Automated microservices deployment and dynamic traffic forwarding through 5G networks

Daniele Rossi[1], Giacomo Tontini[1], Mauro Sgarzi[3], Claudio Guidi[2]
Antonella Bellettini[3], and Franco Callegati[1]

[1] CIRI-ICT, Alma Mater Studiorum - Università di Bologna, Bologna, ITALY
name.surname@unibo.it
[2] italianaSoftware, Imola, ITALY cguidi@italianasoftware.com
[3] Imolainformatica, Imola, ITALY
msgarzi@imolainformatica.it, abellettini@imolainformatica.it

Abstract

In this manuscript we demonstrate the integration of 5G connectivity with applications exploiting distributed microservice architectures.

We have implemented network digital entities that are used to abstract network capabilities and offer to the microservice orchestration platform a standardized and uniform way to talk with the 5G-core network system.

1 Introduction

In this manuscript we report the proof of concept implementation of an integration between microservices applications and the 5G network. The goal is to show that a microservice orchestration platform may seamlessly interact with the 5G network and exploit it to create on demand and dedicated communications between some of the microservices composing the whole application.

To achieve this goal we exploit some concepts that have been defined in the framework of the Industry 4.0 (I4.0) paradigm. The *Asset Administration Shell* (AAS) paradigm, as proposed in [1] is the key concept to allow interoperability of building blocks of different nature or produced by different manufacturer into an Industry 4.0 environment.

5G may be considered one of this assets that can provide on demands and configurable connectivity between end point located either locally or very far away, thanks to its characteristic of being a worldwide adopted standard. Moreover it offers new features such as the support of low latency, to the benefit of time-sensitive applications. In this scenario the 5G Alliance for Connected Industries and Automation, 5G-ACIA, [4] is working to promote the application of the 5G technology to the industrial domain. In [2] is discussed that also 5G components should be equipped with an AAS that allows their full integration in the factory environment.

The goal of this demo is to show a concrete implementation of this concept, integrated into a distributed microservices application.

The manuscript is organized as follows. In section 2 the microservice orchestration platform used in this experiment is briefly described and reviewed. In Section 3 the principle of operation of the 5G AAS is explained and in Section 4 the results of the integration between the two is presented. Finally in Section 5 some conclusions are drawn.

2 Microservices orchestration architecture

Microservices are usually exploited as an architectural approach for addressing the development of applications in a distributed and cloud native manner. But they can be exploited also for dealing with system integration as shown in [3] thus offering the possibility to run process coordinators as independent components. Here, following the idea of the composable enterprise [5], we consider a microservice ecosystem where all the applications and their integration components are developed exploiting a microservice approach. In other words we consider a scenario where *everything is a microservice*. In particular, the orchestration of the microservices ecosystem considered in this work is performed using Jung [3] that is already used in productive scenarios. Jung transparently manages different microservices deployed in different environments, thus allowing to focus on the functional and not functional characteristics of each component by abstracting away from the underlying infrastructure. In this way, Jung offers to the user a uniform view of all the used microservices.

Jung is a platform written in Jolie [6] that is a microservice-oriented language, designed to approach microservices development by crystallizing specific syntactic constructs which enable the developer to direclty *think in (micro)services*. Some characteristics of Jolie are: synchronous and asynchronous communication, workflow-like behavior, protocol agnostic, integrated syntax for interfaces.

As shown in Figure 1 the microservices could be deployed on various physical platforms (K8S, Docker etc.) but are exposed to a user as a uniform set. This is achieved with *cloud nodes*, that are processes instantiated into the physical platform that expose to Jung the primitives to manage the microservices according to the specific rules of the platform. Each cloud-node manages the lifecycle of a set of microservices: upload, deploy, configuration, running, check the if it is alive. Moreover offers to the microservices some additional functionalities such as for instance retrieving registered input endpoint, etc. Jung exposes to the user a control-panel node which manages and interacts with one or more cloud-nodes.

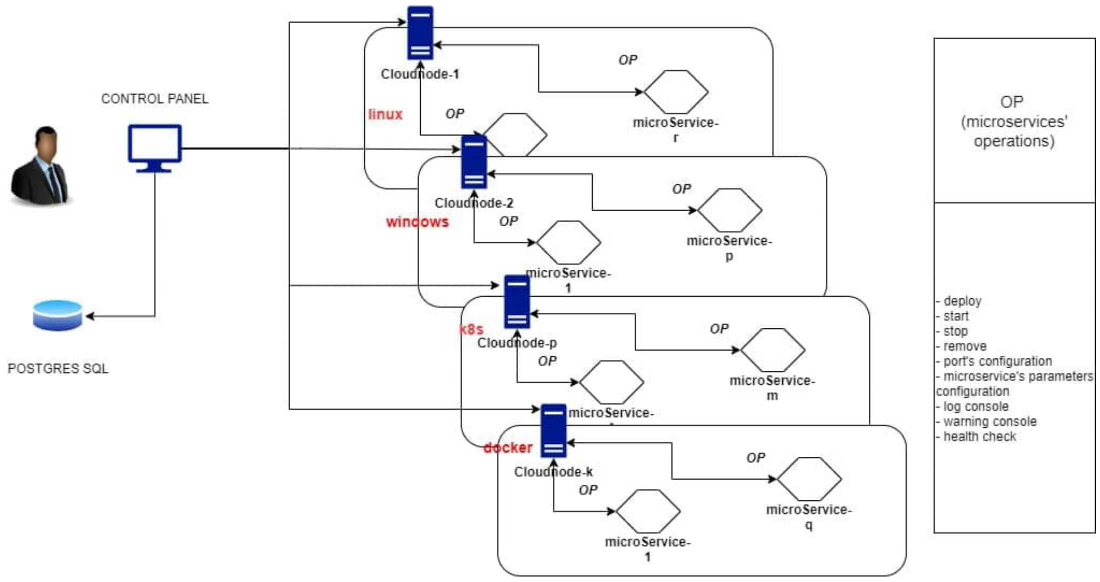

Figure 1: Logical architecture of JUNG

3 Control and management of 5G connectivity

In 5G the communication between end users is implemented as bit pipe mapped over a PDU session. The PDU session is the logical bit pipe providing the connectivity to an external data network. The PDU session spans from the radio access network (RAN) to the gateway towards the external networks (called User Plane Function or UPF). At the radio access it includes one or more Data Radio Bearer (DRB) to the User Equipment (the mobile terminal) and at the IP level it is composed by a GTP-U (GPRS Tunneling Protocol - User) tunnel between the base station (also called gNodeB or gNB) and the UPF.

3.1 5G QoS model

A PDU session includes one or more QoS Flow that is the lowest granularity of a traffic flow where QoS and charging can be applied. Each QoS flow is associated by the Service Data Adaption Protocol (SDAP) layer with a DRB in the RAN. Each QoS flow is identified by a (QoS Flow Identifier) QFI within a PDU session and could be a non-Guaranteed Bit Rate (Non-GBR) or a Guaranteed Bit Rate (GBR) QoS flow.

3.2 The digital entity for PDU management

Following the 5G-ACIA proposal, we decided to model the PDU session management as a set of abstractions that can be used to manage the 5G connectivity in an agnostic way, with the goal to control QoS in a capillary way. This is achieved by defining a set of digital entities with related interfaces. In order to manage the QoS, through the creation and elimination of PDU sessions, we need the following entities:

- **5G_digital_entity**: corresponds to the digital representation of the 5G network and exposes the status of the network core components and the list of slices for which each User Equipment, UE, can activate a PDU session.
- **Physical_ue_proxy**: offers a set of standard APIs regardless of the underlying UE, this guarantees modularity and flexibility of the system. Obviously, this entity is strictly dependent on the UE; in this case, the implementation was carried out for UERANSIM.
- **Ue_digital_entity**: corresponds to the digital representation of a user equipment and displays the status of the latter (e.g whether or not it is connected to a gNB, to which gNB it is connected, which PDU sessions are active and those which can be activated, etc). It also allows you to activate or release one or more PDU sessions.
- **Node_discovery**: has the task of maintaining the network location of active UEs. When a Ue_digital_entity is executed, it registers itself with the node_discovery and periodically sends it a keep alive message notifying its presence; if node_discovery does not receive a keep alive from a UE for more than 'S' seconds, the latter will be considered inactive and removed from the list of connected ones.
- **Dashboard**: interacts with the entities listed above to view the available digital entities, their status and perform actions on them. In particular, there is a section that allows you to monitor the services exposed by the core components of the 5G network and a section that allows you to view which UEs are available in the system. For each UE, interacting with the related digital entity, it is possible to view which PDU sessions are active and possibly establish new ones or release the existing ones.

The deployed architecture is shown in Figure 2. Here the User Equipment and the gNodeB are emulated with the UERANSIM software [7]. Similarly the external network is simulated with an echo server outside the 5G infrastructure. The deployment has been done in such a way that the UE is able to communicate with the Echo server both via a direct connection (*direct_net* in blue) or via a 5G PDU Session previously established (in red).

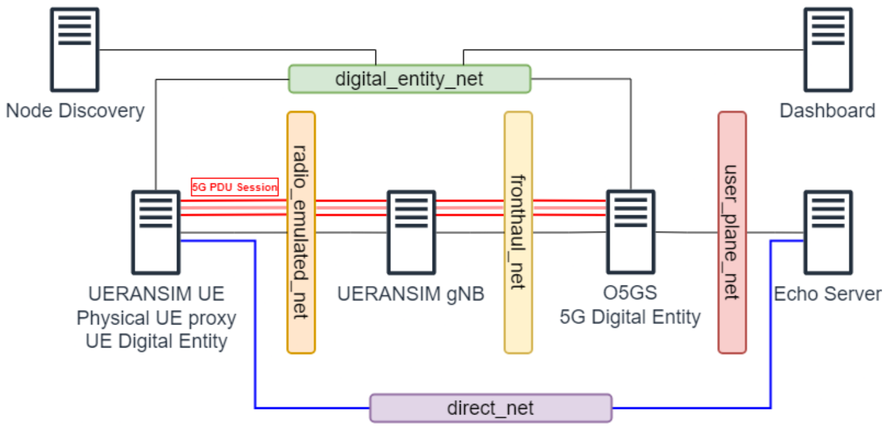

Figure 2: Deployed architecture of the digital entities system

4 Integration between microservices and 5G

The use case presented in this work shows an integration of the Jung microservices management platform with the 5G digital entity providing abstractions to manage PDU sessions. It is structured as follows: a microservice, named **sayMultiport**, is automatically deployed in a cloud node living in the machine UERANSIM UE, which has to create the 5G PDU Session. The sayMultiport microservice has to communicate with two instances of the same microservice, ms2 Echo and ms3 Echo, deployed in the cloud node hosted on the machine Echo Server; these service will return the string received in input [3].

sayMultiport tries to communicate with an instance of Echo with a 5G PDU session and with the other instance using a normal connection. The goal is to show the feasibility of communicating with two different connections. From the programmer's point of view, the only difference is the syntax used to specify the endpoint location: *socket://x1.x2.x3.x4:port* for normal connection and *5G:socket://x1.x2.x3.x4:port* for 5G PDU Session.

To achieve this goal these steps had to be completed:

1. Create a service PDUSession5GService, which manages the asynchronous UEDigitalEntity API REST, exposing a synchronous API to other microservices. It manages a pool of the opened 5G PDU Sessions. Thanks to Jolie's construct this service is embedded inside JUNG, hiding its operation from the user.

2. Generate dynamic service 5GForwarder for each service requiring a 5G connection. It is a proxy for the remote microservice, for this reason it has the same input interface of the latter and is also connected to PDUSession5GService in order to request the establishment of a PDU Session when necessary.

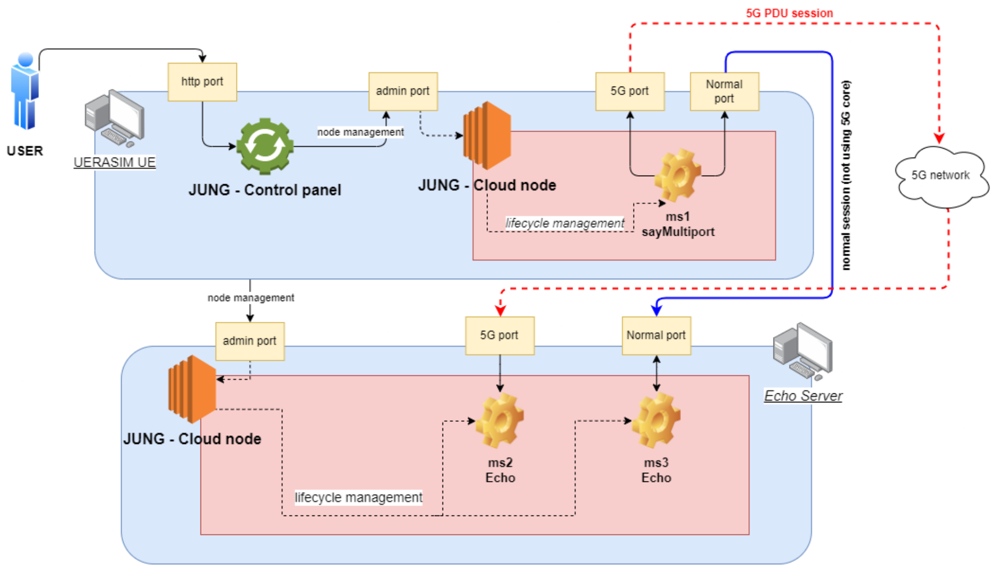

Figure 3: Deployed architecture of three microservices through JUNG, which communicates both on a traditional HTTP session or a 5G PDU session. Fig 2 has to be considered for the 5G network deployment.

3. Automatically connect sayMultiport to 5GForwarder specifying a local port in the output-port configuration. Local port is then linked with the input port of 5GForwarder.

In Figure 4 is shown the result of a working example. It is an example which is not related to a specific operational use case, but simply chosen to show the system dynamics. The sayMultiport sends a bit stream to the echo server using `iperf3`. It starts by using the direct wired connection with `iperf` sending 25 Mbit/s (red line). After 30 seconds it requests a 5G connection and a PDU session is established at 5Mbit/s (green line). After another 30 seconds it requests to change the PDU session at the same bandwidth as the original wired connection (blue line).

The automation introduced with tis approach is based on the fact that the lifecycle of 5GForwarder is completely managed by JUNG: at the time sayMultiport starts, 5GForwarder is created and attached. When sayMultiport stops, 5GForwarder will be destroyed. During its lifetime 5GForwarder provides PDU sessions with the necessary quality of service by interacting with the 5G Digital entity.

5 Conclusions

In this manuscript we presented a distributed microservice application management which exploits 5G for communication. We have shown that the service management platform can be integrated with a 5G digital entity that was built on purpose and provides the abstractions to manage the connectivity and the related QoS.

The result is integrated full automation of the application deployment both for the microservice components and the related interconnections.

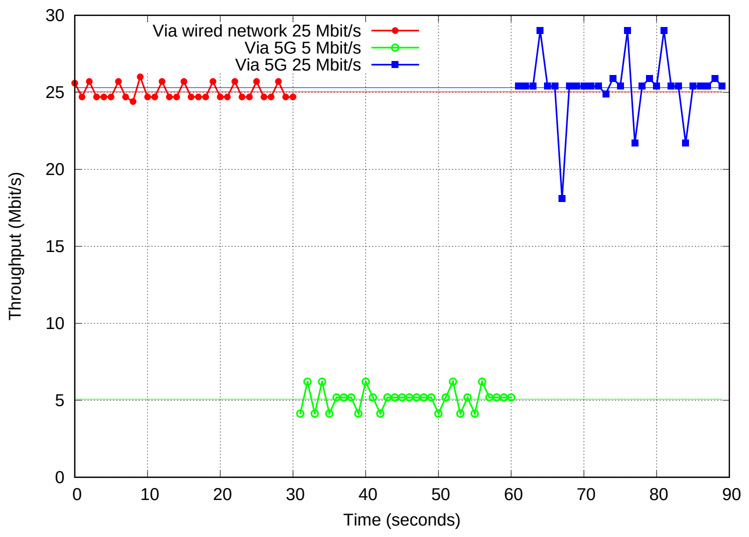

Figure 4: Differentiated QoS levels achieved on 5G network side compared to a traditional wired one without QoS enabled

6 Acknowledgments

This work was partially funded by the Italian Minister for Industrial Development under project 5GConnect.

References

[1] Plattform Industrie 4.0. Asset administration shell reading guide. https://www.plattform-i40.de/IP/Redaktion/EN/Downloads/Publikation/Asset_Administration_Shell_Reading_Guide.pdf?__blob=publicationFile&v=7, 2021.

[2] 5G Alliance for Connected Industries and Automation. Using digital twins to integrate 5g into production networks. https://5g-acia.org/whitepapers/using-digital-twins-to-integrate-5g-into-production-networks, 2021.

[3] Claudio Guidi and Balint Maschio. A jolie based platform for speeding-up the digitalization of system integration processes. Microservices International Conference 2019, https://www.conf-micro.services/2019/papers/Microservices_2019_paper_6.pdf, 2019.

[4] https://5g-acia.org/.

[5] The future of business is composable. Gartner Keynotehttps://www.gartner.com/smarterwithgartner/gartner-keynote-the-future-of-business-is-composable.

[6] www.jolie-lang.org.

[7] https://github.com/aligungr/UERANSIM.