Technical Report

Data Exchange Logistics in Engineering Networks Exploiting Automated Data Integration

Arndt Lüder[2]
Johanna-Lisa Pauly[2]
Felix Rinker[1,3]
Stefan Biffl[1]

[1]TU Wien, Information Systems Engineering, Information and Software Engineering, Quality Software Engineering, Vienna, Austria
<firstname>.<lastname>@tuwien.ac.at

[2]Otto-v.Guericke Universität Magdeburg, Magdeburg, Germany
<firstname>.<lastname>@ovgu.de

[3] Christian Doppler Laboratory for Security and Quality Improvement in the Production System Life Cycle, Information and Systems Engineering, TU Wien, Vienna, Austria
<firstname>.<lastname>@tuwien.ac.at

Technical Report No. CDL-SQI 2019-11
Issued: April 2019

Citation: A. Lüder., J-L. Pauly, F. Rinker, S. Biffl: „Data Exchange Logistics in Engineering Networks Exploiting Automated Data Integration“, Technical Report CDL-SQI 2019-11, TU Wien, Vienna, Austria, April 2019.

Data Exchange Logistics in Engineering Networks Exploiting Automated Data Integration

Arndt Lüder, Johanna-Lisa Pauly
Otto-v.-Guericke University, FMB, IAF,
Magdeburg, Germany
[arndt.lueder| johanna-lisa.pauly]@ovgu.de

Felix Rinker[1,2], Stefan Biffl[1]
TU Wien, [1]Inst. of Information Systems Eng., [2]CDL-SQI
Vienna, Austria
[felix.rinker|stefan.biffl]@tuwien.ac.at

***Abstract*—Data exchange and integration in an engineering network are challenges that each engineering organization has to solve. Most practitioners have realized that these challenges require detailed consideration and involvement of the engineering disciplines involved. However, these engineering disciplines usually do not speak the same language. Thus, a data exchange logistics is required overseeing the complete engineering network and ensuring proper data transformation between the disciplines. This data logistics requires automated mapping and integration of engineering data, following the VDI Guideline 3695. In this paper, we introduce a methodology to set up such a data logistics and envision a new role, the engineering data curator, responsible for the necessary knowledge management within this methodology. Following the technical background of the authors, the methodology is presented and validated in a proof of concept using AutomationML.**

Keywords—Production system engineering, Engineering network, engineering data exchange, data logistics, AutomationML

I. Introduction

Engineering of production systems is a complex and multi-disciplinary process [1], in which engineers take engineering decisions based on appropriate knowledge, input data, and assistance tools that generate output data [2]. These engineering actions form a network, linked by data exchange.

Unfortunately, several challenges complicate the data exchange within these engineering networks [3], challenges related to change, completeness, and consistency management of the involved engineering data, often due to engineering habits of the involved engineers [4].

Here, the term *engineering habit* refers to the usual bordering and execution condition of an engineering activity [5]. The habit is based on the usually applied engineering tools, information models, engineering methods and procedures the executing engineer is accustomed to: the engineer follows the usual education and training of the engineering discipline the engineer belongs to. Thus, application of tools and knowledge aims at supporting the engineering efficiency and quality of a specific discipline and is hard to change in an engineering organization.

In contrast to this one-dimensional view within one engineering discipline, the data exchange among engineering activities (and by this, disciplines) usually requires a multi-dimensional view of multi-model engineering [1][2][4].

Hence, considering engineering networks, we can find, in general, two sets of data models: data models applied within one engineering discipline and data models applied for data exchange between engineering disciplines [6]. Data models used within one discipline are well evaluated and tailored to the needs of this discipline [7]. These data models usually follow domain-specific needs and tools. Data models applied for data exchange between engineering disciplines have been evaluated in less detail. In most cases, these data models have been developed following the needs of data exchange between the involved tools and are intended to follow international standards [8]. Main requirement for the development of these two sets of data models within one engineering network is the coupling of the local discipline-specific data models in a way that allows mapping them onto each other and converting data content between these data models [4].

Fig. 1 illustrates how engineering networks can be considered as a network of two (not necessarily disjoint) sets of nodes: *data sources* (Fig. 1, tag 1) and *data sinks* (tag 2), similar to [21][34]. These nodes represent discipline-specific engineering activities exploiting discipline-specific data models. The *data logistics* (tag 3) shall ensure the data exchange between source and sink nodes, including necessary transport, transformation, selection and integration of engineering data using one or more multi-discipline data models.

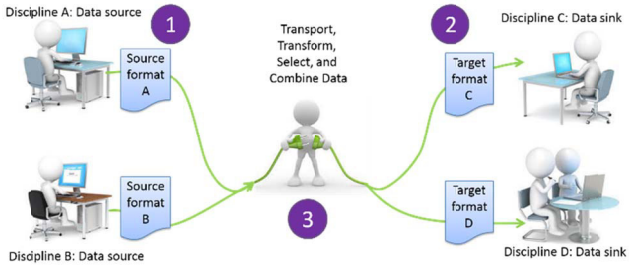

Fig. 2. Data logistics in engineering network, based on [21][34].

As discussed by Drath *et al.* [2], the engineering-tool-related implementation of such an engineering data logistics faces several challenges. Hence, within research and development, researchers have discussed the use of a centralized data exchange platform like [19] and [20].

A precondition for a *centralized data logistics platform* is a common semantics, i.e., the data models applied within the platform shall describe identical facts always in a uniform way and this description semantics has to be possible to map local data models to each other (see the topic *Description languages* in the VDI Guideline 3695, Part 3 [9]).

Unfortunately, a look at practical engineering networks shows limitations, coming from engineering habits within the walls of a discipline, to hinder the evaluation, development, and selection of data models for data exchange within such an intended data logistics. The results include unnecessarily high effort for and duration of data integration, and risks for the data quality of exchanged data due to error-prone manual work for extracting relevant data from engineering artifacts that are not compatible.

To mitigate this problem, a detailed consideration of the *discipline-specific data models* and their integration with a *discipline-connecting data model* is required. This is a new activity calling for a new role within engineering organizations, the *engineering data curator*. Following the VDI Guideline 3695, this role shall have an overview on the complete engineering chain to support its migration.

In this paper, we build on and extend [34] to contribute a methodology for the incremental realization of a data logistics, and discuss (a) a centralized data logistics architecture, (b) a data logistics meta-model, and (c) a data model for data logistics based on AutomationML. The *centralized data logistics architecture* is based on the analysis of *discipline-specific data models* and on the development of an associated *discipline-connecting data model*. Starting point of this methodology is the development of an appropriate *data logistics meta-model* reflecting the common concepts of the involved different engineering disciplines. This *data logistics meta-model* enables the iterative representation of engineering data to be exchanged with their data sources, data sinks, and discipline-connecting dependencies.

For reasons of clarity and applicability, this paper discusses a *data model for data logistics based on the AutomationML standard* as technical foundation, illustrated in a use case with data on a drive chain with the disciplines functional, mechanical, and electrical engineering. We are aware of the possible limitation regarding the generality of the presented approach. Nevertheless, following the current developments in the *Industry 4.0* and *Industrial Internet of Things* (IIoT) fields, AutomationML seems to be widely accepted for data exchange within production system engineering, justifying the use of AutomationML for a proof of concept [31].

To reach the intended goal, the paper is structured as follows. Section II motivates the research questions and approach. Section III summarizes the state of the art related to the research questions. Section IV represents the proposed architecture of an information logistics for engineering networks. Section V introduces a meta-model to conceptualize the information exchanged in this engineering network. Section VI derives from this meta-model an AutomationML-based data model for automating data integration in an engineering network. Section VII validates the modelling methodology. Section VIII discusses the results. Section IX concludes and proposes future research work.

II. Research questions

As mentioned above, the aim of this paper is to solve the problem of engineering data propagation within an engineering network that is characterized by discipline-specific engineering habits that are hard to change. This problem calls for a data logistics reflecting the distinction between engineering data sources and sinks that have to exchange engineering data. This section motivates the research questions (RQs) and the research approach.

It is supposed that the involved engineering tools as well as the work of the involved engineers shall remain unchanged. Thus, the data logistics shall be responsible for transport, transformation, selection, and integration of engineering data. In this paper, we discuss a methodology for designing foundations for a data logistics in three parts: 1. Data logistics architecture, 2. Data logistics meta-model, and 3. Data model in a technical foundation for prototypical realization.

First, it is necessary to envision the architecture of such a data logistics and its use, leading to RQ1.

RQ1: What data logistics architecture within an engineering network can reflect engineering data sources, sinks, and the information, for propagation among sources and sinks?

To answer RQ1, we use the engineering data flow as a starting point. We characterize the elements required to form the data logistics (see Fig. 2) and the behaviour these elements have to show in order to form the overall behaviour of the engineering data logistics.

A main part of this behaviour is the propagation of engineering data between involved engineering tools. As these tools use their own data models, the data logistics has to map these tool-specific data models, based on an appropriate understanding and description of all data models involved. Therefore, RQ2 has to be solved.

RQ2: What data logistics meta-model can represent in an engineering network the discipline-specific data models and the discipline-connecting data model?

To answer this question, we discuss the information types covered in the individual engineering disciplines as well as their interrelations. Based on these information types, we present a data logistics meta-model of interlinked engineering information within multi-disciplinary engineering networks.

Using this meta-model, we can represent the data models within the engineering data logistic. However, for automated mapping of the individual engineering information from data sources to data sinks, RQ3 has to be considered.

RQ3: What AutomationML data model can represent the information required for a data logistics application?

To enable this automated mapping, a representation of detailed data source model element positions, detailed data sink model element positions and data propagation information between these model elements need to be integrated within the overall data model. Therefore, we discuss mechanisms in AutomationML standard to make this

information representable as foundation for an executable data logistics application with automated data integration.

III. State of the art

This section summarizes the state of the art on engineering networks, on the formal representation of data models, and on the data exchange standard AutomationML.

A. Engineering networks

Production systems are a special kind of technical systems and their engineering process has been well evaluated [1]. In general, the engineering process can be considered as a network of engineering decisions that are related to each other. An engineering decision requires input information that gets processed by skilled engineers, who use appropriate tool support to process new engineering information based on the taken engineering decisions [5].

The sequence and granularity of engineering decisions depend on the application case. Following the different industries, production system complexity, and control technologies applied in the production systems, different engineering process guidelines have been developed and standardized [12]. They range from general guidelines as the VDI Guidelines 2221 and 2206 over modelling-oriented guidelines like SysMod to industry-driven guidelines like the STEP and AutomationML reference engineering processes.

In general, there are various engineering steps involved in production system engineering, where some engineering steps can be identified in nearly each engineering network [14], resulting in 30 and more individual engineering steps [13].

Production system engineering networks have been increasingly influenced by and interlinked with other engineering networks. First, product engineering needs to be named [15] as it originates the main requirements to production system engineering and is the main cause of complexity [17]. In addition, the changing automation structures following recent trends like *Industrie 4.0* are increasingly influencing the integration and use of involved manufacturing system components [16].

B. Formal representation of data models

Within the engineering of technical systems, a wide variety of data models, data formats, and modelling means are applied [23]. Therefore, we review main representations showing the historical development of formal representations, coming from domains such as Software Engineering, Model-Driven Engineering, and Semantic Web.

In Software Engineering, the Entity–Relationship model (ER model) [24] is an abstract data model to describe the data structure needed e.g., for the persistence in a relational database [25]. In Model Driven Engineering, the Unified Modelling Language (UML) [26] is the conceptual modelling approach to describe the structure of system and its data or information as domain model. It has been adapted for example to systems engineering as SysML **Fehler! Verweisquelle konnte nicht gefunden werden.**. In Java, the Eclipse Modelling Framework (EMF) [27] is commonly used. It utilizes the Ecore meta-model to describe the domain model. In the Semantic Web, the Resource Description Framework (RDF) [28] is used, in which the RDF data model is described utilizing the RDF Schema. The Web Ontology Language (OWL) [29] is an extension of the RDF Schema and is used to formally describe ontologies. These approaches emphasize the integration integration discipline-specific models based on meta-modelling.

C. Data exchange based on the AutomationML standard

As shown in [18], AutomationML can be applied for modelling engineering data for the data exchange along the complete engineering chain of production systems. In [19] and [20] engineering data logistics based on a centralized engineering data storage using AutomationML are discussed, while in [21] a more decentralized data logistics based on AutomationML is considered.

The AutomationML data format is developed by AutomationML e.V. as an open, vendor-independent, and XML-based data exchange format, which enables a domain and company crossing transfer of engineering data of production systems in a heterogeneous engineering network. It enables the modeling of hierarchies of physical and logical production system components as data objects encapsulating all relevant aspects. Typical objects in plant automation comprise information on topology, geometry, kinematics, and logic, whereas logic comprises sequencing, behavior, and control information. AutomationML is currently standardized within the IEC standard series IEC 62714.

Working horse of AutomationML is the topology description architecture exploiting the XML data format CAEX. This data format enables an object-related approach, where object semantics (i.e., the identification of modelling concepts) are specified using roles and the semanics of relation between objects (i.e., relations between modelling concepts) are indicated by interfaces. Both roles and interfaces are defined and collected in role resp. interface class libraries. In addition, reuseable objects acting as modelling templates are provided by classes of system objects (named system unit classes), defined and collected in system unit class libraries. Finally, the individual objects are modeled in an instance hierarchy as a hierarchy of internal elements, referencing system unit classes they are derived from, role classes defining their semantics, and interfaces used to interlink objects with each other or with externally stored information (e.g., COLLADA or PLCopenXML files). For details on this structure, the authors refer to the AutomationML whitepapers at www.automationml.org.

IV. Centralized Data Logistics architecture

This section proposes a centralized data logistics to solve RQ1. The *centralized data logistics* is based on the idea of separating the concerns of the involved engineering disciplines and the data logistics between them. Hence, we assume the data logistics architecture to contain two main

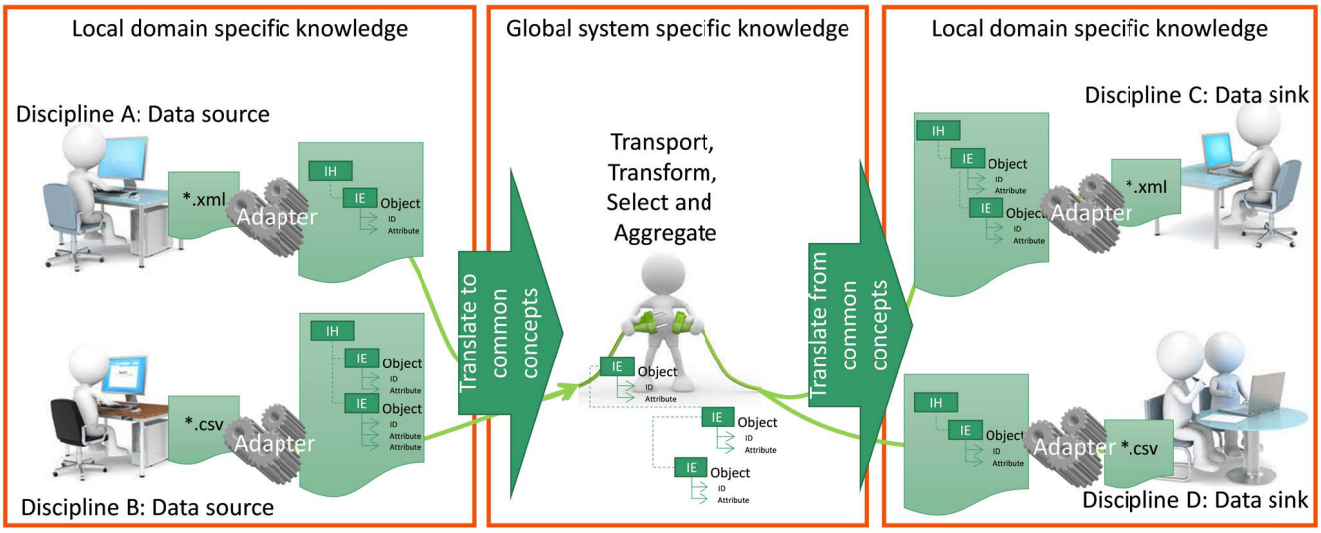

Fig. 2. General architecture of engineering data logistics in a network of engineering data sourcen and sinks.

parts: flexible *tool adapters* (enabling the transformation of discipline-specific data models to the discipline-connecting data model of the data logistics) and a *centralized data logistics information system* (realizing data integration, selection, and propagation).

The main aim of the centralized data logistics is to provide and maintain an integrated, consistent, and complete model of the production system to be engineered over the complete lifecycle of production system engineering. Therefore, this overall model shall be detailed step by step within the engineering phase. This data logistics shall implement the following functionalities:

- *Discipline integration and management* shall provide methods for the representation of the engineering information of all involved engineering disciplines in a consistent way, for the representation of dependencies between this information, and for their propagation across disciplines.
- *Change management* shall provide and maintain information on model versions and releases, also covering information on successfully finished engineering phases and on system states.
- *Completeness management* shall provide information on missing engineering information required for an engineering activity.
- *Consistency management* shall realize the evaluation of consistency rules integrated in the overall engineering data model, including the identification of possible sources of failure.

To enable these functions, the centralized data logistics requires capabilities to import, manage, analyze, and export data and to support different views on the engineering data. This view support shall enable the reflection of different engineering disciplines and different engineering tools involved in the engineering network.

At this point, it shall be emphasized that a centralized data logistic does not need to be realized as a central data storage. The only requirement is a data storing architecture that contains a complete navigable and analyzable data model with a central entrance point.

Flexible tool adapters as intelligent tool interfaces (or interfaces to other data processing systems, like model generators) are intended to support the data transfer between engineering tools and the centralized data logistics. The tool adapters are responsible for the necessary transformation of the tool-dependent data models to the data model applied within the centralized data logistics and vice versa.

The flexible tool adapters enable decoupling the different engineering habits of involved engineers from the centralized data logistics, and extensions and migration scenarios by sequentially designing and realizing tool adapters, depending on the intended application cases.

Fig. 2 depicts the resulting architecture of the data logistics.

The use of the data logistics is intended as follows. The engineering data of a source tool is exported following the discipline-/tool-specific data model. Based on this export, the relevant data has to be selected, extracted, and transformed into the data model of the data logistics within the flexible adapter attached to this tool. After importing the data into the data logistics, the data may be merged and aggregated with engineering data from further data sources to form an integrated model of the complete system with all relevant engineering data.

To import the data to the tool used by the data sink, the reverse process applies. First, the relevant data is exported from the data logistics. Second, relevant data is selected following the internal data model of the sink tool. Third, the data is transformed to an import data format of the sink tool.

Together, these elements form the data logistics architecture as foundation for defining the data in the

discipline-specific views and the connections between the disciplines for automating the data integration.

V. Data Logistics Meta-Model

As visible from the last section, the data model applied in the data logistics needs to reflect the data models of the involved individual engineering disciplines. To enable the development of such a data model for a data logistics and to solve RQ2, this section will discuss the data logistics meta-model defining the necessary information types required to enable modelling the engineering information that shall be utilized for the automated integration of data coming from several engineering disciplines. This discussion needs to stay on an abstract level by defining a data logistics meta-model that can cover all possible disciplines involved in an engineering network. Consequently, the meta-model is independent from any discipline-specific conceptualization.

Starting point of the meta-model development is the observation that, independent of special disciplines, each engineer is used to think in conceptualized objects that can be related to one another. An example is the illustrating use case of *designing a drive chain*. Mechanical engineers discuss static or dynamic objects that are statically or dynamically attached to each other exchanging forces, like drives, gearboxes, and load used in a drive chain. Electrical engineers know devices and wires exchanging electrical energy, like a drives, drive controllers, and energy sources in the drive chain. Finally, the automation engineer discusses control devices (i.e., sensors, actuators, and controllers) exchanging control information, like PLCs, drive controllers, and rotary encoders connected in drive chains.

Thus, it becomes apparent that engineers discuss concepts that usually occur in several disciplines (Fig. 3, tag 1), while keeping their special view on these concepts (tag 2). These discipline-specific views contain properties (tag 3) and relations (tag 4) between the concepts that are typical for the discipline, like size, mass, and material for the mechanical engineer, currency and voltage for the electrical engineer, and control state, like current speed, for the automation engineer, all related to the same drive.

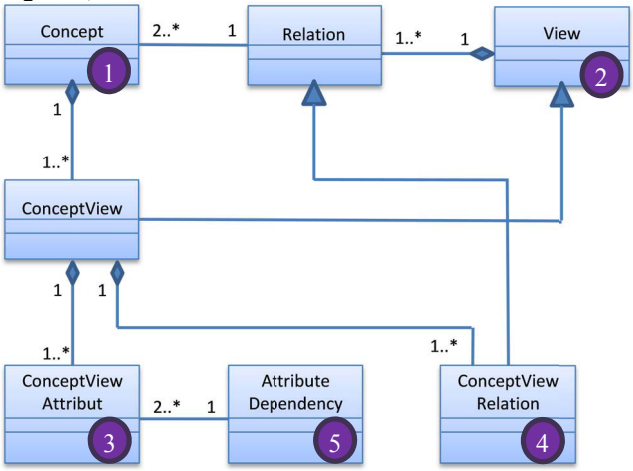

Fig. 3. Data logistics – conceptual meta-model, based on [34].

Finally, the discipline-specific properties, which can be regarded as attributes within the views on the concepts, can depend on each other. These dependencies (tag 5) can be based on natural laws, best practices, regulations, and others, and they usually cross the disciplines. For example, the drive speed in the automation view on a drive depends on the size of the drive and the incoming currency and voltage of mechanical and electrical view.

As a result, Fig. 3 depicts the data logistics meta-model that can be exploited to identify for each engineering discipline the relevant concepts with their attributes and relations, and to combine them following the different views on same things.

VI. Data Logistics Models Based on AutomationML

To make the meta-model in Fig. 3 applicable and to solve RQ3, this section will discuss the AutomationML-based modelling of all required information to enable an automated data integration in a data logistics within an engineering network. In addition, this modelling will reflect the required representation of individual data source, data sink, and propagation relations of the involved engineering information.

As described in Section III.C, AutomationML utilizes the concept of roles to identify concepts, interfaces to interrelate these concepts, and system unit classes to form modelling templates exploiting roles and interfaces. These elements shall be also the foundation of modelling the mapping between the different engineering-discipline-driven data models.

As discussed in [21], we assume the different discipline-specific data formats to be represented by AutomationML and, if necessary, appropriate mappers can be implemented. Therefore, discipline-specific AutomationML dialects can be developed and implemented that are named AML-2 data models. The integration of these discipline specific data models will result in another AutomationML data model named AML-1 data model.

First step of the modelling is the *identification of the different views*, i.e., engineering disciplines with data models applied within the engineering discipline and data models applied for data exchange between engineering disciplines. For all of them data model indicating role classes, i.e., view role classes, are defined. They are accompanied by interface classes required to model the hierarchy relations within the view as it is assumed, that hierarchies within different views can possibly be contradictory.

Next, for each view, the *view-related concepts are identified* and modelled by view concept role classes. They are completed by attributes modelling the properties relevant for the concept in the considered view and interfaces used to represent the relations in the view. These parts are referenced in green in Fig. 4.

To reach the *AML-2 data model for each view*, appropriate *system unit classes* are developed by using the view related roles. For each relevant concept, a system unit class is

developed referencing the related role as supported role class and containing the attributes of this role and, if necessary, the interfaces of this view including the hierarchy interface. This part is referenced in orange in Fig. 4.

To represent the individual data source and data sink information of the individual attributes, the *RefSemantic* element of each AutomationML attribute is used, following [32]. It is assumed that the value of the XML attribute *CorrespondingAttributePath* of the *RefSemantic* element shall be assembled following the URI structure defined in RFC3986 [33], exploiting the elements scheme, path, query, and fragment. The scheme shall identify the source of semantics, like a tool or discipline. Next, the path element is delimited from the source by "://" and indicates the location of the semantics definition depending on the scheme like a name of a file type used. After an "?" as delimiter, the query can follow giving an additional interpretation to the named semantics definition. Finally, after a "#" delimiter, the object identifier of the semantics definition in the semantics definition files is named. An example using the *RefSemantic* element is the string "companycsv://cylinderlist?id#4" indicating that in a company-specific *csv* file named *cylinderlist*, the fourth column contains the attribute of interest.

Fig. 4 depicts the data logistics modelling rules linked to the meta-model (see numbered tags in violet circles).

To create the *AML-1 data model* (tag 1), AML-2 data models (tags 2) are combined in a *common system unit class library* (tag 3). For each concept, an appropriate system unit class is developed containing InternalElements created by instantiating the related AML-2 data model system unit classes of the views relevant for the concept (see tag 1, red box).

The dependencies between attributes of the different InternalElements of the concept related system unit class are modelled. Thereby, the propagation of engineering information between disciplines is represented by using the *RefSemantic* elements of the AutomationML attributes. To indicate that an attribute of one discipline is linked with an attribute of another discipline the string "companyaml://AML1Project#ElementID@AttributName" can enables the data logistics to copy the attribute value.

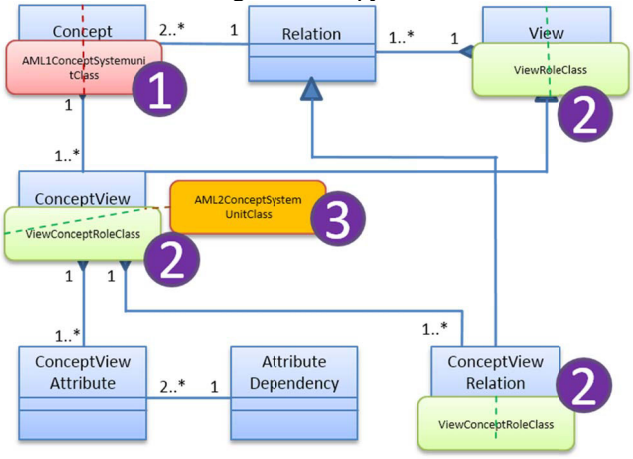

Fig. 4. AutomationML representation of the conceptual data logistics meta-model, based on [34].

Together, these AutomationML data model elementss enable the automated data integration for implementing the data logistics.

VII. Validation

In order to validate the proposed approach with respect to the research questions and to demonstrate the feasibility to automatically derive and process the described AutomationML-based modelling of data integration in an engineering logistics, three prototypical software systems have been developed in the Christian Doppler Research Lab SQI[1] [22].

The *AML12 generator prototype* is capable of generating the AML-1 and AML-2 data models that are exploitable in the engineering data logistics. This prototype has to be provided with an abstract representation of the required information, identified in the data logistics meta-model in Section V, as a YAML file. Fig. 5 shows an illustrative part of a YAML file that can be regarded as validation model in the use case of a drive chain for attributes in four views (tags 1 to 4), regarding function, mechanical, electrical, and automation engineering. The YAML file is user-defined input to a tool that generates AML1 and AML-2 data models that configure the data logistics. The generating tool uses an AutomationML software framework developed at Otto-von-Guericke University to create the AutomationML model.

```
- concept: Project
  views:
  - view: Function
    derivedFrom: AutomationMLBaseRoleClassLib/AutomationMLBaseRole/Structure/ResourceStructure
    attributes:
    - viewAttribute: MUK
      attributeDataType: xs:string
      isIdentifier: true
    - viewAttribute: Description
      attributeDataType: xs:string
  - view: Mechanic
    derivedFrom: AutomationMLBaseRoleClassLib/AutomationMLBaseRole/Product
    attributes:
    - viewAttribute: MechanicID
      attributeDataType: xs:string
      isIdentifier: true
  - view: Electric
    derivedFrom: AutomationMLComponentStandardRCL/AutomationComponent
    attributes:
    - viewAttribute: IdentCode
      attributeDataType: xs:string
      isIdentifier: true
  - view: Automation
    existingRoles:
    - existingRole: AutomationProjectConfigurationRoleClassLib/AutomationProject
    attributes:
    - viewAttribute: AutomationID
      attributeDataType: string
      isIdentifier: true
```

1
2
3
4

Fig. 5. Validation model in YAML for the use case drive chain – functional, mechanical, electrical, and automation engineering views.

Fig. 6 shows illustrative domain-specific AML-2 models for functional, mechanical, and electrical engineering objects (tag 1) and interfaces (tag 2) on the left-hand side and the corresponding domain-integrating AML-1 model for the same disciplines on the right-hand side (tag 3). The generator ensures the consistent and correct declaration of the discipline-specific AML-2 models (tags 1, 2) with the discipline-connecting AML-1 model (tag 3).

The *CSVtoAML transformer* prototype implements the flexible adapters applicable to map *csv* files and *xml* files to AutomationML files following AML-2 data models. This

[1] Christian Doppler Laboratory for Security and Quality Improvement in the Production System Lifecycle (CDL-SQI); https://www.sqi.at

transformer prototype assumes that the individual lines of the csv files or individual tags in an *xml* file represent objects of one engineering discipline following the same concept. Accordingly, in the development of the AML-2 model, the lines and tags are conceptualized by appropriate role classes and system unit classes. The attributes of these role classes and system unit classes are equivalent to the information encoded within the columns of the *csv* files of sub-tags and attributes. To link them to AutomationML attributes the *refSemantic* is applied as described above.

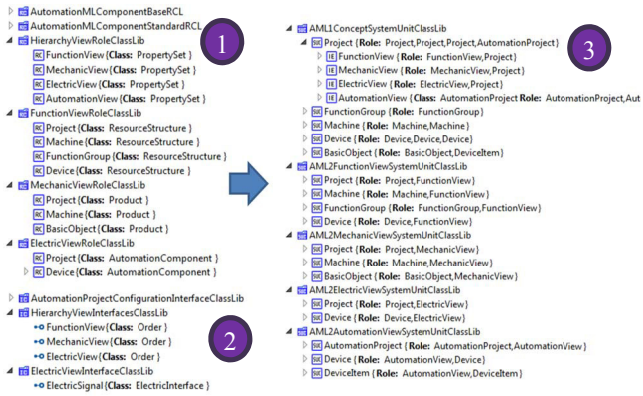

Fig. 6. AutomationML model examples for the use case drive chain – functional, mechanical, and electrical engineering views., based on [34].

The *data logistics* prototype implements the data logistics information system. This prototype builds on the fact that the domain-specific AML-2 data models are an integral part of the domain-crossing AML-1 data model. To load data source-tool-related AML-2 data models into the data logistics AML-1 model elements are created and/or adapted (if existing) based on role classes assigned to AML-2 model elements. The same holds for writing sink-tool-related AML-2 data models that are created by shaping the AML-1 data models.

VIII. Discussion

The presented approach can be applied in different ways.

First, the approach can assist the *engineering data curator* in their task of identifying information to be exchanged in engineering networks. Data exchange curators are assisted in the formal representation of data exchange capabilities and needs within engineering networks.

Second, the data integration for data logistics can be automated over different involved engineering disciplines without requiring adapting the engineering habits of the involved engineers.

Third, the data logistics can be implemented incrementally, forming a migration path from existing engineering networks exploiting bilateral or even *Excel* or *pdf* based data exchange towards a more controlled, consistency ensuring, and data quality ensuring engineering network, as envisioned in [21].

Main drawback of this approach is the need for an *engineering data curator* as a new role within an engineering network. This role has to have an overview about the different involved engineering disciplines and shall have a systems engineering background. This shall enable the engineering data curator to understand the needs and capabilities of the involved engineering disciplines related to engineering information exchange, and to appropriately identify common concepts and dependencies between information within the different views on the same concept. Nevertheless, this new role also opens up new business opportunities as envisioned in [6].

IX. Conclusion

This paper has presented a first attempt towards the design and implementation of a methodology for modelling and using all necessary information required for automatic integration of engineering information within an engineering data logistics of an engineering network. This methodology is facilitating the easy implementation of an engineering data logistics within engineering networks reducing effort and duration of data integration and limiting risks for the data quality of exchanged data.

Therefore, the following research questions have been answered. First, RQ1 related to an general architecture for an engineering data logistic has been addressed. With a system consisting of flexible tool adapters and a centralized data logistics, a system architecture has been presented and evaluated successfully.

Second, RQ2 related to the identification of necessary information required to enable the consistent modelling of data models applied within one engineering discipline and data models applied for data exchange between engineering disciplines has been discussed. Therefore, a data logistics meta-model has been established aggregating the different views coming from different engineering disciplines in common concepts and their properties and relations.

Third, RQ3 related to the formal representation of this meta-model to automate its processing has been addressed by designing an AutomationML representation that enables developing AutomationML-based data logistics applications.

Finally, an initial validation has been sketched that follows the ideas of VDI Guideline 3695 for the automated mapping of data models. This validation has shown that the presented approach is feasible in the use case drive chain and should be explored in a wider range of application contexts.

Future work. The authors of this paper are currently improving the implementations of the proof of concept software systems within the Christian Doppler Laboratory SQI. In cooperation with industry partners, a decentralized data logistics is designed and prototypically implemented and will be improved and applied to test cases with respect to the engineering of steel production systems and mounting systems for automotive industry. Special emphasis will be put on the assistance of the work of the engineering data

curator as well as on the representation of dependencies between engineering data.

ACKNOWLEDGMENT

The financial support by the Christian Doppler Research Association, the Austrian Federal Ministry for Digital and Economic Affairs and the National Foundation for Research, Technology and Development is gratefully acknowledged.

REFERENCES

[1] S. Biffl, D. Gerhard, A. Lüder: Multi-Disciplinary Engineering for Cyber-Physical Production Systems, Springer, 2017.

[2] R. Drath, M. Barth, A. Fay: Offenheitsmetrik für Engineering-Werkzeuge: Die Fähigkeit zur Interoperabilität bewerten, atp-edition, Heft 9/2012, S. 46-55.

[3] S. Biffl., F. Ekaputra A. Lüder J.-L. Pauly F. Rinker, L. Waltersdorfer, D. Winkler.: Experiences with Technical Debt in Parallel Multi-Disciplinary Systems Engineering, Christian Doppler Laboratory for Security and Quality Improvement in the Production System Lifecycle, Technical Report, CDL-SQI 2019-02, http://qse.ifs.tuwien.ac.at/wp-content/uploads/CDLSQI-2019-02.pdf, 2019.

[4] A. Lüder, J.-L. Pauly, K. Kirchheim: Multi-Disciplinary Engineering of Production Systems – Challenges for Quality of Control Software, Software Quality: The Complexity and Challenges of Software Engineering and Software Quality in the Cloud, Springer International Publishing, S. 3-13, Proceedings of 11th International Conference, SWQD 2019, Vienna, Austria, January 15-18, 2019.

[5] T. Schäffler, M. Foehr, A. Lüder, K. Supke: Engineering Process Evaluation - Evaluation of the impact of internationalisation decisions on the efficiency and quality of engineering processes, 22nd IEEE International Symposium on Industrial Electronics (ISIE) Taipei, Taiwan, Mai 2013, Proceedings.

[6] U. Löwen, A. Schertl, S. Runde, M. Schleipen, F. El Sakka, A. Fay: Zusätzliche Wertschöpfung mit digitalem Modell – Neue Rollen im Anlagen-Lebenszyklus. In: VDI – Verein Deutscher Ingenieure e.V. (Hrsg.): AUTOMATION 2018, 03.-04.07.2018, Baden-Baden.

[7] Verein Deutscher Ingenieure (VDI): VDI Guideline 3690 - XML in der Automation, 3 Parts, Beuth publisher, 2012-2017.

[8] A. Lüder, N. Schmidt, R. Drath: Standardized information exchange within production system engineering, In: Multi-disciplinary engineering for cyber-physical production systems: data models and software solutions for handling complex engineering projects - Cham: Springer-Verlag, S. 235-257, 2017

[9] Verein Deutscher Ingenieure (VDI): VDI Guideline 3695 - Engineering von Anlagen - Evaluieren und Optimieren des Engineerings, 5 Parts, Beuth publisher, 2010-2019.

[10] Z. Li, Paris Avgeriou, and Peng Liang. A systematic mapping study on technical debt and its management. Journal of Systems and Software, 101:193–220, 2015.

[11] B. Vogel-Heuser and Susanne R¨osch. Applicability of technical debt as a concept to understand obstacles for evolution of automated production systems. In 2015 IEEE International Conference on Systems, Man, and Cybernetics, pages 127–132. IEEE, 2015.

[12] A. Lüder, M. Foehr, L. Hundt, M. Hoffmann, Y. Langer, St. Frank: Aggregation of engineering processes regarding the mechatronic approach, 16th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA 2011), Toulouse, France, September 2011, Proceedings-CD.

[13] K. Hell: Methoden der projektübergreifenden Wiederverwendung im Anlagenentwurf, PhD Thesis, Otto-v.-Guericke University, Magdeburg, Germany, March 2018.

[14] A. Strahilow, H. Hämmerle: Engineering Workflow and Software Tool Chains of Automated Production Systems, In: Multi-disciplinary engineering for cyber-physical production systems: data models and software solutions for handling complex engineering projects, Springer Publisher , pp 207-234, 2017.

[15] K. Paetzold: Product and Systems Engineering/CA* Tool Chains, In: Multi-disciplinary engineering for cyber-physical production systems: data models and software solutions for handling complex engineering projects, Springer Publisher , pp 27-62

[16] B. Vogel-Heuser, T. Bauernhansl, M. ten Hompel (Hrsg.): Handbuch Industrie 4.0, Bände 1-4, VDI Springer Reference, 2017.

[17] M.R. Mehr, S. MirRashed, M. Mißler-Behr, A. Lüder: Analyzing the causes and effects of complexity on different levels of automobile manufacturing systems, 21th IEEE Conference on Emerging Technologies and Factory Automation (ETFA), Sep. 2016, Berlin, Germany, Proceedings.

[18] A. Lüder , N. Schmidt: AutomationML in a Nutshell, in B. Vogel-Heuser, T. Bauernhansl, M. ten Hompel (Ed.): Handbuch Industrie 4.0, Bände 1-4, VDI Springer Reference, 2017.

[19] F. Himmler, M. Amberg: Data Integration Framework for Heterogeneous System Landscapes within the Digital Factory Domain, Procedia Engineering, Volume 69, 2014, pp. 1138-1143.

[20] R. Mordinyi, S. Biffl: Versioning in cyber-physical production system engineering: best-practice and research agenda, Proceedings of the First International Workshop on Software Engineering for Smart Cyber-Physical Systems, Florence, Italy, pp. 44-47, 2015.

[21] A. Lüder1, J-L. Pauly, K. Kirchheim, F. Rinker, S. Biffl: Migration to AutomationML based Tool Chains – incrementally overcoming Engineering Network Challenges, 4th AutomationML user conference, Oct. 2018, Gothenburg, Sweden, Proceedings, www.automationml.org.

[22] S. Biffl, M. Eckhart, A. Lüder, T. Müller, F. Rinker, and D. Winkler. 2018. "Data Interface for Coil Car Simulation (Case Study) Part II - Detailed Data and Process Models." Technical report CDL-SQI-M2-TR03. TU Wien.

[23] A. Lüder, N. Schmidt, K. Hell, H. Röpke, J. Zawisza: Description means for information artifacts throughout the life cycle of CPPS, In: Multi-disciplinary engineering for cyber-physical production systems: data models and software solutions for handling complex engineering projects - Cham: Springer-Verlag, S. 169-183, 2017

[24] P.P.S. Chen: The entity-relationship model—toward a unified view of data. ACM Transactions on Database Systems (TODS), 1(1), 1976, pp.9-36.

[25] A. Olivé: Conceptual modeling of information systems. Springer Science & Business Media, 2007.

[26] J. Rumbaugh, I. Jacobson, G. Booch: Unified modeling language reference manual, the. Pearson Higher Education, 2004.

[27] D. Steinberg, F. Budinsky, E. Merks, M. Paternostro: EMF: eclipse modeling framework. Pearson Education, 2008.

[28] G. Klyne, J.J. Carroll: Resource description framework (RDF): Concepts and abstract syntax, 2006.

[29] D.L. McGuinness, F. Van Harmelen: OWL web ontology language overview. W3C recommendation, 10(10), 2004.

[30] E. Brusa, A. Calà, D. Ferretto: Systems engineering and its application to industrial product development, Springer International Publishing, 2018.

[31] Plattform Industrie 4.0: Details of the Asset Administration Shell - Part 1 - The exchange of information between partners in the value chain of Industrie 4.0, https://www.plattform-i40.de/I40/Redaktion/DE/ Downloads/Publikation/2018-verwaltungsschale-im-detail.html, April 2019.

[32] AutomationML association: White Paper - AutomationML and eCl@ss integration, Dec. 2017, https://www.automationml.org/o.red/uploads/-dateien/1513936451-WP_AutomationML_and_eClass_integration_-V1.0.1.pdf.

[33] Network working group: RFC3986 Uniform Resource Identifier (URI): Generic Syntax, Jan 2005.

[34] A. Lüder, K. Kirchheim, J-L. Pauly, S. Biffl., F. Rinker, L. Waltersdorfer: „ Supporting the Data Model Integrator in an Engineering Network by Automating Data Integration", Technical Report CDL-SQI 2019-07, TU Wien, Vienna, Austria, March 2019; web link: http://qse.ifs.tuwien.ac.at/wp-content/uploads/CDL-SQI-2019-07.pdf