ACTA IMEKO
ISSN: 2221-870X
December 2020, Volume 9, Number 4, 39 - 52

Cyber-physical manufacturing systems: An architecture for sensor integration, production line simulation and cloud services

Mariorosario Prist[1], Andrea Monteriú[1], Emanuele Pallotta[1], Paolo Cicconi[2], Alessandro Freddi[1], Federico Giuggioloni[3], Eduard Caizer[3], Carlo Verdini[3], Sauro Longhi[1]

[1] *Department of Information Engineering, Università Politecnica delle Marche, Via Brecce Bianche, 60131 Ancona, Italy*
[2] *Department of Industrial Engineering and Mathematical Science, Università Politecnica delle Marche, Via Brecce Bianche, 60131 Ancona, Italy*
[3] *Syncode S.c.ar.l., Spin-off Università Politecnica delle Marche, Via Brecce Bianche, 60131 Ancona, Italy*

ABSTRACT
The pillars of Industry 4.0 require the integration of a modern smart factory, data storage in the Cloud, access to the Cloud for data analytics, and information sharing at the software level for simulation and hardware-in-the-loop (HIL) capabilities. The resulting cyber-physical system (CPS) is often termed the cyber-physical manufacturing system, and it has become crucial to cope with this increased system complexity and to attain the desired performances. However, since a great number of old production systems are based on monolithic architectures with limited external communication ports and reduced local computational capabilities, it is difficult to ensure such production lines are compliant with the Industry 4.0 pillars. A wireless sensor network is one solution for the smart connection of a production line to a CPS elaborating data through cloud computing. The scope of this research work lies in developing a modular software architecture based on the open service gateway initiative framework, which is able to seamlessly integrate both hardware and software wireless sensors, send data into the Cloud for further data analysis and enable both HIL and cloud computing capabilities. The CPS architecture was initially tested using HIL tools before it was deployed within a real manufacturing line for data collection and analysis over a period of two months.

Section: RESEARCH PAPER

Keywords: Industry 4.0; smart factory; OSGi framework; cyber physical system; wireless sensor networks

Citation: Mariorosario Prist, Andrea Monteriù, Emanuele Pallotta, Paolo Cicconi, Alessandro Freddi, Federico Giuggioloni, Eduard Caizer, Carlo Verdini, Sauro Longhi, Cyber-Physical Manufacturing Systems: an Architecture for Sensors Integration, Production Line Simulation and Cloud Services, Acta IMEKO, vol. 9, no. 4, article 6, December 2020, identifier: IMEKO-ACTA-09 (2020)-04-06

Section Editor: Leopoldo Angrisani, University of Naples Federico II, Italy

Received: October 30, 2019; **In final form:** January 17, 2020; **Published:** December 2020

Corresponding author: Andrea Monteriù, e-mail: a.monteriu@staff.univpm.it

1. INTRODUCTION

In the Industry 4.0 era, smart factories are powered by cyber-physical systems (CPSs) and cloud computing (CC) technologies. CPSs using CC technologies enable smart objects (wireless or cabled) and software modules to both communicate and interact with each other. The European H2020 research agenda considers CPSs to be 'the next generation of embedded Information and Communication Technology (ICT) systems that are interconnected and collaborating, providing citizens and businesses with a wide range of innovative applications and services' [1]. CPSs applied to the fields of manufacturing or production are often referred to as cyber-physical manufacturing systems (CPMSs) or cyber-physical production systems (CPPSs) [2] and are expected to increase efficiency, precision, and performance as well as bring unprecedented flexibility to the industrial manufacturing process.

Today, the automated manufacturing facilities are commonly based on hardware solutions from different vendors. This often leads to a heterogeneous and inconsistent mix of automation technologies, each one addressing one or more specific automation problems within the facility. While each of these single systems can acquire and transmit data, they are generally not designed to easily convey the available information to other production lines or manufacturing systems located within the same facilities or in another place, state or country. Indeed, most of the actual automation systems are based on monolithic architectures with very limited communication ports and reduced local computational capabilities, which make it difficult to acquire data for further data processing. In recent years, the use of a wireless sensor network (WSN) as a bridge between the automation technologies at the production level and the CC infrastructure for data analysis [3] (i.e. control, coordination, supervision and management) has become common practice and has even been extended to different environments (e.g. smart grids, smart homes, etc.) [4]-[11]. Consequently, the market is seeking optimised solutions based on industrial WSNs to create parallel backbones to the programmable logic controller (PLC) infrastructure in order to collect valuable data and send them to the Cloud for further processing [12].

For this reason, CPMSs have to be modular, dynamic, flexible, networked, and large-scaled. They are also increasingly based on software, which has become the largest and most complex aspect [13]. Meanwhile, the software development environment needs to be somewhat different from that of standard embedded software given that CPSs integrate computational capabilities and networks in addition to physical processes controlled by embedded computers. CPSs provide abstractions and modelling, design, and analysis techniques for the integrated whole [14]. The interaction and data exchange between computers and physical systems requires new architectures and innovative design approaches [15]. In addition, due to the increased system complexity, it has become crucial to model and simulate the physical system, or part of it, to verify the new algorithms and to reduce the development and production times. Finally, there is the cloud infrastructure. Cloud solutions provide a powerful platform for harmonising incompatible connected devices. At the factory level, gateways are used to provide an intermediate layer between the proprietary protocols integrated in many automation systems and the standard internet protocols that are required to access the Cloud. At the cloud level, different solutions are available for realising large-scale widespread software systems that elaborate data from different sources to control and manage real systems. Thus, the CPS-related challenges faced within the Industry 4.0 paradigm can be summarised as follows [16]-[18]: a connected supply chain (CSC) for real-time data analysis, new modular and dynamic software architectures, the capacity for testing software and hardware before actual deployment via simulation-in-the-loop (SIL) and hardware-in-the-loop (HIL) systems, WSNs and data integration at the cloud level.

In this paper, we focus on a CPPS-related case study based on a real production line in order to address the challenging CPS complexity. Specifically, we present a new software architecture together with a CC layer. The Cooja WSN simulator is used to simulate the data flow from a real production line, with the simulator integrated into a more complex system composed of a modular and extensible and interoperable software for managing heterogeneous devices based on the open service gateway initiative (OSGi) framework. The cloud level is used to manage and analyse the data as well as to expose business data and services via an external interface. This proposed solution has various advantages. On one side, it allows for verifying the correctness of the WSN design using different simulated scenarios, while on the other, it allows for executing a cloud architecture stress test prior to commercial deployment.

The remainder of the paper is organised as follows. Section 2 reviews the relevant works in the existing literature before section 3 outlines the proposed architecture and how each part is integrated within the system as a whole. The OSGi framework and the developed software modules for managing heterogeneous types of sensors, both real and simulated, are then described in section 4 before the cloud architecture is outlined in section 5, with a focus on data management and performance. The WSN for industrial appliances simulated in the Cooja simulator and the gateway model for exchanging data with the Cloud are then explained in Section 6. The preliminary experimental tests and results are reported in section 7 before the paper is concluded with remarks and possible future works in section 8.

2. RELATED WORKS

There exist numerous research papers on CPS modelling techniques. Here, a great deal of effort has been put into constructing compact and accurate mathematical models for complex processes, such as in [19]-[22]. In [23], the authors define a cyber-physical production system architecture, and a virtual factory is modelled and simulated using ETRI CPS model language. The parallel development of computer science (CS) and information and communication technology on one side and of manufacturing science and technology on the other are described in [24], where the authors note the convergence of two worlds, that is, the virtual and the physical realms, in the field of manufacturing. A unified five-level architecture, namely, 5C architecture, is proposed by [25] as a guideline for the implementation of CPPS for manufacturing applications. Meanwhile, the current status, the expectations, the advancement of CPSs in manufacturing and the related R&D challenges are described in [26]. However, only a small number of works demonstrate a simulation or a hybrid architecture of a CPS oriented to Industry 4.0, namely, CPMS. For example, in [27], the authors focus on a collaborative function dimension and describe a list of CPS challenges (e.g. the CPS architecture of the Towers of Hanoi). Finally, in [28], the authors describe an algorithm for managing the energy consumption in autonomous electric vehicles.

Generally, the state-of-the-art research related to CPS applications deals with two different approaches, that is, top-down or bottom-up approaches to aspects related to the flow of control and monitoring [29]. Top-down applications are concerned with the processes that are executable at a high level [30]. However, while they are widely adopted in the world of industry, the real industrial processes are bottom-up since they are concerned with hard-programmed and low-level devices [31]. An example of a bottom-up model was proposed in [32], with both event-driven and service-based models considered in the authors' architectures. The early developments in Industry 4.0 were based on top-down architectures, where the devices notified the events but the process control was defined at a high level [33]. Moreover, in the initial stages, CSP applications were also considered as part of a computational [34] integration for closing the feedback loop of physical process. Nowadays, the implementation of an agent-oriented framework has led to a shift in the views on control loops [35]. An interesting bottom-up architecture was proposed in [36], where the authors also considered the hardware components such as the WSN, the storage units, and the computing system. However, the interoperability and the integration level must be improved to ensure a better and reconfigurable connection between each hardware component. A bottom-up service-based CPS architecture was proposed in [37], where the traditional architecture, composed of sensors, a physical platform, and a control system was combined with an internet service framework to manage the control tasks. While the two systems are fully integrated, the architecture does not provide an efficient and flexible solution that meets Industry 4.0 standards. Another CPS architecture that follows a bottom-up approach is the EuroCPS that was the result of a European project [38]. Here, the architecture is based on a networked collection of hardware platforms used to acquire information in every context (factory, agriculture, smart home, etc.). While the project is Industry-4.0-ready, it lacks the interoperability to integrate the existing industrial commercial solutions and the flexibility to use custom machine learning algorithms for each monitored production line. Meanwhile, the top-down approach has been followed within the development of ISO-IEC architecture, which was proposed by the ISO organisation and is based on a previously realised internet of things architecture [39]. The innovation here lies in the integration of the internet-service-oriented architecture with the virtualisation; however, the proposed solution is not low-level independent and the prevalent focus is on the business level.

Even if CPS applications would appear to be replacing older supervisory control systems using bottom-up models, the existing applications reveal certain limitations related to the interoperability and modularity. While a top-down model has several limitations regarding the communication with single devices, bottom-up models can prove difficult to implement due to the different developmental interface, language, and protocols related to the sensors and each bottom-unit. Within this context, a tentative solution is provided by the reference architecture model for Industry 4.0 (RAMI 4.0) [40]. However, even if the RAMI 4.0 is used in different industrial contexts, it adopts the UPC UA standard to communicate with the production lines, while most production lines do not currently use this standard.

In the present paper, we present a new software architecture composed of a modular and extensible and interoperable software for managing heterogeneous devices based on the OSGi framework along with a CC layer. The proposed architecture is not aimed at defining an Industry 4.0 standard but at demonstrating how the integration of the OSGI framework within Industry 4.0 enables the system to be interoperable, extensible and flexible. As a solution, we analyse the employment of a gateway module for enhancing the interoperability and modularity following the OSGi approach.

3. ARCHITECTURAL APPROACH

This section provides a preliminary description of the modules comprising the proposed architecture for CPS, which are as follows:

- Physical sensor network at the production line level
- Gateway
- CC

The proposed architecture can monitor a production line using a physical sensor network. The gateway sends data to the CC module, which includes features and functions for data storage, data management, analysis, simulation, and visualisation (Figure 1). The proposed architecture does not perform direct control actions on the production lines; rather, it allows for performing monitoring tasks that can be used at the supervisory level or for the decision support systems.

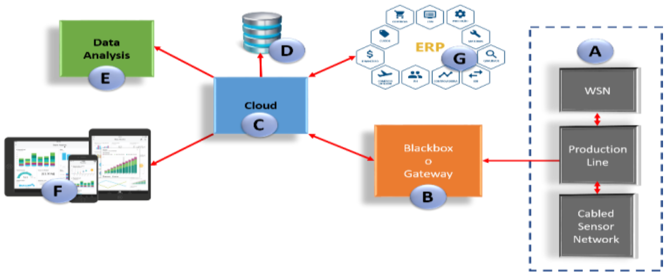


Figure 1. Block diagram of the proposed architecture.

It should be noted that the information provided by this architecture may be also used at the control level but not in the low-level control field, where stability, reliability and synchronism are of utmost importance; these aspects are not the focus of our paper, and are well described in the literature related to sensor networks, as in [41]. The last module, labelled API tools, pertains to the collection of programming procedures and algorithms for providing the access and connection to CC, enterprise resource planning, material requirements planning, and other enterprise repositories, including the gateway, wireless sensors, and front-end applications such as apps and web-apps used for data visualisation [42], [43]. In the remainder of this section, we describe the modules of the architecture while highlighting the main characteristics they should possess but without a detailed description of their implementation (which are provided in sections 4 and 5). Here, whenever the term 'CPS' is used, it refers to 'a CPS compliant with the proposed architecture'. Meanwhile, each of the following subsections first provide a description of the modules at the architecture level before providing a more technical overview of the actual implementation.

3.1. Physical sensor network at the production line level

The production line is the physical system that is monitored by a physical sensor network. The proposed CPS system aims to integrate the physical system into a virtual environment. Here, data are gathered by the physical sensor network connected to the physical system. Data acquisition can be performed in two different ways, the first of which involves the use of a PLC interface with industrial protocols, while the second involves the implementation of a WSN, which can be directly connected to the CC via the gateway. The approach proposed in this work provides both modalities, and the format of the file used for data exchange is the same in both modalities. This flexibility is required in view of enhancing the installation of a sensor network in different industrial scenarios [44]. In fact, in some cases, if the employed PLC does not have external communication ports for data exchange, plug-in wireless sensors are suitable for the implementation of a CPS system. A WSN is a simple solution in many cases where it is difficult to add new sensors to the electrical layout of a physical system, such as a production line. In fact, while the use of a WSN is suitable for the integration of physical sensors in the CC module, the remote control of the production line, after closing the feedback provided by data analysis in the Cloud, requires connected machineries or PLC interfaces, an aspect that is not considered in this paper.

The implementation of the architecture presented in this paper considers a production line monitored by a WSN, and a gateway performing the bidirectional connection from/to the CC module. A WSN simulator was implemented to test the proposed architecture at the cloud and gateway levels prior to actual deployment in a production line. Since the virtual nodes are simulated with the same firmware and properties as the physical nodes, the WSN simulator can also be used to test the sensor network itself before physical deployment. A physical gateway was employed in the architecture for sharing data between the simulated network and the CC. However, this aspect requires the gateway to be developed in a flexible and modular way to perform a re-configurable cloud connection, involving both physical and virtual nodes in the loop, as detailed in the following section.

3.2. Gateway

The gateway (GW) is the embedded device that can connect the PLC interface with the CC. The PLC–GW communication is provided by drivers based on the involved specific protocols, such as the process field bus (PROFIBUS), process field network (PROFINET), Fieldbus Foundation, Modbus, highway addressable remote transducer, controller area network, ethernet for control automation, and the industrial ethernet [45], [46]. Meanwhile, the CC is based on internet connection through a wireless network, ethernet, 3G, or LTE, etc. In this case, the protocols are based on the involved architecture, which includes representational state transfer or message queue telemetry transport [47]. However, the next generation PLC can also send log files via file transfer protocol, email, or text message, etc.

A modular programming approach of the GW system based on the OSGi framework is proposed in our architecture (section 4), which allows for the integration of different types of sensors to be rapidly implemented. In fact, the connection with a new sensor can be performed through the addition of programming modules for data reading and for sending instructions. The modular programming of the gateway based on the OSGi framework also allows for simulated models to be integrated into a GW system. The integration of simulations and simulated data is suitable in the early phase of the system learning, allowing for performing the implemented algorithms prior to installation in the physical production line.

3.3. Cloud computing

CC is a necessary information technology (IT) instrument for the digitalisation of Industry 4.0 factories [48]. The concept of CC includes the storage of data in an ‘external’ repository and services related to data analysis, simulation, and visualisation. Therefore, within the context of smart factories, CC allows for digitising the data to be sent from the manufacturing processes, such that they can be stored and accessed by different smart devices [49].

In the proposed architecture, we present an HTTP server that provides all the functionality required by the cloud service (section 5). The server contains the database with all the information on the connected facilities and their own sensor network. In addition, it provides an interface for any client-side application that may be required to manage values that have been stored over time. Due to the complex queries on large amounts of data, a high level of performance is needed and a database that allows for efficient handling of time series data has thus been used [50].

4. GATEWAY

The GW connects the production line to the cloud services by considering each device, be it physical or virtual, as a software module via the OSGi framework.

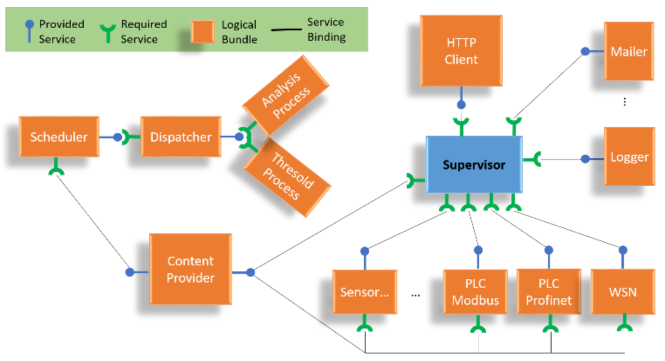


Figure 2: Bundles architecture.

4.1. The OSGi framework

The non-profit OSGi was initiated in 1999 through the union of IBM, Oracle, Philips, Sun, etc., with the aim of standardising the approach to modular application development in the field of home and industrial automation. Today, the OSGi technology has been used in vehicles, the final generation of mobiles, and in software, both in terms of desktops and servers [51]. From the programming point of view, the OSGi present a modular system for Java language, which defines the modality for creating modules and how they mutually interact during runtime [52]. The OSGi architecture is composed of the OSGi framework and a set of bundles. The OSGi implements a centralised service-oriented architecture with loosely coupled services.

The advantages of using the OSGi framework include platform and application independence, multiple service support, dynamic updates, service collaboration support, and security, with more details on this provided in [53].

4.2. Implementation details

The application developed with the OSGi framework is based on an extensible core to manage the dataflow related to multiple bundles that cooperate via the use of various services. The proposed approach provides a supervisor bundle and a collection of bundles that are related to sensors, devices, database management, data analysis, cloud communication and notification mechanisms. The supervisor bundle represents the active extensible core to integrate the sensors and notification mechanisms into the software platform. This core is also used for the management of security events related to the sensors, while it can also run notification events such as messages to the administrator, where threats are logged into the software platform. Specifically, notification events are related to emails, push and text messages, alarms, etc. The management of the supervisor bundle can be also performed using a web-access that provides tools for local and remote bundle administration. It is important to note that while a message notification associated with specific events is logically sent to the final users, it will be physically sent via a cloud infrastructure. The GW creates the message with a specific payload and an alert tag that is interpreted by a dedicated cloud function. Figure 2 shows the proposed bundle architecture, where sensors and devices are involved in the monitoring of any security-related events. In the proposed scenario, several combinations of sensors and devices are possible. For example, a sensor can be connected to a single device or to many, while it can be active, directly sending data to the supervisor, or passive, requiring the polling of the supervisor module for initiating the acquisition. The local database, which is used to save data from the production line sensors before transmission to the Cloud, consists of both static and dynamic tables. Static tables include the collection of every device involved in the system, while dynamic tables contain data and statistics. The information stored in static tables includes the device's ID, credential accesses, and statistical indicators. This information is updated when two types of events are notified: a new cloud setting or a new device added to the OSGi framework (in this case, a new bundle is installed in the software). Meanwhile, dynamic tables are updated during runtime.

The approach proposed in this paper provides the direct transmission of the production data to the cloud computing module. Production data contain every variable related to the production process monitored by the sensor network. Therefore, while every raw data is sent and stored in the cloud repository, the statistical information is first elaborated by a local machine before being sent to the CC for further elaboration and analysis. Static tables also include information on the processing time for statistical analysis. Here, an asynchronous scheduler integrated within a dedicated bundle was used to implement this functionality. This scheduler can also both schedule commands to run after a given delay and periodically execute the production data processing, which increases the computational efficiency. The number of tasks is directly related to the number of different devices installed within a given production line and to the number of indicators to be monitored. When multiple tasks are needed, or when additional flexibility or capabilities are required, one thread is adopted to manage the execution timer and check the tasks to be run at every second. A thread starts only if there is an indicator to be calculated and, as such, the complexity and the number of threads in the runtime are reduced. In addition, the data processing has to be performed with some consideration of a consistent time interval for the extracted information. For example, while it is important to monitor the power consumption during the day, it is also important to observe the status of the production line in terms of the true performance of the equipment's productivity. In these cases, the task can be executed daily, weekly and monthly in order to verify the correct trend. The proposed bundle architecture for the asynchronous scheduler consists of three levels: scheduler, dispatcher, and threshold. Here, the scheduler module runs a thread when each event is verified, while the dispatcher module decides whether the current event has to be associated with a specific indicator and notifies all subscribed services to run the related data processing. Meanwhile, the threshold module executes the elaboration process.

When the GW starts for the first time, it must be identified within the cloud infrastructure using its authentication key. The cloud login allows for all setting parameters to be downloaded and saved to the local machine. Here, the settings contain information related to any alerts and the devices to be elaborated. The supervisor module manages the login of the CPS. The sequence of all handshake phases such as the GW setting requests are related to the first installation of the software platform within an enterprise's system.

A front-end web-based interface was implemented to perform the registration of the GW related to a production line in the Cloud. The bundles, which manage the external connection with the production line, were tested via a bench test using a real PLC and within a simulated scenario using a WSN as part of an HIL architecture, as detailed in section 3. The development of each bundle was implemented using the Eclipse rich client platform, which is a multi-language software development environment for implementing general purpose applications.

Finally, the hardware employed in the research described in this paper is an Advantech embedded PC. It provides two USB ports (one for the integration of the bench test and one for connecting the external hardware with the Cooja simulator for HIL) and two ethernet ports (one for the internet connection and one to integrate industrial communication protocols such as PROFIBUS or PROFINET). The OSGi framework, the attendant bundles, and the WSN simulator were implemented in this system.

5. CLOUD COMPUTING ARCHITECTURE

An HTTP server provides all the functionalities required by the cloud service. Here, the communication is handled through HTTP requests that conform to a pre-specified REST application programming interface (API). The cloud service is comprised of various software components that receive, store and analyse the data streams created by the sensors. It hosts the databases that contain the information on the users, associating them with their own sensor network. In addition, it provides an interface for client-side applications that wish to access the data that have been stored over time.

5.1. Origin of the information

The cloud service receives data through a pre-determined REST API. Any external component that complies to the API is able to use the entire storage and analysis pipeline. This means that the data acquired by the underlying system will be treated the same, regardless of whether the system is real or simulated, enabling usage of the system by both real and simulated components at the same time. The behaviour of the cloud service is consistent for both types of component, provided the API is respected. However the data is acquired, it is ignored by the service, as it is not important for the following steps.

5.2. Forwarding

Once the data has been acquired, a complete HTTP POST request can be built and executed with the data as the payload. The server is always 'listening' for these requests, waiting to execute its pipeline on the incoming data. The server also stores all the information regarding the users of the system and their devices inside a MySQL database. The information will be used to handle each request according to the user and the device that produced the event, to decide, for example, where to store the newly received value. Due to the geographical replication of the database, the system becomes scalable, as the requests can be offloaded to the server that is nearest to the sender.

5.3. Persistent storage

All the incoming values will have an associated timestamp. In fact, each sensor will generate a constant stream of values and timestamps, which can be referred to as time series data. For this reason, a database that allows efficient handling of time series data is required, one that also maintains a high level performance while executing complex queries on large amounts of data, which arises especially when the sensors operate at a high sampling frequency.

Moreover, the database must be intelligent enough to cope with large amounts of constant data, which are typically generated by binary or low-resolution sensors that monitor processes operating at a steady state. MongoDB and Cassandra are the most likely candidates here, largely due to their key-value

Table 1: Comparison between Cassandra and MongoDB [54].

Name	Cassandra	MongoDB
Description	Wide-column store based on ideas of BigTable and DynamoDB	One of the most popular document stores available as a fully managed cloud service or for deployment on self-managed infrastructure
Primary Database Model	Wide column store	Document store
Secondary database models		Search engine
License	Open source	Open source
Implementation language	Java	C++
Data scheme	Schema-free	Schema-free
Secondary indexes	Restricted	Yes
SQL	SQL-like SELECT, DML and DDL statements (CQL)	Read-only SQL queries via the MongoDB Connector for BI
APIs and other access methods	Proprietary protocol Thrift	Proprietary protocol using JSON
Server-side script	No	JavaScript
Triggers	Yes	Yes
Partitioning methods	Sharding	Sharding
Replication methods	Selectable replication factor	Master-slave replication
MapReduce	Yes	Yes
Consistency concepts	Eventual Consistency Immediate Consistency	Eventual Consistency Immediate Consistency
Foreign Keys	No	No
Transaction concepts	No	Multi-document ACID Transactions with snapshot isolation
Concurrency	Yes	Yes
Durability	Yes	Yes
In-memory capabilities	No	Yes

nature, which ensures empty values are not stored on the disk. Repeating values can then be mapped onto the empty values to save huge amounts of memory. As shown in Table 1, while both are schema-free noSQL databases, they differ in terms of the method they use to physically store the data. MongoDB is a document store database, storing all data in JSON documents on the disk, while Cassandra is a wide column store, which means that any values with the same partition key are stored on the same row. Here, Cassandra was chosen over MongoDB after comparing the scalability of the two systems. In fact, MongoDB is more efficient with small amounts of data, while Cassandra outperforms it when large datasets are involved [55]. Since handling large amounts of data is vital to the system, Cassandra was clearly the best option. In addition, Cassandra does not follow a master-slave type architecture, making the addition of a new node as simple as creating and connecting it to the rest of the nodes. As the data is replicated across the nodes based on Cassandra's configuration, it is also less likely that any catastrophic failure could result in the total loss of data.

5.4. Primary keys in Cassandra

Unlike many other database management systems, Cassandra uses the primary key, which is composed of partition and cluster keys, to decide how to physically store the data. In fact, in selecting the timestamp as a clustering key, the data will be physically ordered on the storage based on the date the value was received. This ensures any query requesting the latest available is deal with rapidly, even when huge amounts of data are involved.

To further improve the speed of more complex queries, a partition key that separates each day's worth of data was used, which ensures even time range queries are scalable, regardless of the size of the database. To verify the effectiveness of the adopted solution, an experimental test was carried out to compare the execution time performance and the memory usage of MongoDB and Cassandra when saving raw data from production lines. We used a droplet model (SSD-based virtual machines) from Digital Ocean with an installed Ubuntu server (14.04 64 bit), 2 GB RAM, 40 GB of hard disk, and two virtual CPUs (vCPU) (a virtual processor, which is a physical central processing unit [CPU] that is assigned to a virtual machine). We assumed the monitoring of 10 production lines with an acquisition time of 1 s and a window acquisition time of 1 h. In this scenario, 3.600 (one production line), 10.800 (three production lines) and 108.000 (thirty production lines) records were collected and sent to the Cloud to update the Cassandra and MongoDB databases. MongoDB version 3.0 and Cassandra version 2.2 were used.

The results are shown in Table 2, where it is highlighted that, with an increased number of records, Cassandra has a more stable performance than MongoDB, with a two-fold improvement, while for small amounts of data MongoDB, has better time performance but consumes a great deal of memory, as shown in Table 3.

5.5. Data visualisation

The REST API provided by the server not only regulates data acquisitions but also enables the development of client-side applications to visualise the values permanently stored on the

Table 2: The calculated time to write key-value pairs.

	3.600	10.800	108.000
Cassandra	1.73 ms	3.51 ms	28.22 ms
MongoDB	1.06 ms	2.89 ms	15.16 ms

Table 3: Memory usage for write operation (MB).

	3.600	10.800	108.000
Cassandra	6.04	8.1	212.16
MongoDB	63.56	281.02	373.36

server. Once the client conforms to the REST APIs provided by the cloud service, it can execute a certain number of queries on the stored values, the results of which will then be visualised through a template that can be chosen by the user themselves.

A limited number of templates is provided for various purposes, such as variable visualisation, comparison with another unrelated variable, or simply the display of the current state of the variable. Every instance of a template connected to a variable is referred to as a 'widget' (see Figure 3). In addition, the client application allows for the definition of the notification rules for each variable, with these rules checked by the server whenever a new value is received. If the notification rule matches the newly received value, the user who created it is alerted via the specified message. In short, the conditions are all at the user's discretion.

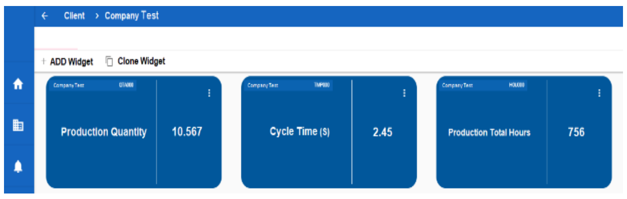
5.6. Data analysis

Two types of analysis are possible in this system: online and offline analysis. To execute online analysis, the algorithms should be constantly running on a dedicated server for each data stream currently in use. For each new value (or window of values) the algorithms will update their state and will generate notifications if needed. For example, one of these algorithms could be tasked with sending an alert whenever a device deviates from its normal behaviour due to a fault, failure or malfunction. These algorithms provide insight into the current state of each device in real time, making it possible to detect problems before they negatively affect the system performance, according to the so-called 'preventive maintenance' paradigm.

Meanwhile, in terms of offline analysis, after the system has been running for a certain period of time, there will be large amounts of data that describe the behaviour of the physical or simulated system. The analysis of this data could provide information on the efficiency of the process as a whole, which can then be used to make better decisions. Due to the nature of the problem, the amount of data to be analysed will be large, which could involve prohibitive amounts of execution time. To resolve this issue, so-called map-reduce methods are required, which improve the performance even when the algorithm needs to go over all the stored data to reach a useful conclusion [56]. The algorithm is run in parallel on each node of the cluster on a different part of the data, and the results are then unified to provide the results of the algorithm as a whole. This is only possible because the chosen database management system provides APIs for user defined map-reduce methods.

6. DATA MANAGEMENT AT PRODUCTION LEVEL: TESTING AND DEBUGGING

A production line can be modelled using a set of inputs and outputs, be they digital or analogue. An embedded PC or PLC is largely involved in a production line in terms of controlling all operations. Generally, the controller of a production line system, which is based on a PLC, reads input data at every fixed time-step and saves them to its memory. The data temporally stored in the memory are elaborated, and the results of the logical or mathematical operations are exported to other memory locations. Following this, another thread reads the values at each



← Client › Company Test

+ ADD Widget Clone Widget

Company Test			
Production Quantity	10.567		
Cycle Time (S)	2.45		
Production Total Hours	756		

Figure 3: Example of a single value widget.

memory location and sets the physical output. The same approach also works if variables are acquired via Fieldbus. However, some PLC devices do not have a communication interface for integration with external applications, while some have proprietary protocols that hinder their integration into software platforms for Industry 4.0. In contrast, many recent PLCs can integrate external applications using standard and known protocols. This scenario continues to characterise the automatic machines market, with old-style machines without any communication interface on one side, and new machines that provide advanced interaction features on the other. As stated in section 3, the presence of a WSN that runs parallel to the cabled sensor network in the production line and is connected to the GW (and thus to the Cloud), opens up new HIL scenarios. The proposed approach allows data to be acquired from the production line without limitations related to the PLC device employed in the system.

Once the specifications are available for the specific industrial process, then the WSN must be designed in view of acquiring the variables needed to comply with the specifications (e.g. providing certain performance indexes). Then, according to the proposed architecture, the compliant OSGi bundle can be developed, such that the data from the sensor network can be parsed in the GW, while the same applies to software wireless sensor nodes. As such, the information from the production line is stored in a cloud database, where analytics are developed and performed. Finally, according to our proposed architecture, the information that is compliant with the specifications is sent back to the production line, where it can be used for decision support and/or automated supervision. As is clear, our architecture is both system independent and flexible, thus allowing for dealing with different industrial processes with the same functional blocks. Moreover, testing and debugging at both the cloud communication and the GW system levels can be enabled prior to deployment.

At the cloud communication level, a software module has been developed to generate random data packets that are sent/received for the testing of the cloud communication. This test is necessary for evaluating the communication performance, including in terms of response time. At the GW level, a WSN simulator has been implemented for testing and debugging the system behaviour in cases such as delay and loss of information. The proposed test case uses the Cooja simulator to reproduce the behaviour of a WSN with different nodes. In this test, the production line and its outputs are simulated by the WSN (software part), and the GW and the cloud communication (hardware part) are tested prior to connection to the production line within a HIL scenario. It should be noted that in the simulated WSN, virtual nodes are simulated with the same firmware and properties of the physical nodes, which means it can be used to test the sensor network prior to actual deployment on a real production line. The schemes for the two tests are presented in Figure 4.

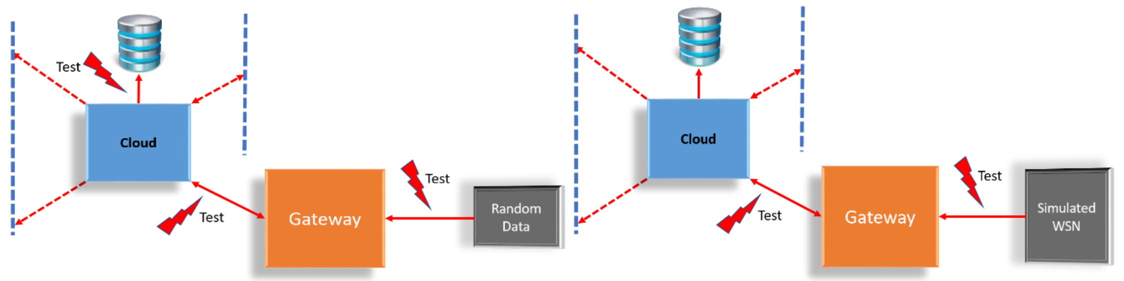


Figure 4: The two simulated scenarios.
(a) Left: Testing and debugging of cloud communication and database performances using random data.
(b) Right: Testing and debugging of cloud communication and database performances using WSN.

6.1. Testing and debugging of the cloud communication

This section describes the first level of simulation, which relates to the generation of data to be sent to the CC for the application testing and debugging. Specifically, the OSGi bundle is proposed and analysed in view of describing the communication formalism. The proposed module creates a background service application that initially connects to the Cloud using the login access to download the configuration file-setting and the variable representation. This service application generates random data for saving on a memory buffer. A synchronised service-thread can read data from the same buffer and format them into a JSON data packet that will be sent to the CC. As is described in the next sections, this JSON packet contains information related to the production line, the reference to the PLC memory location, the acquisition timestamp, and the set of each reading. Each data can be represented by the same variable type used in PLC programming, such as word (2 byte), double word (4 byte), byte, and Int e Char, while for the float type, a custom interpretation is used where the data is separated into two parts (e.g. the type F5.3 is formed by five digits for the integer part and three for the decimal part).

Using the proposed data simulator, it is possible to simulate the interconnection of many production lines to the CC system. This approach also allows for testing and debugging the cloud features prior to the development of a physical production line. The performance of the CC in terms of response time, elaboration time, and data saving can be evaluated and compared with different database solutions such as MongoDB and Cassandra.

6.2. Testing and debugging of the gateway system

In this section, we describe the testing and debugging of the gateway system using a working example involving a small-scale CPS prototype.

A Cooja wireless sensor network simulator was used in the CPS prototype, while the system as a whole is composed of the following:

1) OSGi software bundles to manage the connection with the simulated WSN, as described in the previous section.
2) A GW (PC embedded) used to interconnect the production line with the Cloud and other machines (see Section 4).
3) A WSN simulator, namely, Cooja, and 10 simulated nodes that exchange data using the collect tree protocol (CTP):
    a. five simulated nodes used as a repeater or wireless extender,
    b. four simulated nodes with HIL (interface with real I/O),
    c. one simulated node used as a base-station,
    d. an HIL bench test.

This scenario is represented in Figure 5, which shows the relationship between the bundles running in the OSGi framework and the simulated infrastructure based on Cooja and the HIL model. The WSN simulation was implemented using a Cooja cross simulator, which is an emulator of the programming code of the real nodes of a physical sensor network [57]. The Cooja simulator allows certain features, such as the transmission and connections, to be evaluated using virtual experiments. This activity is suitable for the debugging of the entire firmware since it allows for any possible problems and bottle necks to be identified prior to the release of the firmware and the hardware deployment. Sky motes were used as nodes to reproduce a sensor network in the proposed simulation. The Advanced Sky GUI plugin, an extension of the Contiki Cooja for sky motes [58], was modified and used.

This new plugin includes additional features such as the option to save I/O messages into a log file in addition to the standard functionalities (user button, LEDs pin, serial communication, eight ADC ports and a virtual joystick). This feature is necessary to reproduce the behaviour of the base-station node when it saves a radio-message from the nodes to a log file. The Cooja simulation workflow also includes the OSGi bundle, which reads this log file and sends data to the cloud computing system.

Figure 6 shows the WSN designed using the Cooja-based simulator. This network includes 10 sky motes as wireless sensor nodes. The communication protocol is based on a stable version of CTP, which provides a datagram routing layer used to gather data and the tree routing for the higher level routing [59] and the transport protocols [60], [61]. Each node can be a candidate for the base station that initiates the data collection. Every analysed configuration considers 1-node as the reference node of the

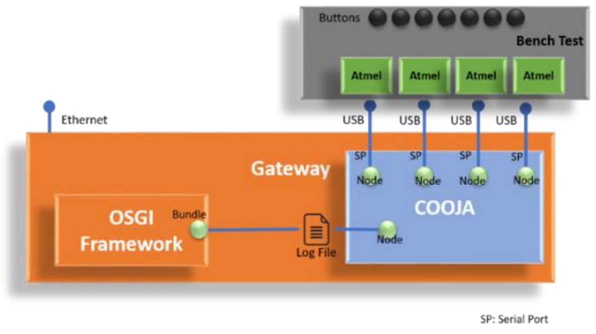
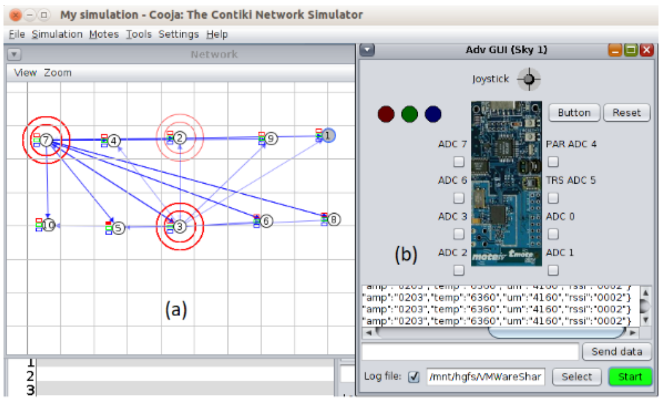


Figure 5: Simulation test architecture.



```
"amp":"0203","temp":"6360","um":"4160","rssi":"0002"}
"amp":"0203","temp":"6360","um":"4160","rssi":"0002"}
"amp":"0203","temp":"6360","um":"4160","rssi":"0002"}
```

Log file: [✓] /mnt/hgfs/VMWareShar

Figure 6: The Cooja cross simulator. The simulated WSN and the new version of the Advanced Sky GUI Plugin.

WSN network, where the proposed plugin is run to save the data collected from other nodes on a file. The log file content is already formatted following the JSON format; thus, the OSGi bundle can directly send data to the CC module for further analysis. The data described above represents the states of a production line.

7. PILOT CASE AND PRELIMINARY RESULTS

As described in section 6, preliminary testing and debugging at both the cloud communication and GW system levels can be performed using simulated data. These data can be generated randomly (cloud communication test) or via a WSN simulator (GW test). In the latter case, the WSN can also be used to test the network capabilities prior to implementation in a real production line. In this section, the focus is on the pilot case of a real industrial automation system. Various tests were conducted to demonstrate the feasibility of the architecture in terms of the functionality of the OSGi framework, correct storage of information into the Cloud, and the definition of services through data analysis techniques.

7.1. Pilot Case

CTF AUTOMAZIONI of Matelica (Italy) – an Italian company leader in the design and construction of special machines, robotised production lines and automatic assembly islands – was monitored for a period of two months via the architecture proposed in this paper. The main focus was on the automatic line for the production of metal filters (aluminium, stainless steel, etc.) designed for SIFIM Srl, an Italian company leader specialised in the production of metallic filters and in the realisation of quality aesthetic metallic components for household and industry and community kitchen appliances (restaurants, hotels, canteens, etc.).

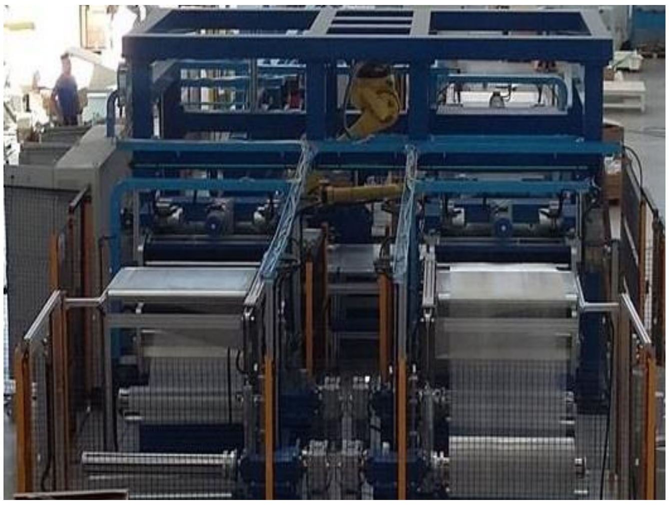


Figure 7. SIFIM’s production line.

In terms of the production line, the machine used as a pilot case produces metal filters for the domestic appliance sector (see Figure 7).

The SIFIM production line can be divided into five functional blocks associated with the five production phases, as shown in Figure 8:

A. The decoilers on which the metal coils are positioned will be used to assemble three cutters, which will process the sheets with a maximum dimension of 1100 × 1100 mm^2.
B. The robot will pick up the cut sheets and place them on a table, creating a stack of 13 sheets.
C. A manipulator will take over the stack and will secure the fourth angular cutter that will scrape the 1100 × 1100 m^2 sheets in the required format.

Table 4: Communication settings: Standard format.

Start	Length	Type	Measure	Key	Description
0	2	*W*	-	*CUST00*	Client Code
2	2	*W*	-	*LIN000*	Code Line
24	20	*C*	-	*CODPROD*	Product Code
44	4	*D*	1/10 s	*TMPCOD*	Cycle Time
48	1	*B*	-	*START*	Run
49	1	*B*	-	*STOP*	Stop
50	1	*B*	-	*MNL*	Manual
51	1	*B*	-	*EM000*	Emergency
52	2	*W*	-	*AL000*	Alarm Code
54	2	*W*	-	*NPZOK*	Good Items
56	2	*W*	-	*NPZKO*	Discarded Items
60	4	*D*	Cycles	*BA_001*	Motor Siemens 1FK7105-2AF71-1QA0
76	4	*D*	Cycles	*BA_002*	Cylinder SMC CDQ2B40-100DZ
…	…	…	…	….	

D. The sheets processed by the cutter will be positioned on a linear axis made up of several trolleys that will be used to assemble the series of presses that will make the bevel on all sides of the filter mat, including on the part reserved for the handle.
E. Finally, a 3-axis manipulator will stack the finished product on an empty pallet which, once filled, is unloaded onto the discharge strip for the finished product.

The environment and the production line structure presented the following integration problems:

- Presence in the production area of an ethernet connection.
- The production line has to mount a pure PLC and not a panel PC. Generally, a panel PC is an embedded PC with a touch panel that integrates PLC runtime and visual software. In this case, to avoid interference between PLC runtime and GW client, a separated solution was adopted.
- Integration with a Siemens PLC S7 series. This type of PLC was selected since it is one of the most commonly used PLCs in production lines, while the PLC used for the demo (Panasonic) is essentially installed for small applications; serial RS232 communication is not the most stable bus solution for the transmission of data in an automation context.
- Availability to integrate the gateway on the electrical panel near to the PLC.
- Different type of sensors.
- Process identified by the state machine, meaning it is possible to match the working state and the process variables.
- Process stability. The production line has to work in production mode and not in test mode.
- Fixed work shifts. The evaluation of performance indexes starts from the definition of the range times in which the machine should work.

The main objectives to be achieved during the test stage were:

- Monitoring the production line state.
- Integration of new bundles in the OSGi framework and a new communication protocol in a plug and play mode.
- Data transmission policy (acquisition from PLC and sending to the Cloud rate).

The main variables for estimation during the monitoring stage for performance index calculation and maintenance support were as follows: cycle time, average cycle time expected, product code, start, stop, alarm, emergency, good object, discard object, principal actuators installed (motors, pumps, etc.).

In order to standardise the settings used by the bundle to access the production lines memory block for data reading, a standard format was used as shown in Table 4.

In terms of software integration, this stage required developing a new OSGi bundle that can, on the one hand, communicate with the coordinator bundle and, on the other, manage the communication between the Siemens PLC and the gateway. The objective was to validate the concept of extensibility, flexibility and interoperability of the developed system and to define a standard for the line configuration process in order to replicate the same solution on other new lines but with different PLCs and communication protocols. The flexibility of the architecture due to the OSGi framework offers the possibility of developing one or more bundles associated with

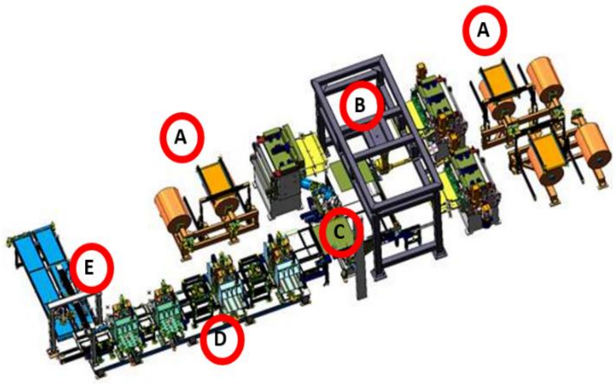


Figure 8. Computer-aided design schema of SIFIM's production line.

the new production line for integration. The developed OSGi bundle differed from that managing the Panasonic PLC in terms of the communication protocol, which, in this case, was the S7comm. The bundle periodically retrieves PLC data, saves them to a buffer and then sends a complete packet containing all the buffer content to the Cloud. The scanning time is defined as 1/3 of the machine's ideal cycle time related to the product code that is in production, while the data transmission time to the Cloud is three times the cycle time. All parameters can be configured from the Cloud and downloaded to the GW during the device authentication phase or when a changing configuration occurs.

7.2. Analytical aspects

The possibility of evaluating the consumption of the production line's critical components, monitoring the performance indicators, diversifying the causes of the downtime via the functional area and systematically identifying where more stop problems are created, allows for a reduction of the losses. In order to analyse the data and the results of the computational analysis, an ad hoc web user interface was developed. Specifically, two main sections were defined, the first of which relates to the efficiency indicators, while the second is used for the management of the component consumption rules. The part associated with the stop causes is directly shown on the overall equipment effectiveness (OEE) web page since a direct visual inspection allows for better identifying the amount of job loss occurring as a result of the stops and for grasping the direct link between the loss trend and the causes. In this first phase, two data analysis algorithms were implemented: one for the analysis of the efficiency indexes and one for the cycle time analysis for the of micro stop calculation.

In terms of efficiency indexes, the analysis consists of quality rate, utilisation level of the equipment, efficiency, availability, and OEE, which are measures of the manufacturing operations performance and productivity, expressed in percentage. The OEE indicates the degree to which a manufacturing plant is truly productive and serves as a general and inclusive measurement of how well a company's manufacturing operations are performing.

Cycle time analysis relates to how, for any manufacturer, the idle time that occurs during production results in a loss of money. Work stoppages reduce the productivity rates and increase the cycle times. Identifying micro stops or reductions in the job speed is one of the most difficult parts of the monitoring process. In order to handle this type of loss, cycle time analysis can be used. In the developed OSGi bundle, an automatic data gathering was implemented to enable the cycle time measurement. By comparing all the real cycles against an ideal cycle time and

Table 5: April-May 2018. Manufacturing line data.

Data Type	Line A
Operating Time	232 h 0 m 0 s
Set Up and Down Time	28 h 41 m 40 s
Planned Down Time	0 h 0 m 0 s
Net Operating Time	203 h 18 m 14 s
Average Cycle Time	0.76 m
Number of Good Units	16148
Rejects	0

through matching the data of the micro stops, a total loss due to speed reduction could be calculated.

7.3. Results

The pilot case was monitored over a period of two months, from April 2018 to May 2018.

During this period, all the macro activities – such as stoppage, efficiency reduction, set-up/adjustments, and quality impairments – and the micro activities – such as the PLC variable state (start, stop, cycle time, etc.) with a resolution of 10 seconds – were stored in the Cloud for analysis. The production analysis consisted of computing and monitoring the following indexes: quality rate (*QR*), efficiency (*E*), availability (*A*) and *OEE*, which are measures of the performance and productivity of the manufacturing operations, expressed in percentage. In brief, the above indexes can be defined as follows [12]:

$$A = \frac{NetOperating\ Time}{Operating\ Time \cdot PlannedDown\ Time} \cdot 100\ \% \quad (1)$$

$$E = \frac{IdealCycle\ Time \cdot TotalCount}{NetOperating\ Time} \cdot 100\ \% \quad (2)$$

$$QR = \frac{GoodCount}{TotalCount} \cdot 100\ \% \quad (3)$$

$$OEE = A \cdot E \cdot QR \quad (4)$$

where *NetOperatingTime* is the time for which the equipment was available for operation, *OperatingTime* is the total calendar period for the OEE calculation, *PlannedDownTime* is the scheduled maintenance time, *IdealCycleTime* is the theoretical minimum time to produce one part, *GoodCount* is the produced parts that meet the quality standard (without reworking), and *TotalCount* is the total of all produced parts (including defects).

The total production output, the losses, and the operating time are listed in Table 5, while the effectiveness indexes and the OEE values are shown in Table 6. Finally, the total number of stoppages and the attendant causes within the total time the production line was stopped were analysed in the Cloud using the cycle time analysis, with the results recorded in Table 7.

The results of the preliminary tests indicated that through using the proposed infrastructure for the management of the data, the Cloud and the physical devices, it is possible, via a simple method, to acquire all the information associated with the production line, including production quality (goods vs. discarded objects), cycle time, start, stop, emergency, alarm and device status (e.g. inverters, motors, etc.). All the data in the Cloud was used to define tailored services for the firm. As an example, the data and signals from the PLC and the operator shifts (see Figure 9) could be merged to provide a consistent job activity, which will be useful for the performance index estimation, while, at the same time, support for the production line maintenance was proposed using the ABC classification to identify the stop causes.

Table 6: Efficiency indexes for April–May 2018.

Efficiency Index	Line A
Actual Availability	87.6 %
Performance Efficiency	82.1 %
Rate of Quality	100.0 %
OEE	72.0 %

Table 7: April-May 2018. Micro Stoppage Analysis.

Data Type	Line A
Robot	989 m
Generic	546 m
Decoiler 4	95 m
Manual	53 m
Decoiler 3	21 m
Decoiler 2	18 m
Decoiler 1	0 m
Total	1680 m

8. CONCLUSION AND FUTURE WORK

The fourth industrial era is taking the automated factory to a whole new level through the introduction of modular, extendible and customised mass production technologies. This means that production lines or machines will be integrated in all management processes to ensure independent operation or cooperation with other machines or humans, with the main objective being to realise and optimise a fully customer-oriented production while constantly maintaining its condition.

Following the paradigm of the new smart factories, the main challenges related to CPS complexity were addressed in this paper through developing both a new modular, extensible and interoperable software architecture based on the OSGi framework for managing heterogeneous devices, simulation and HIL prototype and a CC system for managing and analysing data as well as for exposing business data and services via external APIs interfaces. Specifically, the testing of a smart production line prior to deployment was developed based on two levels of analysis. The first allowed for the development of an OSGi software module for generating random data packets that will be

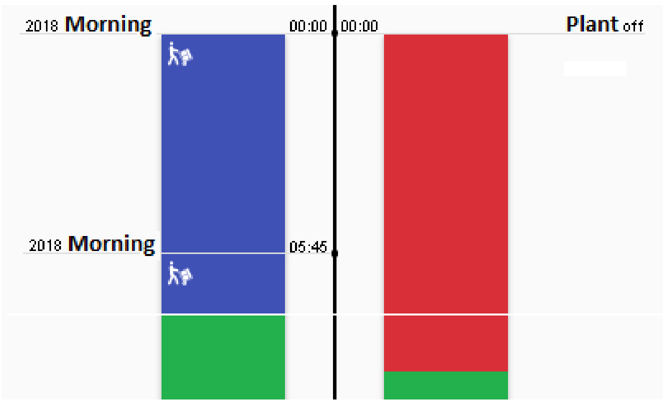


Figure 9. Comparison between operator shift (left) and real production line activity (right).

sent for the testing of the cloud communication system and the database performances. Meanwhile, the second was realised on a simulated WSN using a Cooja cross simulator integrated with an HIL infrastructure. The proposed CPS architecture was tested in terms of a pilot case, the SIFIM Srl production line developed by CTF Automazioni Srl. Here, the performance was evaluated, while high-level cloud analytics were developed to exploit the versatility of the CPS to produce a proactive maintenance scenario. Eight weeks of data, collected during April and May of 2018, were used to undertake the analysis of the production line performances in terms of OEE and machine failure. These analyses will prove useful to improving customer care and maintenance management and for evaluating the structural improvements to new production lines. Further tests and functional integrations will be carried out in the future to enhance the performance of the proposed system.

ACKNOWLEDGEMENTS

Special thanks go to CTF Automazioni Srl for the availability and support in testing the cloud infrastructure. Particular thanks go to the owner, Lucio Bartocci, the project managers, Luigi Carboni and Maurizio Marucci, and, finally, the technical commercial manager, Pierluigi Pettinari. We also thank SIFIM Srl for supporting an important research project and for agreeing to install our developed solution within its production line.

REFERENCES

[1] European Union Horizon 2020, Smart cyber-physical systems. ICT-01-2014. Online [Accessed 29 October 2020]. https://ec.europa.eu/info/funding-tenders/opportunities/portal/screen/opportunities/topic-details/ict-01-2016

[2] Z. Jakovljevic, V. Majstorovic, S. Stojadinovic, S. Zivkovic, N. Gligorijevic, M. Pajic, Cyber-physical manufacturing systems, Proc. of 5th International Conference on Advanced Manufacturing Engineering and Technologies (NEWTECH 2017), Belgrade, Serbia, 5-9 June 2017, pp. 1-14. DOI: https://doi.org/10.1007/978-3-319-56430-2_14

[3] K. Khakpour, M. H. Shenassa, Industrial control using wireless sensor networks, Proc. of 3rd International Conference on Information and Communication Technologies: From Theory to Applications, Damascus, Syria, 7-11 April 2008, pp. 1-5. DOI: https://doi.org/10.1109/ICTTA.2008.4530163

[4] S. Longhi, D. Marzioni, E. Alidori, G. Di Buò, M. Prist, M. Grisostomi, M. Pirro, Solid waste management architecture using wireless sensor network technology, Proc. of 5th International Conference on New Technologies, Mobility and Security, Istanbul, Turkey, 7-10 May 2012, pp. 1-5. DOI: https://doi.org/10.1109/NTMS.2012.6208764

[5] M. Grisostomi, L. Ciabattoni, M. Prist, L. Romeo, G. Ippoliti, S. Longhi, Modular design of a novel wireless sensor node for smart environments, Proc. of 10th International Conference on Mechatronic and Embedded Systems and Applications, Senigallia, Italy, 10-12 September 2014, pp. 1-5. DOI: https://doi.org/10.1109/MESA.2014.6935600

[6] H. Furtado, R. Trobec, Applications of wireless sensors in medicine, Proc. of 34th International Convention MIPRO, Opatija, Croatia, 23-27 May 2011, pp. 257-261. Online [Accessed 07 December 2020] https://ieeexplore.ieee.org/document/5967060/

[7] Z. Iqbal, K. Kim, H. Lee, A cooperative wireless sensor network for indoor industrial monitoring, IEEE Transactions on Industrial Informatics 13 (2017), pp. 482-491. DOI: https://doi.org/10.1109/TII.2016.2613504

[8] X. Ming, D. Yabo, D. Lu, P. Xue, G. Liu, A wireless sensor system for long-term microclimate monitoring in wildland cultural heritage sites, Proc. of the International Symposium on Parallel and Distributed Processing with Applications, Sydney, NSW, Australia, 10-12 December 2008, art. no. 4725151, ISBN 978-076953471-8, pp. 207-214. DOI: https://doi.org/10.1109/ISPA.2008.75

[9] D. Abruzzese, M. Angelaccio, R. Giuliano, L. Miccoli, A. Vari, Monitoring and vibration risk assessment in cultural heritage via wireless sensors networks, Proceedings of the 2nd Conference on Human System Interactions, HIS 2009, Catania, Italy, 21-23 May 2009, art. no. 5091040, ISBN 978-142443960-7, pp. 568-573. DOI: https://doi.org/10.1109/HSI.2009.5091040

[10] F. D'Amato, P. Gamba, E. Goldoni, Monitoring heritage buildings and artworks with wireless sensor networks, Proceedings of the IEEE Workshop on Environmental, Energy, and Structural Monitoring Systems, EESMS 2012, Perugia, Italy, 28 September 2012, art. no. 6348392, ISBN 978-146732737-4, pp. 1-6. DOI: https://doi.org/10.1109 /EESMS.2012.6348392

[11] K. Islam, W. Shen, X. Wang, Wireless sensor network reliability and security in factory automation: A survey, IEEE Transactions on Systems, Man and Cybernetics, Part C: Applications and Reviews 42 (2012), pp. 1243-1256. DOI: https://doi.org/10.1109/TSMCC.2012.2205680

[12] M. Grisostomi, L. Ciabattoni, M. Prist, G. Ippoliti, S. Longhi, Application of a wireless sensor networks and Web2Py architecture for factory line production monitoring, Proc. of 11th IEEE International Multi-Conference on Systems, Signals Devices (SSD14), Castelldefels-Barcelona, Spain, 11-14 February 2014, pp. 1-6. DOI: https://doi.org/10.1109/SSD.2014.6808882

[13] M. Hölzl, A. Rauschmayer, M. Wirsing, Software-intensive systems and new computing paradigms, in: Software-Intensive Systems and New Computing Paradigms: Challenges and Visions, Springer Berlin Heidelberg, Switzerland, 2008, ISBN 978-3-540-89437-7, pp. 1-44.

[14] J. Wan, Y. Hehua, S. Hui, L. Fang, Advances in cyber-physical systems research. KSII Transactions on Internet and Information Systems (TIIS), 9 (2011), pp. 1891-1908. DOI: https://doi.org/10.3837/tiis.2011.11.001

[15] F. Zhang, K. Szwaykowska, W. Wolf, V. Mooney, Task scheduling for control oriented requirements for cyber-physical systems, Proc. Real-Time Systems Symposium, Barcelona, Spain, 30 November-3 December 2008, pp. 47-56. DOI: https://doi.org/10.1109/RTSS.2008.52

[16] F. Shrouf, J. Ordieres, G. Miragliotta, Smart factories in Industry 4.0: A review of the concept and of energy management approached in production based on the internet of things paradigm, Proc. of IEEE International Conference on Industrial Engineering and Engineering Management, Selangor Darul Ehsan, Malaysia, 9-12 December 2014, pp. 697-701. DOI: https://doi.org/10.1109/IEEM.2014.7058728

[17] H. Liang, N. Saraf, Q. Hu, Y. Xue, Assimilation of enterprise systems: The effect of institutional pressures and the mediating role of top management, MIS Quarterly - Society for Information Management and The Management Information Systems Research Center, 31 (2007), pp. 59-87. DOI: https://doi.org/10.2307/25148781

[18] D. Zuehlke, SmartFactory-Towards a factory-of-things, Annual Reviews in Control, 34 (2010), pp. 129-38. DOI: https://doi.org/10.1016/j.arcontrol.2010.02.008

[19] E. Petritoli, F. Leccese, M. Botticelli, S. Pizzuti, F. Pieroni, A RAMS analysis for a precision scale-up configuration of the 'Smart Street' pilot site: An industry 4.0 case study, Acta IMEKO, 8 (2019), pp. 3-11. DOI: https://doi.org/10.21014/acta_imeko.v8i2.614

[20] Y. Xue, P. Bogdan, Constructing compact causal mathematical models for complex dynamics, Proc. of the ACM 8th International Conference on Cyber-Physical Systems, Pittsburgh, PA, USA, 18-21 April 2017, pp. 97-107. DOI: https://doi.org/10.1145/3055004.3055017

[21] P. Bogdan, M. Pajic, P. Pande, V. Raghunathan, Making the internet-of-things a reality: From smart models, sensing and actuation to energy-efficient architectures, Proc. of IEEE International Conference on Hardware/Software Codesign and System Synthesis (CODES+ ISSS), Pittsburgh, PA, USA, 1-8 April 2016, pp. 1–10. Online [Accessed 07 December 2020] https://ieeexplore.ieee.org/document/7750989/

[22] P. Cicconi, A. C. Russo, M. Germani, M. Prist, E. Pallotta, A. Monteriù, Cyber-physical system integration for industry 4.0: Modelling and simulation of an induction heating process for aluminium-steel molds in footwear soles manufacturing, Proc. of IEEE 3rd International Forum on Research and Technologies for Society and Industry (RTSI), Modena, Italy, 11-13 September 2017.
DOI: https://doi.org/10.1109/RTSI.2017.8065972

[23] J. Jeon, S. Kang, I. Chun, CPS-based model-driven approach to smart manufacturing system, Proc. of 5th International Conference on Intelligent Systems and Applications, Barcelona, Spain, 13-17 November 2016, pp. 133-135.

[24] L. Monostori, B. Kádár, T. Bauernhansl, S. Kondoh, S. Kumara, G. Reinhart, O. Sauer, G. Schuh, W. Sihn, K. Ueda, Cyber-physical systems in manufacturing, CIRP Annals 65 (2016), pp. 621-641.
DOI: http://dx.doi.org/10.1016/j.cirp.2016.06.005

[25] J. Lee, B. Bagheri, H. Kao, A cyber-physical systems architecture for Industry 4.0-based manufacturing systems, Manufacturing Letters-Elsevier 3 (2015), pp. 18-23.
DOI: https://doi.org/10.1016/j.mfglet.2014.12.001

[26] L. Wang, M. Törngren, M. Onori, Current status and advancement of cyber-physical systems in manufacturing, Journal of Manufacturing Systems 37 (2015), pp. 517-527.
DOI: https://doi.org/10.1016/j.jmsy.2015.04.008

[27] P. J. Mosterman, J. Zander, Industry 4.0 As a cyber-physical system study, software & systems modeling, Springer-Verlag 15 (2016), pp. 17-29.
DOI: https://doi.org/10.1007/s10270-015-0493-x

[28] J. Wan, H. Yan, D. Li, K. Zhou, L. Zeng, Cyber-physical systems for optimal energy management scheme of autonomous electric vehicle, The Computer Journal 56 (2013), pp. 947-956.
DOI: https://doi.org/10.1093/comjnl/bxt043

[29] B. B. Sánchez, R. Alcarria, D. Sanchez-De-Rivera, A. Sanchez-Picot, Enhancing process control in industry 4.0 scenarios using cyber-physical systems, Journal of Wireless Mobile Networks, Ubiquitous Computing, and Dependable Applications 7 (2016), pp. 41-64.
DOI: https://doi.org/10.22667/JOWUA.2016.12.31.041

[30] C. Luckeneder, H. Kaindl, Systematic top-down design of cyber-physical models with integrated validation and formal verification, Proc. of 40th International Conference on Software Engineering, Gothenburg, Sweden, 27 May-3 June 2018, pp. 274-275.
DOI: https://doi.org/10.1145/3183440.3194967

[31] B. Bordel, R. Alcarria, M. Perez-Jimenez, T. Robles, D. Martín, D. S. De Rivera, Building smart adaptable cyber-physical systems: Definitions, classification and elements, Proc. of the 9th International Conference on Ubiquitous Computing and Ambient Intelligence, Sensing, Processing, and Using Environmental Information, Puerto Varas, Chile, 1-4 December 2015, pp. 144-149. Online [Accessed 07 December 2020]
https://link.springer.com/chapter/10.1007/978-3-319-26401-1_14

[32] Q. Zhu, C. Rieger, T. Başar, A hierarchical security architecture for cyber-physical systems, Proc. of 4th International Symposium on Resilient Control Systems, Boise, USA, 9-11 August 2011, pp. 15-20.
DOI: https://doi.org/10.1109/ISRCS.2011.6016081

[33] M. Conti, S. K. Das, C. Bisdikian, M. Kumar, L. M. Ni, A. Passarella, G. Roussos, G. Troster, G. Tsudik, F. Zambonelli, Looking ahead in pervasive computing: Challenges and opportunities in the era of cyber–physical convergence, Pervasive and Mobile Computing 8 (2012), pp. 2-21.
DOI: https://doi.org/10.1016/j.pmcj.2011.10.001

[34] E. A. Lee, Cyber-physical systems - are computing foundations adequate? Position Paper for NSF Workshop on Cyber-Physical Systems: Research Motivation, Techniques and Roadmap, Austin, TX, USA, 16-17 October 2006, pp. 1-9. Online [Accessed 07 December 2020]
https://ptolemy.berkeley.edu/publications/papers/06/CPSPositionPaper/Lee_CPS_PositionPaper.pdf

[35] A. Ricci, Programming with event loops and control loops - From actors to agents, Computer Languages, Systems & Structures 45 (2016), pp. 80-104.
DOI: https://doi.org/10.1016/j.cl.2015.12.003

[36] J. Wan, M. Chen, F. Xia, L. Di, K. Zhou, From machine-to-machine communications towards cyber-physical systems, Computer Science and Information Systems 10 (2013), pp. 1105-1128.
DOI: https://doi.org/10.2298/CSIS120326018W

[37] H. J. La, S. D. Kim, A service-based approach to designing cyber physical systems, Proc. of the IEEE 9th International Conference on Computer and Information Science (ICIS'10), Yamagata, Japan, 18-20 August 2010, pp. 895-900.
DOI: https://doi.org/10.1109/ICIS.2010.73

[38] Y. Tan, S. Goddard, L. C. Perez, A Prototype architecture for cyber-physical systems, ACM SIGBED 5 (2008) pp. 1-2. DOI: https://doi.org/10.1145/1366283.1366309

[39] ISO, Internet of Things (IoT). Preliminary Report 2014. Online [Accessed 29 October 2020].
https://www.iso.org/files/live/sites/isoorg/files/developing_standards/docs/en/internet_of_things_report-jtc1.pdf

[40] Platform Industrie 4.0, Reference architectural model Industrie 4.0 (RAMI 4.0). Online [Accessed 29 October 2020].
https://ec.europa.eu/futurium/en/system/files/ged/a2-schweichhart-reference_architectural_model_industrie_4.0_rami_4.0.pdf

[41] X. Li, K. Duraisamy, P. Bogdan, T. Majumder, P. Pande, Network-on-chip-enabled multicore platforms for parallel model predictive control, IEEE Transactions on Very Large Scale Integration (VLSI) Systems 24 (2016), pp. 2837-2850.
DOI: https://doi.org/10.1109/TVLSI.2016.2528121

[42] J. Kim, S.-C. Choi, I.-Y. Ahn, N.-M. Sung, J. Yun, From WSN towards WoT: Open API scheme based on one M2M platforms, Sensors 16 (2016), 1645, 23 pages.
DOI: https://doi.org/10.3390/s16101645

[43] C.-Y. Huang, C.-H. Wu, A Web service protocol realizing interoperable internet of things tasking capability, Sensors 16 (2016) 1395, 23 pages.
DOI: https://doi.org/0.3390/s16091395

[44] R. Langmann, L. F. Rojas-Peña, A PLC as an Industry 4.0 component, Proc. of 13th International Conference on Remote Engineering and Virtual Instrumentation, Madrid, Spain, 24-26 February 2016, pp. 100-155.
DOI: https://doi.org/10.1109/REV.2016.7444433

[45] I. Bélai, P. Drahos, The industrial communication system PROFIBUS and PROFInet, Applied Natural Sciences International Conference, Trnava, Slovakia, 25-27 September 2009, pp. 329-336.

[46] D. Jansen, H. Buttner, Real-time ethernet: the EtherCAT solution, International Journal of Computing and Control Engineering 4 (2017), pp. 260-267.
DOI: https://doi.org/10.1049/cce:20040104

[47] N. M. Sonawala, B. Tank, H. Patel, Implementation of MQTT protocol in context with Industry 4.0, International Journal of Advance Research in Engineering, Science & Technology 15 (2004), pp. 16-21. Online [Accessed 07 December 2020]
http://www.ijarest.com/papers/finished_papers/150418122559.pdf

[48] P. Mathur, N. Nishchal, Cloud computing: New challenge to the entire computer industry, Proc. 1st International Conference on Parallel, Distributed and Grid Computing, Solan, India, 28-30

October 2010, pp. 223-228.
DOI: https://doi.org/10.1109/PDGC.2010.5679897

[49] X. Xu, From cloud computing to cloud manufacturing, In International Journal of Robotics and Computer-Integrated Manufacturing 28 (2012), pp. 75-86.
DOI: https://doi.org/10.1016/j.rcim.2011.07.002

[50] A. Nanopoulos, R. Alcock, Y. Manolopoulos, Feature-based classification of time-series data, in: Information processing and technology, Nova Science Publishers Inc. Commack, NY, 2001, ISBN 1-59033-116-8, pp. 49-61.

[51] O. Gruber, B. J. Hargrave, J. McAffer, P. Rapicault, T. Watson, The eclipse 3.0 platform: Adopting OSGI technology, IBM Systems Journal 44 (2005), pp. 289-299.
DOI: https://doi.org/10.1147/sj.442.0289

[52] Spring Framework, Spring Dynamic Modules Reference Guide. Online [Accessed 29 October 2020].
https://docs.spring.io/spring-osgi/docs/current/reference/html/

[53] The OSGi alliance, the dynamic module system for Java. Online [Accessed 29 October 2020].
https://www.osgi.org/osgi-release-7-javadoc/

[54] Editorial information provided by DB-engines, system properties comparison Cassandra vs. MongoDB. Online [Accessed 29 October 2020].
https://db-engines.com/en/system/Cassandra%3B MongoDB

[55] V. Abramova, J. Bernardino, NoSQL databases: MongoDB vs Cassandra, Proc. 13th International Conference on Computer Science and Software Engineering, Porto, Portugal, 26-28 July 2013, pp. 14-22.
DOI: https://doi.org/10.1145/2494444.2494447

[56] D. Jeffrey, S. Ghemawat, MapReduce: Simplified data processing on large clusters, Magazine Communications - ACM 51 (2008), pp. 107-113.
DOI: https://doi.org/10.1145/1327452.1327492

[57] F. Ferracuti, A. Freddi, A. Monteriù, M. Prist, An integrated simulation module for cyber-physical automation systems, Sensors 16 (2016), pp. 645-670.
DOI: https://doi.org/10.3390/s16050645

[58] M. Prist, S. Longhi, A. Monteriù, F. Giuggioloni, A. Freddi, An integrated simulation environment for wireless sensor networks, Proc. 16th International Symposium on A World of Wireless, Mobile and Multimedia Networks (WoWMoM), Boston, MA, USA, 14-17 June 2015, pp. 1-3.
DOI: https://doi.org/10.1109/WoWMoM.2015.7158177

[59] R. Fonseca, S. Ratnasamy, J. Zhao, C. T. Ee, D. Culler, S. Shenker, I. Stoica, Beacon Vector Routing: scalable point-to-point routing in wireless sensor nets, Proc. of 2nd Conference on Symposium on Networked Systems Design & Implementation, Boston, MA, USA, 2-4 May 2005, pp. 329-342.

[60] R. Musaloiu-E., C.-J. M. Liang, A. Terzis, Koala: Ultra-low power data retrieval in wireless sensor networks, Proc. of 7th International Conference on Information Processing in Sensor Networks, St. Louis, Missouri, USA, 22-24 April 2008, pp. 421-432.
DOI: https://doi.org/10.1109/IPSN.2008.10

[61] F. Leccese, M. Cagnetti, S. Tuti, P. Gabriele, E. De Francesco, R. Durovic-Pejcev, A. Pecora, Modified leach for Necropolis scenario (2019), IMEKO International Conference on Metrology for Archaeology and Cultural Heritage, Lecce, Italy, 23-25 October 2017, pp. 442-447. Online [Accessed 07 December 2020]
https://www.imeko.org/publications/tc4-Archaeo-2017/IMEKO-TC4-ARCHAEO-2017-088.pdf