Integration between RFID and OPC UA applied to a Flexible Manufacturing System (FMS)

Gabriel Siles Petrocchi & Eduardo Paciência Godoy
Control and Automation Engineering Department
São Paulo State University (UNESP) – Sorocaba Campus
Sorocaba, São Paulo, Brazil
gabriel.petrocchi@unesp.br; eduardo.godoy@unesp.br

Vitor Mendes Caldana
Electronics Department
Federal Institute of Sao Paulo (IFSP) – Sorocaba Campus
Sorocaba, São Paulo, Brazil
vitor.caldana@ifsp.edu.br

Abstract

In the context of Industry 4.0, several technologies exist to enable and enhance the flexibility and productivity of lines allowing them to adapt to the production needs on demand. Two of these technologies are RFID and OPC UA. RFID is a highly reliable system for data management with the capacity to read and write without the need for physical or electrical. OPC UA is a communication standard which provides a secure exchange of information and data in the area of industrial automation, using an extensible standard protocol that can be used on multiplatform. This work implemented the integration between RFID and the OPC UA applied to the Flexible Manufacturing System to create different routes based on information stored and read by the RFID antennas. The development of the project used the Node-RED environment as a programming and visualization tool to integrate RFID and OPC UA. Final test showed that depending of the RFID code the part makes different stops at the stations of the manufacturing system, thus proving the integration between RFID and OPC UA to achieve Flexible Production.

Keywords
Industrial Automation, Industry 4.0, RFID Technology, OPC UA, Node-RED

1. Introduction

With the growth of industrial automation, companies have been able to increase the quality of their processes, reduce costs and labor, and improve ergonomics. This is all thanks to repetitive manual labor starting to be performed by machines. But this substitution has not led to a higher rate of unemployment because machines need trained people to monitor, control and maintain them (Xavier et al., 2023).

One consequence of accelerated industrial growth is the emergence of the term Industry 4.0, a term that appeared in 2011 in an industry project in Germany called Plattform Industrie 4.0. Industry 4.0 assimilates information and communication technologies to improve the productivity, management and quality of products and processes. It is characterized by the presence of decentralized processes, Cyber-Physical Systems (CPS) and the Internet of Things (IoT) (Sacomano, et al., 2018). The monitoring of production process information in Industry 4.0 is carried out by supervisory systems, made up of sensors, Programmable Logic Controllers (PLCs) and the communication network(s) used by the company. They increase the quality standards of industrial operations, reduce downtime and help to detect faults (Altus, 2023).

RFID (Radio Frequency Identification) technology first appeared in the Second World War, used in radar systems to identify approaching aircraft, but it was not possible to differentiate between enemy and allied planes. As a result, the British created the first active identifier called IFF (Identification Freind or Foe), transmitters were installed in British planes that sent response signals to radars on the ground. With the development of new RFID system architectures, the technology began to perform identification automatically by means of a reader with antennas that send radio

frequency signals to a microchip, which reflects with response signals to the reader. This new RFID technology architecture has made it possible to apply it to product tracking and localization systems (Pinheiro, 2017).

To make it easier to configure supervisory systems on different machines and equipment, the Open Platform Communication Unified Architecture (OPC UA) communication standard was created, making it a universal machine-to-machine communication protocol. OPC UA arose from the good performance of classic OPC, possessing and improving all the functions of its predecessor and presenting new features, being compatible with the most popular operating systems, cloud servers and a reliable security system with encrypted sessions, message signing and authentication system for the servers (OPC Foundation, 2017).

1.1. Objectives

The importance that Industry 4.0 is currently bringing to industrial automation is remarkable, providing total real-time monitoring and tracking of all processes, from production to delivery to the customer, while at the same time providing security of access to information, so the main objective of this work is to integrate a supervisory system into a manufacturing system using RFID technology and OPC UA communication. Specifically, the other objectives of the work are:

- To configure the initialization of the RFID reader and its antennas automatically.
- To read the position of the tagged parts in real time and to be able to write actions to the tag, in which the reading information and writing actions can be accessed and executed by a Dashboard and by the OPC UA server.
- Based on the value of the write, different routes will be mapped by the manufacturing system.

2. Literature Review

In this section we will cover a brief review of the technologies of this paper (RFID and OPC UA) as well as the Node-RED platform used in the development of the integration solution.

2.1. RFID technology

RFID systems mean that any object can be identified and tracked using a tag. The tags have Transponders or labels that emit radio frequency messages that are picked up by the RFID reader's antennas. These messages can contain information such as identification number, user data, etc. Tags also have writable memory, allowing data to be entered and modified. Tags can be classified into two categories, active and passive. Active tags have their own power supply, are larger and cost more, but have a higher signal strength which enables data to be transmitted over a longer distance to the antennas. Passive tags are smaller and cheaper, they can work at high and low frequencies, but their memory capacity is lower, and they emit a signal that travels shorter distances (Weinstein, 2005).

A major advantage of RFID technology is its reliability, because with the presence of tags, it is easy to read without the need for a specific position on the part and without any physical or electrical contact. In addition to being able to simultaneously read multiple objects with tags, reusability and durability and greater practicality in repetitive tasks (Pinheiro, 2017).

2.2. OPC UA communication

OPC UA was created in 2008 with all the features of classic OPC, but also with new features, such as: multi-platform functionality, cloud-based infrastructure, more security features, extensible structure and extensive information modeling (OPC Foundation, 2017).

The OPC UA protocol has an architecture based on web technology standards, making it possible to communicate between clients and servers with different types of networks. On the server, the data is organized within an Address Space standard with a structure that is ideal for vertical integration of automation systems. In this way, OPC UA communication can be adopted in different sectors of a company to exchange information (Melo, 2020).

In address spaces, data is organized in a hierarchy allowing complex structures to be used by clients, such as offering different types of permission to read and write data (OPC Foundation, 2017). Address spaces are represented by sets of Nodes, their characteristics are called Attributes and the links between the Nodes for information to pass through are called References. The Nodes have a series of derived classes*: Variable, Object, View, DataType, ReferenceType, Method, VariableType and ObjectType*. Each of these derivations has its own Attributes, the most common of which are: *NodeID, BrowseName. NodeClass, Description and DisplayName* (Melo, 2020).

2.3. Node-RED

Node-RED was developed in 2013 by IBM engineers. Its development environment is open to the public and is based on JavaScript and Node.js, making it an ideal tool for developing IoT systems. Its simple operating logic makes it easy to handle data for high-level systems (servers, cloud-based services) (Ferencz & Domokos, 2019). Node-RED's virtual programming environment is based on the creation of data flows linking blocks with different functions, which are called nodes (OpenJS Foundation & Contributors, 2019). It also has its own Dashboard interface, which can be edited and allows data to be presented in textual and graphical form (Ferencz & Domokos, 2019).

Currently, Node-RED can be used for interfacing sensor data, wired and wireless communication, social media alerts, data analysis and any implementation that has Node.js functionality services. Node-RED enables inexperienced developers to create reusable code and prototype effectively. At the same time, it allows experienced developers to focus on identifying and developing CPS application solutions, rather than building application components from scratch (Fiaidhi & Mohammed, 2021).

Node-RED has 3 basic elements, namely the node panel, the flow panel and the Info and Debug panel. Each node provides a different function that creates and/or stores data in the form of objects or JSON strings, where JSON is a way of representing objects within JavaScript that facilitates application in various protocols (Ferencz & Domokos, 2019). Two important properties of nodes are *msg.payload* and *msg.topic*. The *msg.payload* is the standard property that most nodes work with, it is where the data sent and moved by the flows is stored in the form of messages, which can be a simple String or even an Object with several classes. The *msg.topic* is used as an identifier of the source of the messages or can be used to identify different types of messages in the same flow, for example, meteorological data of temperature, humidity and wind speed are sent from the same source to an application in Node-RED, by defining a different *msg.topic* for each piece of data, there will be no problems in analyzing the data separately even though they come from the same source (same message flow) (OpenJS Foundation & Contributors, 2019).

3. Method, Materials and Development

This paper will be integrated with other existing and future research on a larger project. As such, some of the choices of protocols and programming are dependent of the existing scenario and other applications that also run on the FMS.

Figure 1 shows an idealized flow of the project's operation, highlighting the trajectory of the data used to write and read the tags and the data related to the automatic configuration of the reader.

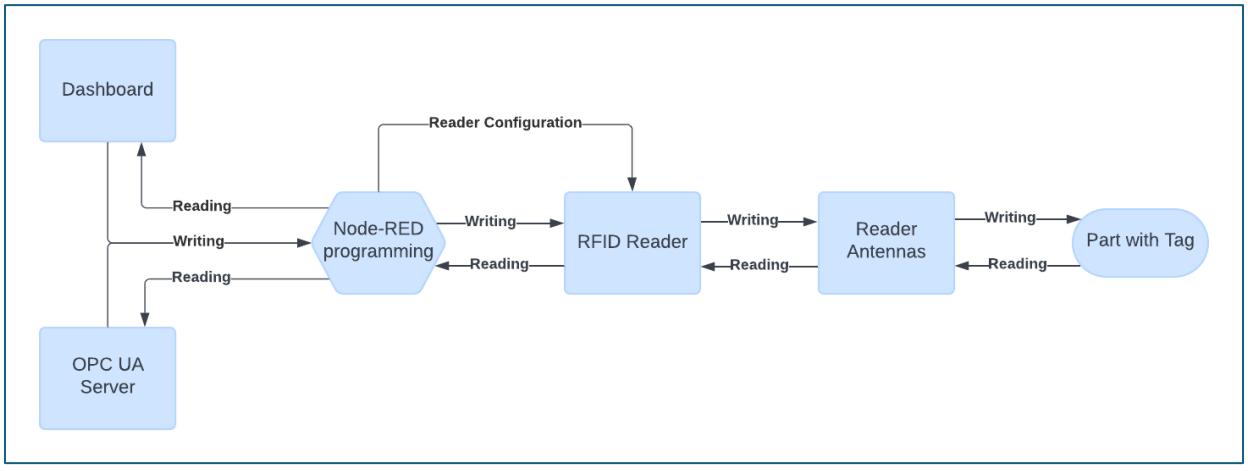

Figure 1: Flowchart showing the general operation of the project (The Authors).

The FMS used in this paper is manufactured by FESTO[1] and is currently installed in the Automation Laboratory II at UNESP Sorocaba Campus. The RFID reader used was the INfinity 510 developed by Sirit[2]. All the programming was

[1] https://www.festo.com/us/en/
[2] https://www.linkedin.com/company/sirit/

developed in the Node-RED environment. The UA Expert software developed by Unified Automation[3] was used to validate and test the OPC UA client by reading and writing the variables.

3.1. FESTO FMS

The FESTO flexible manufacturing system at UNESP Sorocaba is used for teaching and research purposes, with its 6-station modular design, it is possible to create several different routes as well as different production scenarios. As seen on Figure 2 below, the manufacturing system is connected by a conveyor system, that is responsible for part transportation between the several stops of the line. The stops are:

- Distribution/Testing – where parts are fed to the line and identified by color type
- Processing – where part can be tested and drilled
- Vision – where a photo of the part is taken for quality purposes
- Robot – where a lead can be assembled to the part
- Storage – where unfinished parts can be stored for future use
- Sorting – where parts are sorted and exit the line.

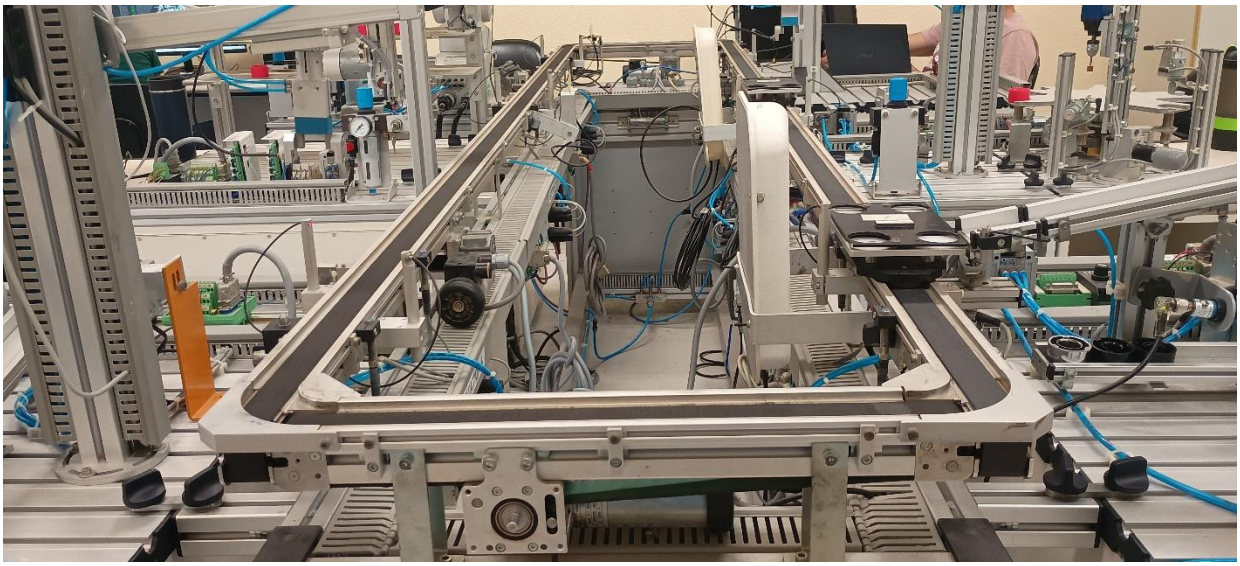

Figure 2: FESTO FMS (The Authors).

3.2. INfinity 510 RFID reader

The INfinity 510 is a high-performance multi-protocol, multi-regional Radio Frequency reader that operates in the 860 to 960 MHz UHF (Ultra-high Frequency) range, supports up to four read and write antennas, has a serial input interface and an Ethernet interface (Sirit Inc., 2010) as seen in Figure 3. The reader is installed next to the FMS and its information is passed on via Ethernet communication.

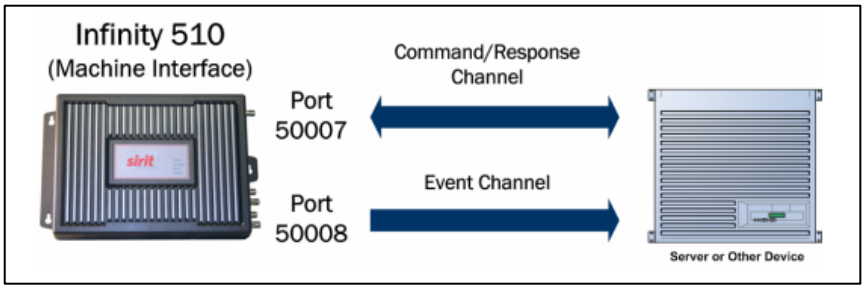

Figure 3: Port addresses and their functionalities for the INfinity 510 reader's M2M communication (Sirit Inc, 2010).

[3] https://www.unified-automation.com

It is very versatile, operating in most RFID system applications. All the reader's functions and parameter control are easily configured by sending a Command Line Interface (CLI) to the reader's command channel. To perform Machine-to-Machine (M2M) communication via Ethernet connectivity, this RFID reader has two port addresses, ports 50007 and 50008. Commands to the reader are received through port 50007. Every command sent to the reader generates a response through the same port, alerting the reader whether the command was successful or not. Port 50008 is a port that only sends (unidirectional) data from asynchronous events such as the arrival and departure of tags (Sirit Inc, 2010).

3.3. Node-RED PLC programming

As described, this paper is part of a larger project and the relevant Node-RED basic programming can be found at Sousa et al. (2024). To develop this paper, information and the status of FMS sensors are collected via Node-RED. The palette of nodes used to transit this information is called *node-red-contrib-s7*, which has nodes for controlling PLC data.

3.3.1. RFID Reader Configuration

The RFID reader's configurations were made via its Command Line Interface (CLI). The Node-RED sends all the necessary configuration in a single message as soon as communication with port 50008 is established with the Node-RED. Every time this communication takes place, the reader sends through port 50008 the so-called Connection ID, which is a 2-digit identification used to configure the event data (arrival and departure of tags) sent through port 50008. Table 1 shows the list of CLI commands used and their functionalities, all these commands are sent to the reader through port 50007 via a single string.

Table 1: Commands used to configure the RFID reader.

Command	Function
setup.operation_mode=autonomous	It sets the reader's operating mode to autonomous mode, making it work every time a tag enters the range of the antennas, signaling asynchronously via the event channel.
modem.protocol.isoc.physical.set (tari=tari_12_50, return_link_freq=LF160, data_1_lenght=d1_len_20, rt_modulation=rt_mod_pr, tr_enconding=tr_enc_fm0, interrogator_mode=dense)	Configuration of physical parameters. • *tari=tari_12_50* sets the reference length for the low signal (0) to 12.5 microseconds. • *return_link_freq=LF160* sets the frequency at which the tags respond to the reader to 160 Kbps. • *data_1_lenght=d1_len_20* sets the reference length for the high signal (1) to 2 times longer than the low signal. • *rt_modulation=rt_mod_pr* sets the modulation type used in the transmission between the reader and the tag to phase-reversed amplitude mode. • *tr_enconding=tr_enc_fm0* sets the encoding used between the tag and the reader to standard mode. • *interrogator_mode=dense* sets regulatory filters on the reader in dense mode, in dense mode all filters are activated.
antennas.X.conducted_power = Y	Antenna feed configuration, where X is the antenna number and Y is the antenna consumption in dBm (milliwatt decibels), for antennas 1 and 4 260 dBm was configured, for antenna 2 180 dBm and for antenna 3 200 dBm.
tag.reporting.arrive_fields=tag_id antenna time user_data tag.reporting.depart_fields=tag_id antenna time user_data.	Commands that configure the RFID reader to display the following tag information via the event channel every time a tag arrives or departs an antenna's range: its tag ID number, the antenna that captured it, the date/time of capture and the user data written on the tag.
reader.events.register(XX, event.tag.arrive) reader.events.register(XX, event.tag.depart)	Register the event channel with the Connection ID for both incoming and outgoing tag information from the antenna range, where XX is the Connection ID provided by port 50008 of the RFID reader.

One issue encountered was the proper configuring the reader's antennas. Initially, even when the part was stationary in front of antennas 3 and 4, the signal oscillated between outgoing and incoming events, especially from antenna 4. There were also occasional errors when the Tag ID was written by antenna 1. The first solution to these problems was to increase the power consumption of these antennas to increase their range, but this caused antenna 3 to start picking up signals from parts that were not near its respective station. These problems were solved when the RFID configuration for the frequency at which the tag responds to the reader, the *return_link_freq* command, which initially had a value of 80 Kbps, was increased to the current value of 160 Kbps. In addition to solving the reading problems, this action brought much greater stability to the writing actions.

3.3.2. Reading and writing from the Node-RED Dashboard

Reading from the Dashboard can be done in two ways, automatically via event port 50008, as seen in Figure 4, and manually via a button on the Dashboard, which can be seen in Figure 5. Automatically, data is sent to the dashboard whenever a tag arrives or departs from the reader's reach. By pressing the button, data is only sent when there is a tag in range of the reader, in the programming, the button sends a CLI to the reader to perform the reading action through a function node,

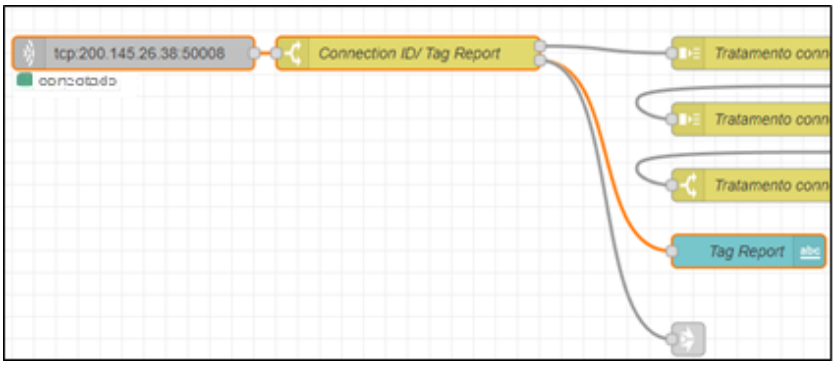

Figure 4: Node-RED programming highlighting in orange the sending of tag readings to the Dashboard (The Authors).

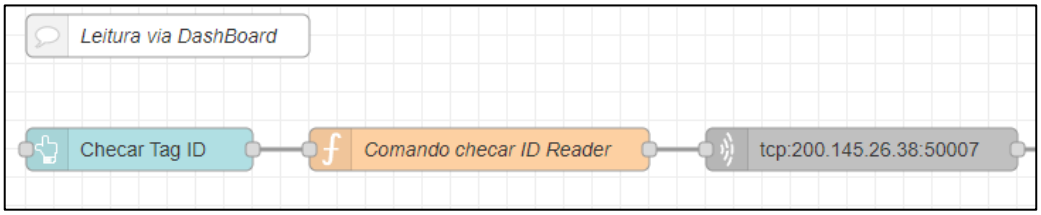

Figure 5: Programming the read action by the button (The Authors).

To write to a tag via the Dashboard, was used a node that creates a form field for the user to type in. Thus, the program creates the CLI based on the data written by the user (function node) and sends it to the reader, writing the new information in the tag. This flow is showed in Figure 6 below.

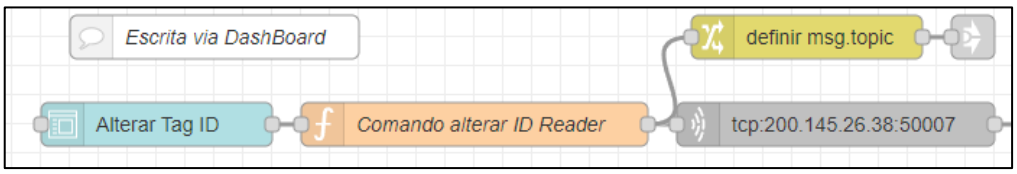

Figure 6: Writing programming via Dashboard (The Authors).

3.4. UA Expert

UA Expert is a software designed for the use of testing a multi-platform OPC UA client, programmed in C++ with a basic structure that can be extended by implementing plugins. However, its free version comes with plugins installed providing the main OPC UA features, such as: OPC UA Data Access View, OPC UA Alarms and Conditions View,

Image View and Server Diagnostics View. In this project, the software feature most used will be the data access view, for testing and validating server data and variables.

3.5. Reading and writing by the OPC UA server/client

Since this project aims to integrate other existing projects, the OPC UA server used between these projects will be the same so that all variables and information are organized in the same place. The method of creating variables for this project takes advantage of the method of existing projects so that there are no errors or data conflicts. Table 2 shows all the variables related to this project.

Table 2: List of variables created on the OPC UA server.

Variable name	Function
Antena1_TagReport	Reads when a tag enters or leaves the antenna's range 1
Antena1_R_TagID	Reads the Tag ID of a tag while it is in range of antenna 1.
Antena1_R_UserData	Reads the User Data of a tag while it is in range of antenna 1.
Antena1_W_TagID	Writes the tag ID to a tag in range of antenna 1.
Antena1_W_UserData	Writes User Data to a tag that is in range of antenna 1.
Antena2_TagReport	Reads when a tag enters or leaves the antenna's range 2
Antena2_R_TagID	Reads the Tag ID of a tag while it is in range of antenna 2.
Antena2_R_UserData	Reads the User Data of a tag while it is in range of antenna 2.
Antena2_W_TagID	Writes the tag ID to a tag in range of antenna 2.
Antena2_W_UserData	Writes User Data to a tag that is in range of antenna 2.
Antena3_TagReport	Reads when a tag enters or leaves the antenna's range 3
Antena3_R_TagID	Reads the Tag ID of a tag while it is in range of antenna 3.
Antena3_R_UserData	Reads the User Data of a tag while it is in range of antenna 3.
Antena3_W_TagID	Writes the tag ID to a tag in range of antenna 3.
Antena3_W_UserData	Writes User Data to a tag that is in range of antenna 3.
Antena4_TagReport	Reads when a tag enters or leaves the antenna's range 4
Antena4_R_TagID	Reads the Tag ID of a tag while it is in range of antenna 4.
Antena4_R_UserData	Reads the User Data of a tag while it is in range of antenna 4.
Antena4_W_TagID	Writes the tag ID to a tag in range of antenna 4.
Antena4_W_UserData	Writes User Data to a tag that is in range of antenna 4.
StatusEscrita	Indicates whether the server write was successful or not.

The data read by the server is treated in a similar way to the data read by the dashboard as showed in Figure 7, but here the programming separates the data by antenna. However, in the case of data sent to the server, it is necessary to inform the Namespace in addition to the variable in which the data will be written by the Node-RED programming.

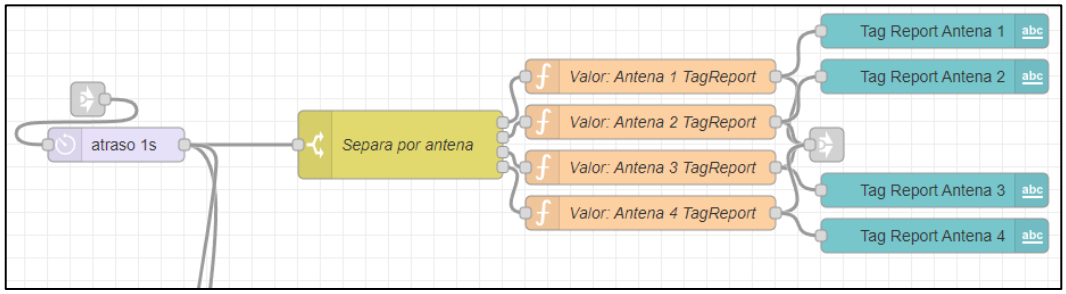

Figure 7: Node-RED programming done to send data to the OPC UA server (The Authors).

When something is written to the server, the data reaches the Node-RED via the output of the OPC UA Server node, where the names of the server variables are stored in the *msg.payload.variableName* property and the value of the variables is stored in the *msg.payload.variableValue* property of the message, so the data is manipulated from these two properties. From the server you can write/change the Tag ID and User Data of a tag as seen in Figure 8.

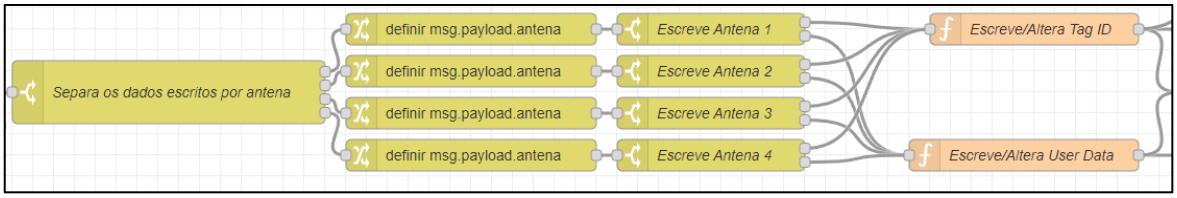

Figure 8: Node-RED programming for writing via OPC UA server (The Authors).

3.6. Configuring the part path in the FMS based on the writing.

Figure 9 shows a flowchart that idealizes the logical operation of the system's programming in relation to the part's journey through the FMS according to the Tag ID value.

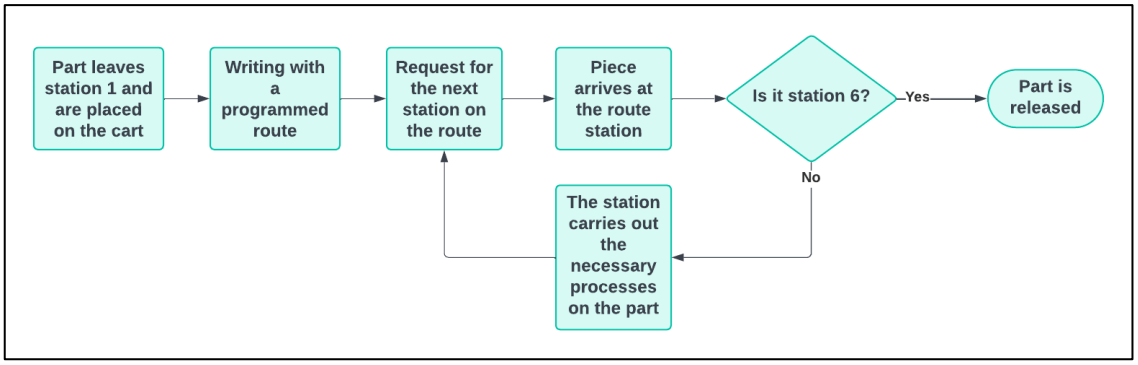

Figure 9: Flowchart of the part's route based on writing (The Authors).

Based on the writing of specific Tag IDs, the part will stop at certain stations in the FMS. The programming related to this functionality aimed to make it easier to define these specific Tag IDs and to make it easier to define which station these tags will stop at and is showed in Figure 10

Writing is carried out by antenna 1, immediately after the part leaves the Testing station (Station 1). Programming requests a stop at a station depending on the flow connections made: Processing (Station 2), Vision (Station 3), Robot Arm (Station 4), Storage (Station 5) and Sorting (Station 6).

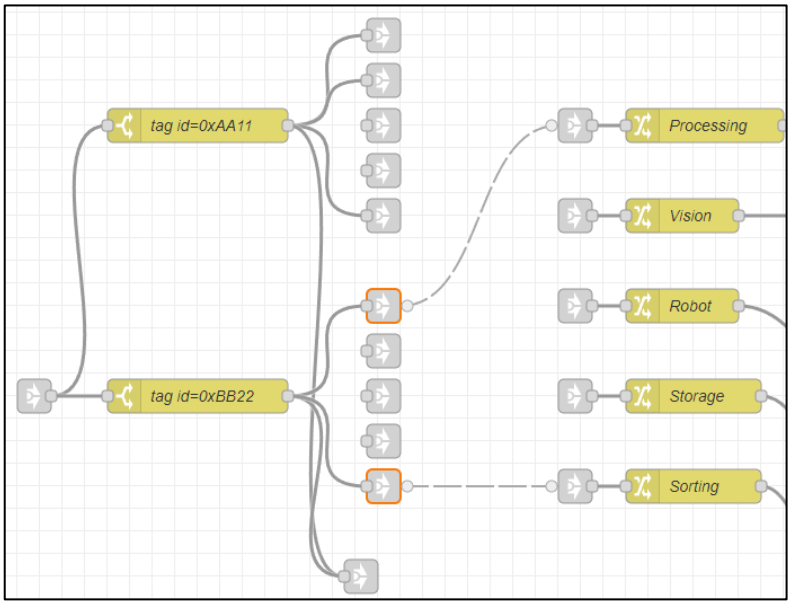

Figure 10: Programming the definition of tag IDs and their stops (The Authors).

Depending on the station where the part is requested from the FMS, it may or may not be manipulated by the station, such as station 3, where the part is only photographed, and no manipulation takes place. So, when the part is not handled, the station is stopped for a timed period.

In the first versions and tests of this functionality, the continuous signal received from the RFID reader's antennas caused unwanted interactions in Node-RED programming, such as interrupting and locking the FMS, as well as sending request signals to the wrong station. This problem was overcome by using Node-RED's *flow.set()* and *flow.get()* functions. With these, it was possible to create variables that indicated conditions and actions for flows that were not directly linked to each other, thus creating a well-defined order of priority for actions.

4. Results and Discussions

Two validation tests of the part's path in the FMS based on the writing were carried out and recorded on video, one with a written value of 0xAA11 in the Tag ID and the other with a value of 0xBB22.

Figure 11 shows the flow of the system's behavior when the write value in the Tag ID is 0xAA11. A video of how the system works with the write value of 0xAA11was made. In the video, after the tag has been written by antenna 1, there are stops at station 2 Processing (Part handling) and station 3 Vision (Time stop) and then the part is forwarded to the last station.

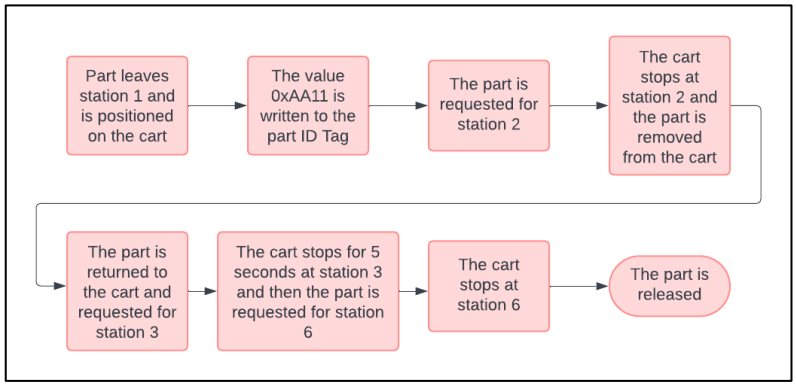

Figure 11: System operation with the value 0xAA11 (The Authors).

Figure 12 shows the flow of the system's behavior when the value written to the Tag ID is 0xBB22. A video of how the system works with the value of writing 0xBB22. In the video, after the tag has been written by antenna 1, there is a stop only at station 2 Processing and then the piece is sent straight to the last station.

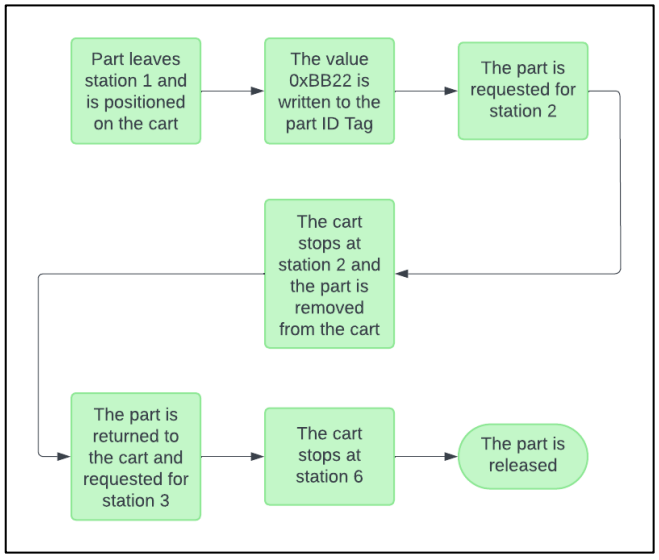

Figure 12: System operation with value 0xBB22 (The Authors).

Figure 13 shows all the variables created in UA Expert's Data Access window; the name of each variable can be seen in the Display Name column. With similar characteristics to reading and writing via Dashboard, but the writing options are more specific. On the OPC UA server, writing is done by antenna, which makes it possible to apply multiple tagged parts to the FMS at the same time in the future.

Data Access View

#	Server	Node Id	Display Name	Value	Datatype
1	OPC	NS4|String|...	Antena1_TagReport	event.tag.arrive tag_id=0xA111 user_data=0x00000000000000000000000000000000, last=2024-07-18T15:45:07.661, antenna=1	String
2	OPC	NS4|String|...	Antena1_R_TagID	tag_id=0xA111	String
3	OPC	NS4|String|...	Antena1_R_UserData	user_data=0x00000000000000000000000000000000	String
4	OPC	NS4|String|...	Antena2_TagReport	event.tag.depart tag_id=0xBB22, user_data=, last=2024-07-18T15:33:29.331, antenna=2	String
5	OPC	NS4|String|...	Antena2_R_TagID		String
6	OPC	NS4|String|...	Antena2_R_UserData		String
7	OPC	NS4|String|...	Antena3_TagReport	event.tag.arrive tag_id=0xBB22, user_data=, first=2024-07-18T15:35:20.124, antenna=3	String
8	OPC	NS4|String|...	Antena3_R_TagID		String
9	OPC	NS4|String|...	Antena3_R_UserData		String
10	OPC	NS4|String|...	Antena4_TagReport	event.tag.depart tag_id=0xBB22, user_data=, last=2024-07-18T15:36:13.472, antenna=4	String
11	OPC	NS4|String|...	Antena4_R_TagID		String
12	OPC	NS4|String|...	Antena4_R_UserData		String
13	OPC	NS2|String|...	StatusEscrita	Escrita feita com sucesso pelo servidor	String
14	OPC	NS4|String|...	Antena1_W_TagID	0xA111	String
15	OPC	NS4|String|...	Antena1_W_UserData		String
16	OPC	NS4|String|...	Antena2_W_TagID		String
17	OPC	NS4|String|...	Antena2_W_UserData		String
18	OPC	NS4|String|...	Antena3_W_TagID		String
19	OPC	NS4|String|...	Antena3_W_UserData		String
20	OPC	NS4|String|...	Antena4_W_TagID		String
21	OPC	NS4|String|...	Antena4_W_UserData		String

Figure 13: OPC UA server read and write variables displayed in the Data Access View window of the UA Expert software (The Authors).

5. Conclusion

This work aimed to integrate RFID technology and the OPC UA communication standard applied to a manufacturing system using the Node-RED environment for its programming, with the following objectives: automatic configuration of the RFID reader and its antennas; reading and writing via the Node-RED Dashboard and via the OPC UA server; and defining the route through the manufacturing system based on the writing carried out. To complete the project, it was necessary to study the functionalities and characteristics of the RFID reader, in addition to studying the functions and nodes of the Node-RED and understanding their interactions between the reader and the OPC UA communication. It was necessary to study the workings of RFID technology and readers, OPC UA communication standards and the functionalities of the Node-RED environment to understand how to integrate these three tools in a functional way and meet the proposed objectives.

With the challenges overcome, it can be concluded that the main objective of integrating RFID technology and OPC UA communication applied to a manufacturing system has been achieved. However, as a recommendation for future work, the project can still be improved by making it more flexible, with the possibility of changing the writing during production, thus making the system reconfigure the part's path through the FMS automatically, taking into account whether it is possible to make the change, for example, if the part has already passed through station 2 in production, not allowing the change for parts that do not pass through this station.

Acknowledgements

This paper was developed in part by researches of the GPI4 Research Group[4]

References

Altus. Como o sistema supervisório ajuda na produtividade de uma empresa? 2023. Available: https://www.altus.com.br/blog/busca/pagina/1?palavrachave=supervisórios&validadorantispam=8185456. 24 jun. 2024.

Ferencz, K., Domokos, J., Using Node-RED platform in an industrial environment. *XXXV. Jubileumi Kandó Konferencia, Budapest*, p. 52-63, 2019. Available:

[4] http://dgp.cnpq.br/dgp/espelhogrupo/350090

https://www.researchgate.net/publication/339596157_Using_Node-RED_platform_in_an_industrial_environment. 26 jun. 2024.

Fiaidhi, J., Mohammed, S., Virtual care for cyber–physical systems (VH_CPS): NODE-RED, community of practice and thick data analytics ecosystem. *Computer communications*, v. 170, p. 84-94, 2021. //doi.org/10.1016/j.comcom.2021.01.029.

Medieros, M. J. A. P. de, et al. Tecnologia RFID: contribuições de uso/aplicação em Unidades de Informação (Bibliotecas). 2011. Available: https://repositorio.ufpb.br/jspui/handle/123456789/2247. 25 jun. 2024.

Melo, P. F. S. de. Dispositivo de controle para a indústria 4.0 baseado no RAMI 4.0 e OPC UA. São João da Boa Vista: Dissertação (mestrado) - Universidade Estadual Paulista "Júlio de Mesquita Filho", Câmpus Experimental de São João da Boa Vista, 2020. Available: http://hdl.handle.net/11449/192851. 25 jun. 2024.

Melo, P. Introdução ao OPC UA (Open Platform Communications Unified Architecture). 2017. Disponível em: https://embarcados.com.br/introducao-ao-OPC UA/. Acesso em 25 jun. 2024.

OPC Foundation. Unified Architecture. 2017. Available: https://opcfoundation.org/about/opc-technologies/OPC UA/ 25 jun. 2024.

OpenJS Foundation & Contributors. Node-RED. 2019. Available: https://nodered.org. 26 de jun. 2024

Pinheiro, J. M. dos S. Identificação por Radiofrequência: Aplicações e Vulnerabilidades da Tecnologia RFID. *Cadernos UniFOA, Volta Redonda*, v. 1, n. 2, p. 18–32, 2017. DOI: 10.47385/cadunifoa.

Sanches, H B. Monitoramento da produção e da eficiência de processos de manufatura usando RFID e internet das coisas. Bauru: Dissertação (mestrado) - Universidade Estadual Paulista "Júlio de Mesquita Filho", Faculdade de Engenharia, Bauru, 2018. Available: http://hdl.handle.net/11449/157236. 24 jul. 2024.

Sirit Inc. INfinity 510 Quick Start Guide. Ontário, v.3.1, 2010.

Sirit Inc. INfinity 510 Protocol Reference Guide. Ontário, v.2.02, 2008.

Sousa, D. R. de, Godoy, E. P., Caldana, V. M. Upgrading legacy systems for Industry 4.0 with Node-RED and OPC UA. 2024. Available: https://www.researchgate.net/publication/378654188_Upgrading_legacy_systems_for_Industry_40_with_Node-RED_and_OPC UA. 24 jul. 2024.

TT Electronics. RFID: The Technology Making Industries Smarter. 2021. Available: https://www.ttelectronics.com/blog/rfid-technology/. 25 jun. 2024.

Unified Automation GMBH. UaExpert—A Full-Featured OPC UA Client. 2023. Available: https://www.unified-automation.com/products/development-tools/uaexpert.html?gad_source=1&gclid=Cj0KCQjw1qO0BhDwARIsANfnkv-iuVF_3VajmJv4Bq5hfFvZx0adfLD664F_9HABfTHT90tv6OPW34EaAiRmEALw_wcB. 28 jun. 2024.

Weinstein, R. RFID: a technical overview and its application to the enterprise. *IT professional*, v. 7, n. 3, p. 27-33, 2005. Available: https://ieeexplore.ieee.org/abstract/document/1490473. 25 jun. 2024.

Xavier, A. B. et al. A Automação Industrial Como Solução e Não Como Ameaça Aos Trabalhadores. *GeSe: Revista de Gestão e Secretariado*. v.14, n.6, p. 9019–9032, 2023 Available: https://unesp.primo.exlibrisgroup.com/permalink/55UNESP_INST/1j0ltst/cdi_proquest_journals_2834510477. 24 jun. 2024.

Biography

Gabriel Siles Petrocchi is a Mechatronics Technician who graduated from the ETEC Rubens de Faria e Souza technical school in 2014 and graduated in Control and Automation Engineering from UNESP Sorocaba campus in 2024. Currently working in the related area of banking automation.

Vitor Mendes Caldana began his career with a technician course in Electronics from Liceu de Artes e Ofícios in 1999, followed by an undergraduate degree in Electronic Engineering from Universidade Presbiteriana Mackenzie in 2004. In 2016, finished his M.Sc. in Industrial Engineering. Since 2023 is a student at UNESP to obtain his Ph.D. in Electronic Engineering in the Industry 4.0 field. In 2014 began his teaching career in FIEB as a substitute teacher, followed by an associate professor position for the Technical Course of Electronics. In 2016 became a full-time professor by joining IFSP, moving to the Sorocaba Campus to implement the Electronics High-School Technical Course. In 2018 started the Research Group in Industry 4.0 at IFSP and has been its leader since. Between 2019 and 2020, along with his colleagues, designed and implemented the first Post-Graduate Program in Industry 4.0 of IFSP at the Sorocaba Campus. He is currently involved in research projects in Industry 4.0. https://lattes.cnpq.br/8361188962318020.

Eduardo Paciencia Godoy received the B.Eng. degree in Control and Automation Engineering at Itajubá Federal University (MG-Brazil) in 2003 and the M.Sc. and Ph.D. degrees in Mechanical Engineering at University of São Paulo (SP-Brazil) in 2007 and 2011. Currently he is an Associate Professor of Unesp (SP-Brazil) with research interests in IoT and Industry 4.0. http://lattes.cnpq.br/0072632067545698.