# MQTT-like Network Management Architecture

O. Jukić[1], R. Filjar[2], I. Heđi[1], E.Ciriković[1]
[1]Virovitica University of Applied Sciences, ICT department, Virovitica, Croatia
{oliver.jukic | ivan.hedi | enes.cirikovic}@vuv.hr
[2]Krapina University of Applied Sciences, Krapina, Croatia; Faculty of Engineering, University of Rijeka, Rijeka, Croatia
renato.filjar@gmail.com

***Abstract* – Network management correlates data from different sources. It enables integrated view on telecommunication network and service status to network personnel. Typically, the same management data are used by number of network management platforms and tools. Unfortunately, these data are very often collected in parallel, from the same sources. It overloads network and network management systems.**
**In this paper we have proposed an MQTT-like architecture of network management system. MQTT is a protocol that uses a publish and subscribe model. It allows „publisher" to send messages belonging to specific topic to one or multiple „subscribers" that are subscribed to the topic. However, there is no direct connection between these entities. Central point of MQTT architecture is a „broker", acting as router between publishers and subscribers.**
**In architecture proposed, publishers are software agents collecting data from managed objects (network elements). Data are published to subscribed entities (e.g., network management applications). Every subscriber can process data received and publish processing results, acting as a publisher of data from higher abstraction level.**
**Main improvement of MQTT architecture is that broker can store published data for a certain amount of time. This promotes broker as a central point from which is possible to obtain management data. In the case of crisis (e.g., earthquake or flooding), proposed architecture allows external stakeholders to obtain data about network availability in real-time manner.**

***Keywords – MQTT, network management architecture, publish and subscribe.***

## I. INTRODUCTION

Telecommunication network is an infrastructure typically designed and implemented by telecom operators. However, main goal of telecom operators is not to set up network itself; rather, they are interested in using network as the infrastructure for service deployment while services are "goods" that operators offer to their customers. Based on that, terms "service management" and "service level agreement" became very important to telecom operators.

Services are dispersed through the whole network allocating different network resources [1]. It means that service availability depends on availability of different network resources as well as network architecture itself. Service management is partially based on network management data (Figure 1):

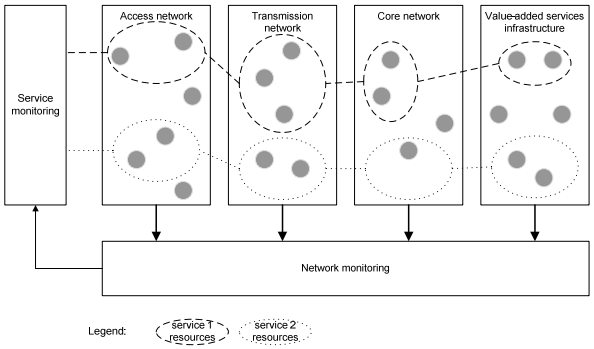


Figure 1. Service spreading over network resources.

Service level agreements (SLA) concluded between telecom operators and their subscribers should guarantee service quality level and minimum service availability time during predefined period (e.g., year).

Knowing about network problems that can degrade quality of service guaranteed to customers can preserve operator of service level agreement violation.

Network management data should be tied and correlated with data from different sources, such as inventory management and service architecture data, help-desk data (e.g., complaints from customers), end-to-end testing results, KPI threshold violations from performance management systems etc. [2] (Figure 2):

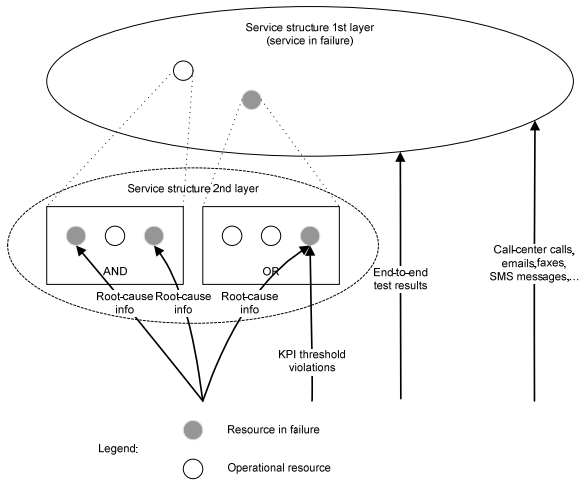


Figure 2. Service structure: general approach.

It leads us toward conclusion that the same network management data are going to be used by number of different service management applications since service management is positioned above network management in Telecommunication Management Network (TMN) pyramid [2],[3] (Figure 3).

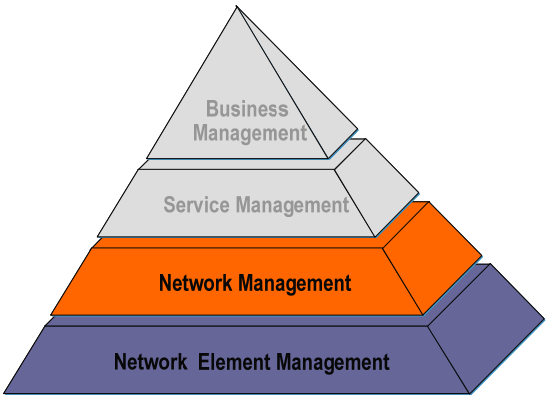


Figure 3. Service management in TMN pyramid.

For example, in UMTS (3G) network architecture radio network controller (RNC) controls number of NodeB nodes. Example of network management data for radio network controller is number of operational NodeB elements as well as state of every NodeB network element.

Since these network elements are part of UMTS Terrestrial Radio Access Network (UTRAN), its operational status will have an impact on different services provided by telecom operator.

Main idea presented in this paper is to propose network management architecture based on MQTT principles that will allow one-time "publishing" of network management data that can be used (immediately or later) by different service management applications.

Paper is structured as follows: first we will present short introduction together with related work. After that, we are going to present the network management architecture overview. For this purpose, MQTT protocol and its architecture will be briefly described. Finally, we have touched basic implementation aspects of architecture presented.

### A. *Related work*

MQTT is very known protocol widely used in Internet of Things (IoT) computing concept describing objects that can deliver information over Internet. Usage of MQTT protocol in IoT paradigm is described in many papers [8], [9], [10], [11]. For Internet connected devices there are many cloud providers offering connectivity between devices and the cloud. MQTT integration with cloud services is described in [5]. Amazon Web Services for Internet of Things (AWS IoT) can collect data from many different devices and connect them to endpoints for other tools, like network management solutions. Implementation of such system using MQTT is described in [4]. In this paper we have proposed an MQTT-like architecture of network management system. Similar approach can be realized using Constrained Application Protocol (CoAP). One of the main differences between MQTT and CoAP can be found in the transport layer of the OSI model. MQTT runs on top of the Transmission Control Protocol (TCP) while CoAP runs on top of the User Datagram Protocol (UDP). Although UDP is not reliable, CoAP provides its own reliability mechanism using „confirmable messages" and „non-confirmable messages" [11]. Many IoT protocols support single-topic messaging. This type of messaging implies that only one topic can be sent per message. In [12] an IoT message protocol that supports multi-topic messaging is proposed. Authors presented protocol with less delay and lower traffic for multi-topics compared to the MQTT. The general IoT architecture is based on real-time operating system (RTOS).

## II. MQTT Overview

### A. *MQTT*

MQTT is a Machine to Machine (M2M) Internet of Things lightweight connectivity protocol. Small code footprint makes it suitable to implement in small devices. Further, protocol fulfills requirements for low power consumption, low bandwidth consumption and low latency [4].

However, advantages mentioned above are not crucial for MQTT appearance in network management architecture proposed in this paper. The most important characteristic of MQTT in context of this architecture is the fact that it uses a publish and subscribe pattern for transmitting and receiving messages between its communication entities, such as devices and applications (e.g., gateways or servers).

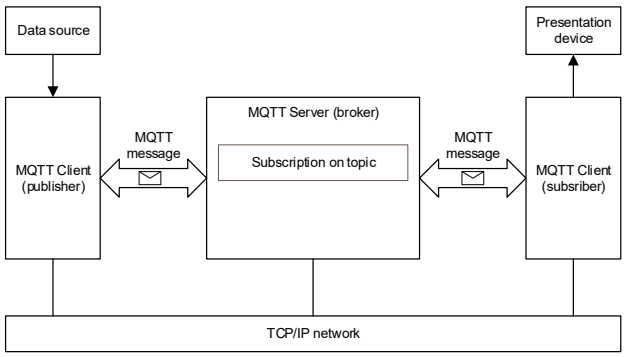


Figure 4. MQTT architecture.

An MQTT server called a „broker" is in the core of MQTT architecture. Devices connected to the broker are known as „clients". Broker acts as a router, handling data exchange between clients.

Data exchanged between clients are classified (based on its content) within one of possible classes, called „topics".

When client wants to distribute data, it publishes data classified within certain topic (client is „publisher"). Broker sends data received to all clients that have expressed interest to that topic. Interest is expressed by previous subscription to certain topic. All clients

receiving topic data are known as „subscribers“ to certain topic (Figure 4) [5]. MQTT client acting as publisher of some topic can acts as subscriber to another topic.

There are several important MQTT aspects that must be considered [6]:

1) Clients do not have addresses like in email systems, and messages are not sent to clients.
2) Messages are published to a broker on a topic.
3) The job of an MQTT broker is to filter messages based on topic, and then distribute them to subscribers.
4) A client can receive these messages by subscribing to that topic on the same broker.
5) There is no direct connection between a publisher and subscriber.
6) All clients can publish (broadcast) and subscribe (receive).
7) MQTT brokers do not normally store messages.

For both publishers and subscribers, it is not necessary to have fixed network addresses; rather, both must know broker address. Broker will take care about specific message to be delivered to all sides interested in specific topic. When publisher publishes message, its job is done.

Further, it is not necessary to have direct connection between communicating entities. In the case of complex network with high security level, number of firewalls and restrictions, it can be very important feature.

Client can receive data as subscriber, but also to send data as publisher at the same time. It allows client to act as a data processor. One client (subscriber and publisher at the same time) can receive data from lower abstraction level and publish processing result to be available to other clients (subscribers) on higher abstraction level.

### B. *MQTT topics*

MQTT topic is very important part of MQTT architecture. Topic allows clients to address specific data that are of importance to client (it allows data filtration). Rather than knowing data source, client must know data specification – which data are important to the client. MQTT topics are a form of addressing that allows MQTT clients to share information.

Topic is text, structured in hierarchical manner, like folders and files on hard-disc. Forward slash (/) is used as a delimiter of textual topic parts.

This mechanism allows creation of user friendly and self-descriptive naming structures. For instance:

```
/students/1st year/Ivan Horvat
```

When referring to the topic, client can use wild cards, for one or multi-level replacement. For instance:

```
/students/+/Ivan Horvat
```

has meaning “all students named Ivan Horvat from any year” (one level replacement). On the other hand:

```
/students/#
```

has meaning “all students” (multi-level replacement).

### C. *MQTT architecture and network management*

As mentioned above, there are number of network elements which are sources of network management data used in service management process. Number of service management applications very often collect network management data simultaneously, doing redundant job and overloading network resources.

The main idea how to use MQTT benefits in network and service management process is to establish stable broker application. Collectors of network management data as well as processors of data collected can act as both subscribers and publishers. Management presentation tools can approach any kind of management data by subscribing to specific topic.

Management data can be described by topic structure, very similar as object identificators in Simple Network Management Protocol (SNMP) protocol [7].

## III. MQTT NETWORK MANAGEMENT ARCHITECTURE

Proposed MQTT-like network management architecture, together with managed network part is shown on figure 5.

On the bottom, there is part of radio access network, covering 2G and 3G mobile network. These parts are well structured. Every network technology has its own controller (BSC – Base Station Controller for 2G and RNC – Radio Network Controller for 3G network). Every controller has number of underlying network elements (BTS – Base Transceiver Station for 2G or NodeB for 3G).

Management data are collected by network management data collecting modules (1). These modules act on “north” side as publishers publishing network management data to clients (2). Any client can be subscribed to specific topic (3) and, after data processing, can “return” processed data as publisher (4). Processed data can be used by client applications, acting as subscribers (5). Data processing can be done in number of iterative and parallel steps, which means steps 3 and 4 can be repeated number of times, even simultaneously.

Normally, MQTT brokers do not store messages. However, messages in our architecture carry network management data. All published data must have been stored at broker side for certain amount of time. That amount is not fixed and will be related to data topic. For instance, network configuration data reliable even days after has been set while alarming data becomes unreliable after couple of minutes. Parameter Time To Live (TTL) is introduced to cope with data reliability issue (see section “Implementation aspects”). It allows defining of time frame within data are reliable. Unreliable data are discarded.

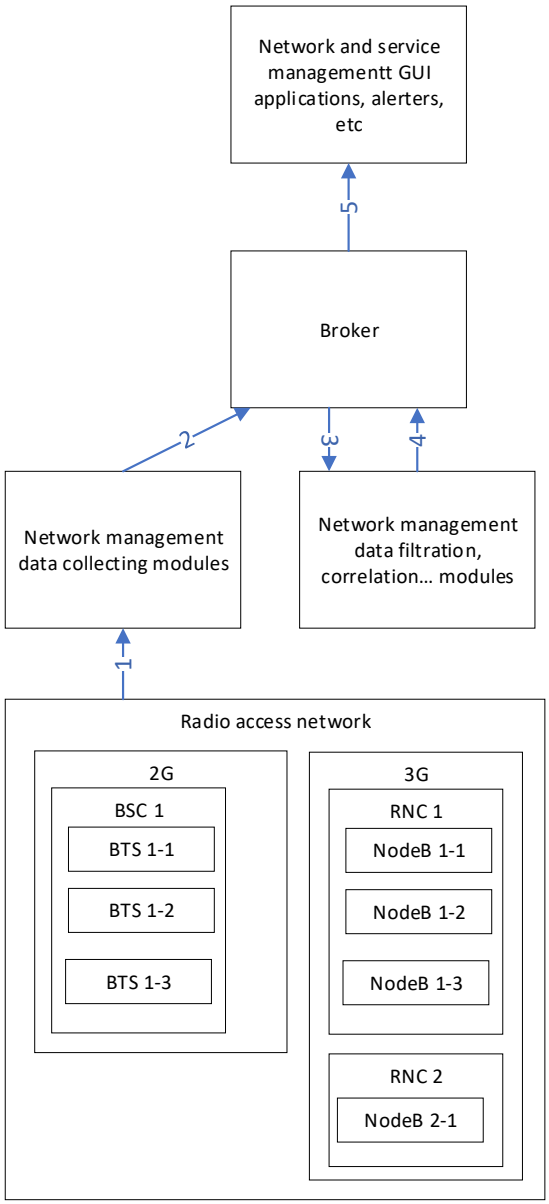


Figure 5. MQTT-like network management architecture.

### A. *Network management topics*

Since network has hierarchical structure, it is suitable to be described by topics. For instance, part of radio access network from the figure 5 can be described as follows.

*/RAN*

presents radio access network itself.

*/RAN/2G*

*/RAN/3G*

presents 2G and 3G network.

*/RAN/3G/RNC/RNC1*

presents radio network controller RNC1. Here we can define number of attributes belonging to the RNC1. For instance:

*/RAN/3G/RNC/RNC1/GPS location*

*/RAN/3G/RNC/RNC1/Number of NodeBs*

*/RAN/3G/RNC/RNC1/Number of Operational NodeBs*

*/RAN/3G/RNC/RNC1/Number of Halted NodeBs*

Further, NodeBs can be shown as own topics:

*/RAN/3G/RNC/RNC1/NodeB 1-1*

together with appropriate attributes:

*/RAN/3G/RNC/RNC1/NodeB 1-1/Operational state=OPER*

*/RAN/3G/RNC/RNC1/NodeB 1-1/Admin state=UNLOCKED*

At the lowest level, network management data collection modules collect data about operational and administrative state of NodeBs. After data are published, subscribed correlation module will count total number and number of operational NodeBs for specific RNC. These aggregated data will be published and available later for any interested modules.

For managed network, “dictionary” of all possible management topics must be created and must be available to all clients. It defines interface from subscribers to network and service management data. Subscribers must know broker address only as well as description of topic in which they are interested.

### B. *Implementation aspects*

Topic structure as described in previous section integrates network inventory data with real-time network management data. For instance, total number of NodeBs for one RNC is inventory data and is rarely changed. On the other hand, number of operational NodeBs is dynamically changed, based on current network status. Sudden storm can drastically decrease this attribute in a couple of minutes.

One client may wish to collect inventory data at startup and then to continue following new messages related to specific topic. Hence, it is necessary to extend MQTT protocol introducing message buffering. This will allow clients to collect even previously published data.

However, it leads us to the question: “how reliable my data are?”. In order to ensure data reliability, we have introduced Time To Live (TTL) attribute to every topic.

For instance, inventory data can be refreshed once per day, with TTL=24 hours. Dynamic data, such as number of operational NodeBs must be collected and recalculated on minute basis. Hence TTL should be measured in

minutes. All published and stored data will be discarded after TTL has expired.

### C. *Existing management tool comparison*

There are number of network management tools collecting and offering network management data to interested parts. These data are usually accessible through Application Programming Interface (API). There are number of protocols that can be used for API implementation such as SNMP or SOAP. However, if we want to collect data from different network parts via SNMP protocol, we must be familiar with network management data structure for every collecting process.

Main idea of proposed architecture is top serve as a central network management data warehouse, where interested parts can find data needed using simple communication protocol and even simpler network management structure description.

Proposed network management architecture should act as an "umbrella" system, covering different network management data types. User of that system is focused on data type needed (described by human readable "topic" in this architecture), rather than technical aspects of data collection.

## IV. NETWORK MANAGEMENT ARCHITECTURE IMPLEMENTATION

Proposed architecture is implemented in javascript programming language with NodeJS and Mosca library. It is running in NodeJS Environment.

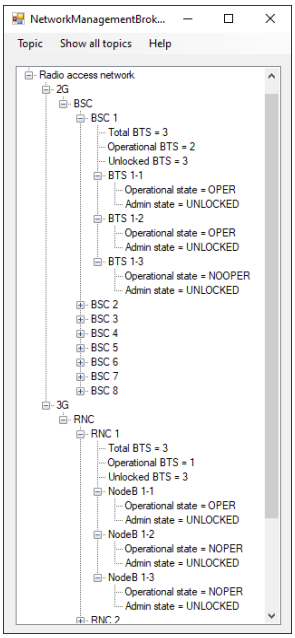


Figure 6. Network management broker – topic presentation.

This is standard architecture since MQTT is part of MOSCA. However, we have extended "regular" architecture with both MQTT extensions we mentioned above: message buffering and TTL. Topic dictionary is created and is available for editing through graphical user interface (Figure 6). It is mandatory part of broker application in this network management architecture since topic dictionary serves as description of interface from subscribers to published network management data.

For publishing and retrieving messages, simple API is developed. Using this API 3rd party applications can be very simple integrated in network management architecture proposed.

## V. CONCLUSION

In this paper, we have presented an MQTT-like architecture of network management system. Using MQTT terminology, "publishers" are applications collecting data from network elements. Data are published to "subscribers", subscribed entities that can process data received and re-publish processing results. All these transactions are handled by "broker".

Main improvement of MQTT architecture is that broker can store published data for a certain amount of time. This promotes broker as a central point from which is possible to obtain management data. To make previously publish data reliable, we have introduced TTL parameter connected to every specific topic. Proof of concept implementation is done describing part of Radio Access Network.

## REFERENCES


[1] O. Jukić and M. Kunštić, "Integrated view on telecommunication network status", Proceedings of the 34th International Convention MIPRO 2011, pp. 141-145, MIPRO, Opatija, 2011.

[2] O. Jukić and I. Heđi, "Service monitoring and alarm correlations", 4th International congress on ultra-modern telecommunications and control systems, pp. 330-334, Saint Petersburg, 2012.

[3] Udupa, K.D., TMN – Telecommunications Management Network, McGraw-Hill Telecommunications, New York, 1999.

[4] O. Jukić, I. Heđi and E. Ciriković, "IoT cloud-based services in network management solutions", Proceedings of the 43rd International Convention MIPRO 2020, pp. 447-452, MIPRO, Opatija, 2020.

[5] O. Jukić, I. Špeh and I. Heđi, "Cloud-based services for the Internet of Things", Proceedings of the 41st International Convention MIPRO 2018, pp. 407-412, MIPRO, Opatija, 2018.

[6] http://www.steves-internet-guide.com/mqtt-works/ visited on February 20th, 2021.

[7] William Stallings: "SNMP, SNMPv2 and CMIP – the practical guide to network-management standards", Addison-Wesley Publishing Company, Reading, Massachusetts.

[8] V. Karagiannis, P. Chatzimisios, F. Vazquez-Gallego, J. AlonsoZarate „A Survey on Application Layer Protocols for the Internet of Things", Transaction on IoT and Cloud Computing 2015.

[9] S. Katsikeas, „A lightweight and secure MQTT implementation for Wireless Sensor Nodes", Technical University of Crete, June 2016.

[10] V.Lampkin, W. T. Leong, L. Olivera, S. Rawat, N. Subrahamanyam, R. Xiang, "Building Smarter Planet Solutions with MQTT and IBM WebSphere MQ Telemetry", First Edition, September 2012.

[11] I. Heđi, I. Špeh and A. Šarabok, "IoT network protocols comparison for the purpose of IoT constrained networks", Proceedings of the 40th International Convention MIPRO 2017, pp. 628-632, MIPRO, Opatija, 2017.

[12] M. Hussein, A. I. Galal, E. Abd-Elrahman and M. Zorkany, "Internet of Things (IoT) Platform for Multi-Topic Messaging", Energies, Volume 13, 2020.