Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 130 (2018) 619–627

Procedia Computer Science

www.elsevier.com/locate/procedia

9th International Conference on Ambient Systems, Networks and Technologies, ANT-2018 and the 8th International Conference on Sustainable Energy Information Technology, SEIT 2018, 8-11 May, 2018, Porto, Portugal

Enhancing Middleware-based IoT Applications through Run-Time Pluggable QoS Management Mechanisms. Application to a oneM2M compliant IoT Middleware

Clovis Anicet Ouedraogo[a], Samir Medjiah[a,b], Christophe Chassot[a,c,*], Khalil Drira[a]

[a] CNRS, LAAS, 7 avenue du Colonel Roche, F-31400 Toulouse, France
Univ. Toulouse, [b] UPS, [c] INSA, LAAS, F-31400 Toulouse, France
{ouedraogo, medjiah, chassot, drira} @ laas.fr

Abstract

In the recent years, telecom and computer networks have witnessed new concepts and technologies through Network Function Virtualization (NFV) and Software-Defined Networking (SDN). SDN, which allows applications to have a control over the network, and NFV, which allows deploying network functions in virtualized environments, are two paradigms that are increasingly used for the Internet of Things (IoT). This Internet (IoT) brings the promise to interconnect billions of devices in the next few years rises several scientific challenges in particular those of the satisfaction of the quality of service (QoS) required by the IoT applications. In order to address this problem, we have identified two bottlenecks with respect to the QoS: the traversed networks and the intermediate entities that allows the application to interact with the IoT devices. In this paper, we first present an innovative vision of a "network function" with respect to their deployment and runtime environment. Then, we describe our general approach of a solution that consists in the dynamic, autonomous, and seamless deployment of QoS management mechanisms. We also describe the requirements for the implementation of such approach. Finally, we present a redirection mechanism, implemented as a network function, allowing the seamless control of the data path of a given middleware traffic. This mechanism is assessed through a use case related to vehicular transportation.

Keywords: Internet of Things; QoS; Middleware; Modular Framework; Dynamic Deployment; Network Function; Autonomic Computing.

1. Introduction

The Internet of Things, its application and their QoS requirements. The future Internet will include not only usual terminals but more generally any form of connected *objects* (or *things*) authorizing the development of new business

[*] Corresponding author. Tel.: +33-561-337-816; fax: +33-561-559-500.
E-mail address: chassot@laas.fr

1877-0509 © 2018 The Authors. Published by Elsevier B.V.
Peer-review under responsibility of the Conference Program Chairs.
10.1016/j.procs.2018.04.112

activities, in various domains such as remote supervision, personal assistance, or urban transport. This IoT will also have to meet non-functional needs (e.g. quality of service - QoS, security) of these new applications.

The interactions between the underlying application software(s) and the connected objects will be based on heterogeneous networks and on *middleware* layers. Indeed, from 2010, a major standardization effort has been conducted, notably via the ETSI and the oneM2M consortium[1, 2]. The resulting frameworks are aimed at abstracting applications from complexity of the underlying technologies (networks and objects); they are also aimed at avoiding *vertical* fragmentation of currently developed IoT solutions thanks to a generic middleware layer. Based on the REST architectural style (http, CoAP, …), this framework makes them appear what can be called *Middleware (MW) nodes*, named *gateway* and *server* in the ETSI vision, and *mn-cse* and *in-cse* in the oneM2M vision (Fig. 1). Both visions make also appear two major bottlenecks with regard to QoS considerations (service availability, bounded response time, etc.): within the connected objects and IP networks, and within the MW nodes.

In this context, several attempts have already been done to face (among other) the QoS requirements at the middleware level. Those attempts are based, for instance, on the deployment of QoS-oriented mechanisms within the MW nodes, with the aim to manage the application traffic (e.g. delaying less priority http requests in case of congestion) and / or allocated resources of the underlying machines[3]. Those propositions have also shown the benefits that can be induced by a deployment of such mechanisms “outside” of the data path, for instance thanks to intermediate proxies configured (for instance) as traffic load balancer / shaper / dropper (Fig. 2).

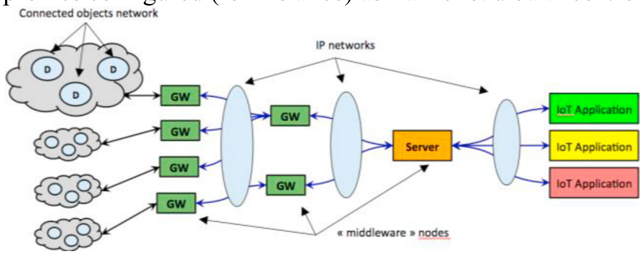

Fig. 1: Overview of an IoT Middleware

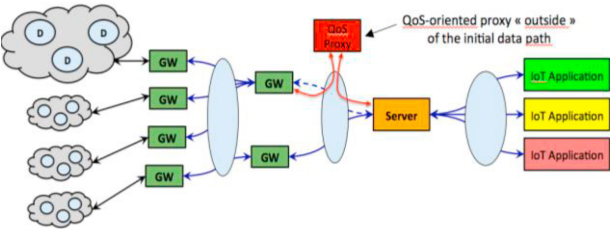

Fig. 2: QoS-oriented middleware with proxy outside of the data path

Network function virtualization. Since a few years, *Cloud* and now *Fog Computing* constitute opportune environments to help meeting IoT applications’ functional needs. More generally, the advent of *virtualization* technologies makes it now possible the deployment of (e.g. QoS-oriented) mechanisms on dedicated equipment but also on private or public *data centers* having hypervisors offering the required functional capabilities. The concept of *virtual network function* (VNF) has been defined by the ETSI as part of its work on standardization of the NFV technology[4], the term “virtual” meaning that a NF is not necessarily implemented on a dedicated equipment. This concept is today to be considered in a wider IT environment involving any node that can host and execute the corresponding program, whether it has a hypervisor or not (i.e. serverless paradigm[5]). For instance, it is possible to deploy and launch an *executable applicative program* on a simple laptop without interrupting the execution of its operating system. It is also possible to deploy an *application module* and to integrate it dynamically within an application code whose design is based on *components* (or *micro services)-oriented approach*[6].

This analysis allows (re)-defining the concept of network function (NF) (Fig. 3), which basically consists in a given processing of packets at any level of the communication stack (Application, Middleware, etc.).

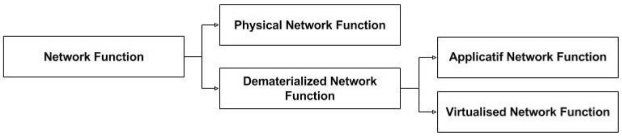
Network Function
Physical Network Function
Dematerialized Network Function
Applicatif Network Function
Virtualised Network Function

Fig. 3: Network function (NF) concept

In our vision, the concept of NF integrates and extends the concept of ETSI VNF which appears as a special case of what we call a *dematerialized network function* (DNF), i.e. deployed outside of its original environment (as opposed