An Open-Source Tool Modeling the ETSI-MEC Architecture in the Industry 4.0 Context

S. Massari, N. Mirizzi, G. Piro, and G. Boggia
DEI, Politecnico di Bari, v. Orabona 4, 70125, Bari, Italy
CNIT, Consorzio Nazionale Interuniversitario per le Telecomunicazioni, Italy
Email: {simona.massari, nicholas.mirizzi, giuseppe.piro, gennaro.boggia}@poliba.it

Abstract— **The transition from traditional to the smart factory, promoted by Industry 4.0, was driven by the introduction of the 5-th generation of mobile systems (5G) and its related key enabling technologies. In this context, Multi-access Edge Computing (MEC) emerged as a promising approach to offload computational tasks from resource-limited Industrial Internet of Thing devices to high-performance edge servers, which ensures higher service reliability and lower communication latency. While an Industry Specification Group of ETSI, namely ETSI-MEC, is sketching a MEC-oriented architecture and its related functionalities, there is an emerging scientific interest to evaluate novel solutions in a standard-compliant framework. To this end, this paper presents an extension of the network simulator ns-3, implementing entities (e.g., base stations, MEC Hosts, etc.), interactions, and functionalities (e.g., orchestration of MEC resources, distribution of tasks, etc.) standardized in the context of ETSI-MEC, while supporting multi-cell scenarios with mobile users and different user load generator models. The proposed tool has been used to evaluate the performance of a MEC-enabled industrial environment, where Unmanned Aircraft System, Automated Guided Vehicle, Industrial Internet of Things, and mixed reality devices jointly exploit computational capabilities deployed at the network edge. This study demonstrates that the proposed tool is a valid instrument to make safer and faster the assembly, maintenance, and testing phases in the upcoming smart factory realities.**

I. Introduction

In recent years, the rapid technological development led to a renewal of production and management systems within industries, fully or partially automating most of the stages of the production process. The term Industry 4.0 emerged to indicate the tendency of industrial automation to integrate new technologies within the industrial scenario to improve working conditions, create new business models and increase productivity [1]. The resulting benefits include: (1) a shorter time-to-market for new products, (2) an improvement customer responsiveness, (3) enabling custom mass production without significantly increased overall production costs, (4) greater flexibility, and (5) more efficient use of resources and energy [2]. The fourth industrial revolution, compared to the previous ones, sees as the protagonist the use of new enabling technologies within the production plant. These technologies include: Autonomous Robotics (programmable interconnected robots equipped with artificial intelligence), Additive Manufacturing (3D printing, digital manufacturing), Mixed Reality (MR), extensive use of simulations to test new processes related to production before putting them into practice in reality, Industrial Internet of Things (IIoT), Horizontal and Vertical System Integration, Cyber-Security and Cyber Physical Systems (CPS), and Big Data Analysis of a vast amount of data to process real-time information useful to optimize the production [3]. The implementation of these new technologies requires the transition from the old factory concept to the new smart factory, characterized by the use of intelligent and interconnected machines, connected to the Internet.

Therefore, to cope with this transformation, innovative solutions must be adopted to fulfill the requirements of ultra low latency and high reliability in Industry 4.0 environment. Fortunately, the emerging fifth-generation (5G) system can provide a significant boost in this direction, especially within the help of Multi-access Edge Computing (MEC). In fact, MEC is a more configurable solution that adapts very well to the most disparate scenarios, such as telecommunications, manufacturing, banking and finance, retail, automotive and so on, guaranteeing in addition to the advantages of the Mobile Cloud Computing (MCC) a lower and predictable latency. MEC allows to provide real-time analysis of road and traffic conditions, perform predictive maintenance, improve yields, reduce defects, decrease waste, strengthen IT security, have an optimized supply chain and inter-operable systems and much more.

As well-known, a simulation software represents an important instrument for designing and evaluating a large-scale system where the use of real prototypes is unfeasible. But, no solutions are available today that well support these activities in an ETSI-MEC compliant framework. For this reason, this paper proposes a new tool, namely *MEC-simulator*, allowing researchers to evaluate the orchestration of MEC resources, model user mobility and traffic load, and measure end-to-end delay. To demonstrate the capabilities of *MEC-simulator*, the behavior of an industrial scenario, where autonomous vehicle, IIoT and MR devices jointly exploit computational capabilities deployed at the network edge, is investigated and deeply discussed.

The rest of the paper is organized as follows: Section 2 presents an overview of edge computing in industrial applications and the ETSI-MEC standardization process. The fundamental *MEC-simulator* models are described in Section 3. Section 4 presents an example simulation scenario and the results provided. Finally, Section 5 presents the conclusions and discusses the future works.

II. Background on Industry 4.0, edge computing and related standardization activities

The scientific literature demonstrated that 5G deployment with edge computing capabilities could efficiently support production processes in the Industry 4.0 ecosystem [4].

A. Reference examples

The first example reported herein considers an industrial system made where Industrial Internet of Things (IIoT) devices are widely adopted when production processes require flexibility and extreme productivity. Here, IIoT devices have access to a huge amount of data such as CNC tool speeds, vibration, temperature, energy consumption, and many others. The serious problem is that such devices cannot process these information locally because of their limited computational and storage capabilities. For this reason, many industries, (e.g., Siemens [5] and Dell [6]) decided to use edge computing to aggregate this data and perform complex activities such as anomaly detection.

Another example refers to Automated Guided Vehicles (AGVs), where vehicles are used on the ground for the transport, sorting, and packaging of goods train or for the assembly and welding of materials. Also, Unmanned Aircraft Systems (UASs) allow maintenance operations to be carried out in dangerous areas or difficult to access and are able to collect a lot of data in a short time and without interruptions. In both cases, edge computing is necessary to calculate the paths of the individual autonomous devices in real-time with high reliability in order to allow them to cooperate while preventing collisions [7].

Finally, mixed reality technologies, like Augmented Reality (AR) and Virtual Reality (VR) are transforming the human-machine interaction (HMI). While projection-based AR systems support manual manufacturing processes, MR provides real-time and context-related information useful for maintenance scenarios. AR guides users through warehouses to highlight interesting objects, and VR can be exploited in the training process. For both AR and VR technologies, edge computing can manage the huge computational load, the very low latency, and the positioning awareness required by these applications [8].

B. ETSI-MEC standardization

To support the challenging and aforementioned requirements, the European Telecommunications Standards Institute (ETSI) started in 2014 the standardization process of the Mobile Edge Computing (MEC). More recently, the working group extended the compatibility of MEC systems to Wi-Fi and wired networks, thus determining the transition from "Mobile" to "Multi-Access". Accordingly, the MEC abbreviation refers now to Multi-Access Edge Computing [14].

ETSI defines MEC as a platform that provides an IT service environment and cloud-computing capabilities at the edge of the mobile network, very closed to mobile subscribers [15]. The reference MEC architecture proposed by ETSI GS MEC 003 [16] consists of two levels: the MEC System level and the MEC Host level. Each level is composed of functional elements (i.e.,virtual machines, according to NFV paradigm) and reference points enabling their interaction. The MEC System level integrates User Application Lyfe-Cycle Management Proxy (UALCMP), Operations Support System (OSS), and Multi-access Edge Orchestrator (MEO). It manages device authorization, mobility, the on-boarding of application packages, and the instantiation/termination procedure of a MEC App by selecting the appropriate MEC Host. Moreover, it has an overview of the overall MEC System (including distributed MEC Hosts, resources, MEC services, and network topology). The MEC Host level, instead, consists of Virtualization Infrastructure (VI), MEC Platform (MEP), and their managers namely Virtualization Infrastructure Manager (VIM) and MEC Platform Manager (MEPM). The VI provides the virtualized resources (RAM, CPU, storage, and networking), requested by MEC applications. MEP, instead, manages MEC services exposed to end users.

The main procedures defined in [17] are:

- **Instantiation**: it allows a user in the network with sufficient permissions or the operator to start an instance of a MEC app as a virtual machine on a MEC Host. The key points of this procedure are: authorizing the request, downloading/importing the application package, finding the best host MEC on which to instantiate the app, allocating hardware and network resources and providing the MEC services necessary for the app for proper operation. At the end of the procedure it is necessary to inform the user with an appcontext creation message.
- **Termination**: it can be initiated by a user that no longer needs for a MEC application or by the network operator in case of system overload, abnormal behavior of the app or for failure to use it. This procedure is necessary to properly release the MEC services to which the app has been subscribed, the network resources, and finally the virtualized hardware resources.
- **Radio Network Information updates**: the MEC System has to acquire RAN information (user position, uplink and downlink bitrate, measured SINR and cell change) with the dual purpose of ensuring MEC app network requirement (i.e. offloading app to a new MEC Host after the user's handover) and providing these information to other MEC app.
- **Mobility procedure**: network requirements (for example minimum latency) may not be satisfied after the handover. Here, the MEC system must be ready to instantiate a new MEC app in the most appropriate MEC Host, transfer the user context data, terminate the old instance and update the user on the new location of the MEC app. The mobility procedure is generally triggered by an RNI update and is managed by an ad-hoc MEC service: the Application Mobility Service (AMS).

TABLE I: Cross comparison among network simulation tools modeling edge computing.

EdgeCloudSim [9]	Sphere [10]	IoTSim-Edge [11]	PureEdgeSim [12]	SimuLTE-MEC [13]	Our work	
EDGE	EDGE (Cloudlet)	EDGE (IoT)	EDGE	EDGE (MEC)	EDGE (MEC)	Target system
✓		✓	✓	✓	✓	Delay (WLAN / LAN)
	✓		✓			Energy consumption
				No MEC App mobility	✓	Handover
✓		✓	✓	✓	✓	Bandwidth
✓	✓		✓		✓	Load Generator
	✓		✓		✓	Resource Orchestrator
✓		✓	✓	✓	✓	Mobility
		✓		✓	✓	Scalability
RAM/CPU		RAM/CPU/Storage	RAM/CPU/Storage	RAM/CPU/Storage	RAM/CPU/Storage	Consumption

C. Available network simulation tools

At the time of this writing, the usage of network simulation tools modeling the high complexity of edge computing systems covers a fundamental role in the evaluation (in terms of delay, bandwidth, resource management, mobility, and many more) of novel applications and services in the Industry 4.0 context, especially in those scenarios where making a test with real hardware could be prohibitively expensive and dangerous. Unfortunately, available instruments [9]–[12] focus on cloud computing simulators, and sometimes support communication and user mobility management capabilities. Nevertheless, they fail to provide a complete tool modeling the ETSI-MEC architecture, where computing resources are mainly deployed at the edge of the network and managed through specific and standardized functionalities (see Table I for more details). [13] proposes an ETSI-MEC partially compliant extension of the OMNeT++ network simulator, however it lacks some essential features such as MEO and MEC App Mobility procedure. To bridge this gap, this paper presents a system-level simulator named *MEC-simulator*, as a library of ns-3 simulator, a discrete-event and packet-level network simulator, mainly targeted for research and educational use. Differently from the other solutions reported in Table I, the developed tool is potentially able to interact with other ns-3 modules (such as WiFi, 3GPP, protocol stack, application models, mobility) and to be integrated within a mixed real/simulated network configuration.

III. The developed *MEC-simulator* tool

The *MEC-simulator* has been conceived as an open-source ns-3 extension, compatible with the rest of existing ns-3 modules. The source code is freely available at [18]. It is written in C ++, by fully leveraging object-oriented and event-driven paradigms of ns-3. At the time of this writing, the developed extension consists of 82 classes, 174 files, and around 18,000 lines of code and is fully working with ns-3 release 3.31. The main supported features are summarized below:

- The MEC management entity supports the instantiation of an application (and its termination as well) on a MEC Host, defined within the MEC System;
- The MEC management decides whether and on which MEC Host to instantiate the application, based on which features and MEC services that application requires;
- It is possible to deploy MEC Hosts in both radio nodes (i.e., base station) and the core network;
- The MEC System is able to maintain, during the handover, the connectivity between end-user and application instance
- The MEP on a MEC Host provides a framework for delivering MEC services and essential functionality to the hosted MEC applications. These information include the RNIS (Radio Network Information Service) and AMS (Application Mobility Service).
- The MEP provides functionality to route uplink and/or downlink traffic from MEC applications to the network.

A. Network topology model

As depicted in Figure 1, the *MEC-simulator* implements four main nodes: User Equipment (UE), Base Station (BS), MEC System and MEC Host. Each node contains at least one `MecEntity` that describes the functionalities related to the MEC behavior. To boost research and deployment activities, a topology helper class has been developed (in line with the vision of ns-3). Specifically, it creates all the `MecEntity`, configures them appropriately with default parameters, and connects them in order to enable the communication.

The `ClientApp` and `MecApp` classes model the application executed on the user equipment and the one executed on the MEC Host as a virtual machine, respectively. They are managed by the `Container` object.

`UeMecEntity` and the `ClientApp` can communicate with the MEC system and the MEC Hosts through the base station. Each user is connected to the physically closest base station. Every time it moves, the user checks the relative distance to the other base stations: if one is found closer, then the handover procedure is triggered.

The MEC System knows the state of all the available MEC Hosts. The i-th MEC Host is characterized by its computational capabilities: CPU_i represents the CPU capability expressed in terms of the number of instructions per second; RAM_i refers to the available RAM memory, and $STORAGE_i$ describes the available storage. Each MEC Host is directly connected to one or more base stations, through a very low latency communication link. On the other hand, MEC hosts form an overlay mesh network, with higher communication latencies.

B. End-to-end delay model

The end-to-end delay is modeled as the sum of the radio delay, the backhaul delay, and the execution time:

$$d_{e2e} = d_{radio} + d_{backhaul} + d_{exe}. \quad (1)$$

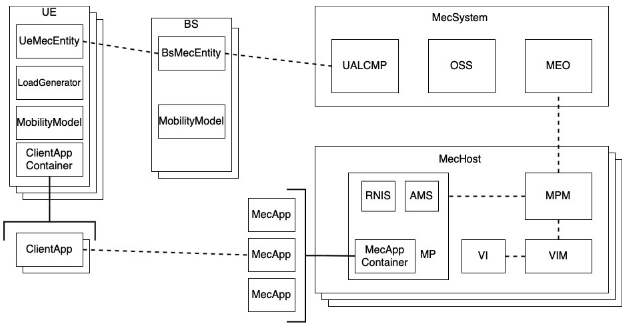

Fig. 1: ETSI-MEC architecture in *MEC-simulator*.

Algorithm 1: TaskScheduling

```
Input: new task arrives or a task is ended;
$N_{tasks}$ ← first 10 tasks in the FCFS queue of the
  $i$-th MEC host;
for each task in N$_{tasks}$ do
    Update number of instructions $I_j$;
    Compute residual execution time $\tau_j = I_j \cdot \frac{N_{tasks}}{CPU_i}$;
    Update scheduled task termination and $d_{exe}$;
end
```

The radio delay, d_{radio}, describes the communication latency over the radio interface, taking care of scheduling decisions, transmission, processing time at both BS and UE, re-transmissions, and propagation delay. The 5G-air-simulator [19] has been used to measure the radio delay in realistic deployments. The collected statistics, shown in Figure 2, are integrated within the *MEC-simulator*.

The backhaul delay, $d_{backhaul}$, represents the amount of time necessary for a packet (correctly received by the base station) to reach the reference MEC Host, and vice versa. Thanks to the usage optical fiber communications, this delay is really low. As default, $d_{backhaul}$ is set to 0 ms for a MEC Host directly connected to the reference base station and to 1 ms for a MEC Host that is in the same access network but not directly connected to the base station.

The execution time, d_{exe}, is calculated by considering a scheduling strategy based on multitask processor and First Come First Serve (FCFS) queue. It is assumed that a maximum number of tasks equal to 10 can be simultaneously served by the MEC Host. Therefore, a new request is directly served if the number of active tasks is lower than 10. Otherwise, it is queued (first) and scheduled (later). Given these premises, d_{exe} is dynamically updated, based on the status of the queue (see Algorithm 1).

C. Orchestration of MEC resources

The `MultiAccessEdgeOrchestrator` class implements the orchestration functionality, which is in charge of distributing MEC applications among the available MEC Hosts by using the `FindBestHost()` method. At the beginning, `MultiAccessEdgeOrchestrator` assumes that all the MEC Hosts are empty (i.e., there are no active tasks). Each time a new request arrives, the orchestrator selects the best MEC Host that is able to serve the related MEC application (i.e., because it is able to offer RAM and STORAGE resources to satisfy the request) and is experiencing lowest CPU usage.

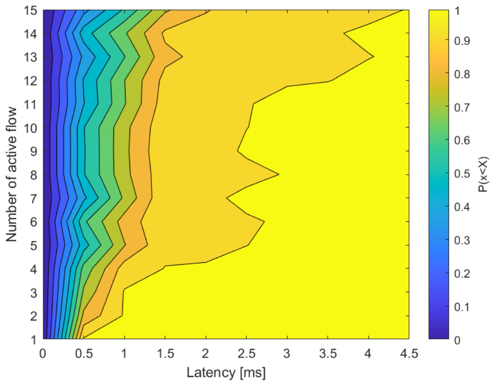

Fig. 2: Empirical CDF of the Radio Latency

D. Mobility and traffic load models

Each user can request a MEC service at any time. As far as mobility is concerned, ns-3 supports a set of mobility models that are used to track and maintain the current cartesian position and speed of an object. While the initial position of objects is typically set with a `PositionAllocator`, `MobilityModel` determines how objects move over time. `ConstantPosition`, `RandomWalk`, `RandomWaypoint`, and the highly flexible `TraceReader` mobility models are available.

Instead, for what concerns MEC service requests, the `LoadGenerator` class has been implemented to properly generate the traffic load. Here, user's requests are modeled through a memory-less Poisson process: the interarrival time between two requests is modeled according to an exponential random variable with parameter λ and probability density function $f_X(t) = \lambda e^{-\lambda t}$. Likewise, the service time, T_S, that refers to the time during which the user uses the MEC service, is modeled according to an exponential random variable with parameter μ and probability density equal to $f_{T_S}(t) = \mu e^{-\mu t}$.

IV. Numerical investigation

To demonstrate its great potentials, the developed simulation tool is used herein to evaluate the performance of a manufacturing industry, exploiting the edge computing paradigm. The considered industrial plant is spread over an area of 800 m x 800 m, where workers, UAS, and AGV move around and request services to the available MEC Hosts. As depicted in Figure 3, the whole area is covered by 16 femtocells distributed in a grid. Each femtocell covers an area of 200 m x 200 m. Moreover, it is assumed that 16 MEC host are co-located with base station, and fully meshed with each other. The MEC servers are configured according

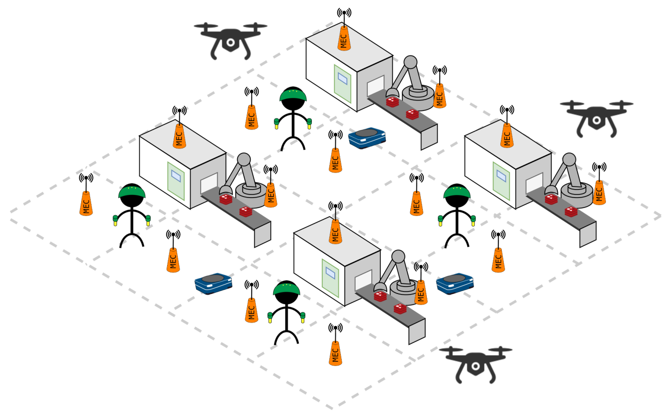

Fig. 3: The reference scenario considered in the numerical evaluation.

to state of art of industrial computer [20]: $RAM_i = 16$ GB, $STORAGE_i = 240$ GB and $CPU = 100000$ MIPS. Performance are evaluated for a different number of connected device, ranging from 10 to 200. The simulation time is equal to 8 hours.

According to the current literature, a specific set of applications is taken into account, whose parameters are reported in Table II [4] and discussed below:

- *Task Size*: It is expressed in MI unit and determines the task execution duration. While the Mixed Reality application is characterized by a video stream rendered starting from the user's context and sent constantly, in IIoT and AGV/UAS cases the computational cost depends on the data that IIoT sensors or autonomous vehicles are able to capture. The data can be one-dimensional, images, videos, and at different rates. In the present study, the task length is uniformly distributed for MR application between 200 and 400 MI, for autonomous vehicles from 50 to 300 MI and for IIoT it is between 20 and 50 MI.
- *Task Interarrival*: it defines how frequently the client app generates a request. It is defined constant for MR (33 ms) and IIoT (2 ms). In autonomous vehicles, it is uniformly distributed between 10 and 100 ms because it can depend on their speed or on the environment that surrounds them.
- *Booting Time*: it describes the amount of time that CPU takes to run operations in booting process, when there is no resources contention. It is a constant value, always.
- *Load Generator Parameters*: Not all devices in the simulation need to always have a MEC app instantiated. For example, autonomous vehicles alternate active periods with periods in which they must charge their batteries, as well as workers can take breaks between one MR session and another. In this case, the parameters μ and λ are properly set to model the interarrival and service time as described in the Section III-D.

TABLE II: Details about the considered application and mobility models.

	MR [21]	**AGV** [22]	**UAS**	**IIoT** [23]
Mobility Model	Random Walk	Random Waypoint		Fixed
Speed	3 Km/h	3 km/h	10 Km/h	-
Task Size (MI)	200-400	50-300		20-50
Memory (GB)	1.3	2		0.5
Storage (GB)	5	1		10
Task Interarrival (ms)	33	10-100		2
Active/Idle (s)	-	-		20/30
Booting Time (s)	2.5	1		2
Device [%]	12	6	6	76
Packet size [byte]	40000	200		50
$1/\lambda$ [min]	10	60	30	Always on
$1/\mu$ [min]	10	90	15	-

Figure 4 shows the usage of CPU, RAM and STORAGE resources of the 11-th MEC Host with 150 connected devices. A moving mean filter with window length of 60 sec is applied in order to mitigate the rapid changes of the CPU state and make the plot visually appreciable. STORAGE is the least used and most stable resource over time, there are peaks in RAM usage due to new instantiation procedure or device mobility events. The average CPU trend follows the behavior of the RAM. Instead, Figure 5a reports the average usage of resources among all the MEC Hosts, for a different number of connected device. As expected, the resource usage increases with the number of connected devices. In particular, the CPU has been identified as the system bottleneck. Of course, CPU usage of MEC Host affected the processing and so the end-to-end delay, as shown by Figure 5b. With no more than 50 devices, the guaranteed delay is, on average, in the order of a few milliseconds. With 100 devices it reaches hundreds of millisecond. Finally, with 150 - 200 devices it exceeds the second. This shows how important it is to make an accurate study in the sizing of the MEC Hosts and their distribution, because computationally intensive tasks (such as mixed reality) could easily occupy the entire CPU, delaying the response of the system towards other delay sensitive applications such as IIoT. Some possible solutions could be test CPU scheduling algorithms that take into account task delay requirements or move some of the CPU load to the GPU which is more suitable for video based computations.

V. Conclusions and Future Works

After discussed that edge computing is an appropriate technology to support the advent of Industry 4.0 paradigm, this paper presents a new Multi-access Edge Computing open-source simulator, called *MEC-simulator*, modeling the ETSI-MEC architecture and useful to test and evaluate the performance of a MEC system in an industrial scenario. Further work should be done to make the presented simulator fully compatible with the other ns-3 libraries in order to implement different access technologies and experiment with realistic realizations beyond 5G.

Acknowledgments

This work was supported by the Italian MIUR PRIN project no. 2017NS9FEY entitled "Realtime Control of 5G

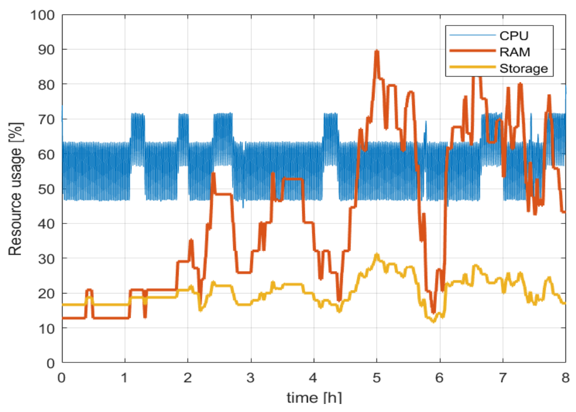

Fig. 4: Example of the resources usage for the 11-th MEC Host with 150 connected devices.

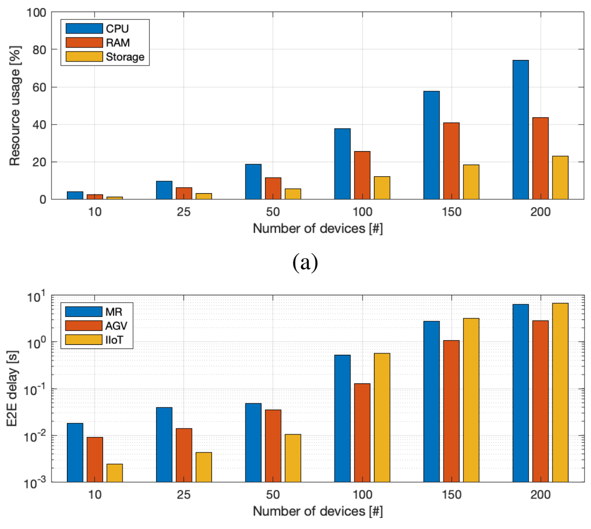

Fig. 5: (a) average MEC Host resource usage and (b) E2E delay for each application type.

Wireless Networks: Taming the Complexity of Future Transmission and Computation Challenges", by the Apulia Region (Italy) Research project INTENTO (36A49H6), and by the Italian MIUR PON projects Pico&Pro (ARS01_01061), AGREED (ARS01_00254), FURTHER (ARS01_01283), and RAFAEL (ARS01_00305).

References

[1] M. Dopico, A. Gómez, D. De la Fuente, N. García, R. Rosillo, and J. Puche, "A vision of industry 4.0 from an artificial intelligence point of view," in *Proceedings on the international conference on artificial intelligence (ICAI)*, 2016.

[2] A. Rojko, "Industry 4.0 concept: Background and overview." *International Journal of Interactive Mobile Technologies*, vol. 11, no. 5, 2017.

[3] S. Vaidya, P. Ambad, and S. Bhosle, "Industry 4.0–a glimpse," *Procedia manufacturing*, vol. 20, pp. 233–238, 2018.

[4] G. Brown *et al.*, "Ultra-reliable low-latency 5g for industrial automation," *Technol. Rep. Qualcomm*, vol. 2, 2018.

[5] H. Staufer. (May 2019) OT meets IT – Siemens Factory in Amberg, Germany. [Online]. Available: https://ingenuity.siemens.com/2019/05/ot-meets-it-siemens-factory-in-amberg-germany/

[6] Dell. Get an edge on your digital future. [Online]. Available: https://www.delltechnologies.com/en-hr/solutions/internet-of-things/

[7] Ericsson. (June 2019) Ericsson and China Mobile showcase automated guided vehicles for industry digitalization. [Online]. Available: https://www.ericsson.com/en/news/2019/6/china-mobile-and-smart-manufacturing

[8] S. Büttner, H. Mucha, M. Funk, T. Kosch, M. Aehnelt, S. Robert, and C. Röcker, "The design space of augmented and virtual reality applications for assistive environments in manufacturing: a visual approach," in *Proceedings of the 10th International Conference on PErvasive Technologies Related to Assistive Environments*, 2017, pp. 433–440.

[9] C. Sonmez, A. Ozgovde, and C. Ersoy, "Edgecloudsim: An environment for performance evaluation of edge computing systems," *Transactions on Emerging Telecommunications Technologies*, vol. 29, no. 11, p. e3493, 2018.

[10] D. Fernández-Cerero, A. Fernández-Montes, F. Javier Ortega, A. Jakóbik, and A. Widlak, "Sphere: Simulator of edge infrastructures for the optimization of performance and resources energy consumption," *Simulation Modelling Practice and Theory*, vol. 101, p. 101966, 2020, modeling and Simulation of Fog Computing.

[11] D. N. Jha, K. Alwasel, A. Alshoshan, X. Huang, R. K. Naha, S. K. Battula, S. Garg, D. Puthal, P. James, A. Zomaya, S. Dustdar, and R. Ranjan, "IoTSim-Edge: A simulation framework for modeling the behavior of internet of things and edge computing environments," *Software: Practice and Experience*, vol. 50, no. 6, pp. 844–867, 2020.

[12] C. Mechalikh, H. Taktak, and F. Moussa, "Pureedgesim: A simulation framework for performance evaluation of cloud, edge and mist computing environments," *Computer Science and Information Systems*, no. 00, pp. 42–42, 2020.

[13] G. Nardini, A. Virdis, G. Stea, and A. Buono, "Simulte-mec: Extending simulte for multi-access edge computing," in *Proceedings of the 5th International OMNeT++ Community Summit*, ser. EPiC Series in Computing, A. F\"orster, A. Udugama, A. Virdis, and G. Nardini, Eds., vol. 56. EasyChair, 2018, pp. 35–42. [Online]. Available: https://easychair.org/publications/paper/VTGC

[14] ETSI MEC ISG. (2019, July) Multi-access edge computing (MEC); fixed access information api. [Online]. Available: https://www.etsi.org/deliver/etsi_gs/MEC/001_099/029/02.01.01_60/gs_MEC029v020101p.pdf

[15] Y. C. Hu, M. Patel, D. Sabella, N. Sprecher, and V. Young, "Mobile edge computing—a key technology towards 5G," *ETSI white paper*, vol. 11, no. 11, pp. 1–16, 2015.

[16] ETSI MEC ISG. (2019, January) Multi-access edge computing (MEC); framework and reference architecture. [Online]. Available: https://www.etsi.org/deliver/etsi_gs/MEC/001_099/003/02.01.01_60/gs_MEC003v020101p.pdf

[17] A. Reznik, R. Arora, M. Cannon, L. Cominardi, W. Featherstone, R. Frazao, F. Giust, S. Kekki, A. Li, D. Sabella *et al.*, "Developing software for multi-access edge computing," *ETSI White Paper*, vol. 20.

[18] (2021, April) MEC-simulator. [Online]. Available: https://github.com/telematics-dev/MEC-simulator

[19] S. Martiradonna, A. Grassi, G. Piro, and G. Boggia, "5G-air-simulator: An open-source tool modeling the 5g air interface," *Computer Networks*, vol. 173, p. 107151, 2020.

[20] Fujitsu. FUJITSU IoT solution INTELLIEDGE. [Online]. Available: https://www.fujitsu.com/emeia/solutions/industry/manufacturing/edge-computing/

[21] K. Toczé, J. Lindqvist, and S. Nadjm-Tehrani, "Characterization and modeling of an edge computing mixed reality workload," *Journal of Cloud Computing*, vol. 9, no. 1, pp. 1–24, 2020.

[22] E. A. Oyekanlu, A. C. Smith, W. P. Thomas, G. Mulroy, D. Hitesh, M. Ramsey, D. J. Kuhn, J. D. Mcghinnis, S. C. Buonavita, N. A. Looper, M. Ng, A. Ng'oma, W. Liu, P. G. Mcbride, M. G. Shultz, C. Cerasi, and D. Sun, "A Review of Recent Advances in Automated Guided Vehicle Technologies: Integration Challenges and Research Areas for 5G-Based Smart Manufacturing Applications," *IEEE Access*, vol. 8, pp. 202 312–202 353, 2020.

[23] H. Liao, Z. Zhou, X. Zhao, L. Zhang, S. Mumtaz, A. Jolfaei, S. H. Ahmed, and A. K. Bashir, "Learning-Based Context-Aware Resource Allocation for Edge-Computing-Empowered Industrial IoT," *IEEE Internet of Things Journal*, vol. 7, no. 5, pp. 4260–4277, 2020.