Industrial Controls and Asset Administration Shells: An Approach to the Synchronization of Plant Segments

Stephan Schäfer[1], Dirk Schöttke[1], Thomas Kämpfe[1], Oliver Lachmann[1], Aaron Zielstorff[1] and Bernd Tauber[2]

[1] *Hochschule für Technik und Wirtschaft (HTW) Berlin, Berlin, Germany*

[2] *EAW Relaistechnik GmbH, Berlin, Germany*

Keywords: Asset Administration Shell, Digital Twin, PackML.

Abstract: The complexity of modular production plants is constantly increasing due to flexible functionalities. The need to be able to flexibly adjust processes to product requirements is thus becoming more relevant. Therefore, limiting production plants to their processes is no longer up-to-date and a division of processes into single, atomic capabilities, which are represented by a Asset Administration Shell (AAS), has proven to be useful. This article deals with the synchronization of individual capabilities at the field level via the use of the PackML State Machine. An approach is presented how individual capabilities can be combined into a composite capability using a higher-level state machine. This approach is similar to the group or control component presented in BaSyx. To be able to represent the data in the AAS, the PackML does not offer a direct interface. This is created via a template in the control layer to be able to represent data in the AAS. This allows the AAS to read data in one structure and independently manipulate parameters in another structure in a non-real-time manner.

1 INTRODUCTION

Due to their flexible functionality and variable system design, convertible production environments exhibit a high degree of complexity. The complexity arises, among other things, from the heterogeneity of the components used, their scalability, their interaction within the systems, and the system-wide communication. Software solutions in the environment of automation systems, among other things, have a major influence on the flexibility and complexity of the plant. With them, plants can be flexibly configured to the respective application case.

Here, the interoperability and scalability of the systems represent a significant challenge. Among other things, the reference architecture model (RAMI) for Industrie 4.0 presents the concept of the Asset Administration Shell (AAS) as an essential basis for interoperability (Bader et al., 2022). The AAS is the digital representative (digital twin) of an object (asset) in the I4.0 environment and enables communication to further assets (Ye et al., 2021). For example, machines, products or controllers are considered as assets. AAS consist of several submodels in which information and functionalities of an object, and other things, are described. The information provided by the AAS includes documents, properties, parameters, and other functions (Kuhn et al., 2020). The AAS thus becomes a provider of bookable services, which can be synchronized via a coordinator, for example.

For the use of the available services with their characteristics, their secure booking in and out of the plant network and a comprehensible process control are essential. Necessary system parts and components may only be integrated from a defined and reproducible plant state. Here, for example, the OMAC PackML with its structure and components can be used for process control. The "BaSyx Control Component" on the AAS side is also suitable for synchronizing processes.

The aspects mentioned (convertibility, interoperability, synchronization of processes) give rise to the question of how these can be harmonized with existing systems and their use, for example. A large number of different industrial controllers are established in the industrial environment. They can communicate via a variety of field buses. This already results in a need for adaptation for further processing of the information. This adaptation can be compensated to a large extent by using AAS and software adapters,

Schäfer, S., Schöttke, D., Kämpfe, T., Lachmann, O., Zielstorff, A. and Tauber, B.
Industrial Controls and Asset Administration Shells: An Approach to the Synchronization of Plant Segments.
DOI: 10.5220/0011527200003329
In *Proceedings of the 3rd International Conference on Innovative Intelligent Industrial Production and Logistics (IN4PL 2022)*, pages 75-84
ISBN: 978-989-758-612-5; ISSN: 2184-9285

since information is provided here via a standardized interface.
In new plants, this can already be taken into account during the conception/design of the plant, but in existing plants this is essentially only possible during migration or with adaptation of the project planning. Currently, industrial control systems do not offer the possibility of initial integration of AAS. This means that additional engineering is always required for their preparation. Here, a simplification of the processes should take place, which can also allow a step-by-step digital transformation.

Furthermore, industrial controller not only automates individual processes, but also realizes networks of plant components. These can be coordinated/synchronized inside or outside the control system. Both approaches can usefully complement each other and are well established. If synchronization takes place at the level of the industrial controller, the end-to-end consistency of the data must also be ensured in the associated AAS, i.e., its digital representative. This applies not only to the relevant data for synchronizing the processes, but also to supplementary information from the industrial control system.

The focus of the paper is the presentation of a practical use case from the project ”OpenBasys 4.0”, which shows possibilities to reduce the initial effort in engineering. The reduction results with the use of defined templates for various industrial controllers (here Codesys - platform) for synchronization and the integration of the data model into the assigned AAS structure.

2 INDUSTRIAL CONTROLS AND STATE OF THE PRACTICE

Controls/automation devices, with their control programs according to IEC 61131-3, are assigned to the lowest level of the RAMI. They represent a close relationship to the operating equipment of the production plants/systems (Cavalieri and Salafia, 2020). This results in significant advantages in the description, commissioning and the process of reconfiguration of the plant components. Despite the connection, there are also deficits in their use (Wallner et al., 2021). For one thing, there is a lack of consistent self-description, information on maintenance, and representation of relationships to neighboring systems/items. On the other hand, breaks in the consistent preparation, storage and use of information exist in the engineering phase. In addition, there is currently no ”established procedure method” for integrating inventory solutions with industrial controls in I4.0 system environments, but rather a variety of proprietary solution methods. Some deficiencies can be compensated for in the context of using Basys 4.0 middleware (Adolph et al., 2020).

Reducing the deficits mentioned requires a methodical approach in order to be able to generate as much added value as possible from the available data. For this purpose, it is first necessary to consider the state of the art for project planning of control systems. In various contributions with their use cases, the initial configuration of plants and required structures are assumed. In (Bouter et al., 2021), for example, extensive descriptions were made and the modeling of a reference plant (pick & place station) with its components was shown. The modeling of capabilities as submodels was also discussed here. The migration/transfer of existing plants, in their overall constellation or in sub-areas, to I4.0 environments, for example, represents a different situation and a challenge here. Further approaches of reference implementations are discussed in (Belyaev et al., 2021) and (Di Orio et al., 2019).

2.1 Process Synchronization

The construction of modular and changeable plants requires capability-based manufacturing systems. A distinction can be made between ”atomic” and ”composed” capabilities (Bayha et al., 2020). Several ”atomic skills” can be combined into a ”composed skill” and thus generate a specific manufacturing process. To bring these capabilities in line, it is necessary that these processes are synchronized. The synchronization of individual sub-processes results in a complex process that can be managed by a higher-level unit.

2.2 Process Control - Synchronization with AAS

For the synchronization of processes BaSys 4.2 proposes a ”Control Component” (Grothoff et al., 2021). This allows individual capabilities to be mapped in a state machine. Each process can be controlled uniformly and triggered individually by a user or an orchestration unit. The orchestration can be done by an AAS or a PLC, which has been determined to be the control component. Itself can be represented by a state machine.The control component has a significant influence on the use of components here. This concerns not only the operation, but also the commissioning and the possible component exchange. It enables subsequent tasks in a changeable environment (Grüner et al., 2021):

- Defined access and asset control at field level, component level, or composite component level
- During initial commissioning or replacement of components, these can be brought into defined states via predefined interfaces and supplemented with relevant parameters
- Encapsulation of capabilities or their supplementation by e.g. virtual components
- Assurance of the process access (prioritization of abilities)
- Realization of dependencies to/between components by using the asset context

The control component itself can be represented by a state machine. BaSys relies on its own state machine, which is similar to the PackML state machine in essential areas, and represents the following states (excerpt) (Grothoff et al., 2021):

Occupation: The occupation state machine defines the occupation state of the component. An occupation realizes an exclusive lock on the control component. It defines who uses a control component. Certain orders may be issued only if the principal occupies the control component, i.e. if he has the lock on the control component.

Execution Mode: The execution mode state machine defines the execution mode of the control component. Execution mode defines how a control component responds to commands. Execution modes include automatic mode, which is the usual mode of operation, or semi-automatic mode for setup operation. A control component is in only one execution mode at any given time.

Execution State: Execution state communicates the execution state of a control component using a PackML state machine. Execution state indicates the state of a control component in a particular execution mode.

Operation Mode: The operation modes of a component are used to distinguish different capabilities of a component. By selecting an operation mode, the user of a component specifies what the component should do.

Work State: The work state of a component defines the current, operation mode-specific state of a control component.

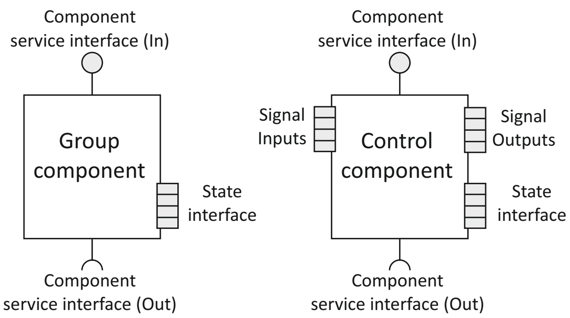

Figure 1: Interface of a control and group component (Grothoff et al., 2021).

Contrary to the BaSys Control Component (CC), the PackML state machine is already a general standard in the packaging industry. There is a generally valid implementation guide for it, which is designed for platforms that follow the IEC 61131-3 standard. The implementation is therefore standardized and makes the PackML state machine robust and usable across manufacturers.

2.3 Grouping of AAS-components

The function of a group component (Fig. 1) is not to interact directly with the process, but to coordinate and orchestrate the control components that then interact with the process. Therefore, a group component combines the various basic capabilities into an composed capability (overall function). It is possible to name a large number of examples here, since diverse areas of manufacturing are usually based on this principle. Among other things, the pick and place task is based on the principle of grouping capabilities. Here, for example, parts are provided, picked up, transported and transferred.

According to the BaSys4.0 specification, group components differ from control components in that they do not have an IO interface to the controlled process. They use the available network interfaces to act as service providers or service users. In addition, the current status of the group component is provided.

2.4 Process Control with PackML

The PackML state machine is an ISA standard that was adopted in 2008. It has already been adapted on a large number of platforms (Fathizadeh et al., 2013). The PackML according to ISA guideline TR88.00.02

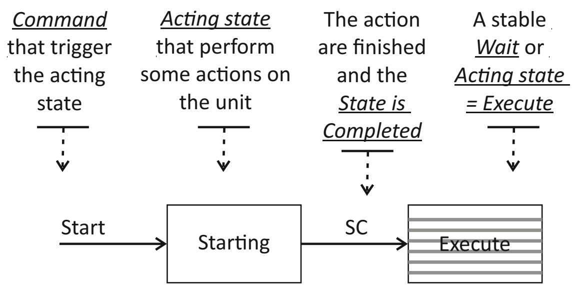

Figure 2: Syntax of the PackML SM (OMAC, 2009).

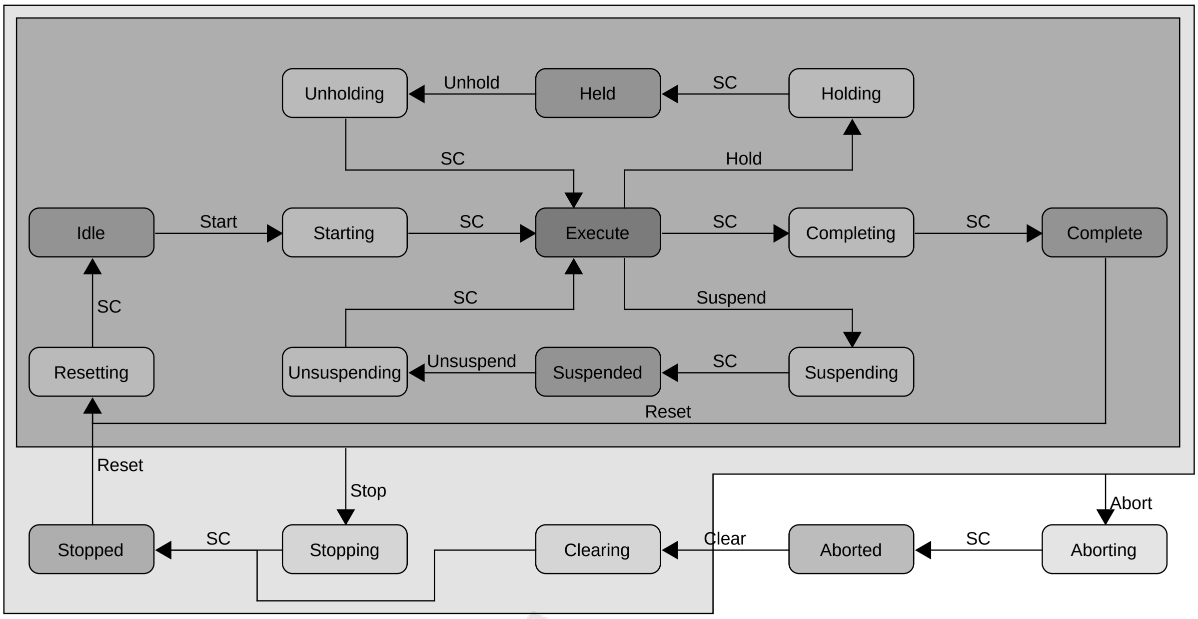

Figure 3: PackML state machine.

takes into account the operating modes ”Production, Maintenance and Manual”. These can be supplemented by the user (ISA, 2015; Mušič, 2015). Thus, each operation mode can have a maximum of 17 states. These are divided into three categories. The Wait-States, with which it is to be signaled that a certain condition of the plant was reached. The Acting-States, in which different activities can be executed and a Dual-State (Execute) (Arens et al., 2006). The general syntax of the PackML state machine is shown with Figure 2.

All 17 states (Fig. 3) are divided into sub-areas. Thus it can be achieved that inner states can be left at any time by a single condition. In addition to these defined states, PackTags were introduced to pass machine data to IT systems. This means that PackML state machines can also be controlled by higher-level or neighboring systems.

Companies benefit from this established standard, which is now widely used in the packaging industry (Dorofeev and Zoitl, 2018). This results from the fact that PackML state machine makes applications more efficient, flexible and reusable. Commissioning also becomes less costly and interoperability between the different control systems can be ensured. Troubleshooting becomes easier due to the consistent operations and thus the uptime of the plants is extended.

3 USE CASE AND ASSET ADMINISTRATION SHELL

An essential requirement for the successful transfer of the existing plant to an I4.0-based solution is the reproducibility of the available plant description and the implemented programs.

In our application, products, in this case high-quality thermal switches, are tested in an industrial environment during production or after the manufacturing process. In this environment, each product passes through predefined test scenarios. The products are placed on a special pallet carrier and made available to the respective processes. The required test frame, the sequence and the dwell time of the products on the test stations vary according to the product batch. As an example, the technological scheme (excerpt) of the plant environment with its processes is shown in Figure 4.

As part of the preparation of the use case, there was a discussion of a possible migration of plant components using AAS. The analysis of the environment of the test station of thermoswitches and their components showed that a variety of processes, starting with the preparation of the test specimens, their provision, the transfer and other processes must be included.

The automation of the stock was carried out with heterogeneous instrumentation (see Figure 5), starting with the use of programmable logic controllers (PLC) from different manufacturers and ending with the industrial robot for handling. Here, an overarch-

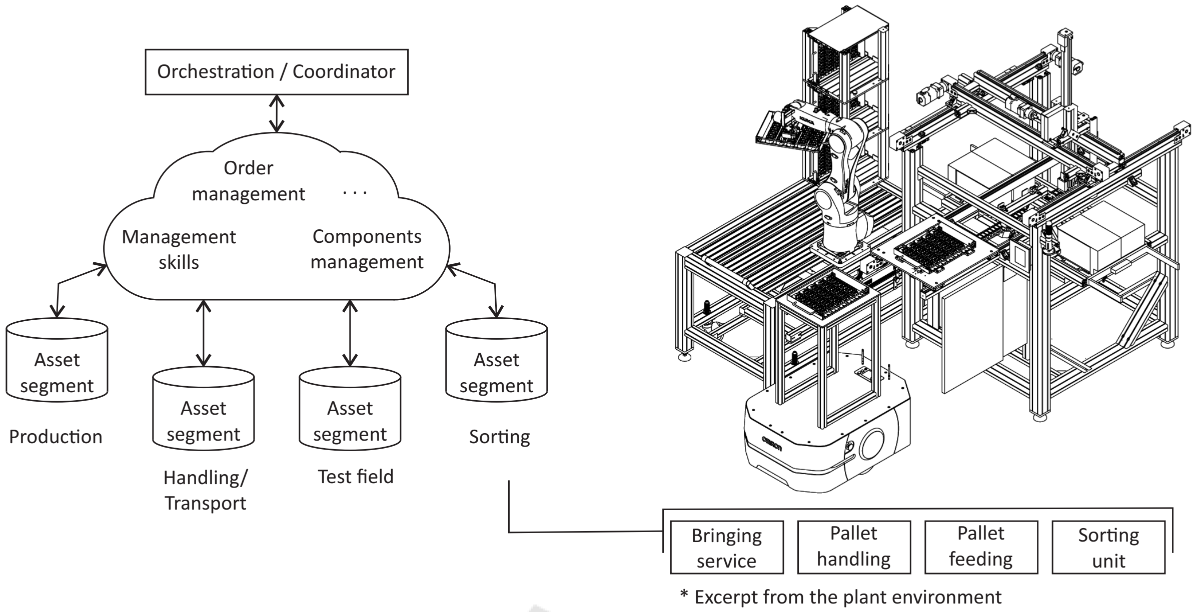

Figure 4: Subsection - pallets feeder.

ing synchronization and coordination of the processes is required. This can be achieved, for example, by establishing control components at the AAS level or by using a PackML state machine implemented at the process level and possible extensions.

Since essential components of the plant were implemented in an IEC 61131-3-compliant Codesys environment, the question arose of transferring components to a future I4.0 environment. Among other things, a large number of function libraries and various communication protocols can be used in this platform(Rayment, 2004). However, there are no solutions that allow simple integration into an I4.0 environment and its synchronization via AAS.

The challenge is thus the effort-reduced realization of a digital representative as I4.0 components and their integration (Koulamas and Kalogeras, 2018). These can be mapped in the form of a AAS. Here, a distinction is made between three types of AAS based on a uniform information metamodel (Bedenbender et al., 2020). Type 1 contains a passive AAS

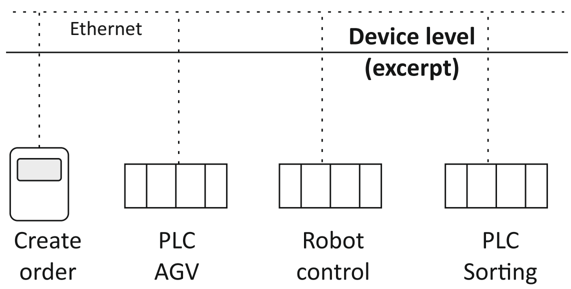

Figure 5: Instrumentation - initial situation.

with asset description. Type 2 represents a reactive form, which includes a communication channel in addition to the asset description. Only the proactive AAS of type 3 enables an independent communication between the AAS. The description of a AAS can be done, among other things, in the AASX Package Explorer (Repository, 2022). It can be used to realize a structured description using submodels and other structural elements. Submodel templates are provided for the modeling of frequently used/recurring asset aspects.

The discussion of the use case resulted in the necessity of both migrating the inventory of industrial controllers to the future environment and implementing their representation in the form of AAS with reduced effort. With the implementation of PackML on the controllers and various extensions, which enable a similar range of functions as the control component, a number of advantages result from the step-by-step migration. These include, for example, the booking of resources, the synchronization of processes and the possibility of equalizing the migration over time.

The SDK BaSyx 4.2 (Platform, 2021) was used in the ”OpenBasys 4.0” project (BMBF, 2019) to implement the representative. It offers the possibility to realize Asset Administration Shells in different programming environments. The project engineering of Asset Administration Shells with SDKs can be done manually according to the general description of the structure with its submodels using the C# programming environment.

However, this means that an IT expert must take

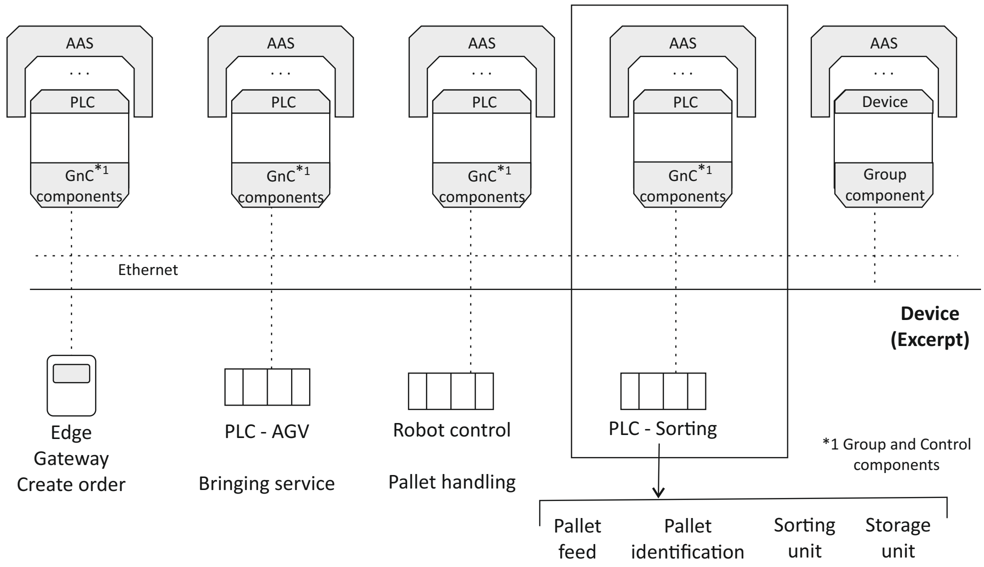

Figure 6: Instrumentation - with AAS assignment.

over the engineering area in the company. This aspect may limit the spread of AAS in the industrial environment and in the company.

For example, a control component for synchronizing the components must currently still be implemented manually on the basis of the IDTA templates. This template is currently only available in the SDK version (Java / C++), which limits its distribution for synchronization.

In many cases, there are also not corresponding employees with the necessary knowledge available in the company. However, automation engineers are already employed for the existing machinery and equipment, who are responsible for supporting the system. Their knowledge and skills can be drawn upon with regard to the design of solutions. This also applies to the coordination/synchronization of processes. Since the manual creation of AAS is widely established, this is not considered in detail in the article.

Instead, the question arises as to how the user of industrial controller can be enabled to use AAS in an industrial environment. One option is the largely generic preparation of AAS without knowledge of the SDK environment. Requirements for this should only be the description of the asset as AAS type 1 and the adaptation of the necessary interfaces in the control software.

Since the software development process goes through several phases, it makes sense to establish the generation of the AAS (Schäfer et al., 2021) online with the controller as well as offline without the controller but with a defined data model and communication protocol. Thus, a generated AAS (offline variant) can already be evaluated via its interfaces with a test application even without a controller. The interfaces used/defined can subsequently be made available to the controller as a template via an exchange format.

4 USE CASE - IMPLEMENTATION

4.1 System Environment/Components

The possibility of a largely generic preparation of Asset Administration Shells was taken up in the "OpenBasys 4.0" project. In addition to simplifying software development, the generic approach enables. This leads to the reduction of time and possible malfunctions. In the context of the project an environment for the generic preparation of Asset Administration Shells was realized. In the future, this environment will be supplemented by accompanying test scenarios (structure-, interface-, unit-tests).

In this environment, without SDK knowledge, Asset Administration Shells of type 2 (reactive AAS) can be realized. The company's employees only have to make minor corrections and additions to their process controls. This mainly concerns the data model used and the communication channel, since the AAS generator in the current version accesses defined global variable lists.

On request, for example by the process control, the generation of a Asset Administration Shell of type 2 is largely automated, which can be stored in a container after completion of the process. It is also possi-

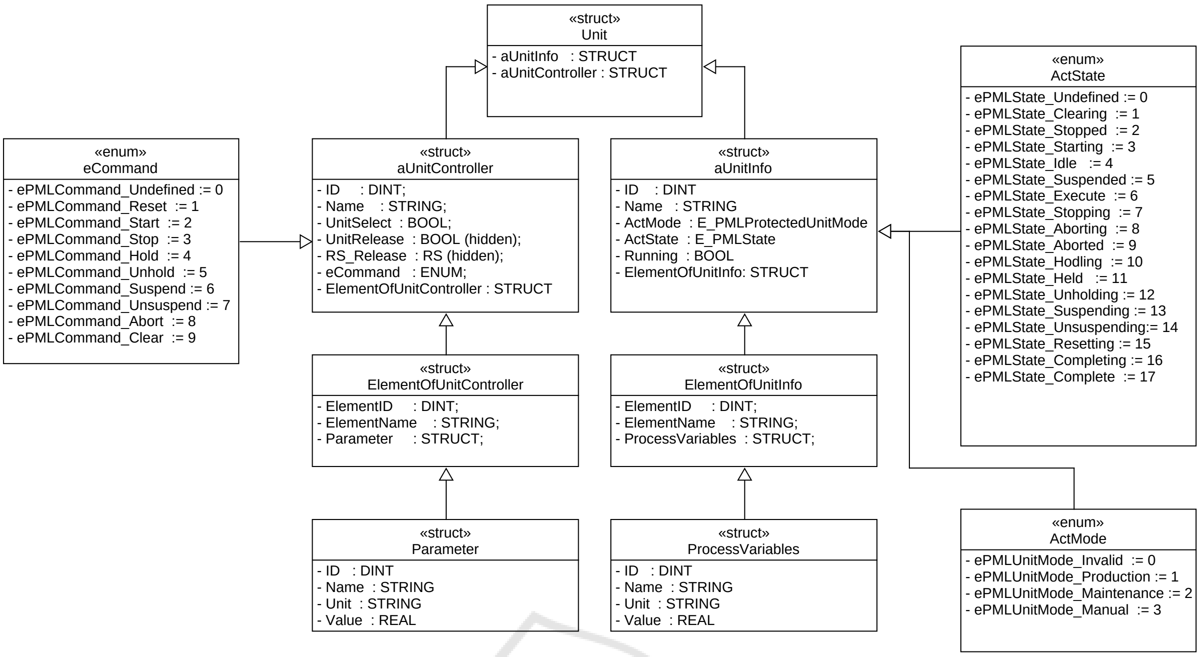

Figure 7: Structure Unit (aUnitInfo and aUnitController) (Schäfer et al., 2022).

ble to trigger this process via the web interface of the AAS generator. The user can use this approach to design reactive AAS relatively easily and transfer them to an I4.0 compliant environment. A machine interpretation of the contents/capabilities is not yet given due to the different usage of the terminology.

In the project, the online variant was implemented, which prepares its own structure in the global variable list (GVL) for the application. Among other things, the PackML state machine (SM) data (including the details of used plant components) and the process data from the process controller (PLC) are transferred. Using the approach resulted in the infrastructure of the migrated plant shown in Figure 6. Each controller

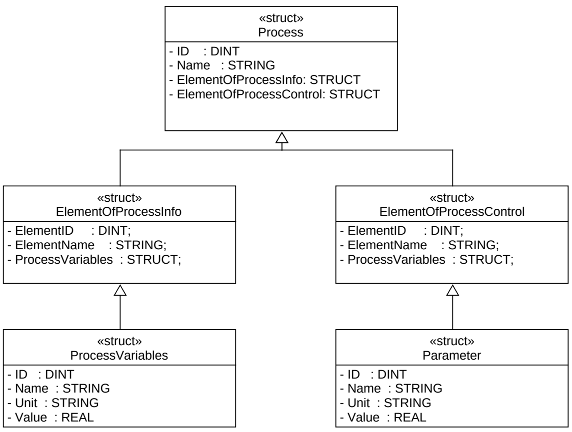

Figure 8: Structure ”Process” with its elements (Schäfer et al., 2022).

was assigned an AAS according to its functions, representing the interface to the controller with the implemented PackML and necessary additions.

4.2 Interface to the AAS

According to the metamodel of the Asset Administration Shell, a structuring of the data in submodels and submodel elements according to their domain is required. Since these were not described in more detail in (Bader et al., 2022), (Lüder et al., 2020) demands that a solution be found/established that allows engineering data to be integrated efficiently and effectively into the Asset Administration Shell. This applies not only to the design of new plants, but also to existing automation solutions. In addition to the technical description, information from the process events and their respective status must also be assigned to the engineering data. For their representation, tags are embedded in the realized system environment, which provide information (unit info) from the processes or influence processes or plant components via the implemented command structures (unit controller). Figure 7 shows an excerpt of the class structure of the tags (UnitInfo and UnitController). These were implemented using the global variable lists (GVL) on the respective target systems. For supplementary and specific information from the plant area/processes, an extended global variable list (Process) is available to the user for use. Both types of global variable lists are considered in the automatic generation of the Asset

Administration Shell in the form of submodels with their properties. In addition, the information is prepared in an associated GUI (FrontEnd) for further use with any end devices. The structure aUnitController is used to control and parameterize the master SM and subordinate machine units (Units).

Units can be, for example, a linear axis or a jointed-arm robot, or, as in this application, a defined and delimitable process of the entire plant. The process is selected via the Unit-Select variable, for example, and the PackML state machine that accompanies or controls this process is activated. The state machine is controlled by changes to the eCommand variable. Changes can be made both by external process control via the OPC UA interface of the controller and by internally programmed process control. In the ElementOfUnitController structure, the parameterization of the unit and its elements is to be carried out. While the aforementioned structure has a controlling character, in the structure aUnitInfo the state of the units and their elements (incl. the description of the plant components) is mapped. OPC UA clients have read-only access to this structure. ActMode and ActState represent the current mode and the current state of the underlying state machine.

Relevant supplementary process information is defined in the ”Process” structure. In the GVL (Fig. 8), relevant process variables are available in the ElementOfProcessInfo and ElementOfProcessControl structure. Whereby only the ElementOfProcessControl area can be used for manipulation via an OPC UA client or the associated AAS.

5 ADDITION OF PROCESS INFORMATION TO AAS TEMPLATE

One of the requirements for the generic generation of a AAS type 2 is the existence of an asset description as AAS type 1 (see Fig. 9) with the submodels (SM) ”Nameplate”, ”Identification”, ”Technical Data”, as well as the SM ”Communication” (option). This information is necessary to carry out the process of generation and to assign the required data from the controller. In the ”OpenBasys 4.0” research project, controllers from the manufacturers Beckhoff, Wago, Schneider Electric, as well as solutions based on firmware from Codesys were essentially used. During the generation process, the structures stored in the global variable list are assigned to the relevant submodels. So that after the generation process one of the following structures results. The above structures (Unit, Process) can be transferred selectively or together into the new asset structure. (cf. Fig.9) For the generator, it is in principle of secondary importance whether this is a controller, an edge controller or other automation device, since the submodels are created and assigned after successful identification on the basis of the OPC UA data structure. With the designated data structures, the entire range of PackML with its synchronization mechanisms is now available

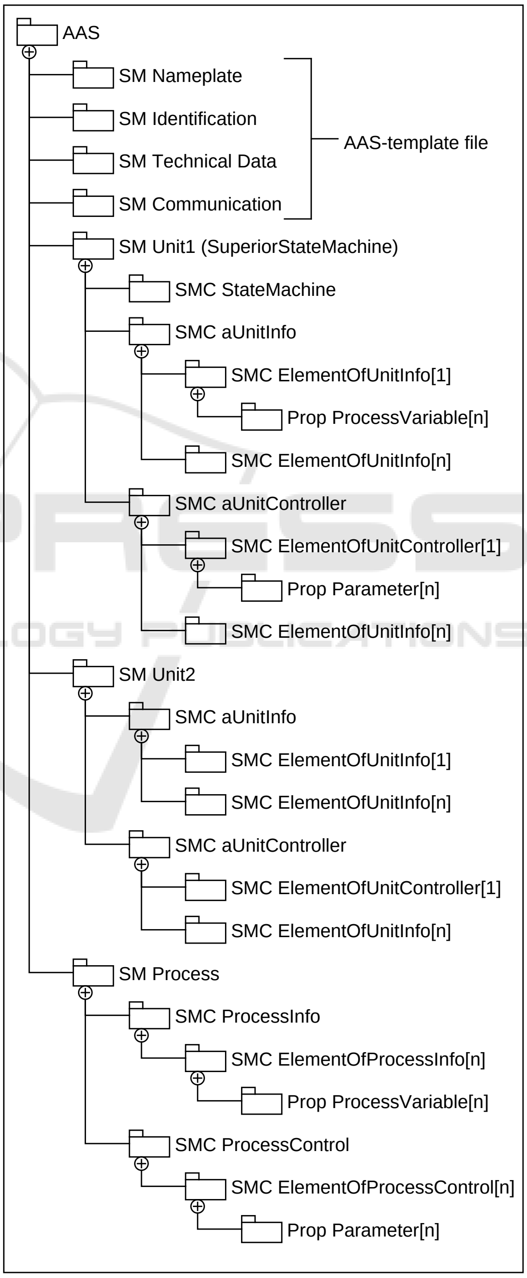

Figure 9: AAS - Supplement with SM ”Unit” (Schäfer et al., 2022).

to the AAS. This simplifies the integration and exchange of system components in the plant operator's I4.0 system environment.

6 REQUIREMENT FOR SYNCHRONIZATION

Exemplary templates with embedded PackML state machines (Fig. 3) and supplementary functions were prepared for different Codesys platforms. Since not only one process can be prepared on a process controller, but a large number of processes have been implemented, as in the present use case, their structuring and synchronization is mandatory. Here the encapsulation of processes in SubUnit-SM was considered. Their coordination and release is done with the "PackML Manager" (SuperiorStateMachine), which is also considered in the template. Like the "Group Component" from the "BaSyx Control Component", the "Superior State Machine" enables the superimposed use/control of subordinate capabilities. The initialization process including the booking of resources is exemplarily shown with figure 10. The individual states are offered in the templates according to the selected mode in the form of methods. Thus, at the level of the AAS it is always visible in which state the plant components/processes are, since all information is stored in the corresponding GVL (unit) and is available as an image in the AAS. Since not only the information from the state machine and its components are relevant for the user, it is made possible for the user to provide further information via another GVL of the AAS.

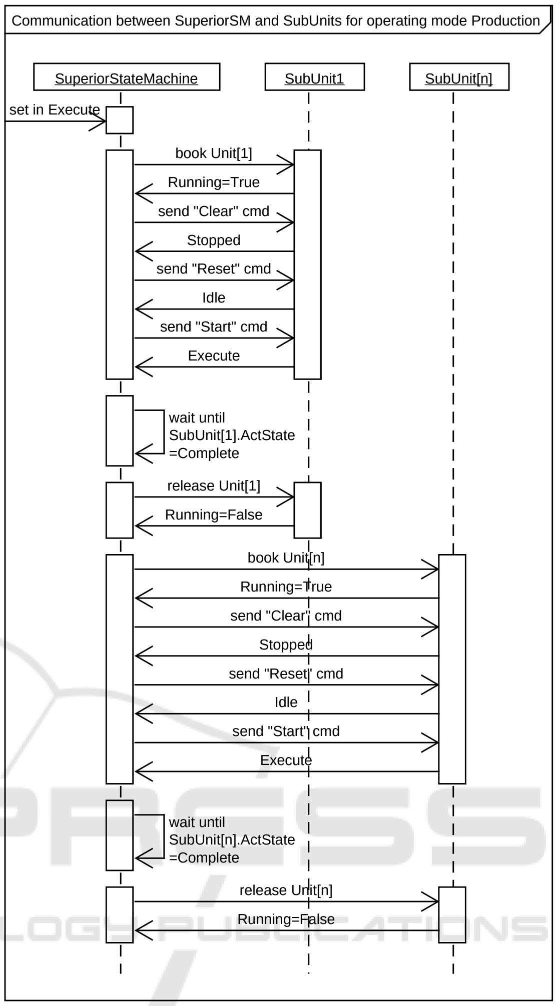

Figure 10: INIT-process PackML SM (Schäfer et al., 2022).

7 CONCLUSION - OUTLOOK

The article illustrates the synchronization of different plant segments with the help of a PackML template. The focus was set on the fact that especially existing plants can comply with an Industrie 4.0 standard. In this context, a proposal is made as to how structures can be stored in a PLC in order to be able to store them in an AAS.

By embedding individual SubUnits within a PLC, these can be described as atomic capabilities. This means that they can be orchestrated via a higher-level unit and combined into a composed capability. The "BaSyx Control Component" follows a similar pattern. However, since these do not yet correspond to a generally applicable standard, this contribution is oriented to the ISA 88 state machine (PackML). In order for this to correspond to the conditions of a control component according to BaSyx, various changes were made to an existing template, which cover a large part of the functions of the BaSyx CC. A management of different atomic skills to compsed skills was realized over a SuperiorStateMachine. Each SubUnit gets its own PackML SM, which is also passed on to the AAS. Thus, a uniform synchronization can be achieved across all plant segments.

Through the agile combination of different skills, a high degree of changeability of the system can be achieved without having to carry out time-consuming programming work.

REFERENCES

Adolph, D. L., Ammon, E., Becker, J., and Bedenbender, H. e. a. (2020). German standardization roadmap industrie 4.0.

Arens, D., Hopfgartner, T., Jensen, T., Lamping, M., Pieper, M., and Seger, D. (2006). Packaging machine language v3. 0 mode & states definition document. *OMAC Motion for Packaging Working Group*.

Bader, S., Barnstedt, E., Bedenbender, H., Berres, B., Billmann, M., and Boss, B. (2022). Details of the asset administration shell: Part 1: The exchange of information between partners in the value chain of industrie 4.0.

Bayha, A., Bock, J., Boss, B., Diedrich, C., and Malakuti, S. (2020). Describing capabilities of industrie 4.0 components.

Bedenbender, H., Bock, J., Boss, B., Diedrich, C., Garrels, K., Graf Gatterburg, A., Heidrich, K., Hillermeier, O., Rauscher, B., Sauer, M., Schmidt, J., Werner, T., and Zimmermann, P. (2020). Verwaltungsschale in der praxis: Wie definiere ich teilmodelle, beispielhafte teilmodelle und interaktion zwischen verwaltungsschalen. Plattform Industrie 4.0, Discussion Paper.

Belyaev, A., Diedrich, C., Espen, D., Lüdicke, B., Miny, T., Mrosowski, A., Müller, M., Neidig, J., Pollmeier, S., Reich, J., Rübel, P., Sauer, M., Schröder, T., Ulrich, O., Urban, C., Vojanec, B., and Wieczorek, M. (2021). Vws-referenzmodellierung: Exemplarische modellierung einer fertigungstechnischen anlage mit aasx package explorer auf basis des vws-metamodells.

BMBF (2019). Basys 4.0 in der anwendung - verbundprojekt openbasys 4.0.

Bouter, C., Pourjafarian, M., Simar, L., and Wilterdink, R. (2021). Towards a comprehensive methodology for modelling submodels in the industry 4.0 asset administration shell. In *2021 IEEE 23rd Conference on Business Informatics (CBI)*, volume 2, pages 10–19. IEEE.

Cavalieri, S. and Salafia, M. G. (2020). Asset administration shell for plc representation based on iec 61131–3. *IEEE Access*, 8:142606–142621.

Di Orio, G., Maló, P., and Barata, J. (2019). Novaas: A reference implementation of industrie4. 0 asset administration shell with best-of-breed practices from it engineering. In *IECON 2019-45th Annual Conference of the IEEE Industrial Electronics Society*, volume 1, pages 5505–5512. IEEE.

Dorofeev, K. and Zoitl, A. (2018). Skill-based engineering approach using opc ua programs. In *2018 IEEE 16th international conference on industrial informatics (INDIN)*, pages 1098–1103. IEEE.

Fathizadeh, M., Yen, J., and Werthman, M. (2013). Integration of packml in engineering education. In *Proceedings of the World Congress on Engineering and Computer Science*, volume 1.

Grothoff, J., Porta, D., Espen, D., and Haque, A. (2021). Basyx/documentation/api/controlcomponent.

Grüner, S., David, A., Motsch, W., Grothoff, J., Fachinger, G., Miny, T., Schaller, G., Hamann, T., Johann, P., and Krämer, N. (2021). D-4.8 entwurf einer generischen plug & produce prozedur.

ISA (2015). Isa-tr88.00.02:2015.

Koulamas, C. and Kalogeras, A. (2018). Cyber-physical systems and digital twins in the industrial internet of things [cyber-physical systems]. *Computer*, 51(11):95–98.

Kuhn, T., Schnicke, F., and Antonino, P. O. (2020). Service-based architectures in production systems: Challenges, solutions & experiences. In *2020 ITU Kaleidoscope: Industry-Driven Digital Transformation (ITU K)*, pages 1–7. IEEE.

Lüder, A., Behnert, A.-K., Rinker, F., and Biffl, S. (2020). Generating industry 4.0 asset administration shells with data from engineering data logistics. In *2020 25th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA)*, volume 1, pages 867–874. IEEE.

Mušič, G. (2015). A low-cost packml-based control solution for a modular production line. *IFAC-PapersOnLine*, 48(10):184–189.

OMAC (2009). Packml implementation guide.

Platform, E. B. (2021). Ready-to-use components and extendable software development kits (sdk).

Rayment, M. (2004). Flexible motion control using iec 61131-3. In *2004 Mini Symposia UKACC Control*, pages 27–36. IET.

Repository, G. (2022). A viewer/editor for the asset administration shell.

Schäfer, S., Schöttke, D., Kämpfe, T., Ralinovski, K., Tauber, B., and Lehmann, R. (2021). Design and deployment of digital twins for programmable logic controllers in existing plants. In *IN4PL*, pages 145–150.

Schäfer, S., Schöttke, D., Kämpfe, T., Lachmann, O., Zielstorff, A., and Tauber, B. (2022). Migration and synchronization of plant segments with asset administration shells. unpublished - will be published in the ETFA2022 conference proceeding.

Wallner, B., Trautner, T., Pauker, F., and Kittl, B. (2021). Evaluation of process control architectures for agile manufacturing systems. *Procedia CIRP*, 99:680–685.

Ye, X., Hong, S. H., Song, W. S., Kim, Y. C., and Zhang, X. (2021). An industry 4.0 asset administration shell-enabled digital solution for robot-based manufacturing systems. *IEEE Access*, 9:154448–154459.