AIAA SciTech Forum
9 - 13 January 2017, Grapevine, Texas
AIAA Modeling and Simulation Technologies Conference

Using System Entity Structures to Model the Elements of a Scenario in a Research Flight Simulator

Umut Durak[1], Insa Pruter[2], Torsten Gerlach [3],
German Aerospace Center (DLR), Institute of Flight Systems, Braunschweig, 38108, Germany

Shafagh Jafer[4]
Embry-Riddle Aeronautical University, Daytona Beach, FL, 32114-3900

Thorsten Pawletta[5]
University of Wismar, Wismar, 23952, Germany

and

Sven Hartmann[6]
Clausthal University of Technnology, Clausthal-Zellerfeld, 38678, Germany

While any simulation study starts with a scenario, scenario development is usually conducted in an unstructured and ad hoc manner. In order to streamline scenario development, a formal approach is envisioned in the research flight simulator facility of German Aerospace Center (DLR), namely Air Vehicle Simulator (AVES). System Entity Structure (SES) which is a high level ontology that was introduced to specify a set of system structures and parameter settings is proposed as the foundations. The paper outlines a model-based methodology for scenario development. SES is exploited for metamodeling in order to capture all possible elements of a scenario that can be simulated in AVES. Then a scenario modeling methodology is built upon this metamodel.

I. Introduction

GERMAN Aerospace Center (DLR) Institute of Flight Systems (FT) has been using flight simulators for many decades. Currently a modern research simulator facility, namely Air Vehicle Simulator (AVES) is being operated at DLR Braunschweig. AVES has interchangeable cockpits for rotorcraft (EC135 ACT/FHS) and an airplane (A320 ATRA) which can be operated on motion and fixed-base platforms according to the particular research requirements.[1] It is possible to execute various and diverse simulation scenarios in AVES, ranging from handling qualities analyses of air vehicles to usability studies of particular pilot vehicle interfaces or situation awareness studies including a full Line-Oriented Flight Training (LOFT) scenario. The simulation study in AVES starts with the description of the simulation scenario by the researcher and ends with a successful simulation

[1]Research Scientist, Flight Dynamics and Simulation, Lilienthalplatz 7, 38108 Braunschweig, Germany, umut.durak@dlr.de, AIAA Member.
[2]Research Scientist, Flight Dynamics and Simulation, Lilienthalplatz 7, 38108 Braunschweig, Germany, insa.pruter@dlr.de.
[3]Group Leader, Flight Dynamics and Simulation, Lilienthalplatz 7, 38108 Braunschweig, Germany, torsten.gerlach@dlr.de.
[4]Assistant Professor, Department of Electrical, Computer, Software, and Systems Engineering, Embry-Riddle Aeronautical University - Daytona Beach Campus, Lehman 343, 600 S Clyde Morris Blvd, Daytona Beach, FL 32114-3900, jafers@erau.edu.
[5]Professor, Computational Engineering & Automation Research Group, PF 1210, D-23952 Wismar, Germany, thorsten.pawletta@hs-wismar.de.
[6]Professor, Department of Informatics, Julius-Albert-Strasse 4, 38678 Clausthal-Zellerfeld, Germany, sven.hartmann@tu-clausthal.de.

execution. Scenarios are not only required to develop executable specifications, but also useful for identifying capability gaps of the simulator for that particular objective. They are used in designing the new simulator features and even for testing them. The simulation scenarios are subjects of interest throughout the whole simulation study. Such an extensive use of scenarios dictates the management of scenarios. It includes scenario development and a set of activities around it such as scenario loading and parsing, scenario distribution and event injection, and scenario-related data collection and briefing. [2]

Taking scenario development as the core activity, a formal scenario definition language with its development environment is envisioned as a unified front end for streamlining the process. Such a language will enable us to utilize model-driven practices for scenario development. Metamodeling is proposed as a mechanism to design a graphical modeling language and model transformations are proposed as automated means to generate an executable scenario specification. Further we aim at modeling-time mechanisms for scenario analysis for consistency, completeness checking and identifying simulator capability gaps.

Generally there are two main challenges in scenario development for flight simulation and particularly for AVES. First, it is important to come up with a scenario definition language that will be natural to the scenario developers who are usually either engineers with a background in aeronautical systems or pilots. The second challenge is the complexity and variability of the system of interest, namely the aircraft that is being simulated as the protagonist entity. Complexity comes from the large subsystem hierarchy. The simulation of an aircraft is usually composed of co-simulation of various subsystems such as electrical, hydraulic or navigation systems which are composed of large number of sensors, actuators and computer networks. Scenario definition inevitably inherits this complexity. Variability comes into the play due to distinct experimentation requirements of the researchers that use AVES. These requirements correspond to variability in systems modeling both in fidelity and resolution. The modeled system structure and its elements may vary, thereby variying the scenario definition.

Durak et al. proposed utilization of the Base Object Model (BOM) structure as a baseline for a scenario development language.[3] This approach is then employed to develop the Aviation Scenario Definition Language.[4] BOM is an effort from the distributed simulation community[5] and it provides adequate constructs to capture the elements of a simulation scenario regardless of the domain of interest. On the other hand a domain specific scenario definition language which is composed of domain specific constructs would be easier to grasp and use. Its user is then able to use the terms and vocabulary that she is familiar with to develop a simulation scenario. Further while a BOM-based approach is suitable for a multitude of simulation entities that are interacting with each other, no particular facilities are provided towards handling hierarchical systems which also include variability in their structure. This paper investigates another approach that has its roots from system theoretic approaches to modeling and simulation, namely System Entity Structure (SES). SES is a high level ontology which was introduced to specify a set of system structures and parameter settings. It has long been used for modeling variable-structure systems and was recently applied to problems of model-based simulation system engineering such as model-based testing[6,7] and variability management.[8]

The background section introduces the simulation scenario development and the SES. Then the paper proposes a model-based approach for scenario development using SES.

II. Background

A. Simulation Scenario Development

Conducting experiments in a research flight simulator like AVES asks for engineered scenarios to investigate all required aspects and behaviors of the system and the flight crew. Scenario development is an extensive process beginning with the stakeholders' descriptions of the scenario and finishing with the generation of the corresponding executable specifications. IEEE 1278[9], the Standard for Distributed Interactive Simulation, defines the scenario as the description of initial conditions and timeline of significant events. The elements of scenarios in AVES include the systems and subsystems of interest, their initial trim states, the environmental conditions, course of events through the simulation run and termination conditions.

Following Siegfried and his colleagues we can distinguish three types of scenarios that are produced in successive stages of the scenario development process: operational scenarios, conceptual scenarios and executable scenarios.[10-12] Operational scenarios are described in the early stages of an AVES simulation study by the researchers. Usually they are documented in a textual form. The key elements in this operational scenario are the entities, their initial states and the events. The key entity of a simulation study in AVES is always the own aircraft. It is the protagonist of the simulation scenario such that the simulation of it and its subsystems is utmost importance. Below is an example of an operational scenario:

The scenario starts with the aircraft approaching Frankfurt Airport. Its initial position is at the waypoint DF444 of the ROLIS7N transition to runway 07L with an altitude of 5000 ft. and a calibrated airspeed of 250 kts. The weather conditions comprise a low visibility with medium CAT I conditions (500 ft. cloud base, 2000 ft. RVR (runway visual range)), gusty cross winds from the right and moderate turbulences. The pilot is instructed to fly a manual CAT I approach. After being established on the glideslope the aircraft is hit by a flock of birds resulting in a lefthand engine compressor stall. Due to a runway incursion of departing traffic ATC instructs the aircraft to swing to RWY 07C. The whole manoeuver is flown manually without auto thrust to increase pilot workload.

The operational scenarios provide a coarse description of the intended situation and its dynamics, but they need to be refined and augmented with additional information pertaining to simulation. This refinement is usually done by the simulator experts and results in conceptual scenarios. Conceptual scenarios specify the piece of the world to be represented in the simulation environment in detail. They should incorporate all crucial information for executing the operational scenario in AVES.

The executable scenario is the specification of the conceptual scenario in a particular format in order to be processed by the simulation applications in AVES for initialization, and execution. They support scenario management activities such as scenario distribution and role casting.[2] For this purpose, the conceptual scenarios need to be transformed into executable scenarios. The transformation from conceptual scenarios to executable scenarios is undertaken primarily by simulator experts. Ideally, the resulting executable scenarios are specified in a way that they are directly processable by the member applications of AVES.

B. System Entity Structure

The system theory-based approach to modeling and simulation has resulted in many enhancements in this field, one of which is System Entity Structure.[13] SES is a high-level ontology which was introduced for knowledge representation of decomposition, taxonomy and coupling of systems. [14]

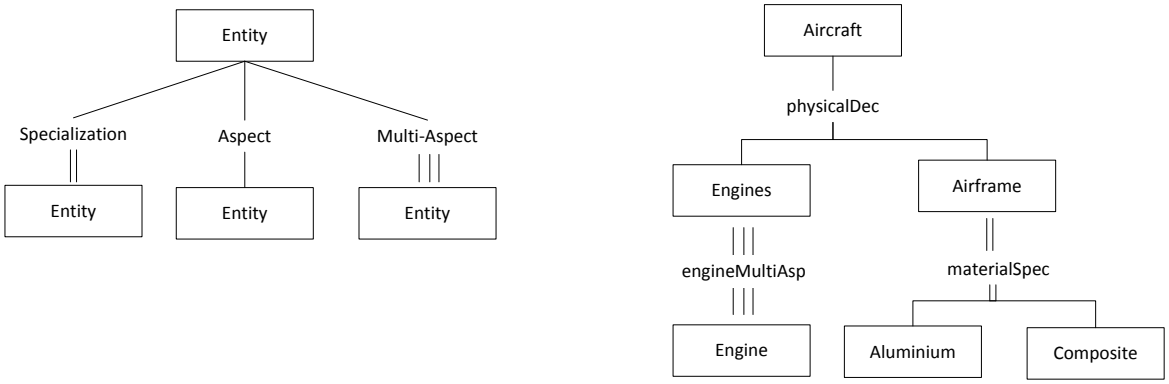

Figure 1 SES Nodes and relationships and an Example

SES is based on a clear, limited set of elements and axioms. It possesses four types of elements: Entity, Aspect, Specialization and Multi-Aspect. [14] It can also be introduced as a directed and labelled tree composed of Entity and Aspect, Specialization and Multi-Aspect nodes. Entity is an object of interest. Variables can be attached to Entities. Aspect denotes the decomposition relationship of an entity while specialization represents its taxonomy. Aspects are represented by vertical lines and Specializations with double line. The Multiple-Aspect is a special kind of Aspect that represents a multiplicity relationship that specifies that the parent entity is a composition of multiple entities of the same type. Three vertical lines are used to represent Multiple Aspects.

The axioms of SES are *uniformity*, *strict hierarchy*, *alternating mode*, *valid brothers*, *attached variables* and *inheritance*. [15] *Uniformity* states that any two nodes with the same labels have isomorphic subtrees. *Strict hierarchy* prevents a label from appearing more than once down any path of the tree. *Alternating mode* endorses that, if a node is an Entity, then the successor is either Aspect or Specialization, and vice versa. *Valid brothers* prohibits having two brothers with the same label. *Attached variables* state that variable types attached to the same item shall have distinct names. With *inheritance,* it is stated that Specialization inherits all variables and Aspects.

Pruning is defined as assigning the values to the variables and resolving the choices in Aspect, Multi-Aspect and Specialization relations. While there may be several Aspect nodes for several decompositions of the system on the same hierarchical level, a particular subset can be chosen in pruning based on the purpose. Specializations enable to capture various variants of an entity, one which needs to be selected during pruning. The cardinality in Multi-Aspect

relations is also specified in pruning, thereafter resulting in the Pruned Entity Structure (PES), which is a selection-free tree.

III. Modeling Scenario Elements with System Entity Structure

Data engineering refers to activities regarding data in design, development, management and utilization of information systems. [16] Scenario development can be categorized as a data engineering activity in which the data elements of a scenario are modeled. Zeigler proposes SES as an ontological framework for simulation data modeling.[17] Following his ideas, in this section we will be proposing a model-based scenario development approach that employs a SES-based metamodeling.

A. Model-based Scenario Development

Model-driven methodology offers the development of models and the generation of executable software entities with successive model transformations through technical spaces.[18] The technical space represents the context for specification, implementation or execution. The elements of the model-driven methodology are modelling languages, metamodels and transformations.[19] Modelling languages enable the definition of a concrete representation for a model and metamodels are used to define modelling languages. Transformations are described as the mappings between models which are specified at metamodel level.

The Unified Modeling Language (UML) has been used in the last two decades for modelling software structures, behaviour and architecture.[20] Model Driven Architecture (MDA)[21], described an integrated approach from business or domain models, through logical system models to implementation models utilizing UML and Meta Object Facility (MOF).[22] MOF is the model driven integration framework of the Object Management Group (OMG) for defining, manipulating, and integrating metadata and data. It provides a meta-metamodel at the top level, which is called M3 layer of the four-layer metamodeling hierarchy. This meta-metamodel is then used to define more specific metamodels at M2 layer, such as the UML metamodel. The UML models that conform to the UML metamodel are at layer M1. Finally, the M0 layer includes the instances created from a UML model.

Similarly, Eclipse Modelling Framework (EMF) that utilizes UML and here EMF CORE (ECORE), has been developed as an alternative framework within the Eclipse ecosystem for Model Driven Development (MDD).[23] ECORE is a standard for data models that offers a metamodel for describing models as well as a persistency support with XMI serialization.

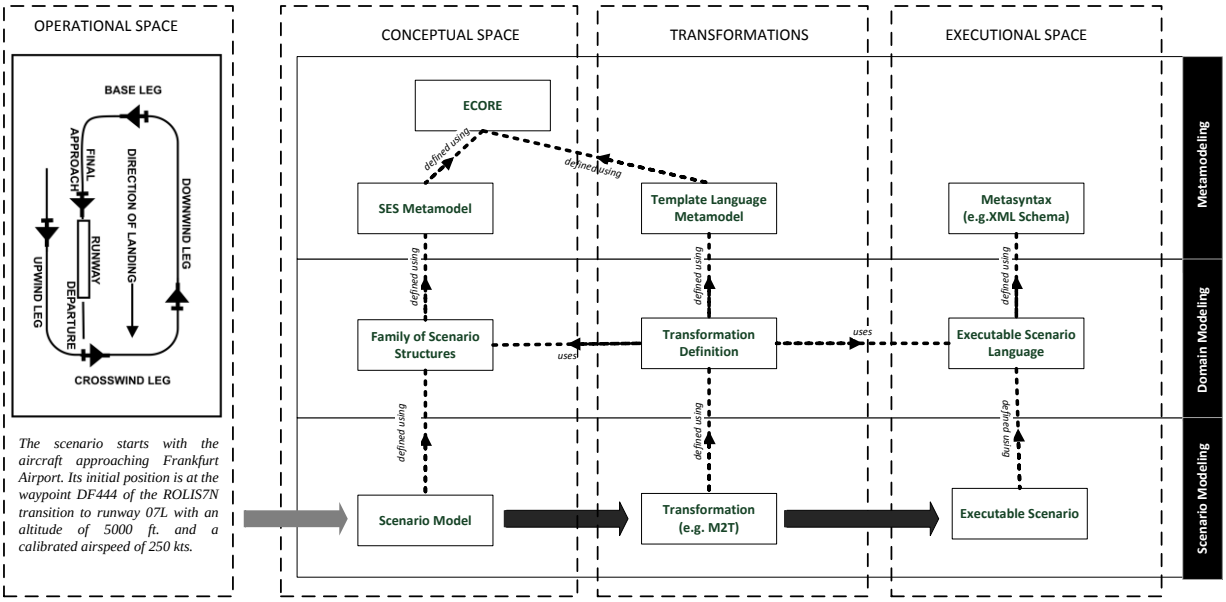

Figure 2 Model-based Scenario Development Methodology

Figure 2 presents an overview of the proposed model-based scenario development methodology. There are three technical spaces of interest. The first one is operational space where the operational scenario is described. The second one is conceptual space in which the conceptual scenario is modelled based on the scenario description provided as an operational scenario. The grey arrow designates a manual modeling process. The third space is the

executional space in which the executable scenario in a particular format is produced. The black arrows present an automated transformation process.

Adopting OMG's layered framework, we propose metamodeling, domain modeling and scenario modeling layers. In order to position the methodology in the EMF ecosystem, the SES metamodel is proposed to be developed using ECORE in the metamodeling level. This metamodel can then be used to specify domain specific scenario definition languages in the domain modeling level. For a flight simulation scenario definition language case, the domain is aviation. The nodes of SES are extended to elements of a simulation scenario in aviation. Scenario modeling then uses the flight simulation scenario definition language, the SES that is specified in domain modeling. It can be described as a manual pruning operation. The result is a PES that specifies a particular flight simulation scenario.

In executional space, a particular target executable scenario definition language can be defined using a metasyntax such as XML Schema. Between these technical spaces, there exist the transformations. Transformations can be defined at domain modeling level using a template language metamodel that is provided in metamodeling level. A template is a text with placeholders for the data to be extracted from the model. Placeholders are the constructs that are specified in the source metamodel. The queries are employed to extract the values from the models that conform to source metamodel. In our case, elements of a scenario with corresponding variable values can be extracted from the scenario model for each placeholder in order to generate an executable scenario.

A standard template language metamodel is offered by OMG for specifying the MOF Model to Text Transformation Language (MTL).[24] Acceleo is an EMF project that provides a pragmatic implementation of MTL. [25]

B. SES Metamodel

The SES metamodel shall capture all SES constructs and their relationships. Figure 3 presents an overview of a representative SES metamodel that has been developed using ECORE. The constructs of the metamodel are Entity, Specialization, Aspect, MultiAspect and Attribute classes. An Attribute has a name and value field which are specified as EString type for the sake of simplicity. In order to exemplify the relationships between the constructs, we can have a look at the unidirectional references between Entity and MultiAspect. An Entity type node can have a zero to n multi-aspect reference to MultipleAspect type node, and further a MultipleAspect type node has a reference to an Entity type nodes.

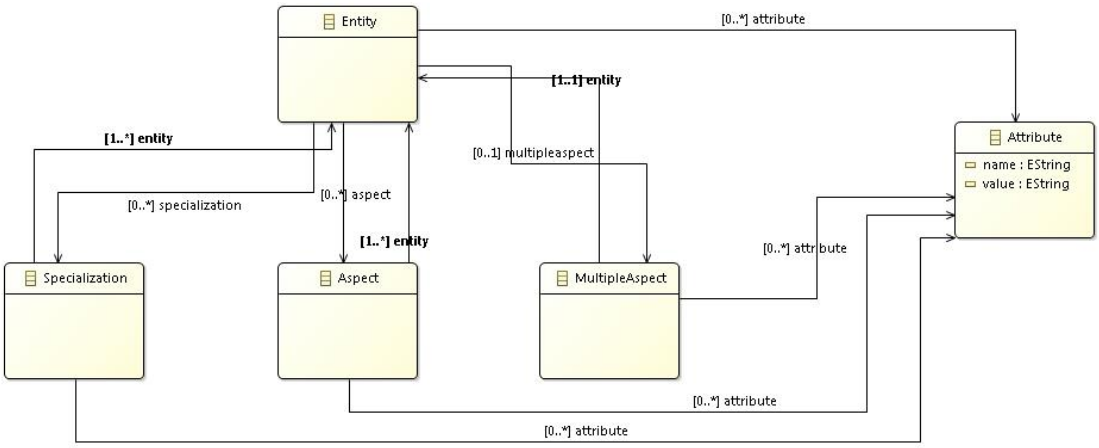

Figure 3 System Entity Structure Metamodel

C. AVES Scenario Metamodel

The AVES Scenario Metamodel captures all possible meta elements of a scenario in AVES by a System Entity Structure. A representative excerpt of the AVES Scenario Metamodel is presented in Figure 4. The top level Scenario entity is decomposed using the scenarioDec aspect node into Environment, Entities and Events. entityMultiAsp multi-aspect node decomposes Entities to multiple nodes Entity. entitySpec specialization node is then used to capture the different types of Entity. Two examples from a larger set that are depicted in the figure are Aircraft and the Airport. aircraftDec is then used to identify the aspects of an Aircraft that are of interest as elements of a scenario. These are namely Flight and Systems which stands for multiple System. Flight is then decomposed into its states: Position, Attitude, Angular Velocity and Translational Velocity. systemSpec defines the types of System that are concerned as the elements of a scenario. The representative set that is depicted in the figure includes

Flight Control, Communication, Electrical and Navigation. The FlightControlSystem is decomposed into Controls and Autopilots. Stick, Yoke and Pedals are kinds of a Control. As an example Stick has a Lateral Position and Longitudunal Position aspects as elements of a scenario. eventDec decomposes an event to a Guard and an Action. Two Guard types are State and Time. eventSpec is on the other side is used to capture various types of Event. Examples are Bird Strike and Engine Stall.

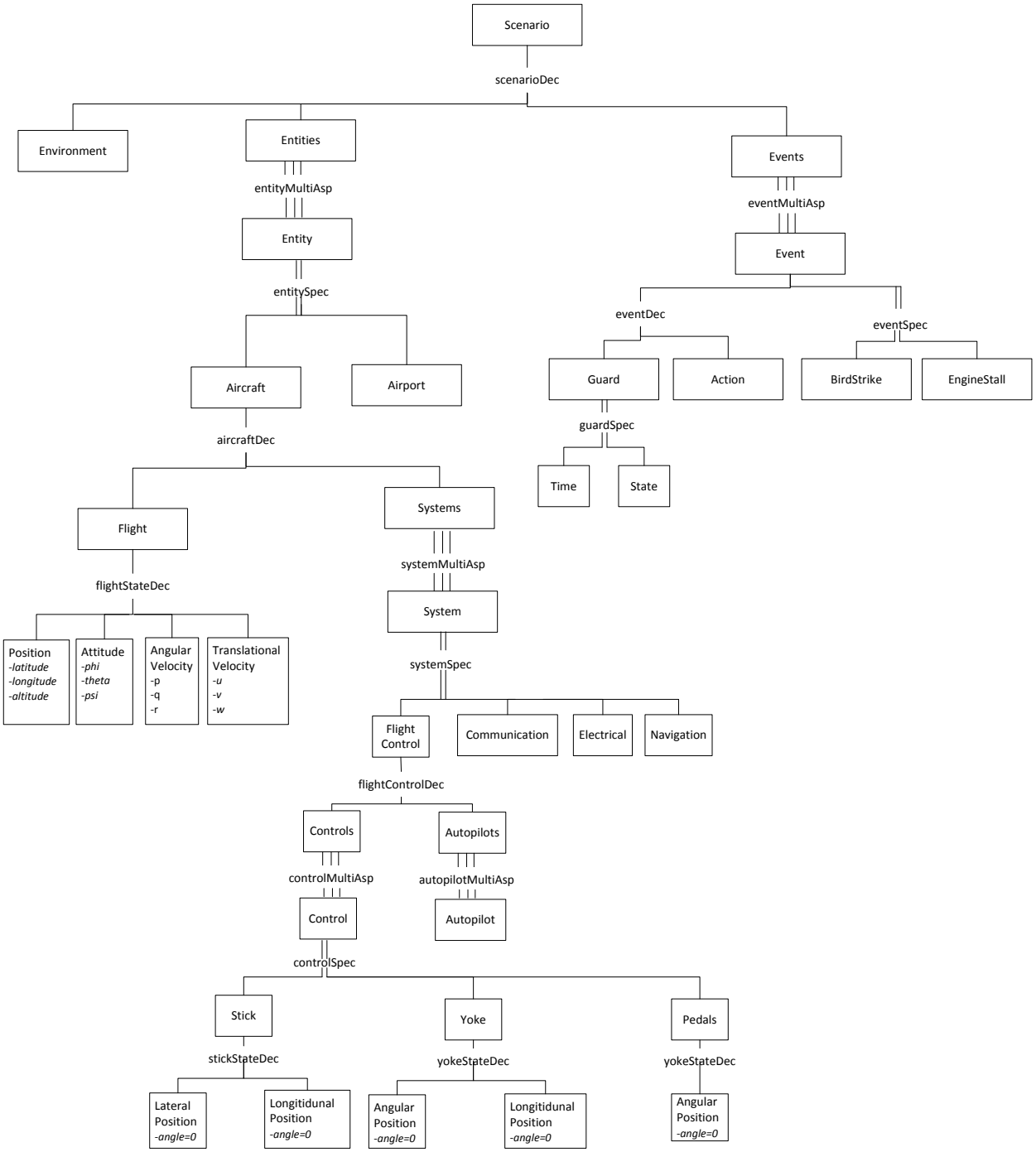

Figure 4 AVES Scenario Metamodel Extract

The SES attributes are used basically at the leaf entities that correspond to a scenario data to capture the data values. As an example the Position entity has the attributes Latitude, Longitude and Altitude. Some of the attributes

have default values which are also specified at the metamodeling level. Examples from Figure 4 include Lateral or Longitudinal Positions of the Stick which are by default set to zero.

While the introduced excerpt is not complete, it includes a sufficient number of elements to be representative as a metamodel that captures various possible scenario elements. The complete metamodel captures all the possible scenario elements that are available in AVES.

D. Scenario Modeling

When such a tree of all possible elements of the simulation scenario exists, modeling a conceptual scenario is nothing but Pruning of this tree to represent a very particular scenario. The values to the attributes needs to be assigned and the choices for Aspect, Multi-aspect and Specialization ought to be done. The selection-free tree at the end is the model of the unique scenario.

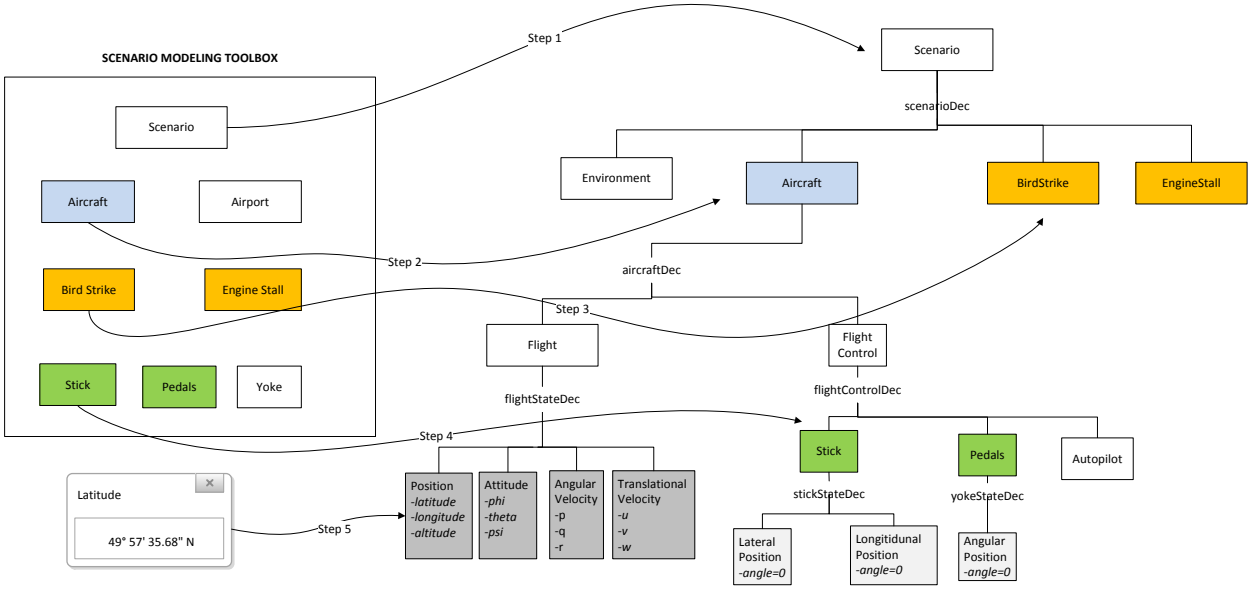

Figure 5 Scenario Modeling via Manual Pruning

Pruning can be accomplished via automated means using a scripting front-end that declares the attribute values and selections in decision nodes such as cardinalities at multi-aspect nodes or types at specialization nodes. Figure 5 on the other hand presents a manual pruning approach that may be received as the preliminary design of an interactive scenario modeling environment. The AVES Scenario Metamodel is used to construct a modeling toolbox that is composed of decision nodes of the SES tree. As the user adds the Scenario to his model, Environment appears as a decision-free element of the scenarioDec. Then the user needs to decide which and how many Entity's will be added. Referring to the example in Figure 5, in Step 2 the user adds the Aircraft to the decomposition and two events, namely Bird Strike and Engine Stall. The incorporation of the Aircraft entity to the model comes with a decision-free substructure that specifies Flight and flight state elements. For the Flight Control, the user pics Stick and Pedals as the controls. As the last step of the modeling, the user needs to specify the values of the attributes that are missing for the entities that are designated by the color code grey. As an example, referring the operational scenario that is introduced previously, the aircraft is in DF444 of the ROLIS7N transition to RWY 07L with an altitude of 5000 ft. It implies that the Latitude is 49° 57' 35.68" North, the Longitude is 7° 48' 56.19" East and Altitude is 1524 meters. While the dark grey entities represent the entities that the user needs to specify the attributes during modeling, the light grey entities have attributes with already assigned default values.

IV. Conclusion

This paper presents a model-based scenario development approach that exploits the System Entity Structure (SES) for metamodeling and modeling. Despite its key role in a simulation study, there is no structured and well-formed methodology for scenario development. In the model-based scenario development approach that is proposed for the German Aerospace Center (DLR) Air Vehicle Simulator (AVES), following the layered Meta Object Facility (MOF) of the Object Management Group (OMG), SES is proposed for metamodeling for designing a domain

specific scenario definition language. The scenario definition language captures all possible elements of a scenario in AVES in directed labelled tree. Then manual pruning is proposed as modeling practice in order to create a decision-free tree that represents a unique scenario. Model transformations are then proposed as a means for generating an executable scenario. Thereby, the user is envisioned to develop simulation scenarios for AVES using a well-formed graphical language whose constructs are terms from the flight simulation domain.

The future work includes developing tools and infrastructure for the proposed approach. It includes a graphical modeling environment and model-to-text transformation templates for generating the executable scenarios. Further research directions comprise adding modeling constraints in metamodeling for guaranteeing completeness and consistency of the developed scenario, and automated scenario generation for competency-based training by matching metamodel elements and desired crew actions.

References

[1]Duda, H., Gerlach, T., Advani, S. and Potter, M., "Design of the DLR AVES Research Flight Simulator," *AIAA Modeling and Simulation Technologies (MST) Conference*, Boston, MA, 2013.

[2]Topcu, O., Durak, U., Oguztuzun, H., and Yilmaz, L., *Distributed Simulation – A Model Driven Engineering Approach*. Springer, Cham, 2016.

[3]Durak, U., Topcu, 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)*, Vienna, Austria, 2014.

[4]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.

[5]SISO, "Base Object Model (BOM) Template Specification," SISO-STD-003-2006, 2006.

[6]Schmidt A, Durak U, Pawletta T., "Model-Based Testing Methodology Using System Entity Structures for MATLAB/Simulink Models," *SIMULATION*, Vol. 92, No. 8, 2016, pp. 729-46.

[7]Durak, U., Schmidt, A. and Pawletta, T., "Model-Based Testing for Objective Fidelity Evaluation of Engineering and Research Flight Simulators," *AIAA Modeling and Simulation Technologies Conference*, Dallas, TX, 2015.

[8]Pawletta, T., Schmidt, A., Zeigler, B.P. and Durak, U., "Extended Variability Modeling Using System Entity Structure Ontology within MATLAB/Simulink," *SpringSim-ANSS*, Pasadena, CA, 2016.

[9]IEEE, "Protocols for Distributed Interactive Simulation Applications-Entity Information and Interaction," IEEE 1278, 1993.

[10]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)*, Orlando, FL, 2012.

[11]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 (S-SIW)*, Orlando, FL, 2013.

[12]MSG-086, "Guideline on Scenario Development for (Distributed) Simulation Environments," NATO TR-MSG-086-Part-II, 2015.

[13]Oren, T.I. and Zeigler, B.P., "System Theoretic Foundations of Modeling and Simulation: A Historic Perspective and the Legacy of A Wayne Wymore," *SIMULATION*, Vol. 88, No. 9, 2012, pp. 1033-1046.

[14]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. 1013-1025.

[15]Zeigler, B., P., *Multifaceted Modelling and Discrete Event Simulation*, Academic Press Professional Inc., London, 1984.

[16]IEEE., "What is the Technical Committee on Data Engineering?," *IEEE technical Committee on Data Engineering*[web site], URL: http://tab.computer.org/tcde/ [cited 20 October 2016].

[17]Zeigler, B., P. and Hammonds, P., E., *Modeling and Simulation-Based Data Engineering: Introducing Pragmatics into Ontologies for Net-Centric Information Exchange*, Academic Press, 2007.

[18]Gasevic, D., Djuric, D., and Devedic, V., *Model Driven Engineering and Ontology Development*. Springer, Berling, 2009.

[19]Brambilla, M., Cabot, J. and Wimmer, M., *Model-Driven Software Engineering in Practice*. Morgan & Claypool Publishers, 2012.

[20]OMG, "Unified Modeling Language™ (UML®) Version 2.5," formal/2015-03-01, 2015.

[21]OMG, "Model Driven Architecture (MDA) MDA Guide Rev. 2.0," ormsc/2014-06-01, 2014.

[22]OMG, "Documents associated with Meta Object Facility™ (MOF™) Version 2.5," formal/2015-06-05, 2015.

[23]Steinberg, D., Budinsky, F., Merks, E. and Paternostro, M., *EMF: Eclipse Modeling Framework*. Pearson Education, 2008.

[24]OMG, "MOF Model to Text Transformation Language™ (MOFM2T™), 1.0," formal/08-01-16, 2008.

[25]Eclipse Foundation., "Acceleo," *Eclipse Wiki* [web site], URL: http://wiki.eclipse.org/Acceleo [cited 20 October 2016].