Evaluating Architectural Patterns for Remote Control of AGVs

Ronal Bejarano, Roope Pääkkönen, Jan Olaf Blech
School of Electrical Engineering
Aalto University
Espoo, Finland
{ronal.bejarano, roope.paakkonen, jan.blech}@aalto.fi

Ian Peake
RMIT University
Melbourne, Australia
ian.peake@rmit.edu.au

Peter Herrmann
Norwegian University of Science and Technology (NTNU)
Trondheim, Norway
herrmann@ntnu.no

Valeriy Vyatkin
Aalto University, Espoo, Finland and
Luleå University of Technology, Luleå, Sweden
valeriy.vyatkin@aalto.fi

***Abstract*—Remote monitoring and control of factory equipment is increasingly attracting interest since it promises a more streamlined and therefore cheaper system operation and maintenance. The geographical distance between a factory and its control center, however, may influence the Quality of Service parameters of the network connections which might stymie the overall control process. To get a better understanding of these potential issues and their impact, we conducted a series of measurements over varying distances for the remote control, operation and simulation of Automated Guided Vehicles (AGVs) that are typical units in a modern factory environment. To achieve these tests, we defined three architectural patterns reflecting local and remote connections as well as the usage of cloud-based services. Applying these patterns, we connected the Factory of the Future at the Aalto University in Finland with the VxLab at the RMIT University in Australia and the Microsoft Azure cloud in the Netherlands. This allowed us to measure several Quality of Service networking parameters for the communication over short, medium, and very long distances. In this paper, we introduce the architectural patterns and the settings of our tests. Moreover, we present first empirical results and discuss their impact on the remote control of AGVs.**

***Index Terms*—Automated Guided Vehicles (AGVs), architectural patterns, cloud computing, remote control, operation and maintenance, virtual private networks (VPNs)**

I. Introduction

Remote operation, maintenance and, to some extent, commissioning of factory or mining equipment has gained increased attention in the last decade. This holds particularly for the manufacturing, material handling, and mining industries (see, e.g., [1]). The remote operation is supported by rising levels of digitization, in particular by trends such as Industry 4.0 and the Industrial Internet of Things (IoT). Also, recent events such as the COVID-19 pandemic increases the attractiveness of such technologies since skilled technicians may not be available, close contact between humans is avoided, and travel needed to maintain service or supply chains may be disrupted.

Nevertheless, the successful use of remote maintenance and operation is subject to some important conditions. A system must respect the needs of human operators and technicians who need to be able to act in real time. In addition, a remote control system has to incorporate existing infrastructures that are operated using legacy application architectures and protocols. Also the data to be transferred can be rich, e.g., when not only sensor values and operator commands but also video streams, that enable the human operator to monitor the operation process visually, have to be transmitted. Finally, a potentially large number of autonomous entities such as AGVs and mobile robots may be used on a plant site often in the vicinity of humans, which may afford immediate reactions under very hard real time conditions.

All the aspects make high demands on the performance of the used networks. This is further complicated by the fact that the underlying networks connecting the data producers and consumers, often comprise a variety of wide-area and local-area computer networks (such as 5G, WiFi, and field buses). Moreover, various cloud and edge data processing techniques might be involved.

One often useful way to meet these challenges is to replicate the situation locally on the operator's control center in order to anticipate potentially critical developments in due time. Such technologies as digital twins and online simulation [2], synchronised with the physical process, can be seen as elements helping to find a solution.

The core of any solution, however, is to be sure that the networks connecting a plant with a control center fulfill certain dedicated Quality of Service (QoS) parameters. Only then, we can guarantee that the technical means allowing us to react timely on critical developments are available. To address the complexity issues of the computer networks mentioned above, we defined a set of architectural patterns that are introduced in this paper. These patterns cover some typical network layouts for remote control. In particular, both local

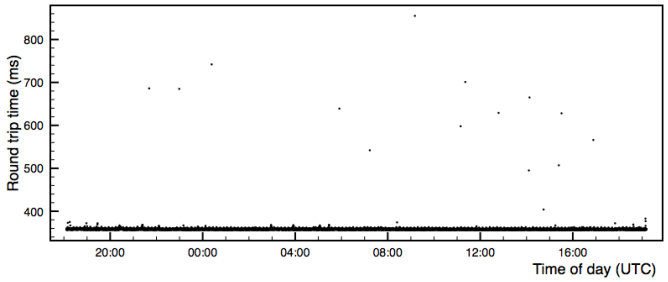

Fig. 1. Tests NTNU-RMIT, see [5]

and remote connections as well as the use of cloud services are reflected.

Based on the architectural patterns, we further present some initial tests, in which we gathered QoS parameters across three complicated communication and computing infrastructures. We see the results of these tests as a starting point to investigate the entire remote failure resolution scenario for a set of AGVs. The equipment used for our tests resides on two sites: The Factory of the Future [3] at the Aalto University in Espoo (Helsinki), Finland, and the Virtual Experiences Laboratory (VXLab) [4] at RMIT University in Melbourne, Australia. Furthermore, the Microsoft Azure cloud with its hosting site in Amsterdam, the Netherlands, is used. We present empirical results on latency, throughput, and availability that shall serve as a guide for other sites and for our own future extensions.

The article is structured as follows: After sketching some related work in Sect. II, we introduce the architectural patterns and the equipment used in Sect. III. Thereafter, the description of the conducted tests is presented in Sect. IV. In Sect. V, the results of these tests and their impact on the use of remote control technologies is discussed followed by some concluding remarks in Sect. VI.

II. Related Work

In previous work [5], we connected the visualization infrastructure of VXLab [4] with a Lego Mindstorms-based train system that was positioned in Trondheim, Norway [6]. This connection was based on the popular server-based IoT protocol AMQP [7] and allowed the remote monitoring and, with some time-based limitations, controlling of the trains from Melbourne. We installed a remote AMQP server in the Australian cloud infrastructure Nectar [8], which was connected to both the VXLab and the controllers and sensors used in the trains. Status information like the current position and speed of a train on a track or the settings of the switches, was directly sent to the remote AMQP server, which made direct monitoring from the VXLab possible. Likewise, control commands issued at the VXLab were sent via the remote server to the controllers on the train system where they were directly executed.

To get an idea about the transmission delays, intensive round trip time tests were carried out, see also [9]. Figure 1 depicts a 24 hour test between the lab in Trondheim and the remote

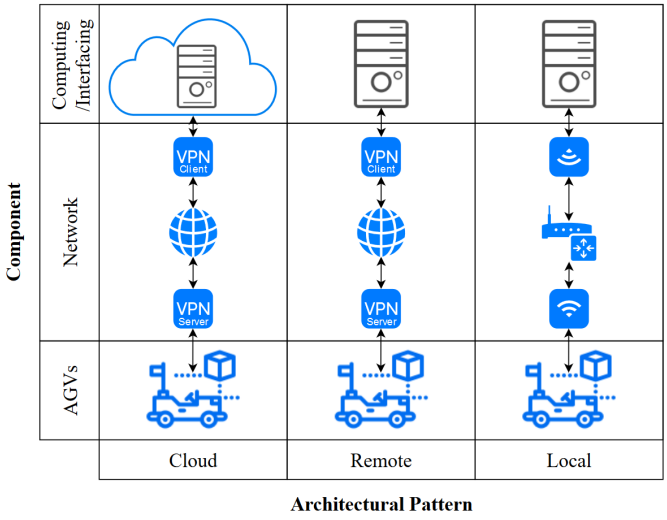

Fig. 2. Component mapping for architectural patterns of remotely controlled AGVs

AMQP server in the Nectar cloud. Here, a ping message was sent every two seconds for a whole day to get an idea if the round trip time is fluctuating. The figure reveals that the transmission delay was quite stable between 350 and 360 milliseconds. We experienced only very few fluctuations that never exceeded 880 milliseconds. Since we used the Internet infrastructure available at NTNU and the VXLab, and there are few distances larger than the one between Melbourne and Trondheim, these results seem promising for carrying out remote monitoring and control of technical systems.

The facilities of the VxLab were also utilized in [10]. Here, we investigated a software architecture for controlling robots that take advantage of the equipment offered in the VxLab (see Sect. III).

Profanter et al. [11] compare OPC UA, ROS and MQTT with respect to performance aspects. This paper concentrates on protocol evaluation, while we focus on architectural patterns.

III. Architecture

Our tests follow three main architectural patterns that are suited to be used in IoT and Industry 4.0 contexts. They are depicted in Fig. 2.

The access to many IoT devices is nowadays provided by cloud-based services, see, e.g., [12]. This is reflected by the *cloud architectural pattern* on the left side of the figure. The sensor readings and actuator commands of an AGV can be remotely accessed by devices using cloud data centers. The communication between these centers and the AGV is provided by a Virtual Private Network (VPN) that runs based on the Internet.

Remote access on AGVs can also be carried out by connecting them directly with the control devices instead of using a cloud. This is addressed by the *remote architectural pattern* shown in the center of Fig. 2. Here, the network is also realized by VPNs running on the Internet. The main difference is

Fig. 3. Industrial AGVs at the Aalto Factory of the Future

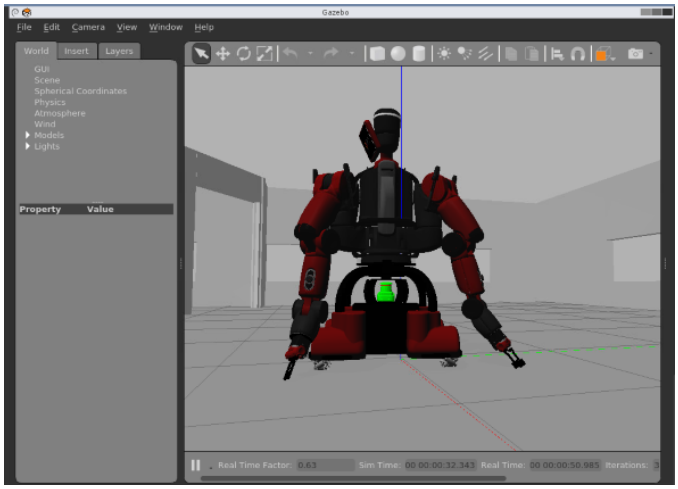

Fig. 4. CSRack-hosted robot simulation of the Baxter robot with a mobility base

that the AGV is directly linked with a control unit and not a cloud service. In the last years, a number of specialized communication protocols for IoT and Industry 4.0 applications such as MQTT [13] and AMQP [7] emerged. ROS uses *topics* for this purpose. These mostly server-based protocols vastly facilitate the use of the remote architectural pattern.

Of course, the access can also be local which is still the predominant way to operate AGVs. The *local architectural pattern* on the right side can be used for this case. As a mobile device, the AGV is connected with its control unit via a Wireless Local Area Network (WLAN) either by a direct peer-to-peer connection or, as shown in the figure, using a wireless bridge.

As mentioned in the introduction, we use the Aalto Factory of the Future , the VxLab, and the Microsoft Azure cloud to connect devices via the patterns introduced above.

The Aalto Factory of the Future [3] is a facility for research, innovation and educational projects at Aalto University. In particular, software aspects of flexible, reconfigurable manufacturing scenarios are studied. Figure 3 shows two AGVs. One is a MIR 100 that can, e.g., be connected to laptops and Raspberry Pi single board computers also residing at the Aalto University using the local architectural pattern.

The VXLab [4] is placed on the city campus of the RMIT University. It was developed to explore themes of software architecture and testing for remote monitoring and collaboration in the control automation industry. The lab consists of industry and collaborative robots, a seven meters long tiled display wall, and virtual reality equipment. These units are augmented by dedicated servers in the Cyber-physical Simulation Rack (CSRack) facility that consists of HP ProLiant blades in an RMIT data center running Ubuntu GNU/Linux. The CSRack hosts several projects including the Gazebo simulation engine (ROS Melodic, Ubuntu 18.04). Further, the VXLab has another MIR 100 as well as Rosie, the integration of a Baxter collaborative robot with a Dataspeed mobility base. Figure 4 depicts a screenshot of a running simulation of Rosie. Combining these units with those in Helsinki allows us to test the remote architectural pattern over a really long distance.

IV. Test Description

The software to test the network parameters was implemented in a Python 3 script. To execute the automated test, a device needs to invoke this script indicating its role as a server. Then, another device activates the testing script with a specific set of parameters in order to act as a client. Parameters like the number of testing iterations or the IP address and port of the server are necessary to launch the test. A single test runs automatically every minute until it reaches the predefined number of iterations. At the end of each test run, the results are logged in a database file hosted by the client device, tagged with the UTC time of execution.

Three relevant QoS parameters are tested: First, the latency L is examined. For that, the well-known networking utility ping is used 10 times with its default 56-bytes large packets, and the arithmetic mean of the response times is computed. If $t_p(k)$ is the response time of the k'th ping run, the latency is therefore calculated as follows:

$$L = \frac{1}{10} \sum_{k=1}^{10} t_p(k) \tag{1}$$

Second, the throughput T of data via TCP connections is tested. To achieve that, the points of time t_0, when a connection is initiated, and t_f, when its termination is confirmed, are extracted. When a TCP connection is established, we send the content of a buffer c times before ending the connection. Moreover, we experiment with different buffer sizes B that can be set to 1, 2, 4, 8, 16, 32 and 64 kilobytes. We can calculate the throughput in the following formula:

$$T = \frac{B * c}{t_f - t_0} \tag{2}$$

The TCP connection used in our tests was implemented by the Sockets module on Python.

Third, the script calculates the connection availability α. It corresponds to the ratio between successfully completed tests to all tests conducted. We consider a test as successful if the pings in the latency tests are confirmed and the throughput implementation does not fall below a certain threshold. Be s_u the number of successful and s_t the number of all tests carried out, the availability can be computed as follows:

$$\alpha = \frac{s_u}{s_t} \tag{3}$$

At the end of the preselected testing iterations, the script computes the availability based on the latency and throughput tests logged during the various test runs.

Our hypothesis is that using the existing AGV technologies with computing and interfacing resources, that are realized by remote agents, will lead to some problems due to subpar QoS parameter values. In particular, we fear that the higher complexity brought by the extended geographical reach of the data network and other external factors not under control of the AGV user may lead to slower data transfer times and a reduced availability that may jeopardize the proper execution of the AGV. To find out if our assumptions are valid, we carry out the proposed tests that also allow us to evaluate and compare the influence of the architectural patterns introduced in Sect. III for the use case involving remote agents.

To test our hypothesis on different geographical bases, we carry out three test scenarios that are highly different with respect to the distances between an AGV and its remote agents. The first scenario is based on the remote architectural pattern. It reflects a city environment, where the separation of the agents is less than 20 kilometers, but there is not a dedicated channel of communication. The second scenario incorporates a cloud environment following the cloud architectural pattern. It connects devices at the Aalto Factory of the Future with one of the Microsoft Azure data centers in Amsterdam / Netherlands which are approximately 1,500 km apart from the Helsinki Area. The third scenario is used to test the connectivity over vast geographical distances. Here, we use the remote architectural pattern to link Helsinki with computing resources located at the VXLab of the RMIT University in Melbourne, Australia. The geographical separation between the two points is approximately 15,200 kilometers.

Carrying out the three scenarios will provide us with a better idea if the AGV realization using remote agents is feasible in practice. Moreover, we like to study the impacts of various external influences on the behavior of the experimental scenarios. Here, we contemplate the following items as influential for the test results:

1) *Buffer size and total amount of data:* The data traffic via a wide-area network is usually realized by a sequence of data packets travelling to their recipient via a number of routers that store the packets temporarily in buffers. To find out the effect of the amount of data sent in such transmissions via routers, it is worthwhile to vary the above discussed buffer sizes B and the number c of transmission iterations in a TCP connection. To determine the value of the buffer size and the total amount of data, this experiment takes a previous experience presented by Manzi et al. in [14] as reference. The authors reported the amount of data required to stream velocity commands remotely for a mobile robot (DoRo). Their robot operates based on the ROS framework. Velocity commands have to be transmitted 64.5 times a second while the size of each message is 236 bytes. Thus, considering an overhead of 30%, the network connection has to guarantee the transmission of around 20 kilobytes per second. In our tests, we replicate this data transfer behavior by using a buffer size of 2 kB and transmitting its content 10 times in a single TCP connection.
2) *Time of the day and network load:* As proposed for the cloud and remote architectural patterns that were presented in Sect. III, we use VPN connections that run on external Internet connections. An important factor for the connectivity parameters mentioned above is the performances provided by the Internet Service Providers (ISPs) on the way which may heavily depend on the network load used by the other customers of an ISP. The network load may vary over the day as it tends to be higher in the working hours than, e.g., at night. Thus, we like to find out if certain periodic patterns can be recognized which would allow us to predict at which time of the day the connectivity parameters might not be sufficient to guarantee an effective AGV operation. To analyze and identify the existence of such periodic behavior, the testing script will run for a whole weekday, i.e., 1440 minutes.
3) *VPN setup:* The VPN software of choice for the cloud and remote architectural patterns is OpenVPN, since it is a free and widely supported open-source Secure Sockets Layer (SSL)-based solution. The community edition of OpenVPN, however, has multiple features and versions that may influence the parameters of its connections. For instance, the newest version 2.4 offers a significantly enhanced behavior than the standard version 2.3 which is the default version pre-loaded for popular operative systems such as Ubuntu 16.04. In the result section of this paper, we will refer to the OpenVPN setup parameters used in the various tests.

V. Analysis of Experimental Results

The experimental results of our tests are summarized in Tab. I. The Figs. 5, 6, and 7 depict the evolution of the latency and throughput over the test period for the city, cloud, and remote use cases discussed in Sect. IV. In addition, the figures include lines describing the linear regressions over the sensed latency and throughput values as well as the corresponding equations (see, e.g., [15]). The lines provide hints about the generalized behaviors across the test periods.

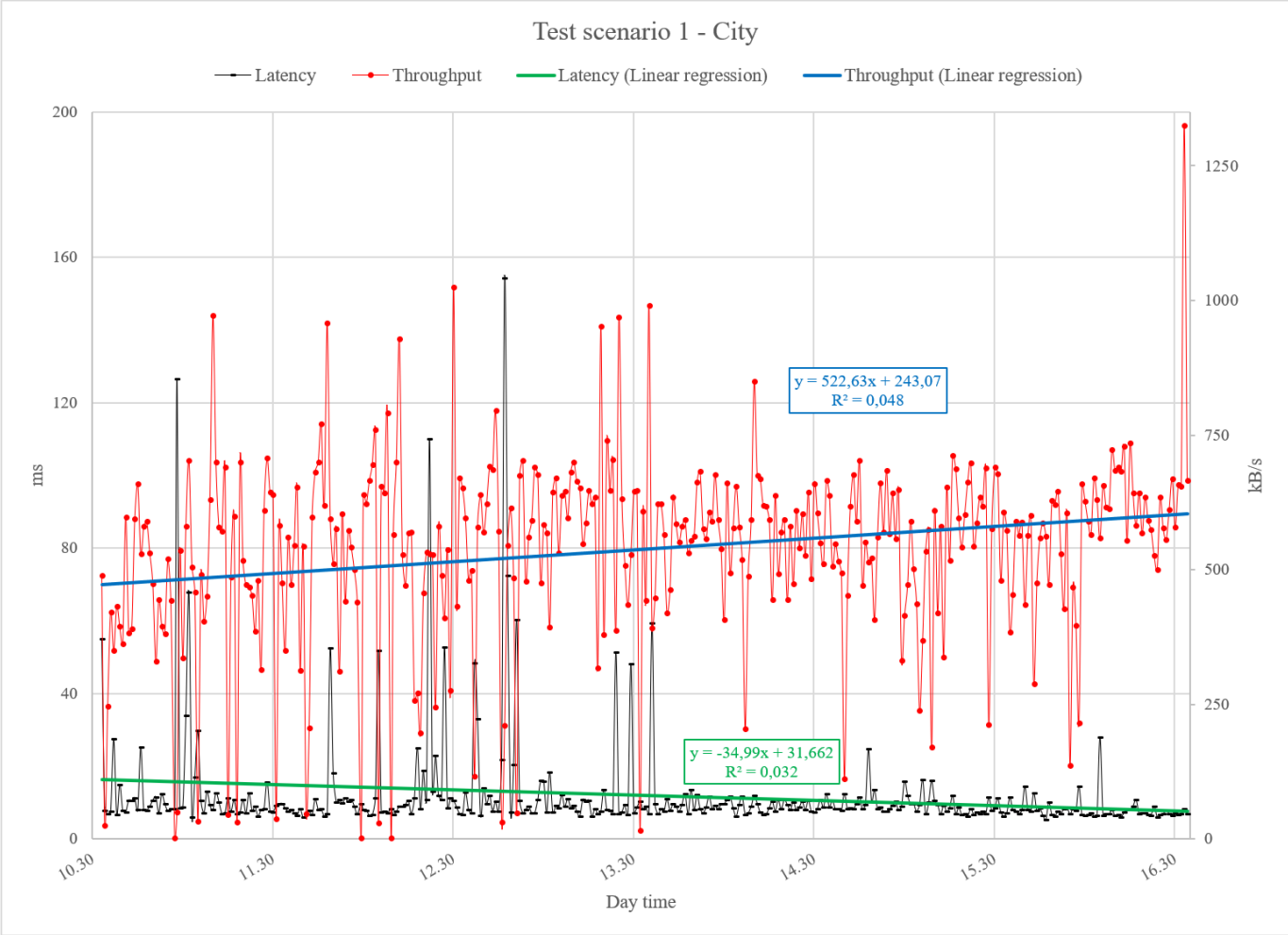

Fig. 5. Test results for the scenario 1: City (24/03/2020). Computing resources in Helsinki area (Finland)

TABLE I
LATENCY, THROUGHPUT AND AVAILABILITY RESULTS FOR THE THREE TEST SCENARIOS

Quality of Service parameter	**Statistic**	**Test scenario**		
		1. City	***2. Cloud***	***3. Remote***
Latency	μ (ms)	11.89	29.11	357.11
	max. (ms)	154.23	77.38	445.24
	min. (ms)	5.32	27.25	327.90
	σ	14.20	3.92	10.28
Throughput	μ (kB/s)	538.40	218.71	19.58
	max. (kB/s)	1324.70	239.82	20.53
	min. (kB/s)	1.28	18.64	2.30
	σ	173.33	20.38	0.73
Availability	Percentage	98.90%	99.93%	8.19%

Comparing the average μ latency results of the three scenarios in Tab. I reveals a positive correlation with the geographical separation between the computational resources. The latency results obtained for the city and cloud scenarios have a magnitude under 30 ms. This is typical for scenarios following the local architectural pattern in which WLANs are used. Thus, the behavior in these scenarios indicates appropriate responsiveness conditions that facilitates a fluid communication for both architectural patterns used. In contrast, the latency of the remote scenario is considerably higher since the average here is around 360 ms. This matches the results of the latency tests between Melbourne and Trondheim [5] that were carried out in 2015 and discussed in Sect. II.

To evaluate the stability of the latency values however, we should also consider the maxima as well as the standard deviations. According to that, the cloud scenario has the most stable latency performance, supported by the lowest standard deviation σ, while the remote architectural pattern-based scenarios (city and remote) indicate a higher dispersion. In particular, the remote scenario depicted in Fig. 7 shows significant fluctuations over time.

The average values for the throughput measurements point to a negative correlation with respect to the distances between the computational resources. A reason for this is that passing a larger number of routers increases the likelihood to pass one that is subject to heavy traffic. This may result in the activation of certain automated methods for congestion control over TCP networks (see Nagle's algorithm and the small packet

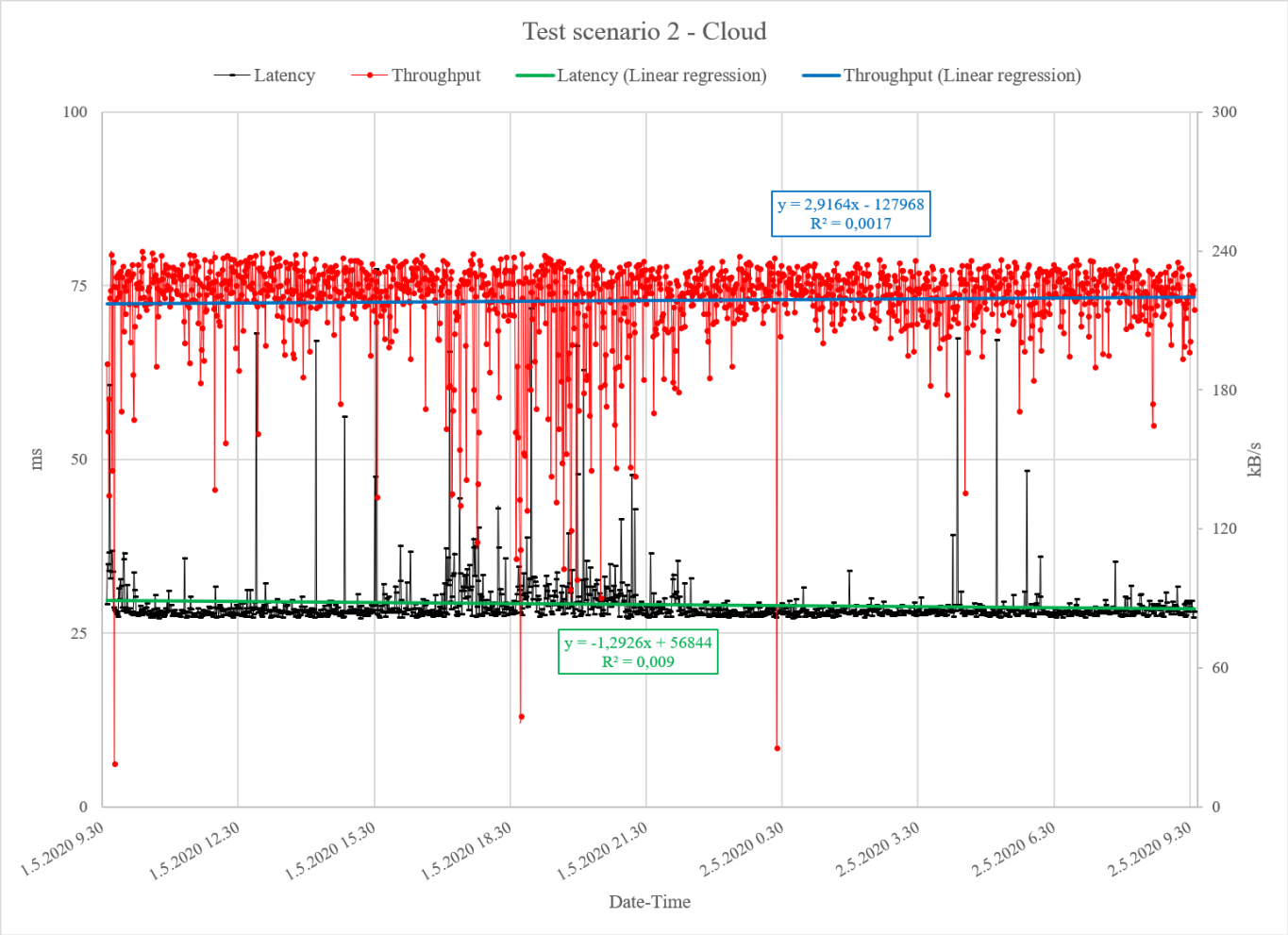

Fig. 6. Test results for the scenario 2: Cloud. Computing resources at Microsoft Azure - West Europe (Amsterdam / Netherlands)

problem [16]) which may reduce the throughput. To look into that, we conducted a separate route tracing test to find out the number of routers that are passed by the data packets. As expected, the networking over longer distances comprised more routers: In the city scenario five hops were enough to reach the recipient while for the remote case 14 routers had to be passed. These test results reveal the complexity scale in Not Standalone (NSA) network components for the architectural patterns. On the other side, the throughput in the remote scenario is more stable than in the city case as indicated by the standard deviation values σ.

To compute the availability values, we used 20 ms as our threshold in order to address the experiment described by Manzi et al. [14] that was discussed in Sect. IV. Here, we see that both, the city and cloud scenarios provide high availability. Depending on the safety requirements of a particular remote system control operation, the QoS may be even good enough to operate based on the equipment and architectural patterns used in these scenarios. But the test also reveals that the long-distance connection used in the remote scenario renders a QoS that is far too unstable to allow us to control such systems between Espoo and Melbourne. Here, only control mechanisms that demand a far lower throughput would work.

To assess the influence of the date and time on the tests, we should consider the time zones at the geographical locations of the computing resources. While carrying out our tests, the time zones of Amsterdam, Helsinki and Melbourne were UTC+2, UTC+3, and UTC+10, respectively. As shown by the low gradient of the linear regression line, the throughput in the remote scenario is quite stable. In contrast, the latency measurements in this scenario exhibits recognizable sinus-like oscillations in periods of four hours without an attributable reason. In addition, the line indicating the linear regression of the latency values follow a low slope. On the other hand, the other two scenarios that cover adjacent time zones between the AGV and its computing resources, depict periods with accumulations of throughput and latency spikes. For the city case, the phenomenon is appreciable in the morning hours, while in the cloud case it happens in the evening. Nevertheless, in both scenarios the effects are not very significant and their throughput values nearly always remain above the availability threshold. Thus, the external disturbances concentrated in time

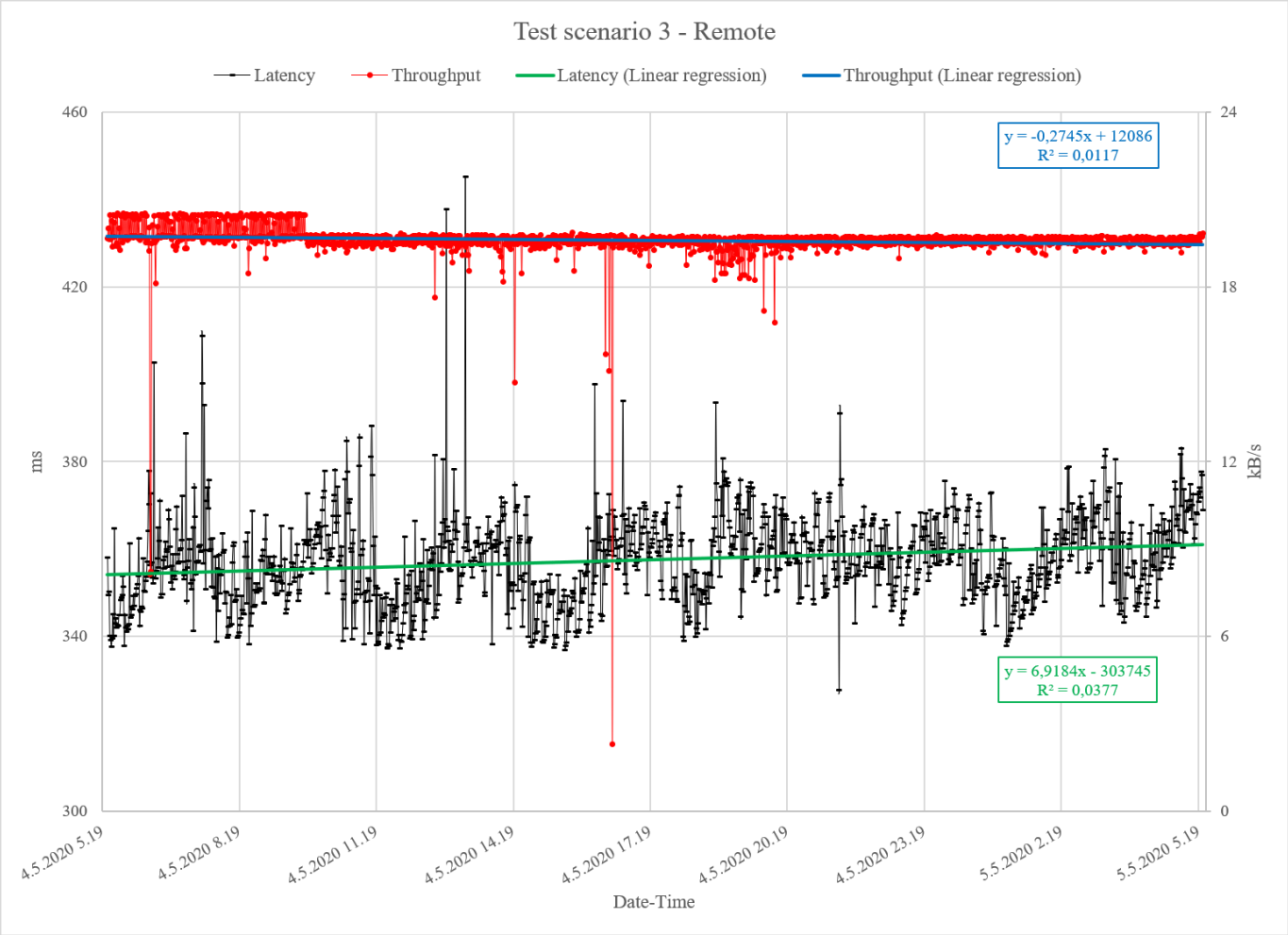

Fig. 7. Test results for the scenario 3: Remote. Computing resources at RMIT's VXLab (Melbourne, Australia)

lapses do not seem to have serious negative impacts on the system availability in these scenarios.

Finally, we want to look on the influence of the VPN used. As discussed, we applied OpenVPN that uses static keys with a size of 2048 bits, complying the Advanced Encryption Standard (AES) on the Cipher Block Chaining mode (CBC). The cryptographical hash in use is based on SHA-256 using a default parameter selection. The standard version of OpenVPN (version 2.3 for Ubuntu 16.04) was set to work initially on the connection-less transport protocol UDP. It could be successfully initialized for the city and cloud scenarios, but did not work in the remote scenario. Thereafter we tried to utilize the fact that OpenVPN is tailored for UDP by using this protocol instead of TCP in the remote scenario. But even then, OpenVPN was unable to establish a secure communication channel. However, the alternative version 2.4 of OpenVPN proved to be a better fit for using it with TCP. Using this version, we easily managed to establish the connections properly in all three scenarios. Therefore we used this version in all test runs.

Figure 8 depicts another latency experiment between Aalto and RMIT using ping. We show it in addition to Fig. 7 since it contains an interesting plateau that lasted for a few hours during the European night. We are not sure about the reason for the slight deviation of the latency from 310 to 340 ms but assume some maintenance work in the global communication network. We mention this case to demonstrate that there can, indeed, be external reasons that are beyond the control of the AGV operator. In a practical use of remote control systems over long distances the appearance of such surprising effects should always be taken into consideration.

VI. Conclusion

In this paper, we presented architectural patterns for the remote control and simulation of AGVs. We investigated connections between our European site in Espoo in Finland and Melbourne in Australia as well as between Espoo and the Microsoft Azure cloud in Amsterdam. The gathering and interpretation of empirical data on the network performance was a key aspect of this paper. This data can serve as a first indication on the expected QoS parameters like latency and throughput in larger setups.

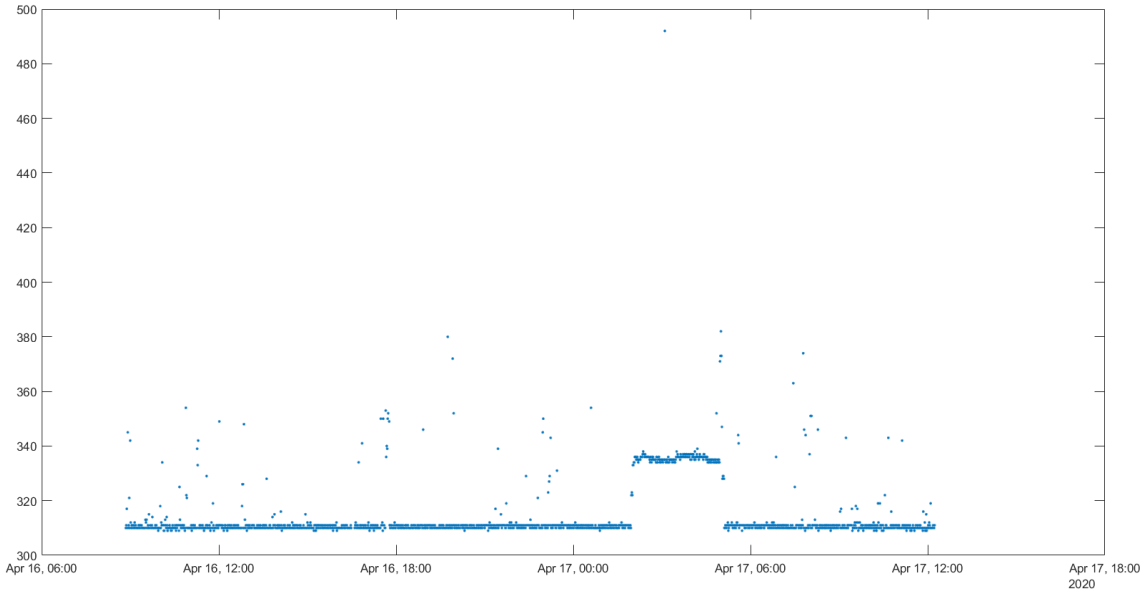

Fig. 8. Ping latency tests Aalto-RMIT

Currently, many industrial AGVs depend on a stand alone WLAN solution (as depicted in the local architectural pattern). In the near future, this standard will be complemented and, perhaps, replaced by interconnected solutions, e.g., based on 5G. These novel remote control procedures will be an enhanced way to link data and functions provided by AGVs to industrial automation platforms demanded by Industry 4.0 requirements. To safely realize the remote control procedures, new remote architectural patterns are required. Further, new use cases to facilitate a better application of distributed computing and the use of digital twins and simulation resources need to be studied.

Future work will incorporate more devices such as UR 3 robots and PLCs or PLC-like devices in our facilities. To research the impact of 5G-technology on the connectivity, we further plan to utilize a new 5G-Lab that is currently established at NTNU in Norway. Moreover, larger applications including Augmented and Virtual reality are in our scope for future work as well.

REFERENCES

[1] J. O. Blech, I. Peake, H. Schmidt, M. Kande, A. Rahman, S. Ramaswamy, S. D. Sudarsan, and V. Narayanan, "Efficient incident handling in industrial automation through collaborative engineering," in *2015 IEEE 20th Conference on Emerging Technologies & Factory Automation (ETFA)*. IEEE, 2015, pp. 1–8.

[2] G. S. Martínez, T. A. Karhela, R. J. Ruusu, S. A. Sierla, and V. Vyatkin, "An integrated implementation methodology of a lifecycle-wide tracking simulation architecture," *IEEE Access*, vol. 6, pp. 15 391–15 407, 2018.

[3] Aalto University, School of Electrical Engineering, "Factory of the Future," https://www.aalto.fi/en/futurefactory, accessed: 2020-05-08.

[4] I. Peake, J. O. Blech, L. Fernando, H. Schmidt, R. Sreenivasamurthy, and S. D. Sudarsan, "Visualization facilities for distributed and remote industrial automation: Vxlab," in *2015 IEEE 20th Conference on Emerging Technologies & Factory Automation (ETFA)*. IEEE, 2015, pp. 1–4.

[5] P. Herrmann, A. Svae, H. H. Svendsen, and J. O. Blech, "Collaborative Model-based Development of a Remote Train Monitoring System," in *11th International Conference on Evaluation of Novel Approaches to Software Engineering (ENASE), special session on Collaborative Aspects of Formal Methods*. Rome, Italy: SciTePress, Apr. 2016, pp. 383–390.

[6] S. Hordvik, K. Øseth, H. H. Svendsen, J. O. Blech, and P. Herrmann, "Model-based Engineering and Spatiotemporal Analysis of Transport Systems," in *Evaluation of Novel Approaches to Software Engineering*, ser. CCIS 703. Springer-Verlag, 2017, pp. 44–65.

[7] AMQP.org, "Advanced Message Queuing Protocol (AMQP)," www.amqp.org/, 2020, accessed: 2020-04-20.

[8] Nectar, "NectarCloud," cloud.nectar.org.au/, 2020, accessed: 2020-04-20.

[9] A. Svae, "Remote Monitoring of Lego-Mindstorm Trains," Project Thesis, Norwegian University of Science and Technology, Trondheim, 2016.

[10] J. O. Blech, I. D. Peake, G. Jahn, and R. Snooks, "A software platform for architectural robots," in *Proceedings of the ASWEC 2015 24th Australasian Software Engineering Conference*, 2015, pp. 54–58.

[11] S. Profanter, A. Tekat, K. Dorofeev, M. Rickert, and A. Knoll, "OPC UA versus ROS, DDS, and MQTT: Performance Evaluation of Industry 4.0 Protocols," in *Proceedings of the IEEE International Conference on Industrial Technology (ICIT)*, 2019.

[12] H. F. Atlam, A. Alenezi, A. Alharthi, R. J. Walters, and G. B. Wills, "Integration of Cloud Computing with Internet of Things: Challenges and Open Issues," in *IEEE International Conference on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData)*. IEEE, 2017, pp. 670–675.

[13] MQTT.org, "Message Queuing Telemetry Transport (MQTT)," mqtt.org/, accessed: 2020-05-07.

[14] A. Manzi, L. Fiorini, R. Limosani, P. Sincak, P. Dario, and F. Cavallo, "Use case evaluation of a cloud robotics teleoperation system (short paper)," in *2016 5th IEEE International Conference on Cloud Networking (Cloudnet)*, 2016, pp. 208–211.

[15] K. P. Murphy, *Machine Learning — A Probabilistic Perspective*. MIT Press, 2012.

[16] "Congestion Control in IP/TCP Internetworks," RFC 896, Jan. 1984. [Online]. Available: https://rfc-editor.org/rfc/rfc896.txt