Simulation Scenarios for Testing Airborne Cyber-Physical Systems

Siddhartha Gupta[1*], Umut Durak[2], Sven Hartmann[1]

[1]Institute of Informatics, TU Clausthal, Julius-Albert-Str. 4,
38678 Clausthal-Zellerfeld, Germany; [*]*siddhartha.gupta@tu-clausthal.de*

[2]Instutute of Flight Systems, German Aerospace Center (DLR), Lilienthalplatz 7, 38108 Braunschweig, Germany

Abstract. Aircraft are extremely complex Cyber-Pyhsical Systems (CPS) with many component interactions to control the flight and provide rich functionality. The rise in the utilization of modelling and simulation based approaches for CPS development is phenomenal. The Model-Based Design (MBD) brought the system models in the center of the development process. Simulation-based verification established the practice for executing system models as the native mechanisms of testing. Model-in-the-loop, software-in-the-loop and hardware-in-the-loop, commonly named as x-in-the-loop testing became the main steps of simulation-based verification. Feedback loops make the controller and plant interaction crucial in CPS. In order to create an execution of a credible test case, the plant and controller models are simulated in a scenario is in a closed-loop fashion. This is called scenario testing. Simulation scenario is a specification for the initial and terminal conditions, significant events and the environment as well as the major entities, their capabilities, behavior and interactions over time. Scenario development is an extensive process beginning with the stakeholders' descriptions of the scenario and finishing with the generation of the corresponding executable specifications. This paper discusses the development and use of scenarios for testing airborne CPS.

Introduction

Cyber-Physical Systems (CPS) are integration of computation and physical processes [1]. An aircraft is an extremely complex CPS, whose physical world includes mechanical aircraft parts, pilots, the physical airspace, the terrain and all other man-made systems associated with aircraft, and whose cyber world encompasses avionic systems, sensors and actuators as well as the networking elements such as data buses. CPS technologies are vital for the future of aeronautics.

Simulations play a major role in the engineering process as they allow engineers to test designs and prototypes without spending excessive resources on construction and manufacturing. A computer simulation is relatively easier to deploy. The cost of failure is minimal, encouraging experimentation and creative thinking.

Scenarios are important artifacts in the simulation engineering process [2]. A simulation study begins with a description of a scenario and ends with a successful simulation execution [3]. They play a key role in designing cost-effective methods for testing existing systems as well as designing new concepts on them. Elements of a scenario generally include the initial states, the environment , entities, and their capabilities, associated events, its timelines as well as the termination conditions. Scenario development is viewed as the transformation of operational scenarios (defined using natural language) to conceptual scenarios (conforming to a metamodel), and eventually to executable scenarios (specified using a scenario specification language) [4]. However, there was no common understanding of the elements and the structure of these elements in a scenario. This leads to deficiencies in communication and presenting test scenarios, sharing and reusing them, verifying and validating them, and prevents the development of common infrastructures for scenario development and execution.

Karmokar et al. [4] proposed System Entity Structures (SES) for metamodeling and its derivative structure, a Pruned Entity Structure (PES) which represents a particular scenario as effective components of an open simulation infrastructure. SES is a high-level ontology which was introduced for knowledge representation of decomposition, taxonomy and coupling of systems [5]. It represents elements of a system and their relationships in hierarchical and axiomatic manner. This helps in modelling all possible elements in form of a scenario metamodel. To extract a particular scenario from this metamodel, an operation called pruning is performed, which results in a decision free tree called a PES.

Durak et al. [6] promoted XML Schema as the computational representation of scenario elements in SES where each scenario in PES is an XML document. This

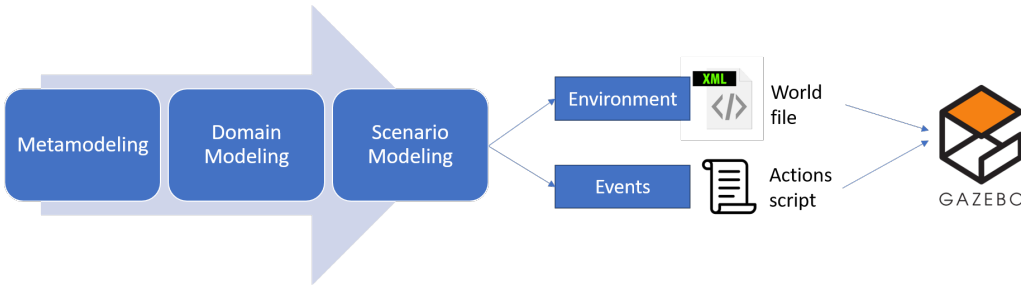

Figure 1: Simplified Workflow for Scenario-based Testing

provides a formal bases not only for the syntax but also for the semantics of scenario description thereby aids sharing and reuse of scenarios.

An important requirement of a scenario-based testing is executable characteristics of the scenarios specification. The user should be able to design a specific scenario from a scenario metamodel and execute it with a selected a simulator. This will allow the user to test the behaviour using scenarios. One such commonly used open source simulator is Gazebo [7], open source robotics simulator . It is generally used together with Robot Operating System (ROS) [8], which is a collection of tools and libraries which helps simplifying the task of building a robot system. Gazebo is a simulation tool which helps to simulate robots in different environments and supports the same communication infrastructure as ROS. ROS programs use plugins to interact with Gazebo robot models within the simulator. Gazebo and ROS have been used quite intensively in the last decade for developing and testing drones. Examples include [9, 10, 11, 12].

In this paper, we introduce very briefly a scenario-based testing approach where we use metamodeling to describe a language to model all the elements of a scenarios using SES, then pruning for scenario modeling. The corresponding infrastructure provides all the required components starting from scenario description to transformation to an executable scenario and the execution of this scenario in Gazebo. An sample use case is intended based on [12] where the components of the Service Oriented Mission System for Autonomous Drones (SOMSAD) will be tested on a Gazebo environment using the ROS infrastructure.

1 Scenario-Based Testing

A simplified representation of the scenario-based testing workflow is presented in Figure 1 At the highest level, it can be broken down into three aspects each described below.

1. **Scenario Modeling** - This includes three steps. We use SES as the metamodel following the footsteps of [4]. Then we model all the possible elements of the scenarios in consideration using SES into a scenario metamodel. Then we prune specific elements to come up with a specific scenario PES. This scenario model is in the XML format. The two main parts of this XML file are the initial setup of entities referred to as the environment, and the events taking place on the entities.

2. **Environment Setup** - We use scripts to extract the Environment section from the scenario model. It is then mapped to World file of the Gazebo simulator. The World file is of Simulation Description Format (SDF) , which is an XML format that describes objects and environments for robot simulators, visualization, and control. This World file can be used to launch the Gazebo simulator with the initial environmental setup.

3. **Events** - We use action scripts to extract events out of the scenario model and execute events in the Gazebo environment. The scripts send the events to the SOMSAD system using the REST interface which gets executed in Gazebo.

The implementation of the workflow in Figure 1 requires an infrastructure, tools and the intermediate scripts or interfaces for the tools to be able to pass data to each

other. The major tools required for this workflow already exist and needs to be extended to able to provide the functionality required for the workflow.

The tools used in the workflow are SES Editor, PES Editor, Gazebo and SOMSAD.

- **SES Editor** - A GUI tool to build SES models using all the SES elements and drag and drop options. It supports an ability to export XML files that can be imported by the PES editor [4].
- **PES Editor** - A GUI tool to import SES models and prune the model for a selection free tree [4].
- **Gazebo** - An open-source robotics simulator [7].
- **SOMSAD** - A service oriented drone mission system [12].

2 Outlook

The scenario-based testing approach introduced in the previous section will be supported by an open scenario infrastructure. The infrastructure will be extended to support interfaces with a variety of simulators. The first use case will be testing collaborative lifting scenarios where drones and cranes working together in a construction site. The future work includes scenario-based testing with various other simulation environments.

References

[1] Lee EA. Cyber physical systems: Design challenges. In: *2008 11th IEEE International Symposium on Object and Component-Oriented Real-Time Distributed Computing (ISORC)*. IEEE. 2008; pp. 363–369.

[2] 1730™-2010 IS. IEEE Recommended Practice for Distributed Simulation Engineering and Execution Process (DSEEP). 2011;.

[3] Jafer S, Durak U. Tackling the complexity of simulation scenario development in aviation. In: *Proceedings of the Symposium on Modeling and Simulation of Complexity in Intelligent, Adaptive and Autonomous Systems*. 2017; pp. 1–10.

[4] Chandra Karmokar B, Durak U, Jafer S, Chhaya BN, Hartmann S. Tools for Scenario Development Using System Entity Structures. In: *AIAA Scitech 2019 Forum*. 2019; p. 1712.

[5] Kim TG, Lee C, Christensen ER, Zeigler BP. System entity structuring and model base management. *IEEE transactions on systems, man, and cybernetics*. 1990;20(5):1013–1024.

[6] Durak U, Jafer S, Wittman R, Mittal S, Hartmann S, Zeigler BP. Computational representation for a simulation scenario definition language. In: *2018 AIAA Modeling and Simulation Technologies Conference*. 2018; p. 1398.

[7] Koenig N, Howard A. Design and use paradigms for gazebo, an open-source multi-robot simulator. In: *2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)(IEEE Cat. No. 04CH37566)*, vol. 3. IEEE. 2004; pp. 2149–2154.

[8] Quigley M, Conley K, Gerkey B, Faust J, Foote T, Leibs J, Wheeler R, Ng AY. ROS: an open-source Robot Operating System. In: *ICRA workshop on open source software*, vol. 3. Kobe, Japan. 2009; p. 5.

[9] Meyer J, Sendobry A, Kohlbrecher S, Klingauf U, Von Stryk O. Comprehensive simulation of quadrotor uavs using ros and gazebo. In: *International conference on simulation, modeling, and programming for autonomous robots*. Springer. 2012; pp. 400–411.

[10] Bernardeschi C, Fagiolini A, Palmieri M, Scrima G, Sofia F. Ros/gazebo based simulation of cooperative uavs. In: *International Conference on Modelling and Simulation for Autonomous Systesm*. Springer. 2018; pp. 321–334.

[11] Sciortino C, Fagiolini A. ROS/Gazebo-Based Simulation of Quadcopter Aircrafts. In: *2018 IEEE 4th International Forum on Research and Technology for Society and Industry (RTSI)*. IEEE. 2018; pp. 1–6.

[12] Gupta S, Durak U. RESTful Software Architecture for ROS-based Onboard Mission System for Drones. In: *AIAA Scitech 2020 Forum*. 2020; p. 0239.