A View on Model-Driven Vertical Integration: Alignment of Production Facility Models and Business Models

Bernhard Wally[1], Christian Huemer[1] and Alexandra Mazak[2]

Abstract— **Smart manufacturing requires deeply integrated IT systems in order to foster flexibility in the setup, rearrangement and use of attached manufacturing systems. In a vertical integration scenario, IT systems of different vendors might be in use and proprietary interfaces need to defined in order to allow the exchange of relevant information from one system to another. In this paper we present a model-driven approach for vertical integration of IT systems. It is based on the application of industry standards for the representation of hierarchy level specific system properties and an alignment of their key concepts in order to provide bridging functions for the transformation between the different systems.**

I. Introduction

Quite a few names have been coined for the concept of deeply integrated, networked manufacturing systems: industrial internet of things, cyber-physical production systems, digital production, smart manufacturing, or Industrie 4.0—just to name a few. They all share the common vision of the automation and individualization of the complete manufacturing process—from product description, over order processing and production to product delivery. To make this vision a reality, different business partners are required that execute specific processes and provide these capabilities as services. Abstractly speaking, two kinds of system integration are required: *horizontal integration* for the linking of systems on the same hierarchy level and for seamless communication between different parties and *vertical integration* for the integration within one partner, from the business floor to the shop floor.

Model-driven engineering (MDE) has developed a rich palette of tools and techniques for the description and manipulation of software models. Formalized cross-disciplinary engineering is supported by translating between the different engineering fields through common meta-metamodels and clearly specified sets of operations for model-to-model transformations, model validations, model querying, etc.

In this work, we will showcase the application of MDE techniques in the field of automated production systems (aPS) and their implications up to the business layer. With regards to "digital production", the externalizing of internal processes through services that can be queried becomes more and more important. Flexible automation systems that can be adapted much faster than it has been the case in the past require rapid adaptation of corresponding business models in order to provide up-to-date service descriptions.

II. Related Work

Since this paper is cross-disciplinary, we present the related work in several subsections, starting with information about the chosen metamodels and their features, followed by aligning the contributions of this paper with related work on *vertical integration*, *metamodel alignment*, and *model co-evolution*.

A. CAEX and AutomationML

Computer Aided Engineering Exchange (CAEX) is a data format that has been defined in the scope of IEC 62424:2008 and provides structures (i) for information exchange between Piping and Instrumentation Diagram (P&ID) tools and Process Control Engineering (PCE) related Computer Aided Engineering (CAE) tools, as well as (ii) for the representation of PCE requests in P&I diagrams [1]. CAEX is based on XML[1] and enables the metamodeling and modeling of e.g., the hierarchical architecture of a plant, including involved machines and controllers and their physical and logical connections.

IEC 62714 is based on CAEX and defines sets of role classes and interface classes with certain restrictions regarding their application [2], [3]. It is more commonly known as Automation Markup Language (AutomationML, AML), which is the term we will use in the remainder of this paper. AML defines an abstract interface class `ExternalDataConnector` which is used to reference external documents and elements therein. Two use cases of this external data connector have been defined so far in separate whitepapers: (i) `COLLADAInterface` specifies how external COLLADA[2] documents are referenced [4] and (ii) `PLCopenXMLInterface` defines how PLCopen[3] XML documents can be referenced from AML documents [6]. These whitepapers provide a rough guideline on the referencing and integration of external data into AML documents and serve as a starting point for the work presented here.

[1]Bernhard Wally and Christian Huemer are with the Business Informatics Group, Institute of Software Technology and Interactive Systems, TU Wien, 1040 Vienna, Austria `{wally, huemer}@big.tuwien.ac.at`.
[2]Alexandra Mazak is affiliated with the Christian Doppler Laboratory MINT at TU Wien funded by the Austrian Federal Ministry of Science, Research and Economy (BMWFW) `mazak@big.tuwien.ac.at`.

[1]cf. https://www.w3.org/TR/2008/REC-xml-20081126/

[2]COLLADA—Collaborative Design Activity: an XML based exchange format for 3D assets (cf. https://www.khronos.org/collada/).

[3]PLCopen is a vendor- and product-independent association active in industrial control (cf. http://www.plcopen.org/). PLCopen XML is a data exchange format for the storage of PLC program information according to IEC 61131-3 [5].

B. ISA-95 and B2MML

ISA-95 is a series of standards that addresses the integration of the enterprise domain with the manufacturing and control domains. It defines a set of object models for the exchanging of information between these domains—it provides a standard terminology and set of concepts for system integration [7]. The relevant part of ISA-95 for this work is part 2, as it is specified in IEC 62264-2:2013 [8].

Part 2 of ISA-95 specifies common objects and attributes, mainly by a set of commented UML[4] class diagrams, that can be roughly differentiated between (i) basic resources that depict the static definitions of an enterprise with regards to its production facilities (e.g., personnel, equipment, and material) and (ii) operations management information that resembles operational data (e.g., operations capabilities, schedules, and performance).

An XML serialization of ISA-95 has been defined in [9], the business to manufacturing markup language (B2MML). The current version of B2MML is compliant with the current version of ISA-95 and has been used in [10] to link ISA-95 information into AML models.

C. Resource-Event-Agent

Resource-Event-Agent (REA) was coined in the early 80's, by consolidating the then current ideas of accounting research within a unified framework [11]. In its initial and very condensed form, REA describes three concepts: economic resources, economic events, and economic agents. Following accounting theory, an economic event resembles something that has actually happened and that causes a record in the general ledger, such as paying for raw material, receiving money for selling finished products, or renting offices.

The initial REA model was extended and refined to a more complete business ontology comprising new types of events for production and a planning layer that allows the specification of contracts, schedules, policies, etc. [12], [13], [14], [15]. REA thus resembles a link between vertical integration (supports the modeling of production processes) and horizontal integration (supports the modeling of resource exchange and internal/external business modeling).

The runtime configurability of well-designed REA-based information systems has been successfully demonstrated in [16], by running a configurable retail information system (RIS) [17]. The RIS domain model can be evolved by adding/removing/altering types, objects, and their properties.

D. Type-Object Modeling Pattern

As a modeling pattern, the type-object (or power type) pattern [18], [19], [20] is often used in runtime configurable systems in order to allow the dynamic creation and manipulation of classifiers (the types) and instances thereof (the objects). The type-object pattern is one solution to the three-level case of multi-level software engineering [21], [22]. The type-object pattern is very convenient for a number of use-cases, which is why it has been adopted in all the previously mentioned technologies: in REA, in ISA-95 (there it is called class layer), and in AML (system unit classes and role classes can be used for creating instances in the type layer).

For REA, the type-object implementation has best been described in [23], [15], [17]. In short, the type-object pattern proposes to provide a type class and an object class for the specification of a specific entity type and its instances. E.g., in order to support multiple types of agents and instances of these agents, a class `Agent Type` and a class `Agent` would be defined. Each `Agent` instance (e.g., a person called "John Smith" who is employed as a salesman at a company) would be associated with the specific `Agent Type` instance "Salesman". That way, a new type of agent (e.g., "Cashier") could be added at runtime by creating a new instance of `Agent Type`; then, `Agent` instances could impersonate this type of agent.

E. Model-Driven Vertical Integration

The importance of vertical integration in production processes is emphasized in [24] and [25], where a domain specific modeling language, derived from *business process model and notation*[5], is introduced.

In [26] some key challenges for software evolution in aPS are collected, including the co-evolving of interdisciplinary engineering models, which is the challenge addressed in this paper. One of the research goals stated in [26] is the development of automatic consistency mechanisms for domain specific systems. With our approach we can contribute to this research goal.

System evolution in the context of aPS and information systems (IS) is investigated in [27]: (i) hardware changes in a pick and place unit require an evolution of the the state chart model as well as changes at the code level and (ii) the migration of IS components to the cloud demands changes in deployment, configuration, etc. Their approach stresses the importance of architecture models (from IS to control systems) and their use in the estimation of change effort estimation and impact analysis. The examples given comprise manual co-evolution of different systems based on changes in the architectural model. In our approach (i) changes in one system should automatically propagate to other systems, where this is possible and (ii) the overall architectural model is not modeled explicitly, but to be inferred from multiple domain models.

Integration of the various models in aPS is studied in [28] by employing a linking metamodel that allows the explicit linking of model elements from different modeling domains in order to track consistency, constraint satisfaction, etc. Their approach could be used in conjunction with the approach presented in this paper by providing explicit mappings between model elements and not relying solely on the metamodel level. It would be worthwhile to investigate an integration of their linking metamodel into our approach.

Co-evolution of production system models and their libraries is examined in [29], where AML models and their

[4]cf. http://www.omg.org/spec/UML/

[5]cf. http://www.omg.org/spec/BPMN/

AML model library prototypes are checked for different kinds of inconsistencies. Repair operations are executed or warning messages are displayed using the Epsilon Validation Language[6] (EVL). The tools and techniques used in their work overlap with what we have used in our approach, however in contrast to their work, our approach targets the comparison and balancing of diverging metamodels with different main focuses.

In [10], an approach for the alignment of ISA-95 and AML has been proposed, where the different metamodel elements are matched against each other, and a technique is presented to reference ISA-95 information from AML documents. Their work can be used as a basis for the formulation of transformation rules between AML and ISA-95.

A high level alignment of a reduced set of REA and ISA-95 has been presented in [30] and [31], showcasing an application scenario where vertical and horizontal integration are brought together to provide an integrated engineering view on internal processes and external dependencies. Parts of their work are used as a basis for the transformation rules presented in this work.

In [32], the usage of ISA-95 as task layer execution script is explored, and a high level alignment of REA and ISA-95 is presented. The idea is to rely on ISA-95 for representing detailed production information and provide only relevant information to the business layer. In their work, high level elements of ISA-95 are aligned with REA, while lower level elements of ISA-95 are used to describe aspects that are beyond the usual application of REA. For our approach this means that these lower level concepts are usually not transformed between ISA-95 and REA, but might be relevant for the information exchange between ISA-95 and AML.

III. Alignment of AML, ISA-95 and REA

Given the alignments already defined in [10] (AML and ISA-95) and [30], [31], [32] (REA and ISA-95), we want to showcase how specific entities and their properties propagate between the different systems of interest in order to keep then in sync. Fig. 1 depicts a high level view on the various alignments that are involved in the translation between the systems in vertical integration scenarios.

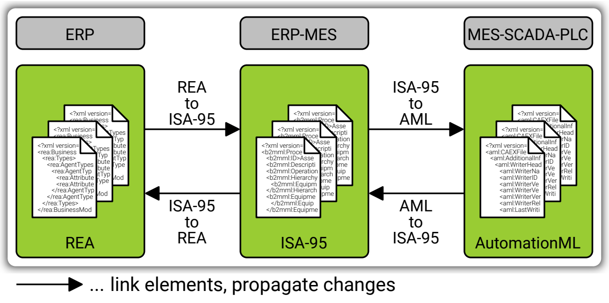


Fig. 1. Overview of model links defined for the purpose of vertical integration.

[6]cf. https://www.eclipse.org/epsilon/doc/evl/

In the remainder, we are using the Epsilon Object Language[7] (EOL) for querying model states, Epsilon Transformation Language[8] (ETL) for model-to-model transformations, and EVL for the validation of models.

Given an entity PT_{ISA95} that resembles an ISA-95 Material Definition with the ID attribute set to "Pine Timber":

- in AML this would be correspondingly instantiated as PT_{AML} of type SystemUnitClass with a reference to RoleClass "ProductStructure",
- in REA, this would be resembled as entity PT_{REA} of type Resource Type.

In order to determine whether a given entity exists in one of the other system levels, we can query its state space and look for a corresponding item. In the case of PT_{ISA95}, we can look into AML documents and query the registered SystemUnitClasses for an attribute with the name "ID" and a matching value. It could also be that the link between ISA-95 and AML has been explicitly defined by an ExternalDataReference of a SystemUnitClass to a B2MML document or element representing PT_{ISA95}.

In the case of REA, the approach would be similar: the REA model would be searched for a Resource Type with a name or "ID" attribute matching "Pine Timber". The link could also be explicitly defined in either the REA element using an attribute "ISA-95 ID" that would hold the value "Pine Timber" (cf. Lst. 1). Vice versa, the link could also be defined in ISA-95, by adding a Material Definition Property with the ID "REA ID" and the corresponding value.

Lst. 1. Querying the REA model for Material Definitions with a matching name or "ISA-95 ID" attribute, expressed in EOL.

```
var needle = "Pine Timber";
var supn = "Material Definition";
for( m in Model.all )
{
   for( rt in m.resourceTypes
      .select( n | not n.superType.isUndefined()
         and n.superType.name = supn
         and n.name = needle ) )
   {
      rt.println( "Found by name: " );
   }
   for( rt in m.resourceTypes
      .select( n | not n.superType.isUndefined()
         and n.superType.name = supn
         and not n.attributes.isUndefined()
         and n.attributes.selectOne( a | a.key =
            "ISA-95 ID" ).value.stringValue =
            needle ) )
   {
      rt.println( "Found by attribute: " );
   }
}
```

Using MDE techniques, it is possible to generate stub models in various domains, generated from a single base model. Using this approach, the different domain models can be created from a common, synchronized understanding

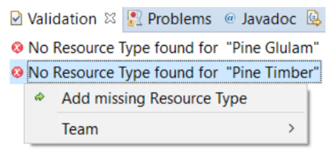
[7]cf. https://www.eclipse.org/epsilon/doc/eol/
[8]cf. https://www.eclipse.org/epsilon/doc/etl/

of core concepts. Lst. 2 shows an excerpt of the transformation from an ISA-95 model (exemplified with Material Definitions) into an REA model. The showcased rule creates a new Resource Type and sets its name to the ID of an input Material Definition (line 7). Additionally, the parent Resource Type “superType” is set to the Resource type with name “Material Definition” (line 8–9). Finally, the Resource Type is added to the REA model (line 10). The whole rule is executed only if there exists no other Resource Type with that name already in the REA model (realized through a *guard* in line 5).

Lst. 2. Transformation of Material Definitions to Resource Types, expressed int ETL.

```
 1 rule MaterialDefinition2ResourceType
 2    transform md : isa!MaterialDefinitionType
 3    to rt : rea!ResourceType
 4 {
 5    guard: rea.resourceTypes.select( rt |
 6       rt.name = md.iD.value ).size() == 0
 7    rt.name = md.iD.value;
 8    rt.superType =
 9       rt.getMaterialDefinition( rea );
10    rea.resourceTypes.add( rt );
11 }
```

In order to determine if two models of different domains are in sync, they can be validated against each other. Lst. 3 checks whether an REA model under test contains all the Material Definitions of a given ISA-95 model, and using Epsilon[9] tooling, a human domain expert would get the chance to fix occurring issues with the click of his/her mouse. Lines 5–6 check whether a Resource Type with a name corresponding to the ID of the current Material Definition exists—and if not, it displays the error message defined in lines 7–8: the user interface integration of Epsilon enables executing the fix defined in lines 9–22 (cf. Fig. 2). The fix creates a new Resource Type with a name corresponding to the ID of the current Material Definition (lines 14–15), sets its parent to the generic “Material Definition” Resource Type (lines 16–18) and adds it to the underlying REA model (lines 19–20).



No Resource Type found for "Pine Glulam"
No Resource Type found for "Pine Timber"

Fig. 2. The error messages of the epsilon validation engine provide a clickable “quick fix” facility.

IV. Application Scenario

An explicit example shall clarify the effect of model-driven synchronized production and business systems under the influence of changes in any of the given subsystems. The use case describes the evolution of involved systems and the

[9]cf. https://www.eclipse.org/epsilon/

Lst. 3. Excerpt of a cross-model validation between ISA-95 and REA, expressed in EVL.

```
 1 context isa!MaterialDefinitionType
 2 {
 3    constraint MaterialDefinitionExists
 4    {
 5       check : rea!ResourceType.all.select( rt |
 6          rt.name = self.iD.value ).size() = 1
 7       message : "No Resource Type found for "+
 8          " \"" + self.iD.value + "\""
 9       fix
10       {
11          title : "Add missing Resource Type"
12          do
13          {
14             var rt = new rea!ResourceType;
15             rt.name = self.iD.value;
16             rt.superType = rea!ResourceType.all
17                .selectOne( md | md.name =
18                "Material Definition" );
19             rea!Model.all.first()
20                .resourceTypes.add( rt );
21          }
22       }
23    }
24 }
```

services they provide: the addition of a new business service offering requires changes in the production facilities. These changes occur at different hierarchy levels, and we show how these changes can be propagated to systems of other hierarchy levels using model-driven engineering techniques.

A. Initial System State

The example is based on a fictitious company “Glulam Ltd.” that has specialized in the production of glued laminated timer (*glulam*). The core production process consists of pieces of timber as raw material that are fed into a continuous finger jointer that produces an endless so called “lamella” that is cut into pieces of required length. Several of these pieces of lamella are then laminated (glued together) to form a thicker piece of wood, a “glulam”, that is often used for building construction work. This production process is depicted in Fig. 3, that roughly correlates with how this process would be modeled in ISA-95 using the Process Segment model. For the sake of simplicity, Personnel Segment Specifications have been omitted.

The corresponding model in REA of the “Lamination” Process Segment is depicted in Fig. 4. It maps to a Transformation Duality Type that consists of a Produce Event Type, a Consume Event Type, and a Use Event Type. The rationale behind this mapping is: a Process Segment is not a specific instance of a production run, but resembles the blueprint for specific production runs (Operations Performances). Similarly, a Transformation Duality Type serves as the blueprint for specific Transformation Dualities that relate Transformation Events to each other. Here, the “Lamination” Transformation Duality Type comprises:

- incremental Produce Event Type “Produce Glulam” that indicates the production of a certain amount of “Glulam” Resource Types,

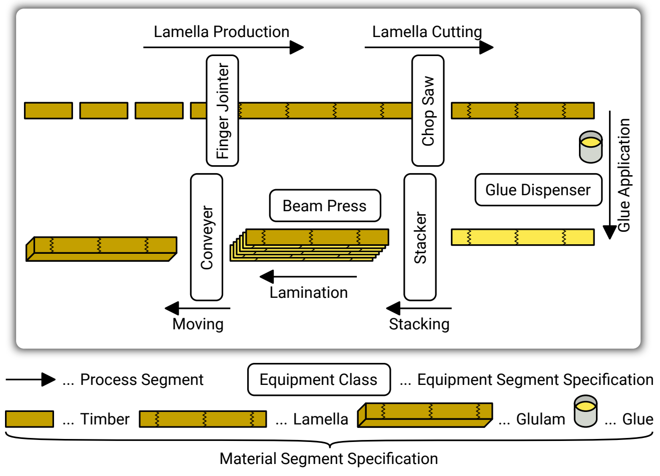


Fig. 3. Overview of the underlying production process of the application scenario in its initial state.

- decremental Consume Event Type “Consume Lamination Material” that explicates the consumption of input material (lamella and glue), and
- decremental Use Event Type “Use Lamination Equipment” that resembles the use of required machinery.

The associated participants for all these Event Types are the “Lamination Operator” and the “Shift Supervisor” Agent Types. In the incremental Event Type the “Lamination Operator” resembles the *providing* Agent Type and the “Shift Supervisor” is the *receiving* Agent Type. In the decremental Event Types the participation roles are inverted.

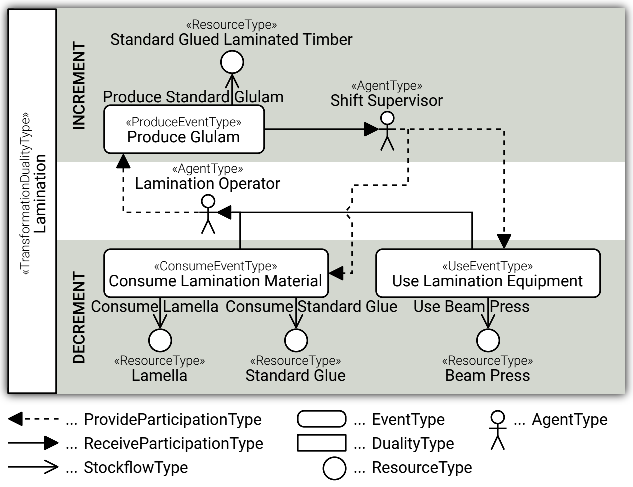


Fig. 4. A view on the REA model of the given application scenario in its initial state.

B. Model-Driven Co-Evolution

So far, the glulam lamination process consisted of pressing the lamellas for a given amount of time. However, recently a new type of glue has been suggested to meet the requirements of some important customers with very specific needs. This type of glue requires a minimum temperature of 25°C while curing; however, it yields a much stronger glulam. Therefore, a heating device is installed next to the beam press in order to ensure the required temperature.

1) Evolution of the business system: For the business system, this means adding three new Resource Types “Heating Device”, “Warm Curing Glue”, and “Strong Glulam” and deciding on how the new elements should be integrated into the business model. Here, it is decided to make a copy of the “Lamination” Transformation Duality Type called “Minimum Temperature Lamination” and adapt it by adding and changing Stockflow Types (cf. Fig. 5): (i) the glue to be used is now a warm curing glue, (ii) the final product is now a strong glulam, and (iii) a heating device is added to the Use Event Type.

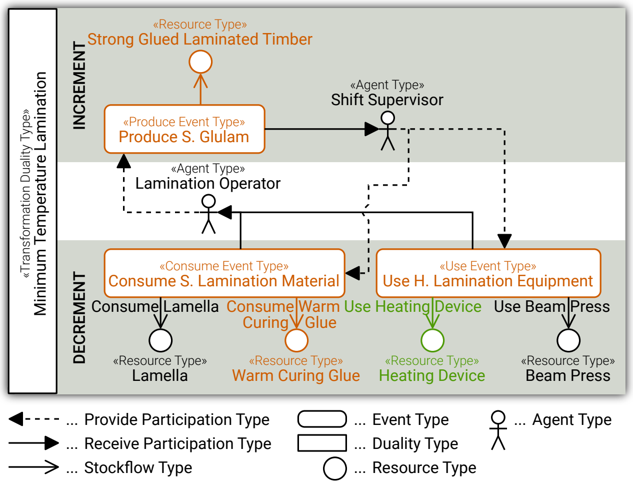


Fig. 5. The REA model of the given application scenario in its evolved state. Green items depict elements that have been added compared to the initial state, orange items resemble elements that have been present in the initial state in a structurally similar way, but that now resemble other entities.

2) Evolution of the ERP-MES transfer model: For the ERP-MES transfer model (expressed in ISA-95), the heating device needs to be integrated into the production process, and this change must also be reflected in the corresponding ISA-95 models, specifically the material model, the equipment model and the process segment model need to be changed. Based on an alignment of the meta-models of REA and ISA-95, an initial skeleton of the ISA-95 model can be created by transforming the REA model. The new Transformation Duality Type is transformed into a Process Segment, the Resource Types into elements of the material and equipment model, and the Stockflow Types into Material Segment Specifications and Equipment Segment Specifications. Some transformations require manual intervention or a specific naming or structuring convention, because it is not intrinsically clear, whether a Resource Type is mapped into a Material Class/Definition, Equipment Class, or Physical Asset Class. Lst. 4 shows an excerpt of a cross-model validation between REA and ISA-95, with the convention that the IDs of ISA-95 elements correspond to names of REA elements. The validation code checks whether all REA Transformation Duality Types have a Process Segment counterpart in ISA-95 and if not, offers a quick fix for generating a skeleton Process

Segment. It does that by traversing all Event Types and adding all of their Stockflow Types as Equipment, Physical Asset, or Material Segment Specification. In order to keep the example concise, only the traversal of Use Event Types is depicted.

Lst. 4. Excerpt of a cross-model validation between REA and ISA-95, expressed in EVL. A backslash (\) denotes a soft line break required to adapt to the line width.

```
context rea!TransformationDualityType
{
   constraint ProcessSegmentExists
   {
      check : isa!ProcessSegmentType.all.exists(
         ps | ps.iD.value = self.name )
      message : "No Process Segment found for "+
         " \"" + self.name + "\""
      fix
      {
         title : "Add missing Process Segment"
         do
         {
            var ps = new isa!ProcessSegmentType;
            ps.iD = new isa!IDType;
            ps.iD.value = self.name;
            for( uet in self.useEventTypes )
            {
               for( sft in uet.stockflowTypes )
               {
                  var ess = new isa!Equipment\
                     SegmentSpecificationType;
                  ess.equipmentClassID = new isa
                     !EquipmentClassIDType;
                  ess.equipmentClassID.value =
                     sft.resourceType.name;
                  ess.equipmentUse = new isa!
                     EquipmentUseType;
                  ess.equipmentUse.value = uet
                     .name + ": " +
                     sft.resourceType.name;
                  ps.materialSegment\
                  Specification.add( ess );
               }
            }
            /* Code intentionally left out */
            isa!ProcessSegmentInformationType
               .all.first().processSegment
               .add( ps );
         }
      }
   }
}
```

3) Evolution of the Plant Topology: The plant topology that is expressed in AML can benefit from the adapted ERP-MES transfer model by following the mapping rules presented in [10]. As a result, SystemUnitClasses for the heating device, strong glulam, warm curing glue, and minimum temperature lamination need to be created. This can again be achieved by a set of transformation rules that transform an ISA-95 model into a corresponding AML model.

V. Critical Discussion

The presented approach provides an initial setup for the transformation between systems of various layers of automated production systems. Some restrictions apply that prohibit a fully automated transformation between the different systems. E.g., the mapping between REA and ISA-95 sports syntactic inter-model heterogeneities including *1:n*, *I2I*, and *BreadthDifference*, following the classification presented in [33], regarding the mapping between REA Resource Types and ISA-95 Material Class/Definition, Equipment Class, and Physical Asset Class. The 1:n heterogeneity causes a decision that needs to be taken in order to determine what kind of output instance should be created for a given input instance. One way to solve this issue is by defining company specific conventions or by providing generic Resource Types (e.g., named “Equipment Class”) that serve as parent types for respective instances.

Another problem that cannot be handled generically are different levels of indirection, such as REA Events, that have no equivalent in ISA-95. While event entities can be skipped when transforming from REA to ISA-95, they cannot be generically created when transforming from ISA-95 to REA: it is not clear, which ISA-95 Segment Specifications should be bundled together in single Event Types as Stockflow Types and Participation Types. Modeling conventions could help in resolving parts of these issues, or additional reasoning steps could be introduced that would provide a meaningful modeling structure for the entities in question.

We have chosen to present the application of our approach on a very specific fictitious company, however, the approach itself and most of its implementation are company and domain agnostic. This can be verified by examining the code listings and asserting that only metamodel classes and features are referenced, except for e.g., the statement in line 18 of Lst. 3, where a company specific modeling convention is exemplified. This is inevitable in some cases in order to e.g., accommodate to a basic set of rules on how company assets are modeled, or to explicitly denote that specific elements of different domains represent in fact the same entity.

We have refrained from presenting details about the transformation between AML and ISA-95 models, as the alignment between these two metamodels has been presented in more detail in [10] already. Corresponding transformation rules can be inferred from the mappings defined there.

The benefit of our approach is that a common understanding of concepts from different domains is accomplished by relating *metamodel* elements with each other. This approach is agnostic to the kind of business a company is involved in. Specific implementations could provide industry related information in order to better acknowledge peculiarities and conventions.

VI. Conclusion and Outlook

We have presented a model-driven approach for the co-evolution of models residing on different levels with respect to the automation hierarchy, based on a generic alignment of corresponding metamodels. While the given technique does not provide a “single point of intervention” when it comes to changes in the models, it facilitates the creation of stub models and provides means for cross-model validation. The main contribution is thus the model-driven propagation of basic model elements and changes of model elements between models of different hierarchy levels. By defining clear rules it might even be possible to overcome some of the main limitations with respect to the automatic propagation of changes. Some of these conventions might be defined in a generic way, others might only be possible in a company's environment.

For the future, we would like to specify the cross-model validations and transformations more exhaustively in order to provide a rather complete setup for the co-evolution of models situated in different hierarchy levels in production systems, with the goal of vertical integration. Further, we plan the incorporation of communication stacks such as OPC Unified Architecture[10]. Here, we would like to investigate cross-domain model mining based on runtime information captured in structured communication streams. E.g., the business layer could be adapted via ISA-95 based on operations performance information sourced in the production process.

References

[1] International Electrotechnical Commission (IEC), *Representation of process control engineering—Requests in P&I diagrams and data exchange between P&ID tools and PCE-CAE tools*, International Standard, Rev. 1.0, August 2008, IEC 62424:2008.

[2] International Electrotechnical Commission (IEC), *Engineering data exchange format for use in industrial automation systems engineering—Automation markup language—Part 1: Architecture and general requirements*, International Standard, Rev. 1.0, June 2014, IEC 62714-1:2014.

[3] International Electrotechnical Commission (IEC), *Engineering data exchange format for use in industrial automation systems engineering—Automation markup language—Part 2: Role class libraries*, International Standard, Rev. 1.0, March 2015, IEC 62714-2:2015.

[4] AutomationML consortium, *AutomationML Whitepaper Part 3—Geometry and Kinematics*, Whitepaper, Rev. 2.0, August 2015, AML Part 3:2015.

[5] PLCopen Technical Committee 6, *XML Formats for IEC 61131-3*, Technical Paper, Rev. 2.01, May 2009, PLCopen XML:2009.

[6] AutomationML consortium, *AutomationML Whitepaper Part 4—AutomationML Logic Description*, Whitepaper, Rev. 2.0, May 2010, AML Part 4:2010.

[7] International Electrotechnical Commission (IEC), *Enterprise-control system integration—Part 1: Models and terminology*, International Standard, Rev. 2.0, May 2013, IEC 62264-1:2013.

[8] International Electrotechnical Commission (IEC), *Enterprise-control system integration—Part 2: Objects and attributes for enterprise-control system integration*, International Standard, Rev. 2.0, June 2013, IEC 62264-2:2013.

[9] Manufacturing Enterprise Solutions Association (MESA) International, *Business To Manufacturing Markup Language*, Standards and Tools, Rev. V0600, May 2013.

[10] B. Wally, C. Huemer, and A. Mazak, "Entwining plant engineering data and ERP information: Vertical integration with AutomationML and ISA-95," in *Proceedings of the 3rd IEEE International Conference on Control, Automation and Robotics (ICCAR 2017)*, 2017.

[11] W. E. McCarthy, "The REA accounting model: A generalized framework for accounting systems in a shared data environment," *The Accounting Review*, vol. 57, no. 3, pp. 554–578, 1982.

[12] G. L. Geerts and W. E. McCarthy, "Modeling business enterprises as value-added process hierarchies with resource-event-agent object templates," in *Business Object Design and Implementation*, 1997.

[13] G. L. Geerts and W. E. McCarthy, "An accounting object infrastructure for knowledge-based enterprise models," *IEEE Intelligent Systems*, vol. 14, no. 4, pp. 89–94, 1999.

[10]cf. https://opcfoundation.org/about/opc-technologies/opc-ua/

[14] G. L. Geerts and W. E. McCarthy, "The ontological foundation of REA enterprise information systems," in *Annual Meeting of the American Accounting Association, Philadelphia, PA*, vol. 362, 2000.

[15] G. L. Geerts and W. E. McCarthy, "Policy-level specifications in REA enterprise information systems," *Journal of Information Systems*, vol. 20, no. 2, pp. 37–63, 2006.

[16] B. Wally, A. Mazak, B. Kratzwald, C. Huemer, P. Regatschnig, and D. Mayrhofer, "REAlist—a tool demo," in *Proc. of the 9th Int. Workshop on Value Modeling and Business Ontology*, 2015.

[17] B. Wally, A. Mazak, B. Kratzwald, and C. Huemer, "Model-driven retail information system based on REA business ontology and Retail-H," in *Proceedings of the 17th IEEE Conference on Business Informatics (CBI 2015)*, vol. 1, 2015, pp. 116–124.

[18] P. Coad, "Object-oriented patterns," *Communications of the ACM*, vol. 35, no. 9, pp. 152–159, 1992.

[19] R. Johnson and B. Woolf, "Type object," in *Pattern Languages of Program Design 3*, R. C. Martin, D. Riehle, and F. Buschmann, Eds. Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc., 1997, ch. Type Object, pp. 47–65.

[20] Object Management Group, Inc., *OMG Unified Modeling Language (OMG UML)*, Specification, Rev. 2.5, March 2015.

[21] C. Atkinson and T. Kühne, "The essence of multilevel metamodeling," in *UML 2001—The Unified Modeling Language. Modeling Languages, Concepts, and Tools*, ser. Lecture Notes in Computer Science, 2001.

[22] T. Kühne and F. Steimann, "Tiefe Charakterisierung," in *Modellierung 2004*, ser. Lecture Notes in Informatics, B. Rumpe and W. Hesse, Eds., vol. P-45. Gesellschaft für Informatik, 2004, pp. 109–120.

[23] G. L. Geerts and W. E. McCarthy, "An ontological analysis of the economic primitives of the extended-REA enterprise information architecture," *International Journal of Accounting Information Systems*, vol. 3, no. 1, pp. 1–16, 2002.

[24] M. Witsch and B. Vogel-Heuser, "Towards a formal specification framework for manufacturing execution systems," *IEEE Transactions on Industrial Informatics*, vol. 8, no. 2, pp. 311–320, May 2012.

[25] M. Witsch, "Funktionale Spezifikation von Manufacturing Execution Systems im Spannungsfeld zwischen IT, Geschäftsprozess und Produktion," Ph.D. dissertation, Fakultät für Wirtschaftswissenschaften, Technische Universität München, 2013.

[26] B. Vogel-Heuser, S. Feldmann, J. Folmer, J. Ladiges, A. Fay, S. Lity, M. Tichy, M. Kowal, I. Schaefer, C. Haubeck, W. Lamersdorf, T. Kehrer, S. Getir, M. Ulbrich, V. Klebanov, and B. Beckert, "Selected challenges of software evolution for automated production systems," in *Proc. of the 13th IEEE Int. Conf. on Industrial Informatics*, 2015.

[27] B. Vogel-Heuser, S. Feldmann, J. Folmer, S. Rösch, R. Heinrich, K. Rostami, and R. Reussner, "Architecture-based assessment and planning of software changes in information and automated production systems state of the art and open issues," in *Proc. of the 2015 IEEE Int. Conference on Systems, Man, and Cybernetics (SMC 2015)*, 2015.

[28] S. Feldmann, M. Wimmer, K. Kernschmidt, and B. Vogel-Heuser, "A comprehensive approach for managing inter-model inconsistencies in automated production systems engineering," in *Proceedings of the 12th IEEE International Conference on Automation Science and Engineering (CASE 2016)*, 2016, pp. 1120–1127.

[29] L. Berardinelli, S. Biffl, E. Maetzler, T. Mayerhofer, and M. Wimmer, "Model-based co-evolution of production systems and their libraries with AutomationML," in *Proceedings of the 20th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA 2015)*, 2015.

[30] A. Mazak and C. Huemer, "From business functions to control functions: Transforming REA to ISA-95," in *Proceedings of the 17th IEEE Conference on Business Informatics (CBI 2015)*, D. Aveiro, U. Frank, K. J. Lin, and J. Tribolet, Eds., vol. 1. IEEE, 2015.

[31] A. Mazak and C. Huemer, "HoVer: A modeling framework for horizontal and vertical integration," in *Proceedings of the 13th IEEE International Conference on Industrial Informatics (INDIN 2015)*. IEEE, 2015, pp. 1642–1647.

[32] B. Wally, C. Huemer, and A. Mazak, "ISA-95 based task specification layer for REA in production environments," in *Proceedings of the 11th International Workshop on Value Modeling and Business Ontologies (VMBO 2017)*, 2017.

[33] M. Wimmer, G. Kappel, A. Kusel, W. Retschitzegger, J. Schoenboeck, and W. Schwinger, "Towards an expressivity benchmark for mappings based on a systematic classification of heterogeneities," in *Proceedings of the First International Workshop on Model-Driven Interoperability*, ser. MDI '10. New York, NY, USA: ACM, 2010, pp. 32–41.