





Article

Impact of Quality of Service on Cloud Based Industrial IoT Applications with OPC UA

Paolo Ferrari ^{1,*} , Alessandra Flammini ¹, Stefano Rinaldi ¹ , Emiliano Sisinni ¹ ,
Davide Maffei ²  and Matteo Malara ²

¹ Department of Information Engineering, University of Brescia, Brescia 25123, Italy;

alessandra.flammini@unibs.it (A.F.); stefano.rinaldi@unibs.it (S.R.); emiliano.sisinni@unibs.it (E.S.)

² Siemens Spa, 20128 Milano, Italy; davide.maffei@siemens.com (D.M.); matteo.malara@siemens.com (M.M.)

* Correspondence: paolo.ferrari@unibs.it; Tel.: +39-030-371-5445

Received: 9 June 2018; Accepted: 5 July 2018; Published: 9 July 2018



Abstract: The Industrial Internet of Things (IIoT) is becoming a reality thanks to Industry 4.0, which requires the Internet connection of as many industrial devices as possible. The sharing and storing of a huge amount of data in the Cloud allows the implementation of new analysis algorithms and the delivery of new “services” with added value. From an economical point of view, several factors can decide the success of Industry 4.0 new services but, among others, the “short latency” can be one of the most interesting, especially in the industrial market that is used to the “real-time” concept. For these reasons, this work proposes an experimental methodology to investigate the impact of quality of service parameters on the communication delay from the production line to the Cloud and vice versa, when gateways with OPC UA (Open Platform Communications Unified Architecture) are used for accessing data directly in the production line. In this work, the feasibility of the proposed test methodology has been demonstrated by means of a use case with a Siemens S7 1500 controller exchanging data with the IBM Bluemix platform. The experimental results show that, thanks to the proposed method, the solutions based on OPC UA for the implementation of industrial IoT gateways can be easily evaluated, compared and optimized. For instance, during the 14-day observation period of the considered use case, the great impact on performance of the Quality of Service parameters emerged. Indeed, the average communication delay from the production line to the Cloud may vary from less than 90 ms to about 300 ms.

Keywords: industry 4.0; distributed measurement systems; automation networks; node-RED; cloud computing; OPC UA

1. Introduction

Nowadays, the industrial world is using many technologies created for the consumer market and Internet: low-cost sensors, advanced computing and analytics [1]. The surprising level of connectivity supports the so called fourth industrial revolution, promising greater speed and increased efficiency. New embedded sensors/instruments with connectivity features move data from the production site (i.e., the machines) to the Cloud. The collection of data takes place from every production site in the world and, then, the giant quantity of gathered data is analysed. In this way, new services, derived from the analysis of the data, are offered with the intention of improving both the general system and the single machine performance [2]. Internet of Things (IoT) and Industrial Internet of Things (IIoT) are the keywords that label this ongoing evolution process in industrial automation [3]. The concept of Industry 4.0 (which has been addressed as the “fourth industrial (revolution) is involved as well but the terms are often misused and additional remarks are reported in the next section.

Even if there is not a clear distinction between IoT and IIoT (because, formally IIoT is a subset of the IoT), what is usually named as IoT can be considered the “consumer IoT”, as opposed to “Industrial IoT”. Consumer IoT is mainly centred around the human beings; indeed the “things” typically are smart appliances interconnected with one another in order to provide improved user awareness of the surrounding environment. On the other side, it is usually said that the aim of IIoT is to integrate Operational Technology (OT) with Information Technology (IT) domains. Differently from consumer IoT, IIoT communications are mainly machine oriented and can involve very different market sectors and activities. As a consequence, despite the most general communication requirements of IoT and IIoT are aimed at large scale connectivity, the specific needs can be very different. For instance, industrial scenarios pay attention to the Quality of Service (QoS, e.g., in terms of determinism and communication delays), the availability and the reliability. Very generally and roughly speaking, it is possible to resume that the IIoT originates when the IoT approach crosses the manufacturing stage of the production cycle.

Unfortunately, the ubiquity of Internet is only one of the aspects of the new era, not even the main one. The most researched subject is the utopic “single protocol” (i.e., accepted by any application market, industry and consumer) that could describe methods and data in a smart and flexible way. There are several examples of shared and widely used protocols in specific application markets and, probably, in industry, the most accepted protocol which harmonizes the machine to machine (M2M) interaction is OPC UA (Open Process Communications Unified Architecture). The OPC Foundation in the past had a great success with the “OPC Classic” and today it is proposing the OPC UA protocol as more powerful successor for its platform independent architecture. OPC UA makes use of most recent concepts to include key features like security, structured information model and auto discovery functions. Thanks to these important foundations, M2M can become “smart” enabling highly flexible scenario where machines could be self-organized. In contrast, the major Cloud platforms for data analytics and sales of services (e.g., Amazon S3, IBM Bluemix, Microsoft Azure, etc.) are still strongly related to the consumer market; they do not natively interface with OPC UA. Usually, gateways/proxies are required because their information model is different. Moreover, OPC UA is for industrial applications and it timestamps any data modification, while, on the contrary, the timely delivering of information among very different applications in Internet with the needed level of flexibility, scalability and geographic coverage is still not possible [4,5]. Considering all these situations, the delay/latency of services may be one of the most important advantages of new services in the near future, especially in the industrial market [6] that is used to real-time. However, despite the fact that it is typically affirmed that an IIoT communication framework is “generally” designed to fulfil timing requirements and packet losses minimization (e.g., for supporting functional safety and self-healing), very few, if any, research exists on experimental verification and tests in real-world scenarios. In particular, we focus on time-related performance, which are considered the most demanding for industrial applications [7].

Accordingly, the target of this paper is to overcome this gap, proposing an experimental methodology to investigate the impact on the delays of the various Quality of Service (QoS) parameters offered by the current Cloud platform, when they are used as sink/source points of data coming from machines with native OPC UA. The paper is voluntary focused only on data transfer passing through usual gateway toward Cloud platform; time for data elaboration in the Cloud is out of the scope of this work. The structure of the paper is the following: Section 2 introduces IIoT and Industry 4.0 scenario and the considered application; Section 3 details the proposed methodology, while Section 4 shows the experimental use case, the experimental results and the discussion about more general considerations. Finally, the conclusions are presented.

2. The Industrial IoT

The “Industry 4.0” tries to increase efficiency in the industry through the exchange/collection of information in the course of the entire product lifecycle. This concept requires the creation of the

so called “digital twin” of the product (or manufacturing process), a virtual repository where all the information about each product instance are stored. The union of physical and cyber components is called Cyber-Physical System (CPS) [8]. On the other hand, the consumer market uses IoT devices (smart devices) for providing better efficiency, comfort and safety by means of data and services exchange on a common infrastructure (Internet) with standard interfaces. The IIoT defines the overlapping area where IoT approach is used inside the Industry 4.0 architecture: this circumstance occurs especially when the product is physically created (manufactured), that is in the so called “operation phase”.

Industrial automation sector was always a pioneer of innovation and most of industrial systems (machines and plants) have a lot of installed smart field devices. Consequently, there are many connectivity options that can be used today but often, only data related to control are used, while the great amount of other available data is simply dropped. Industry 4.0 proposes: to collect data with consistent data models across the whole plant; to analyse that data extracting the meaningful information; and, lastly, to use services based on the revealed information. In the Industry 4.0 world the field devices and the Cloud can directly communicate. Thus, the new industrial communication hierarchy is flat: services are available for any participants of the automation system [3,9]. This approach changes the situation of applications that now use only local parameters (i.e., related to just one plant or machine e.g., [10,11]), allowing for the scope widening of the input data, since now they can come from other machines/plants worldwide. In addition, the high computational power in the Cloud can be exploited to aggregate/analyse/optimize data before their use in the field.

In the near future, Industry 4.0 applications will activate an endless loop with four main blocks: (1) Device in the field send measurements to the Cloud; (2) measurements are analysed with distributed algorithms in the Cloud; (3) system in the field receives back from the Cloud the optimized parameters; (4) those parameters are used to adapt/improve performance and efficiency of the production system.

2.1. The OPC UA in Industry

Up to very few years ago, the communication systems for industrial automation applications were oriented only to real-time performance suitable for industry and maintainability based on “international standards”. Among others, the most diffused industrial protocols today are, for instance; the wired based EtherNET/IP, PROFINET, Powerlink and EtherCAT; and the wireless based IEEE802.11, ISA100.11a and Wireless HART. However, it is known that interoperability between systems of different vendors with different protocols is always difficult because of the incompatible information models for data and services. Industry 4.0 manufacturing systems cannot rely only on such legacy approaches in order to reach the required flexibility level.

The most promising solution for this challenge is the OPC UA of the OPC Foundation [12,13]: OPC UA defines the mode for exchanging information between industrial engineering systems. OPC UA enhances the old OPC Classic with extended features in terms of data modelling, address space architecture, discovery functionalities and security. In OPC UA, servers contain the structured information model that represents the data and the communication model is client-server. Until today, OPC UA has been included in large number of machines and systems and it is a “de facto” reference method for process to process communication. Being more specific, the usual way to proceed is to offer OPC UA as the interface to export data from systems while, internally, the automation at field level is still implemented with traditional fieldbus networks. With OPC UA the data in the address space (described in the following) can be modelled without constrain to a specific communication protocol, obtaining information flows between heterogeneous systems with different data models. In conclusion, OPC UA is the major aspirant to be the backbone protocol for the harmonization of different industrial automation networks and systems [14].

2.2. OPC UA Outline

OPC UA has been designed to facilitate the exchange of information across the hierarchy of systems that commonly coexist in industry: enterprise resource planning (ERP); manufacturing execution systems (MES); control systems; and, last but not least, field devices. OPC UA has a message based communication and a service oriented architecture (SOA) with clients and servers connected to any types of networks.

A client application may use the OPC UA client API (application program interface) in order to send/receive OPC UA service requests/responses to/from the OPC UA server. From the programmer point of view, the OPC UA client API is like an interface that decouples the client application code from the client OPC UA communication stack. In the OPC UA API, there is a discovery service that can be used to find available OPC UA servers and to explore their address space. Clearly, the OPC UA communication stack converts the calls to the OPC UA API to proper messages for the underlying network layers.

In the servers, the OPC UA server API and the OPC UA communication stack are very like the client ones. As additional feature, the server has the so called “address space” in which it can expose the object to be exchanged. In OPC UA, a multiplicity of data structures (called “nodes”) can exist, representing, for instance: variables, complex objects, methods (i.e., remotely called functions) and definitions of new types for creating new OPC UA metadata. A hierarchical structure of arbitrary complexity can be created with OPC UA since an object node may contain other variables, objects, methods and so on. In other words, the OPC UA address space is the information model for the communication: real hardware devices or real software “objects” (sensors, actuators, software applications, etc.) are available for OPC UA communication only if they are modelled, added to the address space and finally discovered by the OPC UA clients.

2.3. Quality of Service

The definition of “Quality of Service—QoS” is usually depending on the application field because the “service” may vary from case to case [15]. Generally speaking, in industry, the QoS is often related to timeliness of services or to the guaranteed availability grade of a given service but it should be remembered that QoS timeliness does not necessarily imply a guaranteed delivery. Some examples are:

- the IEC 61850, where the QoS is related to the achievable class of latency in transferring data from the different parts of an electrical Substation Automation System;
- the Ethernet Time Sensitive Networking (TSN) group of standards, where the QoS is referred to the maximum allowed delay for a stream of transferred data;
- the MQTT (Message Queuing Telemetry Transport), where the QoS is tied to the confirmation that a message is delivered to any subscribers (which in turn impact again on latency).

As a consequence, the definition of “quality of service parameters” is even wider, since it refers to any parameters that can affect the QoS of the desired service.

The task of measure the effect of the variation of some parameters on the QoS of a given service could be cumbersome, requiring specific experimental setup for each different application. On the contrary, in this paper a general approach, focused on measuring latency and jitter of services, is proposed for general cases and not bound to any specific industrial applications.

3. The Proposed Approach

In any communication systems, the delay of a data transfer is related to the path of the data. Specifically, the user cannot modify most of the parts within a Cloud based architecture, which appears as “black box”. There are two methods for the estimation of the overall delay and its sub-components: simulations can be done as described in [16,17], or experiments can be setup as in [18,19]. However, it has to be highlighted that simulations offer the possibility to finely control the impact of varying the

quantity of interest but they can suffer from over simplified models, that do not accurately represent real-world scenarios. For this reason, in this work, a general-purpose test methodology based on experimental approach is designed and discussed that can fit several situations that are actually found in real plants. In particular, as previously stated, the focus is on the evaluation of time-related metrics.

Description of the Experimental Method

In the typical IIoT service scenario there is a machine that sends its data to the Cloud; there, data are elaborated together with other information of other machines; then, the outcome is “sold” to be used in some other field application/machine and, for this reason, it has to come back to the production field. It should be noted that, compared to IoT applications for mobile devices [20], the considered IIoT scenario is simpler.

The proposed method is tailored for the described situation and it is intended to measure the delay of the communication between: a data originator in the field (machine) and a data destination in the Cloud; and, on the reverse route, a data originator in the Cloud and a data destination in the field. The focus of this paper is to evaluate only the delay due to communication without taking into account elaboration of data; for this reason, the originator and the destination are the same machine/cloud application. Moreover, the sample data are sent with a loopback as soon as they reach the Cloud.

The block diagram of the proposed approach is shown in Figure 1. There are three players: the Machine; the IIoT Gateway; and the Cloud Application. Two OPC UA nodes A and B are created such that in their properties they include an array of timestamp values ($T1$ to $T8$). Nodes are sent from the Machine, the IIoT Gateway with OPC UA protocol and from the IIoT Gateway to the Cloud Application using one of the most diffused IoT messaging protocol (e.g., MQTT-Message Queuing Telemetry Transport). As soon as the nodes A and B pass in one of the timestamping points (labelled in the picture with the name of the corresponding timestamp), the matching timestamp value is loaded with the time of the operating system local clock. The Machine publishes data A at time $T1$; the IIoT Gateway gets it at time $T2$; closely after, at time $T3 > T2$, the IIoT Gateway sends A to the Cloud Application that finally receives it at time $T4$. Immediately, the cloud application copies back the object A to object B and it starts the reverse path sending back B at $T5$ to the IIoT Gateway. When B is received at $T6$ in the IIoT Gateway, it is routed to the Machine as a OPC UA node with departure time $T7$ and arrival time $T8$. After a complete roundtrip, the data structure contains all the timestamps (which are useful for the estimation of the delays).

The first important metric is the OPC UA end-to-end delay (OD) that can be calculated in the two directions using the timestamps $T1$ and $T2$ from Machine to Gateway and using timestamps $T7$ and $T8$ from Gateway to Machine:

$$OD_{MG} = T2 - T1 \quad (1)$$

$$OD_{GM} = T8 - T7 \quad (2)$$

The second important metric is the Cloud messaging protocol end-to-end delay (MD) that is obtained for the two directions using the timestamps $T3$ and $T4$ from Gateway to Cloud and using timestamps $T5$ and $T6$ from Cloud to Gateway:

$$MD_{GC} = T4 - T3 \quad (3)$$

$$MD_{CG} = T6 - T5 \quad (4)$$

The last metric is the total end-to-end communication delay ED from Machine to the Cloud and from Cloud to Machine, which is just the sum of the previously calculated partial paths:

$$ED_{MC} = OD_{MG} + MD_{GC} \quad (5)$$

$$ED_{CM} = MD_{CG} + OD_{GM} \quad (6)$$

Summarizing, the proposed experimental approach allows for the characterization of the communication delay of a generic industrial IoT application that uses OPC UA to access generic Cloud services. The three metrics presented above are simultaneously available in the experiments and can be used as performance indexes for the comparison of different setups.

In particular, the effect of different QoS parameter settings on the performance indexes of the communication delay will be studied in this paper, by means of a suitable use case.

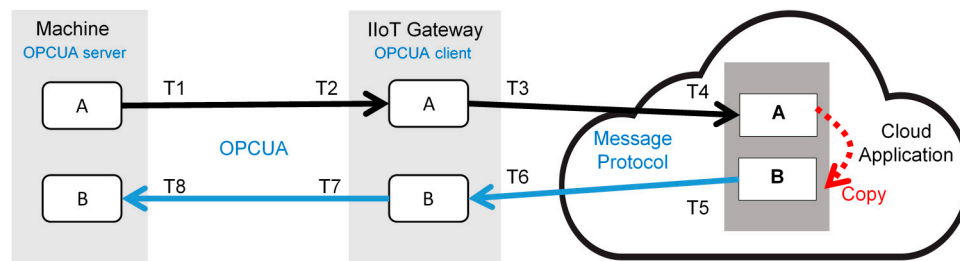


Figure 1. The block diagram of the setup for the experiment about the impact of quality of service parameters on the communication delay between a Machine with OPC UA interface and a Cloud platform. Black arrows show the path from the Machine to the Cloud, while blue arrows represent the reverse path. The red arrow is the software loop in the cloud that enable automatic bidirectional experiments.

4. An OPCUA Use Case

In order to show the applicability of the proposed measurement methodology, a sample use case is considered. It is clear that the obtained results depend on all the blocks, components and software composing this specific use case.

The experimental setup is realistically based on commercially available components and software for industrial automation. The Machine in the field uses a Siemens S7-1516 controller with embedded OPC UA communication stack. The IIoT Gateway has been built with a Siemens IOT2040 (embedded device with Yocto Linux-kernel 2.2.1, Intel Quark X1020, 1 GB RAM, 8GB SD disk, 2 × Ethernet ports, 2 × RS232/485 interfaces, battery-backup RTC). The considered Cloud platform is the IBM Bluemix; it runs the “Internet of Things Platform-m8” service, while it uses “Node-RED” framework for programming the data transfer/elaboration. IBM Bluemix has several access/use profiles: in this paper, the free access version is used, resulting in limited feature in terms of Cloud computational resources (which are out of the scope of this paper). However, no limitation from the communication features are mentioned in the contract.

The IOT2040 Gateway is attached to the same network of the Machine (in Milan at the Siemens Spa headquarter), hence the local area network introduces a negligible delay. Last, the Siemens network connection to the Internet has a very high quality with extended availability. As an additional remark, it has to be considered that network paths are not always guaranteed to be the same, due to the well known internet asymmetry [21]; however, the proposed methodology can identify such an asymmetry as well.

4.1. Node-RED Flows for the Experimental Setup

The experiments have been carried out using specific Node-RED flows. Node-RED is the graphic tool established by IBM for “wiring together hardware devices, APIs and online services in new and interesting ways” (from Node-RED website, <https://nodered.org/>). Node-RED uses the famous JavaScript runtime Node.js and by means of a browser based editor, allows for drag and drop, connection and configuration of “nodes” (i.e., open-source functions and interfaces). Thus, a program in Node-RED describes the data flow from source to destination passing through elaboration steps; for this reason, it is simply called “flow”. Node-RED is an efficient option for applications that are

aimed to prototype some IoT connectivity. In the specific application of the paper, Node.js has been considered as reference platform for the experimental implementation of the use case due to a trade-off between effectiveness and cost of human resources (programmers). The impact of Node.js can be estimated around 10% of the processing power of the gateway used in the demonstration use case. The number of devices connected to the gateway linearly increases the CPU and memory usage. As a consequence, Node.js with embedded devices is only recommended for small projects, with not demanding requirements, or for experimental test environment, as in the case of the current research.

In this paper, Node-RED is used both in the IIoT Gateway and in the Cloud application. The flows that implement the formulas/algorithms described in the previous section are shown in Figure 2 for IIoT Gateway along the path from Machine to Cloud, Figure 3 for IIoT Gateway along the path from Cloud to Machine and Figure 4 for the Cloud application.

It should be noted that the main limit of the OPC UA solution is currently the high computational power required by OPC UA stacks. Commonly available PLCs support a limited number of concurrent connections. In addition, the OPC UA implementation for Node RED in the use case is resource consuming also in the Gateway. Moreover, the Internet connection bandwidth could also affect the delays. Since this paper is focused on the measurement methodology and on the demonstration of the feasibility of such methodology by means of a sample use case, further investigations about different use cases are out of scope.

The flows are voluntarily kept as simple as possible to reduce computational overhead. The Gateway is forwarding the data object (containing the timestamp values as properties) to the Cloud using nodes of the OPC UA Library and IBM Bluemix Library. The properties of the data object are updated at the corresponding timestamping points. The Cloud application sends back any incoming data object after a fixed delay of 60 s. The data object is regularly sampled in the flows and saved to files for backup purpose; in case the complete path is not available, partial metrics can still be computed.

In this paper, for the configuration of the connection to the IBM Cloud, several QoS settings have been used and compared:

- “quickstart” mode (unregistered user)
- QoS = 0 with the “registered user” mode
- QoS = 1 with the “registered user” mode.

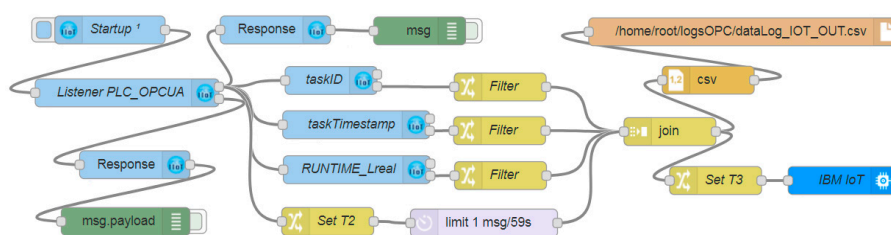


Figure 2. Node-RED flow for the IIoT Gateway related to the path from Machine to Cloud. The timestamps T_2 and T_3 are taken in this flow.

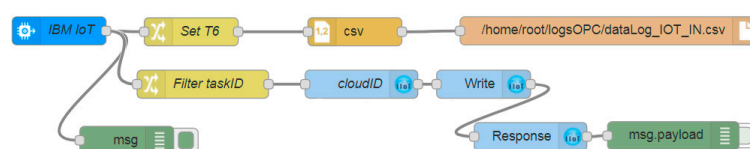


Figure 3. Node-RED flow for the IIoT Gateway related to the path from Cloud to Machine. The timestamps T_6 and T_7 ($T_7 = T_6$) are taken in this flow.

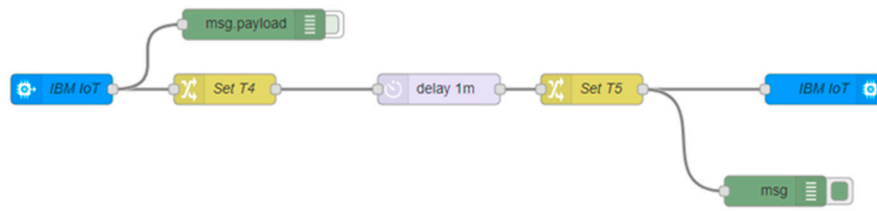


Figure 4. Node-RED flow for the Cloud application. Data is received from the IIoT Gateway and sent back after a fixed delay (one minute). The timestamp $T4$ and $T5$ are taken in this flow.

4.2. Synchronization Uncertainty in the Experimental Setup

In the proposed experimental methodology, the desired metrics are obtained combining timestamps taken by different actors working in a distributed system. Hence, the measurement uncertainty depends on several factors; among those, the major contributions are: the synchronization uncertainty among actors; the frequency uncertainty of the local oscillator in the physical device that takes the timestamp; and the uncertainty of the software delay of routines that correlate the event and the snapshot of the local clock to obtain the “timestamp”.

Fortunately, the entire transaction (from Machine to Cloud and back) is expected to last few seconds; as a result, the contribution of the local oscillator uncertainty can be neglected (usually crystal deviation on short period is in the order of few parts per million).

In this paper, the Machine, the IIoT Gateway and the Cloud platform use the Coordinated Universal Time (UTC) as time reference. The NTP (Network Transfer Protocol) has been used to synchronize each system with a local time server locked to GPS clock. The synchronization uncertainty using NTP may vary with the quality of the network connection in terms of latency [22,23] but all the considered systems are connected via a local area network with time server, reducing the variability to the minimum. Nevertheless, the synchronization uncertainty of the devices involved in this use case is estimated in different ways. For the IIoT Gateway, the time synchronization is experimentally measured considering the residual time offset after compensation, as listed in the NTP statistics [23]. For the Machine, the equivalent synchronization uncertainty is derived from Siemens documentation that sets the maximum error to 0.2 ms and by supposing the error distribution is uniform. Only the synchronization uncertainty of the IBM Cloud platform is difficult to be estimated, since no documents or literature is available on this topic. Anyway, it is hard to think that IBM cloud servers are worse than the other actors of the experiment case. For these reasons, the synchronization uncertainty of the IBM Cloud platform has been considered equal to the contribution of the IIoT Gateway.

The synchronization standard uncertainty results are shown in Table 1, where the experimental standard uncertainty is evaluated as the worst case $u_{sn} = \sqrt{\mu_{sn}^2 + \sigma_{sn}^2}$ because the systematic error μ_{sn} introduced by the operating system is not compensated. Anyway, the resulting synchronization uncertainty is always less than 0.1 ms with respect to UTC.

The timestamping uncertainty has been experimentally estimated at application level in the three actors. In details, a suitable software routine triggers a software delay at time $T9$ and takes a timestamp $T10$ when such a delay expires. Since timestamps and delay are obtained with the local system clock, the quantity $\Delta = T10 - T9$ should be theoretically identical to the imposed delay value. In truth, the timestamp uncertainty u_{tn} affects both $T9$ and $T10$. Including the systematic error μ_{Δ} , the timestamping uncertainty is $u_{tn} = \sqrt{\frac{\mu_{\Delta}^2 + \sigma_{\Delta}^2}{2}}$. Table 2 shows the timestamp standard uncertainty u_{tn} of the considered system.

In particular, if the proposed approach is used (Equation 1 to Equation 6 are considered), the standard uncertainty u_{mn} of any delay calculated between any two points (n and m), in the flow in Figure 1, is modelled as $u_{mn}^2 = u_{sm}^2 + u_{tm}^2 + u_{sn}^2 + u_{tn}^2$. It is clear that it is always dominated by the timestamp uncertainty.

Table 1. Synchronization uncertainty over an observation time of 14 days (ms).

Source	u_{sn}	μ_{sn}	σ_{sn}
Machine (theor.)	0.06	0	0.06
Gateway (exper.)	0.07	0.001	0.07
Cloud (supposed)	0.07	-	-

Table 2. Timestamp uncertainty over an observation time of 14 days (ms).

Source	u_{tn}	μ_{Δ}	σ_{Δ}
Machine (exper.)	0.01	0.016	0.01
Gateway (exper.)	4.2	4.589	3.8
Cloud (exper.)	5.9	4.214	7.3

4.3. Use Case Results

The experimental setup was aimed at the estimation of all the metrics used as performance indicators by means of a single experiment. The experiment has been repeated changing the QoS settings as described in Section 4.1. The measurement campaigns took 14 days. A new measure loop is started every minute; when the roundtrip ends, all the delay values regarding the run are stored. Each experimental campaign has more than 8000 valid samples and the resulting performance indicators are reported in Table 3. The results of Table 3 are discussed in the following Sections 4.4 and 4.5, where the probability density functions of the measurements are also shown.

Table 3. Delay for the considered use case over an observation time of 14 days (ms).

Path		“Quickstart”		QoS = 0 “Registered User”		QoS = 1 “Registered User”	
		Mean	St. Dev.	Mean	St. Dev.	Mean	St. Dev.
Machine to Cloud	OD _{MG}	135	53	173	61	176	64
	MD _{GC}	155	39	47	20	89	33
	ED _{MC}	289	67	219	62	265	69
Cloud to Machine	MD _{CG}	152	17	64	97	99	24
	OD _{GM}	16	6	18	12	17	10
	ED _{CM}	168	18	82	97	116	26

4.4. Discussion of the Use Case Result about Overall Delays

The detailed discussion of the overall delays is carried out only under the QoS setting called “quickstart”, that is the basic setting offered by the Cloud platform of this use case.

The probability density function estimates of the OPC UA delay in the two directions (Machine to Gateway, OD_{MG} e Gateway to Machine, OD_{GM}) are shown in Figure 5. In the Gateway to Machine direction there is a single, well defined, peak centred in the mean value; probably, the reception timestamp is assigned directly in the reception interrupt inside the Machine PLC, because the mean time is low. On the other hand, in the Machine to Gateway, direction the distribution shows three peaks with equal inter-distance of almost exactly 100 ms (please note that the peak centred at 280 ms is almost not visible in Figure 5). Given the analogy with similar situations [24], a multimodal shape of the OD_{MG} distribution signifies that at least one of the tasks that manage the OPC UA data in the Machine to Gateway operates on the basis of discrete cycle times. Under such a hypothesis, it may be inferred that the OPC UA task inside the Machine PLC is executed cyclically every 100 ms and timestamps are assigned inside that task.

The probability density function estimates of Cloud messaging protocol end-to-end delay (MD) in the two directions (Machine to Gateway e Gateway to Machine) are shown in Figure 6. The behaviour

is similar and the two distributions have only two main peaks. Finally, the probability density function estimates of the total end-to-end communication delay ED from Machine to the Cloud and from Cloud to Machine are shown in Figure 7. As expected, the distributions show several peaks, because they are the convolution of the distributions in Figures 5 and 6.

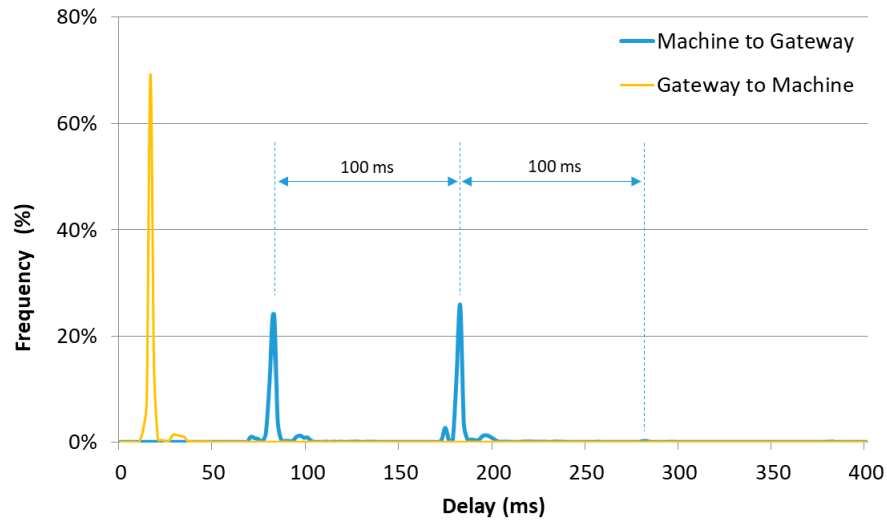


Figure 5. Probability density function estimate of the OPC UA end-to-end delay (OD) in the two directions (Machine to Gateway e Gateway to Machine).

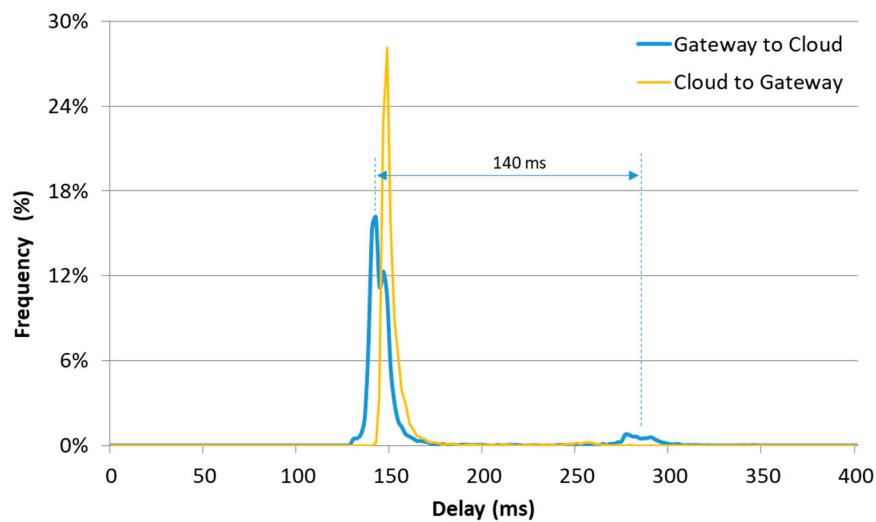


Figure 6. Probability density function estimate of the Cloud messaging protocol end-to-end delay (MD) in the two directions (Machine to Gateway e Gateway to Machine).

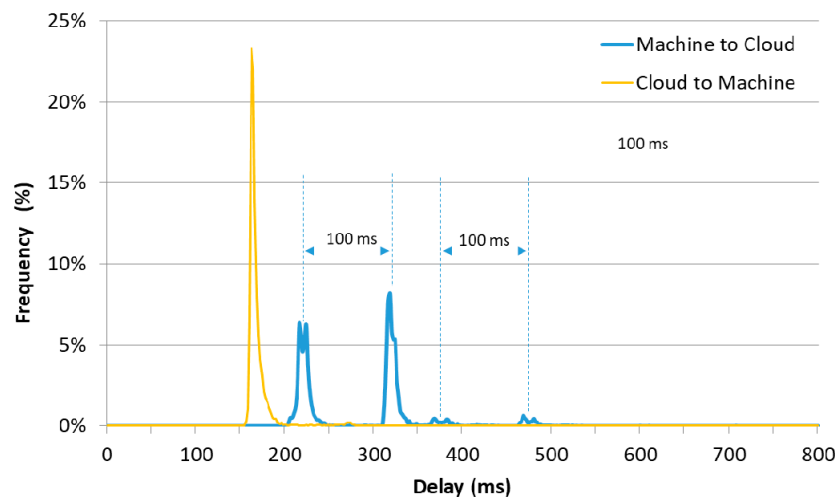


Figure 7. Probability density function estimate of the total end-to-end communication delay ED from Machine to the Cloud and from Cloud to Machine.

4.5. Discussion of the Use Case Result Varying QoS Settings

The three QoS settings described in Section 4.1 can be compared.

The delay OD_{MG} and OD_{GM} vary of few milliseconds in the three situations. The QoS settings (that only influence the Gateway to Cloud and Cloud to Gateway paths) do not affect the OPC UA drivers of the Machine and of the Gateway. For sake of completeness, it should be said that the OD_{MG} distribution maintain the equidistant peaks with any QoS setting.

The probability density function estimates of the MD_{GC} are shown in Figure 8, while the probability density function estimates of MD_{CG} are shown in Figure 9. Here, the three distributions are clearly different. The behaviour of QoS 1 and QoS 0 in registered mode are typical of MQTT data transfers across Internet (as shown in [25–27]): the QoS 0 is faster but it has a larger standard deviation, while QoS 1 distribution is narrow with an higher mean value. The probability density function of the “quickstart” mode has the same shape of the QoS 1 in registered mode but has the highest mean value.

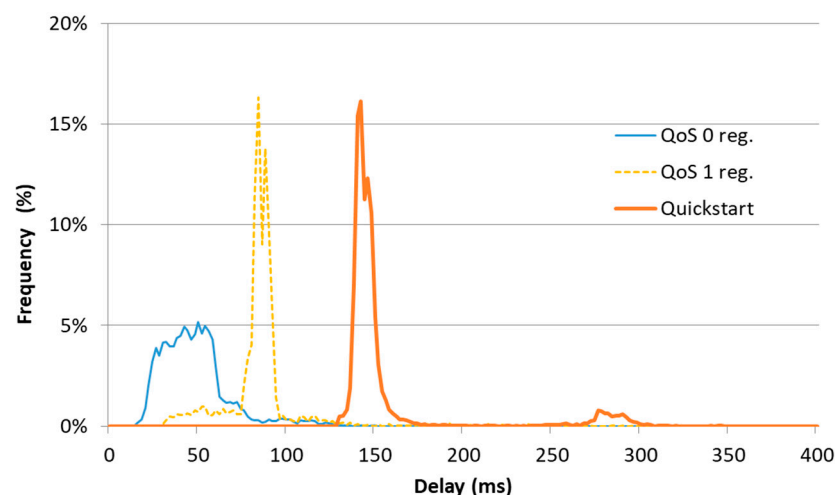


Figure 8. Probability density function estimate of the Gateway to Cloud delay (MD_{GC}) using different QoS parameters in IBM Bluemix.

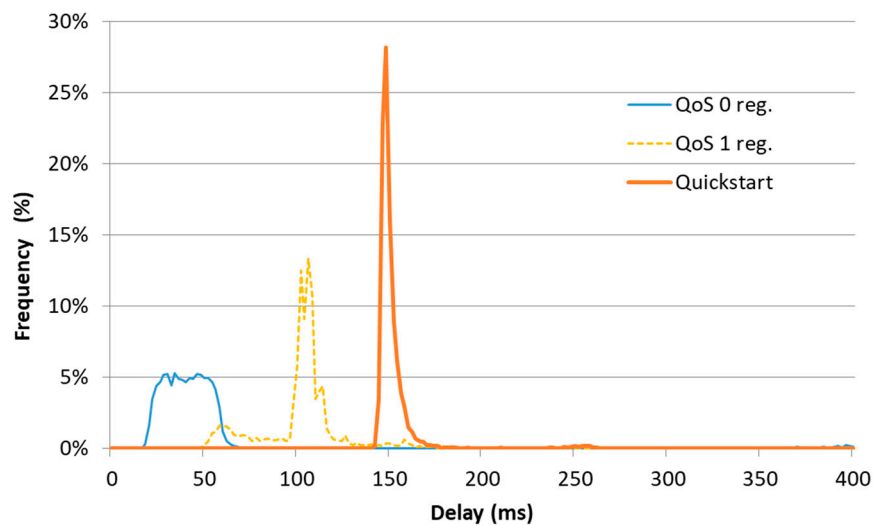


Figure 9. Probability density function estimate of the Cloud to Gateway delay (MD_{CG}) using different QoS parameters in IBM Bluemix.

Moreover, during the experiments, there were some anomalous delay values for MD_{GC} and MD_{CG} ; few samples (i.e., $<0.2\%$) were well above three times the standard deviation and were marked as outliers. A careful analysis revealed that the anomalies appear only at the same hour during the night on different days and the number of outliers were greater in the “quickstart” mode than in the QoS 1 registered. In conclusion, the sporadic anomalous delays may be due to the “free access” version of IBM Bluemix platform that has no guaranteed quality of service but it is clear that, among others, the “quickstart” mode nodes have the lower priority.

4.6. Generalization of the Use Case Results

The sample use case demonstrates the feasibility of the proposed measurement methodology, giving also some directions for more general applications.

First of all, the used PLC (Siemens S7 1516) is a medium performance PLC whose characteristics can be found also in PLCs of other producers, including the OPC UA support; for this reason, the proposed approach can be implemented also with other control systems.

The gateway architecture that has been implemented with a Node.js platform can be easily ported to different hardware (e.g., Raspberry PI devices) and improved, from the performance point of view, by optimizing the OPC UA library. However, the logic flow of the measurement methodology remains the same.

Last, the cloud platform can be changed (e.g., Microsoft Azure, or Amazon S3) provided that a suitable messaging protocol can be supported. MQTT is generally available but AMQP (Advanced Message Queuing Protocol) can be also considered. Anyway, the proposed methodology is not bonded to a specific messaging protocol, guaranteeing the consistency of the results.

Moreover, three important general observations arise from analysing the results of the use case.

The QoS of the OPC UA stack is clearly related to the implementation inside the PLC and, thus, their up/down scale is expected depending on the ratio between PLC computational performance and computational load.

The concern about performance of the Cloud is more about the availability (lack of timely response in some cases) than latency. The QoS at this level is closely related to the messaging protocol performance through the Internet, while Cloud computational power practically does not affect the results (since the proposed methodology correctly decouples it).

The performance of the Gateway is not stressed in the considered use case but a reasonable dependency of the results from the throughput (in terms of message per second) is expected. In large

applications, the gateway must scale his performance accordingly with the desired number of data exchanges with the cloud.

5. Conclusions

The success of Industrial Internet of Things (IIoT), from the economical point of view, depends on several factors. Among others, the “short latency” can be one of the most interesting, especially in the industrial market that is used to the “real-time” concept. This paper deals with a methodology to measure time delay metrics in OPC UA systems in order to study the impact that quality of service parameters have on the communication delay from the production line to the Cloud and vice versa. By means of a sample use case, the proposed method was applied, its feasibility was demonstrated and the results are generalized. In the use case, a Gateway exploiting the widely accepted OPC UA was used for data access directly in the devices inside the production line. The experimental results show that the overall delay is always bound to less than 300 ms, while the impact of the QoS parameters on the communication delay is clearly visible. The major experimental evidence in the medium term (14 days) is that the average delay from production line to Cloud is tightly related to the QoS settings of the IBM Bluemix platform. The “quickstart” mode has the worst performance with an average delay of 290 ms from Machine to Cloud and 170 ms in the opposite direction. If QoS 0 “registered user” mode is used, the average delays decrease, respectively, down to 220 ms and 80 ms. The QoS 1 “registered user” mode has higher delays but a lower standard deviation. Finally, it should be highlighted that the “free access” version of IBM Bluemix platform has no guaranteed quality of service: some samples (>0.2%) may be delayed by several seconds.

Author Contributions: Conceptualization, P.F.; Data curation, S.R. and E.S.; Formal analysis, E.S.; Funding acquisition, A.F.; Investigation, M.M.; Methodology, P.F.; Project administration, P.F. and A.F.; Software, S.R., D.M. and M.M.; Supervision, A.F.; Validation, P.F. and D.M.; Writing—original draft, P.F., S.R. and E.S.

Funding: The research has been partially funded by research grant MIUR SCN00416, “Brescia Smart Living: Integrated energy and services for the enhancement of the welfare” and by University of Brescia H&W grant “AQMaSC”

Acknowledgments: The authors would like to thank Siemens Italy Spa for hosting the experimental setup.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Xu, L.D.; He, W.; Li, S. Internet of things in industries: A survey. *IEEE Trans. Ind. Inf.* **2014**, *10*, 2233–2243. [[CrossRef](#)]
2. Tao, F.; Zuo, Y.; Xu, L.D.; Zhang, L. IoT-Based intelligent perception and access of manufacturing resource toward Cloud manufacturing. *IEEE Trans. Ind. Inf.* **2014**, *10*, 1547–1557.
3. Liu, Y.; Xu, X. Industry 4.0 and Cloud manufacturing: A comparative analysis. *J. Manuf. Sci. Eng.* **2017**, *139*, 034701. [[CrossRef](#)]
4. Yang, C.; Shen, W.; Wang, X. Applications of Internet of Things in manufacturing. In Proceedings of the 2016 20th IEEE International Conference on Computer Supported Cooperative Work in Design (CSCWD), Nanchang, China, 4–6 May 2016.
5. Bellagente, P.; Ferrari, P.; Flammini, A.; Rinaldi, S.; Sisinni, E. Enabling PROFINET devices to work in IIoT: Characterization and requirements. In Proceedings of the 2016 IEEE Instrumentation and Measurement Technology Conference Proceedings (I2MTC), Taipei, Taiwan, 23–26 May 2016.
6. Szymanski, T.H. Supporting consumer services in a deterministic industrial internet core network. *IEEE Commun. Mag.* **2016**, *54*, 110–117. [[CrossRef](#)]
7. Wollschlaeger, M.; Sauter, T.; Jasperneite, J. The Future of Industrial Communication: Automation Networks in the Era of the Internet of Things and Industry 4.0. *IEEE Ind. Electron. Mag.* **2017**, *11*, 17–27. [[CrossRef](#)]
8. Stojmenovic, I. Machine-to-machine communications with in-network data aggregation, processing, and actuation for large-scale cyber-physical systems. *IEEE Internet Things J.* **2014**, *1*, 122–128. [[CrossRef](#)]

9. Battaglia, F.; Iannizzotto, G.; Lo Bello, L. JxActinium: A runtime manager for secure REST-ful CoAP applications working over JXTA. In Proceedings of the 31st Annual ACM Symposium on Applied Computing, Pisa, Italy, 4–8 April 2016; pp. 1611–1618.
10. Rinaldi, S.; Ferrari, P.; Brandao, D.; Sulis, S. Software defined networking applied to the heterogeneous infrastructure of Smart Grid. In Proceedings of the 2015 IEEE World Conference on Factory Communication Systems (WFCS), Palma de Mallorca, Spain, 27–29 May 2015.
11. Fernandes, F.; Sestito, G.S.; Dias, A.L.; Brandao, D.; Ferrari, P. Influence of network parameters on the recovery time of a ring topology PROFINET network. *IFAC Pap. Online* **2016**, *49*, 278–283. [[CrossRef](#)]
12. *OPC Unified Architecture-Part 1: Overview and Concepts*; OPC Foundation: Scottsdale, AZ, USA, 2015.
13. *OPC Unified Architecture-Part 3: Address Space Model*; OPC Foundation: Scottsdale, AZ, USA, 2015.
14. Lee, B.; Kim, D.K.; Yang, H.; Oh, S. Model transformation between OPC UA and UML. *Comput. Standard. Interfaces* **2017**, *50*, 236–250. [[CrossRef](#)]
15. Liao, Y.; Leeson, M.S.; Higgins, M.D.; Bai, C. Analysis of In-to-Out Wireless Body Area Network Systems: Towards QoS-Aware Health Internet of Things Applications. *Electronics* **2016**, *5*, 38. [[CrossRef](#)]
16. Collina, M.; Bartolucci, M.; Vanelli-Coralli, A.; Corazza, G.E. Internet of Things application layer protocol analysis over error and delay prone links. In Proceedings of the 2014 7th Advanced Satellite Multimedia Systems Conference and the 13th Signal Processing for Space Communications Workshop (ASMS/SPSC), Livorno, Italy, 8–10 September 2014; pp. 398–404.
17. Govindan, K.; Azad, A.P. End-to-end service assurance in IoT MQTT-SN. In Proceedings of the 2015 12th Annual IEEE Consumer Communications and Networking Conference (CCNC), Las Vegas, NV, USA, 9–12 January 2015; pp. 290–296.
18. Mijovic, S.; Shehu, E.; Buratti, C. Comparing application layer protocols for the Internet of Things via experimentation. In Proceedings of the 2016 IEEE 2nd International Forum on Research and Technologies for Society and Industry Leveraging a better tomorrow (RTSI), Bologna, Italy, 7–9 September 2016; pp. 1–5.
19. Forsstrom, S.; Jennehag, U. A performance and cost evaluation of combining OPC-UA and Microsoft Azure IoT Hub into an industrial Internet-of-Things system. In Proceedings of the 2017 Global Internet of Things Summit (GloTS), Geneva, Switzerland, 6–9 June 2017.
20. Pereira, C.; Pinto, A.; Ferreira, D.; Aguiar, A. Experimental Characterization of Mobile IoT Application Latency. *IEEE Internet Things J.* **2017**, *4*, 1082–1094. [[CrossRef](#)]
21. He, Y.; Faloutsos, M.; Krishnamurthy, S.; Huffaker, B. On routing asymmetry in the Internet. In Proceedings of the GLOBECOM'05 IEEE Global Telecommunications Conference 2005, St. Louis, MO, USA, 28 November–2 December 2005.
22. Rinaldi, S.; Della Giustina, D.; Ferrari, P.; Flammini, A.; Sisinni, E. Time synchronization over heterogeneous network for smart grid application: Design and characterization of a real case. *Ad Hoc Netw.* **2016**, *50*, 41–45. [[CrossRef](#)]
23. Sherman, J.A.; Levine, J. Usage Analysis of the NIST Internet Time Service. *J. Res. Nat. Inst. Stand. Technol.* **2016**, *121*, 33–46. [[CrossRef](#)]
24. Ferrari, P.; Flammini, A.; Marioli, D.; Taroni, A.; Venturini, F. Experimental analysis to estimate jitter in PROFINET IO Class 1 networks. In Proceedings of the 2006 IEEE Conference on Emerging Technologies and Factory Automation (ETFA), Prague, Czech Republic, 20–22 September 2006; pp. 429–432.
25. Ferrari, P.; Sisinni, E.; Brandao, D.; Rocha, M. Evaluation of communication latency in industrial IoT applications. In Proceedings of the 2017 IEEE International Workshop on Measurements and Networking (M&N), Naples, Italy, 27–29 September 2017; pp. 17–22.
26. Persico, V.; Botta, A.; Marchetta, P.; Montieri, A.; Pescapé, A. On the performance of the wide-area networks interconnecting public-cloud datacenters around the globe. *Comput. Networks* **2017**, *112*, 67–83. [[CrossRef](#)]
27. Ferrari, P.; Flammini, A.; Sisinni, E.; Rinaldi, S.; Brandão, D.; Rocha, M.S. Delay Estimation of Industrial IoT Applications Based on Messaging Protocols. *IEEE Trans. Instrum. Meas.* **2018**, *PP*, 1–12. [[CrossRef](#)]

