http://dx.doi.org/10.25673/116048

Modelling of NOA 178 Verification of Request concept using Pro-Active Asset Administration Shells

H.K. Pakala*, Ch. Diedrich*

**Otto von Guericke University Magdeburg, Magdeburg, LSA 39106 Germany (e-mail: harish.pakala@ovgu.de) Germany (e-mail: christian.diedrich@ovgu.de).*

Abstract:

The NAMUR automation pyramid is the current state of the art in the process control industry. The pyramid is a monolithic structure with information flowing either in top-down or bottom-up directions. Introducing new technologies into this structure would imply modifying or replacing existing equipment and the control software. NE 175 opens up this closed structure with a new open architecture (NOA) to provide access to external interfaces without changing the existing structure safely, securely and reliably. In continuation of the NOA, the NE 178 presents an architecture for the verification of the request concept (VOR) to allow plant-specific M+O applications to make requests for a change in functioning of the plant by-passing the strong hierarchy of the pyramid. This paper attempts to model the actors involved in this concept and their interactions using the Pro-Active AAS concept, it introduces a new interaction protocol and demonstrates the concept using the single water tank case study.

Keywords: automation systems, cyber physical systems, Asset Administration Shell, NOA, verification of request, automation pyramid

1. Introduction

Normen-Arbeitsgemeinschaft für Mess-und-Regeltechnik in der Chemischen Industrie (NAMUR) automation pyramid is the traditional, well-accepted and proven structure within the process control industry. This structure emphasizes and ensures a hierarchical information flow from the field devices at the bottom to the MES at the top, as shown in Figure 1. The field devices, programmable logic controllers, SCADA and the MES applications within the four levels of the pyramid, as pointed out in Figure 1, are tightly coupled and interrelated. Introducing new technologies into the existing plant would require modifications at all levels by the appropriate manufacturers, involving huge man-hours and financial investment by all the players involved.

Within the Information technology (IT) industry during the last two decades, there were significant advancements like better user experience interfaces, virtual reality, blockchain, enterprise integration, ontologies and the semantic web, automation verification of the systems, Chat GPT and many more. Combined with the Artificial Intelligence sectors like energy, health, transportation and supply chain are integrating and benefitting from the IT advancements. The article (Klettner et al. 2017) points out that within the automation sector in reference to the NAMUR ecosystem, there is no significant adoption of IT technologies. In this context, one of the NAMUR working groups has published the (NE 175), which introduces the concept of NAMUR open architecture (NOA). NOA is safe, secure, and reliable and intends to open up the automation pyramid to introduce the latest features and products from the IT industry. The (NE 175) professes that NOA is open, but the pyramid structure stays intact and the scope for the new technologies is rather parallel to the existing structure.

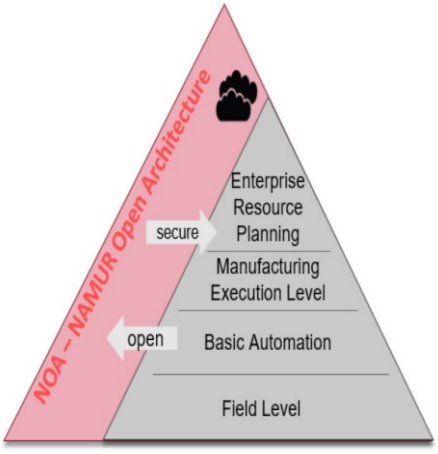

Figure 1 NOA Pyramid (Klettner et al. 2017)

NE 178 further strengthens the idea of the autonomy of the plant and the plant owner and introduces an architecture for the NOA Verification of Request (VOR) concept. The actual document for NE 178 is in draft stage and the details of the VOR concept are published in (Iatrou et al. 2022). It is the systems of systems architecture for the transmission of information between the plant and the interfaces of new IT infrastructure. The aim of the VOR concept is to systematically allow external applications to make requests for a change in the functioning of the plant and accordingly, the NE 178 identifies the actors involved and defines their roles and responsibilities. The need for structured requests and the relevant feedback messages, information about security and authentication mechanisms, step-by-step processing of the incoming requests, and the logging of the requests such that everything gets tracked within the entire system and other aspects are outlined by NE 178 as described in (Iatrou et al. 2022)

(Specification IEC 16593-1) describes an Industry 4.0 (I4.0) component as a combination of multiple sub-components (I4.0). Here, each sub-component or the component as a whole has a specific behaviour. The I4.0 component interacts and reacts to the events generated within its environment. The behaviour of these I4.0 components may be modelled, using explicit procedures or finite-state machines, as shown in Figure 2.

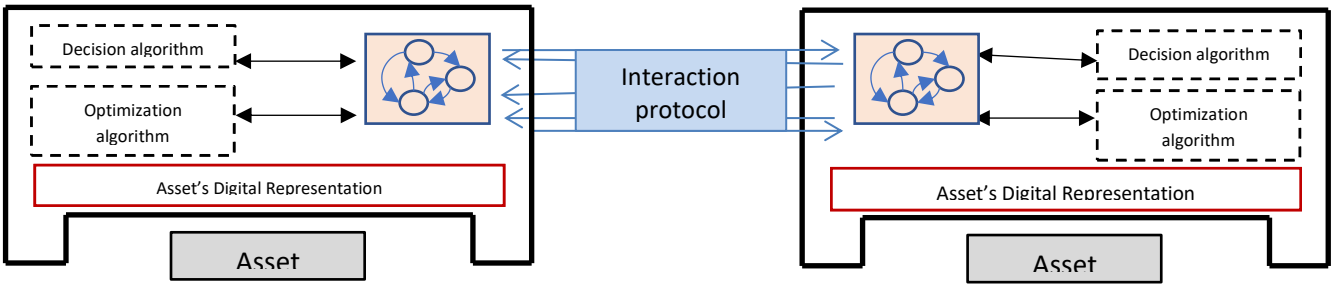

Figure 2 I4.0 environment consisting two Pro-Active AAS each consisting state machines modelling respective behaviours

(INSTITUT FÜR AUTOMATISIERUNGSTECHNIK 2019) address the aspect of pro-active asset administration shells (AAS) furthering the idea of type 3 AAS as presented in (Specification 01002-3-0). Pro-Active AAS are the I4.0 components that communicate with each other using the I4.0 language semantics (VDI-Richtlinien) and have the capability to take independent decisions. In this article, we aim to demonstrate the VOR Systems of Systems architecture using a set of Pro-Active AAS each communicating with each other over a defined interaction protocol. Accordingly, this paper introduces a new interaction protocol with relevant sequence diagrams. A prototype implementation of VOR architecture is presented with CPC domain represented by a single water tank case study (Larsen et al. 2016).

The rest of the paper is organized into seven sections. Section 2 presents a brief summary of NOA concept, AAS and the interaction protocols. Section 3 presents a detailed reported on the VOR concept as presented in (Iatrou et al. 2022). Section 4 presents a description of the single water tank use-case introduced in (Larsen et al. 2016). Section 5 demonstrates the implementation of VOR concept using Pro-Active AAS. Section 6 presents a brief overview of literature of past works. Section 7 concludes the paper.

2. Background

NAMR Open Architecture (NOA)

The NAMUR pyramid shown in Figure 1 is a monolithic and closed structure, with four different layer. As a system integrator, a plant owner has to own the responsibility of the interconnection between different levels, making his job more important and, the same applies to the device and the control software manufacturers. Such a hierarchical structure enables tight coupling between manufacturers of different products required for automation and has worked well over many decades. With the pyramid structure in place, for new technology to be integrated the existing devices should be upgraded, modified or replaced by the new ones, involving high investment from all the players involved.

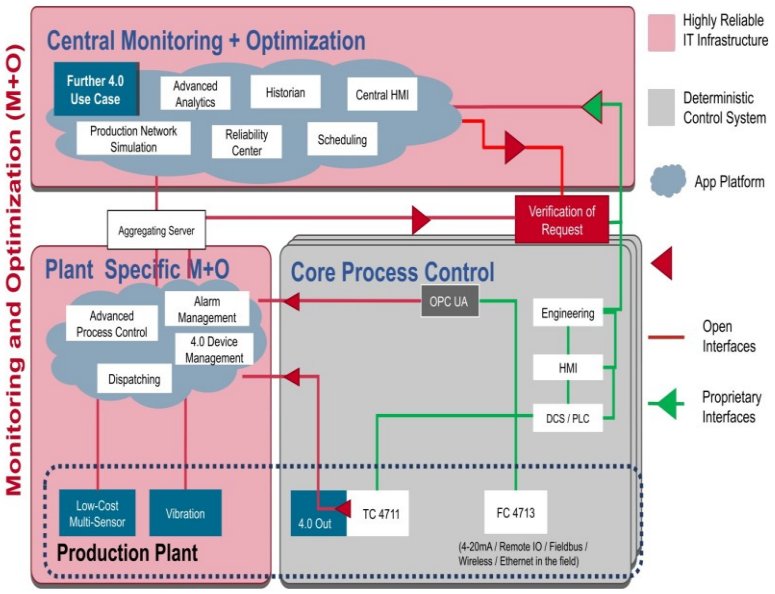

Figure 3 NOA Architecture

In such a background, (NE 175) just like its previous documents, is introduced by one of the NAMUR working groups. The NE 175 document introduces the NAMUR open architecture (NOA) concept. The primary goal of NOA is the introduction of modern IT technologies into the automation architectures without compromising the safety, security and reliability of the existing and future plants. The NE 175 clearly states that the ability to add new technologies is only complementary to the existing structure and the intent is not to replace the existing structure. In Figure 1, the tilted bar on one of the sides of the automation pyramid proposed by NE 175 acts like a plugin to the existing one. For the realization of such an idea, the NE 175 introduces a new architecture (as shown in Figure 3) with the concept of three communicating domains, namely the Core Process Control (CPC), Plant Specific Monitoring and Optimization (psM+O) and, Central optimization and monitoring (M+O). The CPC domain is the automation pyramid itself within the plant. The psM+O is the communication domain owned by the plant owner that is responsible for transmitting of information from the CPC domain to the external interfaces and acts like the demilitarized zone for the plant, where the plant owner has clear control over what kind of information enters and leaves the CPC domain. The M+O is the actual domain purported to be the additional bar, presented in Figure 1, that aims to introduce the technological advances from the IT industry into the automation pyramid.

Asset Administration Shell (AAS)

The concept of Asset Administration Shell, referred to as the standardized digital representation of an asset, is introduced by Platform Industry 4.0 (Plattform Industrie 4.0 2016). It aims to standardize the exchange of information between different partners, primarily within the automation industry. In order to achieve this, a standardized information model is introduced in (Specification 01001-3-0) with a core element called the Submodel that could contain elements of different types (Property, Range, MultiLanguageProperty, SubmodelElementCollection (SMC)). The term Submodel in the context of encapsulation of information about an asset could represent either a Nameplate or a Technical Data or a Documentation or asset access-related information. The information about an asset modelled using the AAS information model could be exchanged as files (Type 1 AAS), or a software component can consume the data and provide standardized interfaces for access (Type 2 AAS) (Specification 01002-3-0).

Interaction Protocols

(VDI-Richtlinien) characterizes the interactions of an active AAS with its environment as interaction protocols (IP) with I4.0 language format. IP is a structured sequence of interactions (or the exchange of messages) between I4.0 components, attaching defined protocols to the interactions makes the communication restricted to a set of message types. Where an I4.0 message as defined in the (VDI-Richtlinien) is organized into two parts frame (representing sender and receiver information, message type, conversation and messages IDs and a reference to the protocol definition) and the interaction elements (constituting the actual data).

3. NE 178 Verification of Request Architecture

In this section, we present different steps of the NE 178 as presented in (Larsen et al. 2016).

The architecture identifies three main actors within the scope of a plant in a process control industry, namely the requester issuer (this could be either from the psM+O or M+O domain, as pointed out in section 2), the VOR server and VAS server (VAS). NE 178 defines a sequence of steps, starting with, authentication until the request execution and provides guidelines on who (the specific actor) should perform them.

Authentication and Authorization Step: Every request issuer (RI) shall authenticate itself with the VOR server and then directly issue the requests. The VOR server first authorizes the RI against its credentials for every incoming request. Successful authentication would result in VOR sending feedback to the RI. Feedback is not issued for the failed authentication step. The credentials may be just the user ID and password or RI-specific certificates and, these may be managed using existing state-of-the-art solutions.

Verification Step: The VOR server first verifies the authenticated request, probably using a set of Boolean expressions. The plant owner is responsible for specifying these expressions.

Mapping Step: A request from the M+O user is more on a higher-level construct, indicating that the issuer does not know the internal dynamics or the operation of the plant. For example, a request from psM+O or M+O domain could be to switch off the valve that inlets the water into the tank. A set of endpoints represents the CPC domain, through which the specific aspects can be accessed and modified. Either the VOR server or the VAS maps an authorized request to the suitable CPC endpoints, in this paper we model the VAS server to perform this task.

Acceptance Step: This step checks whether the current state of the plant is valid and whether the plant is prepared enough to execute the request. One of the situations where a request is not accepted could be, for example, the absence of the operation personnel on the plant who is required during the execution process for monitoring and logging environment situations. NE 178 specifies that efforts must made to automate the acceptance and not bring in human interference.

Mapping Verification: The mapped request is then checked for relevance if it requires execution, for example, does the specific valve within the plant need to be switched off? In case a single request consists of changes to multiple CPC endpoints then the request needs to be executed only if all the changes are possible or else the request is completely rejected. This step finalizes the entire step and forwards the changes to the CPC endpoints.

Feedback Processing: Any request processing step can generate a feedback message. Every feedback submitted by an actor in a specific step is first verified and then dispatched. A feedback created by the VOR server will be dispatched to the RI and the feedback created by VAS will be dispatched to the VOR server only. There is no direct interaction between the VAS and the RI. At every step, the request and the feedback messages are timestamped and stored for traceability and verification in future.

4. CaseStudy

Figure 4 shows the P&ID diagram of a process automation plant with a single water tank, a use case as presented in (Larsen et al. 2016). A pump supplies water to the tank at a specific rate, and the valve (0041) attached between the tank and the pump controls the water flow into the tank. The tank has an outlet to drain the water, and another valve (0043) controls the water flow leaving the tank. The water inlet into the tank and the draining may not happen simultaneously. The two valves in Figure 4 are digitally accessible and have an OPC UA interface. The valves may be closed or opened by updating a boolean variable over the OPC UA interface. A level sensor is attached to the water tank, records the water level (in quantity) in the tank and sends the values as messages to the controller component at regular intervals, not shown in Figure 4. The controller component maintains the water level within maximum and minimum thresholds. Every time the controller receives a message from the level sensor, it checks the level against the threshold. If the value equals (maximum - 1), valve 0041 is closed and valve 0043 is opened. If the value equals (minimum + 1), valve 0041 is opened and valve 0043 is closed. The level sensor attached to the tank has an OPC UA interface, a single variable representing the water level in the tank.

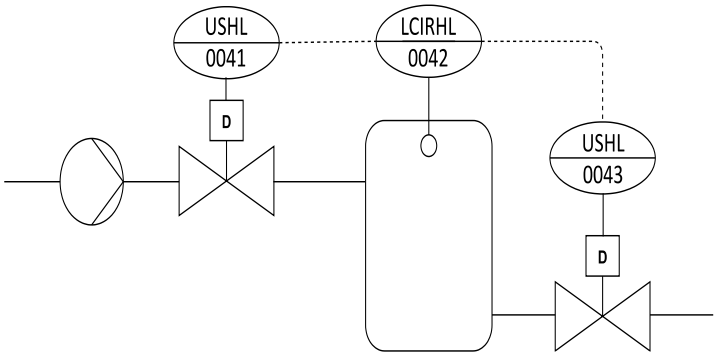

Figure 4 P&ID for the Single Water Tank (SWT) use-case

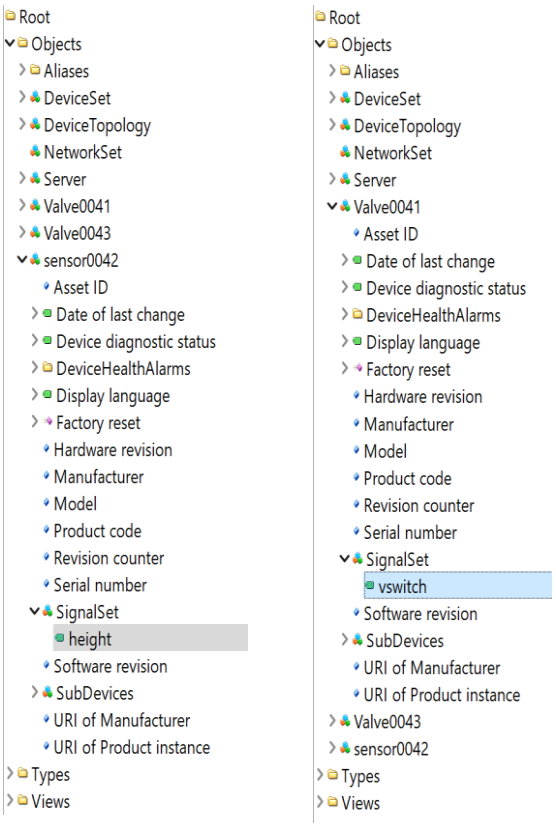

Figure 5 OPC Server representation for the single water tank use case

Figure 5 shows the screenshot from the UAExpert (OPCUA client) interface. The two valves and the level sensor for each, an object of type 'PADIMType' is instantiated. The sensor0042 has the variable 'height' of type LevelMeasurementVariableType under the SignalSet object, it represents the water level in the tank (with real-time updates). The valves 0041 and 0043 have the variable 'vswitch' of type boolean under the 'SignalSet' object and are used to open/close the respective valves. The PADIM-OPCUA companion specification can be accessed from (OPC Foundation 2024). Figure 5 indicates for the three devices, only one namespace is used, representing the entire plant. Based on the NE 175 guidelines, only read-only access to the OPCUA server is provided to the external M&O applications.

In this paper, based on guidelines from NE 178, we present a framework based on the concept of proactive AAS to access the functionality of the controller unit safely and securely. We suppose a scenario in which, due to unforeseen circumstances, external M&O applications may initiate a request to switch off the supply to the water tank. To this scenario we add an additional check, the water level in the tank shall always be above the average of maximum and minimum thresholds, when the source of the water is unexpectedly closed for-ever.

5. Realization of the NE 178 Concept for single water tank use case using Pro-Active AAS

In this section, we demonstrate the NE 178 concept using the single water tank use case, In this context, a new interaction protocol is introduced.

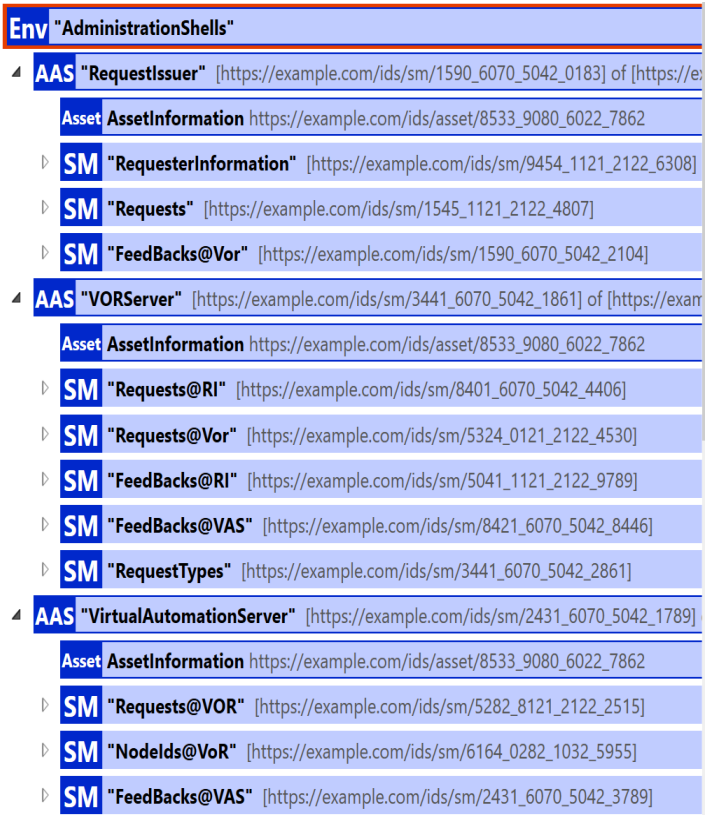

Figure 6 Screen shot from the Package Explorer representing three AAS and their respective submodels

In this paper, we utilize a set of Proactive AAS to demonstrate the NE 178 VOR concept. We model the NE 178 actors RI, VOR server, and VAS server as individual Pro-Active AAS. These representations do not represent a specific asset but encapsulate defined processes. Each AAS has a unique identifier and documents the information about the underlying process using the submodels. Figure 6 presents the screenshot of the package explorer (PE), it shows three AASs, and each AAS has a set of submodels. The RI AAS has three submodels, namely RequesterInformation, Requests and FeedBacks@Vor. The RequesterInformation submodel contains an identifier for the RI and its security-related credentials. The Requests submodel contains a list of SMCs, each representing a request issued by the RI. The FeedBacks@Vor submodel also contains a sequence of SMCs, each representing the feedback received by the RI from the VOR server. The VOR server AAS has four submodels, namely, a) 'Requests@RI', which contains a list of SMCs, each representing the requests received from RI. b) Requests@Vor, which contains a list SMCs, each representing the requests sent to the VAS server. c) FeedBacks@RI, which contains a list SMCs, each representing the feedback sent to RI. d) FeedBacks@Vas contains a list of SMCs, each representing the feedback received from the VAS server. The VAS server AAS has three submodels namely, a) Requests@Vor, which contains a list SMCs, each representing the requests received from the VOR server, b) FeedBacks@Vor, which contains a list of SMCs, each representing the feedback sent to the VOR server.

Similar to the interaction protocol presented in (INSTITUT FÜR AUTOMATISIERUNGSTECHNIK 2019), in this section, we present a protocol representing a series of interactions between the three NE 178 actors, as shown in the sequence diagram from Figure 7. The behaviour of each AAS within the context of the interaction protocol is modelled using the finite state machines (FSM). A specific interaction between any two actors involves the transfer of a message from one actor to the other and is identified by a unique message type. The labels identifying the directed arrows in Figure 7 are the message types. The sequence of the message types between all the actors makes up the interaction protocol. Each message exchanged between the actors over a specific interaction is an I4.0 message as described in (VDI-Richtlinien). The actual content of each I4.0 message is either a submodel or any of the AAS submodel elements.

Authentication and Authorization: As part of the SWT demonstrator, we have maintained a single authentication server. Each Pro-Active AAS representing the NE 178 actor is provided with a certificate. The Pro-Active AAS uses this certificate to get an access token from the authentication server, it then uses it for interaction with the other AAS. Every un-authenticated interaction would be rejected.

We have used PythonAASxServer framework for the implmentation of the Pro-Active AAS (Harish Kumar Pakala 2024). A detailed description of the framework is provided in (Weiss et al. 2023). The Framework provides an order management interface, this is used for creating a request.

The interaction protocol in the Figure 7 begins when the RI, receives an order to create a new request. The RI then creates a request message, that contains an SMC (request1) constituting the elements, identifier, description, impact, the priority of the request and the timestamp of the request creation, as shown in Figure 8 and transmits the message to the VOR server.

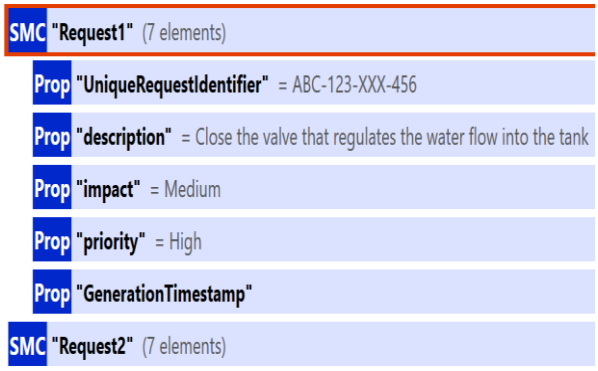

Figure 8 AAS PE screen shot depicting a Request from RI

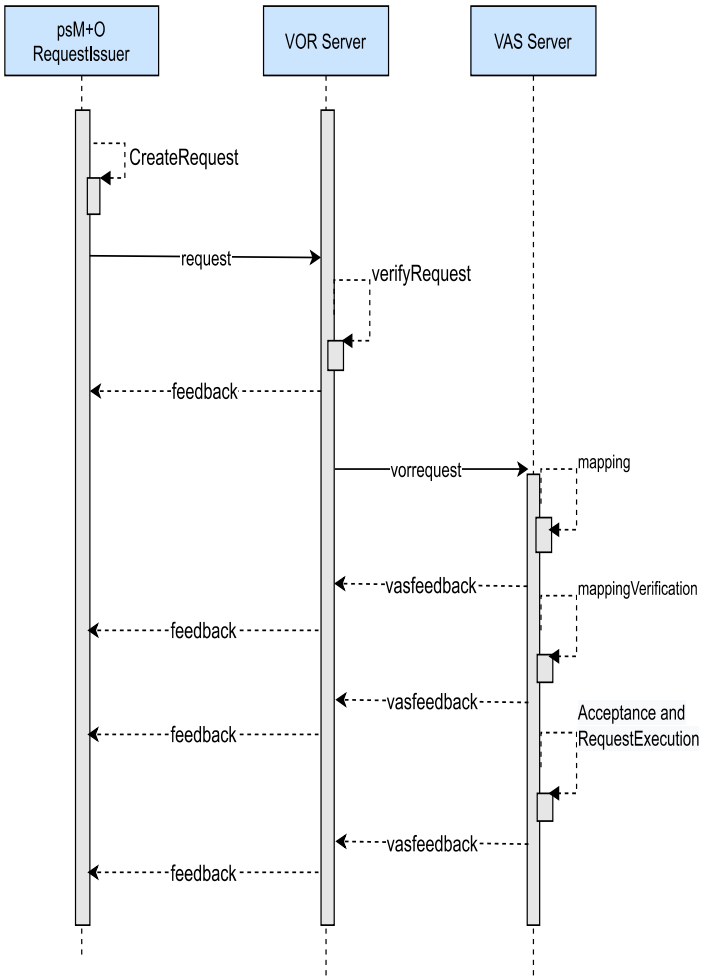

Figure 7 Sequence Diagram representing the VOR concept

In the case of the SWT demonstrator use case, suppose that the water supply into the tank needs to be closed due to some external reasons and to stop the flow of water into the tank. The RI specifies this information in the description property element as shown in the Figure 8. The priority of this request is set to high and, the impact of this request execution is set to medium. The NE 178 mandates that every request generated by the RI needs to be logged and accordingly, the Request1 SMC is added to the 'Requests' submodel.

The RI may issue only two request types “Close the valve that regulates the water flow into the tank” and “Open the valve that regulates the water flow into the tank”. The VOR server first verifies the originator of the request message, it is the eligibility test for the RI to issue a specific request. For every verified request, the VOR server notifies the RI with a feedback message. The VOR server interprets the description of the message and turns it into an executable request, as shown in Figure 9. We suppose that the VOR server has complete knowledge of the structure of the plant. Figure 9 shows that valve0041 needs to be closed. The valve0041 is mentioned under the parameters SMC and, the change to the switch associated with the valve is mentioned under the modification SMC. The VOR server creates a new message of type ‘vorrequest’ and attaches to it the Request1 SMC shown in Figure 9.

Figure 9 AAS PE screen shot depicting a Request from RI

The VAS server examines the incoming vorrequest and attempts to map the request to an executable action, for example, to modify a specific variable from the OPCUA server. In case the mapping step fails, negative feedback of type 'vasfeedback' is sent to the VOR server. In the SWT use case, the request 'close switch related to the Valve0041' is mapped to nodeId of the variable switch, with a change to the variable as 'closed'. The list of the variables and related nodeId mappings are part of the VAS Server AAS shown in Figure 6. After the successful mapping step, as there is no requirement for personal in SWT use case the request is accepted and an acceptance notification is sent to the VOR sever.

In the SWT use case, it is assumed that the valve0041 may be closed only when it results in a scenario where the water level is above the average of maximum and minimum thresholds. The VAS server obtains the water level in the tank by reading the relevant OPCUA variable and checks if the level is already below the average. If yes, the request is rejected indicating the mapping verification is failed, and negative feedback is sent to the VOR server. In case the level is above the threshold, the request is accepted as mapping verified and pushed for execution.

In the execution step, both values 0041 and 0043 are closed one after the other immediately without any delay and positive feedback is sent back to the VOR server (assuming that modifications of the OPC UA variables and the changes to the underlying assets are precise without any errors). Upon receipt of the 'vasfeedback' message from the VAS server, the VOR server examines the message and send an appropriate message of type 'feedback' to the RI. Figure 10 shows the structure of a vasfeedback from the VAS server, screen shot form the package explorer. The intent of the message is specified as the 'implemented', indicating that the the change request is executed succesfully.

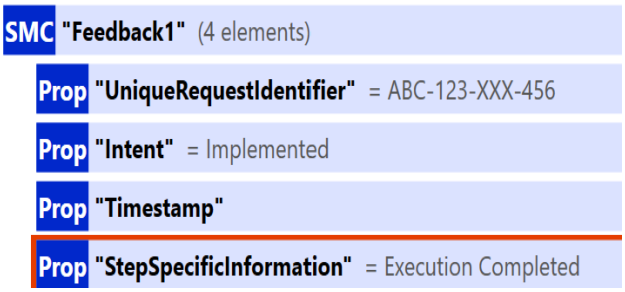

Figure 10 AAS PE screen shot depicting a Feedback from VAS Server

Benefits of using Pro-Active AAS

- Figure 11 shows the submodel RequestTypes representing the type of requests possible by the VOR concept (part of VOR server AAS). The plant owner may dynamically edit this submodel and delete or add new request types.

- The plant owner may also specify a mapping between Request Issuers and Request Types,

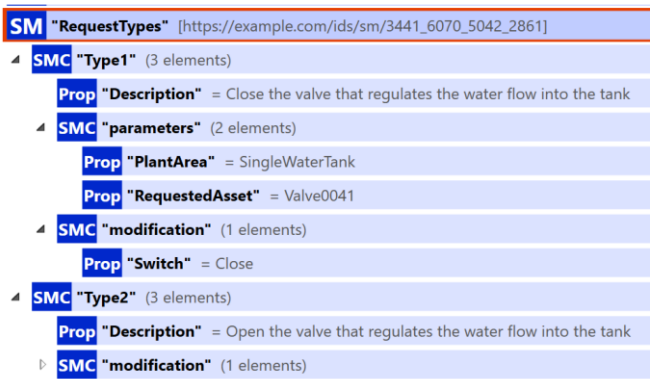

Figure 11 AAS PE screen shot depicting the Request Types from VOR Server AAS

allowing him to decide to deny a specific RI for a specific Request Type.

- All the messages exchanged within one execution of an interaction protocol are logged as submodel elements with respective AAS, as shown in Figure 6.

Single source of the truth for the Plant Owner.

- Apart from the central authentication server, role-based access to the submodels may be granted.

Access to information may be restricted tightly among the three NE 178 actors and also the external applications.

- Each asset shown in Figure 4 may be represented by AAS. Documenting technical properties of the asset using a submodel, may allow the plant owner to change, for example, the threshold values for the water level in the tank at any instant in time.

As part of the SWT demonstrator, we have modelled the assets shown in Figure 4 as AAS, the screenshots are not included in this paper.

FSM implementations for each of the NE 178 actors are implemented in the PythonAASxServer (Harish Kumar Pakala 2024) framework as pointed out in section 5. These FSMs fetch the information from the AAS repositories dynamically, every time they perform a specific operation. FSM implementations are cyclic, this allows the creation of new requests after completing another request without a change in the configurations.

6. Related Work

This section provides a short summary of the previous work within the domain of digital twins, industry 4.0 solutions for control within the automation pyramid, our intention is just to provide a flavor of approaches adapted by different works. (Azarmipour et al. 2020) aims to attach a digital twin (DT) at all levels of the automation pyramid. In the article, the authors have shown an example of DT at the process control level. The basic idea is that the DT requests information from the process level, performs the required simulations or optimizations and returns the result to the MES whenever it is requested.

To ensure the security of the pyramid structure, the authors assert the architecture enables a unidirectional flow of information from the control level to the DT, indicating that the DT will never have the opportunity to send any control signals to the process level. (Martinez et al. 2021) proposes a DT structure for the entire automation pyramid and introduces a virtual model to all the level and within in this context aims to leverage the benefit the AI. (Pessl and Rabel 2022) presents a use-case for dissolution of the automation pyramid within the industrial environment, where dissolution here means that instead of a top-bottom (and vice-versa) flow of information it would be interactions between all the level. To the best of the knowledge of the authors there is no paper which deals with a multi-phase write access to an automation component such a field device from an external application.

7. Conclusion

Industrial automation system migrates from a rigorous hierarchical architecture to an open system. In the process control domain this is accompanied by the NOA concept. It starts with a strict read only approach and is nowadays extended by a secure way of modifying parameters in automation devices. This concept is called “Verification of Request” (VOR). While defining the requirements in the according “NAMUR Empfehlung” NE 178 this paper propose a specification which fulfill the requirements. This is implemented as prototype using the Asset Administration Shell (AAS) which is a standardized specification of digital twins. The VOR is not restricted to the process control domain, it is suitable for a large variety of applications where a secure write access to the automation system is needed

8. Acknowledgement

This work of the project "Administration Shell Networked - Interoperability between I4.0 Components" was supported by the Federal Ministry for Economic Affairs and Energy in the Industry 4.0 initiative under the following numbers: FKZ: 13I40V001A.

Publication bibliography

Azarmipour, Mahyar; Elfaham, Haitham; Gries, Caspar; Kleinert, Tobias; Epple, Ulrich (2020): A Service-based Architecture for the Interaction of Control and MES Systems in Industry 4.0 Environment. In : 2020 IEEE 18th International Conference on Industrial Informatics (INDIN): IEEE.

Harish Kumar Pakala (2024): PythonAASxServer: Github. Available online at https://github.com/harishpakala/PythonAASxServer, checked on 4/26/2024.

Iatrou, Chris Paul; Hoppe, Henry; Erni, Klaus (2022): NOA Verification of Request: Automatisierte Anlagenoptimierung mit Feedback aus Edge und Cloud. In *atp* 64 (1-2), pp. 78–84.

INSTITUT FÜR AUTOMATISIERUNGSTECHNIK (2019): Specification Testbed "AAS networked“. Proactive AAS - interaction according to the VDI/VDE 2193. INSTITUT FÜR AUTOMATISIERUNGSTECHNIK.

Klettner, Christian; Tauchnitz, Thomas; Epple, Ulrich; Nothdurft, Lars; Diedrich, Christian; Schröder, Tizian et al. (2017): Namur Open Architecture. In *atp* 59 (01-02), p. 17. DOI: 10.17560/atp.v59i01-02.620.

VDI-Richtlinien, 04 2020: Language for I4.0 Components - Structure of messages.

Larsen, Peter Gorm; Fitzgerald, John; Woodcock, Jim; Fritzson, Peter; Brauer, Jörg; Kleijn, Christian et al. (2016): Integrated tool chain for model-based design of Cyber-Physical Systems: The INTO-CPS project. In : 2016 2nd International Workshop on Modelling, Analysis, and Control of Complex CPS (CPS Data). IEEE, pp. 1–6.

Martinez, Edwin Mauricio; Ponce, Pedro; Macias, Israel; Molina, Arturo (2021): Automation pyramid as constructor for a complete digital twin, case study: A didactic manufacturing system. In *Sensors* 21 (14), p. 4656.

NE 175, 7/9/2020: NAMUR Open Architecture NOA Concept. Available online at https://www.namur.net, checked on 6/6/2023.

OPC Foundation (2024): OPC 30081: Process Automation Devices - PADIM (1.01). Available online at https://reference.opcfoundation.org/nodesets/129, updated on 2/1/2024, checked on 4/18/2024.

Pessl, Ernst; Rabel, Birgit (2022): Digitization in Production: a Use Case on a Cloud-based Manufacturing Execution System. In : 2022 8th International Conference on Computer Technology Applications. New York, NY, USA. New York, NY, USA: ACM.

Plattform Industrie 4.0 (2016): Fortschreibung der Anwendungsszenarien der Plattform Industrie 4.0. Ergebnispapier. Bundesministerium fÜr Wirtschaft und Energie (BMWi).

Specification IEC 16593-1: RM-SA - Reference Model for Industrie 4.0 Service architectures.

Specification 01001-3-0, April 2023: Specification of the Asset Administration Shell. Available online at https://industrialdigitaltwin.org/, checked on June 2023.

Specification 01002-3-0, April 2023: Specification of the Asset Administration Shell, checked on June 2023.

Weiss, Marco; Wicke, Kai; Wende, Gerko; Pakala, Harish; Gill, Milapji (2023): MaSiMO-Development and Research of Industry 4.0 Components with a Focus on Experimental Applications of Proactive Asset Administration Shells in Data-Driven Maintenance Environments.