Proceedings of the 2013 Winter Simulation Conference
R. Pasupathy, S.-H. Kim, A. Tolk, R. Hill, and M. E. Kuhl, eds.

DEVSMO: an ontology of DEVS model representation for model reuse

Yunping Hu, Jun Xiao, Hao Zhao, Gang Rong

Institute of Cyber-Systems and Control
State Key Laboratory of Industrial Control Technology, Zhejiang University
Hangzhou, Zhejiang 310027, P.R.C

ABSTRACT

There are numerous modeling and simulation environments based on DEVS formalism. Due to the incompatible modeling grammars, it has been a challenge to reuse models in different DEVS implementations. Existing XML-based model representations lack general expressions of the behavior of DEVS models and only support one type of DEVS formalism. In this paper, a modeling ontology named DEVSMO is proposed. DEVSMO uses structured programming theory to express the programming logic and uses MathML to express the mathematical models in the model behavior. Structured programming theory and MathML provide a set of standard terminologies, so the generality of model representation is improved. Furthermore, DEVSMO supports both classic and parallel DEVS formalisms and has good reusability for other formalisms. Three cases are developed to test DEVSMO in the usability of expressing model structure and model behavior and in the reusability for further extension.

1 INTRODUCTION

DEVS (Discrete Event Systems Specification) is a universal formalism for the modeling and analysis of discrete event systems (Zeigler et al. 2000). Though different DEVS-based implementations all support the DEVS formalism, they use different programming languages to implement modeling grammars and simulators. Legacy models as well as models that are available in one implementation are difficult to translate from one language to another.

In order to support model reuse, model representations must be independent of computer programming languages. Furthermore, models represented by these representations can be transformed to or from those established in specific modeling languages. XML is a kind of markup language and can be used to implement the model representations. Various XML-based DEVS model representations including XLSC (Meseth et al. 2009), DEVS Meta Language (Janoušek, Polášek, and Slavíček 2006) and DEVS Modeling Language (Mittal et al. 2007) have been proposed. However, these representations lack general terminologies of the DEVS behavior which contains the programming logic and mathematical calculations. Furthermore, each of them only supports the model representation of one kind of DEVS formalisms, which limits their applications in model reuse. For example, if a model representation is implemented based on the classic DEVS formalism, it is difficult to support the reuse of models created in the parallel DEVS formalism because it has no markup to express the confluent transition function of a model.

In this paper, we propose an ontology-based model representation named DEVSMO (DEVS math ontology) to support the DEVS model reuse. DEVSMO uses the structured programming theory and MathML (Carlisle et al. 2001) to standardize the representation of the DEVS behavior. The structured programming is to standardize the programming logic, while MathML is to standardize mathematical calculations. It also supports the model representation of two kinds of basic DEVS formalisms, including the classic DEVS and the parallel DEVS.

2 Overview of DEVSMO

DEVSMO is composed of three sub ontologies: DEVS model ontology, model structure ontology and model behavior ontology. DEVS model ontology describes the classification of DEVS models according to DEVS formalisms. A DEVS model may be an atomic model or a coupled model. If an atomic model is created based on the classic DEVS formalism, it can be called a classic atomic model. If it is created based on the parallel DEVS formalism, it can be called a parallel atomic model. The classification of the coupled model is similar with the atomic model. Model structure ontology consists of atomic model structure and coupled model structure. The former includes input set, output set and state set, while the later includes input set, output set, component reference and the coupling of EIC , IC and EOC. A classic coupled model may have a select function which can be expressed as a ordered pair set.

Model behavior ontology is used to express the behavior of a DEVS model, including four parts of function, action, math model and control structure. A function may be a state transition function, output function or time advance function. A function refers to a control structure which may be a loop, a selection or a sequence according to the structured programming. A control structure is composed of actions and math models represented by MathML. The operation of complex variables is a kind of action, like the delete operation of a queue variable, while the logic or numerical calculation of simple variables is expressed as a math model.

3 Implementation and Application of DEVSMO

We implement DEVSMO in the OWL language which is a language for publishing and sharing ontologies in the web and is part of the growing stack of W3C recommendations. The MathML codes in DEVSMO can be embedded into the OWL files as the labels of OWL math model instances.

For the application of DEVSMO, we use components of Jena and MathML DOM to translate DEVSMO model instances to or from executable models represented by modeling languages or XML-based model representations. In cases, we use DEVSMO to represent the structure part and the behavior part of a processor model to test its usability and extend DEVSMO to support the fuzzy DEVS formalism to test its reusability. In the future, DEVSMO can be extended to support various DEVS formalisms, in addition to the basic formalisms.

ACKNOWLEDGMENTS

In preparing this paper, the author has benefited from suggestions of Associate Professor Xiaolin Hu in Georgia State University. Furthermore, the financial supports from the National High Technology R&D programs of China (2012BAE05B03 & 2012BAE05B05) are also gratefully acknowledged.

REFERENCES

Carlisle, D., P. Ion, R. Miner, N. Poppelier et al. 2001. "Mathematical markup language (mathml) version 2.0". *W3C recommendation* 21.

Janoušek, V., P. Polášek, and P. Slavíček. 2006. "Towards DEVS Meta Language". *ISC 2006 Proceedings*:69–73.

Meseth, N., P. Kirchhof, and T. Witte. 2009. "XML-based DEVS modeling and interpretation". In *Proceedings of the 2009 Spring Simulation Multiconference on ZZZ*, 152. Society for Computer Simulation International.

Mittal, S., J. L. Risco-Martín, and B. P. Zeigler. 2007. "DEVSML: automating DEVS execution over SOA towards transparent simulators". In *Proceedings of the 2007 spring simulation multiconference-Volume 2*, 287–295. Society for Computer Simulation International.

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