Abstract

To analyze and describe the behavior and the interaction pattern of the product’s subsystems and their components, multiple simulation models have to be developed in each engineering domain. In this paper, a Software Product Line (SPL) approach is adopted to demonstrate the dependencies and associations between models of subsystems. SPL approach provides a structured method for handling variability by using feature models. The particular focus of this paper is on the mapping between components in different tools with more than one modeling depth, the level of detail respectively, through deployment of feature modeling.

© 2014 Published by Elsevier Ltd. This is an open access article under the CC BY-NC-ND license (http://creativecommons.org/licenses/by-nc-nd/3.0/).

Keywords: Feature modeling; mechatronic systems; modeling depth; multidisciplinary systems; Software Product Line; variability management

1. Introduction

In the field of mechatronics, products are described by multiple models belonging to different engineering domains such as mechanical, electrical, and electromechanical. These models are often complex and interact with one another in a heterogeneous way. Therefore, to model the entire system, many domain specific tools will be required. Furthermore, because of the highly coupled nature of mechatronic products, change in one of the product’s subsystem typically results in change in the other subsystems in other domains. To simulate such interconnected systems, one integrated and comprehensive simulation system is needed that encompasses multiple domains.

* Corresponding author. Tel.: +49-5251-60-6286; Fax: +49-5251-60-6297.

E-mail address: farisoroosh.abrishamchian@hni.uni-paderborn.de

Peer-review under responsibility of the Organizing Committee of SysInt 2014.

doi:10.1016/j.protcy.2014.09.031
In a simulated model, the interfaces between the defined system components with input and output variables for actuators and sensors must be matched together. This paper proposes a conceptual design for a model configurator for integrated simulation of multidisciplinary mechatronics products. The proposed model ensures a complete coverage of the entire system without overlaps between the subsystems. In addition, the proposed model supports modeling variables with varying depths. Here, as observed by Just et al. [1], a large modeling depth means a very detailed model. Similarly, a simple model with many assumptions and simplifications is said to be simulated using a low modeling depth. Consistency is another requirement for creating valid interfaces that is addressed by the proposed model. In other words, how is following changes of model issues would be discussed in this paper.

To match the submodels together, there are two main issues which need to be addressed:

- matching the interfaces of a component
- the variant of modeling depth and modeling tools within different domains

To handle this variability, Feature Modeling Approach is used to configure the subsystem models. Feature modeling is developed for the description of the product variants in the field of Software Product Line [2]. Here, the variants of modeling depth and variants of modeling tool create a variant model for a component. What differentiate these models of a component from each other are the properties of modeling. The properties of a component are classified in three categories of inputs, outputs and parameters as shown in Table 1. Inputs and outputs are variables, while the parameters are constants of a component. For example, a pneumatic cylinder requires pneumatic pressure as input to give the displacement of piston as an output. The parameters are total mass, piston mass, piston length etc. They are represented as features of feature diagram. The model configurator is based on the concept of feature modeling.

<table>
<thead>
<tr>
<th>Component</th>
<th>Input</th>
<th>Output</th>
<th>Parameter</th>
</tr>
</thead>
<tbody>
<tr>
<td>Interface</td>
<td>Minimal interface</td>
<td>Variant of depths</td>
<td>Variant of domains</td>
</tr>
<tr>
<td>Modeling Depth</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Modeling Tool</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

The outline of remaining paper is as follows. The related works are reviewed in the next section. The fundamentals of feature models are discussed in the third section. Then uses of feature modeling in mechatronic systems and the structure of model configurator described next in Section 4 and 5. The last section provides the concluding remarks and the future works.

2. Related Works

In multiple research and industrial projects such as “Design Methodology for Intelligent Mechatronic Systems” (ENTIME), several methodologies are proposed for model-based design of mechatronic systems. In particular, for automotive applications, several sophisticated methods and models have been developed [3], [4]. In ENTIME ontologies are developed to connect the abstracted simulation models for solution patterns of the early design stage to the detailed physical simulation models of elements of the solution [5], [6].

Berg et al. propose a solution for managing variations at different levels of abstraction for the software product line [7]. This approach can be used in multidisciplinary mechatronic systems. Using these ideas, a conceptual variability model is created in a third dimension to capture the variability of information. This allows a One-to-One mapping between the problem space and the solution space [7].

Herzig et al. have analyzed and discussed the consistancy of sytems [8]. The authors have detected inconsistent points of the system which are not detected normally. Their work categorizes the inconsistencies to manage specific types of consistency issues. Qamar and Paredis have studied the practical approaches for consistency management.
and model management in the presence of dependencies [9]. They suggest that SysML does not provide sufficiently rich language constructs to model consistency. The current version of SysML does not provide explicit diagram for defining and managing variants in software product lines. Subsequently it is not easy to show interactions of components in different tools with different modeling depth.

There are some works, which configure the simulation model of the whole system. However, a concrete solution has not been developed yet for the configuration with a focus on the modeling depth and tools. In the existing configurators, it is not possible to show the interactions of components in different modelling details or in different tools. Till date without special know how of every domain or tools, it is not possible to change a part of a model with another one or create an entire system simulation. This is because the interfaces of models must be matched together. In this paper, it is tried to cover this know how in the presented model configurator by automatically creating the required matching interfaces.

3. Fundamentals of Feature Modeling

Feature modeling originating from the Feature Oriented Domain Analysis (FODA) [10] has become a standard for management of variability in the field of software product line. These techniques provide an easy, understandable, and generic way of representing the variability information, independent of specific application domain [11].

A Feature model is organized in tree structure, which is implemented the feature diagram [12]. The dependencies between features are shown as constraints. There is not a unique solution for implementation of feature models. Feature modeling has several applications in feature based product specification. Although these meta models have considerable structural differences, their core semantics are similar [11]. In this paper a representation is used for variability information of tools based on the Eclipse Modeling Framework, which uses the Feature Oriented Software Development (FOSA) [13] method. Configuration of the variants is done through a choice tree of the eclipse EMF-Model.

In this paper, feature modeling is the base of a simulation configurator, which connects partial simulation models to create a simulation model for the entire system. A feature is defined differently with respect to modeling depth and modeling tool. The proposed approach shows the interactions between different modeling tools at varying modeling depth for different components of a product.

4. Feature Modeling in Mechatronic Systems

To ensure that the components of a product meet the given requirements, they should be modeled with varying levels of detail, called modeling depth. Detailed models are very important during the development process of mechatronic system [14] but not all the components need to be designed with high level of details. The developer has the option to decide between different levels of details for each component [14] at different stages of product development. In this article, this problem is tackled by describing the design of a product variant model [15] in the sense of variant model tools and variant model depth.

One approach for modeling and mapping of subsystem variability is to use a separate solution model, to associate the features (parameter and variables of a component). It is important to have consistency in properties of the components between the multidisciplinary tools. The complexity of relationships is hidden in the configuration tree. Different factors affect the choice of solutions to be used for converting from variation points in the end solution [15].

To connect the tools, the properties of components are classified into the categories of inputs, outputs and parameters. They are presenting the last rows of the hierarchically arranged set of feature trees. According to feature modeling, they are shown within one feature diagram. We extend them for mechatronic systems and test the methodology using an example of a pneumatic sorting machine.
5. Model Configurator

The simulation models that describe a product should be connected to each other in order to address the interactions that exist among various subsystems of the product. The problem of modeling mechatronic products becomes more challenging as the number of components and the functional layers increases both in hardware and software. For efficient modeling and simulation in the mechatronics field, there is a need for tools that can be used for configuring the interfaces between individual components which are simulated in different simulation tools. Model configurator will adopt the present methodology of co-simulation to allow different domains and tools to interact within the product development process. For coupling the simulation models of different components Model Configurator has the task to match interfaces together before co-simulation. The structure of Model configurator has been proposed as a software product line approach based on feature modeling.

We will first look at the model interfaces of each component. It is necessary to introduce every component as a feature to demonstrate the minimal interfaces of the components within a feature model. Each component has a minimal modeling depth, which is represented in an abstract feature [14]. In addition, every component has parameters and variables such as inputs and outputs as shown in Figure 1. Each interface must include at least one input and one output.

The highest level is the interface level with two mandatory sub-features. Modeling depth I contains the time-discrete state machines. There is no physical modeling at this level [14]. Other sub-features are the Inputs/Outputs as minimal interface of the component. In the first three model depths, there are three kinds of sub-features. One of them is the next model depth, and the others are Inputs/Outputs and Parameters. The or-Binary shows that at least one of the sub-feature must be selected. That means that the next depth always has more Inputs/Outputs or Parameters than the previous depth, because it is more detailed than previous one. Model depths are abstract and just Inputs/Outputs and Parameters are concrete features, which their dependencies influence the model, because one depth cannot exist without any Inputs/Outputs and Parameters, shown in Figure 2. Help-features help a parent feature to have different kind of relationship to his child features. Here Depth-feature has Optional- and Or-relationship to his child features; therefore Depth-feature includes Help-feature.

One component can have a variety of models in different tools. To proceed on, it can be assumed that a component is modeled in a powerful tool like Dymola or Matlab/Simulink. Figure 3 shows one possible selection through Alternative (xor) relationship. Use of other tools like CAD-systems, is optional. Each tool has two kinds of mandatory sub-features: Inputs/Outputs and Parameters to model the behavior of a component’s model. Tools themselves are abstract features and the Inputs/Outputs and the Parameters are concrete features, as shown in Fig. 3.

The last two kinds of feature diagrams must be combined in one diagram to consider the whole dependencies of a component’s properties. For realizing a model of a component as a product many varieties of products considering different modeling depths and different tools need to be considered.

In addition to the logical dependency of the feature elements, constraints between features can be used to show the dependencies which are not hidden in the logic of the diagram. For example in Figure 3, there is a constraint to define the dependency of Inputs/Outputs of the CAD-Model from the same feature of mathematical or topological model. It is proposed to keep a consistency between these two features if one of them is changed. If the
dependencies are represented over the feature model and a constraint through redundancy marking, they could be deleted because they have no effect on the system. This means there are already dependencies of the feature model but could not be detected directly.

Fig. 2. Feature model of depths.

Fig. 3. Feature model of tools.
FeatureIDE [13] created a configuration editor to facilitate the generation of software systems based on a selection of features. A configuration is a subset of all features defined in the feature model. A configuration is valid if the combination of features is allowed by the feature model (i.e. it fulfills the semantics of groups and all cross-tree constraints) [13]. As shown in Figure 4, the validity of configuration can be verified. Filled squares show that the feature cannot be selected. Some selective features, depending on the user’s selections, cannot be deselected anymore. In Figure 4, a green color represents a feature which must be selected. When one of the parent features is selected, the other one is not selectable, as shown in Figure 4(b). There is a dependency between the mathematical/topological model and the CAD model, shown earlier as a constraint in Figure 3. This dependency is demonstrated through green color for features in Figure 4(b). The desired simulation system is implemented by mapping of the necessary interfaces based on a valid selection of features, as shown in Figures 4(c) and 4(d).

In this way, there is an opportunity to represent a combination of varying depths and tools in a feature diagram. Figure 5 presents a small part of pneumatic cylinder as an example to show the interactions between inputs, outputs and parameters dependent to their depth and tool. As an example, force is an acceptable input in RecurDyn while pneumatic pressure is a possible input in Matlab/Simulink. This dependency between these two inputs is demonstrated in a constraint, as shown in Figure 5. This constraint doesn’t change the model’s logic, because it is hidden in the logical results of the cylinder’s feature model. Here, for example in case of double acting and single acting cylinders, there are differences between diverse kinds of the component. In the configurator, the one which is being used can be selected. Appropriate to this selection, an automatic selection of inputs, outputs and parameters are executed. In a configuration of an end product, just the compatible components can be selected in this work. Hardware variability will be handled in future.
6. Conclusion and Future Works

In this paper, a concept for a model configurator is presented, which configures the interface of the subsystems to prepare them for matching them together. The model configurator enables the user to establish a simulation for the entire system. A methodology for configuring the system’s simulation model with adopting the approach of Software Product Line is presented. This is achieved by modeling a system as subsystem in different domains. This approach involves handling the present subsystem’s model with various depths in different tools by using feature models. This paper shows the possibility to connect them together by considering their interactions. By realizing of variety of modeling depth and modeling tool as feature, a configurator was created, which gives the ability to select desired features and create a valid configuration of the model. This configurator shows the user which inputs, outputs and parameters must be defined such that the present subsystems can be connected to one another. If there is not a desired model depth in desired tool, it prepares the necessary inputs, outputs and parameters of absent subsystem. Then user can create the absent part of model with considering of them.

In the future, the meta model of feature modeling will be expanded such that the dependent variables could be defined as proportional equations. Currently, different variants are produced manually and it must be automated in future. The next step of this research is to manage the variety of hardware by introducing the dependencies between hardware and their models.

![Feature model of cylinder](image-url)
Acknowledgements

This work is developed in the project “Systems Engineering”, promoted by Federal Ministry of Education and Research in the context of Spitzencluster, “Intelligente Technische Systeme OstWestfalenLippe (it’s OWL)” (Intelligent technical systems OstWestfalenLippe). This is supervised by Promoter Karlsruhe (PTKA).

References