Centralised Architecture for the Automatic Self-Configuration of Industrial Networks

Inés Álvarez*, Daniel Bujosa*, Bjarne Johansson†, Mohammad Ashjaei*, Saad Mubeen*
*Mälardalen University, Sweden
†ABB, Västerås, Sweden
*{ines.alvarez.vadillo, mohammad.ashjaei, daniel.bujosa.mateu, saad.mubeen}@mdu.se
†bjarne.johansson@se.abb.com

***Abstract*—Novel production paradigms aim at increasing the efficiency and flexibility of production systems. Nonetheless, traditional industrial infrastructures lack the mechanisms needed to support these new paradigms. One of the main limiting factors is the architecture, which follows the automation pyramid in which subsystems are divided in layers depending on their functionalities. This allowed to meet the timing and dependability requirements of the production subsystems, however at the cost of limiting the exchange of information required to provide increased flexibility to the system. For this reason, in this paper we propose a new industrial architecture with a single network infrastructure to connect all the devices that conform to the industrial systems. On top of that, we design an Automatic Network Configurator to support the automatic configuration of the system. To assess the feasibility of our design and evaluate its performance, we implement the first instance of the architecture capable of supporting changes in the traffic requirements during run-time, i.e., without stopping or disrupting the system's operation. Furthermore, we use the implemented instance to measure the time required for reconfigurations.**

***Index Terms*—Self-Configuring Networks, Industrial Architecture**

I. Introduction

The emergence of new production paradigms, such as Industry 4.0 or Smart Factories, comes with the promise of more efficient, cost-effective, and flexible production systems. Nonetheless, traditional industrial communication infrastructures lack adequate mechanisms to support this change in paradigm. Industrial systems, in particular in the automation domain, are organized following the automation pyramid structure in which management (on the top of hierarchy) and production (on the bottom of hierarchy) subsystems are divided into planes, isolated from each other [1], as shown in Fig. 1. This structure allowed to meet the stringent requirements on reliability and timing predictability imposed by production systems, at the cost of low accessibility of data and increased complexity of network configuration [2].

Consequently, configuration of large industrial networks becomes unmanageable using the existing configuration techniques in the traditional paradigms [3]. This also limits the flexibility in the production plane. Due to these limitations, changes and reconfigurations in the current industrial communication infrastructures are rare and usually done from the top of the pyramid with human intervention. Furthermore, changes are mostly deployed offline to avoid undesired effects, which makes the changes very costly. This hampers with realization of the industry of the future with existing infrastructures.

To support the new production paradigms, characterized by flexibility and accessibility of data, there must be a shift in the way industrial communication infrastructures are designed and managed [4]. First, the management entities should have easy access to the information in the production plane, e.g., status and faults in network components like switches. Second, these entities should cater for the evolving resource requirements during the lifecycle of the systems, such as the need for allocation of new network resources (e.g., bandwidth or redundant switches). Third, the network configuration management should be automatized, removing human decisions from the loop to provide faster and more reliable (re)configuration decisions, leading to automatic self-configuring networks.

In this work, we propose a novel industrial architecture, in which the network becomes backbone of the system, enabling the integration of different planes in the automation pyramid into a single one. With the network at the center of the industrial infrastructure, its ability to adapt to changes will dictate the flexibility of the entire system. For this reason, we also provide the network with adequate mechanisms to adapt to changes at run-time, providing the timing predictability and reliability guarantees needed by the operational subsystems, without disrupting the correct operation of the rest of the system. Finally, we present the first instance of the proposed architecture which implements a specific monitoring and reconfiguration mechanism. This allows us to evaluate the feasibility of the proposed architecture, as well as to have the first measurement of the performance of the reconfiguration.

The remainder of the paper is organised as follows. Section II presents the related work. Section III describes the proposed architecture and its mechanisms; while Section IV describe our first instance of the architecture. Section V explains the experiment carried out and discusses the results,

This is a preprint version of a paper accepted in the 28th International Conference on Emerging Technologies and Factory Automation (ETFA 2023).

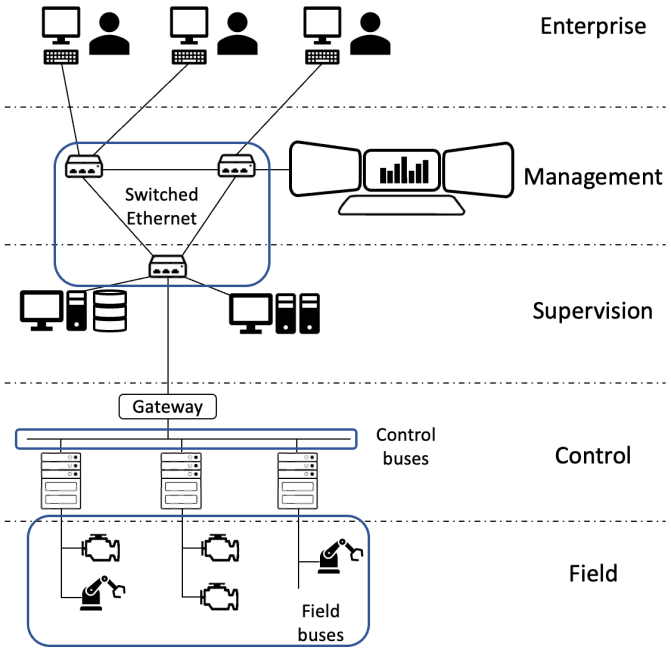

Fig. 1: Traditional industrial architecture.

while Section V-C highlights the most important aspects of our contribution. Finally Section VI concludes the paper.

II. Related Work

Contemporary industrial systems are structured following the automation pyramid, where different networks within the systems are often isolated from each other. In this structure, the management plane is separated from the operational plane, and the flow of information among these planes is limited [5]. This infrastructure limits the flexibility of the system to adapt to changes in a timely and cost-effective manner. There is a consensus on how industrial networks can benefit from the use of centralized solutions to ease their management (i.e., network monitoring and configuration). These solutions leverage a centralized entity that has a complete view of the network, which enables faster and efficient management of complex networks. This approach has been extensively used for the management of Information Technologies (IT), following the so-called Software Defined Networking (SDN) paradigm, which has proved to be adequate to manage the complexity of IT networks. For this reason, there is a wide interest in using SDN-like solutions in industrial communication networks [3], [6], [7], yet SDN lacks support for the configuration of real-time traffic [8], [9].

There are several works that address this limitation of SDN. The works in [10] and [11] explore how to extend SDN to support Flexible Time-Triggered (FTT) Ethernet, a protocol that provides Ethernet with real-time guarantees and online traffic configuration. Nonetheless, FTT-Ethernet has limited adoption by industry, limiting the applicability of these solutions. The work in [12] proposes an SDN-based solution to configure PROFINET. Unfortunately, this solution does not support configuration of other networks and, thus, does not support the heterogeneity of industrial networks. An SDN controller to schedule and configure time-triggered traffic over Ethernet/IP networks is proposed in [13]. However, this solution does not support real-time event-triggered traffic.

Some works aim at providing mechanisms to enable self-configuring networks. The work in [14] discusses the adequacy of existing topology discovery protocols and the use of Open Network Operating System (ONOS) to carry out the topology discovery in legacy networks using SDN. Nonetheless, the work does not support the configuration of the network. The work in [15] proposes an SDN-like approach to reduce the fail-over time in networks after the failure of a network device. However, this work does not adequately support network traffic with real-time requirements.

Parallel to this trend, there is a plethora of research on the use of Time-Sensitive Networking (TSN) standards[1] developed by the IEEE for industrial applications. TSN is a promising technology to accommodate all the needs of future industrial networks, which also relies on a centralized architecture to provide the network with enhanced configuration capabilities [16]. A key element in such architecture is the Centralized Network Configuration (CNC) element, but its specification is not covered by the standards. A conceptual framework to monitor and reconfigure TSN networks using the fully centralised architecture is proposed in [17]. However, this work does not provide implementation of the CNC. The work in [18] presents heuristic approach to configure TSN networks at run-time, but does not discuss the implementation details of the CNC.

The works in [19]–[21] propose mechanisms to schedule time-triggered traffic and deploy the configuration in the network, but do not address the configuration of event-triggered traffic. Whereas, mechanisms to monitor and configure time-triggered traffic in TSN networks are proposed in [22], however, the event-triggered traffic is not supported. In crux, these works do not support contemporary and next-generation heterogeneous network technologies.

The state of the art on self-organizing manufacturing systems (SOMS) is reviewed in [23]. Even though the term was coined in 1994, a complete solution to support the realization of the SOMS is still missing. In the last few years, many efforts have been made in defining standard interfaces to enable the SOMS. There have also been several works to propose mechanisms to support self organization of different aspects of industrial systems, however, there is no complete solution in this regard.

The review of the state of the art shows that there are several attempts to address the challenges of (re)configuration of industrial networks. However, these solutions are limited in various aspects. First, the existing SDN-based solutions focus

[1] https://1.ieee802.org/tsn/

only on the configuration of specific protocols, lack support for heterogeneous networks, and more general solutions cannot support event-triggered real-time traffic (which is integral to industrial systems). Second, the CNC-based solutions are customized for TSN networks with specific tasks and their implementations are either lacking or not available publicly. In this work we target to fill the research gap in the configuration of heterogeneous industrial networks by proposing a complete network architecture that leverages some of the above mentioned techniques [19]–[21] to support the self-configuration of industrial networks. The proposed architecture allows configuration during run-time while guaranteeing timing predictability and reliability of the entire heterogeneous network.

III. Architecture Overview

As we have discussed, traditional industrial architectures divide the infrastructure in planes, as shown in Fig. 1. The different planes are isolated from each other to meet the stringent real-time and dependability requirements of the control and field devices. However, this isolation limits the flexibility of the system to adapt to changes, which clashes with novel production paradigms centered in customization and constant change. To support new production paradigms we need to propose new architectures capable of meeting the timing and dependability requirements of control and field devices, while enabling flexibility to respond to the needs of the enterprise. Such an architecture resembles the concept of Industrial Internet of Things (IIoT) development in various industries [24], while it is not the same as in our proposed architecture we consider the network to be at the center of the architecture and in IIoT architecture commonly data is at the center of design choices. Following, we present the network-centric architecture and a component that performs the automatic network configuration.

A. The Network-Centric Approach

We propose to move from a hierarchical architecture to a network-centric architecture, where the network becomes the backbone of the industrial infrastructure, as depicted in Fig. 2. In this way, we ease the exchange of information among the different entities of the architecture, which is key to manage the resources in an efficient and flexible manner.

The ability of such a centralized network to dynamically adapt to changes can be leveraged to support the flexibility of the entire system. Our vision is to support the network with novel configuration mechanisms to automatically adapt to changes at run-time, providing the timing predictability and reliability guarantees needed by the operational subsystems, with minimum disruption to the correct operation of the entire system. We aim at realizing this vision by developing novel techniques to support the Automatic Network Configurator, depicted in Fig. 2, which will have the complete view of the network capabilities and requirements.

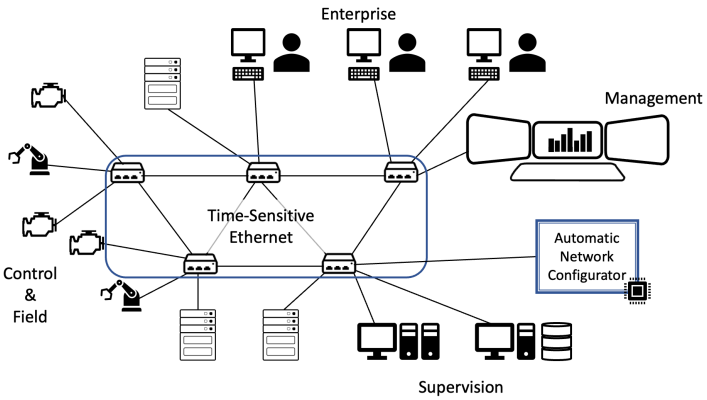

Fig. 2: Proposed industrial architecture.

According to the architecture depicted in Fig. 2, the network is a unified communication channel among various nodes including the field level devices (e.g., control devices), the control-level devices, the supervision and monitoring devices, and the top-level management and enterprise devices. Such an architecture allows data transparency among different devices in the factory as well as flexibility of the entire communication system. Such a unified and converged communication is envisioned by the recent advancements in IEEE task forces, i.e., TSN standardization task force[2]. Although the concept of network configuration mechanism is not limited to any specific industrial network technology, for the sake of argumentation and simplicity we present the network as TSN network. We also consider a component named Automatic Network Configurator that is connected to the network. This component is responsible for monitoring the entire network, decide on automatic configuration when needed, and to deploy new configurations. The details of this component will be discussed in the next section.

Within the above-mentioned context, we envision the TSN standards to be the underlying network technology of our proposed industrial architecture. TSN represents a promising technology to support the integration of the Information Technologies (IT) and Operation Technologies (OT), by supporting traffic with different characteristics and requirements to be transmitted over a single network infrastructure. Furthermore, TSN proposes the use of a fully centralised network architecture to ease the management of the communications during run-time, which aligns with our proposed architecture. Even though the TSN task force does not provide mechanisms to carry out the reconfiguration of the network in an autonomous manner, it does provide mechanisms in the bridges that enable their remote configuration, simplifying the realization of our architecture.

In addition, to provide interoperability in such a unified network, we leverage the use of OPC UA which is an IEC standard that aims at providing better interoperability of industrial

[2]https://1.ieee802.org/tsn/

systems. Specifically, OPC UA defines data models for data exchange between a great variety of industrial systems. This enables interoperability, but also support for easy reconfiguration of the communication. Furthermore, we propose to use broker-less OPC UA PubSub (Publisher-Subscriber) for time-sensitive control and field traffic. Together, TSN and OPC UA PubSub can provide adequate support for the integration of IT and OT traffic [25] with different timing requirements.

B. The Automatic Network Configurator

As we have discussed, we envision our network-centric industrial infrastructure to leverage a central element, capable of reconfiguring the communications during run-time in an automatic manner. This device, which we call the Automatic Network Configurator in Fig. 3, has the complete view of the network status and system requirements. This provides it with a strategic advantage to support changes in the communications in an efficient and cost-effective manner.

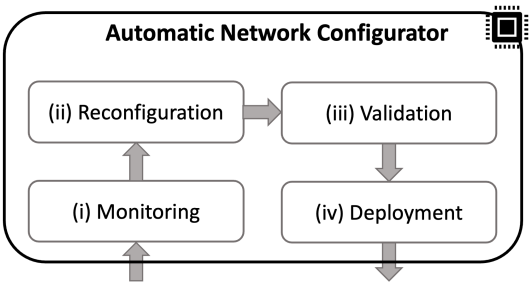

Fig. 3: Automatic Network Configurator mechanisms.

The Automatic Network Configurator includes the following mechanisms, depicted in Fig. 3: (i) automatic network monitoring, (ii) automatic network configuration and dynamic reconfiguration, (iii) feasibility and predictability verification of the configuration, and (iv) configuration deployment. The Monitoring component is responsible for monitoring and gathering information about the status of the network including the occurrence of faults, changes in the topology or changes in the bandwidth utilization, to name a few, and report this to the second component, Reconfiguration. Then, the Reconfiguration component utilizes the collected information about the network status and decides whether any change in the network is required. These changes include port configuration, bandwidth allocation to specific traffic services, re-routing of traffic due to identified faults, or handling new connected nodes and traffic.

Once a decision has been made by the Reconfiguration component, the new changes should be validated before they are deployed. For instance, if a change is decided on the bandwidth allocation, the consequences of that on the timing predictability of traffic has to be verified. Moreover, if a new routing should be imposed, the effects on the latency of traffic should be evaluated. When the changes are validated using the Validation component, the new changes will be deployed by the Deployment component. The main responsibility of the latter component is to apply the changes on the network with minimum disruption on the functionality of the network. Therefore, specific mechanisms are required to perform such a deployment. Note that this is the overall functionality for the Automatic Network Configurator which is not defined and customized for a specific configuration. Obviously, different mechanisms should be designed and evaluated for every type of changes that are considered in the network.

In order to show the feasibility and plausibility of the proposed architecture and the Automatic Network Configurator, in the following sections we describe how we instantiate the solution for a specific case, and we evaluate its performance.

IV. The first Instance of the Architecture

In order to prove the feasibility of our proposal, as well as to study its adequacy, we have implemented the first instance of our architecture. The objective of this implementation is to provide timing guarantees to time-triggered (TT) traffic transmitted through a TSN network in the presence of changes. To that, our system monitors the traffic to detect deviations from the planned schedule and calculates a new schedule to ensure that the TT traffic meets its deadlines.

As we have discussed, we use TSN as the network technology, and the Automatic Network Configurator is implemented within TSN's CNC. Specifically, we have extended the CNC presented in [21] with the mechanisms required to support the modification of the schedule during run-time. Fig. 4 shows the instance of the Automatic Network Configurator within the CNC, where gray boxes represent the components implemented in previous works, while black boxes represent the components implemented in this work. Furthermore, in the boxes with dashed lines we can see to which mechanism of our architecture the modules correspond to.

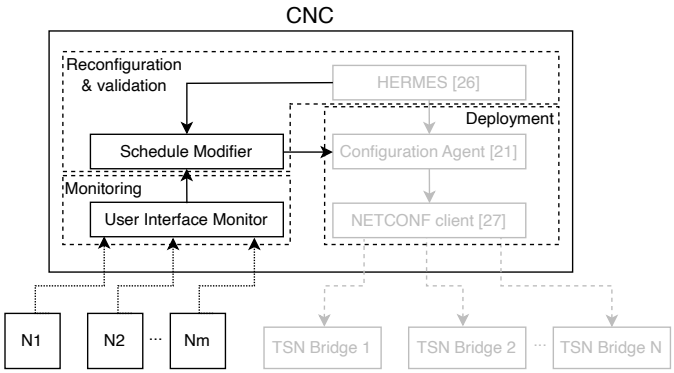

Fig. 4: Instance of the Automatic Network Configurator within the CNC.

If we focus on the right side of the figure, we see that the CNC is responsible for creating a valid schedule for the TT traffic using the HERMES scheduler [26]. This schedule is sent to the configuration agent [21], which in turn converts it to

the so-called Gate Control Lists (GCLs), which dictates when the transmission queues of the TSN bridges can transmit the frames enforcing the schedule. After that, the CNC deploys the new GCLs in all the involved bridges using the NETCONF [27] protocol, which is also responsible for activating the new configuration in the bridges in a coordinated manner.

If we now focus on the bottom-left part of the figure, we see the User Interface Monitor (UIM), which receives information from the nodes about their traffic requirements, and uses such information to decide whether any changes are required in the existing schedule. If any changes are required, the UIM provides the updated information about the traffic requirements to the Schedule Modifier module, which uses the information to decide whether a new schedule is required, to calculate a new schedule when needed and to validate it. We can also see that the Schedule Modifier uses the schedule calculated by HERMES as input to recalculate the schedule. We must note that the changes requested by nodes are related to the timing needs of the traffic they exchange, e.g., changing the period of a stream. If these changes are simple enough to not require rescheduling or rerouting the whole traffic again, the Schedule modifier produces the new schedule, validates it and forwards it to the Deployment mechanism. Otherwise, HERMES would produce a new schedule.

We next describe how we have validated the design of our architecture, as well as how we have evaluated its performance.

V. Experimental Evaluation

We have implemented an instance of the proposed architecture to show the feasibility of our design, as well as to measure its performance. We next describe the setup used, as well as the experiments carried out and the results acquired.

A. Experimental Setup

We have used the topology shown in Fig. 5, which is formed by four nodes, labelled in the figure using Ni where i is the node number, and four bridges, labelled using Bj where j is the bridge number, connected in a ring topology. We also have a CNC, connected to the network through a single bridge. Note that the CNC can communicate with all TSN bridges and the end nodes using the same network infrastructure used by the application.

In our implementation, nodes exchange time-triggered (TT) traffic, and they are implemented using Raspberry Pi, while Bridges are TSN bridges responsible for enforcing the timing guarantees of the TT traffic with the help of the CNC. The CNC is implemented in software and runs in a docker container over a PC running Windows 11.

In our experiments, we send two different streams, one from node N1 to N2 and another from N3 to N4. These streams have a period of 1 second each, and such period is reduced every 100 frames. As we have discussed, nodes are responsible for communicating the changes in the traffic requirements to the CNC, so every time the period is reduced, the node indicates

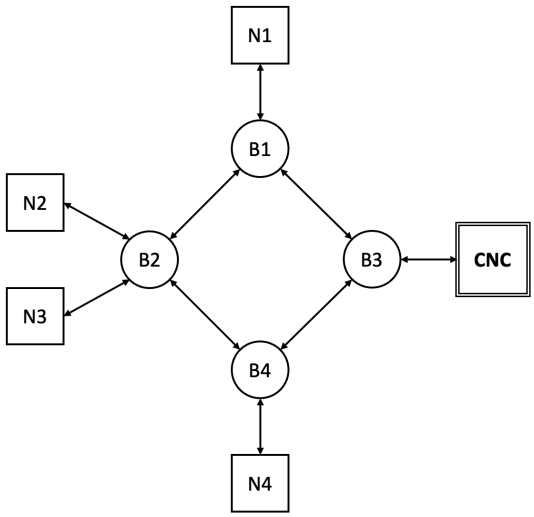

Fig. 5: Topology used in the experimental evaluation.

the new period to the CNC, triggering the reconfiguration process. We must note that the transmission of the traffic requirements is done using standard TSN frames with the same network that is used by the nodes to communicate among them, e.g., if node N3 needs to send information to the CNC it does so through bridges B2, B4 and B3.

In our implementation, the transmission of the new traffic requirements is implemented in software in each node. In this sense, the monitoring can be considered to be partially distributed, as it is performed jointly by the CNC and the nodes. Implementing part of the monitoring as software in the nodes allows us to monitor the traffic requirements in a cost-effective manner, as it does not require the addition of extra hardware in the network nor the end nodes. Furthermore, it allows using commercial off-the-shelf bridges, reducing the cost of the network and guaranteeing interoperability.

When the CNC receives information about new traffic requirements from an end node it uses such information to decide whether the schedule needs to be modified and, if so, to modify it adequately. Once the new schedule is calculated, it is transformed into gate control lists (GCLs), and deployed in the network using NETCONF messages, encapsulated in Ethernet frames. In turn, bridges process the NETCONF messages, change their configuration adequately, and notify the CNC about the change.

Using the aforementioned setup, we have been able to successfully reconfigure the traffic exchanged in our network during run-time. In this way, we have successfully assessed the feasibility of our designed architecture in supporting changes in the traffic without interrupting the ongoing communications. We next evaluate the performance of this first implementation in terms of the time required to carry out the reconfiguration.

B. Description of the Experiment and Results

We use the topology shown in Figure 5 to measure the time required to carry out the reconfiguration of the network. Specifically, we measure the time that elapses since a node changes the period of a stream until the bridges notify the CNC that the configuration has been successfully implemented. Thus, we account for the time required to encapsulate the information related to the traffic, send the information to the CNC, process such information, calculate a new configuration, send the new configuration to the bridges, and receive the confirmation from bridges that the new configuration is implemented.

We have transmitted a total of 4000 frames, 2000 frames through each stream. Since the period is modified every 100 frames, we have measured 40 reconfigurations. Figure 6 shows the times required to reconfigure the network traffic. The X axis show the configuration time in seconds, divided in intervals of 0.1 second; the blue Y axis show the number of experiments that have a configuration time within a certain interval, while the red Y axis show the probability distribution of the configuration times.

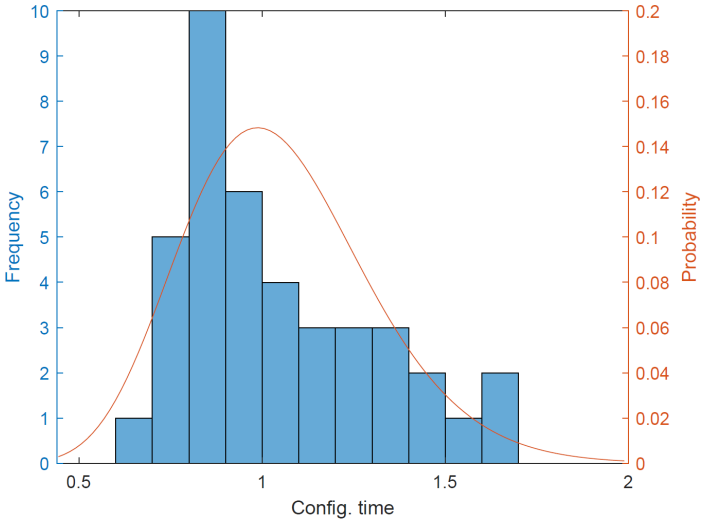

Fig. 6: Time required to carry out the reconfiguration of the network traffic.

As we can see, the measured reconfiguration times range from 0.6 to 1.7 seconds. This can be due to the fact that our implementation has been done using non-real-time operating systems. Furthermore, the frames conveying monitoring and configuration information are treated as best effort traffic with no specific timing requirements. We must note that the main objective of this implementation is to prove the adequacy of the designed architecture to support adaptive systems with changing traffic requirements during run-time. For this reason, the implementation has not been optimized to minimize the reconfiguration time. In any case, we note that in current industrial systems this type of changes are typically done offline, and with human intervention, and can take from hours to days to be realized. Therefore, the results represent a significant improvement compared to the times required nowadays to perform changes in the communications in existing industrial infrastructures.

C. Discussion

In Section III we have designed a new industrial architecture with the objective of increasing the flexibility of industrial infrastructures and, in this way, supporting novel production paradigms. This architecture is characterised by the use of a single network infrastructure which is managed in an automatic manner by a centralised component called Automatic Network Configurator. In order to prove the feasibility of our design and evaluate its adequacy, we have implemented the first instance of our Automatic Network Configurator, described in Section IV. This instance implements a single reconfiguration mechanism, a simplification required to enable the evaluation of our concept without increasing the complexity of the evaluation. This instance is sufficient to prove the adequacy of the design, as it encompasses all the mechanisms envisioned to support the reconfiguration of the network: monitoring, reconfiguration, validation and deployment.

Furthermore, the results obtained in the experimental evaluation prove that the architecture can be used to support the automatic reconfiguration of the network during run-time. For this reason, this implementation represents a good base to build other mechanisms to support the adaptation of the network due to other reasons, e.g., due to the presence of faults or the addition of new components in the system.

VI. Conclusions

Novel industrial paradigms such as Industry 4.0, promise to bring efficiency and flexibility to the production systems. Nonetheless, traditional industrial infrastructures cannot support this change in paradigm. For this reason, we have presented a new industrial architecture that can increase the flexibility of the system by putting the network in the core of the industrial infrastructure. Specifically, we propose to use a converged network topology to connect all the entities of our industrial architecture, which eases the exchange of information, enabling an efficient and flexible management of resources.

Furthermore, the proposed architecture counts with an Automatic Network Configurator, an element responsible for reconfiguring the communications during run-time in an automatic manner. This element implements the mechanisms to manage the network in a centralised manner, taking advantage of the complete knowledge of the network status and requirements to make complex decisions in an efficient manner. We propose to divide the operation of this element into four mechanisms: monitoring, reconfiguration, validation and deployment.

In order to prove the feasibility of our design, we have implemented the first instance of our architecture, which

supports the modification of the traffic requirements during run-time. Specifically, our instance allows for nodes to request changes in the traffic they exchange, e.g., the period of a stream. We use this instance to modify the requirements of two streams during run-time, and we have measured the time required to carry out the reconfiguration of the network when a change is requested. The results obtained allow us to prove that the designed architecture is adequate to support the automatic reconfiguration of the network during run-time. Furthermore, the results also show a significant improvement in the time required to reconfigure the network when compared to the current methods to implement changes in industrial infrastructures, which usually require changes to be done offline, disrupting the operation of the infrastructure.

ACKNOWLEDGEMENT

This work is supported by the Swedish Governmental Agency for Innovation Systems (VINNOVA) via the DESTINE, PROVIDENT and INTERCONNECT projects and by the Swedish Knowledge Foundation through the projects DPAC, HERO and FIESTA. The authors also thank all industrial partners, especially ABB, Arcticus Systems, HIAB and Volvo Construction Equipment, Sweden for valuable input.

REFERENCES

[1] T. Williams, "A Reference Model for Computer Integrated Manufacturing from the Viewpoint of Industrial Automation," *IFAC Proceedings Volumes*, vol. 23, no. 8, Part 5, pp. 281–291, 1990, 11th IFAC World Congress on Automatic Control, Tallinn, 1990 - Volume 5, Tallinn, Finland. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1474667017517486

[2] S. Vitturi, C. Zunino, and T. Sauter, "Industrial communication systems and their future challenges: Next-generation ethernet, iiot, and 5g," *Proceedings of the IEEE*, vol. 107, no. 6, pp. 944–961, 2019.

[3] K. Ahmed, J. O. Blech, M. A. Gregory, and H. W. Schmidt, "Software Defined Networks in Industrial Automation," *Journal of Sensor and Actuator Networks*, vol. 7, no. 3, 2018.

[4] B. Bajic, A. Rikalovic, N. Suzic, and V. Piuri, "Industry 4.0 Implementation Challenges and Opportunities: A Managerial Perspective," *IEEE Systems Journal*, vol. 15, no. 1, pp. 546–559, 2021.

[5] "Enterprise-Control System Integration - Part 1: Models and Terminology." 2010.

[6] M. Herlich, J. L. Du, F. Schörghofer, and P. Dorfinger, "Proof-of-concept for a software-defined real-time Ethernet," in *21st International Conference on Emerging Technologies and Factory Automation*, 2016.

[7] L. Silva, P. Pedreiras, P. Fonseca, and L. Almeida, "On the adequacy of SDN and TSN for Industry 4.0," in *IEEE 22nd International Symposium on Real-Time Distributed Computing (ISORC)*, 2019, pp. 43–51.

[8] D. Henneke, L. Wisniewski, and J. Jasperneite, "Analysis of realizing a future industrial network by means of Software-Defined Networking (SDN)," in *IEEE World Conference on Factory Communication Systems*, 2016, pp. 1–4.

[9] M. Ehrlich, D. Krummacker, C. Fischer, R. Guillaume, S. S. Perez Olaya, A. Frimpong, H. de Meer, M. Wollschlaeger, H. D. Schotten, and J. Jasperneite, "Software- Defined Networking as an Enabler for Future Industrial Network Management," in *2018 IEEE 23rd International Conference on Emerging Technologies and Factory Automation (ETFA)*, vol. 1, 2018, pp. 1109–1112.

[10] C. Ternon, J. Goossens, and J.-M. Dricot, "FTT-OpenFlow, on the Way towards Real-Time SDN," *SIGBED Rev.*, vol. 13, no. 4, p. 49–54, nov 2016. [Online]. Available: https://doi.org/10.1145/3015037.3015045

[11] L. Moutinho, P. Pedreiras, and L. Almeida, "A Real-Time Software Defined Networking Framework for Next-Generation Industrial Networks," *IEEE Access*, vol. 7, pp. 164 468–164 479, 2019.

[12] K. Ahmed, J. O. Blech, M. A. Gregory, and H. Schmidt, "Software defined networking for communication and control of cyber-physical systems," in *2015 IEEE 21st International Conference on Parallel and Distributed Systems (ICPADS)*, 2015, pp. 803–808.

[13] N. G. Nayak, F. Dürr, and K. Rothermel, "Time-Sensitive Software-Defined Network (TSSDN) for Real-Time Applications," in *Proceedings of the 24th International Conference on Real-Time Networks and Systems*, ser. RTNS '16. New York, NY, USA: Association for Computing Machinery, 2016, p. 193–202. [Online]. Available: https://doi.org/10.1145/2997465.2997487

[14] S. K. Panda, M. Majumder, M. Ehrlich, A. Neumann, L. Wisniewski, and J. Jasperneite, "Topology Detection as a Base for Efficient Management of Heterogeneous Industrial Network Systems Using Software-Defined Networking," in *15th IEEE International Workshop on Factory Communication Systems*, 2019, pp. 1–8.

[15] N. N. Josbert, W. Ping, M. Wei, and Y. Li, "Industrial Networks Driven by SDN Technology for Dynamic Fast Resilience," *Information*, vol. 12, no. 10, 2021.

[16] "IEEE Standard for Local and Metropolitan Area Networks–Bridges and Bridged Networks – Amendment 31: Stream Reservation Protocol (SRP) Enhancements and Performance Improvements," *IEEE Std 802.1Qcc-2018 (Amendment to IEEE Std 802.1Q-2018 as amended by IEEE Std 802.1Qcp-2018)*, pp. 1–208, Oct 2018.

[17] M. Gutiérrez, A. Ademaj, W. Steiner, R. Dobrin, and S. Punnekkat, "Self-Configuration of IEEE 802.1 TSN Networks," in *22nd IEEE International Conference on Emerging Technologies and Factory Automation (ETFA)*, 2017, pp. 1–8.

[18] M. L. Raagaard, P. Pop, M. Gutiérrez, and W. Steiner, "Runtime Reconfiguration of Time-Sensitive Networking (TSN) Schedules for Fog Computing," in *2017 IEEE Fog World Congress*, 2017, pp. 1–6.

[19] T. Gerhard, T. Kobzan, I. Blöcher, and M. Hendel, "Software-defined Flow Reservation: Configuring IEEE 802.1Q Time-Sensitive Networks by the Use of Software-Defined Networking," in *24th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA)*, 2019, pp. 216–223.

[20] A. Garbugli, A. Bujari, and P. Bellavista, "End-to-end QoS Management in Self-Configuring TSN Networks," in *17th IEEE International Conference on Factory Communication Systems*, 2021, pp. 131–134.

[21] I. Álvarez, A. Servera, J. Proenza, M. Ashjaei, and S. Mubeen, "Implementing a First CNC for Scheduling and Configuring TSN Networks," in *IEEE 27th International Conference on Emerging Technologies and Factory Automation (ETFA)*, 2022, pp. 1–4.

[22] N. S. Bülbül, D. Ergenç, and M. Fischer, "SDN-based Self-Configuration for Time-Sensitive IoT Networks," in *2021 IEEE 46th Conference on Local Computer Networks (LCN)*, 2021, pp. 73–80.

[23] Z. Qin and Y. Lu, "Self-organizing manufacturing network: A paradigm towards smart manufacturing in mass personalization," *Journal of Manufacturing Systems*, vol. 60, pp. 35–47, 2021. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0278612521000960

[24] B. Craggs, A. Rashid, C. Hankin, R. Antrobus, O. Şerban, and N. Thapen, "A reference architecture for iiot and industrial control systems testbeds," in *Living in the Internet of Things*, 2019, pp. 1–8.

[25] P. Denzler, M. Ashjaei, T. Frühwirth, V. N. Ebirim, and W. Kastner, "Concurrent OPC UA information model access, enabling real-time OPC UA PubSub," in *2022 IEEE 27th International Conference on Emerging Technologies and Factory Automation (ETFA)*, 2022, pp. 1–4.

[26] D. Bujosa, M. Ashjaei, A. Papadopoulos, T. Nolte, and J. Proenza, "HERMES: Heuristic Multi-queue Scheduler for TSN Time-Triggered Traffic with Zero Reception Jitter Capabilities," in *The 30th International Conference on Real-Time Networks and Systems*, June 2022. [Online]. Available: http://www.es.mdh.se/publications/6446-

[27] R. Enns, M. Björklund, J. Schönwälder, and A. Bierman, "Network Configuration Protocol (NETCONF)," *RFC 6241*, June 2011.