Upgrading legacy systems for Industry 4.0 with Node-RED and OPC-UA

Daniel Ribeiro de Sousa & Eduardo Paciência Godoy
Scientific Initiation Program PIBIC Junior CNPQ/Reitoria UNESP
São Paulo State University (UNESP) – Sorocaba Campus
Sorocaba, Sao Paulo, Brazil
Daniel_21sousa@outlook.com; eduardo.godoy@unesp.br

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

Abstract

Even though Industry 4.0 has been advancing significantly in the last years, the upgrade of Legacy systems is still a reality that needs to be attended to due to their large presence in developing countries. To achieve this goal, this paper proposes a continuation of previous research to increase knowledge and improve efficacy of a Node-RED / OPC-UA based IIoT automation solution for a Festo FMS Legacy System with S7-300 PLC´s. This paper will focus on scaling and adding new stations, as well as improving the existing structure for future upgrades. At the end, the results showed a possible scenario to keep using Node-RED and OPC-UA with some reservations as well as future possible work.

Keywords
Industry 4.0, Node-RED, OPC-UA, Legacy Systems, Industrial Automation

1. Introduction

Introduced in Germany in 2011. The 4th industrial Revolution, or Industry 4.0, describes the integration of modern and complex Technologies, such as Industrial Internet of Things (IIOT), Artificial intelligence (AI) amongst others bringing a new way to produce. One of the main aspects of Industry 4.0 is the fusion of the real and virtual worlds, creating what is now known as Cyber Physical Systems (CPS). These systems represent a new approach on production management, allowing control e monitoring of all production lines virtually, creating modular and flexible factories. Flexible Manufacturing Systems (FMS) also allow the creation of personalized products based on user needs, as well as allowing, with the addition of CPS and Digital Twins (DT) for better maintenance programs, production run simulations, defect prediction and . (Singh et al., 2021; Teixeira et al., 2016)

Industry 4.0 brought challenges and opportunities both in the management aspect (Bajic et al., 2021) as in the technical aspect (Rikalovic et al., 2022), however the implementation of new technologies is facing several obstacles, with one of the most relevant being interoperability (Burns et al., 2019). When studying Legacy Systems, the investment needed for upgrading the production lines to Industry 4.0 makes it unpractical (Kutscher et al., 2020; Zielstorff et al., 2023), making all the benefits of Industry 4.0 and CPS out of the reach of these segment of industry that is still significant in developing countries.

To face the interoperability issue, several initiatives were undertaken to create standards that can be used to communicate between systems. From shop floor automation to high end Execution Systems and Enterprise Resources Paling (ERP) such as OPC-UA (Open Platform Communications – Unified Architecture), Profibus, Modbus and Ethernet. One of the alternatives for the high cost in replacing legacy systems is upgrading them using Technologies such as OPC-UA (Park & Wook Jeon, 2019) and Node-RED (Tabaa et al., 2018), integrating legacy systems to the IIOT platform and to CPS and DT. With this upgrade, with lower cost and easier implementation, the Legacy Systems

can have access to the advantages of Industry 4.0 (Schleich et al., 2019), like CPS and DT. It is necessary, however, to considered the necessity for clear nodes and flows as described by Clerissi et al. (2018). Based on the proposed methodology of incremental upgrades by Kutscher et al. (2020) this paper continues the development for the previous research named "Node-RED for PLC Automation" (Sousa & Caldana, 2023). We will continue to explore, expand and optimize the ideas and concepts from the first work while taking a deeper understanding into Node-RED.

1.1. Objectives

The main objective of this research is to advance on the development of a Node-RED solution for PLC Automation, using it as an interface as well as a toll for IIOT for Legacy Systems using OPC-UA Server. To achieve the main goal, two secondary objectives were defined as below:

a) Continue the development and implementation from the first research mentioned above, with a focus on enhancing the use of Node-RED and reduce the number of nodes needed to perform the operation while improving the end-user interface. This will involve the critical analysis of the last work and points of improvement discussed in congresses as well as a deeper understating of the Node-RED language and its uses.

b) Since the first work was a pilot test, our objective with the continuation will be to implement all stations of the FMS that can be seen in Figure 1**Erro! Fonte de referência não encontrada.** below. This will add 5 new programing stations as well as necessary commands for the conveyor belt. There will be need to identify the variables by each station as the number increase significantly.

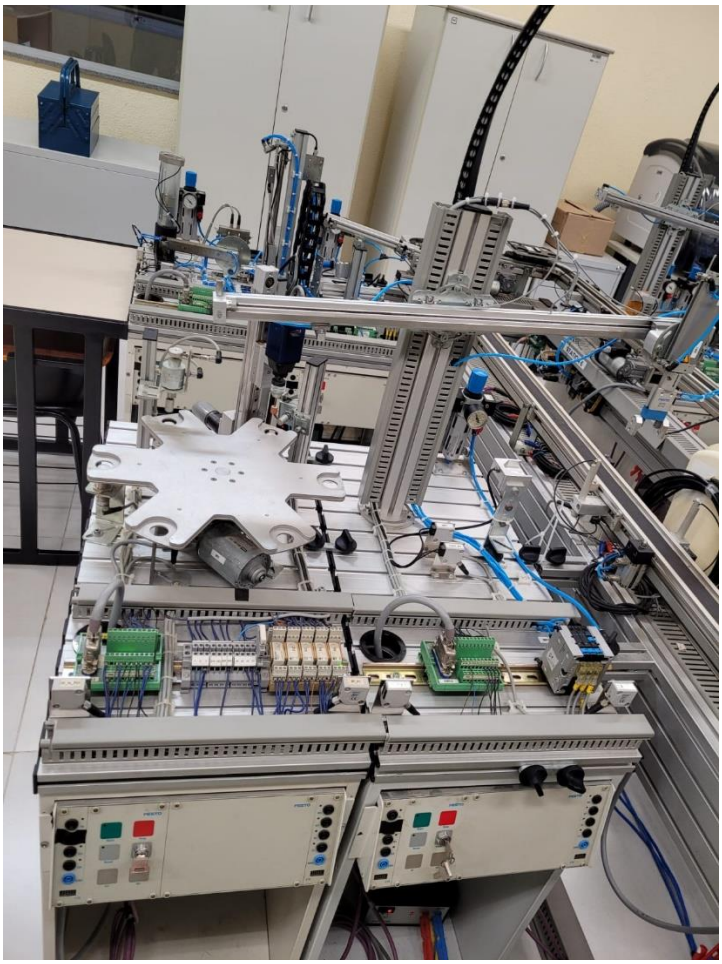

Figure 1: FESTO FMS Legacy System

The structure of the paper is as follows: on section 2 we provide the necessary literature review for Node-RED. On section 3 we address the updated flows to achieve objective listed above. Section 4 shows the results and discuss the work performed. Section 5 is the conclusion and future work.

2. Literature Review

To follow is the necessary literature review to achieve the objectives described in section 1.1. The method used was the "integrative review" that revises, investigates and comprehend different studies, unifying them towards creating a solution for the problems and objectives of the work. Since this is a continuation from previous work, this review will focus on the part developed on this paper alone, and for more information about the infrastructure of the FMS and the PLC please refer to Sousa & Caldana (2023).

2.1 Node-RED

Node-RED is a tool that uses a visual approach based on message flows and blocks, known as nodes. This structure allows that the users connect pre-defined programmable blocks (or nodes) in an intuitive way. This feature allows for faster and easy development, When compared to traditional programing based on coded lines such as C/C++. (OpenJS Foundation & Contributors, 2013). The node provides different functions, such as to monitor the flow as the debug out node, or to read and write with GPIO pins of Raspberry Pi as the Raspberry Pi node. Created flows are stored using JSON (JavaScript Object Notation) (Lekic & Gardasevic, 2018)

2.1.1 JSON

In Node-RED, JavaScript Object Notation (JSON) is the main information Exchange format between nodes. All programming and configurations of the nodes is translated to JSON list. JSON is a textual representation of an object that is easily legible for humans and simple for machines to interpret. Even though it has "JavaScript" on its name is completely independent from that programming language and is a structure universally accepted with support to a great variety of human languages. Data on JSON is presented as a pair of name/value, that are comma separated and delimitated by Keys. For example, in Node-RED a message can be represented in JSON as: msg = (payload: *value*; msg.topir: *value*). This makes JSON a extremely useful tool for manipulating data in diverse contexts, including industrial automation. (Pezoa et al., 2016)

2.1.2 Messaging

Messages transmitted by and to nodes are named "msg" and are JavaScript's objects that contain proprieties the be processed and handled. Nodes can add Other Properties to the "msg" object that can be used to transmit Other relevant information to other nodes. The main Properties of the msg objects are:

- **msg.payload**: Contains the most important information to be handled, processed and manipulated
- **msg.topic**: Gives contextual information about the message

2.1.3 Flows, Nodes and Wires

A flow in Node-RED is a set of nodes connected by wires that exchange information between themselves, forming a program. Flows are described in an object list, detailing their nodes and connections. Nodes are the fundamental components of Node-RED, with well-defined functions and can be divided into 3 separate types as listed below. Node-RED also allow the programmer to build his own nodes for their specific needs, allowing for great customization. An example can be seen in, for example, the S7 communication nodes[1] that were used on this project or a DT Semantics node[2] set. Once in operation messages are inputted, produced, used and processed on nodes.

- **Input nodes:** Allow the input of data in an application, via user-interface, an API, or from reading a PLC for example.
- **Output nodes:** Allows the sending of data outside the flows, either to an OPC-UA server, a Dashboard, or a PLC for example.
- **Processing nodes:** Are used to process and manipulate data between input and output nodes.

Wires define the connections between input, output and processing nodes allowing the messages to be exchanged between them. It is possible to make multiple connections if necessary.

[1] https://flows.nodered.org/node/node-red-contrib-s7
[2] https://flows.nodered.org/node/@digitaltwins/node-red-contrib-digital-twin

2.1.3.1 Function Node

The function node, an essential component of Node-RED, offers the programmer the ability to manipulate messages using a JavaScript code. This allows an amazing flexibility, as it allows programmers to customize the behavior of the flow of information to meet the specific needs of the project. Besides, the function node can be used to transform, filter or combine messages in several different ways. For example, you can use the function node to extract the Topic of the message or any other specific information, as well as creating new messages depending on the content of the input.

2.1.3.2 Context

Context is a valuable resource of Node-RED that allows the import or export f data beyond the nodes. Its purpose is to store data in memory for future use (a key resource for historical analysis and big data driven decisions on DT). When configured it will store the flows information guaranteeing that the information won´t be lost once the Node-RED system is offline or the hardware is shut down. The resource is exclusive used by the Function Node and is divided in two distinct types (OpenJS Foundation & Contributors, 2024):

- **Local Context:** This type is visible to all the nodes in the same flow. It allows communication and sharing of data between local nodes only.
- **Global Context:** This type has a broader range, being visible to all nodes regardless of the flow they are in. They will allow communication and sharing of data between multiple flows.

2.1.4 Web Editor

The user interface of Node-RED is web-browser based as seen on Figure 2 below. It is a graphical display for the user to configure the application. It can be accessed by the link http://localhost:1880/ or http://127.0.0.1:1880/. The editor, as showed in Figure 2, is divided in 4 major components as described below:

- **Header**: Located on the top part of the screen, has the “Deploy” and Menu Button
- **Pallet**: On the left side, it contains all the available nodes that can be deployed. Additional libraries or proprietary nodes can be added.
- **Workspace**: The center of the screen, Where the flows are created and modified. It contains the tabs for multiple flows as well as the actual workspace for deploying Nodes and Wires.
- **Sidebar**: On the right side of the screen, has dropdown menus for Debug, Additional Information, Dashboards and etc.

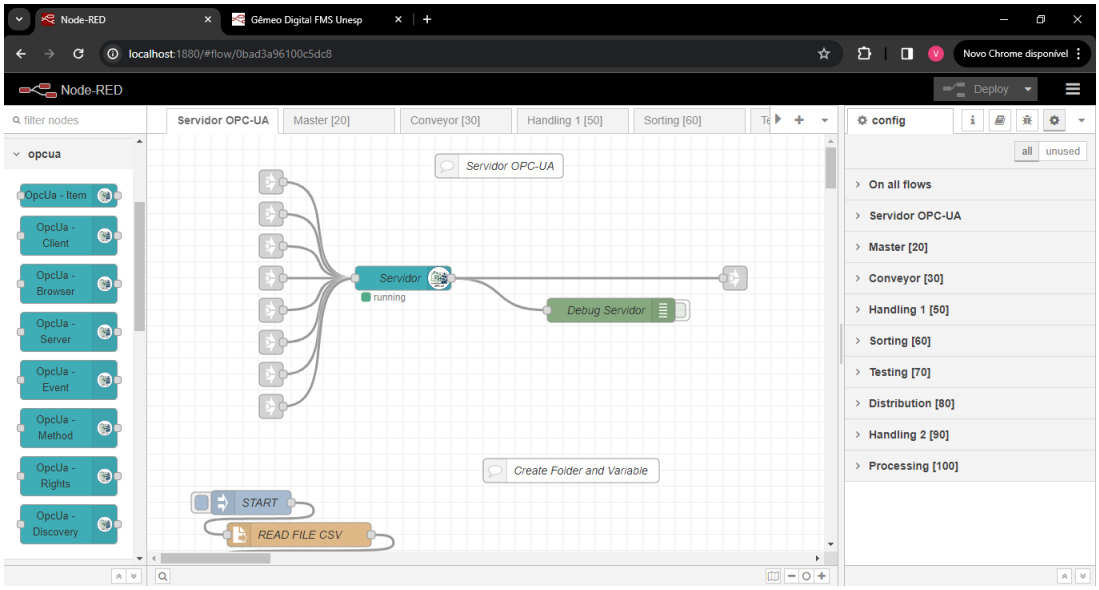

Figure 2: Node-RED Web Editor (The Authors)

3 Updated Node-RED flow

In this section we will detail the updated Node-RED flows from the first research (Sousa & Caldana, 2023) for both the Server and the Stations flows. The upgrade was a necessity once the new FMS stations were added, as there was an increase in the amount of data being transferred on the Project.

3.1 OPC-UA Server Flow

The OPC-UA Server flow was updated as showed in Figure 3 below, with the inclusion of the addition of the FMS Stations. On the left-hand side is the OPC-UA server and on the right-hand side is the OPC-UA Client used to write and read values to the Flows.

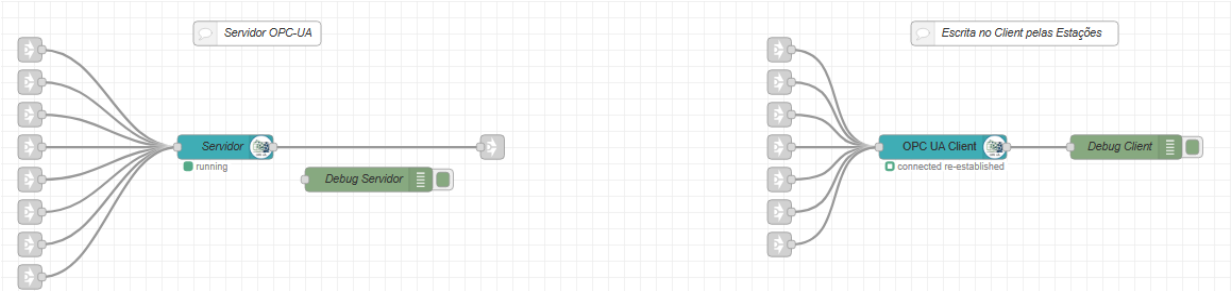

Figure 3: OPC-UA Server (The Authors)

3.1.1 Variables Structure

The most important updated is showed below in Figure 4 and Table 1. The folders and names of the variables for the FMS stations, instead of being created on each station flow, are now created on a single CSV file on the Server Flow.

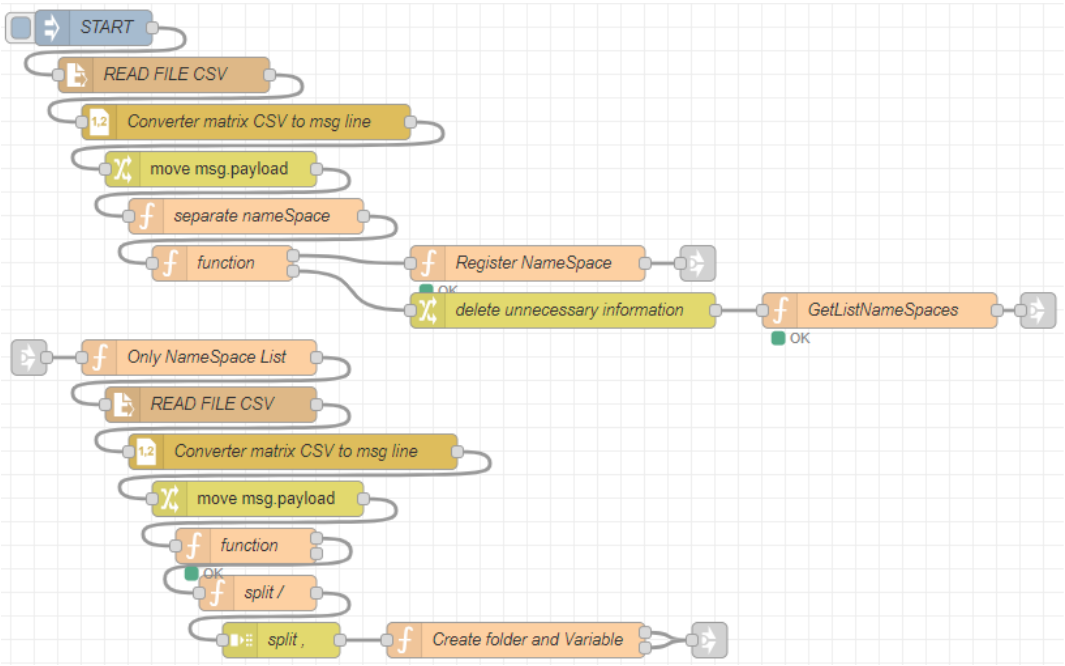

Figure 4: Folder Creation Structure (The Authors)

The automation developed in Node-RED was designed to facilitate the organization and handling of the variables in the OPC-UA environment. The creation of the folder structures allows for an efficient method to categorize and access similar types of information all at once. Additionally, the automated creation of variables simplifies the monitoring, maintenance and upgrading (if necessary) of the Whole structure within the OPC-UA Server

To achieve this automation, a CSV file was created and stored inside the main Project folder. At the end of the file, it is necessary to add %END in all columns so that the function creating the structure knows when the file is finished. This file (partially displayed in Table 1) is structured so each collum has the following information:

- **VariableName**: Defines the name of the variable to be created.
- **Function**: Describes what the variable do on the real FMS. Even though it has no impact on the program itself, this Semantic field helps end users to select the correct variable for other uses, such as DT 3D animation.
- **NameSpace**: Specifies the "namespace" of the Variable, that will be the folder in which that variable will be created in.
- **dataType**: Specifies the Type of Variable. So far, on the FMS, only Boolean types were required to describe the sensors and actuators.
- **initialValue**: Specifies the initial value that the Variable must have when created.
- **address**: Points to the specific location where the Variable will be stored, using "…" as a placeholder to point to the exact variable location.

Table 1 – Example of Variable Declarations on CSV file (The Authors)

VariableName	Function	NameSpace	dataType	initialValue	address
Q_50_Part_Reserve	Part in Cradle	Handling[50]	Boolean	False	root/stations/050-Handling1/$Inputs/...
Q_50_Claw_Cart	Claw in Cradle Position	Handling[50]	Boolean	False	root/stations/050-Handling1/$Inputs/...
Q_50_Claw_Del	Claw in Delivery Position	Handling[50]	Boolean	False	root/stations/050-Handling1/$Inputs/...
Q_50_Claw_Reserve	Half-way Indicator	Handling[50]	Boolean	False	root/stations/050-Handling1/$Inputs/...
Q_50_Claw_Low	Claw Down	Handling[50]	Boolean	False	root/stations/050-Handling1/$Inputs/...
Q_50_Claw_High	Claw Up	Handling[50]	Boolean	False	root/stations/050-Handling1/$Inputs/...
Q_50_Part_in_Claw	Part In Claw	Handling[50]	Boolean	False	root/stations/050-Handling1/$Inputs/...
Q_51_On	On	Handling[50]	Boolean	False	root/stations/050-Handling1/$Inputs/...
Q_51_Off	Off	Handling[50]	Boolean	False	root/stations/050-Handling1/$Inputs/...
Q_51_Key_Pos	Key Position	Handling[50]	Boolean	False	root/stations/050-Handling1/$Inputs/...
Q_51_Reset	Reset	Handling[50]	Boolean	False	root/stations/050-Handling1/$Inputs/...
Q_51_Panel_I4	Panel I4	Handling[50]	Boolean	False	root/stations/050-Handling1/$Inputs/...
Q_51_Panel_I5	Panel I5	Handling[50]	Boolean	False	root/stations/050-Handling1/$Inputs/...
Q_51_Panel_I6	Panel I6	Handling[50]	Boolean	False	root/stations/050-Handling1/$Inputs/...
Q_51_Panel_I7	Panel 17)	Handling[50]	Boolean	False	root/stations/050-Handling1/$Inputs/...

3.2 Station Flows

In the following section we present the updated flows for Input variables, output variables and operator interface. All the variables that used to be created in the station flow, as described in section 3.1, do not exist anymore as they are created now in the OPC-UA server flow with a CSV file.

3.2.1 Input Signals from PLC

The input variables were updated to reduce the number of connections as to enhance visualization as showed in Figure 5. This resulted in a significant reduction in nodes (approx. 60) for each station. The left collum reads the values from the PLC, the middle collum updates the into the OPC-UA server via the connection node and the right collum shows the information on the dashboard.

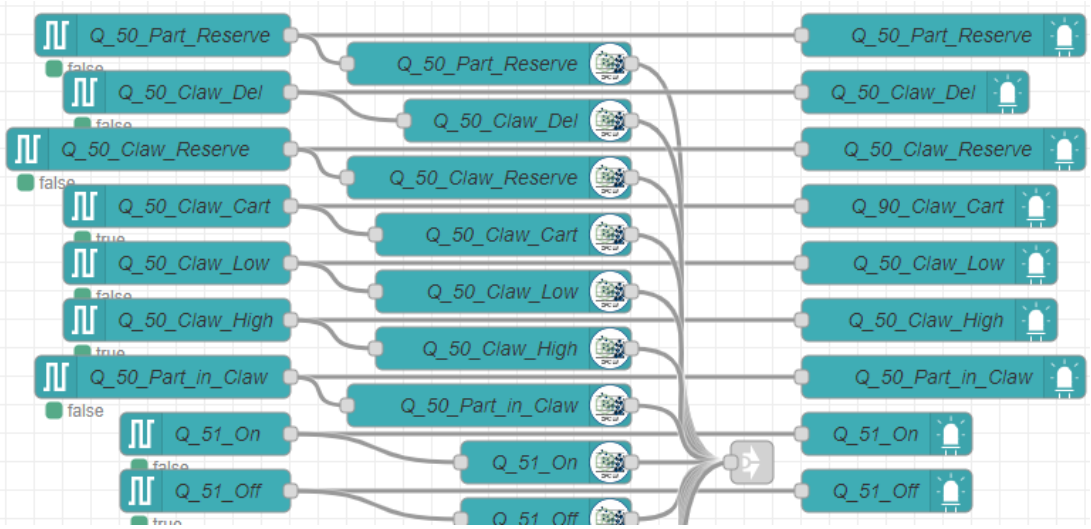

Figure 5: Updated Input Variables (The authors)

3.2.2 Output Signals to PLC

In this set of nodes, due to the better understanding of Node-RED, it was possible to transfer the programming to the function node and make it cleaner as showed in Figure 6. The left collum are the functions to select the correct variable and output the correct value (TRUE or FALSE) and the right collum is responsible to send the signals.

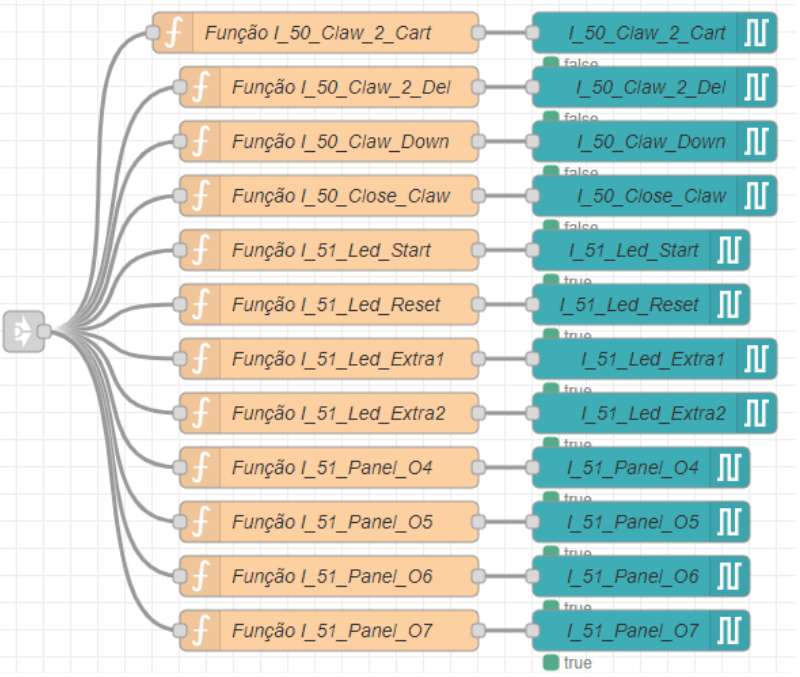

Figure 6: Output Signals to PLC (The Authors)

3.2.3 Dashboard Commands

Dashboard commands has been updated to allow several FMS stations. Buttons and visualizations were also updated, to reduce the total number of nodes and simplify visualization. Details are showed in Figure 7. The left collum are the Dashboard switches and the right collum are the OPC-UA Variables updates.

Figure 7: Dashboard commands (The Authors)

4 Results and Discussion

The process of flows development was very challenging, with the greatest difficulty on the lack of documentation on "node-red-contrib-opcua"[3] – the library that contains the OPC-UA nodes for Node-RED – since it was developed in 2015. The documentation provides a few examples and superficial explanations that were not sufficient to clarify the doubts while the programming was taking place during the development of the project. Also, the error management of the Debug screen from Node-RED of the OPC-UA Server is not particularly effective, as errors are not showed directly on the debug screen, with no flow return. This lack of feedback halts the implementation of more complex algorithms.

The optimization of the Node-RED programming due to the increase in knowledge was significant. The file with the complete project went from 870 kb to 542 kb, a reduction of 37,7% in size with the increased functions (the folder structure). The number of nodes was also significantly reduced – from 1516 nodes to 1061 nodes, a reduction of 30,0%. The difference between nodes reduction and file reduction is due to connectors and other components that were reduced.

As described by Clerissi et al. (2018), there was a great concern in making the final version of the software presented on this paper easy to use and update, as well as keeping it easy to be increased in terms of scale (new stations). With the presented structure of folders and the modular "stations by flow" in Node-RED, as well as the Server/Client component this is achieved. The simplification of the upgrades of the nodes, when compared to the first published results, are also pointing to a better solution that can be tested by peers.

In following subsections, we will present the results of the upgrades explained in section 3. This will be divided in two parts: The Dashboard and the OPC-UA Interface.

4.1 Dashboard Update

As a result of the addition of the stations of the FMS and the new programming of the dashboard as explained in section 3.2.1 and 3.2.3 the new dashboard is showed in Figure 8 below. On the left-hand menu you can select between

[3] https://flows.nodered.org/node/node-red-contrib-opcua

the several different modules of the FMS and on the main dashboard you have the feedback from the sensors (first left most 4 columns indicated by the red/green LED´s) and the possibility to manipulate the outputs (first 2 right most columns with the switches).

Figure 8: New Dashboard (The Authors)

4.2 OPC-UA Folders

With the structure explained on section 3.1.1 it is now possible to see the variable aggregated into not only the stations but also the type that they correspond. The division of folders, as seen on Figure 9 and Figure 10 below, shows the different variable that are now possible to be easily located and followed. The variables are divided into:

- **Inputs:** Describes the variables related to the sensors, such as limit switches, capacitive and inductive sensors
- **Outputs:** Describes the actuators of each station, such as motors or pneumatic actuators
- **Feedback:** Describes the status of the actuators.

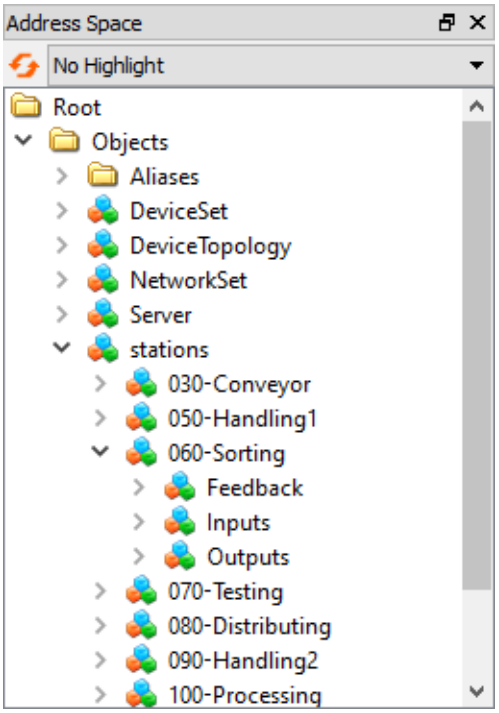

Figure 9: Station Structure (The Authors)

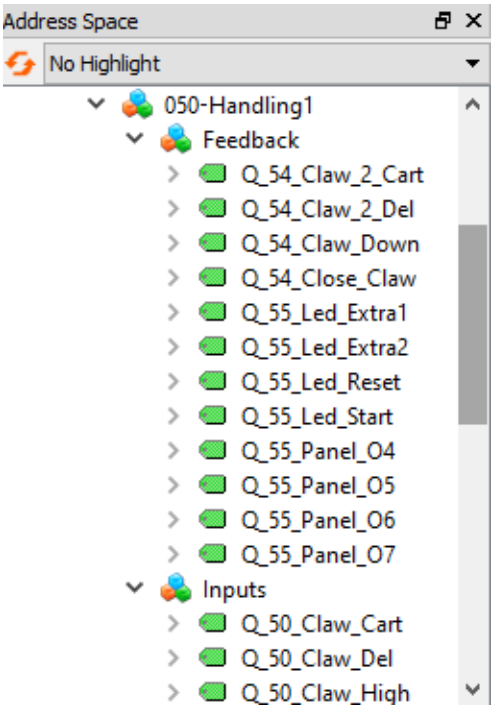

Figure 10: Variables in Folders (The Authors)

To maintain the same order as seen on the Dashboard of section 4.1, the first 3 digits that refer to the PROFIBUS address of each of the stations were added to the names so the order in the UaExpert software[4] would match the order of the PROFIBUS list. The final namespaces of the new folders can be seen below in Figure 11. It shows the number and the description of each namespace. The description is set on the CSV file described in Table 1.

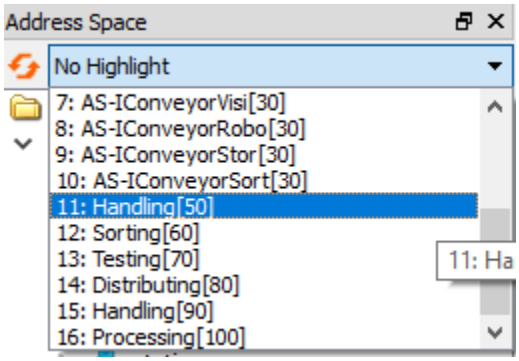

Figure 11: Folders namespace addresses and descriptions (The Authors)

5 Conclusion

During the execution of this project a better knowledge of Node-RED, OPC-UA, MQTT, Industrial protocols (specially PROFIBUS) and its diverse applications were obtained. This is showed by the review of section 2 and the advances on the programming from the first research. Several case studies and forum discussions that were not included in the references due to redundancies in base content, contributed to enhance the understanding of the programming languages. The results showed in sections 3.2.1, 3.2.2 and 3.2.3 are consistent with the objective a) of section 1.1 and is considered successfully achieved by the researchers, especially due to the reduction in both file size (37%) and nodes (30%). Objective b) of section 1.1 was also considered successfully achieved, as seen on the new updated Dashboard in section 4.1 and the folders and variables on OPC-UA in section 4.2.

The new method for creating the variables showed in section 3.1.1 allows for easy maintenance and upgrade if new stations are added. It also significantly reduced the number of nodes necessary and the size of the final project file, increasing its efficiency. As described in section 4.2, the variables are now divided into stations. This helps greatly in the understanding and mapping of the variables for future use as some variables are similar in all stations (For example the "On" signal), and with the new folder structure it is easier to select the correct variable. However, for extensive and complex projects Node-RED, even though flexible and easy to use, is still in need of maturing, specially in documentation and testing procedures. For now, an experimental role for the programming language is what is recommended.

Future work in this area will continue, with the integration of semantics DT information as described by Steinmetz et al. (2022) and the continuous improvement of the flows as needed.

References

Bajic, B., Rikalovic, A., Suzic, N., & Piuri, V. (2021). Industry 4.0 Implementation Challenges and Opportunities: A Managerial Perspective. *IEEE Systems Journal*, *15*(1), 546–559. https://doi.org/10.1109/JSYST.2020.3023041

Burns, T., Cosgrove, J., & Doyle, F. (2019). A Review of Interoperability Standards for Industry 4.0. *Procedia Manufacturing*, *38*, 646–653. https://doi.org/10.1016/j.promfg.2020.01.083

Clerissi, D., Leotta, M., Reggio, G., & Ricca, F. (2018). Towards an approach for developing and testing Node-RED IoT systems. *Proceedings of the 1st ACM SIGSOFT International Workshop on Ensemble-Based Software Engineering*, 1–8. https://doi.org/10.1145/3281022.3281023

Kutscher, V., Olbort, J., Anokhin, O., Bambach, L., & Anderl, R. (2020). Upgrading of legacy systems to cyber-physical systems. *Proceedings of TMCE 2020*. https://www.researchgate.net/profile/Vladimir-Kutscher/publication/346315936_UPGRADING_OF_LEGACY_SYSTEMS_TO_CYBER-

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

PHYSICAL_SYSTEMS/links/5fbe3ba8458515b7976b02c5/UPGRADING-OF-LEGACY-SYSTEMS-TO-CYBER-PHYSICAL-SYSTEMS.pdf

Lekic, M., & Gardasevic, G. (2018). IoT sensor integration to Node-RED platform. *2018 17th International Symposium INFOTEH-JAHORINA (INFOTEH)*, 1–5. https://doi.org/10.1109/INFOTEH.2018.8345544

OpenJS Foundation & Contributors. (2013). *Node-RED*. https://nodered.org/

OpenJS Foundation & Contributors. (2024). *Working with context: Node-RED*. https://nodered.org/docs/user-guide/context

Park, H. M., & Wook Jeon, J. (2019). OPC UA based Universal Edge Gateway for Legacy Equipment. *2019 IEEE 17th International Conference on Industrial Informatics (INDIN)*, *1*, 1002–1007. https://doi.org/10.1109/INDIN41052.2019.8972187

Pezoa, F., Reutter, J. L., Suarez, F., Ugarte, M., & Vrgoč, D. (2016). Foundations of JSON Schema. *Proceedings of the 25th International Conference on World Wide Web*, 263–273. https://doi.org/10.1145/2872427.2883029

Rikalovic, A., Suzic, N., Bajic, B., & Piuri, V. (2022). Industry 4.0 Implementation Challenges and Opportunities: A Technological Perspective. *IEEE Systems Journal*, *16*(2), Artigo 2. https://doi.org/10.1109/JSYST.2021.3101673

Schleich, B., Dittrich, M.-A., Clausmeyer, T., Damgrave, R., Erkoyuncu, J. A., Haefner, B., de Lange, J., Plakhotnik, D., Scheidel, W., & Wuest, T. (2019). Shifting value stream patterns along the product lifecycle with digital twins. *Procedia CIRP*, *86*, 3–11.

Singh, M., Fuenmayor, E., Hinchy, E. P., Qiao, Y., Murray, N., & Devine, D. (2021). Digital Twin: Origin to Future. *Applied System Innovation*, *4*(2), Artigo 2. https://doi.org/10.3390/asi4020036

Sousa, D. R. de, & Caldana, V. M. (2023). Node-RED for PLC Automation. *Proceedings of the 4th South American International Industrial Engineering and Operations Management Conference*.

Steinmetz, C., Schroeder, G. N., Sulak, A., Tuna, K., Binotto, A., Rettberg, A., & Pereira, C. E. (2022). A methodology for creating semantic digital twin models supported by knowledge graphs. *2022 IEEE 27th International Conference on Emerging Technologies and Factory Automation (ETFA)*, 1–7. https://ieeexplore.ieee.org/document/9921499/

Tabaa, M., Chouri, B., Saadaoui, S., & Alami, K. (2018). Industrial Communication based on Modbus and Node-RED. *Procedia Computer Science*, *130*, 583–588. https://doi.org/10.1016/j.procs.2018.04.107

Teixeira, A. F. S., Visoto, N. A. R., & Paulista, P. H. (2016). Automação industrial: Seus desafios e perspectivas. *Revista Científic@ Universitas*, *3*(2). http://revista.fepi.br/revista/index.php/revista/article/viewFile/404/278

Zielstorff, A., Schöttke, D., Hohenhövel, A., Kämpfe, T., Schäfer, S., & Schnicke, F. (2023). Harmonizing Heterogeneity: A Novel Architecture for Legacy System Integration with Digital Twins in Industry 4.0. Em S. Terzi, K. Madani, O. Gusikhin, & H. Panetto (Orgs.), *Innovative Intelligent Industrial Production and Logistics* (Vol. 1886, p. 68–87). Springer Nature Switzerland. https://doi.org/10.1007/978-3-031-49339-3_5

Biography

Daniel Ribeiro de Sousa is currently a student in the Electronics High-School Technical course of IFSP Sorocaba Campus, starting in 2021 and with graduation due in 2024. He has worked with Prof. Dr. Sérgio Shimura in 2021 on an extension project to instruct teachers at elementary schools of the city of Porto Feliz in microcontroller programming based in Arduino. Since 2022 is working with Prof. M.Sc. Vitor Caldana in the Node-RED for PLC Automation research project and since 2023 is a scholarship holder on the Scientific Initiation Program PIBIC Junior CNPQ/Reitoria Unesp in the Sorocaba Campus of UNESP.

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 and left the industry and FIEB to join 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.

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.