

A dark blue vertical bar runs down the left side of the slide. A blue arrow points to the right from this bar, containing the date.

9/28/2020

Modelling production system components as Industry 4.0 components

Several thin, curved lines in dark blue and light grey originate from the bottom left and sweep upwards and to the right.

Nishad Pawaskar, Akshay Sharma, Sukreet Pal

Declaration

We hereby declare that the project entitled “**Modelling production system components as Industry 4.0 components**” written and submitted by us to Otto von Guericke Universität Magdeburg is an authentic record of our genuine work done under the guidance of **apl. Prof. Dr.-Ing. habil. A. Lüder** and **Dipl. -Wirtsch. -Ing. Christian Deppe**.

Group members:

1. Nishad Pawaskar (225992) _____
2. Akshay Sharma Nekkare Krishna Prasad (229740) _____
3. Sukreet Pal (225928) _____

Date: **28 September 2020**

Place: **Magdeburg**

Acknowledgement

This masters project was supported by Otto-von-Guericke Universität and Festo Didactic GmbH. We would like to pay our sincere gratitude to **apl. Prof. Dr. -Ing. habil. A. Lüder** and **Dipl. -Wirtsch. -Ing. Christian Deppe**, representative from Festo Didactic GmbH for providing us with their valuable insights and expertise. Without their precious support, it would be not possible to conduct this project.

Last but not the least, the dedicated efforts and cooperation between all the group members, has crucially helped in concluding the project.

Table of Contents

List of Figures	5
1 Introduction	6
1.1 Background	6
1.2 Objectives and scope	6
1.3 Sources.....	7
1.4 Structure of work	7
2 Festo Didactic.....	8
2.1 CP Factory	8
2.2 CIROS Studio	8
2.3 Simulation model	9
2.3.1 Large Process	9
2.3.2 Medium Process.....	10
2.3.3 Small Process.....	11
3 Asset Administration Shell	12
3.1 Reference Architectural Model for Industry 4.0 (RAMI 4.0).....	12
3.1.1 Axis 1 - Hierarchy: The Factory	13
3.1.2 Axis 2 - Product Life Cycle	13
3.1.3 Axis 3 - Architecture.....	14
3.2 Structure of Asset Administration Shell.....	14
3.2.1 Identifier.....	15
3.2.2 Submodel	16
3.2.3 Property	17
3.2.4 Range	17
3.2.5 Operation	17
3.2.6 Collection	17
3.2.7 Entity	17
3.2.8 Relationship	18
3.2.9 View.....	18
3.2.10 Concept Description.....	18
3.3 Asset Administration Shell Package Explorer	18
3.3.1 Package File format for AAS (AASX).....	19
4 Modelling of the system	21
4.1 Base Module - CP Factory Linear (CP-F-LINEAR).....	21
4.1.1 Module Layout	21

4.1.2	Module Design	22
4.1.3	Module Relationships	23
4.2	Application Module- Muscle Press (CP-AM-MPRESS)	25
4.2.1	Module Layout	26
4.2.2	Modelling of the module	26
4.2.3	Submodel contents	27
4.2.4	Relationships between submodel elements.....	29
4.2.5	Module views and Instances.....	32
5	Comparison & Evaluation between two modes of modelling.....	35
5.1	Modelling of CP-AM-MPRESS Method 1.....	35
5.2	Modelling of CP-AM-MPRESS Method 2.....	36
6	Introduction to AutomationML.....	39
7	Interlinking AASX and AML	41
8	Conclusion.....	42
9	References	43

List of Figures

Figure 1: CP Factory – Large process.....	9
Figure 2: CP Factory – Medium process.....	10
Figure 3: CP Factory – Small process.....	11
Figure 4: Reference Architectural Model for Industry 4.0	12
Figure 5: Industry 3.0 architecture (Left); Industry 4.0 architecture (Right)	13
Figure 6: Product life cycle explained by an axis of RAM I4.0 model.	14
Figure 7: Usage View architecture of RAMI4.0 model.....	14
Figure 8: Basic structure of Asset Administration Shell	15
Figure 9: Possible submodels of Asset Administration Shell	16
Figure 10: Entity Attributes.....	18
Figure 11: Structure of AAS package explorer	19
Figure 12: Example of a logical model for the AAX format.....	20
Figure 13: Example of a physical model based on a sample product and its mapping to the logical model ..	20
Figure 14: CP Factory Linear Module	21
Figure 15: CP Factory Linear – AAS Model.....	22
Figure 16: CP-F-LINEAR Submodel – Information	23
Figure 17: CP-F-LINEAR Submodel - Communication	23
Figure 18: Information submodel with relationship elements from Communication submodel.....	24
Figure 19: Relationship between parent AAS (CP-F-LINEAR) and child AAS (Miniature circuit breaker)	24
Figure 20: UML diagram representing parent-child relationship between asset administration shells	25
Figure 21: Application module – CP-AM-MPRESS.....	25
Figure 22: CP Factory Muscle Press AAS Model.....	26
Figure 23: Additional submodels	27
Figure 24: CP-AM-MPRESS submodel – Engineering	27
Figure 25: CP-AM-MPRESS submodel – Communication.....	28
Figure 26: CP-AM-MPRESS submodel – Configuration	28
Figure 27: CP-AM-MPRESS submodel – Information	29
Figure 28: CP-AM-MPRESS submodel – Measurement	29
Figure 29: Relationship between reference tag & corresponding communication interface	30
Figure 30: Relationship between IO signal & the corresponding communication interface.....	30
Figure 31: Relationship between asset & entity	31
Figure 32: Asset Types & Instances.....	31
Figure 33: Referencing of instance assets to AAS	32
Figure 34: Instance class of a component (Contact block)	33
Figure 35: Template class of a component (Contact Block).....	33
Figure 36: UML diagram representing relationship between submodel elements.....	34
Figure 37: Structuring of parent and child asset.....	35
Figure 38: Derived from feature to establish parent-child relationship.....	36
Figure 39: Structuring of assets as co-managed entities	37
Figure 40: Additional defined submodels for listing down entities & their associated data.....	37
Figure 41: Basic architecture of AML	39
Figure 42: Required information sets to be presented in AML	40
Figure 43: Screenshot of the exported aasx file opened in AML.....	41

1 Introduction

Industrial revolution has brought major changes in the field of manufacturing and also influenced the economic growth of the countries. The Industrial revolution is the continuous transformation of social, economic and working conditions. The Industry 4.0 (or fourth industrial revolution) is the ongoing automation of traditional manufacturing and industrial practices, using modern smart technology. Large-scale machine-to-machine communication and the internet of things (IoT) are integrated for increased automation, improved communication and self-monitoring, and production of smart machines that can analyse and diagnose issues without the need for human intervention [1].

1.1 Background

Producing companies have to cope with increasing individualisation of products, increased quality requirements, increased technological progress, market requirements, etc which encourages the integration of upcoming and existing IT technologies with several use cases along the life-cycle of the production system under Industry 4.0. The use of Industry 4.0 requires the components to be interoperable so that they seamlessly communicate across companies and industries. Asset Administration Shell (AAS) provides the base for development and use of such Industry 4.0 components.

The Industry 4.0 component is regarded as self-controlling, self-planning and self-maintaining Cyber physical system where a physical and/ or a logical asset is controlled by an Asset Administration Shell (AAS). These components form a hierarchy of cooperating entities all together establishing a production system with its involved resources, production processes executed, and the products produced. The enormous amount of information and data resource used in and for the Industry 4.0 has to be maintained in a structured manner. Also, in order to individualise and modularise the production system components, we need to provide all the relevant information and capabilities. The Asset Administration Shell (AAS) then gives a way to provide the information of the structure, usability, capability and use conditions of the assets and enable & control the access to it.

1.2 Objectives and scope

The goal of the master's project is to investigate and evaluate the possibilities of modelling an Asset Administration Shell (AAS) structure for Industry 4.0 system component by considering the example of a physical production system. We considered one of the production systems from the Festo Didactic learning systems for Industry 4.0 – Cyber-Physical Factory (or *CP-Factory*).

One of the application modules of the CP-Factory is modelled as an Industry 4.0 component using asset administration shell package explorer as a tool. The capability of AutomationML to interlink with the AAS model also needs to be tested. Modelling a production system component as an Industry 4.0 component required –

- Identification of the asset and its stakeholders
- Identification of the properties, functions and capabilities of the asset
- Understanding the structure of asset administration shell

1.3 Sources

The main sources of the literature for the platform Industry 4.0 based Asset Administration Shell (AAS) modelling were, the ZVEI Elektroindustrie website and corresponding research papers. Also, the Platform Industry 4.0 website maintained by Federal Ministry of Education and Research, Germany, provides a number of research papers and technical publications on administration shell in the context of Industry 4.0. Most of the project work is supported by the document “*Details of Asset Administration Shell – Part1: The exchange of information between partners in the value chain of Industry 4.0 (Version 2.0)*”. Information regarding the Festo production system and the necessary properties of the system components are available at Festo didactic InfoPortal at [2].

1.4 Structure of work

The forthcoming units summarize the relevant information regarding the system, modelling structure and the analysis of the models. The chapter 2 gives an overview of the CP Factory model, a subset of which is modelled according to data structuring strategies, evaluated and analysed. Chapter 3 & 4 deals with a general overview and information structure in Asset Administration Shell (AAS) and modelling in AAS package explorer respectively.

Analysis and evaluation of the created model is presented in chapter 5. Basic structure of AutomationML, and interlinking AML and AASX is discussed in Chapter 6 and Chapter 7 respectively. Chapter 8 concludes the project work and suggests an area of implementation for each modelling strategy used.

2 Festo Didactic

Festo Didactic is a leading service provider in the field of technical education. Festo Didactic is not limited to educational sector, it also plays a crucial role in research and development in the field of production. Festo is an active member of the association ZVEI and is contributing to Industry 4.0 with the help of CP factory, a Festo didactic learning system for Industry 4.0 which illustrates the practical implementation of digital production. The reason for selecting the CP Factory learning system is that, it is small enough to be used for educational purpose and complicated enough to be used in Research and Development. According to Festo, the CP Factory smartly integrates the relevant technologies of mechatronics and automation [3].

2.1 CP Factory

The CP in CP-Factory stands for Cyber-physical system. The cyber-physical system enables the intelligent networking of physical systems, which is one of the important characteristics of Industry 4.0. The CP-Factory learning system covers a variety of areas in production like CNC assembly, robot assembly, conveyor and diverter assembly, heating assembly, muscle press, etc shown in Figure 1. These assemblies are available on both sides of the linear system and are made into standard modularized stations, because of which the stations can be arranged in a number of ways. The process control of the CP factory is highly dependent on the product and hence is equipped with RFID technology. MES4, specially developed for use in learning systems, is used as the production control system. It helps creating, managing and visualising sales orders and also allows the production cells to receive information as needed.

The work flow process of the factory varies according to the product and is divided based on three processes – small process, medium process and large process. The CP Factory has a Warehouse with a mechanism for loading and unloading, component assembly with a robot, a CNC mechanism, an Optical vision system, a Muscle press module, Cover placing module and other application modules such as drilling, heating, labelling, etc. For the sake of simplicity and convenience, small process was selected and a submodule was modelled in AAS which is discussed later in this report.

2.2 CIROS Studio

CIROS provides an environment to render and view realistic 3D simulation of any automation system. Furthermore, it can be linked with MES4 for resource management. CIROS is available in two formats – Education and Studio for downloading at [4]. In order to have all the functionalities of CIROS, the respective license is required. The setup is only available for Microsoft Windows.

CIROS studio is a professional working tool for creating simulation models. It is used for creating simulations, modelling and also programming. It supports all basic CAD functions. It provides libraries for the industrial robot systems and various autonomous systems. CIROS education offers all the functionalities of CIROS studio and mainly focuses on training in search for malfunctions, production planning and control, PLC programming and Robot programming. CIROS provides all the simulations models for Festo Didactic learning systems in the field of automation, CP-Factory being one of them.

CIROS also supports various scenarios of PLC simulation. It is also possible to connect a real PLC with CIROS via EasyPort which would enable CIROS to receive PLC outputs, transmit sensor values to PLC and simulate the controlled process. This makes CIROS very convenient to use in automation technology for learning as well as development.

2.3 Simulation model

As mentioned earlier, the work flow of the CP factory depends on the product. Depending on the product, there are three work flows of the CP factory namely small process, medium process, large process. The simulation model of each work flow was provided in CIROS studio.

A GPS bike-computer is to be produced in CP factory. The product is classified into two as road and off-road and further classified based on the memory (16GB / 32GB / 64GB). 64GB product requires large process, 32GB requires medium process, while 16GB requires small process. These processes have minor changes based on the off-road type. However, we consider only the product for the road and not off-road.

2.3.1 Large Process

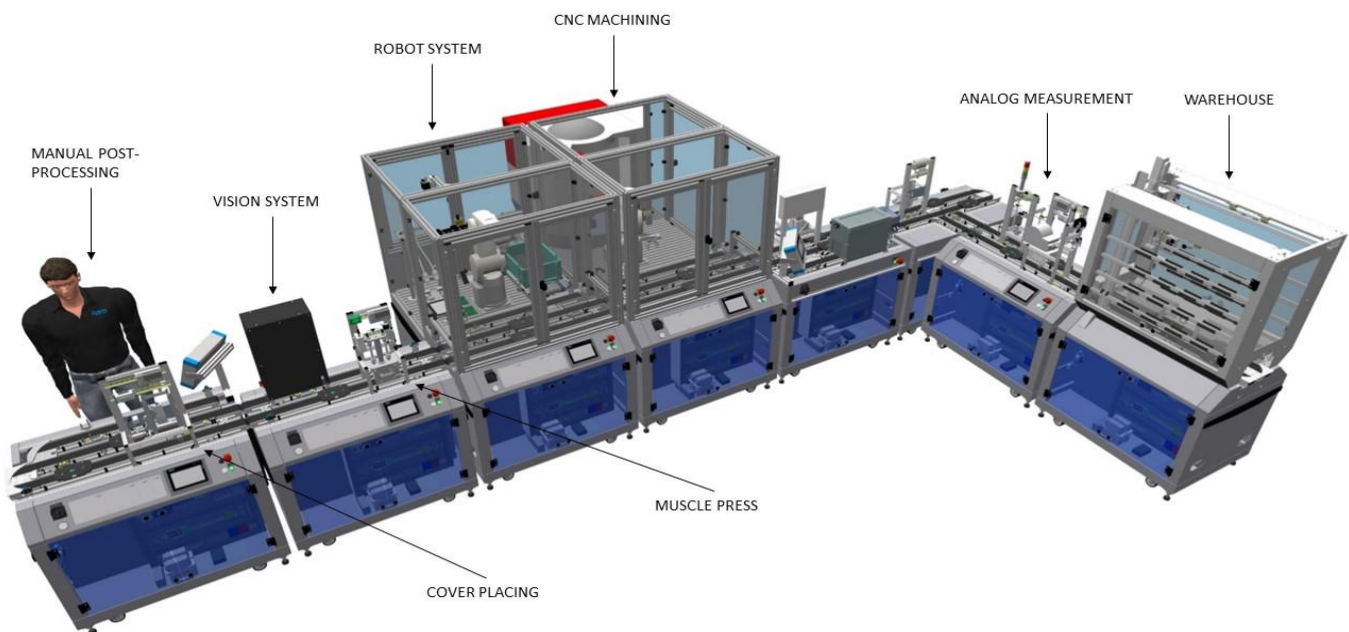


Figure 1: CP Factory – Large process

A snapshot of the simulation model of the large factory process from CIROS studio is as shown in Figure 1. The workflow of the large process along the CP-Factory is as follows:

Warehouse → Analog Measurement → Turn-table → CNC assembly → Robot assembly → Optical vision system → Cover placing machine → Muscle press assembly → Labelling → Warehouse.

The process involves 11 steps in general and they are as follows.

1. Removal of the front cover of the corresponding colour from the warehouse.
2. Analog measurement to check the position of the front cover.
3. Turning the front cover, if necessary.
4. CNC machining of the front cover.
5. Assembly of the circuit board and fuses using Robot.
6. Optical check for the correct placement of the fuses using vision system.
7. Possibly manual post-processing if step 6 found an error.
8. Put on the black back cover using cover placing module.

9. Pressing both the housing halves using muscle press module.
10. Labelling of the product.
11. Storing the finished product in the warehouse.

2.3.2 Medium Process

A snapshot of the simulation model of the medium factory process from CIROS studio is as shown in Figure 2. The workflow of the medium process along the CP-Factory is as follows:

Warehouse → Drilling machine → Robot assembly → Optical vision system → Cover placing machine → Muscle press assembly → Turn-table → Warehouse.

The process involves 9 steps in general and they are as follows.

1. Removal of the front cover of the corresponding colour from the warehouse.
2. Drilling of the front cover.
3. Assembly of the circuit and fuses using Robot.
4. Optical check for the correct placement of the fuses using vision system.
5. Possibly manual post-processing if step 4 found an error.
6. Put on the black back cover using cover placing module.
7. Pressing both the housing halves using muscle press module.
8. Turning the product.
9. Storing the finished product in the warehouse.

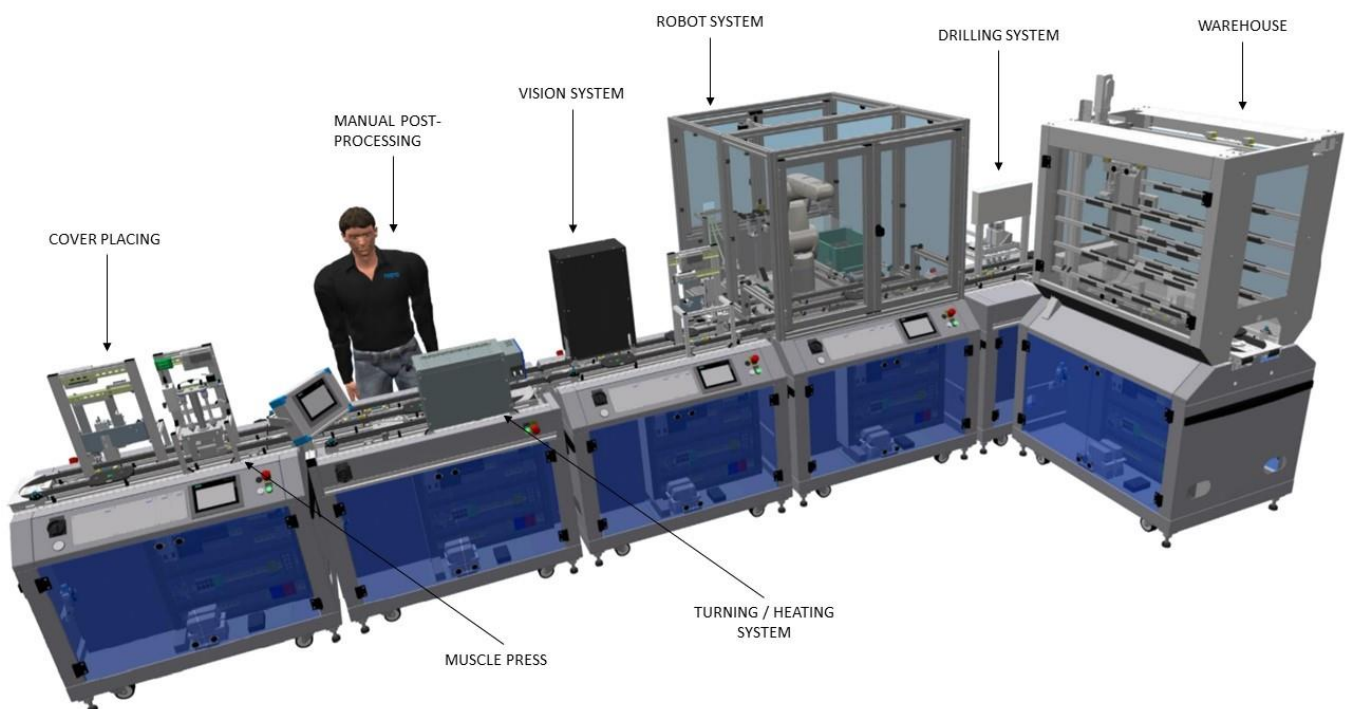


Figure 2: CP Factory – Medium process

2.3.3 Small Process

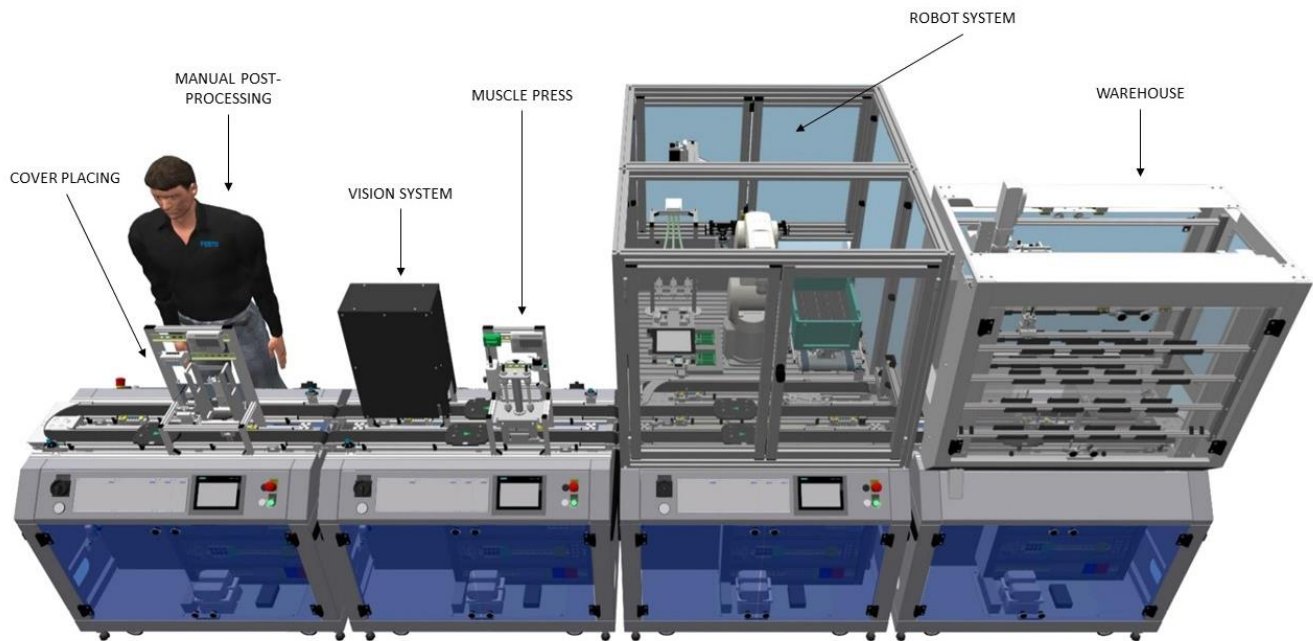


Figure 3: CP Factory – Small process

A snapshot of the simulation model of the small factory process from CIROS studio is as shown in Figure 3. The workflow of the small process along the CP-Factory is as follows:

Warehouse → Robot assembly → Optical vision system → Cover placing machine → Muscle press assembly → Warehouse.

The process involves 7 steps in general and they are as follows.

1. Removal of the front cover of the corresponding colour from the warehouse.
2. Assembly of the circuit board and fuses using Robot.
3. Optical check for the correct placement of the fuses using vision system.
4. Possibly manual post-processing if step 3 found an error.
5. Put on the black back cover using cover placing module.
6. Pressing both the housing halves using muscle press module.
7. Storing the finished product in the warehouse.

3 Asset Administration Shell

The asset is anything that needs to be connected in order to create an Industry 4.0 solution, i.e. all the items that are of a certain value in a specific case. In Industry 4.0, an asset can be a physical product or piece of equipment, software item or any document [5]. The Asset Administration Shell is the interface connecting Industry 4.0 standards to the physical assets, and stores all the data and information regarding the asset. The Administration shell is addressable in the network and identifies the asset unambiguously and provides a standardised and secure communication interface. It is a corner stone of interoperability between the applications managing the manufacturing systems. The description of the asset and the properties in the administration shell are described based on RAM architecture for Industry 4.0. This makes the Asset Administration Shell

- To establish cross-company interoperability.
- To be available for both non-intelligent (passive) and intelligent assets.
- Cover the complete life cycle of the assets and services.
- Enable integrated value chains.
- To be the digital basis for autonomous systems and AI.

The Asset Administration Shell helps implement the digital twin for Industry 4.0 and create interoperability across the components of different suppliers, where digital twin is the digital representation of any physical asset. The 'Asset Administration Shell' and 'Administration Shell' will be used synonymously throughout this report.

3.1 Reference Architectural Model for Industry 4.0 (RAMI 4.0)

In order to model any system component as an Industry 4.0 component, it is important to know the RAMI 4.0 model. The **RAMI 4.0** is three dimensional and organises the life-cycle, value streams and the manufacturing hierarchy levels across the six layers of the IT representation of Industry 4.0 (Figure 4). The first axis defines the factory hierarchy, second axis defines the product life-cycle, while the third axis defines the organisational architecture. We can say that, RAMI 4.0 provides us a solution space with coordinate system for Industry 4.0.

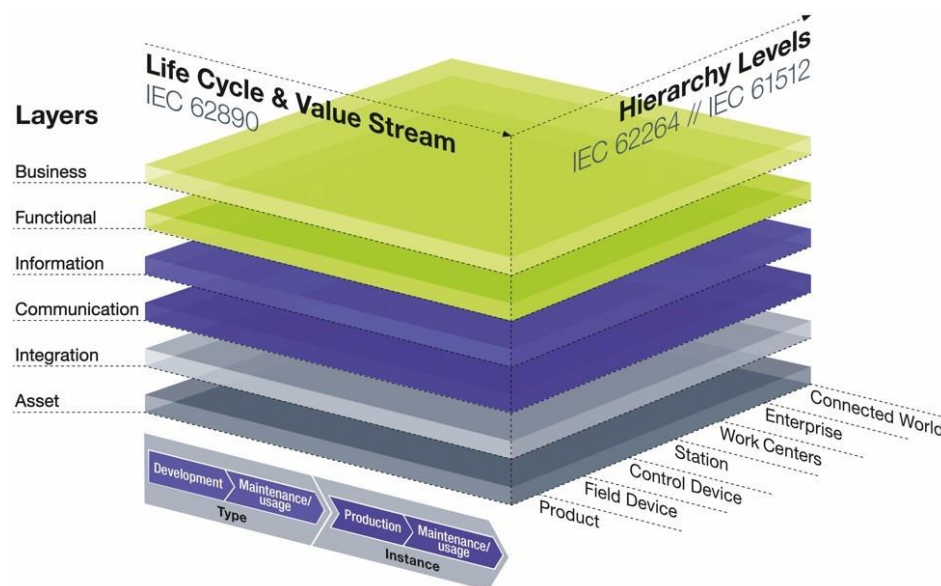


Figure 4: Reference Architectural Model for Industry 4.0

RAMI 4.0 can also be seen as a map showing how to approach the issue of Industry 4.0 in a structured manner. The RAMI architecture ensures that all the participants (components, devices, machines, softwares) involved in Industry 4.0, can communicate with each other without any hassle. It combines all the elements and IT components in a layer and life-cycle model. It breaks down complex processes into simple packages, including data privacy and IT security.

3.1.1 Axis 1 - Hierarchy: The Factory

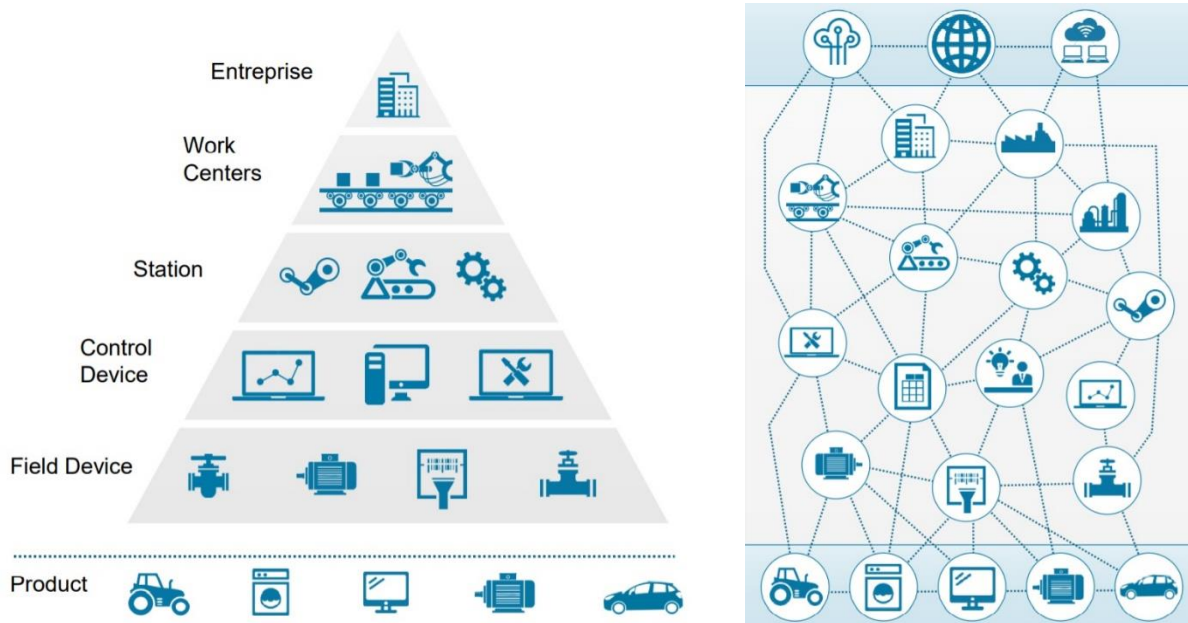


Figure 5: Industry 3.0 architecture (Left); Industry 4.0 architecture (Right)

Looking at the images above, first and the most important difference we see is that the product is isolated from rest of the structure in Industry 3.0, while it is a part of the Industry 4.0 network. In Industry 3.0, the product is at lowest level while the enterprise or the business is at the highest level. The structure earlier was purely hardware based and the functions were bound to hardware. Also, the communication between the components was hierarchy based. With the advent of Industry 4.0, the devices or components were now able to communicate with every other device or component in the network irrespective of the level of hierarchy it is present in. Unlike the previous structure, the Industry 4.0 structure has flexible system and the functions are distributed throughout the network. Communication exists among all the participants of the system and hence is also called as the *connected world*. Each participant interacts across hierarchy levels. Each level of the industry architecture has access not just to the product data but also the data from each other level. Hence, it is important to have a structured modelling approach.

3.1.2 Axis 2 - Product Life Cycle

The product life cycle, as its name suggests, represents the complete life-cycle of the product from the First idea to the Scrapyard. In the *Type* section, the product is in construction and development phase. While in *Instance*, production, development and maintenance of the product is done. Consider a product is manufactured and developed by a company 'X' and sold to a company 'Y'. The product manufactured and developed at 'X' has kind *Type* and when sold to 'Y', an instance of the product is created for 'Y'.

Another such instances will be created for each customer. The life cycle of the instance can be also maintained based on the state/ phase such as maintenance, recycling, scrapping, etc.

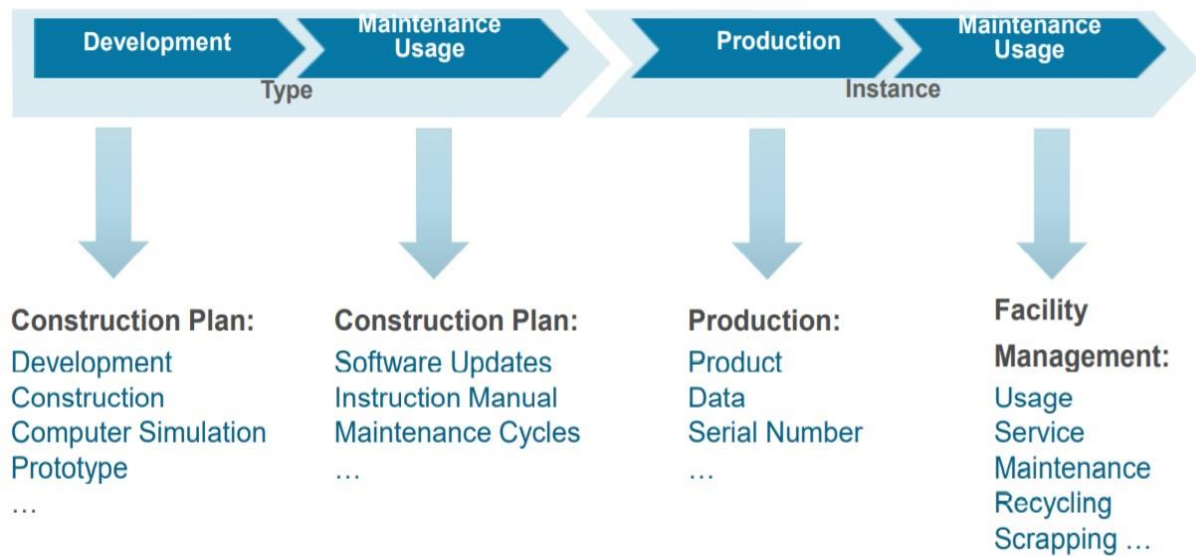


Figure 6: Product life cycle explained by an axis of RAM I4.0 model.

3.1.3 Axis 3 - Architecture

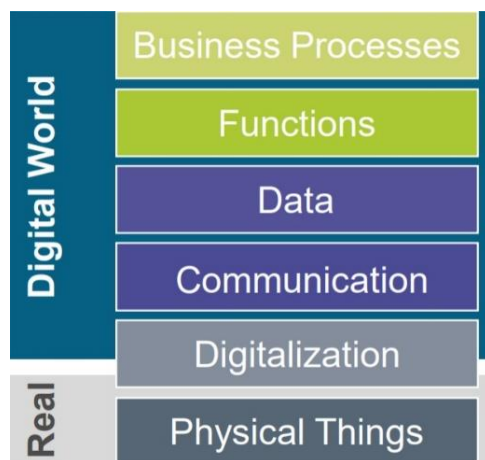


Figure 7: Usage View architecture of RAMI4.0 model.

The third axis of the RAMI 4.0 model is divided into six layers based on the usage view of the asset. The bottom most layer is the asset itself which can be any physical thing in the real world. It doesn't matter whether the asset is an intelligent device/ machine or not, as long as it is of a certain value to the stakeholder. The second layer allows digitalization of the physical asset and integrates it into the digital world. Hence, this layer is also called as the *Integration layer*. Communication and Information forms the third and fourth layers respectively. These layers enable the access of information from all the connected devices and secure collection of necessary data (on request).

The functional layer defines all the functions of the asset. Programming of the device and its operation is done in this layer. The business layer is responsible for the organisational and other business purposes. Identification of other stakeholders, resource planning and management, etc are some of the tasks done in the business layer. This architecture can be used as basis for *Views* for various stakeholders in Asset administration shell.

3.2 Structure of Asset Administration Shell

In order for the Administration shell to represent any physical asset, it is important to know the properties, relations and functions of the asset. This can be represented in the administration shell in a structured manner. The Asset Administration Shell comprises of a *Header*, which contains the

identification details of the shell and the represented asset, and a *Body*, which contains all the other properties related to the functions of the asset, relationships, security, etc.

The Asset Administration Shell follows an object-oriented paradigm. Each asset is modelled as an object and the administration shell can be seen as the Class. It has all the characteristics of an object-oriented methodology and implements the fundamental principles of *Inheritance*, *Polymorphism*, *Abstraction*, and *Encapsulation*.

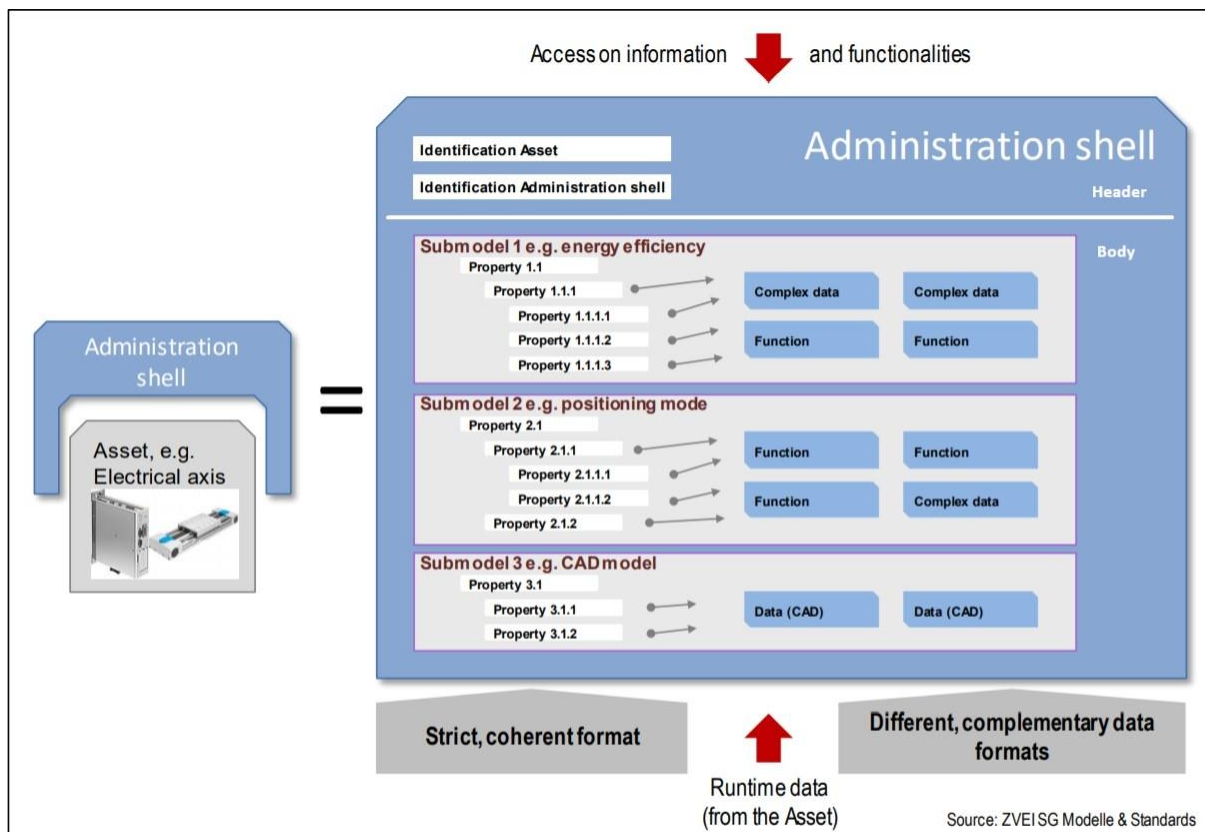


Figure 8: Basic structure of Asset Administration Shell

As shown in above figure, the administration shell is made up of submodels, which represent various properties and functions of the concerned asset. Defining an administration shell for an asset requires us to consider a number of attributes pertaining to the asset. These attributes are a part of the class used to define a specific functionality or data of the asset. To enable binding semantics, asset administration shells, assets, submodels and properties must be clearly identified [6].

3.2.1 Identifier

The identifier, as the name suggests, provides the identity information that distinguishes one object from another. It can also be understood as a label used to identify an object. There are numerous ways of doing so, based on the standards used and object to be identified. Hence, it is necessary to provide the *IdType* along with the *IdName*. Globally unique identifier (GUID), International registration data identifier (IRDI), Internationalised resource identifier (IRI) and Uniform resource identifier (URI) are few identifiers used in AAS Explorer. Identifier is mandatory to be defined as this makes it easy to find and identify any object or process.

3.2.2 Submodel

Submodel is nothing but a model technically separated from other models and is a part of the asset administration shell [6]. A Submodel, as mentioned in [7], is used to structure the digital representation and technical functionality of an AAS into distinguishable parts. Each submodel refers to a well-defined domain which provides the relevant information and data of the asset using a set of properties. An administration shell is made up of a number of submodels. Submodel can have its own life-cycle and hence the concepts of *Template* and *Instance* can also be applied. Some submodels can be standardized according to the convenience and can become *submodel template*.

Consider “Drilling” as an example of a standardized submodel. Here, the submodel has standardized the drilling process, modelled as a template, which can be performed by mentioning only the varying parameters like type of material, size of the hole, etc. There can be an instance of the drilling with a different size of hole to be drilled or an instance with different amount of power required depending on the material. Submodels are further explained in more detail in the paper for “Examples of Asset administration shell for Industry 4.0 components” [6].



Source: ZVEI

Figure 9: Possible submodels of Asset Administration Shell

The submodel element provides a description and differentiates assets with the help of data properties, operations, events, capabilities and other elements. The data element can also be seen as a submodel element which has a value or a number of values like range data. Object descriptions can also be provided in a File or a Binary Object (BLOB) file. Certain properties are provided in a repository such as eCl@ss. Such elements are all a part of data element which in turn are submodel elements. The data elements (like reference element, property, file, range), submodel element collection, operation, relationship element, events and entities are all submodel elements which can be organized and managed keeping in mind the future amendments to the Industry 4.0 component.

3.2.3 Property

The property is a data element with a single value. Some properties of the entity can be described in multiple languages. Such properties are defined as Multi-Language property and a single property value is defined in multiple languages. The property type can be defined in dictionaries like eCl@ss or IEC component data dictionary. The property describes one characteristic of an object. It can be an attribute such as code, revision or version. The property of an entity includes nominal value, actual value, runtime variables, etc.

3.2.4 Range

The range is a data element which defines a range of values of a property with *min* and *max* value. This signifies that a value of a property lies within the range of min value and max value. If the minimum value of the property is not provided, then it is considered to be negative infinity. If maximum value is not provided, then it is considered to be positive infinity.

3.2.5 Operation

The execution of a function of an asset is known as an Operation. The operation needs a list of parameters or variables in order to complete the execution of the function. Operation is a submodel element with input and output variables, whereas Operation variable is a submodel element that can be used as input or output for an operation of a function.

3.2.5.1 Input Variable

This is a type of an operation variable. Input variable is an input parameter of the operation.

3.2.5.2 Output Variable

This is a type of an operation variable. Output variable is an output parameter of the operation.

3.2.5.3 Inoutput Variable

This is also a type of an operation variable. Inoutput variable is a parameter that is input as well as output to the operation.

3.2.6 Collection

The Collection is a submodel element which is a set or a list of other submodel elements. A collection can allow duplication too. An ordered collection is implemented as an indexed array while, by default, the collection is unordered.

3.2.7 Entity

Entity, in general, means something that has separate and distinct existence and objective. Here, entity is a submodel element that is used to model entities. The image below gives a better understanding of the Entity submodel element.

Class:	Entity			
Explanation:	An entity is a submodel element that is used to model entities.			
Inherits from:	SubmodelElement			
Attribute (* = mandatory)	Explanation	Type	Kind	Card.
statement	Describes statements applicable to the entity by a set of submodel elements, typically with a qualified value.	SubmodelElement	aggr	0..*
entityType*	Describes whether the entity is a co-managed entity or a self-managed entity.	EntityTypeEnum	attr	1
asset	Reference to the asset the entity is representing. <u>Constraint AASd-014:</u> The asset attribute must be set if entityType is set to "SelfManagedEntity". It is empty otherwise.	Asset	ref*	0..1

Figure 10: Entity Attributes

3.2.8 Relationship

A relationship element is used to define a relationship between two referable elements. It has two attributes – first and second. The attribute *first* refers to the first element in the relationship taking the role of the Subject, while *second* refers to the second element in the relationship taking the role of the object.

3.2.9 View

View is a projection of a model or models from a given perspective and omits entities that are not relevant to this perspective. A large number of submodel elements can be filtered by views, so that different user groups can see only relevant elements. It can be seen as a collection of elements with respect to a specific viewpoint of one or more stakeholders. For example, an *instance* (product for the customer) of an asset need not be provided with all the information, which may be only relevant for asset of kind *type* (product at the manufacturer).

3.2.10 Concept Description

A concept description defines a concept or the semantics of a property or other elements that may have a semantic description. Different types of submodel elements require different attributes for describing the semantics of them. That is why, concept description is associated with at least one data specification template.

3.3 Asset Administration Shell Package Explorer

In order to create and implement any idea, a tool is required. Modelling an Industry 4.0 component needs a software tool and a well-defined file format for interoperability. There are a couple of software available for implementation of Asset Administration Shell such as '*aasx package explorer*' and '*BaSyx*'. We use aasx package explorer to create asset administration shell models of the Industry 4.0 components.

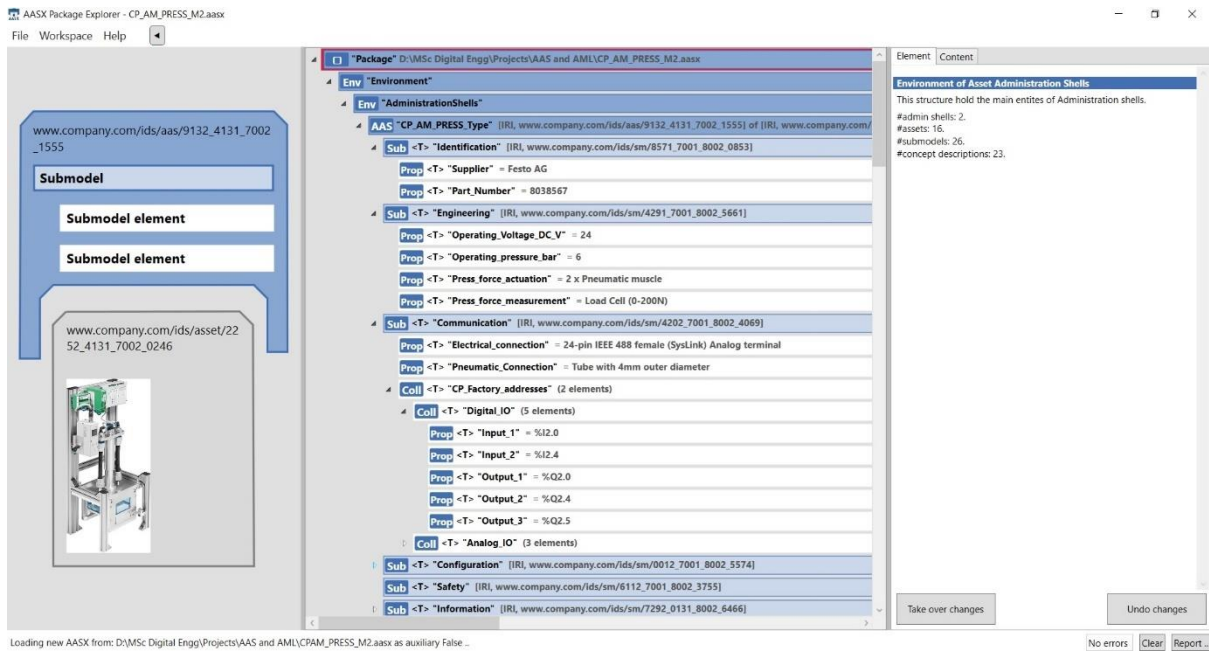


Figure 11: Structure of AAS package explorer

The aasx package explorer is an open source browser and editor to create asset administration shell models. The asset administration shell package explorer allows import of elements of an administration shells from other packages which makes hierarchical modelling simpler. Similarly, submodels can be referenced in the administration shell or exported to JSON. Since it is an open source implementation, the aasx package explorer is continuously updated and new features are being added. Additional information and code for aasx explorer can be found here [8]. The software can be downloaded from the link given in [9].

3.3.1 Package File format for AAS (AASX)

The models are saved in a package with '.aasx' extension. The AASX package explorer supports XML and JSON serialisation of asset administration shells. It also supports export formats like AutomationML (AML), BMEcat, OPC-UA server node set, etc [7]. This inter-usability of submodels and submodel elements in asset administration shell is possible because of the package file format (aasx) following international standards. The package file format for AAS is defined based on the following requirements [7]:

- Generic package file format to include Asset Administration Shell structure, data and other related file.
- Main use cases are the exchange between organizations/ partners and storage/ persistency of 'Asset Administration Shells' information.
- Without any legal restrictions and no royalties. Preferably based on international standard with high guarantees of future maintainability of that format.
- Existence of APIs to create, read and write this format.
- Digital signatures and encryption capabilities must be provided.
- Policies for authenticity and integration of package files.

A package can contain number of administration shells, hence all the files linked to every administration shell along with the administration shell itself is packaged into an aasx zip file. This zip package follows ‘*open packaging convention*’ and hence is independent of how files are stored in the package. It also supports digital signature and encryption. The package as a whole is encrypted for secure storage and transfer of data. Although, it is also possible to encrypt individual files in a package.

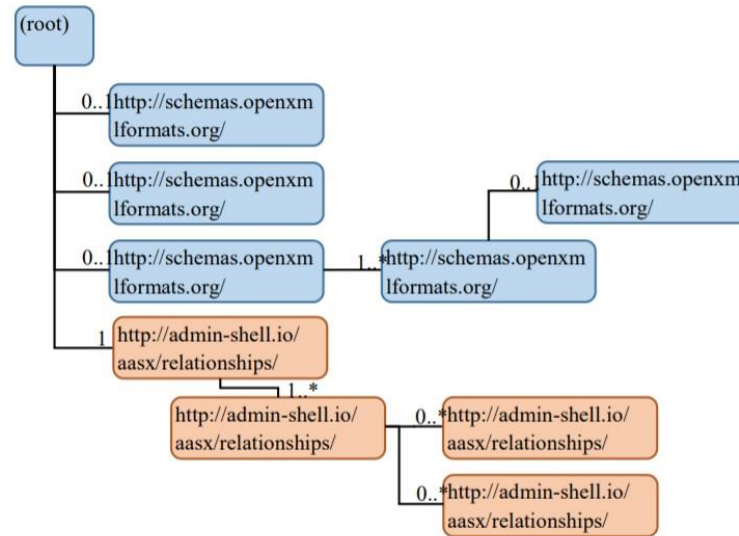


Figure 12: Example of a logical model for the AAX format

The AASX format can be identified by the file extension ‘.aasx’. The AASX package is defined by two models – logical model and physical model. The logical model (Figure 12) is made up of relationship types, their cardinality and the source of relationships. The physical model (Figure 13) defines how the files are stored in the package and how they are addressed in the relationships. The physical package format is a ZIP file that can be opened and edited in any ZIP compatible application.

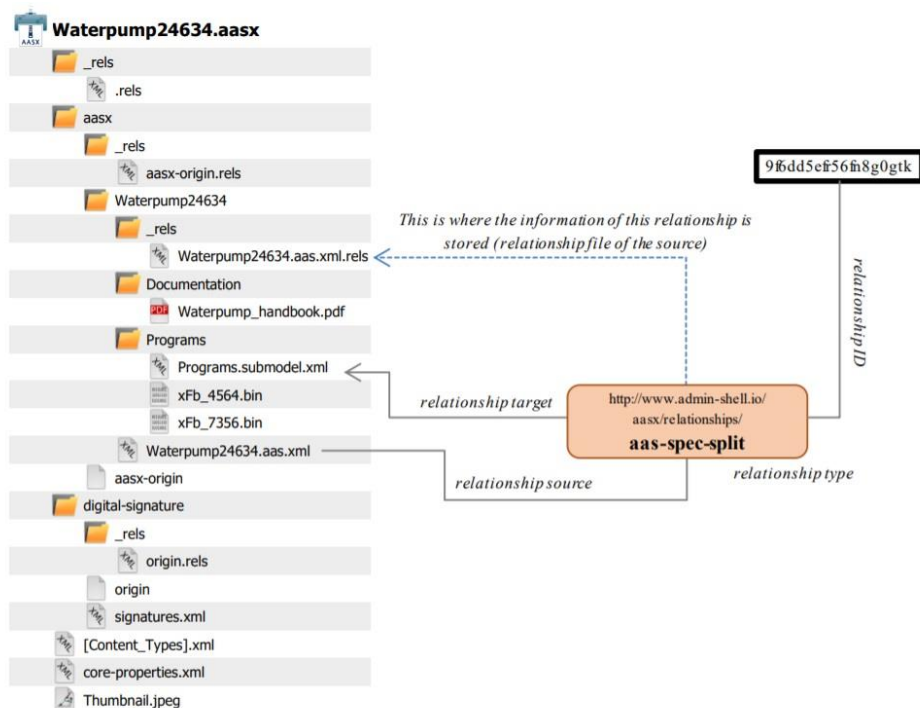


Figure 13: Example of a physical model based on a sample product and its mapping to the logical model

4 Modelling of the system

Out of the entire CP Factory system – small process, an application module & a base module was selected for the purpose of modelling to avoid complexity. For modelling the two modules considered two different approaches were considered:

1. First approach was to provide an Asset Administration Shell (AAS) to each of the components making up the base module as well as the base module as a whole. The complex whole (CP-F-Linear) is considered as the primary asset & the individual components making up the entire CP-F-Linear is considered as secondary assets.
2. Second approach of modelling is applied to the asset CP-AM-MPRESS, wherein a single AAS is provided to the complex whole. The components making up the entire MPRESS are considered as co-managed entities & are included in the same AAS as that of MPRESS.

4.1 Base Module - CP Factory Linear (CP-F-LINEAR)



Figure 14: CP Factory Linear Module

The base model works on the principle of linear material flow and its bidirectional. It has more than one working positions for the application module to attach. The module itself has a mirror design and both halves are attached to a PLC which in turn helps in attaching different application module and connection to MES4.

4.1.1 Module Layout

The module consists of these components: -

- Base module frame
- 2x integrated electronics cabinet
 - PLC
 - Emergency-stop relay
- 2x transport belt
- 2x pneumatic stopper unit, with inductive sensors and RFID read-write-head
- 2x control panel with buttons, touchscreen and emergency-stop button

4.1.2 Module Design



▲	AAS	"CP_F_LINEAR"	[IRI, www.company.com/ids/aas/3463_0122_7002_2657] of [IRI, www.company.com/ids/asset/8173_0122_7002_1391, Template]
▷	Sub	<T> "Identification"	[IRI, www.company.com/ids/sm/3482_5101_8002_4282]
▷	Sub	<T> "Engineering"	[IRI, www.company.com/ids/sm/5492_5101_8002_2691]
▷	Sub	<T> "Configuration"	[IRI, www.company.com/ids/sm/6203_5101_8002_7440]
▷	Sub	<T> "Communication"	[IRI, www.company.com/ids/sm/5503_5101_8002_0764]
	Sub	<T> "Safety"	[IRI, www.company.com/ids/sm/8113_5101_8002_4750]
	Sub	<T> "Lifecycle"	[IRI, www.company.com/ids/sm/7313_5101_8002_6693]
	Sub	<T> "Energy Efficiency"	[IRI, www.company.com/ids/sm/8513_5101_8002_9124]
▷	Sub	<T> "Information"	[IRI, www.company.com/ids/sm/9391_8041_8002_4460]
▷	View	"Business"	(1 elements)
▷	View	"Function"	(3 elements)
▷	View	"Information"	(2 elements)
▷	View	"Communication"	(3 elements)

Figure 15: CP Factory Linear – AAS Model

The basic structure or the environment of the model consists of: -

- 40 – Asset Administration Shells
- 40 – Assets
- 432 – Sub model Elements
- 167 – Concept Descriptions

The submodels in Figure 15 are structured based on industry standards to specify the asset specific characteristics of the asset administration shell. Out of the 8 submodels considered, 7 submodels are standardised according to ZVEI (Zentralverband Elektrotechnik und Elektronikindustrie e.V) structure. The structure of the information these submodels can contain are specified by various IEC, ISO & VDMA standards as mentioned below:

- **Identification** – ISO 29005
- **Engineering** – IEC 61360/ISO 13584 standard data element, IEC 61987 for data structure and elements
- **Configuration** – IEC 61804 EDDL, IEC 62453 FDT
- **Communication** – IEC 61784
- **Safety** – EN ISO 13849, EN/IEC 61508 functional safety discrete, EN/IEC 61511 functional safety process, EN/IEC 62061 safety of machinery
- **Life Cycle** – IEC 62890
- **Energy Efficiency** – ISO/IEC 201405-5

With respect to all the above mentioned submodels, an additional submodel known as **Information** is created to establish relationship between various components mapped to the specific submodel.

Sub <T> "Information" [IRI, www.company.com/ids/sm/9391_8041_8002_4460]	
Coll	"Digital inputs" (9 elements)
Opr	"Emergency stop not acknowledged"
Opr	"Emergency stop button pushed"
Opr	"Stopper cylinder in lower end position"
Opr	"Carrier ident code bit 0 detected"
Opr	"Carrier ident code bit 1 detected"
Opr	"Carrier ident code bit 2 detected"
Opr	"Carrier ident code bit 3 detected"
Opr	"Carrier detected at conveyor entry"
Opr	"Carrier detected at conveyor exit"
Coll	"Digital outputs" (4 elements)
Opr	"Drive belt in forward direction"
Opr	"Drive belt in reverse direction"
Opr	"Select belt speed"
Opr	"Move stopper cylinder down"

Figure 16: CP-F-LINEAR Submodel – Information

4.1.3 Module Relationships

The Digital inputs and outputs of various operations performed as shown in the Figure 16 above of the base module provides information of the operation which takes place with respect to the PLC inputs and outputs. The PLC addresses or the digital inputs and outputs are modelled in the communication submodel which provides the necessary information to establish relationships between submodels in an AAS. The example provided in Figure 17 shows the description of a property of communication submodel and its relationship with an operation in Information submodel is as shown in Figure 18.

Sub <T> "Communication" [IRI, www.company.com/ids/sm/5503_5101_8002_0764]	
Prop	"Electrical connection" = CEE (16A) for connecting to room infrastructure, Preconfigured modular plug for connecting to another CP Factory station
Prop	"Pneumatic connection" = Hose coupling for connecting to room infrastructure, Preconfigured modular plug for connecting to another CP Factory station
Coll	"PLC Addresses" (1 elements)
Coll	"Digital IO" (13 elements)
Prop	"Input 1" = %I0.0
Prop	"Input 2" = %I0.3
Prop	"Input 3" = %I1.0
Prop	"Input 4" = %I1.1
Prop	"Input 5" = %I1.2
Prop	"Input 6" = %I1.3
Prop	"Input 7" = %I1.4
Prop	"Input 8" = %I1.6
Prop	"Input 9" = %I1.7
Prop	"Output 1" = %Q0.4
Prop	"Output 2" = %Q0.5
Prop	"Output 3" = %Q0.6
Prop	"Output 4" = %Q1.0

Figure 17: CP-F-LINEAR Submodel - Communication

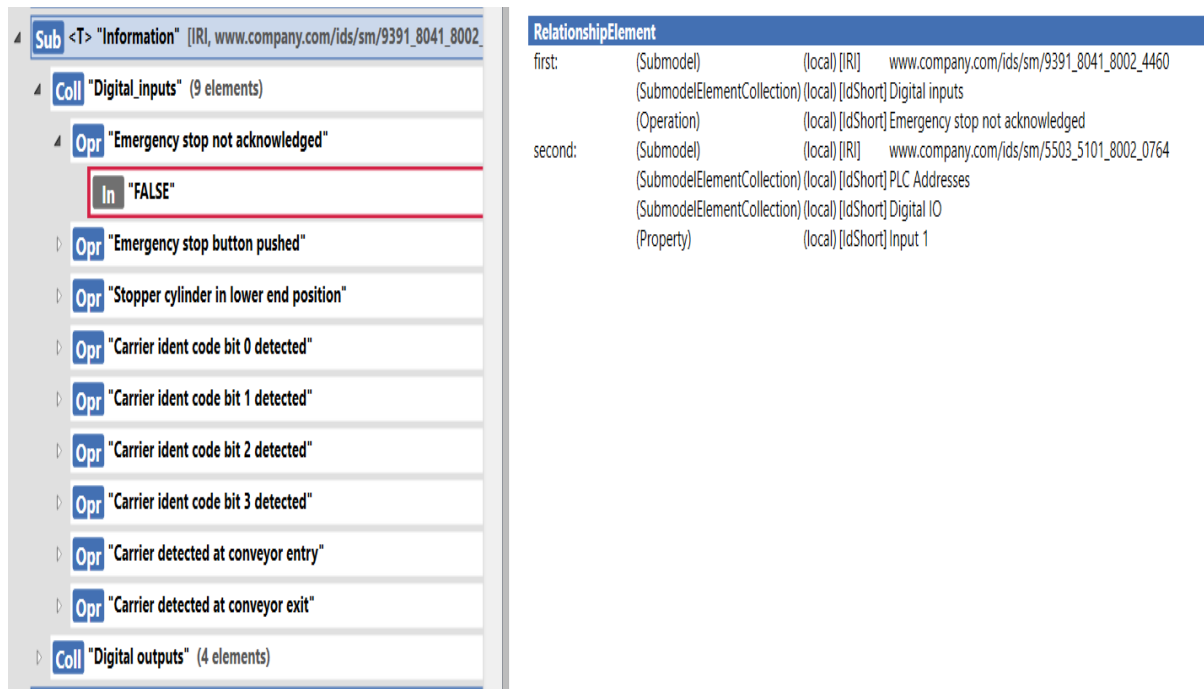


Figure 18: Information submodel with relationship elements from Communication submodel

The whole CP-F-LINEAR and its components are modelled with respect to parent-child relationship. The base module itself is considered to be a parent AAS and all the listed components are child AAS which are derived from the parent AAS and a complex relationship is established. Layers of AAS form a parent AAS provides unique identification and maintains a structured hierarchy.

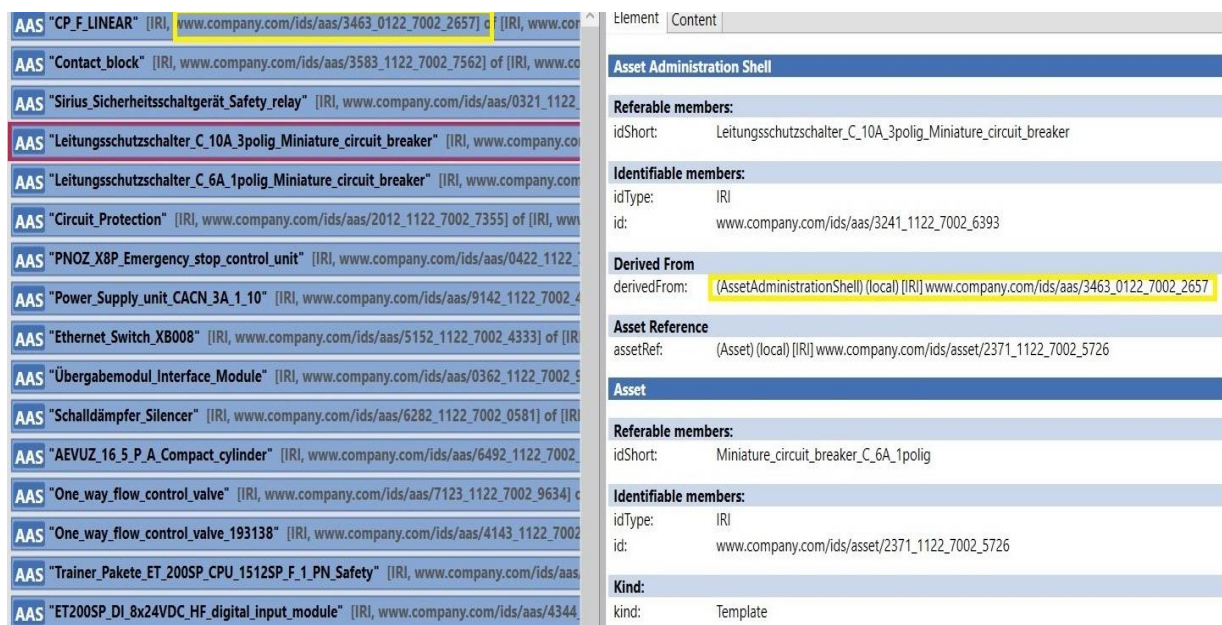


Figure 19: Relationship between parent AAS (CP-F-LINEAR) and child AAS (Miniature circuit breaker)

The above Figure 19 shows the relationship between the parent AAS 'CP-F-LINEAR' and the child AAS 'Leitungsschutzschalter C 10A 3polig Miniature circuit breaker' which is derived from the parent shell. Each and every other component listed are derived from the parent AAS which provides unique

relationship identity to every component listed in the production system. The overall idea of the parent-child relationship of the administration shells is more clearly represented in the UML diagram shown in Figure 20.

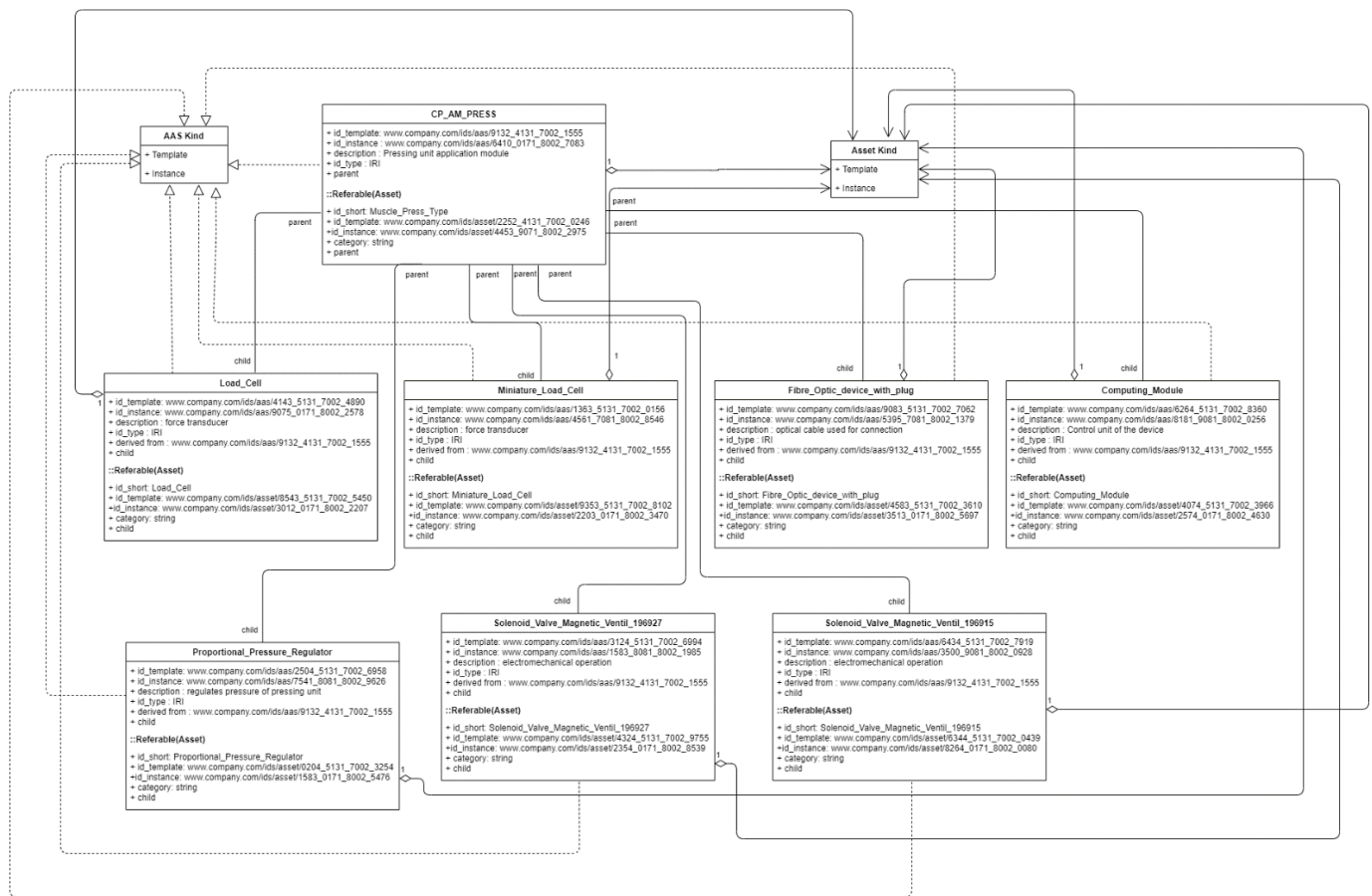


Figure 20: UML diagram representing parent-child relationship between asset administration shells

4.2 Application Module- Muscle Press (CP-AM-MPRESS)



Figure 21: Application module – CP-AM-MPRESS

The application module Muscle press carries out the function to press cubic work pieces. The pressing process is carried out by means of a proportional pressure control. The generated force is measured precisely by means of a load cell.

4.2.1 Module Layout

The module consists of these components:

- Module frame made from aluminium profiles
- Pneumatic muscle press
- Proportional pressure regulator
- Force gauge, analogue
- Signal interface

4.2.2 Modelling of the module

4	AAS "CP_AM_PRESS Type" [IRI, www.company.com/ids/aas/9132_4131_7002_1555] of [IRI, www.company.com/ids/asset/2252_4131_7002_0246, Template]
▷	Sub <T> "Identification" [IRI, www.company.com/ids/sm/8571_7001_8002_0853]
▷	Sub <T> "Engineering" [IRI, www.company.com/ids/sm/4291_7001_8002_5661]
▷	Sub <T> "Communication" [IRI, www.company.com/ids/sm/4202_7001_8002_4069]
▷	Sub <T> "Configuration" [IRI, www.company.com/ids/sm/0012_7001_8002_5574]
	Sub <T> "Safety" [IRI, www.company.com/ids/sm/6112_7001_8002_3755]
▷	Sub <T> "Information" [IRI, www.company.com/ids/sm/7292_0131_8002_6466]
▷	Sub <T> "Measurement" [IRI, www.company.com/ids/sm/3303_1132_8002_0524]
▷	Sub <T> "Control" [IRI, www.company.com/ids/sm/8451_5132_8002_8451]
▷	Sub <T> "Computing" [IRI, www.company.com/ids/sm/9181_8132_8002_4377]
	Sub <T> "Lifecycle" [IRI, www.company.com/ids/sm/8512_7001_8002_8876]
	Sub <T> "Energy_Efficiency" [IRI, www.company.com/ids/sm/1122_7001_8002_9641]
▷	View "Business" (2 elements)
▷	View "Function" (5 elements)
▷	View "Information" (11 elements)
▷	View "Communication" (3 elements)

Figure 22: CP Factory Muscle Press AAS Model

The basic structure of the model consists of:

- 2 – Asset Administration Shells
- 16- Assets (Including the instance assets which is the same as type asset, but considered for modelling requirements)
- 26- Submodels
- 23- Concept descriptions

The Figure 22 shows the structuring of the asset CP-AM-MPRESS made using series of submodels. Each submodel represents particular aspect of the asset concerned. Out of the 11 submodels considered,

7 submodels are standardised according to ZVEI (Zentralverband Elektrotechnik und Elektronikindustrie e.V) structure. The structure of the information these submodels can contain are specified by various IEC, ISO & VDMA standards as mentioned below:

- **Identification:** ISO 29005
- **Communication:** IEC 61784 Fieldbus profiles chapter 2 (Ethernet-real-time capable)
- **Engineering:** IEC 61360/ISO13584 Standard data elements, IEC 61987 Data structures & elements, eCl@ss Database with product classes
- **Configuration:** IEC 61804 EDDL, IEC 62453 FDT
- **Safety:** EN ISO 13849, EN/IEC 61508 Functional safety- discrete, EN/IEC 61511 Functional safety-process, EN/IEC 62061 Safety of machinery
- **Lifecycle:** IEC 62890 Lifecycle
- **Energy Efficiency:** ISO/IEC 20140-5

In addition to the standardised submodels mentioned above, the following extra submodels (Figure 23) are considered to incorporate the modelling of the information related to the individual components making up the asset CP-AM-MPRESS.

▷	Sub <T> "Information" [IRI, www.company.com/ids/sm/7292_0131_8002_6466]
▷	Sub <T> "Measurement" [IRI, www.company.com/ids/sm/3303_1132_8002_0524]
▷	Sub <T> "Control" [IRI, www.company.com/ids/sm/8451_5132_8002_8451]
▷	Sub <T> "Computing" [IRI, www.company.com/ids/sm/9181_8132_8002_4377]

Figure 23: Additional submodels

The submodels Measurement, Control and Computing are smaller subprocesses in the process of pressing. These subprocesses are carried out with the help of separate entities. The entities are defined by certain properties of their own and relationship with the asset by Relationship Element, as explained in section 4.2.4 – Relationships between Submodel Elements.

4.2.3 Submodel contents

- The submodel "Identification" contains basic information of the asset like name of the supplier, part no & series no. Submodel "Engineering" is provided with information related to the technical functionalities of the asset as shown in the Figure 24.

◀	Sub <T> "Engineering" [IRI, www.company.com/ids/sm/4291_7001_8002_5661]
	Prop <T> "Operating Voltage DC (V)" = 24
	Prop <T> "Operating pressure (bar)" = 6
	Prop <T> "Press force actuation" = 2 x Pneumatic muscle
	Prop <T> "Press force measurement" = Load Cell (0-200N)

Figure 24: CP-AM-MPRESS submodel – Engineering

- The submodel "Communication" is provided with information related to pneumatic & electrical connections. In addition, addresses of the various I/O interfaces of the asset are described as shown in the Figure 25.

Sub	<T> "Communication" [IRI, www.company.com/ids/sm/4202_7001_8002_4069]
Prop	<T> "Electrical connection" = 24-pin IEEE 488 female (SysLink) Analog terminal
Prop	<T> "Pneumatic Connection" = Tube with 4mm outer diameter
Coll	<T> "CP Factory addresses" (2 elements)
Coll	<T> "Digital IO" (5 elements)
Prop	<T> "Input 1" = %I2.0
Prop	<T> "Input 2" = %I2.4
Prop	<T> "Output 1" = %Q2.0
Prop	<T> "Output 2" = %Q2.4
Prop	<T> "Output 3" = %Q2.5
Coll	<T> "Analog IO" (3 elements)
Prop	<T> "Input 1" = %IW6
Prop	<T> "Input 2" = %IW8
Prop	<T> "Output 1" = %QW6

Figure 25: CP-AM-MPRESS submodel – Communication

- The submodel "Configuration" contains information on the reference tags of various inputs & outputs which represents location & product aspect in accordance with DIN EN 81346 (Figure 26). Also, relationship element is included which relates the particular reference tag with the respective I/O addresses.

Sub	<T> "Configuration" [IRI, www.company.com/ids/sm/0012_7001_8002_5574]
Coll	<T> "Reference tags" (16 elements)
Prop	<T> "Reference Tag 1" = +HL-BG1
Rel	<T> "Relation 1"
Prop	<T> "Reference Tag 2" = +HL-KF11:D3
Rel	<T> "Relation 2"
Prop	<T> "Reference Tag 3" = +HL-MB1
Rel	<T> "Relation 3"
Prop	<T> "Reference Tag 4" = +HL-KF11:D1
Rel	<T> "Relation 4"
Prop	<T> "Reference Tag 5" = +HL-KF11:D2
Rel	<T> "Relation 5"
Prop	<T> "Reference Tag 6" = +HL-KF10:12
Rel	<T> "Relation 6"
Prop	<T> "Reference Tag 7" = +HL-KF11:X
Rel	<T> "Relation 7"
Prop	<T> "Reference Tag 8" = +HL-KF11:W+
Rel	<T> "Relation 8"

Figure 26: CP-AM-MPRESS submodel – Configuration

- The submodel “Information” specifies various digital & analog I/O to the asset, the respective operation & establishes relation between the respective I/ O interface as shown in Figure 27.

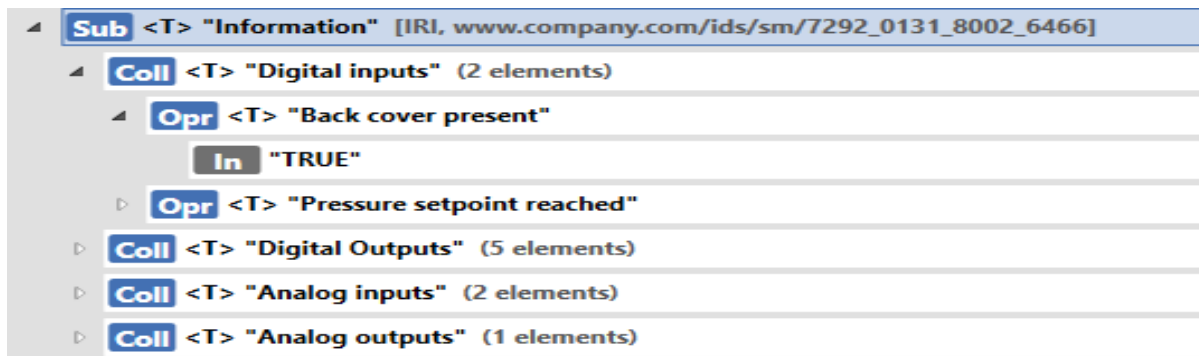


Figure 27: CP-AM-MPRESS submodel – Information

- The submodel “Measurement” is introduced to generalize & list properties, information of the components of MPRESS related to measurement category as shown in Figure 28. Under this submodel, the individual components are represented as a co-managed entities & a relationship element is added to establish the relationship with the respective asset. The entities in turn have further elements within them similar to the submodels to represent various aspects of the entities.

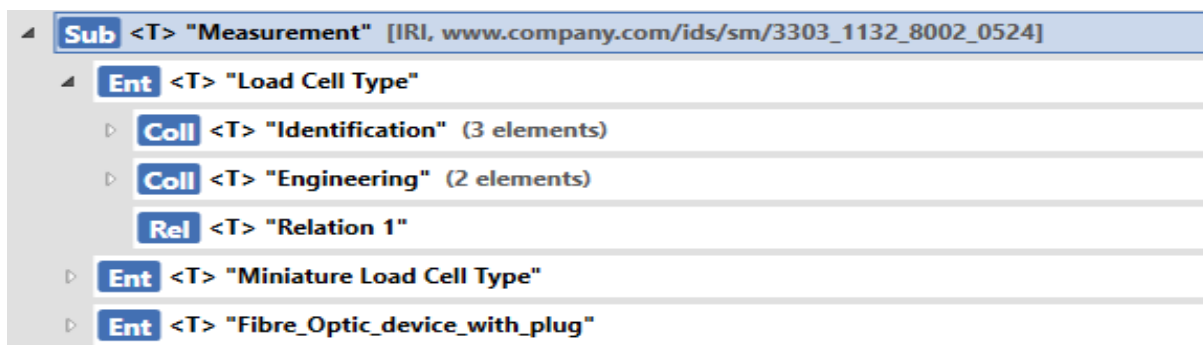


Figure 28: CP-AM-MPRESS submodel – Measurement

Similar to the submodel category “Measurement”, two more submodels namely “Control” & “Computing” are introduced to generalize & list properties, information of the components of MPRESS related to control & computing categories.

4.2.4 Relationships between submodel elements

- The reference tags of the various communication interfaces represented in the submodel “Configuration” are linked to the respective communication interface through the relationship element as shown in Figure 29. In the relationship element, first relationship element is specified as the individual reference tags & the second relationship element is the corresponding address of the communication interface.

Sub <T> "Configuration" [IRI, www.company.com/ids/sm/001]

Coll <T> "Reference tags" (16 elements)

Prop <T> "Reference Tag 1" = +HL-BG1

Rel <T> "Relation 1"

Prop <T> "Reference Tag 2" = +HL-KF11:D3

Rel <T> "Relation 2"

Prop <T> "Reference Tag 3" = +HL-MB1

Rel <T> "Relation 3"

Prop <T> "Reference Tag 4" = +HL-KF11:D1

Rel <T> "Relation 4"

Prop <T> "Reference Tag 5" = +HL-KF11:D2

Rel <T> "Relation 5"

Prop <T> "Reference Tag 6" = +HL-KF10:12

Rel <T> "Relation 6"

Element

Content

Submodel Element

Referable members:

idShort: Relation 1

category: VARIABLE

Kind:

kind: Template

Semantic ID

Qualifier

RelationshipElement

first: (Submodel) (local) [IRI] www.company.com/ids/sm/7292_0131_8002_6466 (SubmodelElementCollection) (local) [IdShort] Reference tags (Property) (local) [IdShort] Reference Tag 1

second: (Submodel) (local) [IRI] www.company.com/ids/sm/4202_7001_8002_4069 (SubmodelElementCollection) (local) [IdShort] CP Factory addresses (SubmodelElementCollection) (local) [IdShort] Digital IO (Property) (local) [IdShort] Input 1

Figure 29: Relationship between reference tag & corresponding communication interface

- The various operations performed by the MPRESS & their corresponding type of signal in the submodel "Information" are also linked to the respective communication interfaces through the relationship element as shown in Figure 30. In the relationship element, first relationship is specified as the operation performed for the given type of signal & the second relationship is the corresponding address of the communication interface.

Element	Content
<div> <div> <div>Sub <T> "Information" [IRI, www.company.com/ids/sm/7292_0131_8002_6466]</div> <div> <div>Coll <T> "Digital inputs" (2 elements)</div> <div> <div>Opr <T> "Back cover present"</div> <div>In "TRUE"</div> <div>Opr <T> "Pressure setpoint reached"</div> </div> <div>Coll <T> "Digital Outputs" (5 elements)</div> <div>Coll <T> "Analog inputs" (2 elements)</div> <div>Coll <T> "Analog outputs" (1 elements)</div> </div> </div> <div> <div>Sub <T> "Measurement" [IRI, www.company.com/ids/sm/3301_0131_8002_4069]</div> <div> <div>Ent <T> "Load Cell Type"</div> <div> <div>Coll <T> "Identification" (3 elements)</div> <div>Coll <T> "Engineering" (2 elements)</div> <div>Rel <T> "Relation 1"</div> </div> <div>Ent <T> "Miniature Load Cell Type"</div> </div> </div> </div>	
<div> <div>OperationVariable</div> <div> <div>OperationVariable value (is a SubmodelElement)</div> <div>Submodel Element</div> <div> <div>Referable members:</div> <div>idShort: TRUE</div> <div>category: VARIABLE</div> <div>description: [EN] Input signal to the operation</div> </div> <div>Kind:</div> <div>Semantic ID</div> <div>Qualifier</div> <div>RelationshipElement</div> <div> <div>first:</div> <div> <div>(Submodel)</div> <div>(local) [IRI]</div> <div>www.company.com/ids/sm/7292_0131_8002_6466</div> </div> <div> <div>(SubmodelElementCollection)</div> <div>(local) [IdShort]</div> <div>Digital inputs</div> </div> <div> <div>(Operation)</div> <div>(local) [IdShort]</div> <div>Back cover present</div> </div> </div> <div> <div>second:</div> <div> <div>(Submodel)</div> <div>(local) [IRI]</div> <div>www.company.com/ids/sm/4202_7001_8002_4069</div> </div> <div> <div>(SubmodelElementCollection)</div> <div>(local) [IdShort]</div> <div>CP Factory addresses</div> </div> <div> <div>(SubmodelElementCollection)</div> <div>(local) [IdShort]</div> <div>Digital IO</div> </div> <div> <div>(Property)</div> <div>(local) [IdShort]</div> <div>Input 1</div> </div> </div> </div> </div>	

Figure 30: Relationship between IO signal & the corresponding communication interface

- The entities defined under the submodels - Measurement, Control & Computing are related to their corresponding assets defined in the assets section through relationship element as shown in Figure 31. The entities considered here are all co-managed entities since they don't have a separate Administration Shell but are included in the same Shell as that of MPRESS. In the relationship element, first relationship is specified as the parent asset (CP-AM-MPRESS) & the second relationship is the entity itself.

<div> <div>Sub <T> "Measurement" [IRI, www.company.com/ids/sm/33]</div> <div> <div>Ent <T> "Load Cell Type"</div> <div> <div>Coll <T> "Identification" (3 elements)</div> <div>Coll <T> "Engineering" (2 elements)</div> <div>Rel <T> "Relation 1"</div> <div>Ent <T> "Miniature Load Cell Type"</div> <div>Ent <T> "Fibre_Optic_device_with_plug"</div> </div> </div> <div> <div>Sub <T> "Control" [IRI, www.company.com/ids/sm/8451_51]</div> <div> <div>Ent <T> "Proportional_Pressure_Regulator Type"</div> <div>Coll <T> "Identification" (3 elements)</div> </div> </div> </div>	<div> <div>Submodel Element</div> <div> <div>Referable members:</div> <div> <div>idShort: Relation 1</div> <div>category: CONSTANT</div> </div> <div>Kind:</div> <div>kind: Template</div> <div>Semantic ID</div> <div>Qualifier</div> <div>RelationshipElement</div> <div> <div>first: (Asset) (local) [IRI] www.company.com/ids/asset/2252_4131_7002_0246</div> <div>second: (Submodel) (local) [IRI] www.company.com/ids/sm/3303_1132_8002_0524</div> <div>(Entity) (local) [IdShort] Load Cell Type</div> </div> </div> </div>
--	--

Figure 31: Relationship between asset & entity

For the fact that several interested parties (stakeholders) can exist & new stakeholders can arise during the lifecycle, modelling of both the modules is done with this factor in consideration. Hence, in the Assets section an instance asset is added in addition to the type asset for modelling means as shown in Figure 32.

Env "Assets"
Asset "Muscle_Press_Type" [IRI, www.company.com/ids/asset/2252_4131_7002_0246]
Asset "Muscle_Press Instance" V. [IRI, www.company.com/ids/asset/4453_9071_8002_2975]
Asset "Load_Cell Type" [IRI, www.company.com/ids/asset/8543_5131_7002_5450]
Asset "Load_Cell Instance" [IRI, www.company.com/ids/asset/3012_0171_8002_2207]
Asset "Miniature_Load_Cell Type" [IRI, www.company.com/ids/asset/9353_5131_7002_8102]
Asset "Miniature_Load_Cell Instance" [IRI, www.company.com/ids/asset/2203_0171_8002_3470]
Asset "Fibre_Optic_device_with_plug Type" [IRI, www.company.com/ids/asset/4583_5131_7002_3610]
Asset "Fibre_Optic_device_with_plug Instance" [IRI, www.company.com/ids/asset/3513_0171_8002_5697]
Asset "Proportional_Pressure_Regulator Type" [IRI, www.company.com/ids/asset/0204_5131_7002_3254]
Asset "Proportional_Pressure_Regulator Instance" [IRI, www.company.com/ids/asset/1583_0171_8002_5476]
Asset "Solenoid_Valve_Magnetic_Ventil_196927 Type" [IRI, www.company.com/ids/asset/4324_5131_7002_9755]
Asset "Solenoid_Valve_Magnetic_Ventil_196927 Instance" [IRI, www.company.com/ids/asset/2354_0171_8002_8539]
Asset "Solenoid_Valve_Magnetic_Ventil_196915 Type" [IRI, www.company.com/ids/asset/6344_5131_7002_0439]
Asset "Solenoid_Valve_Magnetic_Ventil_196915 Instance" [IRI, www.company.com/ids/asset/8264_0171_8002_0080]
Asset "Computing_Module Type" [IRI, www.company.com/ids/asset/4074_5131_7002_3966]
Asset "Computing_Module Instance" [IRI, www.company.com/ids/asset/2574_0171_8002_4630]

Figure 32: Asset Types & Instances

The assets of kind type will be referenced to Administration Shell of kind type, similarly assets of kind instance will only be linked to the Administration Shell of kind instance. In this way, several Administration Shells of kind instance can be created based on the stakeholders present & these instance assets can be referenced to them as shown in Figure 33.

Figure 33: Referencing of instance assets to AAS

This Asset Identification Model Reference in the assets of kind instances can contain several references to the Asset Administration Shells, which in turn display the corresponding list of stakeholders related to that particular asset.

4.2.5 Module views and Instances

Both the base module (CP-F-LINEAR) & application module (CP-AM-MPRESS) are designed with respect to manufacturer as well as stake holders' point of view. The manufacturers point of view is designed with kind – template and the stake holders' point of view is designed with respect to kind – instance.

The kind – template provides all the necessary component information which is needed for the manufacturer and kind – instance is modelled according to restricted information which needs to be provided for a customer/stake – holder. Different AAS are modelled as instances to show relevant information for the customer which differs from the template class. Each and every shell (AAS) is referred to an asset which defines the particular component which is modelled for both the template and the instance classes.

The UML diagram in Figure 36 shows a representation of the relationship between submodel elements. It also shows an overview of relations between some properties and operation elements.

AAS	"Contact_block_Instance"	[IRI, www.company.com/ids/aas/8372_3132_8002_8861] of [IRI, www.company.com/ids/asset/3451_0132_8002_8558, Instance]
Sub	"Identification"	[IRI, www.company.com/ids/sm/8372_3132_8002_8861]
Sub	"Engineering"	[IRI, www.company.com/ids/sm/1491_4132_8002_3512]
Sub	"Configuration"	[IRI, www.company.com/ids/sm/3102_4132_8002_1069]
Sub	"Communication"	[IRI, www.company.com/ids/sm/6402_4132_8002_2660]
Sub	"Safety"	[IRI, www.company.com/ids/sm/8112_4132_8002_8898]
Sub	"Lifecycle"	[IRI, www.company.com/ids/sm/8412_4132_8002_2339]
Sub	"Energy_Efficiency"	[IRI, www.company.com/ids/sm/1222_4132_8002_8120]
View	"Business"	(3 elements)
View	"Function"	(5 elements)
	→	www.company.com/ids/sm/9584_5122_7002_8570/ Connection_Type
	→	www.company.com/ids/sm/9394_5122_7002_7192/ Operating_Frequency
	→	www.company.com/ids/sm/9394_5122_7002_7192/ Operating_Torque
	→	www.company.com/ids/sm/9394_5122_7002_7192/ Rated_operating_Voltage
	→	www.company.com/ids/sm/9394_5122_7002_7192/ Rated_Operating_Current

Figure 34: Instance class of a component (Contact block)

AAS	"Contact_block"	[IRI, www.company.com/ids/aas/3583_1122_7002_7562] of [IRI, www.company.com/ids/asset/7393_1122_7002_8755, Template]
Sub	<T> "Identification"	[IRI, www.company.com/ids/sm/3284_5122_7002_9944]
Sub	<T> "Engineering"	[IRI, www.company.com/ids/sm/9584_5122_7002_8570]
Sub	<T> "Configuration"	[IRI, www.company.com/ids/sm/9394_5122_7002_7192]
Sub	<T> "Communication"	[IRI, www.company.com/ids/sm/5194_5122_7002_2235]
Sub	<T> "Safety"	[IRI, www.company.com/ids/sm/4594_5122_7002_1604]
Sub	<T> "Life_Cycle"	[IRI, www.company.com/ids/sm/3105_5122_7002_5006]
Sub	<T> "Energy_Efficiency"	[IRI, www.company.com/ids/sm/6405_5122_7002_8437]
View	"Business"	(3 elements)
View	"Function"	(8 elements)
	→	www.company.com/ids/sm/9584_5122_7002_8570/ Connection_Type
	→	www.company.com/ids/sm/9584_5122_7002_8570/ Connection_position_type
	→	www.company.com/ids/sm/9394_5122_7002_7192/ Operating_Frequency
	→	www.company.com/ids/sm/9394_5122_7002_7192/ Actuating_Force
	→	www.company.com/ids/sm/9394_5122_7002_7192/ Operating_Torque
	→	www.company.com/ids/sm/9394_5122_7002_7192/ Rated_operating_Voltage
	→	www.company.com/ids/sm/9394_5122_7002_7192/ Max_rated_operating_voltage
	→	www.company.com/ids/sm/9394_5122_7002_7192/ Rated_Operating_Current

Figure 35: Template class of a component (Contact Block)

The above shown component in Figure 34 and Figure 35 shows the difference in level of information for both manufacturer and stake-holders, which are modelled with respect to views of the AAS.

The views added in this model separates information with respect to the following: -

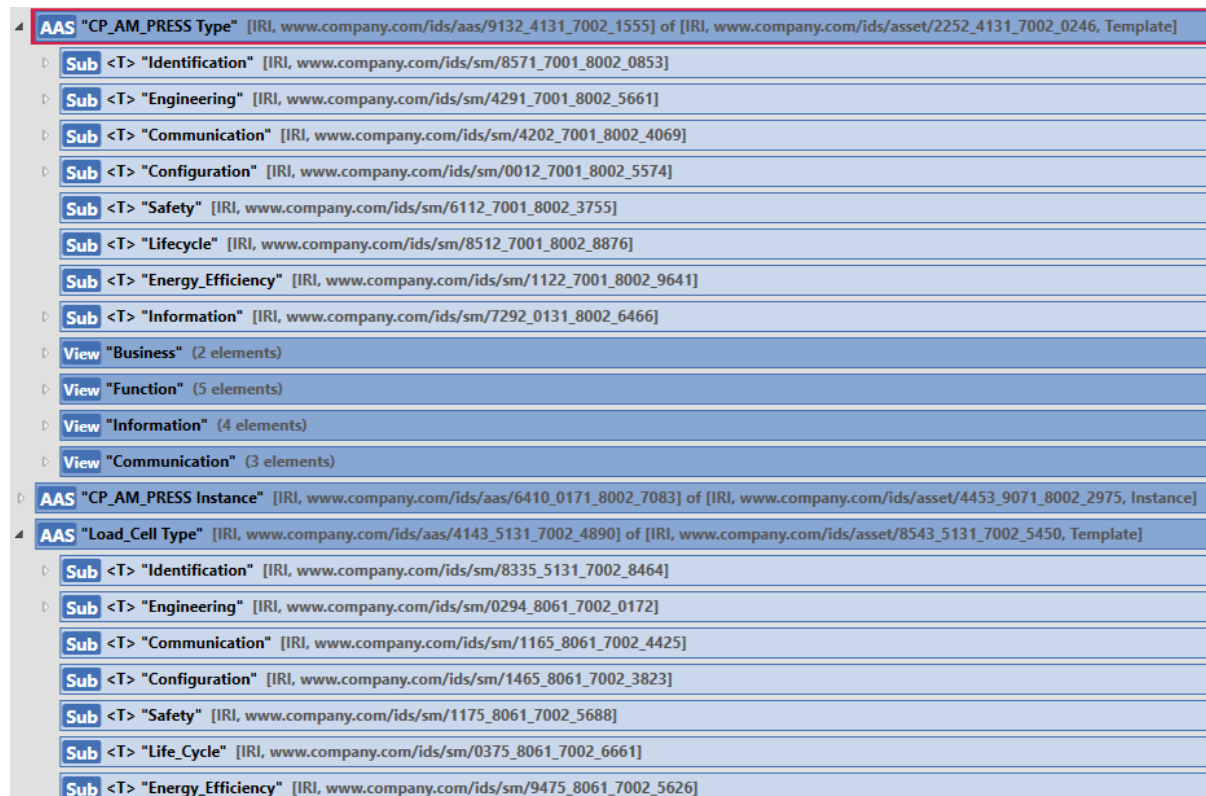
- Business – provides information on supplier, supplier id and model reference number.
- Function – provides information on electrical, pneumatic and mechanical functions of the component modelled.
- Information – defines relationship of different components modelled as AAS.
- Communication – information on connection to other components and inputs and outputs

The information provided inside views varies with respect to a template and instance classes.

5 Comparison & Evaluation between two modes of modelling

For the comparison & evaluation, the application module CP-AM-MPRESS is modelled in the previously described two methods.

5.1 Modelling of CP-AM-MPRESS Method 1



▲	AAS "CP_AM_PRESS Type" [IRI, www.company.com/ids/aas/9132_4131_7002_1555] of [IRI, www.company.com/ids/asset/2252_4131_7002_0246, Template]
▷	Sub <T> "Identification" [IRI, www.company.com/ids/sm/8571_7001_8002_0853]
▷	Sub <T> "Engineering" [IRI, www.company.com/ids/sm/4291_7001_8002_5661]
▷	Sub <T> "Communication" [IRI, www.company.com/ids/sm/4202_7001_8002_4069]
▷	Sub <T> "Configuration" [IRI, www.company.com/ids/sm/0012_7001_8002_5574]
	Sub <T> "Safety" [IRI, www.company.com/ids/sm/6112_7001_8002_3755]
	Sub <T> "Lifecycle" [IRI, www.company.com/ids/sm/8512_7001_8002_8876]
	Sub <T> "Energy_Efficiency" [IRI, www.company.com/ids/sm/1122_7001_8002_9641]
▷	Sub <T> "Information" [IRI, www.company.com/ids/sm/7292_0131_8002_6466]
▷	View "Business" (2 elements)
▷	View "Function" (5 elements)
▷	View "Information" (4 elements)
▷	View "Communication" (3 elements)
▷	AAS "CP_AM_PRESS Instance" [IRI, www.company.com/ids/aas/6410_0171_8002_7083] of [IRI, www.company.com/ids/asset/4453_9071_8002_2975, Instance]
▲	AAS "Load_Cell Type" [IRI, www.company.com/ids/aas/4143_5131_7002_4890] of [IRI, www.company.com/ids/asset/8543_5131_7002_5450, Template]
▷	Sub <T> "Identification" [IRI, www.company.com/ids/sm/8335_5131_7002_8464]
▷	Sub <T> "Engineering" [IRI, www.company.com/ids/sm/0294_8061_7002_0172]
	Sub <T> "Communication" [IRI, www.company.com/ids/sm/1165_8061_7002_4425]
	Sub <T> "Configuration" [IRI, www.company.com/ids/sm/1465_8061_7002_3823]
	Sub <T> "Safety" [IRI, www.company.com/ids/sm/1175_8061_7002_5688]
	Sub <T> "Life_Cycle" [IRI, www.company.com/ids/sm/0375_8061_7002_6661]
	Sub <T> "Energy_Efficiency" [IRI, www.company.com/ids/sm/9475_8061_7002_5626]

Figure 37: Structuring of parent and child asset

- In the first method of modelling the complex whole MPRESS is considered as the parent asset having its own Asset Administration Shell & having the overall information of the unit presented with the help of series of submodels which describes information of different domains concerned with the asset. The individual components making up the whole asset are also provided with individual Administration Shells & all the information related to the components are presented with the series of submodels similar to the parent asset as shown in Figure 37.
- The components making up the parent asset are the child assets. These child assets are referenced to their corresponding parent asset in their Administration Shell through the "Derived From" function as shown in Figure 38.

AAS "Load_Cell Type" [IRI, www.company.com/ids/aas/4143_5131_7002_4890] of [I]	
Sub	<T> "Identification" [IRI, www.company.com/ids/sm/8335_5131_7002_846]
Sub	<T> "Engineering" [IRI, www.company.com/ids/sm/0294_8061_7002_0172]
Sub	<T> "Communication" [IRI, www.company.com/ids/sm/1165_8061_7002_4]
Sub	<T> "Configuration" [IRI, www.company.com/ids/sm/1465_8061_7002_382]
Sub	<T> "Safety" [IRI, www.company.com/ids/sm/1175_8061_7002_5688]
Sub	<T> "Life_Cycle" [IRI, www.company.com/ids/sm/0375_8061_7002_6661]
Sub	<T> "Energy_Efficiency" [IRI, www.company.com/ids/sm/9475_8061_7002_]

Asset Administration Shell	
Referable members:	
idShort:	Load_Cell Type
Identifiable members:	
idType:	IRI
id:	www.company.com/ids/aas/4143_5131_7002_4890
Derived From	
derivedFrom:	(AssetAdministrationShell) (local) [IRI] www.company.com/ids/aas/9132_4131_7002_1555 (AssetAdministrationShell) (local) [IRI] www.company.com/ids/aas/9132_4131_7002_1555

Figure 38: Derived from feature to establish parent-child relationship

- This modelling approach takes into account the complex whole asset as well as its components as primary assets allocating individual Asset Administration Shells to each one of them. The parent-child relationship is established using "Derived from" feature.
- This modelling approach can be considered if complex relationships & information of every single component of the asset has to be presented. Also, it can be considered if communication between individual components as well as the complex whole asset has to be governed by certain qualifying factors which can be specified in individual Asset Administration Shells.
- Individual components can communicate directly with other components as individual AAS has been provided for every component & access to information is governed by that particular AAS only.

Drawbacks:

- Modelling the complete system with separate AAS for every individual part would be a tedious task
- Storage requirements for such a model would be enormous.
- Complex relationships will be needed to properly relate the parent & child elements.

5.2 Modelling of CP-AM-MPRESS Method 2

- In the second method of modelling only the complex whole asset MPRESS is provided with an Asset Administration Shell. The components making up the whole MPRESS are considered as co-managed entities & are listed within the same Administration Shell as shown in the Figure 39.

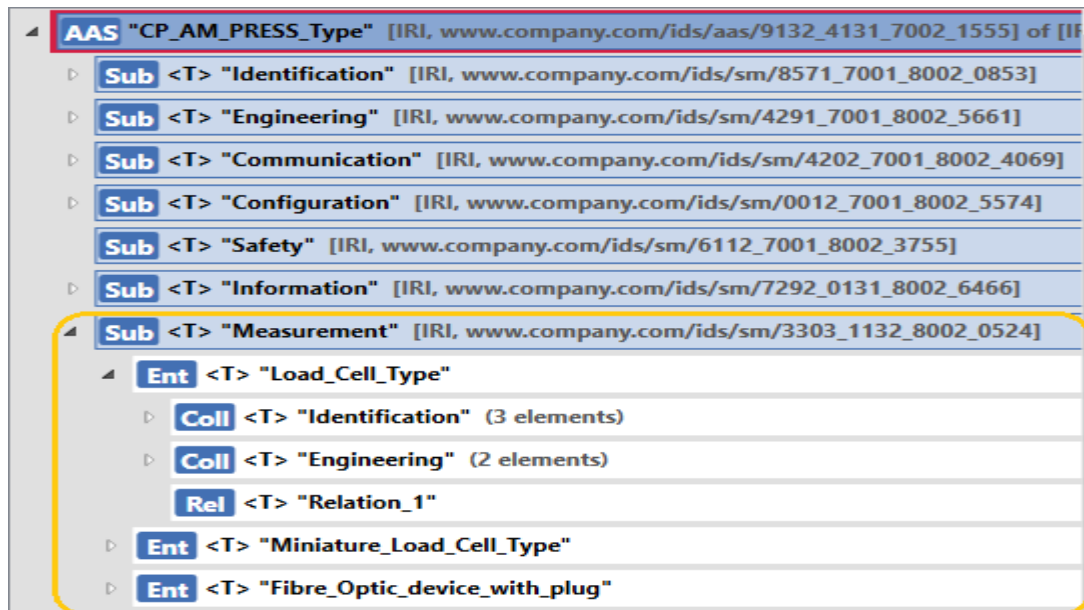


Figure 39: Structuring of assets as co-managed entities

- Extra submodels are introduced other than the standard ones, to list down components making up the whole asset & describe their associated data in a structured way as shown in Figure 40.

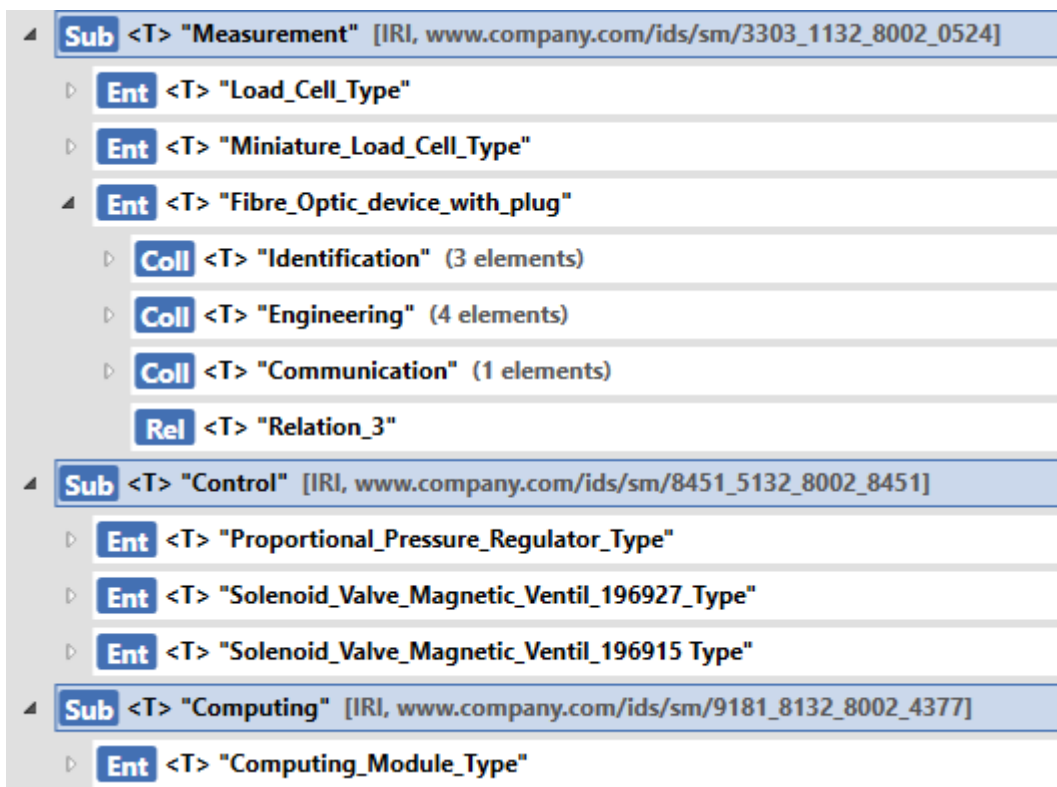


Figure 40: Additional defined submodels for listing down entities & their associated data

- The components making up the whole asset (CP-AM-MPRESS) are listed as co-managed entities & are governed by the same AAS as that of its parent (CP-AM-MPRESS). These entities in turn are related to the corresponding assets through the relationship element.

- Components belonging to a particular parent asset are easily identified as they are listed as entities within the same AAS as that of parent asset.
- This modelling approach can be considered if every individual component need not be governed separately through individual AAS but can be considered as co-managed entities having the same AAS as that of parent (CP-AM-MPRESS).
- This approach is better suited if detailed description of each & every component of the parent asset is not needed to be described. This approach specifically focuses on the details pertaining to the overall parent asset & the components making up the parent are considered as secondary.

Drawbacks:

- The complexity of modelling increases if more detailed information of child assets has to be presented.
- Access to information by the communicating component from the AAS would be complex as there will be complex relationships established within individual AAS for individual parts of the whole asset & the cycle would be relatively slow as the individual components cannot directly interact with other components but are governed by the AAS of the parent.

6 Introduction to AutomationML

Automation Mark-up Language (AML) is an XML based data format which is extensively used & supports data exchange between various engineering tools. Production system engineering is a complex process wherein several engineers of various disciplines execute engineering activities using several engineering tools creating a whole lot of engineering artefacts required to build, run & maintain the production system [10]. The main goal of AML is interlinking engineering tools of different disciplines & supporting data exchange between these tools with minimal or no data loss. AML considers Object Oriented Modelling approach to model & store any engineering or other related information of the object. This object can be any kind of asset to the company from being a physical component to any kind of documentation related to the company. In other words, AML combines various industry related data formats used for storing & exchanging different engineering related data between different disciplines. The core of AML is the top-level data format CAEX that interlinks different industry related data formats. Therefore, AML has a built-in distributed document architecture. The figure below illustrates the basic architecture of AML.

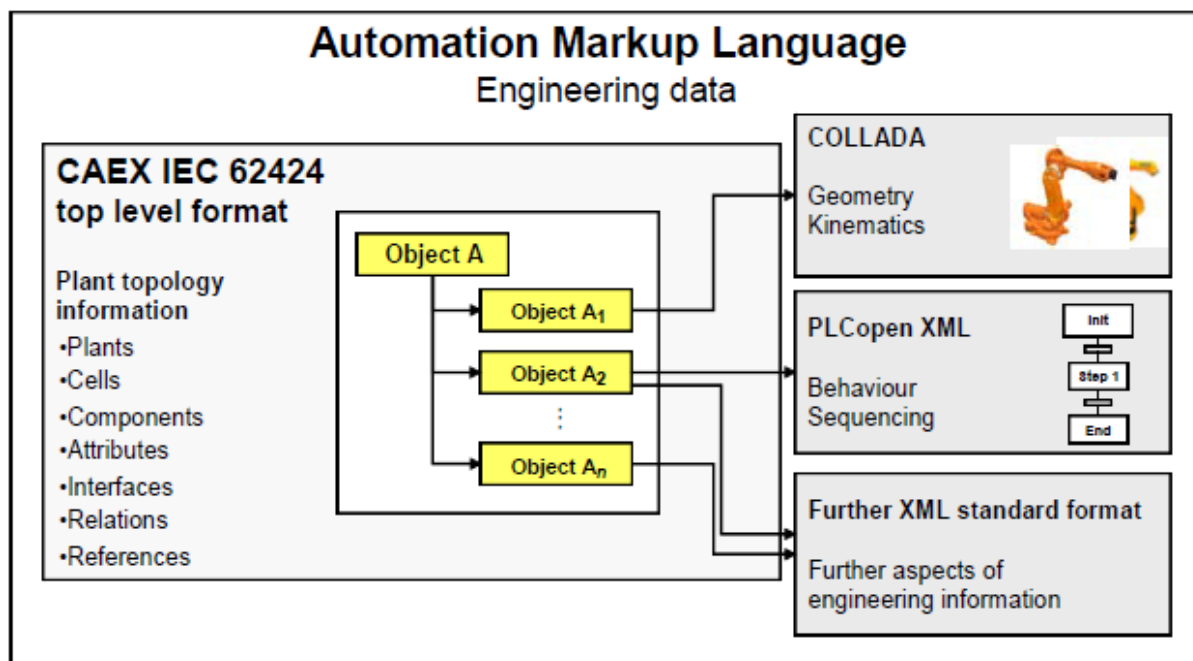


Figure 41: Basic architecture of AML

The modelling approach in AML editor consists of four different aspects for the ease of modelling the necessary information namely:

- Instance Hierarchy: They serve the purpose to store individual information including properties, interfaces, relations & references
- System Unit Class Library: Generic classes will be created under System Unit Class Library which can be used as template for modelling means.
- Role Class Library: Various roles taken by the objects are defined under role class library
- Interface Class Library: They serve the purpose to specify & define interfaces for communication between objects.

To facilitate exchange of engineering data of different formats, AutomationML has to be able to represent different relevant engineering information as described in the figure below.

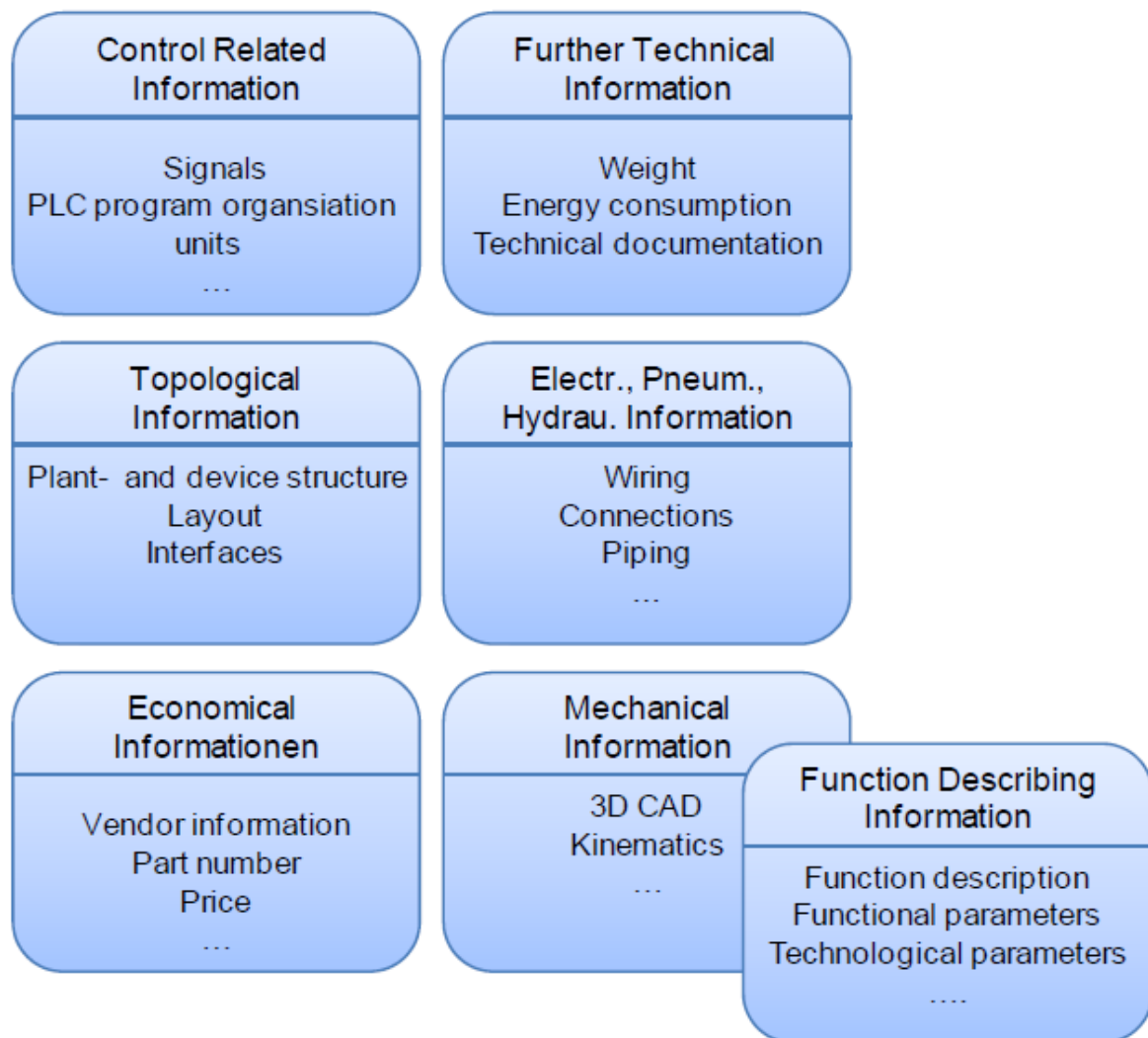


Figure 42: Required information sets to be presented in AML

Industry 4.0 requires increased flexibility related to interlinking engineering activities & tools used in different life cycle phases of the production system. This flexibility also means that the different layers of the production system from the control layer to the company networks have to be integrated [10]. To facilitate these requirements a common data format known as AutomationML was introduced which serves the purpose of integration of engineering activities & tools along engineering chains of production systems & additionally serves to apply the specific engineering data within the use phase of a production system [10].

7 Interlinking AASX and AML

The master project scope is to model the technical & other related information of the selected modules from CP Factory using the AASX Package Explorer which also considers the object-oriented approach for modelling the information. The model generated will have the extension aasx. The main goal is to interlink this aasx file to aml format & check the possible deviations in the hierarchy in AML from the one modelled already in AASX Package Explorer.

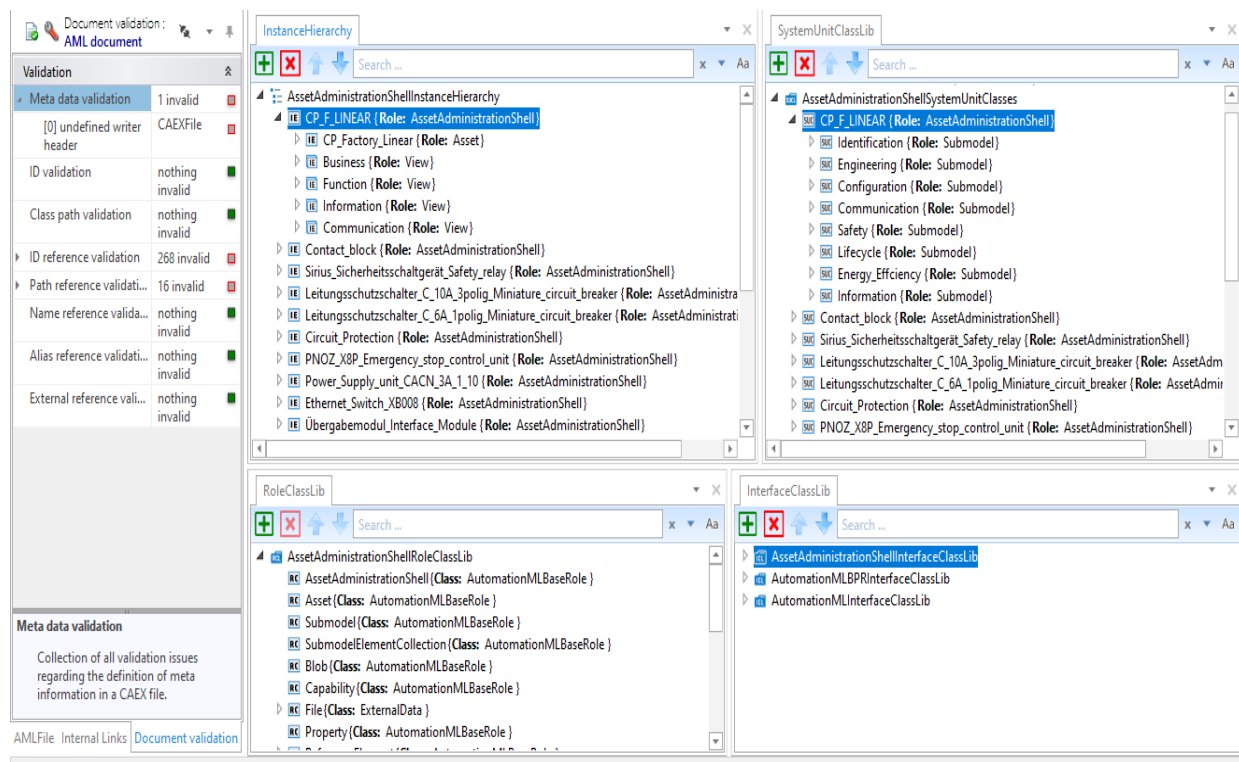


Figure 43: Screenshot of the exported aasx file opened in AML

The above figure is the aasx file exported to aml and opened in an AML editor. The modelled hierarchy seems not to deviate much from the original structure of hierarchy (more details in [5]) as modelled in AASX Package Explorer. Some finer detailing has to be however carried out in the model. The Asset Administration Shells are modelled as System Unit Classes or Internal elements in Instance Hierarchy depending whether the AAS is of kind Type or Instance. All submodel elements are mapped to internal elements of Instance hierarchy. Upon validating the file in AML several validation issues are detected which are related to ID referencing & path referencing. These validation issues arise because the semantics as defined by the Working Group 'Reference Architectures, Standards & Norms' of Platform Industrie 4.0 for Asset Administration Shells doesn't necessarily suit up with that of Automation ML.

8 Conclusion

We have seen two modelling strategies in this document. The first strategy was based on the parent-child relationship, where each component or a device was modelled in a sperate AAS as child AAS and later link to larger main AAS as parent AAS. While, the other strategy was to model the whole production system component in one single AAS. The second method can be used at a micro level, for example, in a factory where a single system component such as a small Pressing machine can be defined in one AAS. Each device used in the system component and its relationship with other devices in the system component will be modelled in the same AAS. A number of such administration shells then can be interlinked & can communicate with each other following the industry standards, thus modelling a whole factory system. This is where, the first method can be helpful, as there will already be a number of administration shells available corresponding to various modules/ sections of the production system.

9 References

- [1] M. Moore, “What is Industry 4.0? Everything you need to know,” TechRadar, 5 November 2019. [Online]. Available: <https://www.techradar.com/news/what-is-industry-40-everything-you-need-to-know>.
- [2] “Festo Didactic InfoPortal,” Festo Didactic GmbH, [Online]. Available: <https://ip.festo-didactic.com/InfoPortal/EN/index.html>.
- [3] “CP Factory - The Cyber-Physical Factory,” Festo Didactic GmbH, [Online]. Available: <https://www.festo-didactic.com/de-de/lernsysteme/lernsysteme-fuer-industrie-4.0/cp-factory/cp-factory-die-cyber-physical-factory.htm?fbid=ZGUuZGUuNTQ0LjEzLjE4LjE3ODUuNzY0Mw>.
- [4] “Download link for CIROS simulation software,” Festo Didactic GmbH, [Online]. Available: <https://ip.festo-didactic.com/InfoPortal/CIROS/EN/Download.php>.
- [5] “The Asset Administration Shell: Implementing digital twins for use in Industry 4.0,” 2018. [Online]. Available: <https://www.plattform-i40.de/PI40/Redaktion/EN/Downloads/Publikation/VWSiD%20V2.0.html>.
- [6] ZVEI: Die Elektroindustrie, “Examples of Asset Administration Shell for Industry 4.0 components - Basic part,” April 2017. [Online]. Available: https://www.zvei.org/fileadmin/user_upload/Presse_und_Medien/Publikationen/2017/April/Asset_Administration_Shell/ZVEI_WP_Verwaltungschale_Englisch_Download_03.04.17.pdf.
- [7] ZVEI: Die Elektroindustrie, “Details of Asset Administration Shell: The exchange of information between partners in the value chain of Industry 4.0,” November 2019. [Online]. Available: <https://www.zvei.org/en/press-media/publications/details-of-the-asset-administration-shell/>.
- [8] Admin shell, “Development link for AASX Package Explorer,” [Online]. Available: <https://github.com/admin-shell-io/aasx-package-explorer>.
- [9] Admin shell, “Download link for AASX Package Explorer,” [Online]. Available: <https://github.com/admin-shell/aasx-package-explorer>.
- [10] N. Schmidt and A. Lüder, “AutomationML in a Nutshell,” 2015.