Document title: Examples of Arrowhead Tools Semantic Approach

Version	**Status**	**Date**
1.0	final	2019-12-10
Author	**Contact**	
Géza Kulcsár	geza.kulcsar@incquerylabs.com	

Examples of Arrowhead Tools Semantic Approaches

IQL: Géza Kulcsár
geza.kulcsar@incquerylabs.com

Abstract

This document is a collection of semantic approaches as proposed by the partners involved in such activities, loosely adhering to the general principles identified in an accompanying document (O4).

ECSEL EU project 826452 - Arrowhead Tools
Project Coordinator: Professor Jerker Delsing | Luleå University of Technology

Table of contents

1. Examples of Arrowhead Tools Semantic Approaches

This document contains a collection of Arrowhead Tools semantics proposed by project partners involved in Task 4.2.

The document is split into (1) *engineering tool semantics*, where most of the proposals available so far conceptually belong; and (2) *interoperability of semantics,* where some initial considerations are presented here, along with an example planned contribution.

Most of the proposals follow the template in the document **O4**, also being part of this delivery, **D4.1**. We recall here the meaning of the letters A-F as in **O4**; in addition, G provides a standard reference to related literature. The proposals not formally addressing points A-G also contain the relevant information and should be converted to adhere to that list as part of future work.

1.1 Engineering tool semantics

1.1.1 System model semantics (IQL)

This activity aims at supporting the Arrowhead engineering process by adding a SysML-based systems design profile and methodology to the AH landscape. Similarly to an ontology, the model represents the semantics interpretation of an SoS instance with the abstraction of various layers. Moreover, systems modeling makes it available to address the execution semantics of the system by well-defined behavioral specifications attached to the model.

A.) SysML, a UML-based (meta-)modeling platform for systematic systems design (*design and modeling*).
B.) Not directly, although it will be evaluated based on concrete UCs.
C.) SysML models allow for a customizable', textual output, which can be consumed by other AH tools. E.g., a standardized description format can be read by the Arrowhead Management Tool to initialize a local cloud configuration.
D.) Typically, MagicDraw, a well-established tool is used, along with other technologies from No Magic, e.g., Teamwork Cloud for collaborative modeling.
E.) Native support of various export formats. We're looking at JSON, among others, for exporting model information towards run-time tools.
F.) Models are static artifacts *per se*, however, behavioral specifications might be investigated in this context, thus addressing the dynamic (yet design-time) aspect as well.
G.) Sanford Friedenthal, Alan Moore, and Rick Steiner: A Practical Guide to SysML. The Systems Modeling Language. MK/OMG Press (2015).

1.1.2 Digital Reference (IFAG)

The Digital Reference developed by the ECSEL consortium is based on Semantic Web technologies. The concept of Semantic Web was introduced by Tim Berners-Lee in 2001 with the aim to provide web contents not only in human-readable form as in the traditional World Wide Web but also in machine-readable form. The IT systems should be able to process information from web sites and other data sources in order to recognize relationships as well as dependencies between pieces of data, make implicit knowledge explicit and link data from different data sources effectively (Baumgärtel 2018). The Digital Reference ontology, introduced to Productive4.0, is a supply chain-related Semantic Web mirror of the semiconductor industry depicting a combination of different supply chain pillars and semiconductor production concepts e.g. Digital Production, Supply Chain Networks, and

Product Lifecycle Management. The Digital Reference organized in topic clusters (lobes) represents all stages of the supply chain and provides a knowledge base readable for humans and computer alike. The Digital Reference contains around 500 classes that describe concepts in different domains. Currently, the Digital Reference is the overarching ontology containing several sub-ontologies that represent hierarchies, processes and taxonomies e.g. product ontology, sensor ontology, organization ontology and process ontology. The scale of the Digital Reference continues to expand with the integration of further relevant ontologies.

A.) A RDF/OWL based semantic web representation of the semiconductor supply chain and of supply chains that contain (employ) semiconductors. It is visualized in WebVOWL (*design and modeling, data interaction*).
B.) The approach is attached to UC-05: *Support quick and reliable decision making in the semiconductor industry*.
C.) The knowledge graph enables acquisition, processing and extraction of data that is considered expert knowledge. Data is understandable across disciplines and for humans as well as machines.
D.) Currently, Protégé is used for ontology development, WebVOWL for visualization as well as certain rule sets, reasoner mechanisms and pre-developed semantic web standards.
E.) Various formats possible, most relevant are JSON, RDF, OWL, XML.
F.) Primarily static, however dynamic behavior for some use cases is intended, yet not reached so far.
G.) Hitzler, P., M. Krötzsch, and S. Rudolph. 2009. Foundations of Semantic Web Technologies. Boca Raton, Florida: Chapman & Hall/CRC.
Baumgärtel, H., H. Ehm, S. Laaouane, J. Gerhardt, and A. Kasprzik. 2018. "Collaboration in Supply Chains for Development of CPS Enabled by Semantic Web Technologies". In Proceedings of the 2018 Winter Simulation Conference, edited by M. Rabe, A. A. Juan, N. Mustafee, A. Skoogh, S. Jain, and B. Johansson, 3627-3638. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.

1.1.3 OSLC-enabled Program Analysis Tool (BUT)

The goal is to extend some selected program analyser (either static or dynamic) by an OSLC-based interface, allowing for easy integration of such a tool into various development environments and for its easy combination with other program analysis tools. OSLC (Open Services for Lifecycle Management) is a community, standard, and technology that aims at the integration of software development tools. OSLC is based on the REST (Representation State Transfer), RDF (Resource Description Framework), and Linked Data technologies. OSLC covers various domains, out of which the Automation domain, which focuses on the integration of analysis, build, and deployment tools, is of interest to the considered scenario. Brno University of Technology (BUT) has already got some experience with building prototype support of OSLC for one of its analysers, namely, the ANaConDA framework for dynamic analysis of concurrent programs. The goal is to improve this prototype OSLC support into a more mature form and, if possible, extend some complementary analysis tool, such as the Perun dynamic performance analyser, by an OSLC support too. For that, the OSLC Core and Automation domains are to be suitably used (or possibly specialised) and complemented by a specific domain for the chosen analyser. The tools considered in the above are, in particular, used in UC1 that targets verification methods, but the result can be useful more broadly.

[OSLC] Open Services for Lifecycle Collaboration. https://open-services.net/.

1.1.4 Heterogeneous engineering models (TUE)

Model-based systems design involves various modeling languages and activities across different product versions and variants, and additionally throughout the entire product lifecycle. In order to support the design, management and evolution of model-based systems, we would like to integrate the following items into the framework:
- Model repository: a model repository allowing controlled CRUD operations on modelling artifacts, with additional features such as versioning and provenance.
- Model analytics functionality: services for performing complicated analysis on modeling artifacts, such as detecting duplication and other problems, monitoring the size and evolution of models.
- Model management functionality: a dashboard-like system which runs model analytics services on the model repository for the purpose of monitoring the model-based system, its quality and evolution.
- Model consistency: in typical model-based systems engineering, multiple models (e.g., covering different parts of a system, or coming from different domains) are used. They need to stay consistent when changes are made to one or more of them. As a first step, impact analysis and flagging of potentially inconsistent models and model parts is needed; ideally, automatic co-evolution is possible.

Interoperability and Integration
We will investigate the workflows for design and run-time integration/interoperability in our case. However, we can comment on the following aspects:
- Design-time integration/interoperability: The tools have different formats and syntaxes for models, e.g., XML vs. plain text. Some tools already natively offer conversions, such as IBM Rhapsody for Simulink models.
- Run-time integration/interoperability: In our underlying academic use case, we have the native run-time integration of IBM Rhapsody with Simulink and additionally an ad-hoc network-based integration with Unity.
- Storage and integration via the model repository: As the artifacts are located and managed in a repository, this also serves directly to the communication and integration of the tools in a centralized manner.

A.) Federation of model storage, analytics and management services with the following categories of interaction: *Design and Modeling, Data Storage*.
B.) The approach is not attached to the existing use cases, but aims to support them.
C.) The model storage acts as a central mechanism to ensure persistence, versioning and provenance. Model analytics and management tools developed at TU/e lead to (automated) checks for quality and consistency supporting design-time interoperability. The tools IBM Rhapsody and Simulink along with the future extension for Unity will tackle run-time operability in the toolchain.
D.) We will use the existing tools IBM Rhapsody and MATLAB/Simulink, which already support a certain level of interoperability. We will further develop extensions as well as model analytics and management tools ourselves.
E.) Standard exchange format planned to be used for interoperation: EMF models, XML/XMI serialization.
F.) The semantics is mostly static in nature, but the run-time interaction of the tools (e.g. co-simulation, user interaction) will possess a dynamic aspect as well.

1.1.5 Ontologies for CPS modeling (HIOF)

This work aims to address data interoperability challenges among the services and systems within an Arrowhead Local Cloud. Specifically, Arrowhead Systems employ heterogeneous (domain-specific) models for the design of their service interfaces and exchanging data, which limits the level of interoperability among the systems. Furthermore, the heterogeneity in the design obstructs the monitoring and analysis of the services and systems. Our approach allows dynamically orchestrating the services and re-configuring the architecture of System-of-Systems. We consider different standardized ontologies to model the services, systems, system-of-systems architecture within the Arrowhead Framework. Based on those ontologies, state-of-the-art machine learning and natural language processing methods can be employed to extract information about the services and systems from the heterogeneous information models. Semantic web technologies are used for analysis and reasoning for orchestration and configuration of the systems.

A.) Standardized and well-known ontologies such as STEP standard, W3C SSN, Productive4.0 ontologies would be considered at the design stage of the approach (*design and modeling*).
B.) The approach will be applied for condition monitoring and configuration of the Knuckle Boom Crane of the Norwegian use-case (UC-12).
C.) A system will be developed to cooperate with other Core Systems of the Arrowhead Framework as well as the Arrowhead Management Tool to exchange information and high-level policies for orchestration and configuration.
D.) Protégé is used for ontology development. Semantic web technologies will be used for the analysis and reasoning.
E.) Various formats could be used. Currently, JSON, RDF, SPARQL, SWRL are employed.
F.) Design-time models could be static. However, the extracted information could be updated during runtime.
G.) [1] Lam, An Ngoc, and Øystein Haugen. "Supporting IoT semantic interoperability with autonomic computing." *2018 IEEE Industrial Cyber-Physical Systems (ICPS)*. IEEE, 2018.
[2] Lam, An Ngoc, and Øystein Haugen. " Applying semantics into Service-oriented IoT Framework." *2019 Industrial Applications of Artificial Intelligence (INDIN)*. IEEE, 2019

1.1.6 **Ontologies for workflows (MGEP)**

Although MGEP does not participate in any use case, MGEP will investigate on the ontologies and semantic technologies that enable the representation of the data exchanged and the systems and processes involved in the scope of Arrowhead. The results of this research will enable the construction of a semantic repository where inference and intelligent agents are deployed. The focus of this research will concentrate on process orchestration addressed by Arrowhead with the Workflow Manager or Workflow Choreographer. This implies that the Semantic Engine to be created will collaborate with the Workflow Choreographer and Workflow Executor in monitoring and managing processes.
To achieve this goal ,several tools and technologies are necessary. The following sections present the technologies and tools to be used in the construction and deployment of semantic ontologies for workflows.

Design and modeling

To model the elements and relations that represent the workflow domain ontologies are necessary. Semantic ontologies are formal vocabularies stored as documents on the Web. They describe and represent a data domain as a set of concepts and complex relationships between them. Ontologies enable to create a general knowledge that can be queried, processed and shared across different software applications. Ontologies are developed in **OWL** (Web Ontology Language) language [1], the standard ontology language proposed by

the W3C (World Wide Web Consortium). OWL is used to represent complex knowledge about things for applications that need to process the content of information instead of just presenting it to humans. OWL provides the basis for creating vocabularies used to describe web data with high expressiveness. With this data representation, intelligent agents can perform advanced data analysis and reasoning for knowledge extraction and decision-making.
MGEP plans to develop an ontology to represent the data used for monitoring and managing processes by the Arrowhead services dedicated to Workflow Management that we will implement in WP3. It will be an experimental testbed to support scheduling. No use cases are related to the framework proposed here. The ontology will include the vocabularies that define how the data will be represented. The ontology will be developed with the **Protégé ontology editor** [2] a well-known ontology development tool that enables the creation of ontologies in OWL language.

Storage

Data collected and structured according to an ontology needs to be stored for later consumption. Semantic repositories are used to store data represented with an ontology. Semantic repositories are used to store web data in OWL language. All these statements together form a knowledge base. These semantic repositories are similar to database management systems. In addition, the semantic repository includes semantic schemas to automatically reason about the queried data. Many semantic frameworks and repositories use rule-based inference engines, which combine knowledge base assertions and a set of logical rules to infer new information about the knowledgebase.
A semantic repository will be created to store the data represented with the ontology. The semantic repository will be built using **Ontotext Graph DB** [3] multiplatform semantic repository since MGEP have successfully applied this repository in previous projects.

Semantic applications

Applications that interact with the semantic repository and extract knowledge are also necessary. These applications are the sample of the added value offered by the semantic technologies. To support workflows and processes, a semantic converter will be built. A semantic converter is an application that enables to manage (query, add or remove) data represented with the vocabularies of an ontology. These data are stored usually in the semantic repository. A semantic converter includes the libraries of a semantic framework. Semantic frameworks are collections of tools and libraries used to manage semantically represented data.
MGEP plans to develop a semantic converter to manage the data represented with the ontology vocabularies. In addition, the semantic converter can be used to perform intelligent data analysis and decision making based on the queried data. The semantic converter will use the libraries of a semantic framework to manage the semantically represented data. In particular, MGEP will use the **RDF4J** [4] framework (formerly SESAME), which is written in Java. RDF4J is an open-source semantic web framework that supports RDF data storage, retrieval and analysis. Bearing in mind that RDF4J is implemented in Java, the semantic repository will be implemented in the same language.
The results obtained with the application (converter) will be presented using visualization tools. Thus, a Human Machine Interface (HMI) will query the data from the semantic repository using the semantic converter and display it.

Implementation Plan

To build the solution proposed in the previous sections the following steps will be conducted:

1. Development of the ontology. The ontology will be developed following the guidelines of a well-known ontology development methodology. The main phases of the ontology development process will include the following:
    a. Definition of the ontology requirements.
    b. Analysis of previously developed ontologies that represent the knowledge about automation processes.

c. Selection and reuse of the elements of a set of the analysed ontologies.
d. Implementation of the ontology
e. Evaluation of the syntax and logical consistency of the ontology.
f. Publication of the ontology on the web.

2. Creation of a semantic repository, to store the data represented with the ontology vocabularies. The ontology will be loaded into the semantic repository to enable the data storage with the ontology vocabularies.
3. Development of the semantic converter. The development of the semantic converter will follow the following steps:
a. Definition of the semantic converter requirements (depending on the use cases supported by the proposed solutions).
b. Definition of the methods included by the semantic converter.
c. Implementation of the semantic converter using the libraries of the RDF4J framework.
d. Verification tests of the converter against the semantic repository in a local environment.
4. Development and testing of the HMI.

Semantic Characterization of the Approach

The following points characterize this semantic approach (Ontologies for Workflows) according to the assessment criteria presented in Section 1.2 in the delivery document O4, being part of the same delivery D4.1 as the present document.

A.) The scheduling semantic approach presented in this section is called **Process Scheduling Semantic Repository** and uses ontologies to represent processes and schedules and a semantic repository to store semantic data necessary to make decisions. Consequently the category or interaction is twofold; **Design and modelling**, and **Data Storage**.
B.) This semantic approach is not attached to any use case within Arrowhead Tools. It is a prototype to support the Workflow Choreographer in addressing scheduling.
C.) Interoperability is supported as explained in subsections Design and Modelling and Storage.
D.) The following standard tools and techniques will be employed:
  a. At Design and Modelling level, Protégé ontology editor will be used for ontology creation/adaptation.
  b. At Storage level, a semantic repository will be built using Ontotext Graph DB.
  c. At semantic application level, a semantic converter will be used (RDF4J)
E.) In relation to standards exchange format for interoperation, OWL will be used for ontology development and RDF for semantic data storage.
F.) The semantic data representation will change in time depending on the state of the processes and the resources they use. Consequently, semantics will have a dynamic behaviour.
G.) [1] https://www.w3.org/OWL/
[2] https://protege.stanford.edu/
[3] http://graphdb.ontotext.com/
[4] http://rdf4j.org/

1.1.7 Smart Testing (UNIKL)

UNIKL works on a data model for development-operation integration. The data model strives to join data from both from run-time and development-time with
a) Numerical analysis techniques ,such as verification, optimization. This data includes:

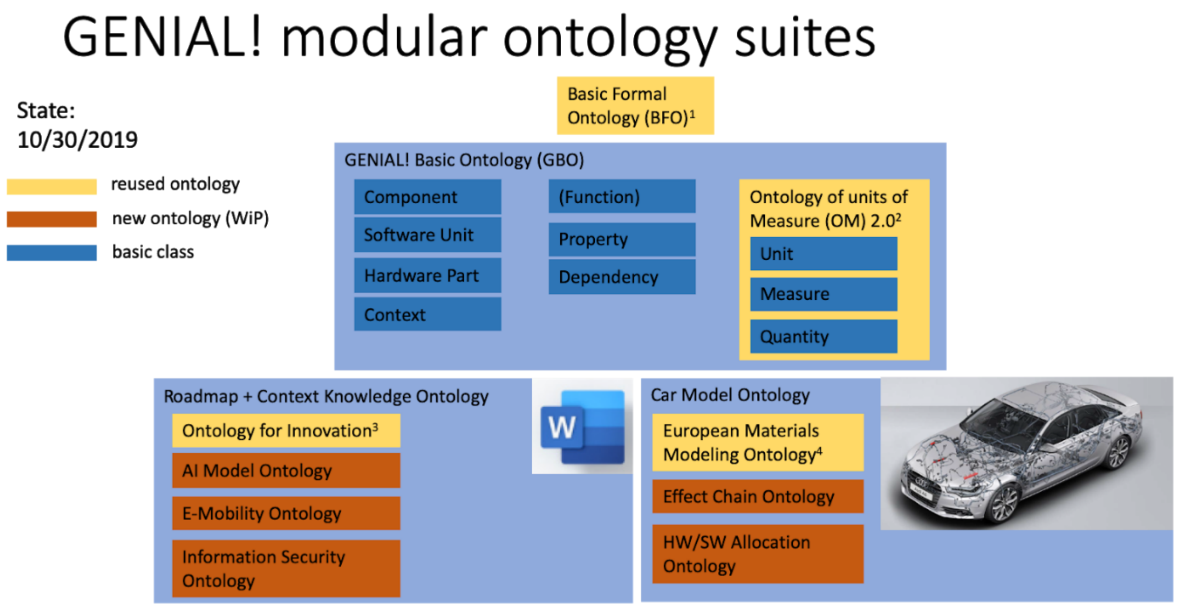
- Reachability and sensitivity data of selected quantities that act as a basis for a sound and system-wide verification and monitoring approach as described in (see *Zivkovic 2019*).
- Samples and data obtained from operation.

b) Semantic technologies that semantify and disambiguate data throughout the product life cycle and across the value chain, in particular, of the automotive industry (tier-2, tier-1, OEM), assigning the data from operation some semantics.
The link between the numerical techniques and the semantic technologies is created by

- Hierarchical decomposition of the system in parts and functions, that defines the context of the data, and the associated
- Properties (and constraints thereof) of parts and functions, and the
- Dependencies of parts and functions from other properties.

Properties, constraints thereof, and dependencies permit us to, during runtime, check the consistency of the samples obtained from operation with the models from development (represented by reachability analysis results). This is permitted by the dependencies that are just ASCII representations of constraints and equations that must hold universally.
Focus of semantic interoperability is on the extension of the ontologies of the GENIAL! Project, that is a German BMBF-Project, into the Development-Operation continuum. However, this ontology is just defining a top-level ontology into which other ontologies can easily be integrated.The focus on the use case smart testing is to extend the symbolic model checking approach from development time to run-time while providing a sound, scalable ,and effective data-model across the product lifecycle. The GENIAL! Ontology that we use is based resp. is compatible with:

- The Basic Formal Ontology (http://purl.obolibrary.org/obo/bfo/2.0/) gives the highest-level structure.
- The GENIAL! Ontology defines the means to represent configurations, specializations, and hierarchical de-composition of a system.


GENIAL! modular ontology suites
State: 10/30/2019
reused ontology
new ontology (WiP)
basic class
Basic Formal Ontology (BFO)¹
GENIAL! Basic Ontology (GBO)
Component
Software Unit
Hardware Part
Context
(Function)
Property
Dependency
Ontology of units of Measure (OM) 2.0²
Unit
Measure
Quantity
Roadmap + Context Knowledge Ontology
Ontology for Innovation³
AI Model Ontology
E-Mobility Ontology
Information Security Ontology
Car Model Ontology
European Materials Modeling Ontology⁴
Effect Chain Ontology
HW/SW Allocation Ontology

- We furthermore use ontologies for the units (http://www.ontology-of-units-of-measure.org/resource/om-2/) and materials (http://emmc.info/emmo).

(Zivkovic 2019) *C. Zivkovic, C. Grimm, M. Olbrich, O. Scharf, E. Barke (2019): Hierarchical Verification of AMS Systems with Affine Arithmetic Decision Diagrams. In: IEEE Transactions on CAD of Circuits and Systems; Vol. 38, No. 10, Oct. 2019.*

1.1.8 **Ontologies for STEP and Open BIM (Jotne)**

There is a need for data interoperability in industry. In 1984 ISO started the „Industrial Data" sub-committee ISO/TC 184/SC 4 pursuing the following vision: „The vision is of a data model that includes the product and process data necessary for the operation of all the enterprises in the network of enterprises, inter-related by producer-purchaser and partnership interactions, in a consistent manner. This data model is easily partitioned, both as to schema, and to actual data, into subsets representing the data necessary for the successful operation of business functions in each individual enterprise. Such subsets support all the data requirements for the product and process development activities within that enterprise."

This vision resulted in several series of standards, the most relevant one for the manufacturing industry being ISO 10303, „Product data representation and exchange", which consists of several hundred standard documents. ISO/TC 59/SC 13 used the same methodology to develop ISO 16739, „Industry Foundation Classes (IFC) for data sharing in the construction and facility management industries", also called Open BIM.

Both standards represent the ontologies of their engineering domains in object-based data models written in the EXPRESS data modelling language (ISO 10303-11). The objects or entities of the data models describe real-world items on a medium semantical level; examples are „product", „property", „representation" and „representation_item". They are interrelated to allow the description of products and processes down to the finest level of granularity to allow data exchange among, for example, PLM, CAD, CAE, manufacturing and facility management systems. However, the end item semantics are not standardized; thus, you will not find terms, like car, ship and aircraft or length, width and height in the data models. Those may be associated with the ISO 10303 and ISO 16739 standards by reference data libraries. The EXPRESS schemas may there be used as upper ontologies and supplied by trees of subclasses. Data converters are able to check exchange data sets against both the semantics of the schema and the semantics of the associated reference data.

A.) Name of the approach/tool/standard and the field it belongs to: STEP/ISO 10303 and Open BIM/ISO 16739
B.) If the approach is attached to a use-case: Use case in Task 8.7, „Digital twins and structural monitoring", will apply STEP/ISO 10303. Both standards are under consideration by other use cases.
C.) An example of how the approach will work and be supported by tools: The use case goal is to determine structural changes of an offshore crane through sensor variations over time. Based on available historical sensors and data, and later augmentations of sensors and measurements, likely historical damage (when and what) will be quantify together with their impact on future failure and maintenance. Using physical inspections and in-situ measurements in parallel with the (ideally) historical digital models (or regenerated historical models) will identify and classify damage with respect to severity from a safety standpoint as well as merely cost issues. The design and analysis aspects of the digital model of the crane will be described in ISO 10303-209, „Multidisciplinary analysis and design"; this is supported by the Jotne tool *EDMopenSimDM*. The overall product life-cycle management data of the crane and its sensors, including the information for predictive maintenance based on real-time measurements, will be stored compliant to

ISO 10303-239, „Product lifecycle support" (PLCS) ; this is supported by the Jotne tool *EDMtruePLM*. The exchange of product data among the constituents of the use case will – as much as reasonable – apply the STEP model and file format. Translators to and from proprietary formats will be implemented using the Jotne software development kit *EDMsdk*.
D.) If there is a mature/standardized tool supporting the activity (or if it should be developed): EDMsdk is an off-the-shelf product, whereas *EDMopenSimDM* and *EDMtruePLM* will be enhanced as part of the project.
E.) Standard exchange format, if any (JSON, XML, EMF models, …): ISO 10303-21
F.) Mostly dynamic.
G.) Pointers to the underlying theory, with literature citations: iso.org; https://authors.elsevier.com/sd/article/S0965997818301947

1.2 Interoperability of semantics

In order to provide an integrated solution, addressing an interworking of different parallel engineering semantics (cf. the previous section) and the corresponding tools, we plan to introduce a *translation* mechanism.

On the side of the Arrowhead Framework, a dedicated Translation system will be developed to tackle such issues (detailed elsewhere). As for the scope of tool interoperability, we consider two modes of translation:

- 1-to-1 dedicated translation: such a translation is based on an *a priori* mapping between two different semantics based on expert knowledge, meta-models or ontologies and realizes an interoperability interface between those.
- Learning-based translation: In contrast, we will consider plug-ins for automated translation based on machine learning techniques, which will, thus, hopefully be able to adapt to diverse translation scenarios without having to rely on a mapping (although expert knowledge is still required to supervise learning).

In the following, we describe a first example proposal for the latter concept, with leaving the study of further suggestions for future work.

Example: Autoencoder based dynamic message translation tool (LTU)

This task builds on work carried out in the Productive 4.0 project and focuses on the problem to establish dynamic interoperability in heterogeneous and evolving Systems-of-Systems that are subject to different standards and information models. The approach is to combine and adapt state of the art machine learning methods for text and graph processing and address the semantic message translation problem using an unsupervised autoencoder based approach. Message text data and graph metadata are to be combined in the learning protocol to align the latent representations learned by the message autoencoders so that any two autoencoders can be combined dynamically to translate a message from one system (/semantic domain) to another. Thus, the core problem addressed by the tool is dynamic message translation in the Arrowhead Framework by autoencoder alignment. Furthermore, the possibility to complement the unsupervised learning protocol for alignment of autoencoders with the SoS goals and constraints to further improve the translations will be explored. The prototype tool is to be implemented in Python using standard machine learning tools (like PyTorch) using the Python interface of the Arrowhead Framework.

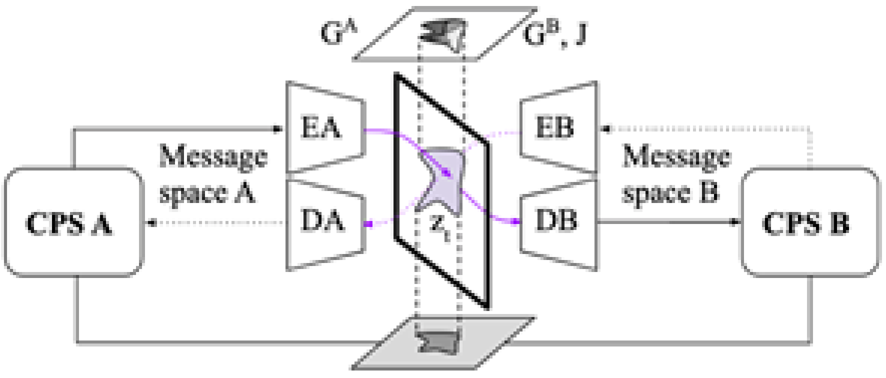


Figure 1. Message translation and back translation by aligned autoencoders (work in progress). Encoders EA/EB and decoders DA/DB are trained to align the latent spaces using graph metadata GA/GB and, optionally, a system of system utility function J.

Contact: Jacob.Nilsson@ltu.se (PhD student), Fredrik.Sandin@ltu.se (supervisor).

Reference: Jacob Nilsson, Fredrik Sandin, Jerker Delsing. *Interoperability and machine-to-machine translation model with mappings to machine learning tasks*, IEEE International Conference on Industrial Informatics, INDIN'19, Industrial Applications of Artificial Intelligence (2019).

2. Revision history

2.1 Contributing and reviewing partners

Contributions	Reviews	Participants	Representing partner
X	X	Géza Kulcsár	IQL
	X	László Gáti	IQL
	X	Federico Montori	IUNET
	X	Marek Tatara	DAC
	X	Barbara Jung	Expleo
X	X	Hans Ehm, Patrick Moder, Nour Ramzy	IFAG
X	X	Felix Larrinaga, Javier Cuenca	MGEP
X	X	Önder Babur	TU/e
	X	George Suciu, Romulus Chevereșan, Sorin MIhala	BEIA

X	X	Tomas Vojnar	BUT
X	X	Kjell Bengtsson	Jotne

2.2 Amendments

No.	Date	Version	Subject of Amendments	Author
1	2019-09-30	0.1	First draft	Géza Kulcsár
2	2019-10-22	0.2	Added contributions	Géza Kulcsár
3	2019-12-10	1.0	First Sanity Check	Federico Montori, Marek Tatara

2.3 Quality assurance

No	Date	Version	Approved by
1	2019-12-10	1.0	Jerker Delsing