Data Management and Time Synchronization in PlugSim: A DEVS-Based Framework for Interoperation of Simulations

Jang Won Bae[1], Kyung-Min Seo[2], and Tag Gon Kim[3]

[1] Department of Industrial and Systems Engineering
KAIST, Daejeon, Republic of Korea,
jwbae@smslab.kaist.ac.kr

[2, 3] Department of Electrical Engineering
KAIST, Daejeon, Republic of Korea,
kmseo@smslab.kaist.ac.kr[2], tkim@ee.kaist.ac.kr[3]

Abstract. As modern systems are increasing in complexity, modeling and simulating them are also becoming difficult. In particular, modeling and simulation (M&S) with one modeling method reveals the limitations of many simulation purposes. In order to tackle this limitation, the interoperation concept has been developed. For the interoperation of simulations, data management and time synchronization are indispensable. In this paper, we explain how the data management and time synchronization are performed in PlugSim environment. PlugSim is the DEVS-based framework for interoperation of simulations and its theoretical basis depends on the DEVS-BUS. PlugSim helps developers create an environment of interoperation of simulations. For better understanding, we provide an example and show the result of simulation in the case study.

1 Introduction

As modern systems are increasing in complex, modeling and simulating them are also becoming difficult. The systems are classified into two groups according to complexity. The first is the view of the number of sub-models in the system; the second thing is the view of various sub-systems in the system [1]. In the first case, modeling and simulating the system is not a difficult problem. However, in the second case, this is hard to tackle, because many systems (e.g. continuous system (CS), discrete event system (DES)) have their own characteristics.

For this reason, researchers have developed new methods for presenting the systems in the second case, such as multi-formalism and interoperation of simulations [2] [3]. In the second case, Sub-systems should be modeled using the most appropriate formalism and tool. In the multi-formalism approach, however, a single formalism is identified into which each of the sub-systems may be symbolically transformed [1]. Interoperation of simulations can be used with the most appropriate formalism and tool for the sub-systems.

There has been a great deal of researches regarding the representation of complex systems. For example, High Level Architecture (HLA) is a standard in IEEE which describes a specification for interoperation between heterogeneous simulators. Run-Time Infrastructure (RTI) is software that is the result of implementing HLA [7] [8] [9]. HLA/RTI supports interoperability and reusability of simulators. However, interoperation of simulations is inconvenient using HLA/RTI. In order to use HLA/RTI, users should understand the APIs of HLA/RTI and create interfaces that can handle HLA/RTI services. Another example is the Virtual Laboratory Environment (VLE) [4]. VLE is the framework for the integration of heterogeneous formalisms using Discrete Event System Specification (DEVS) formalism and associated extensions adapting a multi-formalism method.

PlugSim is the DEVS-based framework for interoperation of simulations [6]. It supports the interoperation of various simulators. Therefore, the models of various sub-systems are modeled and implemented by appropriate methods in PlugSim environment. The theoretical background of PlugSim is on DEVS-BUS [5], the interface that can communicate over the RTI. However, PlugSim is a complete environment for interoperation of simulations, not using RTI.

In this paper, we introduce the internal structure and algorithms of PlugSim in detail using an example. Data management and time synchronization in PlugSim are our main consideration. Then, we refer to the advantage of using PlugSim environment with the case study.

The rest of this paper is organized as following: In section 2, we briefly introduce the DEVS as our background. Section 3 shares concepts and components in PlugSim and explains how data and time are managed in PlugSim using torpedo engagement as an example. Section 4 shows our case study in PlugSim environment. We then conclude this paper in Section 5.

2 Background – DEVS formalism

PlugSim is based on the DEVS formalism. This section briefly explains the DEVS formalism. The DEVS formalism specifies discrete event models in a hierarchical and modular form. There are two kinds of models in the formalism: atomic model M and coupled model DN, as follows [10] :

$M = < X, Y, S, \delta_{ext}, \delta_{int}, \lambda, ta >$,
Where
X : a set of input;
Y : a set of output events;
S : a set of sequential states;
$\delta_{ext} : Q \times X \rightarrow S$, an external transition function,
where $Q = \{(s,e) | s \in S, 0 \le e \le ta(s)\}$ is the total state set of M;
$\delta_{int} : S \rightarrow S$, an internal transition function;
$\lambda : S \rightarrow Y$, an output function;
$ta : S \rightarrow$ Real, time advance function.

$DN = < X, Y, M, EIC, EOC, IC >$,
Where
X : a set of input events;
Y : a set of output events;
M : a set of all component models;
$EIC \subseteq DN.X \times \cup M.X$: external input coupling;
$EOC \subseteq \cup M.Y \times DN.Y$: external output coupling;
$IC \subseteq \cup M.Y \times \cup M.X$: internal coupling;
SELECT: $2^M - \phi \rightarrow M$: tie-breaking selector.

3 PlugSim: DEVS-based Framework for Interoperation Simulation

In this paper, we focused on the data management and time synchronization performed in PlugSim. Before we handle these subjects, we briefly present the concept and components of PlugSim.

3.1 Concept and Component of PlugSim

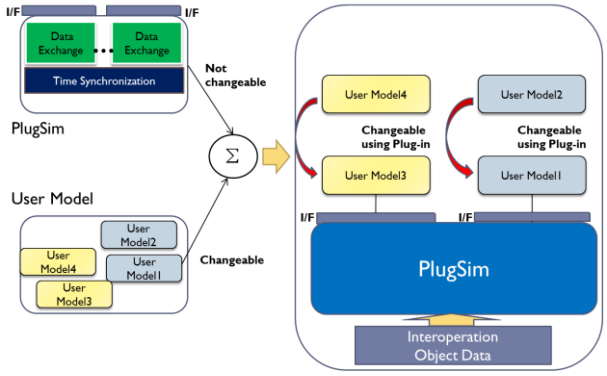

Fig. 1. Conceptual Design of PlugSim

Figure 1 shows the conceptual design of PlugSim. PlugSim provides data management and time synchronization services, which are the most important services in the interoperation simulation to users. Using these services and Interoperation Object Data (IOM), which is created with user-defined data information, users can join their models to the interoperation of simulations. Moreover, users can change their models during the simulation through PlugSim environment.

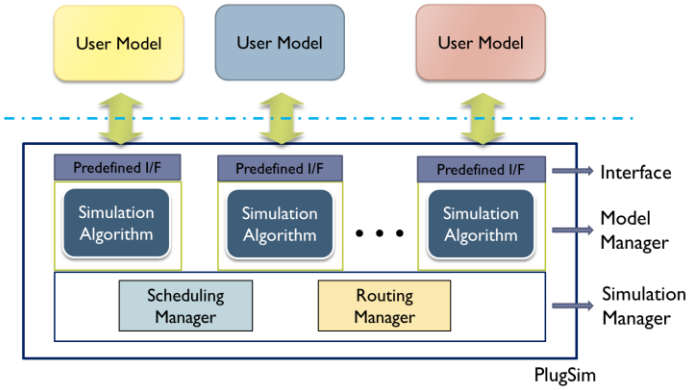

Fig. 2. Components in the PlugSim

In order to provide this environment, PlugSim consists of three main components: simulation manager, model manager and interface. Figure 2 shows the components of PlugSim. Roles of each component are as follows:

- Simulation Manager: The simulation manger consists of the scheduling manager and routing manager. The simulation manager synchronizes overall simulation time and gathers the information of data exchange then makes the IOM.
- Model Manager: The model manager has a discrete event based simulation algorithm. Using this algorithm, one model manager simulates a user model. It handles time and data information of the user model.
- Interface: The Interface allows the model manager to communicate with a user model. Through the interface, model manager can simulate a user model and obtain time information for the user model. According to the type of user model (i.e., DEVS model or non-DEVS model), the kind of interface employed can be different.

A detailed description of PlugSim, how to make the user model and other information can be found in [6].

3.2 Data Management and Time Synchronization in PlugSim

In this section, we explain data management and time synchronization of in PlugSim in detail. To aid in understanding, we use a specific example. In the following subsection, we briefly introduce the example.

3.2.1 Example: Torpedo Engagement [11]

We use the concept of torpedo engagement as our example. In this torpedo engagement, the engagement scenario and specification of model are defined.

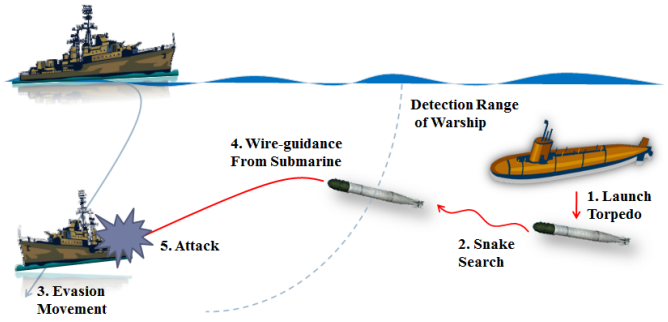

Fig. 3. Scenario of torpedo engagement

Figure 3 depicts a brief scenario of torpedo engagement. We explain this scenario in detail as followings.

1. A submarine launches a torpedo toward the warship
2. The torpedo searches for the position of the warship using its searching algorithm
3. When the warship detects the torpedo, the warship performs an evasion movement according to the direction of the torpedo
4. The submarines control the torpedo toward the warship using wire-guidance
5. As the control from the submarine, the torpedo goes to the warship and attacks it.

In the torpedo engagement simulation, we used only two models, a warship model and a torpedo model. The submarine model in the figure 3 is abstracted, because it does not affect the result of the simulation. The warship model and the torpedo model are separated by three parts: maneuver, sensor, control models. The maneuver model is based on a continuous system, while the others are based on a discrete event system, according to the characteristics of each model.

3.2.2 Time Synchronization

Time synchronization is an essential part of the simulation, particularly in the interoperation simulation, in which the various types of models can be joined. In order to synchronize simulation models by the simulation time, PlugSim should get the next simulation time of the models.

Before the simulation has begun, all the joined models notify their next simulation time to their model manager. Then, each model manager forwards the next simulation time to the simulation manager. The simulation manager decides the model, which has the minimum next simulation time, to simulate until its next simulation time. After model simulation concludes, the model notifies updated next simulation time to its model manager again. With iterating these steps, PlugSim can maintain the time synchronization of the interoperation simulation.

The time synchronization method, using in PlugSim, is based on the coordinator algorithm of DEVS [10]. In the coordinator algorithm, the coordinator obtains the subordinate simulators' next event times. The coordinator defines the minimum next event time among all next events times. Then, the coordinator sends a message to the simulator to simulate the atomic model.

We give an example of time synchronization in PlugSim using torpedo engagement. The warship and torpedo models are joined in the interoperation simulation using PlugSim. The warship and Torpedo models consist of three sub-models. However, according to the coordinator algorithm, it is not considerable in the time synchronization. Before beginning the simulation, two models send next simulation times to their model managers. When the simulation begins, the model managers serve the next simulation time to the simulation manager. The simulation manager defines the minimum next simulation time, then, lets the model manager simulate its model (for example, the warship model). The warship model will be simulated until the next simulation time. At the next simulation time, the warship model can define the next simulation

time by its model specification. The model manager can obtain the updated next simulation time and forward it to the simulation again. Figure 4 shows this example as a sequence diagram

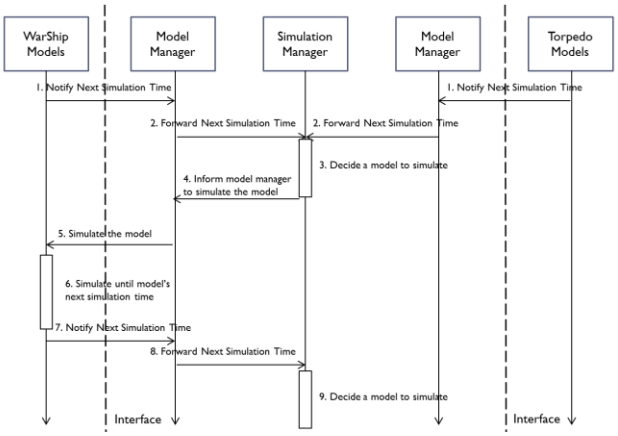

Fig. 4. Sequence Diagram of Time Synchronization in the example

The Discrete Event model (like the DEVS model) should retain its next event time. Therefore, the model manager can receive the next simulation time of its model. However, in a non-DEVS model case, for example continuous system model, there is no specification about the next schedule simulation time.

In the form of continuous system model, it is impossible to obtain the next simulation time of the model. There needs to be an additional method to the model. In [12], the interface which converts continuous time information to discrete event time information is well described. Continuous models, which are joined in PlugSim, can use that interface.

3.2.3 Data Management

In order to exchange data between user models, it is essential to define the transferred data first. We define this transferred data form as an Interoperation Object Model (IOM). Using user-defined IOM, the simulation manager makes the RoutingInfo which is used as a reference table in exchanging data. When the data is in transit from or to model, the model manager can transfer the data to/from the target model using the RoutingInfo. In the following subsection, we explain data management in PlugSim in detail.

3.2.3.1 Interoperation Object Model (IOM)

The IOM describes all the transferred data among user models. The IOM consists of local data set (LDS) of user models. LDS is the set of data that represents input/output in a model. In order to join to PlugSim, users should define the LDS of the joining model. However, the LDS format differs by the kind of user model (i.e. DEVS or non-DEVS model).

In the DEVS model case, the LDS format consists of the number of publish/subscribe (output/input) data and the name of the publish/subscribe data. However, in the non-DEVS model case, the LDS format is more complicated than the DEVS model case. Packet ID, field name and type of packet are additionally needed. This is because non-DEVS models communicate with PlugSim by packet. Figure 5 shows an example of LDS in the torpedo engagement. The torpedo control model is based on a discrete event system and torpedo maneuver mode is based on a continuous system.

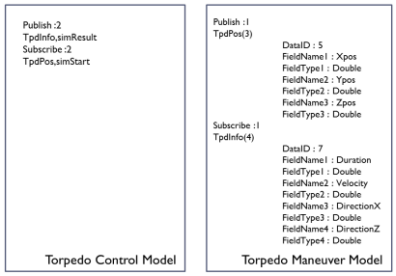

Fig. 5. Examples of LDS (L: CS model, R: DES model)

3.2.3.2 RoutingInfo

RoutingInfo is the table which represents from which the data is generated and to which it is forwarded. Using the user-defined LDS, the simulation manager defines the RoutingInfo table. For example, the RoutingInfo table indicates that 'TpdPos' data comes from the Torpedo Maneuver model and goes to the Torpedo Control model (in Fig. 6). When the RoutingInfo has been defined, the simulation manager sends the RoutingInfo to each model manager.

3.2.3.3 Data Exchange

Data exchange is performed among the model managers. We explained the detailed step of exchange data as follows

When the data has been generated in the user model, it is forwarded to its model manager. The model manager finds target model referring the name of the output data to the RoutingInfo. The model manager forwards the output data to the model manager of the target model. Then, the model manager forwards the data to the target model. When the target model receives the data, it handles the data according to its specification. After the end of data handling, the target model updates its next simulation time and forwards it to the simulation manager through its model manager. Moreover, the model manager of the target model informs the model manager of the source model of response of the data. When the source model is informed of response of the data through its model manager, the source model updates its next simulation time and forwards it to the simulation manager. Figure 6 describes the process of data exchange using torpedo model and warship model in torpedo engagement example.

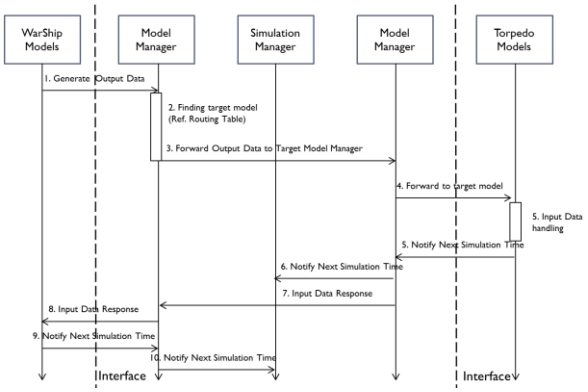

Fig. 6. Sequence diagram of data exchange

4 Case Study: Torpedo Engagement

Figure 7 shows the experimental design of the case study. The torpedo engagement simulation consists of the warship and the torpedo models. Each model is separated into three parts: sensor model, control model and maneuver model. These sub-models are developed as DEVS models (sensor, control model) and MATLAB model (maneuver model). For easy management of the simulation, we added Experimental Frame (EF) parts. The scenario Editor and statistical estimator are components of the EF. With separating EF and simulator, we can easily manage and obtain statistical data [10]. In particular, we use SIMDIS as a display tool for the simulation results [13]. Moreover, with SIMDIS, we can figure out the simulation status during the simulation at once. The IOM, which is used in the interoperation simulation, is represented in the Fig. 8. Scenario of this case study is same as that found in 3.2.1.

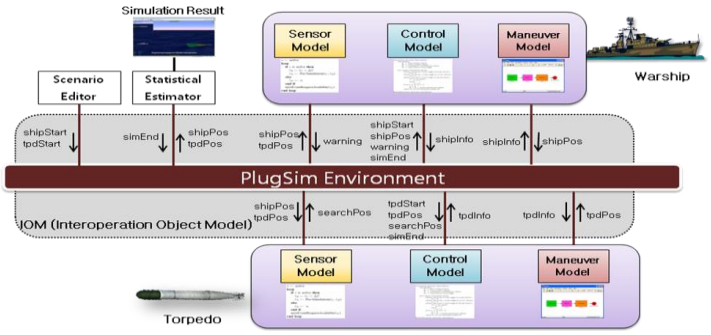

Fig. 7. Experimental design of the case study: torpedo engagement

With this case study, we have attempted to demonstrate two things: first, the interoperation simulation, which consists of the different kinds of models, can be con-

ducted in PlugSim environment; second, changing simulation models during the simulation is effective in some simulations. For the first aspect, we designed the case study with various kinds of models (the DEVS model and the MATLAB model).For the second, we designed the scenario of case study to emphasize the changing models during the simulation. In the wire-guidance torpedo, maneuver algorithm of the torpedo is changed by the submarine. In PlugSim Environment, maneuver model of the torpedo model can be changed during the simulation. Therefore, using PlugSim, It is very similar to the wire-guidance torpedo.

Figure 8 depicts changing the maneuver model of the torpedo model during the simulation and the simulation result of that changing on the SIMDIS. We checked that the movement of the torpedo changed by SIMDIS, when the maneuver model of the torpedo model changed.

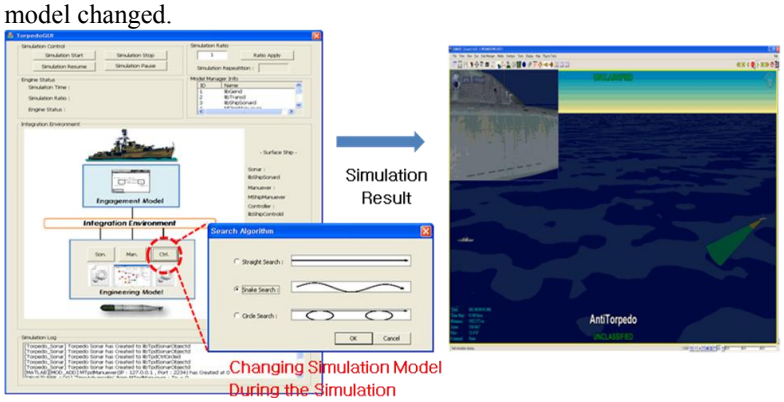

Fig. 8. Changing the maneuver model of torpedo during the simulation

5 Conclusion

In this paper, we introduce PlugSim environment. PlugSim supports interoperation simulation, which consists of various kinds of models (both DEVS and Non-DEVS model). Moreover, in PlugSim environment, simulation models can be changed during the simulation. In particular, we represent the data management and time synchronization in PlugSim, which are essential services in the interoperation simulation. We explain these services using torpedo engagement as an example. In the case study, we design the experiment using PlugSim and show that PlugSim environment can support the interoperation simulation and change the simulation models during the simulation. Changing simulation models during the simulation is particularly useful in some simulations, such as the development of tactics of the wire-guidance torpedo.

In the further work, we plan to develop PlugSim as a hybrid environment. For the hybrid environment, the time synchronization algorithm and interfaces need to be modified. Moreover, we plan to use Inter-Process Communication (IPC) as the communication method for efficient matter.

Acknowledgement

This research is supported by Ministry of Culture, Sports and Tourism (MCST) and Korea Creative Content Agency (KOCCA) in the Culture Technology (CT) Research & Development Program 2009

References

1. Hans L.M Vangheluwe "DEVS as a common denominator for multi-formalism hybrid system modeling", Proceedings of the 2000 IEEE International Symposium on Computer-Aided Control System Design, Anchorage, Alaska, USA, Sep 25-27, 2000.
2. Juan de Lara and Hans Vangheluwe "AToM^3 : A Tool for Multi-formalism and Meta-modeling" FASE '02 Proceedings of the 5th International Conference on Fundamental Approaches to Software Engineering, 2002.
3. Judith S. Dahmann and Katherine L. Morse "High Level Architecture for Simulation: An Update", Distributed Interactive Simulation and Real-Time Applications, Proceedings of 2nd International Workshop, p 32-40,Montreal, Que., Canada, July 19-20, 1998.
4. Gauthier Quesnel, Raphael Duboz, Eric Ramat, "The Virtual Laboratory Environment – An Operational Framework for multimodeling, simulation and analysis of complex dynamic systems", International Journal of the Federation of European Simulation Societies: Simulation Modeling Practice and Theory 17, pp. 641- 653, 2009.
5. Yong Jae Kim, Jae Hyun Kim and Tag Gon Kim, "Heterogeneous Simulation Framework Using DEVS Bus", *Simulation*, Vol. 79, No. 1, pp. 3 - 18, Jan., 2003.
6. Jang Won Bae and Tag Gon Kim, "DEVS Based Plug-in Framework," *Spring Simulation Multiconference 2010*., Orlando, FL, USA, pp. 147 - 153, Apr., 2010.
7. "IEEE standard for modeling and simulation (M&S) High level architecture (HLA) - framework and rules," IEEE Std 1516-2000, pp. i–22, Sep 2000.
8. "IEEE Standard for Modeling and Simulation (M&S) High Level Architecture (HLA) - Federate Interface Specification," IEEE Std 1516.1-2000, pp. i–467, 2001.
9. "IEEE standard for modeling and simulation (M&S) High level architecture (HLA)-object model template (OMT) specification," IEEE Std 1516.2-2000, pp. i–130, 2001.
10. Bernard P. Zeigler; Herbert Praehofer; Tag Gon Kim,Theory of Modeling and Simulation. ACADEMIC PRESS, 2001.
11. Kyung-Min Seo, Hae Sang Song, Se Jung Kwon and Tag Gon Kim, "Measurement of Effectiveness for an Anti-torpedo Combat System Using a Discrete Event Systems Specification-based Underwater Warfare Simulator," *The Journal of Defense Modeling and Simulation: Applications, Methodology, Technology*, Vol. 8, No. 3, pp. 157 - 171, July., 2011.
12. Chang Ho Sung, Jeong Hee Hong and Tag Gon Kim, "Interoperation of DEVS Models and Differential Equation Models using HLA/RTI: Hybrid Simulation of Engineering and Engagement Level Models," *2009 Spring Simulation MultiConf*., San Diego, CA, USA, Mar., 2009.
13. U.S. Naval Research Laboratory. *SIMDIS user's manual*. 2006.