An Extension for the Specification and Automated Selection of System Variants Based on the System Entity Structure Using a Problem from Process Industry

Hendrik Folkerts[1*], Thorsten Pawletta[1], Umut Durak[2]

[1]Research Group Computational Engineering and Automation, University of Applied Sciences Wismar, Philipp-Müller-Straße 14, 23966 Wismar, Germany; *hendrik.folkerts@hs-wismar.de

[2]Institute of Informatics, Aeronautical Informatics, Technical University Clausthal, Albrecht-von-Groddeck-Straße 7, 38678 Clausthal-Zellerfeld, Germany

SNE 33(1), 2023, 17-25, DOI: 10.11128/sne.33.tn.10633
Selected ASIM SST 2022 Postconf. Publication: 2023-02-01;
Received Revised Improved: 2023-03-04; Accepted: 2023-03-10
ISSN Print 2305-9974, Online 2306-0271, www.sne-journal.org

Abstract. Modeling and Simulation (M&S) is widely used in various fields of engineering, manufacturing or process industry to investigate different system variants. An increasing problem of M&S is the complexity and variety of system variants. This concerns on the one hand the modeling effort and on the other hand the management of system variants in simulation studies. A general approach to the specification of different system variants is offered by the System Entity Structure (SES). It describes a set of system designs with different system structures and parameter configurations. In combination with a Model Base (MB), an SES can be used to describe different configurations of simulation models. The SES/MB framework and extended software architectures based on it define methods for the automated selection of system/model variants as well as for the generation and execution of simulation models. In this paper, the special descriptive element of the SES, the multi-aspect, is discussed. It is shown, how with hierarchically arranged multi-aspects certain forms of system variants can be specified very efficiently. Furthermore, a method for an automated derivation of system variants is presented in the context of hierarchical multi-aspects. This is important for the automation of simulation studies. For a practical illustration of the general method, it is presented using a problem from the process industry.

Introduction

Today's systems are often characterized by a high degree of variability. Variability modeling means to describe several system configurations. A system configuration represents one variant characterized by a system structure and parameter settings. Zeigler [1] introduced with the System Entity Structure (SES) a general high level approach for variability modeling. To describe and manage different configurations of simulation models, the SES was combined with a Model Base (MB) and extended to an SES/MB framework [2, 3]. The MB is a repository for organizing a set of basic dynamic models. Moreover, the framework specifies two general methods: (i) the pruning method for selecting specific system configurations from an SES and (ii) the build method for generating executable Simulation Models (SMs). The general framework does not define concrete algorithms for these methods.

Since the introduction of the SES/MB framework, it has been continuously developed by different researchers, such as presented in [4, 5, 6, 7, 8, 9, 10]. There are several approaches to the pruning method. Originally, pruning is an interactive process. However, interactive pruning is costly and error prone for a high number of variants coded in an SES [6]. Therefore, automation of the pruning process is crucial.

The automated goal-driven selection of system variants is a prerequisite for automating simulation studies involving different system configurations. Accordingly, Schmidt [9] proposes an extended SES/MB-based software architecture to automate simulation studies in the MATLAB/Simulink environment. Based on [9], a software architecture is developed in [10] that supports the generation of executable simulation models for different target simulators. Among other things, this is based on the use of the Functional Mock-up Interface (FMI). In particular, the build method and the MB have been further developed so that, in addition to the SES, the MB is also largely simulator-independent.

A long unsolved problem has been the automated derivation of system configurations from an SES when using multi-aspects hierarchically in an SES [5, 6]. Zeigler and Hammonds [5] propose restructurings of the SES for this purpose. In [11], the authors developed an alternative approach which, in their view, is much easier to implement. This paper focuses on the basic approach in [11]. SES modeling with multi-aspects is introduced step by step and deepened by means of an example from the process industry. Before that, basic aspects of SES' are briefly discussed and an SES/MB-based software architecture is presented, in which the pruning method has been integrated to automate simulation studies.

1 Some Basics of SES

An SES is a tree structure with entity nodes, descriptive nodes, and attributes. While entity nodes describe an object of the real or imaginary world, descriptive nodes specify the relations among at least two entities. The descriptive nodes are divided into aspect, multi-aspect, and specialization node types. Aspect and multi-aspect nodes describe the composition of an entity. Coupling relations between entities can be specified in a special attribute (*couplings*). The multi-aspect node is a special aspect node that specifies a composition of several entities of the same kind. Number of Replications (numRep) is an additional attribute, which can be used to define a variable number of entities. Specialization nodes specify the taxonomy of an entity. For automated pruning specialization nodes have to define a selection rule as attribute. Zeigler [1, 5] defined six axioms for the construction and pruning of an SES. Applying the axioms, it follows among other things, that the root node is always an entity, representing several or one system configuration. The leaf nodes represent entities that are not further decomposed. Like descriptive nodes, entity nodes can specify attributes to define characteristic features. For example, as Schmidt [9] shows, it is useful to define the reference to a model in an MB and its parameter settings as attributes of a leaf node.

In terms of variant and variability modeling, multi-aspect and specialization nodes represent variation points. In order to derive one specific system configuration by pruning an SES all variation points have to be resolved by evaluating the node attributes. However, not only attributes at nodes of variation points have to be evaluated. Pruning can also involve adjustments to coupling relationships or attribute changes to entity nodes when resolving specializations. The result of each pruning operation is a Pruned Entity Structure (PES). A PES codes exactly one system configuration. There are several ways to implement a pruning method [4, 6]: (i) interactive pruning, (ii) automated pruning, (iii) enumerative pruning to derive all possible variants, (iv) selective pruning to derive one variant, etc. This paper focuses on the last one, the automated, goal-directed selection of exactly one system variant by pruning. For this purpose, we use the extension of the SES with information about pruning in the node attributes according to [8, 9, 12].

2 An Extended SES/MB-based Software Architecture

For applications in the field of Modeling and Simulation (M&S), the SES/MB framework was introduced [2, 3]. Basic dynamic models, which are referenced from leaf nodes in an SES, are organized in an MB. In addition to the pruning method discussed previously, the framework must provide a build method for generating executable SMs. In the following, we briefly present an extended SES/MB-based software architecture, which supports an automation of simulation studies.

The architecture is shown in Figure 1. The concept of the architecture was introduced in [8]. Besides the two new components, Experiment Control (EC) and Execution Unit (EU), also some new SES features like SES variables (SESvars) and SES functions (SESfcns) have been introduced.

The EC is a higher-level control unit. It defines the experiment goals, steps, and settings. Experiment steps and settings can reactively depend on previous simulation results. The EC activates the other components and evaluates their operations. The EU is an interface to target simulators on which the generated SM is executed. It is a kind of wrapper and uses the Application Programming Interface (API) of a target simulator to execute the simulation and to collect results. The SES/MB framework provides an interface to communicate with

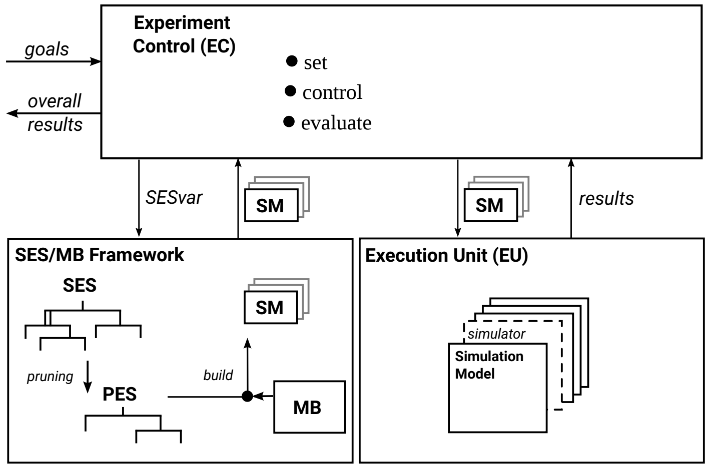

Figure 1: Extended SES/MB-based software architecture.

the EC. The newly defined SESvars are used as input interface. As output, the framework returns the SM derived by pruning and generated by the build method to the EC.

Using SESvars, value assignments to attributes of the SES can be defined variably and depending on settings in the EC. The value of the SESvars is determined before a pruning operation depending on the experiment specification in the EC. With regard to the SES, the SESvars are variables with a global scope. The same applies to the newly introduced SESfcns. SESfcns allow the specification of procedural knowledge and can be called in attributes of the SES. A typical application is the definition of dynamic coupling relations [8].

Currently there are two implementations of the architecture, which are freely reusable [13, 14].

3 Modeling and Pruning of SES with Multi-Aspect Nodes

Multi-aspect nodes are a powerful modeling element. However, pruning without user interaction is challenging for multi-aspects with a succeeding specialization or for several multi-aspect nodes in one path. Unlike the other descriptive nodes, pruning a multi-aspect does not lead to a reduction of the tree, but to an expansion due to the replication of the following entity node.

Restructuring SES' to avoid hierarchies of multi-aspects and specializations as suggested in [5] is challenging to automatize. Additional attributes may be needed for the restructured SES describing the same set of system configurations and it is doubtful how values can be assigned to these attributes during pruning. In the next subsections it is demonstrated how hierarchies of multi-aspect nodes in combination with specialization nodes can be pruned automatically.

For illustration, we use a problem from the field of process industry, which is stepwise extended. Flexibilization in the process industry through modularization leads to a large number of process alternatives and parameter variants in plant design and operation [15]. On the other hand, modularization increases the reusability of models or model components [16]. The variant management in modular plant design according to [17] could be efficiently handled with the SES/MB approach.

3.1 Single Multi-Aspect

The simplest case is an SES with a single multi-aspect in a path as depicted in Figure 2. Here, the SES specifies a plant system that can consist of any number of identical partial plants.

In the SES tree, the root entity *plant* is followed by the multi-aspect node *plantMASP* with the attributes *numRep* and *couplings* and this is followed by the entity node *partialPlant* with the attribute *_partialPlant*. The *numRep* attribute at node *plantMASP* describes the varying number of *partialPlant*s and the *couplings* attribute describes their coupling relations.

Above the SES tree the SESvar *NumPartialPlants* is defined as input interface. The following Semantic Condition describes the permissible value range of the

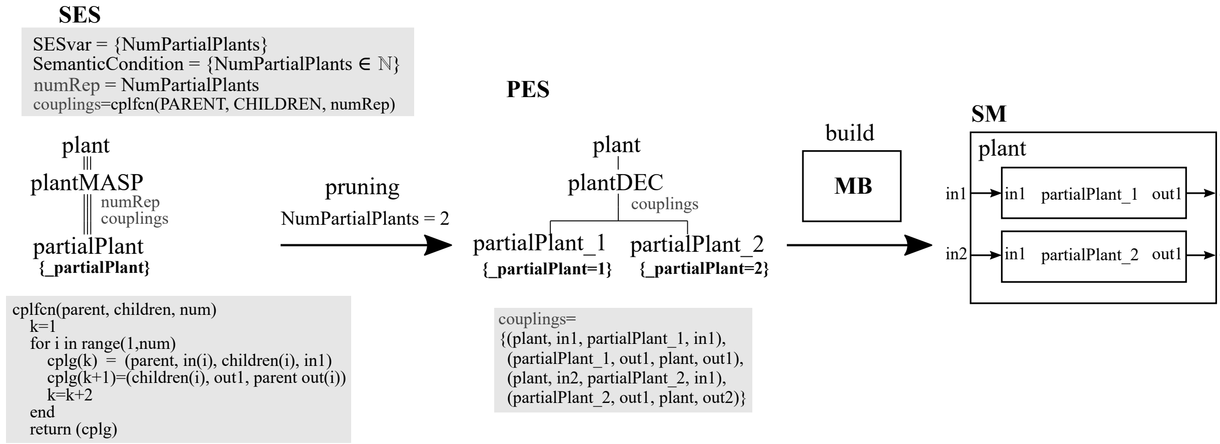

Figure 2: SES with a single multi-aspect, derivation of a possible PES, and the resulting model.

SESvar *NumPartialPlants*. After that the attribute *numRep* is defined using the SESvar *NumPartialPlants* and the *couplings* attribute using an SESfcn. The SESfcn *cplfcn* is called with the implicit variables *PARENT* and *CHILDREN*, and the *numRep* attribute as input parameters. The implicit variables refer to the parent and children nodes of the multi-aspect. The SESfcn specifies a parallel connection of partial plants. The attribute values were not defined directly in the tree only for reasons of clarity.

Before pruning, the SESvar must be assigned a value to initialize the attribute *numRep*. Then, during pruning the entity node *partialPlant* is replicated according to the current value of its attribute *numRep*. Since entities are replicated at a multi-aspect node, the entities following a multi-aspect are called generating entity in [5]. We propose to add at a generating entity node an attribute starting with an underscore followed by the name of the generating entity. The value of this underscore attribute remains undefined in the SES. Pruning implicitly assigns a value that represents a numbering of the generated entities. Thus, the entities are distinguishable based on the attribute value. The reason of this procedure is demonstrated in the next subsection.

Under the SES, Figure 2 shows the derivation of a possible PES by pruning. In the example, the value two was assigned to the SESvar *NumPartialPlants* before pruning. Accordingly, two replications of the entity node *partialPlant* are generated with the names *partialPlant_1* and *partialPlant_2*, and their attribute values are implicitly assigned to them. Also, the multi-aspect node is converted into an aspect node and the *couplings* attribute is computed using the SESfcn. Thus, the aspect node describes the parallel composition of the two partial plants.

The derived PES describes exactly one system configuration. If one extends the leaf node *partialPlant* by an attribute with a link to a basic model in an MB, an SM could be generated with the build method as shown in the resulting model. To focus on the new extensions, the specification of coupling relations and some other attributes, such as references to the MB, are omitted in the following subsections and the build method is not considered.

3.2 Multi-Aspect with Succeeding Specialization

The example from Section 3.1 is now extended in the form of describing configurations of a plant system consisting of a variable set of subplants with identical input/output interfaces, but which are structured internally differently. The two subplant types of power station and waste treatment are used as an example. The further structural decomposition of the subplants is neglected here, but it is shown in the next subsection in Figure 4.

Figure 3 shows in the left part the specification of the problem with an SES. A specialization node called *partialPlantSPEC* is added to the SES following the generating entity *partialPlant* of the multi-aspect *plantMASP*. The specialization node defines a taxonomy of its parent node *partialPlant*, which can be assigned to either the *power station* or *waste treatment* category.

The child entity nodes describe the categories. How the category is assigned when pruning the SES is defined in the *specrule* attribute of the specialization.

The box above the SES tree in Figure 3 defines the input interface and the two attributes *numRep* and *specrule* of the SES. The input interface has been extended by the SESvar *PartialPlantTypes*. The Semantic Condition specifies that the SESvar *PartialPlantTypes* is a vector whose dimension must correspond to the value of the SESvar *NumPartialPlants* and whose elements can have the values *'ps'* or *'wt'*.

The Semantic Condition is followed by the definition of an SESfcn and the two SES attributes. The attribute *numRep* is defined analog to the example in Figure 2. The *specrule* attribute assigned to the *partialPlantSPEC* node defines rules for selecting the partial plant category using the SESfcn.

The middle and right tree in Figure 3 show step-by-step results of a pruning operation. In the example shown, the SESvar were previously assigned as follows: $NumPartialPlants = 2$ and $PartialPlantTypes = ['ps','wt']$.
In the 1st pruning step the multi-aspect *plantMASP* is resolved and the two entities *partialPlant_1* and *partialPlant_2* are generated. The subsequent subtree starting with the specialization node is appended to each of the generated entities. The result of this step is called *intermediate PES*. In the 2nd step the variation point specified by the specialization *partialPlantSPEC* is resolved. Pruning a specialization results in the union of the parent node with one selected child. Children, which are not selected, are removed like the specialization node itself. The selection is controlled by the rules in the *specrule* attribute. In this example, the SESfcn *ppTypes* is called in the specrule. It receives as input arguments the value of the underscore attribute *_partialPlant* of the parent node and the vector defined in the SESvar *PartialPlantTypes*. Depending on the value in *_partialPlant* an element of the vector *PartialPlantTypes* is returned, which describes the category to be selected and thus the selection of a child node. The selected child node and the parent node are merged into one entity node, as indicated by the merged node name in the PES.

The underscore attribute on the generating entity of a multi-aspect and the implicit value assignment during pruning when generating the entities makes branches in the tree distinguishable. The distinguishability enables the subsequent automated assignment to different categories when resolving the specialization node.

3.3 Several Multi-Aspects and Specializations in a Common Path

The introduced example is now extended to two multi-aspects and two specializations in a common path. A third category of subplant, called *chemicalProduction*, is introduced. A plant may comprise several subplants of this type. A subplant *chemicalProduction* may in turn consist of a varying number of further subplants, called *chemicalSubProduction*, serving either *acid* or *base* production. This extension increases the number of possible system configurations exponentially.

Figure 4 shows the specification of the extended problem with an SES and Figure 5 shows step by step the pruning to derive a possible PES.

The SES. The first four layers of the tree correspond to the SES in Figure 3. In the fourth layer, the entity node *chemicalProduction* was added as a further category of a *partialPlant*. The configuration of the *powerStation* and *wasteTreatment* type subplants is not illustrated.

The multi-aspect *chemicalProductionMASP* with the subsequent entity node *chemicalSubProduction* describes the composition of any entity *chemicalProduction* from interface-compatible entities *chemicalSubProduction*. Through the subsequent specialization node *chemicalSubProductionSPEC*, a categorization of each entity *chemicalSubProduction* into *acid* or *base* production is performed.

The box above the SES tree specifies the necessary SESvars, SESfcns, and node attributes. Their semantics are explained below in the step-by-step description of pruning to derive exactly one system configuration.

A pruning example. The SES input interface is extended by the two SESvars *NumChemicalSubProductions* and *ChemicalProductionTypes*, which are formally defined in the Semantic Condition. To derive a system variant, the four variables must be assigned values before pruning. We derive a system configuration as depicted in Figure 5.

1st and 2nd pruning steps: The 1st pruning step is performed analogously to Section 3.1. According to the *numRep1* attribute at the *plantMASP* node, four entity nodes *partialPlant* are generated. Then, the 2nd pruning step is executed analogously to Section

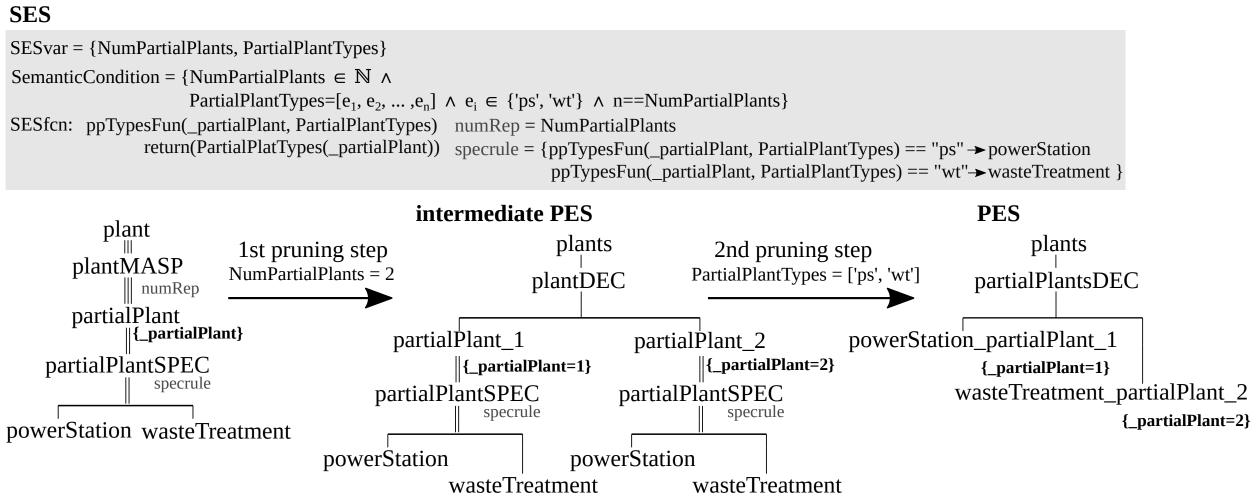

Figure 3: SES with a multi-aspect followed by a specialization and stepwise derivation of a possible PES.

3.2. By pruning the node *partialPlantSPEC* with the attribute *specrule1*, the four previously created entity nodes *partialPlant_1 ... partialPlant_4* are specialized according to the value assignment of the SESvar *PartialPlantTypes* to: *powerStation_partialPlant_1*, *chemicalProduction_partialPlant_2*, *chemicalProduction_partialPlant_3*, and *wasteTreatment_partialPlant_4*. The last operation of the 2nd pruning step is to attach the remaining subtree of the SES to the two entity nodes *chemicalProduction_partialPlant_2* and *chemicalProduction_partialPlant_3*. The result of these pruning steps is illustrated in Figure 5 as *intermediate PES 2*.

3rd pruning step: In the third pruning step the two nodes *chemicalProductionMASP* are resolved. For each of the two nodes, the number of replications of the *chemicalSubProduction* node to generate is computed by the attribute *numRep2* using the SESfcn *cpNumFun*. The principle operation of *cpNumFun* corresponds to *ppTypesFun* in Section 3.2 (Figure 3). Function *cpNumFun* evaluates the underscore attribute *_partialPlant* at the parent in the *intermediate PES 2* as well as the SESvar *PartialPlantTypes* and *NumChemicalSubProductions* and calculates the number of entities to be generated. Besides, an underscore attribute is implicitly added for each *chemicalSubProduction_i* node generated. The variability of a multi-aspect is resolved after entity replication. It is renamed to an aspect and the remaining subtree of the SES is attached to each node created. In this case, the subtree starts with *chemicalSubProductionSPEC*. The result of the 3rd pruning step is called *intermediate PES 3* in Figure 5. It should be noted that the couplings attribute must also be adjusted, as shown in Section 3.1.

4th pruning step: In the fourth pruning step, the specialization *chemicalSubProductionSPEC* is resolved for each entity *chemicalSubProduction_i*. This results in the categorization of each *chemicalSubProduction_i* into an *acid* or *base* production. The value assignments of the SESvar *ChemicalProductionTypes* define the system configuration to be derived in this respect. The necessary selection rules are variably defined in the *specrule2* attribute of the SES using the SESfcn *cpTypesFun*. The operation of *cpTypesFun* corresponds to the previous explanations of the SESfcn. The return value *'ac'* or *'ba'* decides which specialization to select. Deleting the specialization node and uniting the selected child node with the parent node is done analogously to Section 3.2. The final result of pruning is shown with the PES in Figure 5.

Short evaluation. The implicitly managed underscore attribute on entities generated during pruning in combination with SESfcns allows automated derivation of a system configuration (PES) specified with SESvar. However, the complexity of SESfcns increases significantly as the number of multi-aspects in a path increases. The axiom of uniformity specified for the SES states: nodes with the same name need to have the same variables and isomorphic subtrees. In the SES this axiom is fulfilled, but relaxed in the PES. In Figure 5 the nodes *chemicalProductionDEC* do not have isomorphic

SES

SESvar = {NumPartialPlants, PartialPlantTypes, NumChemicalSubProductions, ChemicalProductionTypes}

SemanticCondition = {NumPartialPlants $\in \mathbb{N}$ ∧
PartialPlantTypes=$[e_1, e_2, \ldots ,e_n]$ ∧ $e_i \in$ {'ps', 'cp', 'wt'} ∧ n==NumPartialPlants ∧
NumChemicalSubProductions=$[e_1, e_2, \ldots ,e_n]$ ∧ $e_i \in \mathbb{N}$ ∧ n==|{x | x ∈ (PartialPlantTypes=='cp')}| ∧
ChemicalProductionTypes=$[e_1, e_2, \ldots ,e_n]$ ∧ $e_i=[a_1, a_2, \ldots ,a_m]$ ∧ n==|NumChemicalSubProductions|
∧ $a_i \in$ {'ac', 'ba'} ∧ m==NumChemicalSubProductions(e_i) }

SESfcn: ppTypesFun(_partialPlant, ppTypes)
#map _partialPlant to ppTypes
#return type_of_partial_plant

cpNumFun(_partialPlant, ppTypes, cpNum)
#map _partialPlant to ppTypes
#if ppType is 'cp'
map to cpNum
return number_of_chemical_subProductions

cpTypesFun(_partialPlant, _chemicalSubProduction, ppTypes, cpTypes)
#map _partialPlant to ppTypes
#if ppType is 'cp'
map _partialPlant and _chemicalSubProduction to cpTypes
return type_of_chemical_subProduction

numRep1 = NumPartialPlants

specrule1 = {ppTypesFun(_partialPlant, PartialPlantTypes) == "ps" → powerStation
ppTypesFun(_partialPlant, PartialPlantTypes) == "cp" → chemicalProduction
ppTypesFun(_partialPlant, PartialPlantTypes) == "wt" → wasteTreatment }

numRep2 = cpNumFun(_partialPlant, PartialPlantTypes, NumChemicalSubProductions)

specrule2 = {cpTypesFun(_partialPlant, _chemicalSubProduction, PartialPlantTypes, ChemicalProductionTypes) == "ac" → acid
cpTypesFun(_partialPlant, _chemicalSubProduction, PartialPlantTypes, ChemicalProductionTypes) == "ba" → base}

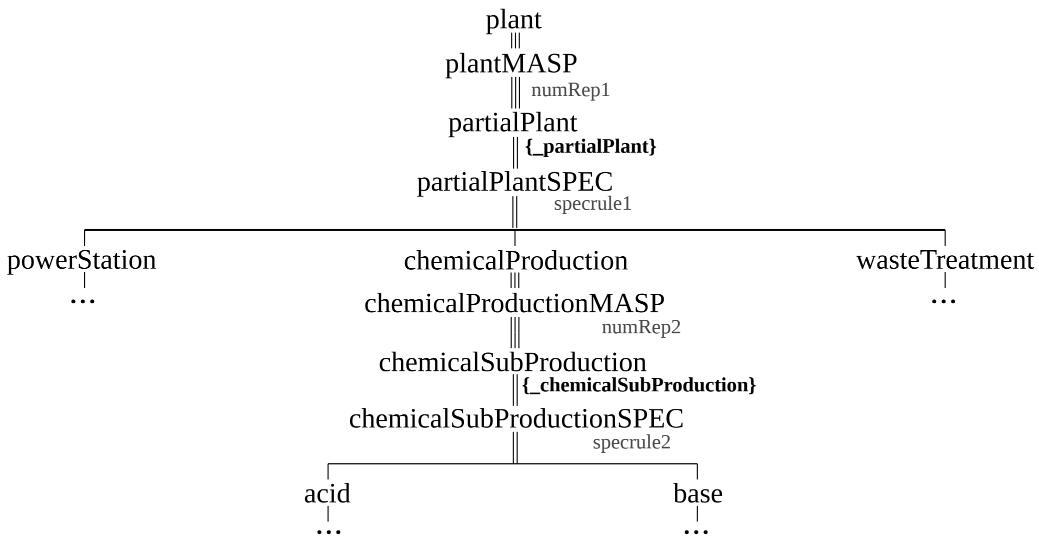

Figure 4: SES with two multi-aspects in one path and each followed by a specialization.

subtrees. In the context of simulation engineering this does not pose any problem. The extended pruning approach is implemented in the Python-based toolset [14].

4 Conclusion

The example has shown that the combination of multiple multi-aspects with subsequent specializations in an SES path supports a compact specification of a large number of system variants. The introduction of an implicitly managed attribute on entities generated by multi-aspects during pruning supports an automated derivation of a goal-directed system configuration (PES). Thus, an automatic model generation using the extended SES/MB architecture is supported.

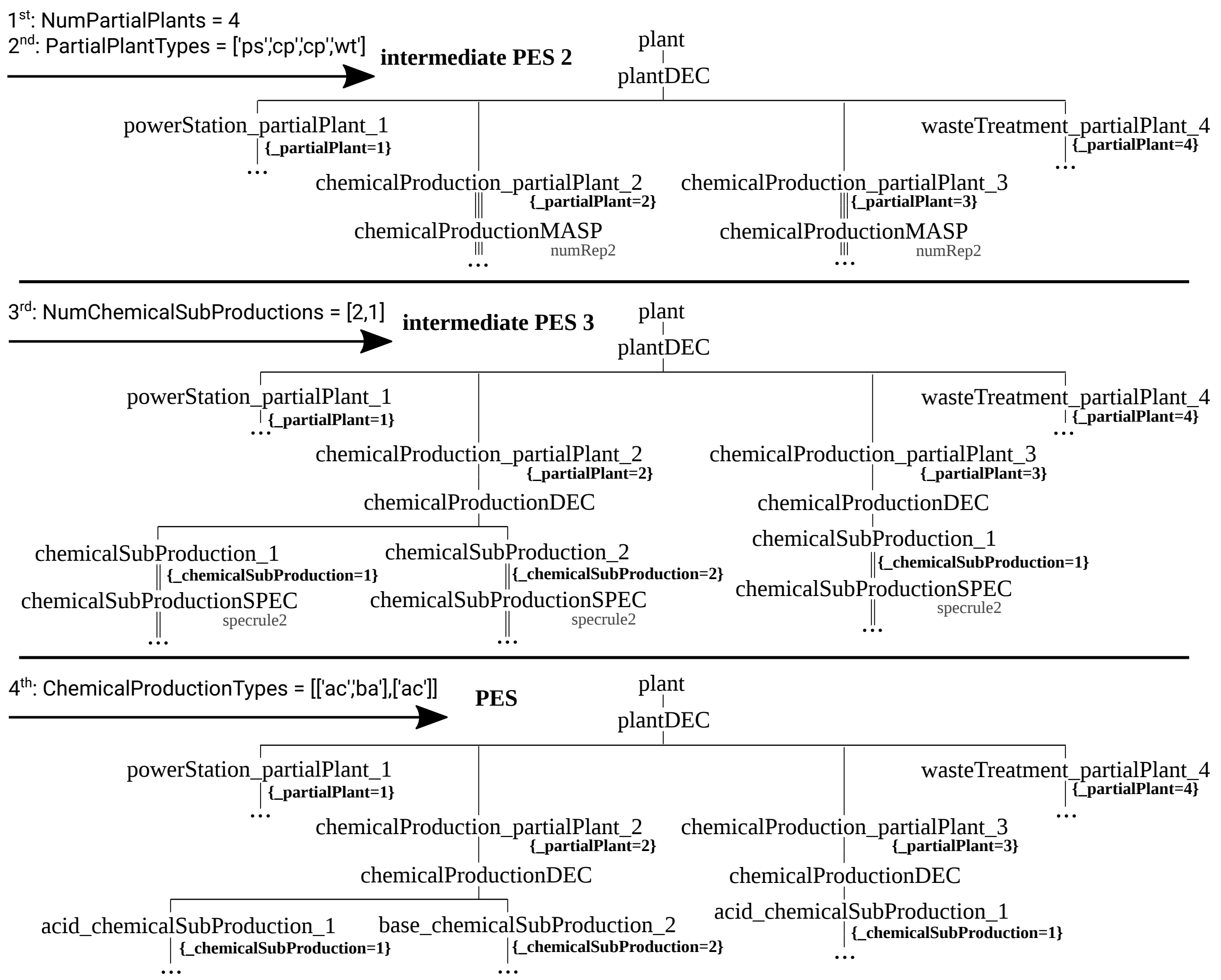

Figure 5: Step-by-step derivation of a system variant by pruning.

References

[1] Zeigler BP. *Multifacetted Modelling and Discrete Event Simulation*. USA: Academic Press Professional, Inc. 1984.

[2] Rozenblit JW, Zeigler BP. Representing and constructing system specifications using the system entity structure concepts. In: *Proceedings of the 25th Winter Simulation Conference, Los Angeles, California, USA, December 12-15, 1993*, edited by Evans GW, Mollaghasemi M, Russell EC, Biles WE. ACM Press. 1993; pp. 604–611.

[3] Zeigler BP, Kim TG, Praehofer H. *Theory of Modeling and Simulation*. USA: Academic Press, Inc., 2nd ed. 2000.

[4] Rozenblit JW, Huang Y. Rule-Based Generation of Model Structures in Multifaceted Modeling and System Design. *INFORMS J Comput*. 1991;3(4):330–344.

[5] Zeigler BP, Hammonds PE. *Modeling & Simulation-Based Data Engineering: Introducing Pragmatics into Ontologies for Net-Centric Information Exchange*. Orlando, FL, USA: Academic Press, Inc. 2007.

[6] Zeigler BP, Sarjoughian HS. *Guide to Modeling and Simulation of Systems of Systems*. Simulation Foundations, Methods and Applications. Springer. 2013.

[7] Santucci JF, Capocchi L, Zeigler BP. System Entity Structure Extension to Integrate Abstraction Hierarchies and Time Granularity into DEVS Modeling and Simulation. *Simulation*. 2016;92(8):747–769.

[8] Pawletta T, Schmidt A, Zeigler BP, Durak U. Extended Variability Modeling Using System Entity Structure Ontology Within MATLAB/Simulink. In: *Proceedings of the 49th Annual Simulation Symposium*, ANSS '16. San Diego, CA, USA: Society for Computer Simulation International. 2016; pp. 22:1–22:8.

[9] Schmidt A. Variant Management in Modeling and Simulation Using the SES/MB Framework. Ph.D. thesis, Rostock University. 2019.

[10] Folkerts H, Pawletta T, Deatcu C. Model Generation for Multiple Simulators Using SES/MB and FMI. *SNE Simulation Notes Europe*. 2019;31(1):25–32.

[11] Folkerts H, Pawletta T, Deatcu C, Zeigler BP. Automated, Reactive Pruning of System Entity Structures for Simulation Engineering. In: *Proceedings of the 2020 Spring Simulation Conference*, SpringSim '20. San Diego, CA, USA: Society for Computer Simulation International. 2020; .

[12] Deatcu C, Folkerts H, Pawletta T, Durak U. How to Define SES Trees for Variability Modeling. *SNE Simulation Notes Europe*. 2019;29(3):117–126.

[13] RG CEA. MATLAB SES Tbx. `https://github.com/cea-wismar/SES_Tbx_Matlab`. 2023. Accessed Feb. 28, 2023.

[14] RG CEA. Python-based SES/MB Architecture. `https://github.com/cea-wismar/SESMB_Inf_Python`. 2023. Accessed Feb. 28, 2023.

[15] DECHEMA. Modulare Anlagen – Flexible chemische Produktion durch Modularisierung und Standardisierung - Status quo und zukünftige Trends. *Tech. rep.*, Germany. 2017.

[16] von Wedel LA. An Environment for Heterogeneous Model Management in Chemical Process Engineering. Ph.D. thesis, RWTH Aachen. 2004.

[17] Hady L, Wozny G. Multikriterielle Aspekte der Modularisierung bei der Planung verfahrenstechnischer Anlagen. *Chemie Ingenieur Technik*. 2012;84:597–614.