enables the handling of different notification streams based on their priority, could be extended to support applying/modifying the transport protocols, i.e. HTTP, CoAP or MQTT.

The AdM2M framework consists of a Monitoring Module that constantly collects a set of parameters related to the data stream generated by registered ASN. Theses parameters include the average roundtrip time of previous requests, sampling data rate and data packets size. The data are presented in constanceInstance resources created in response to application's POST requests, as specified by oneM2M. The Monitoring Module serve in providing an updated vision of the network condition status. An Interworking proxy is also integrated within the framework to enable a syntactic interoperability with other M2M platforms [19].

Based on the application ID (originator of the request), the Policy and Rules component in the M2M gateway either has provisioned policies stating the protocol to be used or learn which is the most appropriate one, and what parameters to use. The Connectivity Manager operations enable and disable interfaces and plugins depending on the operating system (Android or Linux) Application Programming Interfaces (APIs). For a normal HTTP transport the connection time can be learned in order to meet the traffic pattern of the application, such as sending multiple chunks of data at a specific hour and for another time only a few. For example, in case of air quality monitoring use case, at night the interest on pollution monitoring is decreased and the application might be instructed to aggregate the data and send it less often than during the day.

When an ASN subscribes to receive notifications related to a new content or updated information on front-end server or back-end server, it could specify a set of criteria to receive data in specific conditions only. OneM2M define further attributes to enable the ASN to control the rate of notifications to receive, however, no specification is provided on how to implement this. These attributes, the rateLimit and batchNotify, are used by the AdM2M to select proper connectivity channel that matches the capability of application nodes and the size of notification payload. The rateLimit attribute is used by the subscriber application to limit the notification rate while batchNotify is used to enable the batching of notifications. The HTTP protocol will be selected for sending batched notification of big sized. For small data packets of high rate CoAP will be selected. The thresholds of the rates and packet size will be continually updated by the Monitoring Model, based on the collected parameters.

In Figure 2, the requests flow of a common M2M scenario is depicted, which illustrates the functionality of the AdM2M modules. As a first step, registration requests are issued from the gateway towards the core server, and from the applications toward associated front-end/back-end server (step ①). The successful registration of M2M nodes with unique IDs is required prior to further interactions. Applications have the ability to create multiple Container resources that could be used later as a mediator for data buffering.

Based on ETSI M2M and oneM2M specifications, resources presenting applications and containers at the gateway/MN shall be announced to other SCLs (step ②). An announced resource shall point to the original resource and consists of only a limited set of attributes such as the link to the original resource and searchStrings (step ③). Providing this information over announced resource, facilities the discovery process specified by ETSI/OneM2M for searching of data and devices. Thus, the issuer of the discovery request does not have to contact all SCLs in order to find the resources.

The monitoring model of AdM2M measures the response time of each request issued by the gateway or core server as shown in (step ④). This measurement provides the M2M node with insights about the connection status and therefore used later in selecting proper protocol in line with the defined policy.

Registered application starts to push data, aggregated from attached sensors, by issuing a create ContentInstance resource request (step ⑤). The AdM2M monitoring model measures the size of data and the sampling rate of the create ContentInstance resource requests of each App, as shown in (step ⑥).

On the Network domain, applications interested in receiving notifications related to data/events from applications/devices located in the field domain

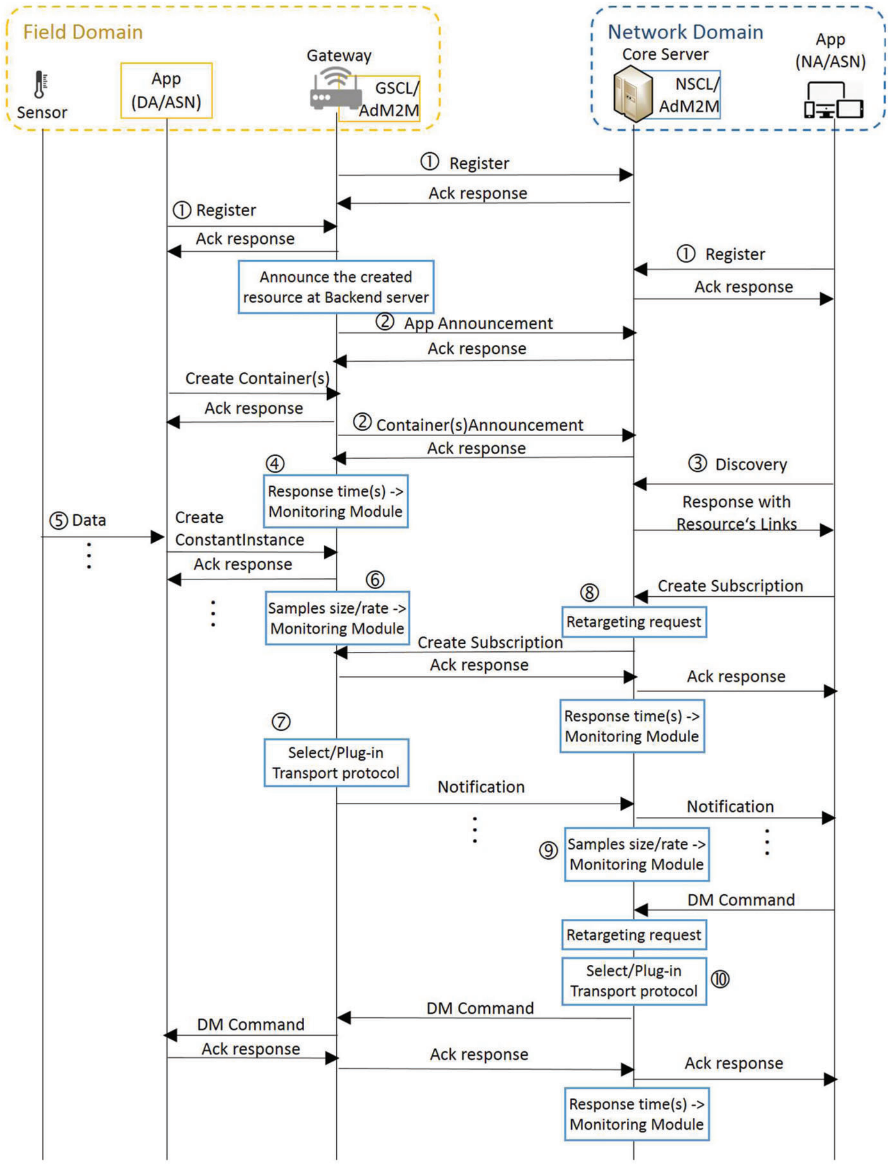

Figure 2 Requests flow of a common M2M scenario.

could send a request to create Subscription resource. As the target resource of the subscription request is not stored in the core server, the retargeting mechanism is used to forward the request to its destination. Whenever the notification process is triggered (i.e. the specified filter criteria are met), an evaluation process is executed to provide a decision on selecting which protocol to use in forwarding notifications (step ⑧).

Similarity, the monitoring module shall collect measurements from forwarded notifications (step ⑨), to upgrade the policy rules and used this information to plug-in transport protocol for further requests, such as sending a device management request to devices (step ⑩).

4 Performance Verification

As a first stage of verifying the performance of the implemented framework, protocols plug-ins and libraries, some experiments were executed using a M2M testbed. The experiments assist gaining a better understanding of the expected performance of M2M applications associated with a special traffic model when using any of these protocols in sending or retrieving data to an M2M front-end server. Apache JMeter [20] was used to carry out all evaluation scenarios, which is a pure Java application designed for load testing and performance measurement of different servers/protocols such as HTTP. Also, it could be extended to support other protocols, such as CoAP and MQTT through multiple pluggable samplers. Additional sampler plugins for CoAP, MQTT and AMQP were deployed to perform the performance evaluation during this research work.

Generally, the verification of the OpenMTC platform, presented in [21], has showed an acceptable level of reliability and scalability taking into account the hardware resource-capabilities and number of connected sensor nodes. Furthermore, the obtained performance results provide more insight on the importance of selecting a proper protocol stack with certain M2M use cases. In general, HTTP is an ideal transport protocol for requesting data from known sources over the Internet. However, the high demanding of bandwidth and computation resources limits its suitability for a number of M2M applications. Also, it does not provide scalable means for bi-direction communication such as sending notifications, and the textual encoding of HTTP headers obtains unnecessary overhead for parsing. Furthermore, CoAP is proposed as a lightweight client/server application protocol that uses the UDP transport protocol to avoid the overhead of connection oriented protocols. The advantage of the CoAP protocol was more noticeable on the resource-constrained gateway, as the response time of data pushing and retrieving requests is less affected than with HTTP protocol. A similar observation is obtained for the consumption level of hardware resources.

Consequently, HTTP is a good choice for forwarding data over a reliable connection to backend servers, while CoAP is better suited to exchange data and control commands between gateways and constrained-devices. For example, in the case of a Smart-Building gateway connected to multiple sensors nodes, with limited resources, and forwarding aggregated data/events to a backup server once a day. It is advisable to deploy a connectivity stack using CoAP to interact with the sensors while using an HTTP connectivity stack for forwarding the big payload of aggregated data.

Investigating the effect of the payload size in pushing data using both MQTT and AMQP protocols, which follow the Publish/Subscribe approach, shows that MQTT performs much better with big payload messages than both HTTP and CoAP. However, CoAP is more efficient in energy usage and produces less extra traffic in the case of small-size messages. Both MQTT and AMQP were able to handle payload messages bigger than 100KB. Table 3 shows the threshold limit of message rate with different payload sizes. It is observed that MQTT has a slightly higher message limit than AMQP. This is mainly related to the difference in frame header sizes of each protocol. The ordering of messages delivered to subscribers is guaranteed with both protocols, only when using the same QoS level with MQTT, and the same queue in AMQP. Both protocols provide efficient mechanisms for transferring one-to-many notifications, which makes them a good choice for forwarding notifications towards network applications.

Furthermore, we conducted more experiments to verify the concept of protocol selection. In the following, the Substation Automation application is considered as a use case to test the concept validation. The Substation Automation application is a Smart Energy services for monitoring, protection and control functions performed on Smart Grid substation and feeder equipment. The analysis of communication requirements and traffic loads for this application presented in [22] was assumed in our experiment.

Based on our observations and research work, it was concluded that CoAP is recommended to be used for pushing data flows at low rate frequency

Table 3 Threshold limits of message rate with MQTT and AMQP

Payload Size (Byte)	AMQP Message Rate (mps)	MQTT Message Rate (mps)
10	1022	1044
100	1020	1039
1 K	1002	1010
10 K	1001	1005
100 K	497	986
1 M	–	231

carrying small size payloads, while HTTP is recommended for data flows at high rate frequency with relative bigger payloads. In this experiment, we measured the end-to-end response time of two data flows: the first presents sampling data from emulated sensors, the second flow presents control signal flow from an OpenMTC front-end gateway. The data flows are pushed over the testbed using PC-Linux front-end OpenMTC with 2GB RAM. Each POST request carries a JSON formatted payload following the oneM2M specifications of data modeling. Figure 3 shows two plot diagrams for the

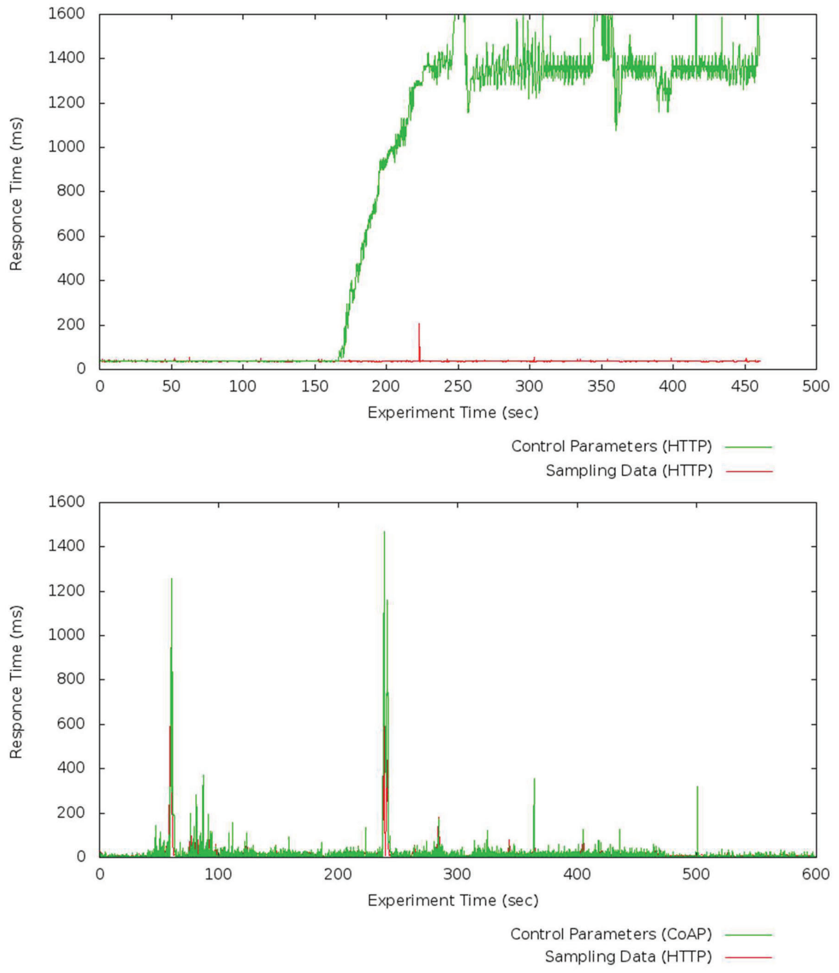

Figure 3 Performance of emulated substation automation application.

response time of data flows, that represent aggregated data: i) sampling data (rate 1 sample/second and payload size 1600Byte), ii) control parameters (rate: 100 request/s, payload size 200Byte). The substation automation application is a strictly delay-sensitive service. Therefore, it is required that the delay remains in the level of 200ms or less. The plots present the improvement on overall performing when using CoAP in transporting the control interactions instead of HTTP.

5 Conclusion

The M2M communication technology is representing an essential part of IoT concept, where machines communicate to each other with limited or no human interaction. This enhances the connectivity of Any-thing or Any-one, Any-time and Any-place via Any-path/network to use Any-service. The primary value that M2M creates is a direct result of the data that can be captured from connected things, and the resulting insights that drive business and operational transformation.

The characteristics of the M2M traffic are different from the existing Human-to-Human and Human-to-Machine traffic. Due to the heterogeneity of connecting objects in terms of capability and generated traffic, the M2M middleware needs to adapt its handling to the M2M traffic based on the application requirements and traffic pattern. In this paper, we proposed a novel concept aiming to handle the heterogeneous traffic from embedded devices/sensors in a Smart City environment.

The requirements and constraints of Smart City services raise new challenges to the traditional transport protocol layer. Several protocols have been proposed and standardized, each focus on a specific aspect of M2M/IoT communication. A sophisticated protocol able to handle all heterogeneous aspects related to this kind of communication and satisfying requirements of real-time streaming is still missing. Such protocol shall provide mechanisms to support self-configuration and self-adaptability to various network resources (e.g. wireless network systems) and various devices' capabilities. Additionally, it should be light enough to operate on resource-constrained devices. Nevertheless, a persistent connection might be essential for some cases. The approach proposed in this article was to enable M2M nodes to adjust the utilized communication stack by using pluggable protocols libraries, in order to overcome the lack of sufficient protocol to different M2M/IoT applications. The proposed framework aims to enhance the adaptability of M2M nodes, in order to accommodate the needs of M2M communications and applications.

References

[1] Rausand, M., and Høyland, A. (2004). *System Reliability Theory: Models, Statistical Methods, and Applications*. Hoboken, NJ: Wiley.

[2] Nikaein, N., et al. (2013). "Simple traffic modeling framework for machine type communication," in *Proceedings of the Tenth International Symposium on Wireless Communication Systems*, Ilmenau, 783–787.

[3] Elmangoush, A., Corici, A. A., Steinke, R., Corici, M., and Magedanz, T. (2015). A framework for handling heterogeneous m2m traffic. *Proc. Comput. Sci*. 63, 112–119.

[4] Shafiq, M. Z., Ji, L., Liu, A. X., Pang, J., and Wang, J. (2012). "A first look at cellular machine-to-machine traffic – large scale measurement and characterization," in *Proceedings of the 12th ACM SIGMETRICS/PERFORMANCE Joint International Conference on Measurement and Modeling of Computer Systems*, London, 65.

[5] Ahmed-Zaid, F., et al. (2011). *Vehicle Safety Communications – Applications (VSC-A)*. Final Report.

[6] Predojev, T., Dohler, M., Al-Hezmi, A., and Alonso-Zarate, J. (2014). "A real-time middleware platform for the smart grid," in *Proceedings of the IEEE Online GreenComm*, Singapore, 1–6.

[7] Castillejo, P., Martinez, J.-F., Rodriguez Molina, J., and Cuerva, A. (2013). "Integration of wearable devices in a wireless sensor network for an health application," in *Proceedings of the IEEE Wireless Communication*, Shanghai, 38–49.

[8] oneM2M-TS-0001 (2015). *OneM2M Functional Architecture*, Vol. 1.

[9] oneM2M-TS-0009 (2015). *HTTP Protocol Binding*, Vol. 1. 13.

[10] OneM2M-TS-0008 (2015). *CoAP Protocol Binding*, Vol. 1. 14.

[11] International Business Machines Corporation (IBM) (2010). *MQTT V3.1 Protocol Specification*. Paris: IBM.

[12] OneM2M-TS-0010 (2015). *MQTT Protocol Binding*, 1–27.

[13] Vinoski, S., and Technologies, I. (2006). Advanced message queuing protocol. *IEEE Internet Comput*. 6, 87–89.

[14] SIGFOX (2017). *SigFox*. Available at: http://www.sigfox.com/en/

[15] Tariq, M., Zhou, Z., Wu, J., Macuha, M., and Sato, T. (2012). "Smart grid standards for home and building automation," in *Proceedings of the 2012 IEEE International Conference on Power System Technology (POWERCON)*, Auckland, 1–6.

[16] Naito, K. (2017). A survey on the internet-of-things: standards, challenges and future prospects. *J. Inf. Process*. 25, 23–31.

[17] OpenMTC Platform (2017). Available at: http://www.open-mtc.org/index.html
[18] Elmangoush, A., Al-Hezmi, A., and Magedanz, T. (2014). "The Development of M2M Standards for Ubiquitous Sensing Service Layer," in *Proceedings of the 2014 IEEE Globecom Workshops (GC Wkshps)*, Austin, TX, 624–629.
[19] Kosolwsorrawattanakul, N., Elmangoush, A., Magedanz, T., and Aswakul, C. (2014). *Development of Real-Time Data Synchronization for IEEE1888 and ETSI M2M Standards*. IEICE Technical Report. Rome: IEEE, 79–84.
[20] Apache JMeter (2017). Available at: http://jmeter.apache.org/ [accessed October 12, 2015].
[21] Berrhouma, C., Elmangoush, A., Al-Hazmi, A., Steinke, R., and Agedanz, T. M. (2016). "Performance evaluation of an M2M platform in different deployment setups," in *Proceedings of the 2016 IEEE First International Conference on Internet-of-Things Design and Implementation (IoTDI)*, Berlin, 223–228.
[22] Khan, R. H., and Khan, J. Y. (2013). A comprehensive review of the application characteristics and traffic requirements of a smart grid communications network. *Comput. Netw.* 57, 825–845.

Biographies

Asma Elmangoush received her B.E. and M.Sc. in Computer Engineering from the College of Industrial Technology – Misurata, Libya, and received her Ph.D. degree from the Technische Universität Berlin, Germany, in 2016. She is currently a lecturer at the College of Industrial Technologies – Misurata, Libya. Her current research interests are in the area of the Internet of Things, Smart Services and network applications in the Smart grid power systems. She is an IEEE member.

Thomas Magedanz is a full professor in the electrical engineering and computer sciences faculty at the Technical University of Berlin, Germany, leading the chair for Next Generation Networks. In addition, he is director of the "Next Generation Network Infrastructures" division of the Fraunhofer Institute FOKUS. Since more than 20 years he is working in the convergence field of fixed and mobile telecommunications, the Internet and information technologies, which resulted in many international R&D projects centered on Next Generation Service Delivery Platforms prototyped in a set of globally recognized open technology testbeds. In 2007 he joined the European FIRE (Future Internet Research and Experimentation) Expert Group. In the course of his research activities he published more than 250 technical papers/articles. In addition, he is a senior member of the IEEE, and editorial board member of several journals. He received his diploma and his Ph.D. in computer sciences from the Technische Universität Berlin, Germany, in 1988 and 1993, respectively. In 2000 he finished his postdoctoral lecture qualification in Applied Computer Sciences at the Technische Universität Berlin, Germany.