Stephan Bögel and Werner Esswein

Vertical Model Integration in Frameworks

Evaluation of Integration Approaches and Investigation of the Feasibility of Implementing a Case Study

The model driven simulation of mobile construction equipment is investigated in the BMBF-funded project INPROVY. A framework was introduced to structure the problem and divide it into different model layers. The vertical integration of the different model layers is the subject of this article. An evaluation through the method of feature-based comparisons of the integration approaches that represent the state of research is carried out. In a small case-study the applicability of vertical model integration in the field of model driven simulation is analysed.

1 Introduction and Motivation

In many technical disciplines, the simulation of machines or processes is an important means to gain knowledge and save costs. For the most part, a simulation is oriented to a specific case and is not systematically created. In order to establish a systematic approach, the creation of simulation models for mobile work machines in the BMBF supported research project INPROVY will be investigated.

Because mobile construction equipment, such as wheel loaders or concrete injection pumps, is made up of various components from various suppliers, the systematic creation of reusable simulation models is a complex task. To reduce the complexity, a framework that divides the problem into manageable units of analysis was created (Esswein et al. 2009).

The framework is made up of, among others, the layers (Esswein et al. 2009):

- Domain layer: Describes the structure of the mobile machinery from the perspective of the business processes. Here, for example, the concept of 'product components' is to be found. This is characterised through an item number that is used in the ordering or storage process.
- Simulation layer: Looks at the problem oriented processing of the machine simulation. Here, for example, the concept 'simulation components' is to be found.

The layers must be integrated in order to propagate a change from one layer to all layers involved and to recognise inconsistencies between layers. In particular, the coordination between the domain and the simulation layers is crucial for the simulation. Information from the business processes can be used in the creation of simulations only if the business model is integrated with the simulation models. Furthermore, only then can the simulation results be quickly integrated into the business processes. This is donated as vertical integration of framework layers. Solutions for the vertical integration of layers are required for the consistent, model-supported usage and creation of simulation models.

In order to achieve these objectives, the dependence between the framework layers must be de-

Revised version of: Boegel S, Esswein W: Vertikale Modellintegration in Rahmenwerken – Evaluation von Integrationsansätzen und Untersuchung der Implementierbarkeit anhand eines Fallbeispiels. In: Esswein W, Turowski K, Juhrisch M (eds) Proceedings of MobIS 2010: Modellierung betrieblicher Informationssysteme, September 15-17 2010, Dresden, Germany. LNI 171, pp. 79–98.

scribed (Aier et al. 2008). This raises the question of how these dependencies can be modelled. In this study, model integration approaches representing the current state of research will be evaluated. In addition to the assessment of these approaches, this study will identify the key design dimensions required to adapt an approach to the problem of vertical integration in frameworks.

The evaluation's universe of discourse (Bögel et al. 2011) as well as the goal of the presented work suggest a constructivist perspective. This research is attributed to the Design Science approach (Hevner et al. 2004). Based on the research methodology from Verschuren and Hartog (2005), Sect. 2 will study the problem under theoretical principles. Subsequently, the requirements analysis, the formulation of the evaluation framework and the execution of the evaluation are found in Sect. 3. The case study is examined in Sect. 4 and the need for further research is identified in Sect. 5.

2 Theory of Vertical Model Integration

The problem of vertical model integration of layers in frameworks is hereafter narrowed to the integration direction (vertical vs horizontal), integration type (connecting vs unifying) and integration approach (transformation vs classification). The delimitation of the problem is accomplished through the consideration of various features of the integration terms and the exclusion of certain characteristic values from observation. A summary is given in Table 1.

2.1 Vertical vs. Horizontal Model Integration

The vertical and horizontal model integration can be distinguished as possible integration directions (Rosemann 2002). Following the generic framework of Sinz (1999), Fig. 1 shows excerpts from two exemplary presented frameworks. These are implemented by two different companies.

Table 1: Integration dimensions

Integration characteristic	Value			
Type	Connecting	~~Unifying~~		
Direction	Vertical	Horizontal	Hybrid	
Realization	~~Organization layer~~	~~Technology layer~~	Model layer	
Subject-matter	~~Data, Processes, Functions, Objects~~, Models, ~~Methods~~			
Scope	Task	Individual	Organisational unit	Company
	Internal	Inter-company	External	
Automation	Automatic	Semi-automatic	Manual	
Semiotic dimension	Syntax	Semantics	~~Pragmatics~~	
Approach	~~Transformation~~	Mapping		

In framework A company 1 and company 2 use the same architectural framework with the same modelling language. In framework B the two companies use different modelling languages. This results in different meta-models MMk,x within a layer.

Furthermore, two different models, model Mi,1 and Mj,1, that exist on two different layers have different viewpoints (Sinz 1999). Since the models describe the same company, they refer to the same universe of discourse. Two models Mi,1 and Mj,2, from two different companies or organisational units, which are on the same layer, differ then in the universe of discourse, but not in the viewpoint.

This scenario of horizontal integration occurs, for example, if both companies merge or seek close cooperation (e.g., with the goal of an integrated supply chain). The models Mi,1 and Mj,2 in framework A represent such a case. In contrast, if both companies use different modelling languages, as indicated in framework B, this must be taken into account during the integration. Therefore, overcoming language barriers must be considered during vertical integration, in horizontal integration only in case B. In the frame

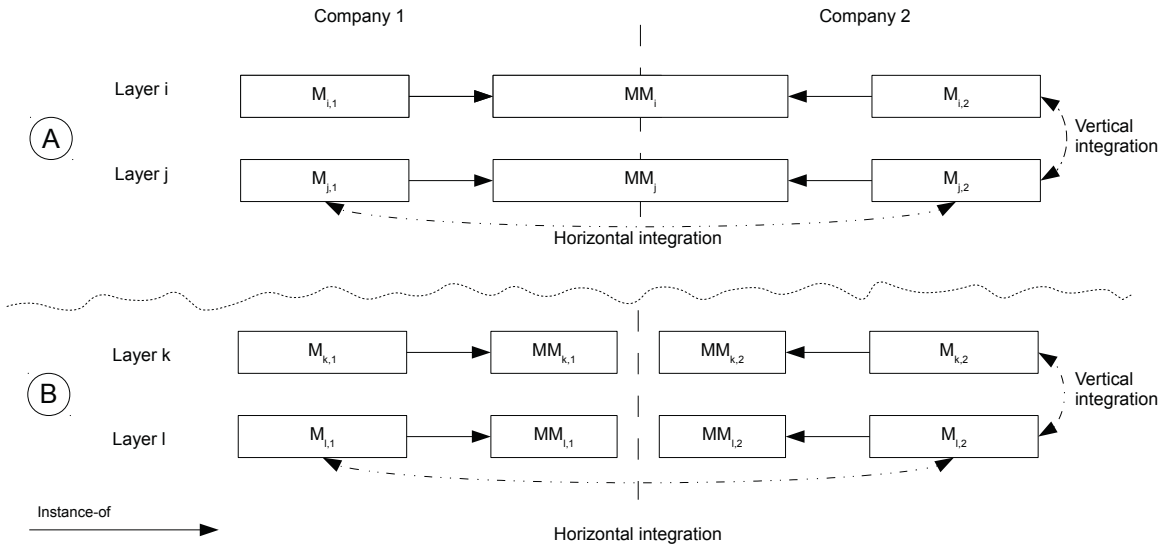

Figure 1: Vertical vs. horizontal model integration in frameworks

of INPROVY, a framework for model driven simulation is developed including the modelling languages. Therefore case A is presented in the following.

2.2 Connective vs. Unifying Model Integration

With respect to the type of integration an integration can occur through connection or unification. Through the integration by connection, previously unrelated elements with logical relationships to one another are connected to a system (Rosemann 2002). Through the integration via unification a new, merged model which replaces the old model is the integration result (Rosemann 2002).

In frameworks different layers are introduced for the purpose of reducing complexity. With respect to the vertical integration of framework layers, an integration that merges these layers is contradictory. Therefore, a vertical integration is preferably of the type 'connection'.

2.3 Transformation vs. Mapping

The integration of model layers can be accomplished through mapping or transformation (Sinz 1999). An example of a transformation-oriented approach is the Model Driven Architecture (MDA) (Mukerji and Miller 2003). Frankel et al. (2003) show how the MDA can be used for the integration of model layers in the Zachman framework (Zachman 1987). Each layer is associated with a MDA model-type, which are related by model-transformations.

Such a transformation relationship exists in the INPROVY framework as a procedure through which the layers are created top-down (Esswein et al. 2009). This approach makes sense in the (initial) development of simulation models, but meets its limits during recirculation from lower layers to the higher-lying layers. In the semi-formal models considered here, it cannot be assumed that these can be generated automatically (Gehlert 2007). Manual transformation is not possible because of the expenditure required for each change. In terms of a round-trip model engineering, ways to integrate the layers of the model so that a change to one layer does not lead to underlying layers having to be generated from scratch (Sendall and Köster 2004) must be found.

In this case, an integration though mapping is useful. Furthermore, mappings are also a pre-

requisite for transformation by which transformation rules based on the meta-model element are described (Mukerji and Miller 2003). It is therefore that the integration through mapping is reviewed.

2.4 Integration conflicts

The heterogenity of models leads to integration conflicts between models. Diverse classifications of integration conflicts can be found in the literature. These classifications have to be consolidated to carry out the evaluation. We show that the conflicts can be divided into two groups. One group of conflicts concerns the (artificial) modelling language and the other group concerns the (natural) domain language.

A common classification distinguishes type, structural and name conflicts (Hars 1994; Pfeiffer and Gehlert 2005; Rosemann 2002). A type conflict is caused either through the use of different modelling languages or the use of different constructs of a modelling language to express the same issue (Pfeiffer and Gehlert 2005). By using the entity relationship model language, a certain issue can be modelled as attribute or as entity (Batini et al. 1986). If a part of the real world is modelled semantically different in two models, it is denominated as structural or semantic conflict (Pfeiffer and Gehlert 2005). Dependency conflicts, abstraction conflicts and level of detail conflicts can be distinguished (Kashyap and Sheth 1996).

Dependency conflicts arise if the same structure is modelled by mutually exclusive relations between elements. Different multiplicities or different generalisation or specialisation relations are examples of a dependency conflict (Kashyap and Sheth 1996). An abstraction conflict occurs if the same issue is modeled in a differing level of abstraction or generalisation. For example, in one model an issue is modeled as book in the other model as a publication (Kashyap and Sheth 1996). A level of detail conflict arises when the models have a different level of comprehensiveness compared to each other (Kashyap and Sheth 1996).

If the domain language is used differently, this is denominated as name conflict (Batini et al. 1986). Synonyms and homonyms are typical examples of name conflicts.

The triple part classification of name, structure, and name conflicts goes back to database schema integration. According to our understanding of integration, this classification is based on the horizontal integration.

Dijkman (2008) considers the horizontal integration of similar process models. He differentiates authorisation, activity and control-flow conflicts. (Weidlich et al. 2009) extend the list by process and data conflicts. Those conflicts relate to constructs of a modelling language. Others, like Becker et al. (2010), are differentiating homonym, abstraction, separation, type, synonym, annotation and control-flow conflicts.

We consolidate these classifications according to Table 2. In the first dimension, a conflict is related to the domain language or the modelling language. The second dimension represents the three layers of the semiotics: syntax, semantics and pragmatics.

A modelling language consists of a grammar and a set of domain expressions. The grammar defines the relations between concepts. Concepts of modelling languages are constructed by abstracting one or more domain expression (symbols) to a concept (Pfeiffer & Becker 2008). For example the domain expressions ‘customer’, ‘bill’ and ‘product’ can be abstracted to the concept ‘class’. The before mentioned conflicts either relate to symbols of the domain languages or to concepts of modelling languages. Name conflicts, for example, relate to symbols of the domain language. Type conflicts relate to concepts of modelling languages. The conflicts mentioned by Dijkmann and Becker et al. also relate to concepts of the modelling language by using concepts for describing these conflicts.

In the dimension of semiotics, the conflicts can be classified by looking at their occurrence. Name

Table 2: Integration conflicts

	Domain language	Modelling language
Syntax	Name conflict (homonym conflict, synonym conflict)	Type conflict
Semantics	-	Structural conflict (Authorisation conflict, Activity conflict, Process conflict,Data conflict, Dependency conflict, Abstraction conflict, Level of detail conflict, Annotation conflict, Control-flow conflict, Order conflict, Separation conflict)
Pragmatics	-	-

and type conflicts can be identified at the level of the syntax, e.g., through the comparison of strings or two type identifiers. All the other conflicts mentioned Becker et al. (2010); Dijkman (2008) can only be detected by looking at the semantic layer. No conflicts could be found on the pragmatic layer. This is because the intension of the model creator is not accessible for a third person.

The universe of discourse, depicted by the models we observe, is of material semantics. In this case the semantics cannot be reduced to syntax. This can be justified by the nonunderpinnability of language (Holenstein 1982). According to Wittgenstein (2010), semantics are constituted by the use of the symbols of a language. The mapping between syntax and semantics is an on-going process executed by human beings. Humans use symbols corresponding to his or her intentions and therefore constitute semantics.

As a consequence, for the solution of these conflicts, the next semiotic level has to be considered. To solve a (syntactical) synonym-conflict, the semantics of the symbol have to be consulted. To resolve semantic conflicts, the intention of the model creator has to be considered, i.e., the pragmatics. For example, to solve a control-flow conflict, the intention of the model creator has to be consulted to clarify whether the conflict is a result of the freedom of the modeller or if the order of the activities is mandatory.

3 Evaluation

For the evaluation, the method of feature-based comparison is used. For this purpose, a list of ideal integration characteristics is created to evaluate the approaches (Siau and Rossi 1998). The problem with this method is a high degree of subjectivity. To avoid this and to ensure traceability of the results, both the selection and interpretation have to be well documented (Siau and Rossi 1998).

3.1 Evaluation Framework

The evaluation procedure is based on the steps proposed by Heinrich (2000). The evaluation units were found through a systematic literature review (Die Sprecher der Wissenschaftlichen Kommission Wirtschaftsinformatik im Verband der Hochschullehrer für Betriebswirtschaft (WKWI) und des Fachbereichs Wirtschaftsinformatik der Gesellschaft für Informatik (GI-FB WI) 2008). Table 1 is used as a filter for the literature that is considered. In order to ensure that the evaluation method follows good documentation of the measurement, the considered approaches itself must be well documented (Siau and Rossi 1998). Therefore, only approaches featured in two or more publications are observed.

The following will briefly describe the evaluation criteria. A key requirement of an approach for model integration is to overcome the integration conflicts. The framework can only be considered

Table 3: Evaluation criteria and metrics

Criteria			Metric
A1	Overcoming of integration conflicts		
	1	Type conflict	Fullfilled, not fullfilled
	2	Structural conflicts	Fullfilled, partly fullfilled, not fullfilled
	3	Name conflict	
A2	Toolsupport (Automation)		Automatic, semi-automatic, manuel
A3	Implementation effort		Low, medium, high
A4	Support of		Supported, not supported
	1	Reusablity	
	2	Adaptability	
	of the meta-models		

integrated if these conflicts can be resolved. If conflicts are overcome both at the layer of syntax and the layer of semantics, the requirement is considered to be fulfilled. If conflicts are resolved only at the layer of syntax, the requirement is assessed as partly fulfilled. Type conflicts can be sufficiently resolved at the layer of the abstract syntax (Gehlert 2007). The requirement A1.1 is satisfied if the type conflicts are resolved either at the layer of syntax or semantics.

Due to the enormous number of entities in complex models, not all relationships can be manually created on the model layer. For this reason approaches must be found that allow for an automatic or semi-automatic definition of mappings at the model layer (Weidlich et al. 2009). Therefore, the degree an approach allows automation of the mappings at the model layer is assessed. This feature applies to the integration target 'actuality' (Rosemann 2002).

During the integration, the cost of autonomy must be compared with the integration effort. The effort required for the implementation of an approach is, therefore, integrated in the assessment, even if the costs and benefits cannot be measured quantitatively. Impacts have all the conditions that must be met for the integration of an approach. For this purpose, it is examined whether additional artefacts, such as relational meta-models or ontologies, need to be created or if certain integration prerequisites must be fulfilled.

Key quality features of frameworks are the reusability and the adaptability of the framework (Sinz 1999). The integration should give a substantial advantage and should not integrate the layers of the framework in a manner that those are tightly coupled. The requirements of reusability and adaptability are therefore related to the meta-models. They should, at least in similar projects, be reusable and adaptable. If the meta-models need to be modified in order for an integration to take place, the requirement A4 will be rated as 'not supported'.

3.2 Model Integration Approaches - Status of Research

The following will present and evaluate the status of research in model integration approaches. First, the approaches are divided into meta-model based (syntactic) and ontology-based (semantic) integration approaches (Arnarsdóttir et al. 2006).

The following approaches were found through literature research, but because of previously described reasons (see Table1) were not included in the evaluation:

- Fellmann and Thomas (2009) study model relations with semantic wikis. However, the relationships were not observed at the model element layer, but rather with the model as a closed artefact. The approach was only described in one publication.
- Simon and Mendling (2007) describe a horizontal approach to the integration of process models. The focus, however, is the unifying integration of the models.

- Fengel et al. (2008) provide an approach for the semantic linking of heterogeneous models. The model elements of the models to be integrated models are tagged with keywords. The approach is pursued in the frame of a research project and is not yet described by a sufficient number of publications.
- Hahn (2005) examines the integration of distributed product models with help from semantic web technologies. The documentation for the evaluation of the approach is also not sufficient.

3.2.1 Meta-model Based Integration Approaches

Meta-model based integration approaches allow for an integration using the abstract syntax.

Multi-Perspective Enterprise Modelling (MEMO)

The approach addresses the vertical integration of the layers 'strategy', 'organization' and 'information system' (Frank 2002). The integration of the languages is achieved using the same concept in every meta-model that should be integrated (Frank 2002). The MEMO SML, for example, uses the concept of an organisational unit from the MEMO OrgML together with the concept 'StrategicBusUnit' and 'HumanResource' (Frank 2002).

It is therefore classified as a connective integration using mappings. In essence, this approach corresponds to the integration of views within a model. As a consequence, a 'Super' language, which extends over all layers of the framework, is created.

Enterprise Model Integration (EMI)

Central concepts of the EMI approach are mappings and integration rules (Kühn et al. 2003). Mappings describe which parts of the meta-models will be integrated. Integration rules define how these mappings will be implemented (Zivkovic et al. 2007).

Integration rules can implement either a connective integration (through alignment rules) or a unifying integration (through connection rules). Integration models include the mappings between the language concepts of the integrated models (Zivkovic et al. 2007). This corresponds to the idea of relational meta-models described by Sinz (1999).

3.2.2 Ontology-based Integration Approaches

Ontology-based approaches seek to integrate models using semantics.

Intergration through Spoken Language Terms

Höfferer's approach relates the domain language expressions inside a model element to constructs of an ontology (Höfferer 2007). Karagiannis and Höfferer (2008) considered the horizontal integration of models and how they occur in the integration of processes of different companies.

Integration through Semantic Annotations from Lin

As with Höfferer Lin et al. (2006) considers the horizontal integration of process models. Unlike Höfferer, the model element concepts are assigned to ontology constructs and not the domain language terms that are contained in model element (Lin and Krogstie 2009).

3.3 Evaluation Results

Table 4 summarises the evaluation results.

3.3.1 Comparative Assessment of the Integration Approaches (Evaluation Objective 1)

A comparison of the criteria 'integration direction' shows that among the tested approaches only the meta-model-based approaches are used for vertical integration. The ontology-based integrations have not been, to this point, used for vertical model integration. There is a research gap that will be more closely examined in Sect. 4.

Table 4: Result of the evaluation

		Metamodel-based		Ontology-based	
		MEMO	EMI	HÖFFERER	LIN
Int. direction		vertical	vertical/ horizontal	horizontal	horizontal
A1	1	+	+	+	+
	2	-	o	+	+
	3	o	-	+	+
A2		-	o	o	-
A3		+	o	-	-
A4	1	-	+	+	+
	2	-	+	+	+

Legend + (A1: fullfilled / A2: automatic / A3: low / A4: supported)
o (A1: partly fullfilled / A2: semi/automatic / A3: medium)
- (A1: not fullfillef / A2: manuel / A3: high / A4: not sup.)

With all the approaches it is possible to overcome type conflicts. Structural conflicts are completely overcome with the ontology-based approaches. With the EMI approach, it is possible through the refinement of relation types to have a limited explication of structural conflicts at the layer of abstract syntax (Zivkovic et al. 2007). This is not possible at the layer of semantics because material semantics cannot be traced back to the syntax.

Name conflicts can only be fully overcome when using ontology based approaches. With the MEMO approach, this can be partially avoided in that the technical language terms are included in the meta-model (Frank 2002). A uniform understanding of domain language is, however, not given.

The requirement of automation is not completely fulfilled by any approach. EMI and the approach from HÖFFERER allow for, through their partial automation, support for the model creator through a tool. The approach from LIN and MEMO allow only for manual integration and are not suitable for the integration of complex frameworks.

The MEMO integration approach has the lowest implementation effort. The ontology based approaches require the highest effort, but they also fulfill most of the requirements.

The MEMO approach is the worst in respect to reusability because the modelling language on the various layers couldn't be used in separation from one another. Through its usage of separate integration models, the EMI approach can be viewed as significantly better in terms of reusability. The ontology based approaches require no modification on the meta-model layer and are therefore also positively assessed in the requirement of reusability. The same is true of the adaptability of the meta-models.

With regard to the prototypical integration of the INPROVY framework layers, the requirements A1, A2 and A4 could be identified as minimum requirement. Therefore, the approach from Höfferer is the only one suitable for the solution of integration issues. However, Table 4 also shows that a clever combination of the properties of meta-model based approaches could also fulfill the minimum requirements. For this purpose, the relevant design dimensions are identified.

3.3.2 Identified Design Dimensions (Evaluation Objective 2)

The comparative analysis of approaches shows the following design dimensions. In the meta-model based approaches, design leeway exists regarding the detailed nature of the meta-model. A higher layer of detail means that the model constructs, whose relationships are described in the meta-model, will have a finer resolution. The more detailed the meta-model is, the easier it is to describe relationships between two meta-models.

A coarse-granular meta-model contains, for example, the constructs ‘body’ and ‘organization unit’, whereas a fine-granular meta-model differentiates between tags like ‘controller’ and ‘accountant’ (Winter 2009).

A high layer of detail is, however, in conflict with the reusability (Frank 2002). If the terms

are taken from domain language into the meta-model, it cannot be assumed that other companies will use them with the same meaning.

Another design dimension is spanned by the integration relationships. The approaches differ in what types of relationships are allowed. This design dimension is also considered important from Weidlich et al. (2009). Integration relationships could be simple or complex such as generalisation or aggregation.

Interestingly, there is a difference between the two ontology based approaches. Complex relationships are either used only within the ontology (Höfferer) or between ontology and model (Lin). The construction effort shifts from the ontology creation towards relating model and ontology. Furthermore, if the reusability of ontologies is not studied because they aren't part of the frameworks, the first option is much better. The relationships between ontologies can be reused, the relationships between model and ontologies cannot.

The influence of the two integration dimensions will be further examined in the following case-study.

4 Case Study

The case-study observes a concrete injection pump. In Sect. 4.1 of the case-study, the influence of identified design dimensions will be examined. In Sect. 4.2, the application of ontologies on the vertical integration will be inspected in relation to the case-study.

On the domain layer of the INPROVY framework is a conceptual product model that presents the product components of this concrete injection pump. A product component is somewhat characterised in that it has an item number which can be used to order parts from a supplier or to locate them in a warehouse. The description of the product components is a static view of the machinery that can be referenced in a process view in order to describe an order process.

On the conceptual simulation layer, the simulation components of the concrete injection pump that are the starting point for various simulation applications are described. Examples of such applications are the real time simulation or the detailed simulation of single components (Esswein et al. 2009). This conceptual simulation model is also a static view of the concrete injection pump, however, from a different viewpoint.

In the product model, components such as the cable drum or the water tank are shown separately because of different item numbers. In the conceptual simulation model, if the focus of the simulation is for example on the spray arm, the cable drum and the cleaning water tank are summarised by the model element ‘carrier vehicle’.

Figure 2 presents an example model of the technical layer as well as an example of the simulation model layer. A concrete injection pump is made out of hundreds of parts. Therefore, both models present only an excerpt. Above the example model are the example meta-models (although only in part). Between the two models various integration conflicts exists from which four examples are shown in red.

1. Name conflict: In the technical model the concrete injection pump is noted with the product name, whereas in the simulation model, the company's internal type identifier is used.
2. Type conflict: The swing arm is characterised as type ‘product component’ in the technical model. In the simulation model it is assigned the type ‘simulation component’.
3. Abstraction conflict: the different parts of the spraying arm are connected with hydraulic elements. In the technical model is such an element modelled as a ‘hydraulic cylinder’. In the simulation model this is modelled as ‘differential cylinder’. The latter is a special hydraulic cylinder.
4. Level of detail conflict: Since the different elements of such a differential cylinder are important for the simulation because they influence the behaviour, the piston, the barrel

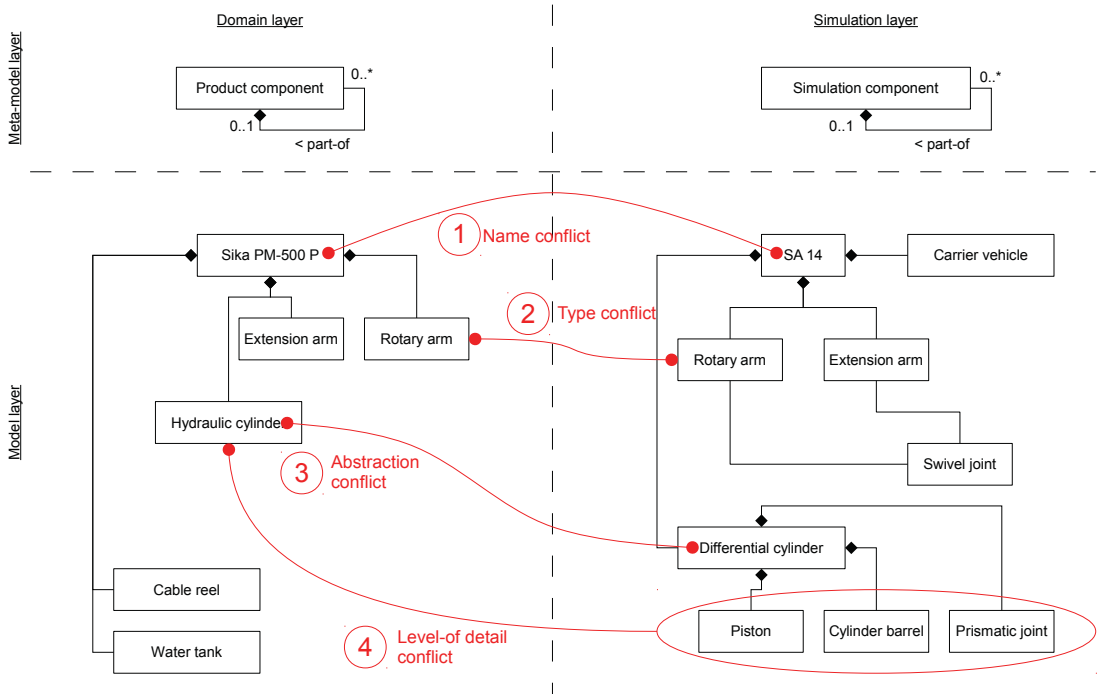

Figure 2: Models of the Case study

and prismatic joint are presented separately in the simulation model. In the technical model, however, the cylinder is modelled as one element because only the part as a whole has an item number.

4.1 Meta-model Based Solution

The first possibility is the integration based on meta-models. For this solution, the relationships between the elements of the two meta-models are modelled. In a first step, a relationship between the element 'product component' and the element 'simulation component' can be modelled.

By modelling such a relationship on the meta-model layer, it is possible to explicate the type conflict described above in that a connection on model layers between the two elements 'product component: rotary arm' and 'simulations component: rotary arm' is modelled.

The remaining integration conflicts, name and structural conflicts, cannot be resolved through the mapping of the original meta-model elements. However, the evaluation in Sect. 3.3.2 identified two design dimensions. One was the semantic refinement of the relationship between the meta-model elements, and the other was the refinement of the domain expression construct that are used in the meta-model.

A refinement of the meta-model means that domain expressions are added to the meta-model. The name conflict can be solved through the meta-model presented in Fig. 3. The example also shows the serious disadvantage of this approach. Such a refinement leads to an extremely complex meta-model whose reusability is limited.

Another design possibility is to refine relationships between the meta-model elements, e.g., a 'specialization relationship'. This could be used on the model layer to describe the abstraction

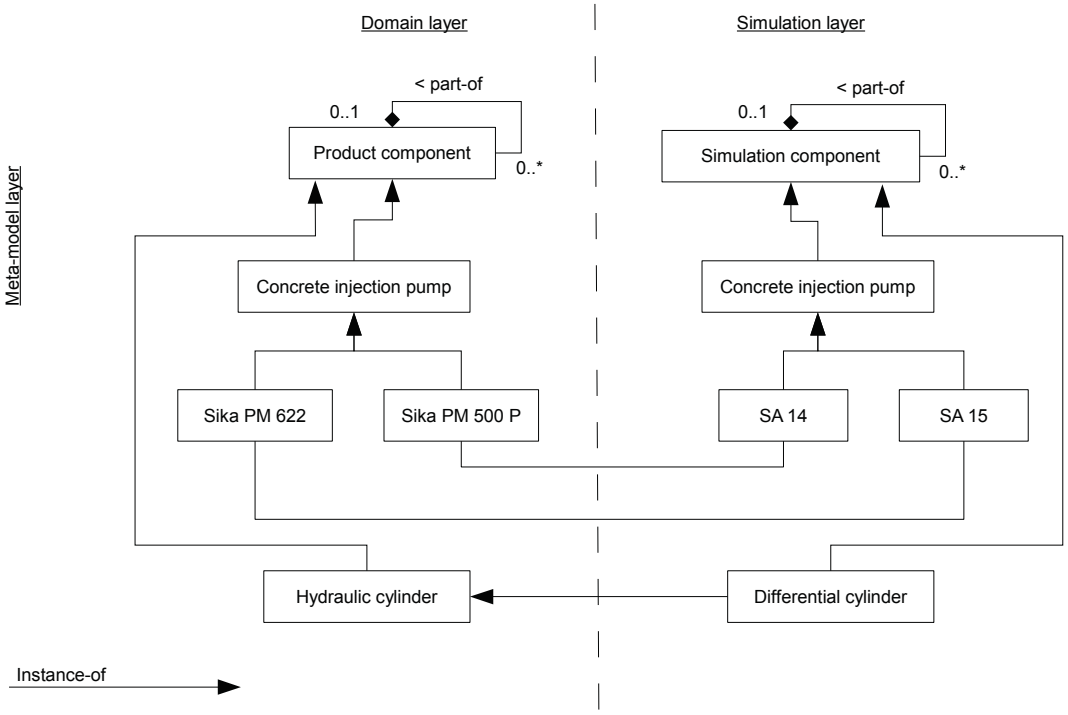

Figure 3: Refined Meta-model-based Solution

conflict between the model element 'Hydraulic cylinder' and the element 'Differential cylinder'. This must be done manually for every model, so with large models it takes disproportionately high effort to be accomplished.

The use of this specialisation relationship at the meta-model layer assumes a refined meta-model so that the relationship between the character classes 'hydraulic cylinder' and 'differential cylinder' can be modelled. Consequently, the meta-model is also hardly to be reused.

Despite these limitations, it can be shown that the refined model relationships and a detailed meta-model are mutually dependent. Because of the manageable implementation effort, these are to be regarded as positive. The conflict between reusability and the usability for resolving conflicts in refined meta-models should be addressed in further research. The separation between a core meta-model and an domain-expression meta-model is conceivable (Kugeler and Rosemann 1998, Kühne 2006).

4.2 Ontology-based solution

The described integration conflicts can also be resolved with the help of explicit ontologies and therefore an indirect integration of the models. In Fig. 4 two example ontologies are presented.

The first ontology describes a correlation between technical language terms from the engineering field. The second describes terms from the domain of mobile construction machinery.

At the model layer, the individual model elements are associated with concepts of the ontologies (shown in Fig. 4 as a dashed line with points). This can, for example, be made partially automatic with help from similarity measurements through the modelling tool (Hofferer 2007).

The naming conflict can be overcome by interpreting the ontologies. Both terms represent a

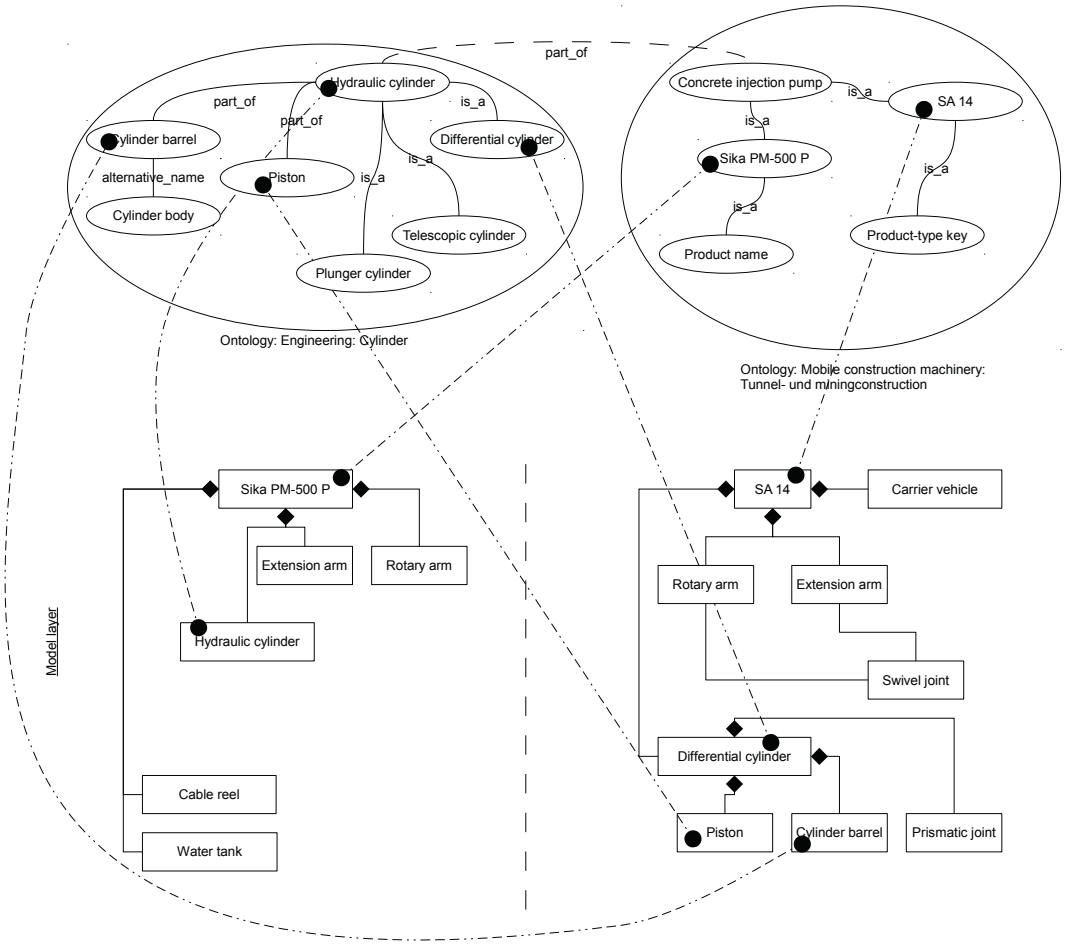

Figure 4: Ontology-based solution

concrete injection pump and differ from each other in that they are either a product name or a type identifier. Alternatively, it would also be possible to model a synonym relationship between 'SA 14' and 'Sika-PM 500 P'. The abstraction conflict between 'hydraulic cylinder' and 'differential cylinder' can be overcome in that the terms are associated with a specialisation relationship in the ontology. The level-of-detail conflict becomes explicit in that the piston and cylinder barrel are both parts of a hydraulic cylinder.

In the example, all the conflicts could be resolved. For the integration, however, two ontologies are needed. They also have to be integrated. In the example, a part-whole relationship between the two concepts was manually modelled (in Fig. 4 this is represented by a long-dashed line). For large ontologies such relationships quickly become numerous and require a high integration effort. By example, the applicability of ontology based approaches to the vertical integration can be proven.

5 Conclusion and Further Research

The evaluation has shown that the approach from Höfferer is the only one that overcomes all integration conflicts, supports reusability and

adaptability and allows for at least a partially automated integration. On the negative side is the high implementation effort.

Furthermore, the overall positive performance of the ontology based approaches must be put into perspective. If it is assumed that the semantics of characters materialise through their use in language, it is not certain that ontologies can map this usage completely and accurately (Gehlert 2007).

Another problem arises from the use of different ontologies. Since at different layers of the framework different terminologies will be used, it is unlikely that all technical terms for all layers of the model in can be defined in a single ontology. All the observed approaches also utilised different ontologies. Hence, the ontologies must be integrated. In this case, similar problems occur as with the integration of the model layers (Visser et al. 1998). The integration problem is therefore only shifted to another layer.

A potential solution could be to utilise the design dimensions identified in the evaluation. Through the use of detailed meta-models and refined integration relationships, the dependencies between the model layers are described through an integration model that will be interpreted by a modelling tool. The enrichment of a meta-model with linguistic expressions heightens the complexity and reduces the reusability. A solution needs to be found that ensures the expandability of a core meta-model. This requires the development of an integration language in which the integration model is created.

Furthermore, the evaluation has shown that the conflicts observed, until now, only in horizontal integrations are also to be overcome in a vertical integration.

All considered approaches are based on the integration of languages. Whether it is an integration of the modelling language through a meta-model based approach or an integration of domain language through an ontology based approach. This is the idea of the tower of babel. Overcoming the language barrier will lead to understanding or in this case to integrated models. However, integration without an integration objective is not very realistic.

6 Acknowledgments

The research and development project 'Integrated product development with virtual Prototypes (INPROVY)' is funded by the Federal Ministry of Education and Research (BMBF) within the framework concept 'Research for the Production of Tomorrow' and promoted by the Karlsruhe (PTKA) under management.[1]

References

Aier S., Riege C., Winter R. (2008) Unternehmensarchitektur – Literaturüberblick und Stand der Praxis. In: WIRTSCHAFTSINFORMATIK 50(4), pp. 292–304

Arnarsdóttir K., Berre A. J., Hahn A., Missikoff M., Taglino F. (2006) Semantic mapping: ontology-based vs. model-based approach Alternative or complementary approaches? In: Proceedings of the Open Interop Workshop on Enterprise Modelling and Ontologies for Interoperability, Luxembourg

Batini C., Lenzerini M., Navathe S. B. (1986) A comparative analysis of methodologies for database schema integration. In: ACM Computing Surveys (CSUR) 18(4), p. 364

Becker J., Pfeiffer D., Falk T., Räckers M. (2010) Semantic Business Process Analysis. In: Jan vom Brocke M. R. (ed.) International Handbook on Business Process Management 1st ed. Springer, Berlin et al., pp. 187–211

Bögel S., Schlieter H., Esswein W. (2011) Compliance Check of Health Care Process Models. In: AMCIS 2011 Proceedings - All Submissions

[1] More information at `http://www.inprovy.de`.

Die Sprecher der Wissenschaftlichen Kommission Wirtschaftsinformatik im Verband der Hochschullehrer für Betriebswirtschaft (WKWI) und des Fachbereichs Wirtschaftsinformatik der Gesellschaft für Informatik (GI-FB WI) (2008) WI-Orientierungslisten. In: WIRTSCHAFTSINFORMATIK 50(2), pp. 155–163

Dijkman R. (2008) Diagnosing Differences between Business Process Models. In: Business process management: 6th international conference, BPM 2008, Milan, Italy, September 2-4, 2008: proceedings Vol. 5240. Springer, Berlin, pp. 261–277

Esswein W., Greiffenberg S., Lehrmann S. (2009) Framework zur modellgestützten Simulation. In: Wissensportal baumaschine.de (2) http://www.baumaschine.de/Portal/Aktuell_0902/Special/simulation/simulation.pdf

Fellmann M., Thomas O. (2009) Management von Modellbeziehungen mit Semantischen Wikis. In: Proceedings der 9. Internationalen Tagung Wirtschaftsinformatik (WI 2009), 25.-27. Februar in Wien. Österreichische Computer Gesellschaft, Wien, pp. 673–683

Fengel J., Rebstock M., Nüttgens M. (2008) Modell-Tagging zur semantischen Verlinkung heterogener Modelle. In: Proceedings der EMISA 2008, Bonn/St. Augustin 18.-19.09.2008. Bonn u.a.

Frank U. (2002) Multi-Perspective Enterprise Modeling (MEMO): Conceptual Framework and Modeling Languages. In: Proceedings of the Hawaii International Conference on System Sciences (HICSS-35). Honolulu, pp. 72–81

Frankel D. S, Harmon P., Mukerji J., Odell J., Owen M., Rivitt P., Rosen M., Soley R. M (2003) The Zachman Framework and the OMG's Model Driven Architecture. In: Business Process Trends (9), pp. 1–14

Gehlert A. (2007) Migration fachkonzeptueller Modelle. Logos, Berlin, XVI, 392 S.

Hahn A. (2005) Integration verteilter Produktmodelle durch Semantic-Web-Technologien. In: Wirtschaftsinformatik 47(4), 278–284

Hars A. (1994) Referenzdatenmodelle: Grundlagen effizienter Datenmodellierung. In: Gabler, Wiesbaden

Heinrich L. J. (2000) Bedeutung von Evaluation und Evaluationsforschung in der Wirtschafsinformatik. In: Heinrich L. J., Häntschel I. (eds.) Evaluation und Evaluationsforschung in der Wirtschaftsinformatik. Oldenbourg, München u.a., pp. 7–22

Hevner A. R, March S. T, Park J., Ram S. (2004) Design science in information systems research. In: Management information systems quarterly 28(1), 75–106

Höfferer P. (2007) Achieving business process model interoperability using metamodels and ontologies. In: Proceedings of 15th European Conference on Information Systems, 1620–1631

Hofferer P. (2007) Achieving business process model interoperability using metamodels and ontologies. In: Proceedings of 15th European Conference on Information Systems, pp. 1620–1631

Holenstein E. (1982) On the cognitive underpinnings of language. In: Semiotica 41(1-4), 107–134

Karagiannis D., Höfferer P. (2008) Metamodeling as an Integration Concept. In: Filipe J., Shishkov B., Helfert M. (eds.) Software and Data Technologies. Communications in Computer and Information Science 10 Vol. 1. Springer, Berlin, pp. 37–50

Kashyap V., Sheth A. (1996) Semantic and schematic similarities between database objects: a context-based approach. In: The VLDB Journal 5(4), 276–304

Kugeler M., Rosemann M. (1998) Fachbegriffsmodellierung für betriebliche Informationssysteme und zur Unterstützung der Unternehmenskommunikation. In: Fachausschuss 5.2 der Gesellschaft für Informatik e. V. (GI): Informationssystem-Architekturen 5, 8–15

Kühn H., Bayer F., Junginger S., Karagiannis D. (2003) Enterprise Model Integration. In: 4th International Conference, EC-Web Prague,

Czech Repbulic, September 2-5, 2003 Proceedings. Lecture Notes in Computer Science Vol. 2738/2003. Springer, Berlin, pp. 379–392

Kühne T. (2006) Matters of (Meta-) Modeling. In: Software and Systems Modeling 5(4), pp. 369–385

Lin Y., Krogstie J. (2009) Quality Evaluation of a Business Process Semantic Annotations Approach. In: International Journal of Interoperability in Business Information Systems (IBIS) 1(3)

Lin Y., Strasunskas D., Hakkarainen S., Krogstie J., Solvberg A. (2006) Semantic Annotation Framework to Manage Semantic Heterogeneity of Process Models. In: Advanced Information Systems Engineering. Lecture Note in Computer Science 4001/2006. Springer, Berlin, pp. 433–446

Mukerji J., Miller J. (2003) MDA Guide Version 1.0.1

Pfeiffer D., Gehlert A. (2005) A framework for comparing conceptual models. In: Desel J., Ulrich F. (eds.) Proceedings of the Workshop in Klagenfurt: Enterprise Modelling and Information Systems Architectures. Klagenfurt, pp. 108 –122

Rosemann M. (2002) Komplexitätsmanagement in Prozeßmodellen. Gabler, Wiesbaden

Sendall S, Köster J. (2004) Taming Model Round-Trip Engineering. In: Proceedings of Workshop on Best Practices for Model-Driven Software Development

Siau K., Rossi M. (1998) Evaluation of information modeling methods-a review. In: Proceedings of the Thirty-First Annual Hawaii International Conference on System Sciences Vol. 5, 314–322

Simon C., Mendling J. (2007) Integration of Conceptual Process Models by the Example of Event-driven Process Chains. In: 8. Internationale Tagung Wirtschaftsinformatik WI, Karlsruhe, Germany, February 28 - March 2, 2007. Universitätsverlag Karlsruhe, Karlsruhe, pp. 677–694

Sinz E. J. (1999) Architektur von Informationssystemen. In: Rechenberg P., Pomberger G. (eds.) Informatik-Handbuch 2nd ed. Hanser, München, pp. 1035–1046

Verschuren P., Hartog R. (2005) Evaluation in Design-Oriented Research. In: Quality and Quantity 39(6), pp. 733–762

Visser P. R., Jones D. M, Bench-Capon T. J. M., Shave M. J. R. (1998) Assessing heterogeneity by classifying ontology mismatches. In: Formal ontology in information systems: proceedings of the First International Conference (FOIS'98), June 6-8, Trento, Italy, pp. 148–162

Weidlich M., Barros A., Mendling J., Weske M. (2009) Vertical Alignment of Process Models-How can we get there? In: Nurcan S., Schmidt R., Soffer P., Ukor R. (eds.) CAiSE 2009 Workshop Proceedings-10th Workshop on Business Process Modeling, Development, and Support (BPMDS). LNBIP Vol. 29, 71–84

Winter R. (2009) Management von Integrationsprojekten: Konzeptionelle Grundlagen und Fallstudien aus fachlicher und IT-Sicht, 1st ed. Springer, Berlin

Wittgenstein L. (2010) Tractatus Logico-Philosophicus. Cosimo, Inc.

Zachman J. A. (1987) A framework for information systems architecture. In: IBM systems journal 26(3), pp. 276–292

Zivkovic S., Kühn H., Karagiannis D. (2007) Facilitate Modelling Using Method Integration: An Approach Using Mappings and Integration Rules. In: Proceedings of the 15th European Conference on Information Systems (ECIS2007). St. Gallen, pp. 2038–2050

Stephan Bögel, Werner Esswein

Chair for Information Systems,
esp. Systems Engineering
Dresden University of Technology
Münchner Platz 3
01062 Dresden
Germany
{Stephan.Bögel | Werner.Esswein}
@tu-dresden.de