

Generating Test Systems in Simulink Models for Testing Product Lines with ASTERYSCO

Aitor Arrieta¹, Leire Etxeberria¹, and Justyna Zander²

¹ Mondragon Goi Eskola Politeknikoa
{aarrieta, letxeberrria}@mondragon.edu

² NVIDIA
justyna.zander@gmail.com

Abstract. Simulink models are commonly employed to simulate and test complex systems such as Cyber-Physical Systems (CPSs). These systems are becoming highly configurable, and techniques from the product line engineering context (e.g., feature models) are being acquired by industrial practitioners to model the variability. Having variability in these systems means that there might be several configurations to test. Selecting relevant configurations by considering feature models following combinatorial techniques has been widely investigated by the software engineering community. However, efficiently testing each configuration has attracted little attention, which is not that trivial. One important aspect when testing such systems is automation. This tool paper presents ASTERYSCO, which aims at automatically generating test system instances in Simulink for testing specific configurations of configurable CPSs.

Keywords: Feature Modeling, MATLAB/Simulink, Product Line Engineering, Cyber-Physical Systems

1 Introduction

Complex systems involving software with physical components (e.g., Cyber-Physical Systems (CPSs)) are becoming highly configurable. As a result, product line engineering techniques are being acquired to model and manage their variability [4, 1]. Testing these systems is extremely complex as there might be thousands to millions of configurations, what requires combinatorial techniques to derive relevant configurations to test. However, even if a subset of relevant configurations are derived, testing each of them might be time consuming due to several reasons. First, testing CPSs requires multi-level testing (i.e., testing at the Model, Software and Hardware-in-the-Loop test levels). Secondly, when testing these systems, apart from the software, the physical layer needs to be considered. This physical layer is typically modeled with complex mathematical models that require high computational capabilities to simulate. As a result of this, several approaches have considered test selection and prioritization techniques [2, 3].

However, to efficiently test these system automation is required. Test systems provide a great possibility to automatically execute test cases and evaluate them by means of test oracles. However, when variability is considered requirements can change from product to product. Consequently, variability in the test system needs to be also considered. This demo paper presents a method supported by a tool named ASTERYSCO (Automatic Simulation-based TEst system geneRator for cYber-physical Systems COnfigurations) [1]. ASTERYSCO employs FeatureIDE [6] to model the variability of both, the system and the test system. In a first step, ASTERYSCO parses the feature model it generates a 150% model of both the test system as well as the Cyber-Physical System Under Test (CPSUT). In a second stage, it parses a specific configuration to be tested and it instantiates the test cases and test oracles that are necessary to carry out the automatic testing of the specific configuration. The web-page of the tool together with a demonstrative video is available in: <https://sites.google.com/a/mondragon.edu/asterysco/>

The rest of the paper is structured as follows: Section 2 briefly explains ASTERYSCO. Section 3 explains the case study that will be used during the demonstration. Lastly, Section 4 concludes the paper.

2 Generating Test System Instances with ASTERYSCO

This section presents ASTERYSCO, our tool for the generation of test system instances. ASTERYSCO follows two steps: first, it generates a generic test system including the 150% model of the CPSUT and the 150% model of the test system by considering the feature model, the CPSUT and the context environment (i.e., the environment in which the CPS operates). In a second step, this generic test system is obtained and it is instantiated to test a specific configuration, returning a test system instance. This is done by reusing the test cases, the configuration file for the configuration under test and the requirements monitors for the test oracles. Figure 1 shows the overview of both steps for the generation of a test system instance in SPEM.

The returned test system is designed for testing Cyber-Physical Systems (CPSs) and the detailed architectural overview of the system is explained in [1]. This test system is based on a previous work for the automotive domain [7]. It consists of four main sources: (1) test stimuli, (2) test oracle, (3) test control and (4) context environment. Figure 2 depicts an example of the general overview of the test system returned by ASTERYSCO in the first step. The test stimuli encapsulates test cases and converts test cases into stimuli signal that excite the inputs of the system. The test oracle evaluates that the outputs of the system correspond to the inputs; to this end, each requirement is individually monitored. The test control schedules the order of a set of selected test cases. Last, the context environments simulates the environment that directly affects the system, i.e., the “world” in which it operates.

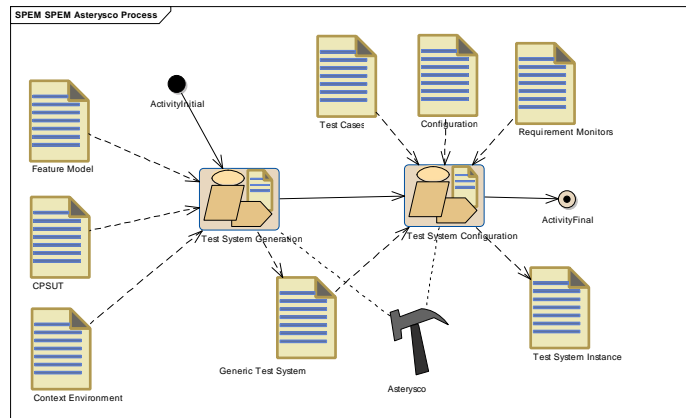


Fig. 1: Overview of ASTERYSCO’s activities for the generation of a test system instances in SPEM [1]. The arrows with the solid line indicate the control flow, whereas the arrows with the dashed lines indicate the object flow (i.e., inputs and outputs of each process).

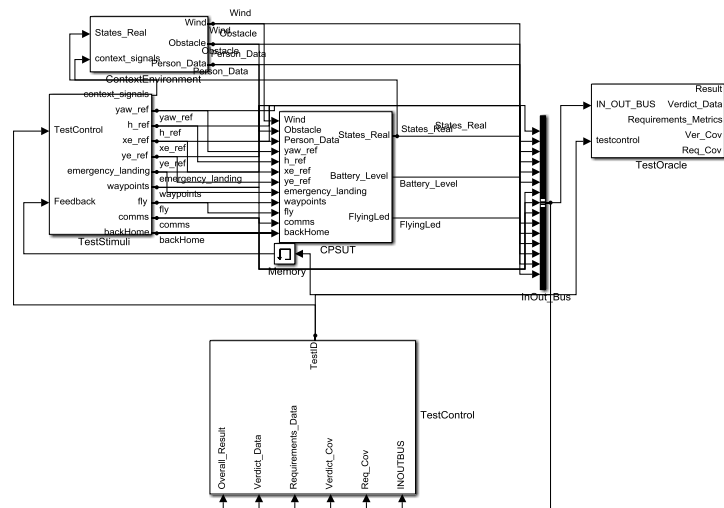


Fig. 2: The automatically generated generic test system model [1]

3 Case Study

The system has been validated with two case studies: (1) a configurable Unmanned Aerial Vehicle adapted from [5] and (2) the system that controls the windows of a car. The demonstrative example of the former is available in the

web-page of the tool. As for the latter, we foresee to use it as the tool demo during the conference. This case study involves a Simulink model with 9 inputs and 4 outputs and a feature model with 22 features and 3 constraints.

4 Conclusion

This tool demo paper presents the overview of ASTERYSCO, a tool that automatically generates test system instances for testing configurations of Simulink models of a configurable CPS. ASTERYSCO automates the process of generating test system instances in two main steps by parsing a feature model of the system and the system model configuration and generating the Simulink model of the test system instance. These steps are done within around 3 seconds, as demonstrated in our previous paper [1].

Acknowledgments

This work was carried out by the embedded systems group of Mondragon Goi Eskola Politeknikoa, supported by the Department of Education, Universities and Research of the Basque Government.

References

1. Arrieta, A., Sagardui, G., Etxeberria, L., Zander, J.: Automatic generation of test system instances for configurable cyber-physical systems. *Software Quality Journal* 25(3), 1041–1083 (2017)
2. Arrieta, A., Wang, S., Sagardui, G., Etxeberria, L.: Search-based test case selection of cyber-physical system product lines for simulation-based validation. In: *Proceedings of the 20th International Systems and Software Product Line Conference*. pp. 297–306 (2016)
3. Arrieta, A., Wang, S., Sagardui, G., Etxeberria, L.: Test case prioritization of configurable cyber-physical systems with weight-based search algorithms. In: *Proceedings of the Genetic and Evolutionary Computation Conference 2016*. pp. 1053–1060. *GECCO '16*, ACM, New York, NY, USA (2016)
4. Behjati, R., Yue, T., Briand, L., Selic, B.: Simpl: A product-line modeling methodology for families of integrated control systems. *Information and Software Technology* 55(3), 607 – 629 (2013), special Issue on Software Reuse and Product Lines
5. Mosterman, P.J., Sanabria, D.E., Bilgin, E., Zhang, K., Zander, J.: Automating humanitarian missions with a heterogeneous fleet of vehicles. *Annual Reviews in Control* 38(2), 259–270 (2014)
6. Thuem, T., Kastner, C., Benduhn, F., Meinicke, J., Saake, G., Leich, T.: Featureide: An extensible framework for feature-oriented software development. *Science of Computer Programming* 79, 70 – 85 (2014)
7. Zander-Nowicka, J.: *Model-based Testing of Real-Time Embedded Systems in the Automotive Domain*. Ph.D. thesis, Technical University Berlin (2008)