Business Process Simulation: Transformation of BPMN 2.0 to DEVS Models

Hassan Bazoun, Youssef Bouanan, Gregory Zacharewicz, Hadrien Boyer, Yves Ducq

Business Process Simulation: Transformation of BPMN 2.0 to DEVS Models

Hassan Bazoun* **, Youssef Bouanan*, Gregory Zacharewicz*, Yves Ducq*, Hadrien Boye**

* **Univ. Bordeaux,**
IMS, UMR 5218 CNRS,
33405 Talence, France.

**** Hardis / Hardis Conseil,**
3 rue Guglielmo Marconi,
44800 Saint Herblain, France

hassan.bazoun@ims-bordeaux.fr, youssef.bouanan@ims-bordeaux.fr, gregory.zacharewicz@ims-bordeaux.fr, yves.ducq@ims-bordeaux.fr, hadrien.boye@hardis.fr

Keywords: BPMN, DEVS, Model transformation, Business Process simulation.

Abstract

Industrial enterprises have gradually moved their goals towards production of physical products supplemented by intangible services to differentiate themselves in a compatible market. The study of these services, their set up, and the evaluation of their efficiency is a rising research domain. In the frame of Model Driven Service Engineering Architecture (MDSEA), a service system is modeled from different point of views (static and dynamic) at the different MDSEA levels: Business Service Model (BSM), Technology Independent Model (TIM), and Technology Specific Model (TSM). Simulation is a dynamic feature of MDSE and which explains the need of coherent M&S formalisms for simulation activities. Accordingly, this paper presents the simulation of service systems based on DEVS models. It defines a transformation approach of BPMN models into DEVS simulation models based on the metamodel approach, and describes the enrichment of obtained DEVS models through performance indicators (time and costs).

1. INTRODUCTION

To remain competitive, a company must differentiate itself from other competitors. Since improving the product's performance can reach some limits, one open solution is to improve the enterprise service system, redefine its business processes and share more information (considered as additional services) with customers and suppliers.

In the frame of Model Driven Service Engineering Architecture (MDSEA) [Bazoun et al. 2014], a distinction can be made between static and dynamic service system modeling [Cardoso et al. 2012]. A business process is a series of activities that produces a product or service for a customer. Business Process Modeling (BPM) [Cardoso et al. 2012] results in a representation of an organization's business processes to be analyzed and improved [Weske 2007]. Business process's models provide a suitable static view, but frequently missing the temporal dimension to express output performance such as an expected cost or a desired duration. In detail, the impact of correct or incorrect behavior of complex models over time is not clearly visible using static view. This issue can be solved by running a business process simulation for analyzing and understanding the business process model according to its dynamic.

This paper presents research work results performed in the frame of the FP7 MSEE (Manufacturing Service Ecosystem) Integrated Project [FP7 2011]. The main result of MSEE is the development of a Model Driven Service Engineering Architecture (MDSEA). The first step of MDSEA concerns the transformation of Business models (represented with the Extended Actigram formalism) to Technical models (represented with BPMN [OMG 2011]) and has been presented in [Bazoun et al. 2013]. This paper introduces the second step. It defines a transformation of BPMN models into DEVS simulation models based on metamodel matching. This paper is presenting a work in progress, it draws the general overview of the work avoiding details due to unfinished work and space limits. The paper is organized as follows: first, a brief overview of the research literature studying the transformation BPMN to DEVS is proposed.Then the meta-models for BPMN and DEVS are presented. After that, the model transformation from BPMN to DEVS is explained. Finally, the perspectives of this work will be proposed at the end of this paper.

2. STATE OF THE ART

2.1. Transformation from BPMN to DEVS

In the context of BPMN to DEVS transformation, authors in [Cetinkaya et al. 2012] and [Mittal et al. 2012] presented a Model Driven Development (MDD) framework for modeling and simulation (MDD4MS). In the frame of this framework they defined a model to model transformation from BPMN as a conceptual modeling language to DEVS as a simulation model specification. BPMN and DEVS Meta-models were presented. In addition, a set of transformation rules were defined in order to transform BPMN models into DEVS models. According to these rules, some BPMN concepts (Pool, Lane, SubProcess) were mapped to DEVS coupled component, while Task, Event (Start, End, and Intermediate), and Gateway were mapped to DEVS atomic component.

Comparing the BPMN metamodel defined with the latest version of BPMN 2.0 metamodel [OMG 2011] we can conclude that several concepts are missing and thus were not transformed into their corresponding DEVS concept. Authors didn't mention the different types of BPMN Tasks (User Task, Manual Task, Service Task…) and BPMN Intermediate Events (Message, Signal…) that can be mapped differently when transformed into DEVS concepts. The difference would be in the number of states forming each DEVS Atomic Model. Based on these remarks, the work presented in this paper takes into consideration these points in an attempt to benefit from previous work and propose new mapping and transformation rules.

2.2. DEVS Simulators

Electing a target DEVS tool for model transformation requires a literature review of current DEVS Simulation tools. The DEVS group standardization maintains on its website the updated list of most used DEVS tools known by the DEVS community [Wainer 2013]. In [Hamri and Zacharewicz 2012], the authors have given a brief description and comparison of popular tools.

ADEVS was the first DEVS tool developed in C++ by the Arizona University. It consists in an ad-hoc simulator. DEVS abstract classes should be extended by users to define atomic and coupled models, and then the simulation can be launched. The drawback resides in the fact that users need programming skills to code the models.

DEVSJAVA is a Java framework in which the kernel simulator is ADEVS. It supports also modeling and simulation of DEVS with variable structures. However, at atomic level, the user should implement the corresponding DEVS behavior in Java (in our opinion the user has not enough skills to program his atomic models).

CD++ Builder is a DEVS modeling and simulation environment that integrates interesting features and facilities for the user. It allows modeling and simulation of other DEVS formalisms (cell-DEVS, Quantized-DEVS, etc). It provides a DEVS graphical editor to model coupled and atomic models, and to encapsulate them through components for further reuse.

Other DEVS tools are dedicated to specific areas. VLE, this is a C++ M&S framework that integrates heterogeneous models from different scientific fields. This integration is based on the agent paradigm. In addition, JDEVS is the Java implementation of a DEVS formal framework. It supports multi-modeling paradigms based on DEVS. It ensures the interoperability among the reused components. Also SIMSTUDIO can be considered, it is focused on a simplified DEVS editor for DEVS non Expert. The authors also investigate LSIS_DME that is focused on a graphical interface and code source generation in order to complete the model by complex Java functions.

At the end each DEVS editor is covering interesting aspects that complete basic DEVS facilities or propose different model views. Nevertheless we found it difficult to import by the tool non DEVS models other than hard coded matching, i.e. the customization is limited. We suggest that the feeding by other model can be facilitated if following a Model Driven approach, e.g. MDA. One core concept of MDA is the Meta Model that is required for model matching. In the paper [Garredu et al. 2012], a Meta model is proposed.

3. MODEL TRANSFORMATION FROM BPMN 2.0 TO DEVS MODELS

This section introduces the main transformation principles from BPMN model to DEVS model, including the transformation architecture, DEVS metamodel, the mapping of BPMN concepts to DEVS concepts, and the implementation using a transformation language.

3.1. Concept

3.1.1. Transformation Architecture

The metamodel approach [OMG 2003] is one of the most used transformation techniques. Figure 1 presents the metamodel approach adapted to the context of model transformation from BPMN2.0 model to DEVS model. Three different levels are identified: model, metamodel, and meta-metamodel. The BPMN model is the source model to be transformed, while the DEVS model is the target model resulting from the ATL transformation. BPMN and DEVS models conform to the BPMN 2.0 and DEVS metamodels respectively.

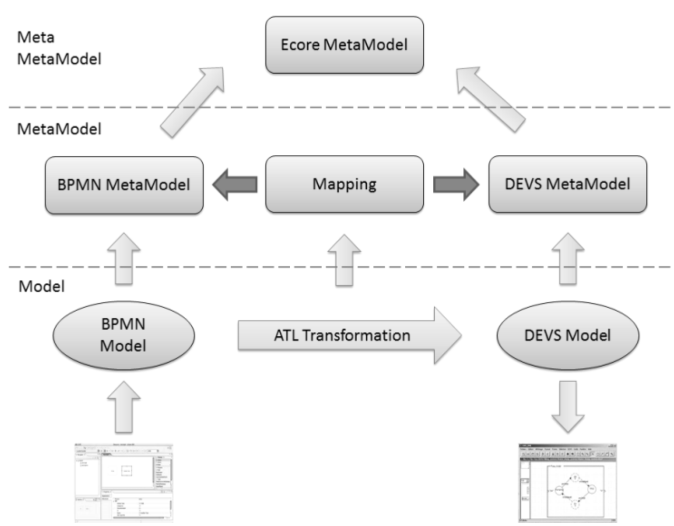

Figure 1.Transformation architecture

In addition both metamodels conform to a meta-metamodel named Ecore [McNeill 2010] metamodel (metamodels were developed using an Ecore based modeling framework). A mapping is defined between the concepts belonging to BPMN2.0 and DEVS metamodels. This mapping is implemented by ATL (Atlas Transformation Language) [ATL 2013].

3.1.2. BPMN and DEVS MetaModels

Source and target metamodels should be well identified to proceed with the transformation (Figure 1). BPMN 2.0 metamodel specified in [OMG 2011] is the source metamodel. There is no endorsed metamodel for the target DEVS metamodel, but several researches were held for the purpose of building a DEVS metamodel but a synthesis work is proposed in [Garredu et al. 2012]. The transformation from BPMN to DEVS models has required gathering previous works for setting a DEVS metamodel, as a result the authors proposed a simplified DEVS metamodel. It is used as a target metamodel which conforms to the DEVS specification [Zeigler et al. 2000].
Figure 2 presents the DEVS metamodel defined in Eclipse Ecore format.

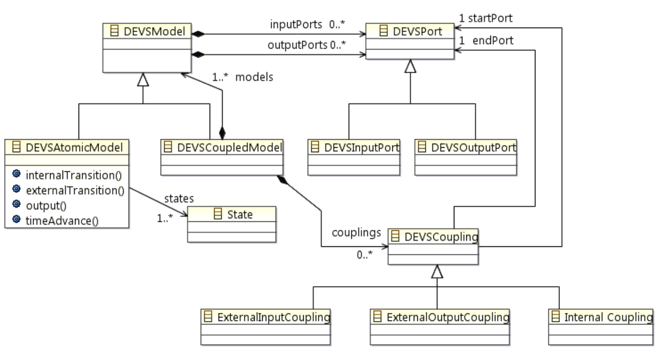

Figure 2. Simplified DEVS metamodel

In DEVS, there are two types of models: atomic and coupled models. Each model has a list of InputPorts and OutputPorts. An atomic model has four main methods: internal transition, external transition, output, and time advance. A coupled model is a decomposition of DEVS models (atomic or coupled) and DEVS Coupling. In addition, there are three types of coupling between ports: External Input Coupling (connections between the inputports of the coupled model and its internal components), External Output Coupling (connections between the internal components and the output ports of the coupledmodel, and Internal Coupling (connections between the internal components).

3.1.3. Mapping of concepts

The role of mapping in model transformation is to define links between concepts and relations from both metamodels (BPMN and DEVS). In [Mittal et al. 2012], a first mapping was proposed by the authors. Nevertheless, this early mapping didn’t distinguish all the various types of tasks and events existing in BPMN 2.0 which differ with respect to the potential situations a task might treat.

To complete this approach, different types of tasks are detailed (Receive task, Send Task, User Task, Service Task, and Manual Task); all of these tasks are mapped to “DEVS Atomic Model” concept but with different local behavior. This is also applied to intermediate events (Receiving and Sending Messages). Also we clearly distinguish between tokens and messages. The structure of tokens and messages is a multi-value event as described in G-DEVS [Zacharewicz 2008] that is implemented by one object with several variables. Each variable is representing one data. Some information of the token will be updated by the workflow according to actions defined in the task, current values of the token, and message received. At the end, the token reflects the path taken, the duration, etc. All the data are tracked in order to compute some performance indicators. This paper will not detail each concept, but only the most relevant are elaborated in the following.

3.1.3.1. Tasks

Basic Task model: a task is an activity where a work is performed by a resource. It consumes a certain amount of time. Token represents the work item with its arrival status. This status is evolving during simulation. At the end, token data are employed to analyze performance indicators regarding the service process completion.

A task is specified by the following parameters:

- Working time required to complete the task by a resource on a token.
- Once a task is executed the value of a token changes, the token is described by variables that are affected by the process.

To represent the behavior of a business process with some duration, the simulation component of the task will delay a token arriving at the port of entry for a specified period of time before sending it to the output port.

When a task is in the "Init" state, it means that no resource currently performs this task. Due to the arrival of an external event, the state changes to "State_X" with {X∈[1...*]}. Figure 3 is describing the basic task with its equivalent DEVS model according to DEVS graphical representation. The task is triggered by the token only. Then the activity executes during a certain period of time and after the token is released with some modification on its variable attributes.

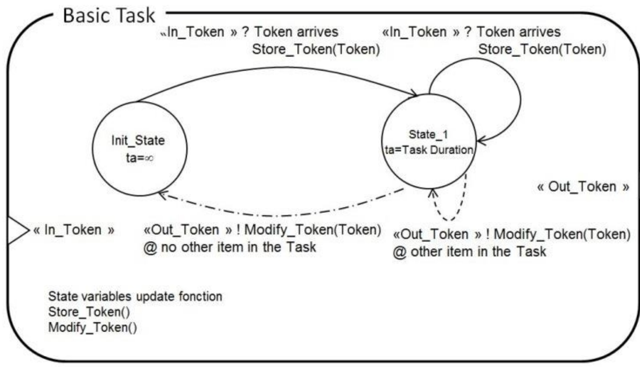

Figure 3.Basic Task DEVS State diagram

Reception Task Model: For a more accurate matching between BPMN model and DEVS model it has been chosen to distinguish the "Reception Task" from the "Basic Task" (Figure 4). The reason is based on the synchronization between the considered task and a triggering message that can come from another resource lane or pool. In that case the reception of the token is not sufficient to launch the task; the task is submitted to a triggering message.

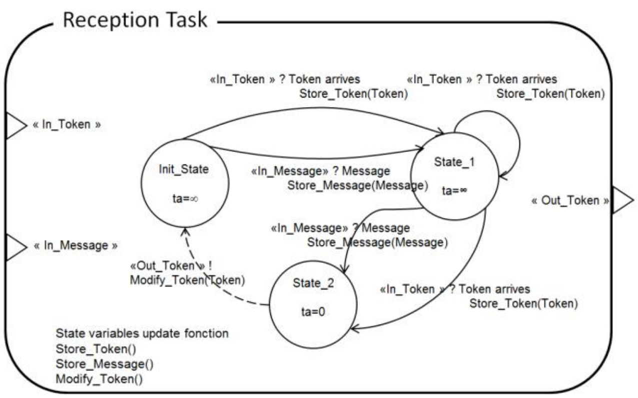

Figure 4.State diagram Task Reception Model

We distinguish two Types of Inputport: Message Object and Token Object. The outputport Type is only a Token Object. The execution of this task is based on the received input message's information that will be used to modify the token.

3.1.3.2. Events

The notion of Event is used to represent something that "happen" during the execution of the process, it represents a step in the process and its meaning differs from DEVS event. These events affect the flow of the process. There are three types of events, based on when they affect the flow: Start Event, Intermediate Event, and End Event. In this paper we will present an example of an Intermediate Event; Intermediate Reception Event (Figure 5).

An Intermediate Event can occur during a process flow. It means that a triggering event is required to continue the process. An Intermediate Event may occur on the edge of "Tasks" and "Sub Processes". In this case, it is a triggered event during the course of the activity. It indicates that an event coming from another lane or pool can occur between the beginning and the end of a process.

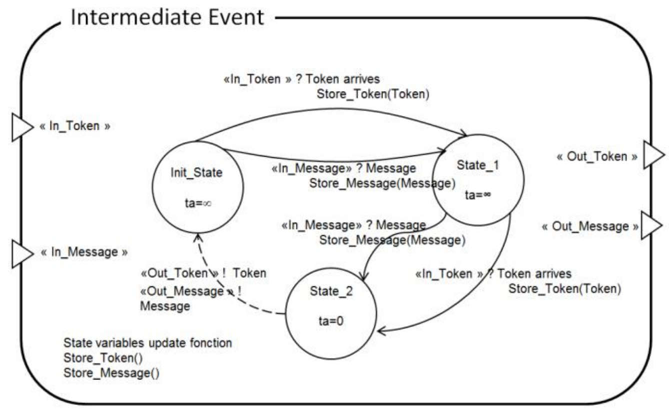

Figure 5.State diagram Intermediate Event Model

Table 1 presents a non-detailed mapping between BPMN and DEVS. It states the new concepts (*) added regarding the previous approaches in the literature introduced in section 2.1.

BPMN	DEVS
Pool	DEVS Coupled Model
Lane	DEVS Coupled Model
Sub process	DEVS Coupled Model
Flow Message Flow* Sequence Flow*	DEVS Atomic Model
Task Basic Task Send Task* Receive Task*	DEVS Atomic Model
Event Start* { Message, Timer, Conditional} Intermediate* {Message, Signal, Conditional} End* {Message, Timer, Conditional}	DEVS Atomic Model
Gateway Exclusive Gateway Inclusive Gateway* Parallel Gateway	DEVS Atomic Model

Table 1.BPMN elements to DEVS components

This conceptual mapping has been implemented into transformation rules using ATL transformation language. Each atomic component is generated from the BPMN model than the generated components are assembled in the coupled model.

3.2. Implementation

3.2.1. Transformation Language

ATL is a model transformation language specified as both a metamodel and a textual concrete syntax. In the field of Model-Driven Engineering (MDE), ATL provides developers with a mean to specify the way to produce a number of target models from a set of source models.

ATL is notable for its hybrid approach to model transformation. Most parts of a transformation to be implemented can be specified in ATL's declarative style. Because declarative style code is not as expressive as imperative code, some model transformation problems are hard to implement by using a declarative-only approach. Therefore ATL offers also support for imperative code. Imperative code can be used in do blocks of transformation rules, or completely separated in helper rules.

ATL-code is compiled and then executed by the ATL transformation engine. ATL supports only unidirectional transformations. ATL offers dedicated support for tracing. The order of the rule execution is determined automatically, with the exception of lazy rules, which need to be called explicitly. Helper functions provide imperative constructs. ATL does not support incremental model transformation, so a complete source model is read and complete target model is created.

An ATL M2M (eclipse) component is developed in the Eclipse Modeling Project (EMP). The ATL Integrated Environment (IDE) provides a number of standard development tools (syntax highlighting, debugger, etc.) that aims to ease development of ATL transformations. The ATL project includes also a library of ATL transformations. The project is using ATL M2M for compliance reason with SLMToolBox also developed under Eclipse and presented in the next section.

Due to non-sufficient space in this paper, transformation rules and specifications will be introduced in another paper representing the authors' final work.

3.2.2. SLMToolBox

SLMToolBox [Boye et al. 2014] is a software tool developed by Hardis [Hardis 2013] in the frame of MSEE project. The SLMToolBox will be used by enterprises willing to develop a new service or improve an existing one, within a single enterprise or a virtual manufacturing enterprise. The tool will be used at the stage of "requirement" and "design" of the service engineering process. The SLMToolBox is regarded to be an integration of several scientific concepts related to services into one tool. These concepts can be summarized by MDSEA methodology, services' modeling, engineering, simulation, monitoring and control.

The simulation feature is based on model transformation from BPMN to DEVS models. Source BPMN model is extracted from the BPMN graphical editor (integrated in SLMToolBox), a transformation engine is implemented based on ATL, and the output of this engine is DEVS model. A new developed version of [Zacharewicz et al. 2008] will be integrated in the SLMToolBox for graphical visualization and simulation of DEVS models.

3.3. Case Study

One use case model from the MSEE European project has been reused to serve in this research as a case study. The process consists in the creation of a cloth patron adapted and fitted to each client by tailoring thanks to customer data.

In the project, the modeling is starting from BSM level with an Extended Actigram model. Then the next step is going down to the BPMN model at TIM level. At this level before the creation of service from the model it could be valuable to simulate its behavior in order to correct potential errors of conception that can be detected through dynamical aspects not seen by reading a static model. The next part of the section will focus on the transformation to the simulation model.

One extract from the BPMN model is detailed in Figure 6. Two pools of the client and manufacturer are described in the use case model presented. In particular the sequence and the messages exchanged with the client are considered. The distinctive contribution of this research work permits first to differentiate the type of BPMN event. For instance the model shows an intermediary "Message Event". In addition, the task 1 is emitting a message to another blind pool (with basic a reception and triggering behavior). We consider this possibility as expressing representatively BPMN 2.0 collaboration model.

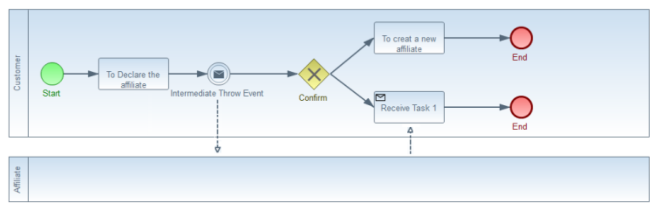

Figure 6.BPMN2.0 model for DEVS transformation

At DEVS level, the LSIS_DME editor [Zacharewicz et al. 2008] was tentatively selected to perform tests on the DEVS models obtained from BPMN matching before moving to final development stage, to the DEVS engine of the SLMTOOLBOX. One interest for the tool comes from the fact that it enables the creation, storage library, modification and composition of XML based models that can be feed in our case by the ATL transformation from ATL BPMN models. Also, the editor allows editing visually a model with geometric shapes representing the different elements of a DEVS atomic or coupled DEVS model.

Mapping realized the DEVS Coupled Model based on the library developed from BPMN components (Table 1) and integrated in the LSIS_DME DEVS models library of BPMN diagram. The DEVS coupled model presented in Figure 7 is the transformation results of the selected extract from the Figure 6 BPMN model of MSEE Case. Each atomic DEVS component is selected from the library and instantiated according to data values coming from the BPMN description. Then the models are coupled to represent the BPMN chain of tasks and it take into account resources represented by lanes. In this example we differentiate between a fully described lane and another non detailed lane (blind lane).

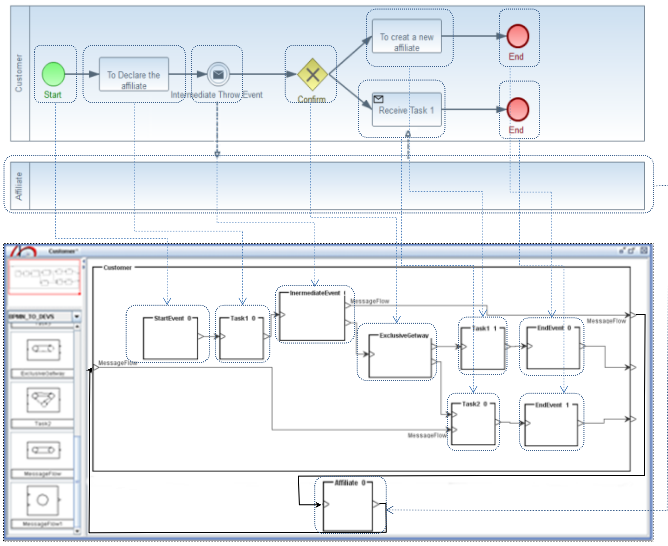

Figure 7. Equivalent DEVS model example in LSIS DME

Then Figure 7 has been run to present an extract of the simulation results provided by the tool. In this simulation it was confirmed that the token variables declared in the initial state of each "start event" atomic model can be followed in term of evolution of their attributes values accordingly to activities actions of the process and regarding time. The new values depend on the operation of the task and message received. The main idea resulting from the first simulations performed is the proof of feasibility in terms of definition and monitoring of quality indicators, the capacity to measure the impact of input factors and parameters. The goal is to provide simulation feedbacks to parameters tuning to reach as close as possible the services desired results.

The simulation result in Figure 8 shows an extract of the output of the simulation. The simulation has been set up to follow performance indicators on tokens that circulate through the different process's components. Tokens gather information on service development and its delivery, they can be considered as the memory of service development. For instance the time to complete the service delivery can be traced during the simulation. The number of resources called to achieve the service delivery process and the cost of material and human resources can be computed using the simulation. Another point is to analyze failure in the service delivery. Some service building can lead to bottle necks. Several scenarios can be proposed and run to evaluate the best one before the next implantation step: the architecture implementation.

At the moment, results are not handled to be displayed graphically nor interpreted by BPMN modeler.

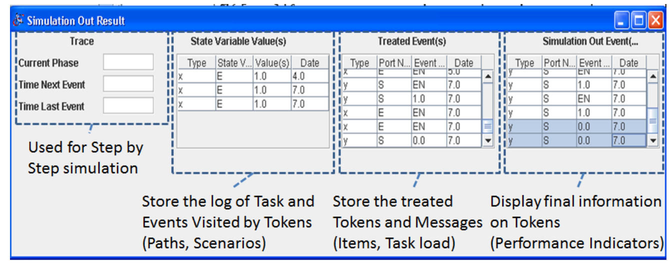

Figure 8. DEVS Workflow model results example

4. PERSPECTIVE

Transformation from BPMN models to DEVS models is one key step in a procedure covering business process modeling languages, model transformations, and simulation. It remains to visualize the DEVS models resulting from the transformation to be later displayed in a DEVS Graphical editor completely integrated in the SLMToolBox. The DEVS metamodel will be completed independently from any simulator's architecture to take into account multi value state variables. In addition, new features such as export format will be developed. Storage will be improved. Authors claims that the durability of this work relies on the adoption of the open platform. In addition, BPMN models (subject of simulation) will be animated for better understanding of the process. Thanks to the visualization of DEVS models, users will be capable of tuning more precisely performance indicators' values (time, costs and combined indicators) needed for simulation. The simulation results offer sufficient information needed for business process analysis, but the problem frequently faced is the lack of temporal data from enterprises because of the domain no long experience.

5. CONCLUSION

This paper introduced business process modeling and simulation in the frame of the Model Driven Service Architecture (MDSEA) project. As a result, it presented a transformation of BPMN models into DEVS models based on previous researches done in this domain. The approach has now proposed an exhaustive mapping, based on existing works plus additional concept mapping from BPMN concepts to DEVS concepts. It detailed also the transformation architecture, and an implementation in an M&S tool (SLMToolBox). The work is still ongoing, and it still lacks the final integration of the tailored simulation code in the SLMToolBox and the dynamic animation of the BPMN model. Also the implementation of the performance indicators is still under discussion.

Acknowledgement

This work has been partially supported by the FP7 Research Project MSEE "Manufacturing Service Ecosystem" ID 284860. http://www.msee-ip.eu/

References

[ATL 2013] *"ATL/User Guide – The ATL Language"* http://wiki.eclipse.org/ATL/(accessed 10 November 2013).

[Bazoun 2013]: Bazoun, H., Zacharewicz. G., Ducq. Y., Boye, H. *"Transformation of Extended Actigram Star to BPMN 2.0 and Simulation Model in the frame of Model Driven Service Engineering Architecture"*. TMS, (2013).

[Bazoun 2014]: Bazoun, H., Zacharewicz. G., Ducq. Y., Boye, H. *"SLMToolBox: An implementation of MDSEA for Servitisation and Enterprise Interoperability"*. Paper accepted in I-ESA (2014) 7th international conference.

[Boye 2014]: boyé, H., Bazoun, H., Belkhelladi, K. *"SLMToolBox: ATool Set For Service Engineering"*. Paper accepted in MODELSWARD 2014 2nd international conf on Model-Driven Engineering and Software Development

[Cardoso 2012]: Cardoso, J., Pedrinaci, C., Leidig, T., Rupino, P., De Leenheer P. *"Open semantic service net-works."* Paper presented at: The international Symposium on Service Science (ISSS); (2012).

[FP7 2011]: *FP7 – FoF-ICT-2011.7.3 – "Manufacturing SErvice Ecosystem Project- Annex 1 description of work" – July 29th 2011*. http://interop-vlab.eu/

[Garredu 2012]: Garredu, S., Vittori, E., Santucci, J-F., Bisgambiglia, P-A. *"A Meta-Model for DEVS Designed following Model Driven Engineering specifications*." SIMULTECH, page 152-157. SciTePress, (2012).

[Hardis 2013]: *Hardis is a software company with specialist expertise in management computing* http://www.hardis.fr/eng/ jsp/site/Portal.jsp (accessed 18 October 2013).

[Hamri 2012]: Hamri, M. and Zacharewicz, G. "*Automatic generation of object-oriented code from DEVS graphical specifications.*". In WSC'12. Article 409 , 12 pages, 2012.

[Cetinkaya 2012]: Çetinkaya, D., Verbraeck, A., Seck, M.D. *"Model Transformation from BPMN to DEVS in the MDD4MS Framework"*, TMS-DEVS, (2012): 304-309

[Mittal 2012]: Mittal, S., and Risco Martin, J.L. *Netcentric System of Systems Engineering with DEVS Unified Process.* 610-613, 2012. CRC Press.

[McNeill 2010] Ken McNeill *"How to extend the Eclipse Ecore metamodel."* http://www.ibm.com/developerworks/library/os-eclipse-emfmetamodel/index.html

[OMG 2011]: OMG, "*Business Process Model and Notation (BPMN) version 2.0*" document num: formal/2011-01-03.

[OMG 2003] : OMG, *"MDA Guide Version 1.0."* document number: omg/2003-05-01.

[Thoben 2001]: Thoben, K.-D., Jagdev, H., Eschenbcher, J. *"Extended Products: evolving traditional product concepts"* In the 7th International Conference on Concurrent Enterprising: Bremen, Germany, June 2001.

[Wainer 2013]: DEVS TOOLS, hosted by G. Wainer at Carlton University, November 2013, http://www.sce.carleton.ca/faculty/wainer/standard/tools.htm

[Zacharewicz 2008]: Zacharewicz G.; Frydman C.; Giambiasi N. "G-DEVS/HLA Environment for Distributed Simulations of Workflows", Simulation, 2008, 84(5), pp. 197–213

[Weske 2007]: Weske, M., 2007. "*Business Process Management: Concepts, Languages, Architectures*". New York, Springer-Verlag, (2007): p. 368.

[Zeigler 2000]: Zeigler, B.P., Praehofer, H. and Kim, T.G. *"Theory of Modeling and Simulation"*, NY, 2000.

Biography

Hassan Bazoun is a software engineer at Hardis and PHD student at IMS Lab. His research domain includes business process modeling, model transformation, and Discrete Event Simulation. He is involved in an European project (MSEE) and has participated in the development of the modeling tool SLMToolBox.

Youssef Bouanan

Gregory ZACHAREWICZ is Associate Professor at University of Bordeaux (IUT MP) with both competences in enterprise engineering and computer sciences. His research interests include Discrete Event Modelling (e.g. DEVS), Distributed Simulation, HLA, and Workflow. He has been involved in several European projects (e.g. MSEE and FITMAN). He is recently focused on Enterprise Modelling and Semantic Interoperability. He has published more than 50 papers in international journals and conferences.

Yves Ducq

Hadrien Boye is a project manager at HARDIS. He is in charge of the e-solution skills center in the western offices of the company. His main fields of interest include web technologies, software architecture and enterprise information systems.