Result of Research Project

SPECIFICATION

IEC 61131 Software Package Manager

01 June 2023

Submodel Template of the

Asset Adminstration Shell

Imprint

Publisher

Steinbeis Innovation gGmbH
Adornostr. 8
70599 Stuttgart
Germany

Source for Specification Document

Plattform Industrie 4.0
Bertolt-Brecht-Platz 3
10117 Berlin
Germany

Authors

BCON² GmbH

Version history

2023-06-01	Draft 1.0	Release of the Submodel template

Content

List of Figures

List of Tables

Foreword

This AAS Submodel Template was created in an InterOpera Project called “IEC 61131 Software Package Manager”.

With the information technology approach, the AAS offers the opportunity to significantly improve the previously fragmented software world via a manufacturer-independent open interface standard for IEC 61131 software packages.

The goal is a uniform description of such software packages by means of an AAS Submodel.

The Submodel contains all the information required for automated software package management and associated automated dependency and version management. In addition, the Submodel contains information about different package sources in order not to create a single-source vendor log-in for the user via package management.

Many thanks go to Hans Egermeier of talsen team GmbH as use case supplier.

In addition, a big thank you goes to the participants in the working group for their valuable contributions:

- Justus Glöckler, IT & Engineering Solutions
- Daniel Haack, DIN e.V.
- Tobias Klausmann, Lenze SE
- Henning Mersch, Beckhoff Automation GmbH & Co. KG
- Michael Rudschuck, VDE DKE
- Christoph Legat, HEKUMA GmbH

The BCON² GmbH would like to thank the Steinbeis Institute and especially Carina Gliese as project manager for the support in this project within the InterOpera project.

1 General

1.1 About this document

This document is a part of a specification series. Each part specifies the contents of a Submodel template for the Asset Administration Shell (AAS). The AAS is described in [1-3] and [6]. First exemplary Submodel contents were described in [4], while the actual format of this document was derived by the "Administration Shell in Practice" [5]. The format aims to be very concise, giving only minimal necessary information for applying a Submodel template, while leaving deeper descriptions and specification of concepts, structures and mapping to the respective documents [1-6].

The target group of the specification are developers and users of IEC 61131 Software Packages, which are describing and understanding assets in smart manufacturing by means of the Asset Administration Shell (AAS) and therefore need to create a Submodel instance with a hierarchy of SubmodelElements. This document especially details on the question, which SubmodelElements with which semantic identification shall be used for this purpose.

1.2 Scope of the Submodel

This Submodel template aims at interoperable provision of information describing an IEC 61131 Software Package in regard to the asset of the respective Asset Administration Shell. Central element is the provision of properties [7], ideally interoperable by the means of dictionaries such as ECLASS and IEC CDD (Common Data Dictionary). The purpose of this document is to make selected specifications of Submodels in such manner that information about assets can be exchanged in a meaningful way between partners in a value creation network. It targets software components for automation systems by defining standardized meta data.

The intended use-case is the provision of a standardized property structure for IEC 61131 Software Package management, which enables a common description of an IEC 61131 Software Package incl. retrieving of necessary data in relation with the package and supports version as well as update management.

This concept can serve as a basis for standardizing the respective Submodel. The conception is based on existing norms, studies of common practices at enterprises, directives and standards so that a far-reaching acceptance can be achieved.

Beside standardized Submodel this template also introduces standardized SubmodelElementCollections (SMC) in order to improve the interoperability while modelling aspects of software metadata and dependency management within other Submodels.

1.3 Not in Scope

In addition to the objectives described above, the motivation was also to resolve the different dialects of IEC 61131-3. However, the project had no examples of these different dialects. In addition, the basis for a package management by means of AAS had to be created first. Therefore, automatic conversions are not in scope. However, initial features were taken from ECLASS to describe IEC 31360-3 features.

Furthermore, this Submodel Template is designed to be extensible by future versions of this, or similar Submodel Templates.

1.4 Relevant standards and sources of concepts for the Submodel template

According to [3], interoperable properties might be defined by standards, consortium specifications or manufacturer specifications. Useful standards providing sources of concepts are:

Table 1: List of standards defining interoperable properties

IDTA 02007-1-0 Submodel_SoftwareNameplate_InReview	https://industrialdigitaltwin.org/en/content-hub/submodels
IDTA 02002-1-0 Submodel_ContactInformation	https://industrialdigitaltwin.org/en/content-hub/submodels
IDTA 02011-1-0 Submodel_HierarchicalStructuresEnablingBoM	https://industrialdigitaltwin.org/en/content-hub/submodels
InterOpera Software BOM	
IDTA 02005-1-0 Submodel_ProvisionOfSimulationModels	https://industrialdigitaltwin.org/en/content-hub/submodels
SPDX	https://spdx.dev/wp-content/uploads/sites/41/2020/08/SPDX-specification-2-2.pdf
SPDX Licenses	https://spdx.org/licenses/
CyclonDX	https://cyclonedx.org/docs/1.4/json/
Software Identification (SWID) Tags ISO/IEC 19770-2:2015	https://csrc.nist.gov/publications/detail/nistir/8060/final

So called property dictionaries are used identify information elements (see Terms and Definitions of [6]). Such property dictionaries include:

- ECLASS, see: https://www.eclasscontent.com/
- IEC CDD, see: https://cdd.iec.ch/cdd/iec61987/iec61987.nsf and https://cdd.iec.ch/cdd/iec62683/cdddev.nsf

In this document, properties are aimed to be described by ECLASS.

1.5 Further standardization & outlook

At the moment this model defines a basis for a common package management with optional details about IEC 61131-3 features. In a next development cycle it should be extend in a deeper description to support an automatic detection of different dialects in IEC 61131-3 programming languages which could be solved automatically or an human is triggered to serve a mapping.

2 Information set for Submodel IEC 61131 Software Package Manager

While defining Submodels the following three aspects must be considered as suggested in [5]:

Use and economic relevance

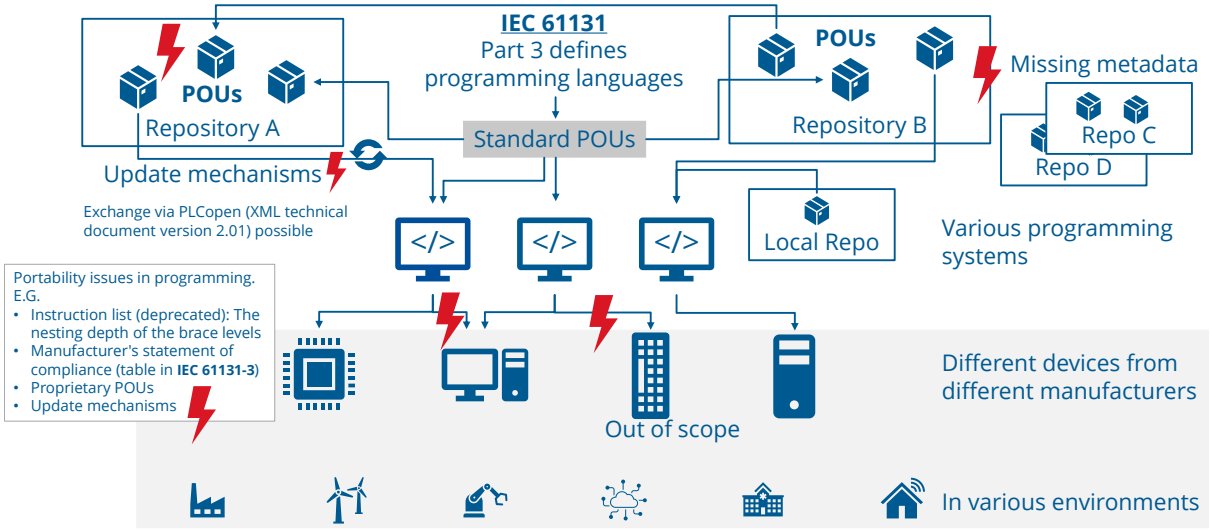

Figure 1: Use Case

The Submodel IEC 61131 Software Package Manager is designed for …

- A common description of an IEC 61131 Software Package
- Retrieving of necessary data in relation with the package
- Version management
- Update management
- Automatic dialect resolving between IEC 61360-3 interpretations.

Possible functions and interactions

With the usage of the Submodel template specified here, it will be possible to manage a software package as an asset and to obtain the necessary metadata as well as required source and/or program files from different sources and to resolve dependencies if required. The concept is show in Figure 2.

Each software package is an asset, which is described by an AAS. The dependencies refer to another asset. The AAS of the other asset can be used to provide all the necessary information and define repositories.

This avoids redundant data management. There are only relations between the AAS and the AAS Submodel refers to necessary files.

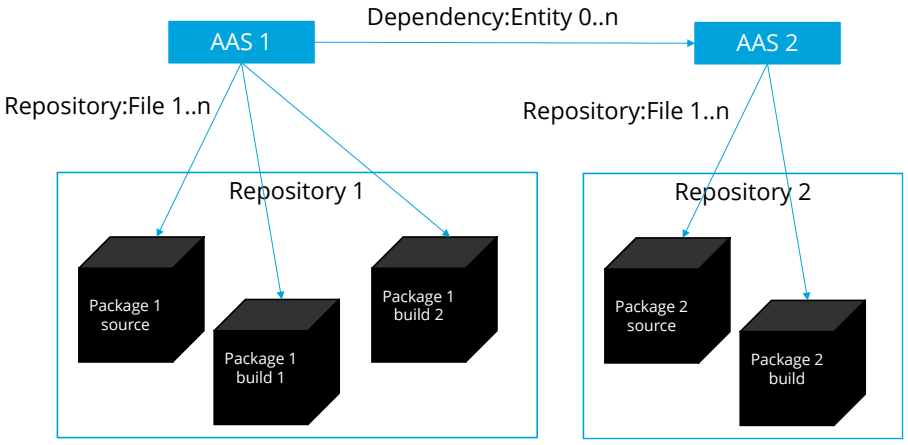

Figure 2: AAS Submodel concept for a general package management

Property specification

See section 3 Submodel and Collections.

3 Submodel and Collections

3.1 Properties of the Submodel “IEC_61131_Software_Package_Manager”

The Figure 3 below shows the UML-diagram defining the relevant properties which need to be set.

Note: This Submodel Template uses the Submodel Element Collection ContactInformation which is defined in IDTA 02002-1-0. All embedded properties and structures are not included in this specification. For further details and examples please use the IDTA Submodel Template Specification.

Note: This Submodel specification uses IRIs from the IDTA 02007-1-0 for Software Nameplate which is in review at the moment. Please use the final IRIs as semanticIDs when IDTA 02007-1-0 is published.[1]

Figure 3: UML-Diagram for Submodel "IEC_61131_Software_Package_Manager "

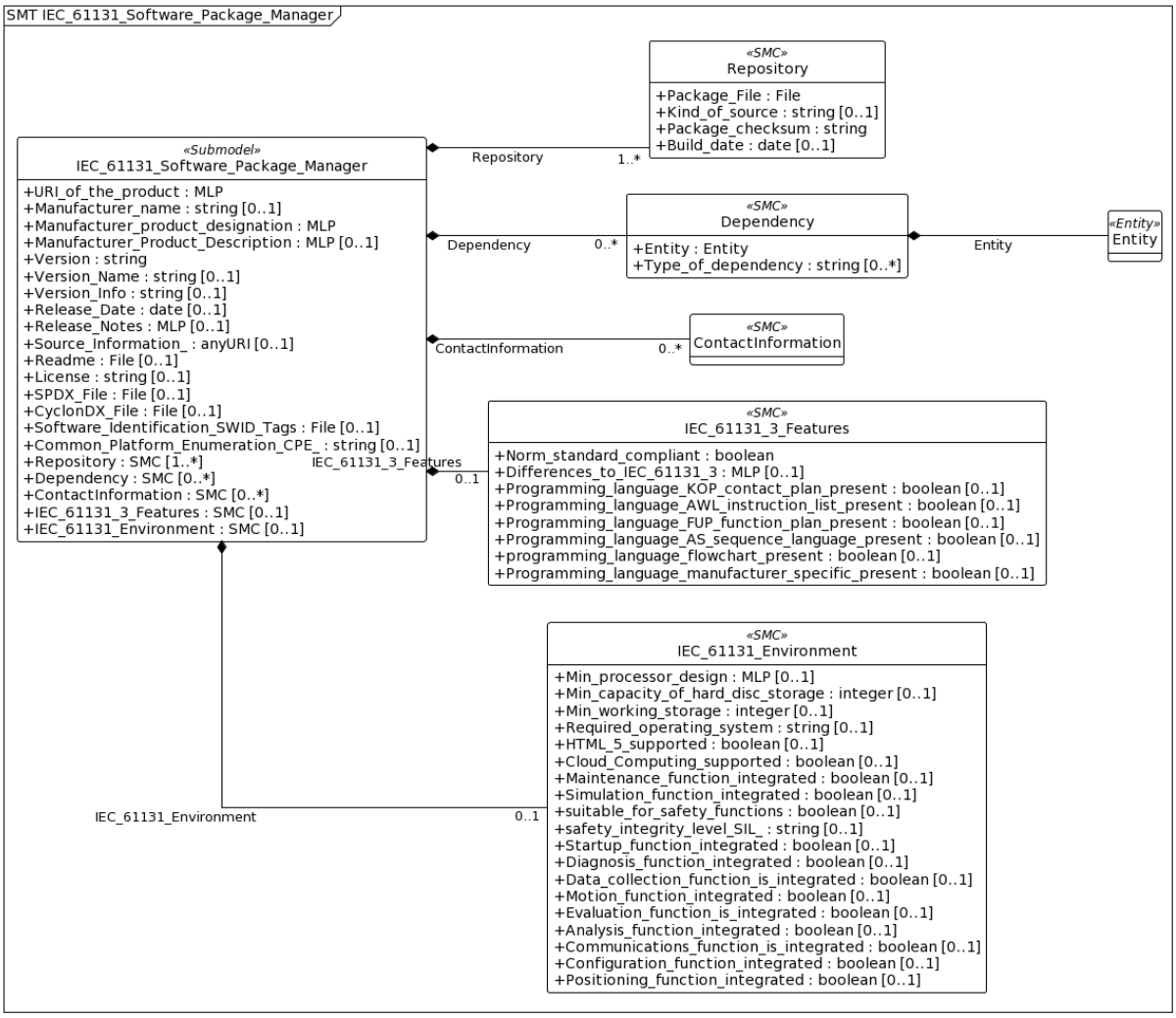

Table 2 describes the details of the Submodel structure combined with examples.

[1] This Submodel specification uses IRIs from the IDTA 02007-1-0 for Software Nameplate which is in review at the moment. Please use the final IRIs as semanticIDs when IDTA 02007-1-0 is published.

Table 2: Properties of Submodel “IEC_61131_Software_Package_Manager"

idShort	IEC_61131_Software_Package_Manager		
Class	Submodel		
semanticId	[IRDI]0173-EX-1#01-HQE037#001		
Explanation	A submodel template to support the package management of IEC 61131-3 based software packages		
[SME type]	**semanticId = [idType]value**	**[valueType]**	**card.**
idShort	**Description@en**	**example**	
[MLP] URI_of_the_product	[IRDI]0173-1#02-AAY811#001 unique global identification of the product using an universal resource identifier (URI)	[LANGSTRING] http://interopera.de/spm/1.0	[1]
[Property] Manufacturer_name	[IRDI]0173-1#02-AAO677#002 legally valid designation of the natural or judicial person which is directly responsible for the design, production, packaging and labeling of a product in respect to its being brought into circulation	[STRING] InterOpera	[0..1]
[MLP] Manufacturer_product_designation	[IRDI]0173-1#02-AAW338#001 Short description of the product (short text)	[LANGSTRING] IEC 61131 SPM	[1]
[MLP] Manufacturer_Product_Description	[IRI]https://admin-shell.io/idta/SoftwareNameplate/1/0/SoftwareNameplate/ManufacturerProductDescription[1] Description of the product, it's technical features and implementation if needed	[LANGSTRING] IEC 61131 Software Package which controls input and outputs	[0..1]
[Property] Version	[IRI]https://admin-shell.io/idta/SoftwareNameplate/1/0/SoftwareNameplate/Version[1] The complete version information consisting of Major Version, Minor Version, Revision and Build Number	[STRING] 1.0.0.0	[1]
[Property] Version_Name	[IRI]https://admin-shell.io/idta/SoftwareNameplate/1/0/SoftwareNameplate/VersionName[1] The name this particular version is given	[STRING] Release 1	[0..1]
[Property] Version_Info	[IRI]https://admin-shell.io/idta/SoftwareNameplate/1/0/SoftwareNameplate/VersionInfo[1] Provides a textual description of most relevant characteristics of the version of the software	[STRING] Please do not install in productive environments!	[0..1]
[Property] Release_Date	[IRI]https://admin-shell.io/idta/SoftwareNameplate/1/0/SoftwareNameplate/ReleaseDate[1] The moment in time, when this version of the software was made publicly available	[DATE] 2023-05-24	[0..1]

[MLP] Release_Notes	[IRI]https://admin-shell.io/idta/SoftwareNameplate/1/0/SoftwareNameplate/ReleaseNotes[1] contains information about this release	[LANGSTRING] This release requires special configuration.	[0..1]
[Property] Source_Information_	[IRDI]0173-EX-1#02-QLT562#001 A link to the source code as repository, directory or packaged file	[URL] http://github/Project/Repo	[0..1]
[FILE] Readme	[IRDI]0173-EX-1#02-WQL561#001 A reference to the corresponding readme file.	[FILE] http://example.com/readme.html	[0..1]
[Property] License	[IRDI]0173-EX-1#02-QLY594#001 The license identifier according https://spdx.org/licenses/	[STRING] Apache-2.0	[0..1]
[FILE] SPDX_File	[IRDI]0173-EX-1#02-DST698#001 A reference to the corresponding SPDX File. See E.g. https://spdx.dev/wp-content/uploads/sites/41/2020/08/SPDX-specification-2-2.pdf	[FILE] http://example.com/Package.spdx	[0..1]
[FILE] CyclonDX_File	[IRDI]0173-EX-1#02-LMK051#001 A reference to the corresponding CyclonDX File e.g. https://cyclonedx.org/docs/1.4/json/	[FILE] http://example.com/cyclonedx.json	[0..1]
[FILE] Software_Identification_SWID_Tags	[IRDI]0173-EX-1#02-TTR198#001 SoftWare IDentification SWID Tags according to ISO/IEC 19770-2:2015. See https://csrc.nist.gov/publications/detail/nistir/8060/final as well.	[FILE] http://example.com/SWID.xml	[0..1]
[Property] Common_Platform_Enumeration_CPE_	[IRDI]0173-EX-1#02-OVQ186#001 A formal correct CPE URI according https://nvd.nist.gov/products/cpe	[STRING] cpe:2.3:a:interopaer:package1:1.0.0.0:*:*:*:*:*:*:*	[0..1]
[SMC] Contact_information	[IRDI]0173-1#02-AAQ837#006 Information on a contact person Note: It is recommended to use the a value from https://www.loc.gov/marc/relators/relaterm.html for the RoleOfContactPerson. Note: See IDTA 02002-1-0 „Submodel for Contact Information“ for futher information. The SMC from this specifcatiuon is used.	[-]	[0..*]
[SMC] Repository	[IRDI]0173-EX-1#02-VLA027#001 Describes the location of the package and defines the path to the reference. Thus, a package can be deployed across multiple locations and different forms	[-]	[0..*]
[SMC] Dependency	[IRDI]0173-EX-1#02-RCI804#001 Defines a necessary software dependency and the link to the corresponding asset.	[-]	[0..*]

[SMC] IEC_61131_3_Features	[IRDI]0173-EX-1#02-CAI425#001 Defines the used IEC 61131-3 features of a software package	[-]	[0..1]
[SMC] IEC_61131_Environment	[IRDI]0173-EX-1#02-DVX269#001 Description of the runtime or development environment as hardware or software	[-]	[0..1]

3.2 Properties of the SMC “Repository”

The following table describes the details of the SMC “Repository” structure combined with examples.

Table 3: Properties of SMC " Repository"

idShort	Repository		
Class	SubmodelElementCollection		
semanticId	[IRDI]0173-EX-1#02-VLA027#001		
Explanation	Describes the location of the package and defines the path to the reference. Thus, a package can be deployed across multiple locations and different forms		
[SME type]	**semanticity = [idType]value**	**[valueType]**	**card.**
idShort	**Description@en**	**example**	
[FILE] Package_File	[IRDI]0173-EX-1#02-CDL566#001 Indicates the resource, where the software is being provided. Since there is no uniform file format globally and it is also not unified defined whether a package is exchanged e.g. as source code or binary, a mime type must be specified	[FILE] http://interopera.de/spm/1.0.bin	[1]
[Property] Kind_of_source	[IRDI]0173-EX-1#02-WJA108#001 Defines the package provision form. Possible values are "source", "binary", "interpretable" or "intermediate" . Note: If the list of values is complete, then this property should be defined as mandatory.	[STRING] binary	[0..1]
[Property] Package_checksum	[IRDI]0173-EX-1#02-VYU337#001 A checksum to validate the provided file. Algorithms that can be used: SHA1, SHA224, SHA256, SHA384, SHA512, MD2, MD4, MD5, MD6	[STRING] SHA1: 8….	[1]
[Property] Build_date	[IRI]https://admin-shell.io/idta/SoftwareNameplate/1/0/SoftwareNameplate/BuildDate[1] The moment in time, when this particular build of software was created	[DATE] 2023-05-24	[0..1]

3.3 Properties of the SMC “Dependency”

The following table describes the details of the SMC “Dependency” structure combined with examples.

Table 4: Properties of SMC "Dependency"

idShort	Dependency		
Class	SubmodelElementCollection		
semanticId	[IRDI]0173-EX-1#02-RCI804#001		
Explanation	Defines a necessary software dependency and the link to the corresponding asset.		
[SME type]	**semanticity = [idType]value**	**[valueType]**	**card.**
idShort	**Description@en**	**example**	
[Entity] Entity	[IRDI]0173-EX-1#02-OVC398#001 The Entity can be a co-managed or self-managed entity representing an dependent software package. Note: See IDTA 02011-1-0 „Hierarchical Structures enabling Bills of Material“ for a comparable concept.	[-]	[1]
[Property] Type_of_dependency	[IRDI]0173-EX-1#02-KOH046#001 Defines the kind of a dependency. Possible values are "Lib" or "Plugin"	[STRING] Lib	[0..*]

3.4 Properties of the SMC “IEC_61131_3_Features”

The following table describes the details of the SMC “IEC_61131_3_Features” structure combined with examples.

Table 5: Properties of SMC " IEC_61131_3_Features "

idShort	IEC_61131_3_Features		
Class	SubmodelElementCollection		
semanticId	[IRDI]0173-EX-1#02-CAI425#001		
Explanation	Defines the used IEC 61131-3 features of a software package		
[SME type]	**semanticity = [idType]value**	**[valueType]**	**card.**
idShort	**Description@en**	**example**	
[Property] Norm_standard_compliant	[IRDI]0173-EX-1#02-NHI719#001 According to norm standard IEC61131	[BOOLEAN]	[1]
[MLP] Differences_to_IEC_61131_3	[IRDI]0173-EX-1#02-CPE360#001 Description of the differences from the norm IEC 61131-3	[LANGSTRING]	[0..1]

[Property] Programming_language_KOP_contact_plan_present	[IRDI]0173-1#02-BAD206#008 whether the PLC programmable software/hardware is equipped with the programming language KOP (contact plan)	[BOOLEAN]	[0..1]
[Property] Programming_language_AWL_instruction_list_present	[IRDI]0173-1#02-BAD203#008 whether the PLC programmable software/hardware is equipped with the programming language AWL (instruction list)	[BOOLEAN]	[0..1]
[Property] Programming_language_FUP_function_plan_present	[IRDI]0173-1#02-BAD204#008 whether the PLC programmable software/hardware is equipped with the programming language FUP (function plan)	[BOOLEAN]	[0..1]
[Property] Programming_language_AS_sequence_language_present	[IRDI]0173-1#02-BAD202#008 whether the PLC programmable software/hardware is equipped with the programming language AS (sequence language)	[BOOLEAN]	[0..1]
[Property] programming_language_flowchart_present	[IRDI]0173-1#02-AAC395#003 whether the PLC programmable software/hardware is equipped with the programming language flowchart	[BOOLEAN]	[0..1]
[Property] Programming_language_manufacturer_specific_present	[IRDI]0173-1#02-BAD205#008 whether the PLC programmable software/hardware is equipped with a manufacturer-specific programming language	[BOOLEAN]	[0..1]

3.5 Properties of the SMC “IEC_61131_Environment”

The following table describes the details of the SMC “IEC_61131_3_Features” structure combined with examples.

Table 6: Properties of SMC "IEC_61131_Environment"

idShort	IEC_61131_Environment		
Class	SubmodelElementCollection		
semanticId	[IRDI]0173-EX-1#02-DVX269#001		
Explanation	Description of the runtime or development environment as hardware or software		
[SME type]	**semanticity = [idType]value**	**[valueType]**	**card.**
idShort	**Description@en**	**example**	
[MLP] Min_processor_design	[IRDI]0173-1#02-BAF723#003 Defined brand or make of a processor	[LANGSTRING]	[0..1]

[Property] Min_capacity _of_hard_disc_storage	[IRDI]0173-1#02-AAC860#008 Required minimum capacity of available hard disc storage	[INTEGER_MEASURE]	[0..1]
[Property] Min_working_storage	[IRDI]0173-1#02-AAC866#008 Statement of the capacity required for working storage	[INTEGER_MEASURE]	[0..1]
[Property] Required _operating_system	[IRDI]0173-1#02-AAO603#002 Statement of the operating system that is required	[STRING]	[0..1]
[Property] HTML_5_supported	[IRDI]0173-1#02-AAS515#001 Software supports HTML 5 as a data model	[BOOLEAN]	[0..1]
[Property] Cloud _Computing_supported	[IRDI]0173-1#02-AAS514#001 Software supports data storage and date exchange via a aloud and/or SaaS (Software as a service, i. e. data analysis)	[BOOLEAN]	[0..1]
[Property] Maintenance _function_integrated	[IRDI]0173-1#02-BAD626#008 whether the PLC programming software has an integrated maintenance function	[BOOLEAN]	[0..1]
[Property] Simulation _function_integrated	[IRDI]0173-1#02-BAD597#008 whether the PLC programming software has an integrated simulation function	[BOOLEAN]	[0..1]
[Property] suitable _for_safety_functions	[IRDI]0173-1#02-BAD722#009 whether the device is suitable for safety functions	[BOOLEAN]	[0..1]
[Property] safety _integrity_level_SIL_	[IRDI]0173-1#02-AAO382#009 discrete level (one out of a possible four), corresponding to a range of safety integrity values, where safety integrity level 4 has the highest level of safety integrity and safety integrity level 1 has the lowest Note: According to ECLASS it is recommended to use a value from the ECLASS valuelist 0173-1#09-AAD743#006	[STRING]	[0..1]
[Property] Startup _function_integrated	[IRDI]0173-1#02-BAD542#009 whether the SPC programming software is equipped with an integrated startup function	[BOOLEAN]	[0..1]
[Property] Diagnosis _function_integrated	[IRDI]0173-1#02-BAD511#008 whether a diagnosis function is integrated in the SPC programming software	[BOOLEAN]	[0..1]

[Property] Data_collection _function_is_integrated	[IRDI]0173-1#02-BAD506#008 whether the SPC programming software contains a data collection function	[BOOLEAN]	[0..1]
[Property] Motion _function_integrated	[IRDI]0173-1#02-BAD495#008 whether a motion function is integrated in the SPC programming software	[BOOLEAN]	[0..1]
[Property] Evaluation_function _is_integrated	[IRDI]0173-1#02-BAD488#008 whether an evaluation function is integrated into the SPC programming software	[BOOLEAN]	[0..1]
[Property] Analysis _function_integrated	[IRDI]0173-1#02-BAD477#008 whether the PLC programming software has an integrated analysis function	[BOOLEAN]	[0..1]
[Property] Communications _function_is_integrated	[IRDI]0173-1#02-BAD547#008 whether a communications function is integrated in the SPC programming software	[BOOLEAN]	[0..1]
[Property] Configuration _function_integrated	[IRDI]0173-1#02-BAD549#008 whether a configuration function is integrated in the SPC programming software	[BOOLEAN]	[0..1]
[Property] Positioning _function_integrated	[IRDI]0173-1#02-BAD573#008 whether the PLC programming software is equipped with an integrated positioning function	[BOOLEAN]	[0..1]

4 Usage as SubmodelElementCollection

During the development there was already an exchange with the project Software Bill of Materials (SBOM) by InterOpera. As this Submodel also uses elements and SubmodelElementCollection from other templates, harmonization also makes sense here. Depending on the use case and further development, the SubmodelElementCollection Repository and Dependency may be interesting for SBOM.

Annex A: Explanations on used table formats

General
The used tables in this document try to outline information as concise as possible. They do not convey all information on Submodels and SubmodelElements. For this purpose, the definitive definitions are given by a separate file in form of an AASX file of the Submodel template and its elements.

Tables on Submodels and SubmodelElements
For clarity and brevity, a set of rules is used for the tables for describing Submodels and SubmodelElements.

- The tables follow in principle the same conventions as in [5].
- The table heads abbreviate 'cardinality' with 'card'.
- The tables often place two informations in different rows of the same table cell. In this case, the first information is marked out by sharp brackets [] form the second information. A special case are the semanticIds, which are marked out by the format: (type)(local)[idType]value.
- The types of SubmodelElements are abbreviated: SME

SME type Submodel	Element type
Property	Property
MLP	MultiLanguageProperty
Range	Range
File	File
Blob	Blob
Ref	ReferenceElement
Rel	RelationshipElement
SMC	SubmodelElementCollection
ENT	Entity

- If an idShort ends with '{00}', this indicates a suffix of the respective length (here: 2) of decimal digits, in order to make the idShort unique. A different idShort might be choosen, as long as it is unique in the parent’s context.
- If a table does not contain a column with "parent" heading, all represented attributes share the same parent. This parent is denoted in the head of the table.
- Multi-language strings are represented by the text value, followed by '@'-character and the ISO 639 language code: example@de.
- The [valueType] is only given for Properties.

Bibliography

[1] “Recommendations for implementing the strategic initiative INDUSTRIE 4.0”, acatech, April 2013. [Online]. Available: https://www.acatech.de/Publikation/recommendations-for-implementing-the-strategic-initiative-industrie-4-0-final-report-of-the-industrie-4-0-working-group/

[2] “Implementation Strategy Industrie 4.0: Report on the results of the Industrie 4.0 Platform”; BITKOM e.V. / VDMA e.V., /ZVEI e.V., April 2015. [Online]. Available: https://www.bitkom.org/noindex/Publikationen/2016/Sonstiges/Implementation-Strategy-Industrie-40/2016-01-Implementation-Strategy-Industrie40.pdf

[3] “The Structure of the Administration Shell: TRILATERAL PERSPECTIVES from France, Italy and Germany”, March 2018, [Online]. Available: https://www.plattform-i40.de/I40/Redaktion/EN/Downloads/Publikation/hm-2018-trilaterale-coop.html

[4] “Beispiele zur Verwaltungsschale der Industrie 4.0-Komponente – Basisteil (German)”; ZVEI e.V., Whitepaper, November 2016. [Online]. Available: https://www.zvei.org/presse-medien/publikationen/beispiele-zur-verwaltungsschale-der-industrie-40-komponente-basisteil/

[5] “Verwaltungsschale in der Praxis. Wie definiere ich Teilmodelle, beispielhafte Teilmodelle und Interaktion zwischen Verwaltungsschalen (in German)”, Version 1.0, April 2019, Plattform Industrie 4.0 in Kooperation mit VDE GMA Fachausschuss 7.20, Federal Ministry for Economic Affairs and Energy (BMWi), Available: https://www.plattform-i40.de/PI40/Redaktion/DE/Downloads/Publikation/2019-verwaltungsschale-in-der-praxis.html

[6] “Details of the Asset Administration Shell; Part 1 - The exchange of information between partners in the value chain of Industrie 4.0 (Version 3.0RC01)”, November 2020, [Online]. Available: https://www.plattform-i40.de/PI40/Redaktion/EN/Downloads/Publikation/Details-of-the-Asset-Administration-Shell-Part1.html

[7] “Semantic interoperability: challenges in the digital transformation age”; IEC, International Electronical Commission; 2019. [Online]. Available:https://basecamp.iec.ch/download/iec-white-paper-semantic-nteroperability-challenges-in-the-digital-transformation-age-en/

[8] https://www.zvei.org/fileadmin/user_upload/Presse_und_Medien/Publikationen/2023/April/Software_Bill_of_Material/2023_04_14_ZVEI_Software_Bill_of_Material__SBOM__Grundlagen-Einschaetzungen-Ausblicke.pdf

[9] https://www.loc.gov/marc/relators/relaterm.html