AIAA SciTech Forum
7-11 January 2019, San Diego, California
AIAA Scitech 2019 Forum

10.2514/6.2019-1712

Tools for Scenario Development Using System Entity Structures

Bikash Chandra Karmokar *
Leibniz University Hannover, Hannover, 30167, Germany

Umut Durak†
German Aerospace Center (DLR), Braunschweig, 38108, Germany

Shafagh Jafer‡,
Bharvi Chhaya§
Embry-Riddle Aeronautical University, Daytona Beach, FL, 32114-3900

and
Sven Hartmann¶
Clausthal University of Technnology, Clausthal-Zellerfeld, 38678, Germany

Despite the gravity of scenarios in modeling and simulation, there is a lack of common tools, formats and practices. This eventually impairs shareability and interoperability. There is a recent interest in aviation domain for the standardization of scenario definition languages. System Entity Structure (SES) is proposed as a formal approach for a standard simulation scenario definition language. SES provides formal means to capture concepts and their relations in a domain. It enables specifying family of concept structures and properties which can then be pruned to a Pruned Entity Structure (PES). Simulation scenario definition language is formally specified as an SES and its pruning yields a particular scenario. This paper utilizes an XML-based representation of SES and PES. It presents tools for scenario development using SES: an SES modeling environment, namely SES Editor, and an interactive pruning tool, namely PES Editor.

I. Introduction

Simulation scenarios are the subjects of interest throughout the whole simulation study. Siegfried et al. [1, 2] identifie three types of scenarios: operational scenario, conceptual scenario and executable scenario. A simulation study typically starts with the description of the simulation scenario and ends with a successful simulation execution. Scenarios are used as executable specifications of a series of events and conditions being simulated. They are also used for identifying simulator capability gaps and identifying minimum simulator requirements. Taking scenario development as one of the core simulation engineering activities, we need shared understanding, common practices among stakeholders, operational subject matter experts and technical personnel and engineers, and we further need standards to boost interoperability and sharability. Durak et al. summarize the panel discussion on standardization for simulation scenario development in aviation in their paper [3]. Despite the known importance of scenarios, there is still a lack of common understanding and standardized practices in simulation scenario development which motivates a recent effort for standardization of scenario languages in aviation. System Entity Structure (SES) was proposed as a formal approach for a standard simulation scenario definition language [4]. Then, an XML-based computational of representation for the simulation scenario definition language was introduced in [5]. Based on these two studies [4, 5], this paper presents the tools for scenario development using SES. It introduces an SES modeling environment, namely SES Editor and an interactive pruning tool, namely PES Editor. Furthermore, it exemplifies their utilization with two case studies.

*Master Student, Internet Technologies and Information Systems, bikash.chandra.karmokar@stud.uni-hannover.de
†Research Scientist, Institute of Flight Systems, umut.durak@dlr.de, AIAA Senior Member
‡Assistant Professor, Department of Electrical, Computer, Software, and Systems Engineering, jafers@erau.edu, AIAA Member.
§Graduate Student, Department of Electrical, Computer, Software, and Systems Engineering, chhayab@my.erau.edu.
¶Professor, Department of Informatics, sven.hartmann@tu-clausthal.de.

II. Related Work

Domain Specific Languages (DSLs) are custom-tailored computer languages for a particular application domain that specifically target problems in that specific domain, and stresses upon the main idea, features, constraints and characteristics of that domain [6]. In early 2000s, industry adapted the idea for developing a standard scenario definition language for military simulations. The Military Scenario Definition Language (MSDL) was published as a standard by the Simulation Interoperability Standards Organization (SISO) in 2008 [7].

The research on extending the idea to other simulation domains and eventually with better formal basis is, however, still ongoing. A scenario specification based on a DSL is presented by Steffen Schütte in his paper [8], which can be used to formally describe simulations and scenarios. Siegfried et al. [2] proposed using Base Object Model (BOM) for modeling simulation scenarios in military domain.

There is a recent effort coordinated by the American Institute of Aeronautics and Astronautics (AIAA) Modeling and Simulation Technical Committee (MSTC) towards development of a standard scenario definition language for aviation. There are currently two modeling approaches: Jafer et al. [9, 10] propose metamodeling using Eclipse Modeling Framework, extending the idea presented in [2], Durak et al. [4, 5] utilize SES for metamodeling in order to specify the language. Both of the approaches yield a unified computational representation based on XML Schema [11].

III. Background

A. System Entity Structure

SES is a high-level ontology which was introduced for knowledge representation of decomposition, taxonomy and coupling of systems [12]. It has a set of elements and axioms. The elements of SES are Entity, Aspect, Specialization and Multiple-Aspect [13]. Real or artificial system components can be represented using Entity nodes [14]. An Entity is an object of interest, and can also have variables attached to it. An Aspect denotes the decomposition relationship of an Entity node. Specialization nodes represent the taxonomy of an entity. A Multi-Aspect is a special kind of aspect, which represents a multiplicity relationship that specifies the parent entity as a composition of multiple entities of the same type. Aspect, Specialization and Multi-Aspect are represented by one, two and three vertical lines respectively. Fig. 1 shows the elements of SES and Fig. 2 presents an example. There are six axioms of SES: (1) uniformity, (2) strict hierarchy, (3) alternating mode, (4) valid brothers, (5) attached variables, and (6) inheritance [15]. According to the uniformity axiom, any two nodes with the same labels have isomorphic subtrees. Strict hierarchy defines a restriction that prevents a label from appearing more than once down any path of the tree. Alternating mode recommends that, if a node is an Entity, then the successor is either Aspect or Specialization and vice versa. Valid brothers axiom disallow two brothers from having the same label. An attached variable specifies a constraint that variable types attached to the same item shall have distinct names. With inheritance, it is indicated that Specialization inherits all variables and Aspects.

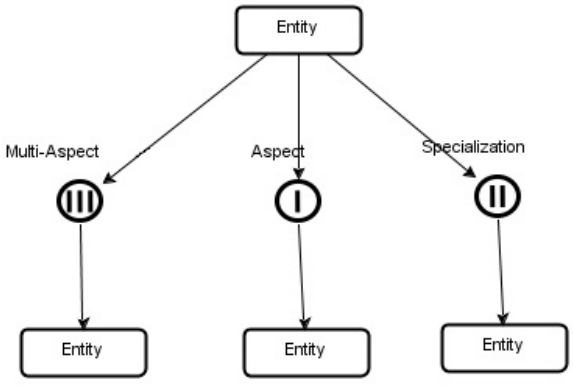

Fig. 1 System Entity Structure: Elements.

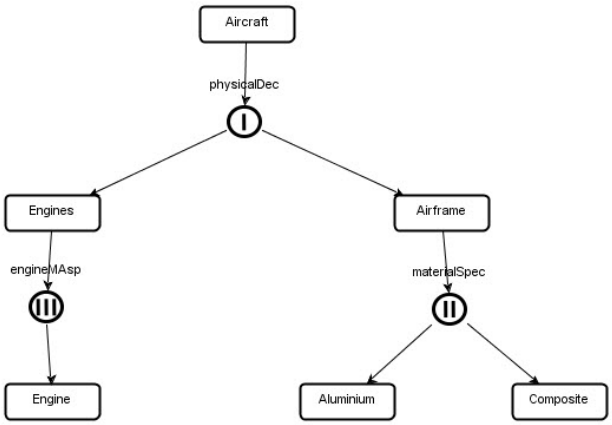

Fig. 2 System Entity Structure Example.

B. Pruning

Pruning is the operation in which a unique system structure is derived from an SES and the result is called Pruned Entity Structure (PES). SES represents a model of a given application domain in terms of decompositions, component taxonomies and coupling specifications. During modeling using SES, all the possible variations or configurations of a system are considered. As an SES describes a number of system configurations, the SES tree needs to be pruned to get a particular configuration. PES is therefore a selection-free tree. The pruning process normally reduces SES by removing choices of entity which has multiple aspects and specialization consists of multiple entities. An SES tree can be pruned by performing following tasks:

(i) Assigning values to the variables.
(ii) Choosing particular subject from several Aspect nodes for several decomposition of the system on same hierarchical level.
(iii) Selecting one entity from various options of Specialization node.
(iv) Specifying cardinality in Multi-Aspect node.

IV. Tools for Simulation Scenario Development

A. Scenario Development

Siegfried et al. [1] distinguish three types of scenarios:

1) Operational scenarios are described in early stages by the user or the sponsor. The intended situation and its dynamics are described in operational scenarios.
2) Conceptual scenario can be understood as a modeling and simulation expert's specification of the operational scenario. It is a structured scenario specification that identifies and clarifies all points required for consistency and completeness.
3) Executable scenario is defined as the specification of the conceptual scenario to be processed by the simulation applications for preparation, initialization and execution.

Scenario development is viewed as the transformation of operational scenarios to conceptual scenarios and eventually to executable scenarios. As the development process occurs, the scenario models are refined and transformed [16]. The scenario development process starts with operational scenarios, mostly in text form constructed in natural language by domain experts. They usually present key information from the user's perspective and are often not complete and consistent. Operational scenarios are then used to develop conceptual scenarios, essentially to be based on a metamodel, either an explicit or an implicit one. The result is then a complete and a consistent scenario specification, often compiled in a table. Finally conceptual scenarios are transformed into executable scenarios for target simulation environments using a set of rules specific to the target simulation environment.

Fig. 3 shows the transformation details of operational scenarios to executable scenarios during scenario development using SES. At the beginning, domain experts provide text of the scenario in operational space. From that text, possible entities and their classifications are filtered. The common classification that applies to all simulation scenarios in aviation are captured in Core Scenario SES. Based on the specific area of the application, such as air traffic simulation or flight simulation, we propose extensions to the Core Scenario SES. The options include using any existing extension as it is, enhancing an existing extension based on particular requirements or developing a completely new extension for a particular application. There we propose SES Editor as the tool for metamodeling, editing, enhancing or developing new SESs. SES Editor is a graphical tool for metamodeling using SES. It also checks the well-formedness of the metamodel against SES axioms.

In order to create a particular scenario from that metamodel, we propose another tool, namely PES Editor, which is a graphical tool for interactive pruning of an SES model. By pruning the SES for the simulation scenario definition language metamodel, a number of specific scenarios can be derived. Each derived scenario is then a PES; in computational space, it is an XML file. Finally the scenarios in XML can be translated to other executable scenario formats, using for example Extensible Stylesheet Language Transformations (XSLT).

B. SES Modeling Environment

SES Editor has been developed as an application agnostic SES modeling environment. Fig. 4 shows its Graphical User Interface (GUI) which is designed in such a way that a user can draw SES graphs on the screen almost as one would on paper. An SES is represented by a directed tree structure. Here, objects are represented by nodes which are

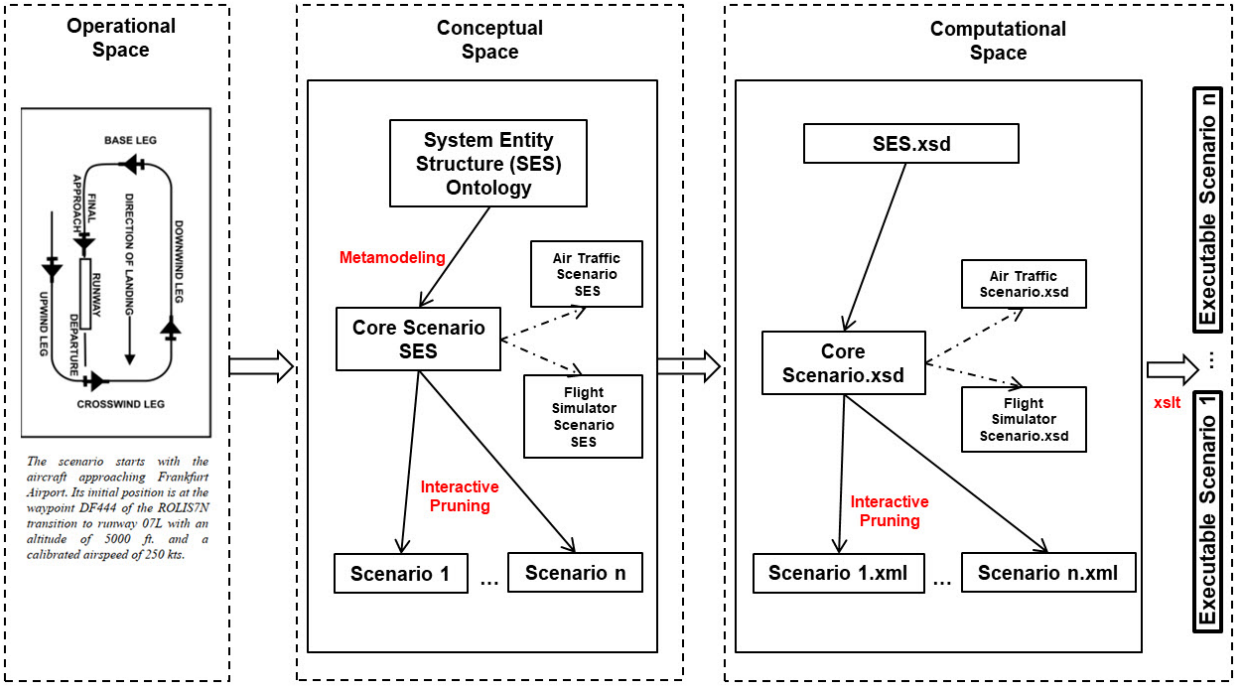

Fig. 3 Scenario Development Using SES

connected using edges. In Fig. 4, an SES model is presented in the editor where the root entity Aircraft is decomposed into Engines and Airframe using physicalDec aspect node. The Multi-Aspect node enginesMAsp decomposes Engines to Engine representing that Engines is made of multiple instances of Engine. Specialization node materialSpec is used to express that Airframe can be Aluminum or Composite.

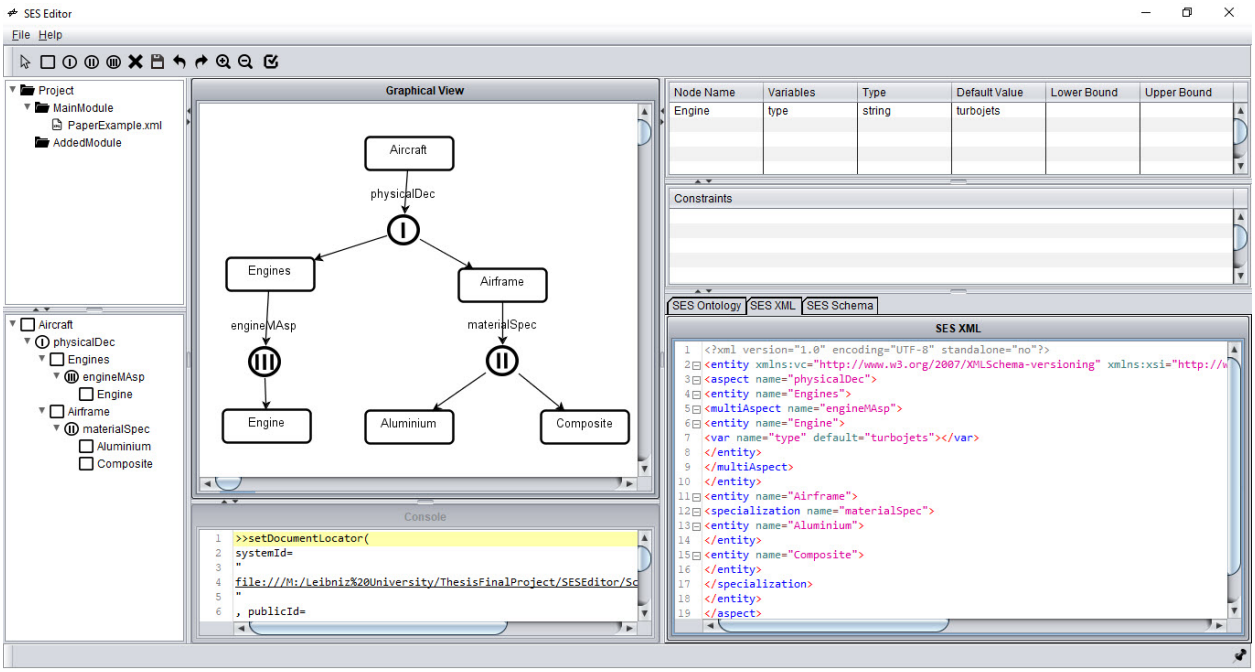

Fig. 4 A Screenshot from SES Editor.

Elements icons are added in the toolbox for easy access. The vertices or elements and edges can be drawn by clicking and dragging the mouse. Also nodes and edges can be easily moved to any position. The drawing panel is synchronized with the bottom-left tree. If there is an element addition in one place, either in the white drawing panel or the left tree, it will be added in both the sections automatically. Variables can be attached to the nodes. Eventually, the attached variables are listed in the variable table on the right top corner during any node selection from either of the trees. Furthermore, selection constraints can be added to the aspect node to restrict the choices of entities. These constraints are specified using XPath. Like variables, constraints are also listed on the constraint table on the right side of the editor during aspect node selection. The SES Editor allows the user to save part of the designed model as a module for future use. That saved module can be reused in any project later. SES Editor also has the ability to validate the created model against SES axioms using a predefined XML Schema. Validation result is displayed in the console window and for valid models, the SES XML and SES Schema are displayed in the bottom-right display window. The editor's export and import options increase the shareability of the designed models.

C. PES Pruning Tool

PES Editor has been developed as an interactive pruning tool. The user basically selects each and every decision point, such as a specialization node, and resolves the decision. It supports all the design patterns proposed in Deatcu et al. [17] for interactive pruning of SES. Its GUI looks very similar to the SES Editor. It also has variable table for displaying variables or editing values of variables. Constraint table and console window work exactly the same way. Here, the left side tree is also synchronized with the white drawing panel and nodes are movable. Unlike SES Editor, here we can not create new projects, but only open SES models created in SES Editor. New elements cannot be added or deleted; the existing SES cannot be edited. The main functionality of PES Editor is interactive pruning of an SES model.

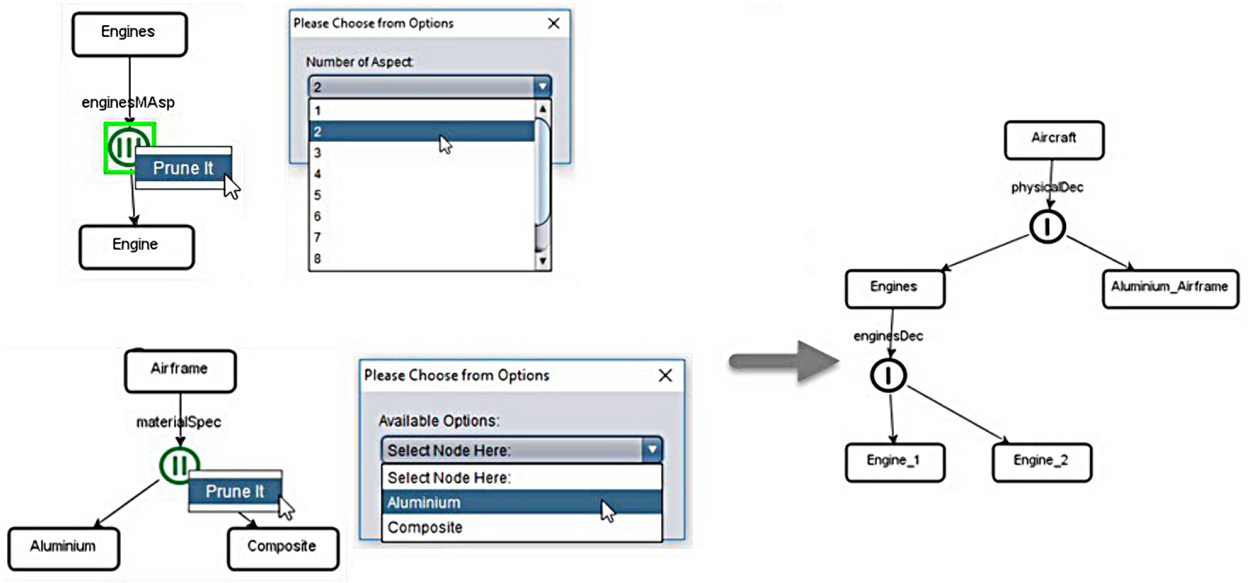

Fig. 5 An Excerpt from PES Editor.

Fig. 5 depicts an instance from interactive pruning using PES Editor. During pruning, MultiAspect node enginesMAsp is pruned and aspect node enginesDec is added with children Engine_1 and Engine_2 of the same type as Engine. From the available options, the specialization node is being pruned and Aluminum_Airframe is being selected. Thus the completely pruned structure is created, where there are no choices left and a specific model of an Aircraft is shown. When the pruning is complete, PES Editor also supports transformations using XSLT. The user can transform the PES to the schema of choice without leaving the PES Editor.

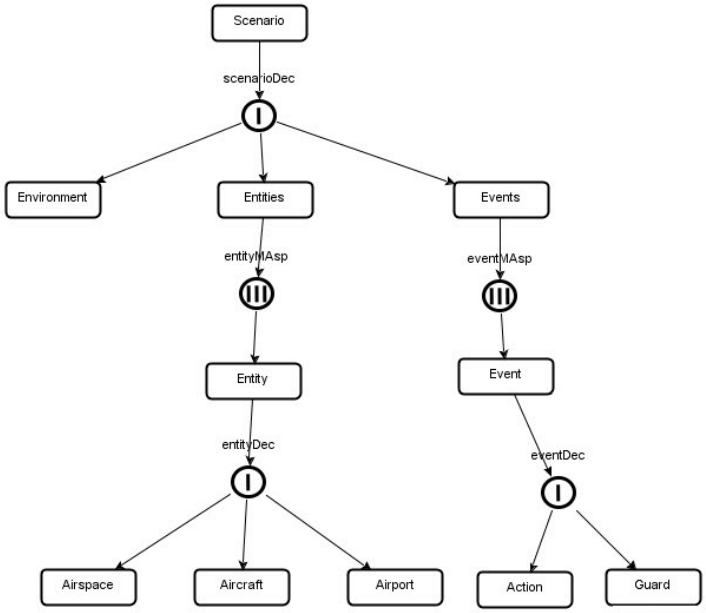

Fig. 6 Simulation Scenario Definition Language Core.

V. Case Studies

The scenario development approach discussed above in Section IV.A already mentions the idea of having a core language specification and its extensions. This requirement was first mentioned at the panel discussion which is summarized in [3]. SES Editor is first employed to model the Simulation Scenario Definition Language Core. A simplified version of the model is given in Fig. 6. It encompasses only the entities that are common to all different types of simulation scenarios.

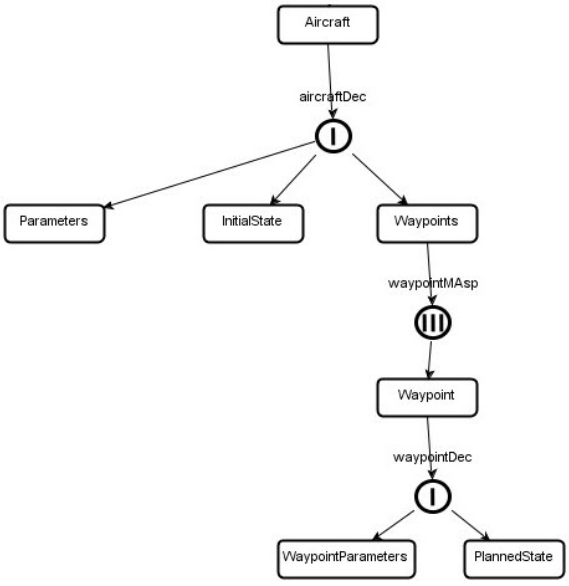

Fig. 7 Excerpt from the Computer Generated Traffic Simulation Extension.

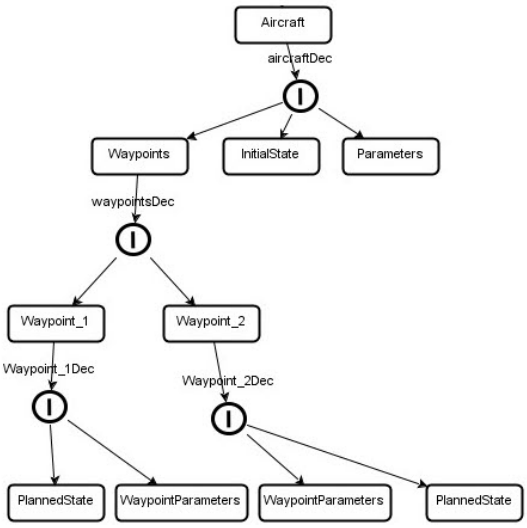

Fig. 8 Excerpt from the Pruned Scenario.

In order to exemplify an extension, we took German Aerospace Center (DLR) Computer Generated Traffic Simulation

as a baseline. It has a proprietary XML-based scenario format. We rendered the elements of its scenarios and modeled an extension SES using the module feature of SES Editor. An excerpt from this SES is given in Fig. 7. It was then possible to use PES Editor to prune the scenario metamodel (the core and extension) and reach a particular scenario. An example is given in Fig. 8. Finally, we utilized the XSLT-based transformation feature of PES Editor to produce executable scenarios.

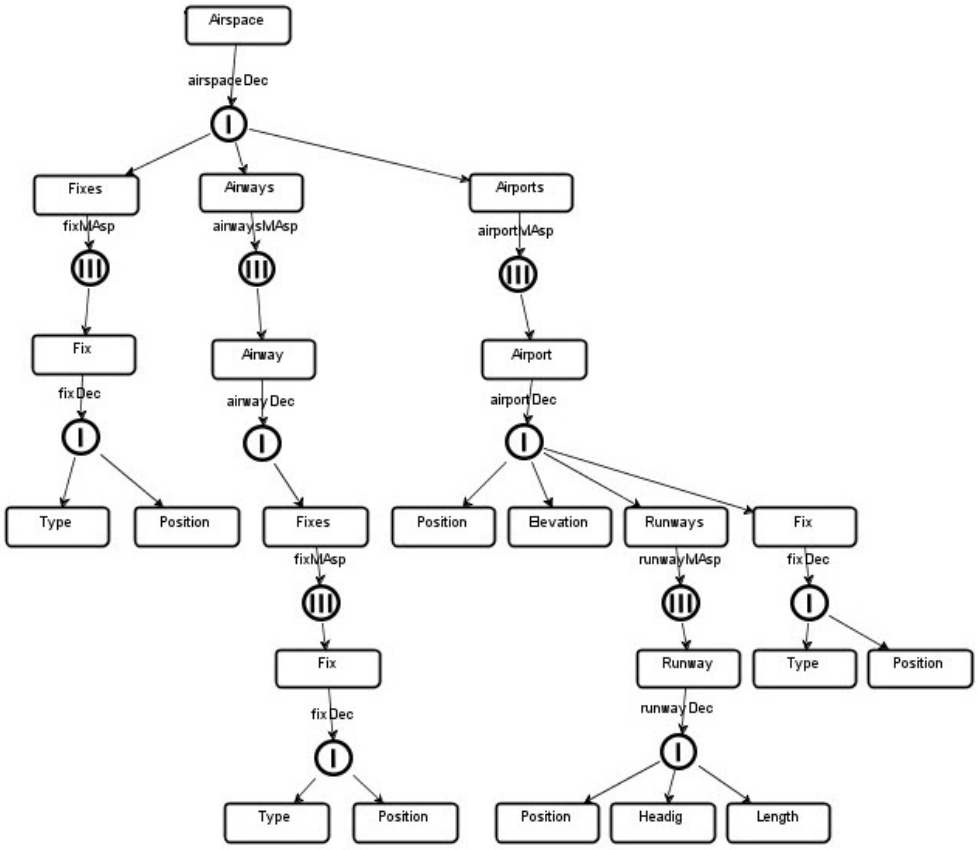

Fig. 9 Excerpt from the Air Traffic Control Simulation Extension.

As the next step, we tried to run the same process for another area. Thereby, the aim was to demonstrate the support of tools for more than one extension. This time, we took the Embry-Riddle Aeronautical University (ERAU) Air Traffic Control Simulation as the baseline. Excerpt from air traffic control simulation extension that is developed using SES Editor is given in Fig. 9. Like in the previous case, PES Editor is then employed for pruning and XSLT-based transformations. As expected, the approach and the tool support was successfully utilized to yield the generation of an executable scenario.

VI. Discussion

Rather than designing tools particularly for simulation scenario development, the SES modeling environment and the PES editor are developed as application agnostic tools. The reason is to provide a transparent way to demonstrate the conformance to the formalism, the SES ontology, that lies underneath the approach. We are well aware that it brings all the complexity and difficulty of language engineering. Its user is expected to have basic understanding of modeling, metamodeling, fundamentals of SES and XML technology. But that is only required until the demonstration of the concept in research setting and drafting the standard. There are currently two efforts going on for formal modeling of simulation scenarios in aviation: using Eclipse Modeling Framework [9, 10], and using SES [4, 5]. Both of the approaches yield a unified computational representation based on XML Schema [11] which the standardization effort aims to reach. This provides firm evidence about the formal basis of the proposed language. The tools proposed in

this paper enabled us to demonstrate the approach with use cases. These use cases will eventually lead to a draft standard. After a successful standardization process, the industry can proceed with using general purpose XML tools or developing specific editors.

VII. Conclusion

The paper introduced the tools that have been developed for scenario development with System Entity Structure. The tools are utilized in two different use cases in order to demonstrate an end-to-end scenario development process. The simulation scenario definition that is specified using System Entity Structure formalism is used to model conceptual scenarios. Then pruning is exercised as a way of scenario modeling. XML-based serialization enabled making use of stack of XML tools, standards and technologies for various purposes including constraint checking, well-formedness and text-to-text transformations. The tools presented in this paper implement the methodology and technology presented in [4, 5]. It complements and even completes the effort on proposing a formal approach for developing a scenario definition language.

It must be well understood, that this effort does not propose the language itself, but only the formal approach to develop it. A language specification, however, has been developed for conducting demonstrations and early use cases. Thereby, the benefits of using formal scenario definition language could be demonstrated. Furthermore, the use cases experimented the recommendation from the panel discussion [3] about having a core language and its extensions.

Based on the experience of the use cases, the next step is now to use the tools, techniques and methods to draft the standard simulation scenario definition language for aviation. This includes developing a commonly accepted, cross application Simulation Scenario Definition Language Core, and encouraging application specific communities to build their extensions.

References

[1] Siegfried, R., Laux, A., Rother, M., Steinkamp, D., Herrmann, G., Lüthi, J., and Hahn, M., "Scenarios in military (distributed) simulation environments," *Spring Simulation Interoperability Workshop (S-SIW)*, 2012.

[2] Siegfried, R., Oguztüzün, H., Durak, U., Hatip, A., Herrmann, G., Gustavson, P., and Hahn, M., "Specification and documentation of conceptual scenarios using Base Object Models (BOMs)," *Spring Simulation Interoperability Workshop*, Simulation Interoperability Standards Organization (SISO) San Diego, CA, 2013.

[3] Durak, U., Jafer, S., Beard, S. D., Reardon, S., Murphy, J. R., Crider, D. A., Gerretsen, A., Lenz, H., Macchiarella, N. D., Rigby, K. T., et al., "Towards a Standardization for Simulation Scenario Development in Aviation-Panel Discussion," *2018 AIAA Modeling and Simulation Technologies Conference*, Kissimmee, FL, 2018. doi:10.2514/6.2018-1395.

[4] Durak, U., Pruter, I., Gerlach, T., Jafer, S., Pawletta, T., and Hartmann, S., "Using System Entity Structures to model the elements of a scenario in a research flight simulator," *AIAA Modeling and Simulation Technologies Conference*, Grapevine, TX, 2017. doi:10.2514/6.2017-1076.

[5] Durak, U., Jafer, S., Wittman, R., Mittal, S., Hartmann, S., and Zeigler, B. P., "Computational Representation for a Simulation Scenario Definition Language," *AIAA Modeling and Simulation Technologies Conference*, Kissimmee, FL, 2018. doi:10.2514/6.2018-1398.

[6] Brambilla, M., Cabot, J., and Wimmer, M., "Model-driven software engineering in practice," *Synthesis Lectures on Software Engineering*, Vol. 1, No. 1, 2012, pp. 1–182.

[7] Group, M. S. D. L. P. D., "Standard for: Military Scenario Definition Language (MSDL)," Tech. Rep. SISO-STD-007-2008, Simulation Interoperability Standards Organization (SISO), Orlando, FL, 2008.

[8] Schütte, S., "A Domain-Specific Language For Simulation Composition," *Proceedings 25th European Conference on Modelling and Simulation (ECMS)*, 2011, pp. 146–152.

[9] Jafer, S., Chhaya, B., Durak, U., and Gerlach, T., "Formal Scenario Definition Language for Aviation: Aircraft Landing Case Study," *AIAA Modeling and Simulation Technologies Conference*, Washington, DC, 2016. doi:10.2514/6.2016-3521.

[10] Jafer, S., Chhaya, B., and Durak, U., "Graphical Specification of Flight Scenarios with Aviation Scenario Defintion Language (ASDL)," *AIAA Modeling and Simulation Technologies Conference*, Grapevine, TX, 2017.

[11] Jafer, S., Chhaya, B., Updegrove, J., and Durak, U., "Schema-based Ontological Representations of a Domain-Specific Scenario Modeling Language," *Journal of Simulation Engineering*, Vol. 1, 2018.

[12] Kim, T.-G., Lee, C., Christensen, E. R., and Zeigler, B. P., "System entity structuring and model base management," *IEEE Transactions on Systems Man and Cybernetics*, Vol. 20, No. 5, 1990, pp. 1013–1024.

[13] Zeigler, B. P., and Hammonds, P. E., *Modeling and simulation-based data engineering: introducing pragmatics into ontologies for net-centric information exchange*, Elsevier, 2007.

[14] Zeigler, B. P., Praehofer, H., and Kim, T. G., *Theory of modeling and simulation: integrating discrete event and continuous complex dynamic systems*, Academic Press, 2000.

[15] Zeigler, B. P., *Multifacetted modelling and discrete event simulation*, Academic Press, 1984.

[16] Durak, U., Topçu, O., Siegfried, R., and Oguztuzun, H., "Scenario development: A model-driven engineering perspective," *2014 International Conference on Simulation and Modeling Methodologies, Technologies and Applications (SIMULTECH)*, IEEE, 2014, pp. 117–124.

[17] Deatcu, C., Folkerts, H., Pawletta, T., and Durak, U., "Design patterns for variability modeling using SES ontology," *Proceedings of the Model-driven Approaches for Simulation Engineering Symposium*, Society for Computer Simulation International, 2018.