G-DEVS / HLA Environment for Distributed Simulations of Workflows

Gregory Zacharewicz, Claudia Frydman, Norbert Giambiasi

▶ **To cite this version:**

Gregory Zacharewicz, Claudia Frydman, Norbert Giambiasi. G-DEVS / HLA Environment for Distributed Simulations of Workflows. SIMULATION: Transactions of The Society for Modeling and Simulation International, 2008, 84 (5), pp.197-213. 10.1177/0037549708092833 . hal-00204203

HAL Id: hal-00204203
https://hal.science/hal-00204203

Submitted on 12 Sep 2016

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

G-DEVS / HLA Environment for Distributed Simulations of Workflows

Gregory Zacharewicz
IMS-LAPS / GRAI UMR CNRS 5218 - Université de Bordeaux
351, cours de la Libération, 33405 - Talence cedex, FRANCE
gregory.zacharewicz@u-bordeaux1.fr

Claudia Frydman, Norbert Giambiasi
LSIS UMR CNRS 6168 - Université Paul Cézanne
Avenue Escadrille Normandie Niemen, 13397 - Marseille cedex 20, FRANCE
{claudia.frydman, norbert.giambiasi}@lsis.org

In this paper, we present a Workflow environment allowing distributed simulation based on DEVS / G-DEVS formalisms. A description language for Workflow processes and an automatic transformation of a Workflow into a G-DEVS model have been defined. Then, we introduce a new distributed Workflow Reference Model with HLA-compliant Workflow components. We detail the HLA objects shared between Workflow federates and we present the publishing/subscribing status of each of these federates. Finally, we illustrate the use of this distributed environment with an example of a Microelectronic production Workflow.

Keywords: Workflow, DEVS, G-DEVS, validation, XML, distributed simulation, Interoperability, HLA.

Pre Print

1. INTRODUCTION

Workflow Management Coalition (WfMC) provides a good framework to develop business process. The description of a Workflow may involve a process model, different programs, and actors which are essential to its execution. This description is user-oriented and does not need to develop programming code (it can be automatically generated from a graphical description). But the drawback is there is no clear simulation semantics associated to these Workflow engines. Almost of these engines are ad hoc. This fact may lead to errors that are difficult to detect.

DEVS, Statecharts, Petri nets are well-known formalisms to describe the behavior of complex discrete event systems. They give formal frameworks in which modeling and simulation processes are clearly separated. DEVS seems to be more general and flexible than the other formalisms. However Workflow users are not familiarized with DEVS. Thus we propose a set of rules (grouped in form of an algorithm) that transforms automatically a Workflow specification into a G-DEVS model.

The paper is organized as follows. Section 2 gives an overview of Workflow, G-DEVS and HLA. Section 3 illustrates the proposed approach to transform a Workflow specification into G-DEVS. Section 4 proposes a new Workflow Reference Model HLA compliant and Section 5 illustrates it on an industrial application.

2. RECALLS

2.1. Workflow

According to WfMC in [1], a Workflow is the automation of a business process, in whole or part, during which documents, information or products are passed from one participant (program, machine or human) to another for action, according to a set of procedural rules.

The WfMC has purpose to develop standards in the field of Workflow [2] [3]. It particularly defines an architectural representation of a workflow management system, identifying most important system interfaces, mostly adopted in the Workflow management field (cf. Figure 1).

This representation contains the process definition tool (to describe a model of the process), the administration tool (to control and monitor the process execution), the Workflow client application (to implicate human-machine interface in the process), the invoked applications (to interface with specific application computation not tackled by the model) and the facilities to link with other Workflow environments. We focus on the process definition phase to make it computerized.

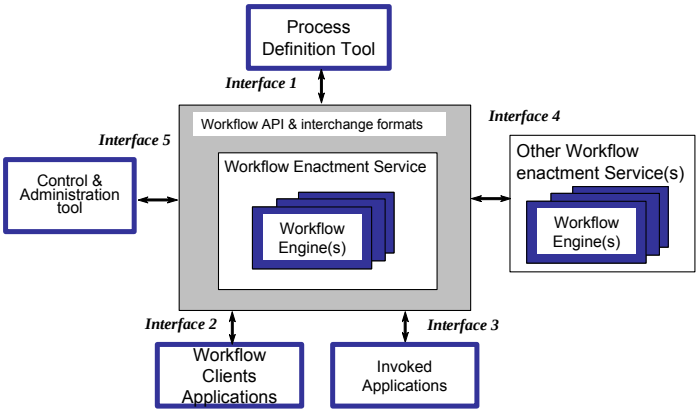


Figure 1. Workflow Reference Model – Components & Interfaces

	Tibco [3]	Workey [5]	Yasper [6]	Yawl [7]
Development	Commercial	Commercial	University	University
Underlying M&S formalism	ad hoc	ad hoc	Petri nets	Petri nets
Workflow type	Administrative groupware	Administrative	Administrative productions	Administrative productions
Exportation of internal model	Not available	Not available	Not available	Not available
Simulation	No	No	yes	yes

Figure 2. Workflow Environments

A Workflow consists of procedures named also tasks and logical expressions or controllers that describe the roads (routes, flows) of items. A Workflow is a graphical representation (specification) in which tasks are represented with rectangles, controllers with nodes and arrows which determine the flows between tasks.

There are many environments that allow the specification and the simulation of Workflows. Most relevant Workflow tools are presented in Figure 2. Most of professional tools (e.g. Tibco [3] and Workey [5] are most relevant administrative Workflow environment) are based only on ad hoc execution software engines, so they do not take profits of concepts offered by the discrete event simulation theory [8]. For example, this theory separates the modeling phase from the simulation one allowing the reuse of the validated specifications to other applications.

In Yasper and Yawl, developed respectively by the University of TU/Eindhoven [6] and the University of Queensland [7], the specification and execution are separated. The authors developed an editor tool (in order to have a graphic process definition) and an execution engine based on the Petri net formalism. They argued the choice of using Petri nets by the following reasons:

- Formal semantics despite the graphical nature,
- State-based instead of event-based,
- Abundance of analysis techniques.

We believe that a simulation tool based on the Discrete Event System Specification (i.e. DEVS [9]) formalism can enhance the simulation and validation processes. Ziegler in [9] discusses that DEVS modeling is more accurate than Petri nets modeling due to the facts that:

- DEVS gives a more general framework for modeling and simulation of complex systems,
- DEVS integrates naturally the notion of time contrary to Petri nets which require an extension of the formalism,
- DEVS offers a formal (and separated from model definition) definition of the simulator.

2.2. G-DEVS

Traditional discrete event abstraction (e.g. DEVS) approximates observed input-output signals as piecewise constant trajectories. Generalised-DEVS (G-DEVS) defines abstractions of signals with piecewise polynomial trajectories [10]. Thus, G-DEVS defines the coefficient-event as a list of values representing the polynomial coefficients that approximate the input-output trajectory. Therefore, a DEVS model is a zero order G-DEVS model (the input-output trajectories are piecewise constants).

G-DEVS possesses the concept of coupled model introduced in [9]. Every basic model of a coupled model interacts with the other models to produce a global behavior. The basic models are, either atomic models, or coupled models stored in a library. The model coupling is done using a hierarchical approach.

The concept of abstract simulator of [9] to define the simulation semantics of the formalism can be used for G-DEVS models. The architecture of the simulator is derived from the hierarchical model structure. Processors involved in a hierarchical simulation are Simulators which insure the simulation of the atomic models, Coordinators, which insure the routing of messages between coupled models, and the Root Coordinator, which insures the global management of the simulation. The simulation runs by exchanging specific messages (corresponding to different kind of events) between the different processors. The specificity of G-DEVS model simulation is that the definition of event is a list of coefficient values as opposed to a unique value in DEVS.

2.3. Distributed Simulation System: HLA (High Level Architecture)

The High Level Architecture (HLA) is a software architecture specification that defines how to create a global simulation composed of distributed simulations. In HLA, every participating simulation is called federate. A federate interacts with other federates within a HLA federation, which is in fact a group of federates. The HLA definitions set gave place to the creation of the standard 1.3 in 1996, which then evolved to HLA 1516 in 2000 [11].

Pre Print

The interface specification of HLA describes how to communicate within the federation trough HLA specification implementation: the Run Time Infrastructure (RTI).

Federates interact among them using the services proposed by the RTI. They can notably "Publish" to inform about an intention to send information to the federation and "Subscribe" to reflect some information created and updated by other federates. The information exchanged in HLA is represented in the form of classical object oriented programming. The two kinds of object exchanged in HLA are Object Class and Interaction Class. The first kind is persistent during the simulation, the other one is just transmitted between two federates. These objects are implemented with Extensible Markup Language (XML) format. More details on RTI services and information distributed in HLA are presented in [12] and [11].

In order to respect the temporal causality relations in the simulation; HLA proposes to use classical conservative or optimistic synchronization mechanisms [13].

2.4. G-DEVS / HLA Components mapping

We proposed, in [14], an environment, named DEVS Model Editor (LSIS_DME), for creating G-DEVS models HLA compliant and simulating them in a distributed fashion.

In LSIS_DME, a G-DEVS model structure can be split into federate component models in order to build a HLA federation (i.e. a distributed G-DEVS coupled model). The environment maps DEVS Local Coordinator and Simulators into HLA federates, it maps Root Coordinator into RTI. Thus, the "global distributed" model (i.e. the federation) is constituted of federates intercommunicating. The Figure 4 illustrates the decomposition into federates AB and ACD (i.e. Figure 4 b)) of a G-DEVS coupled model A (i.e. Figure 4 a)).

The G-DEVS models federates intercommunicate by publishing and subscribing to HLA interactions (Figure 3) that map the coupling relations of the global distributed coupled model. The first attribute is the type of the message that qualifies if the message is an external event or an init message. The second identifies the model that emits the message. The third precise the port concerned. The fourth defines the degree of the polynomial function in the case of G-DEVS models. The last attribute is the (or the list of) event values.

Parameter Table

Interaction	Parameter	Datatype	Available Dimensions	Transportation	Order
CouplingRelation	Message Type	HLAASCIIchar	TypeMessage	HLAreliable	TimeStamp
	Transmitter	HLAASCIIstring	NA		
	Event time stamp	HLATimeType	NA		
	Concerned Port	HLAASCIIstring	NA		
	Event dimension	HLAinteger32LE	NA		
	Event Value	HLAopaqueData	NA		

Figure 3. HLA Parameter Table

The information containing events exchanged between distributed coupled models is routed between federates by the RTI in respect to time management and Federation Object Model description. The federation execution is based on conservative synchronisation algorithm and event-driven mechanism. The Lookahead used in this environment, (useful data for conservatives distributed simulations) computing was improved in [15] regarding to previous approach proposed in [16] and [17].

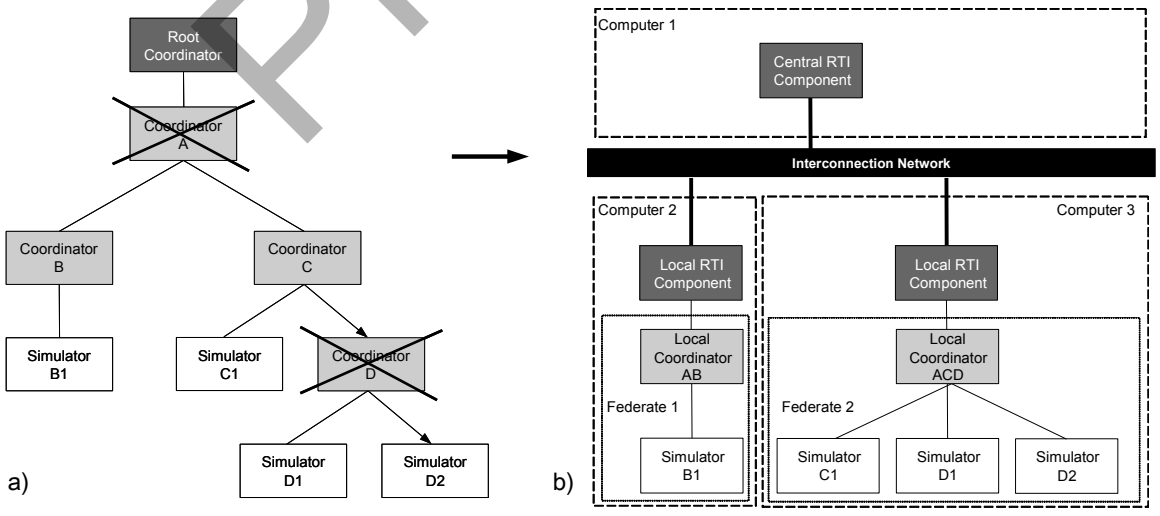


Figure 4. G-DEVS distributed simulation structure

Pre Print

3. TRANSFORMATION OF WORKFLOW SPECIFICATIONS INTO G-DEVS MODELS

Workflows are most commonly graphically modeled. The limitation of this representation comes from the fact it is not based on strong formal concepts. Thus, it does not allow properties of semantic verification and validation of the model. Furthermore, these models are often simulated by ad-hoc engines that could not be compared in terms of correctness and efficiency regarding to others.

One solution is to use or to define a unified language for the specification of Workflow in order to be applied as a common output of Workflow editors. This language will support algorithms to transform a Workflow model into a classical formal specification for simulation, regardless of the Workflow editor.

3.1. Workflow Structure

The WfMC proposed an XML representation of Workflow that is accepted as a standard in Workflow community [18]. The XML Workflow process model structure correctness can be certified by referring to a Workflow Document Type Definition (DTD). This XML representation is not fully convenient for the XML specification of production Workflow.

In details, on the one hand, specificities of data transiting in a flow of production need to be identified in order to be handled by production software and exploited at the end of flow. In the other hand some definition of this DTD are relative to administrative Workflow, they are not required for the kind of Workflow under our scope and overcast the description for non Workflow expert users.

Thus, we propose a simple language to represent the components involved in Workflow dedicated to the representation of production systems.

We describe a Workflow Model **MWF** structure as composed of the following basic components:

<Name, RESS, A, Ct, L, IT, ST>
Name, variable containing the name of the Workflow,
RESS, is the list of the resources of the Workflow,
A, is the list of the tasks of the Workflow,
Ct, is the list of the controllers of the Workflow,
L, is the list of the link of the Workflow,
IT, is the list of the items of the Workflow,
ST, is the list of the stocks of the Workflow.

A XML Workflow process model is composed of tasks components that treat items and controllers components that route items between tasks. Items pass over a sequence of these components. This model could be transformed into a coupled G-DEVS model by coupling G-DEVS atomic models representing the Workflow basic components. This G-DEVS model takes advantage of formal properties enounced in § 2.1 and can be simulated.

We propose a general method in three steps, described in Figure 6, to transform a Workflow process model into a G-DEVS model. This method is applied, notably, for transforming Workflow models of an industrial process of electronic components manufacture operated by STMicroelectronics on its production site of Rousset.

In the following, we detail the transformation of a simple model from this industrial process. This model is depicted in Figure 5 with the graphical Workflow Model Editor developed at LSIS (LSIS_WME). It represents the high-level Workflow model of an assembly line of chips named Route H80XX. This model consists of a initialization task (Play symbol), an end task (Stop symbol), six tasks on electronic wafers, named operations (Oper…), and two controllers (OR-join and OR-split) to route the electronic wafers.

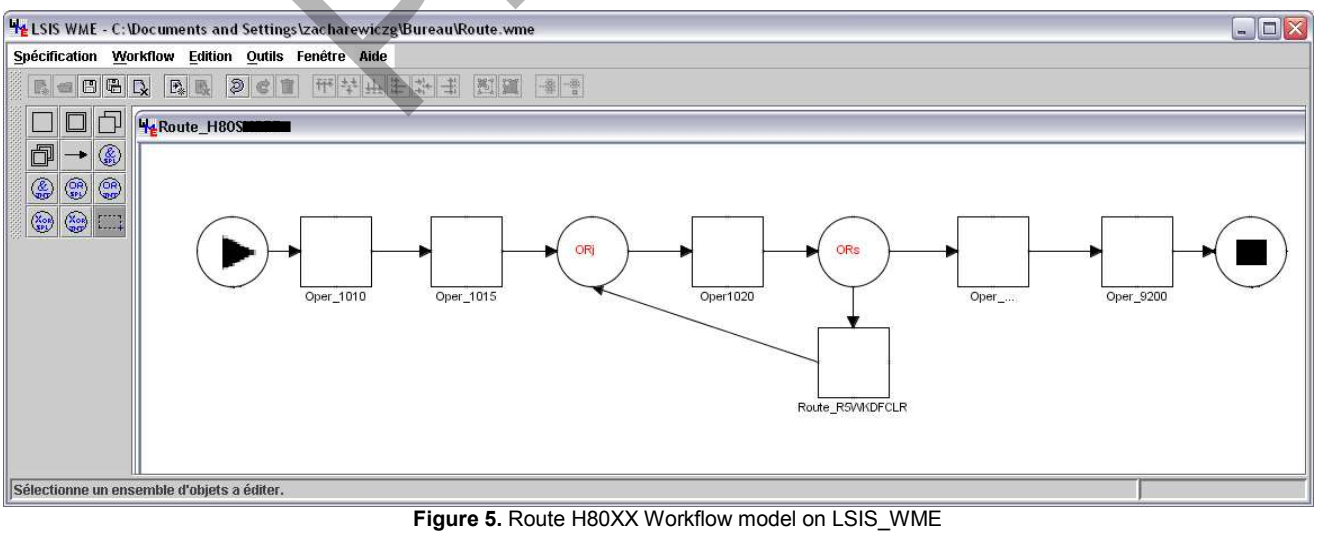


Figure 5. Route H80XX Workflow model on LSIS_WME

Pre Print

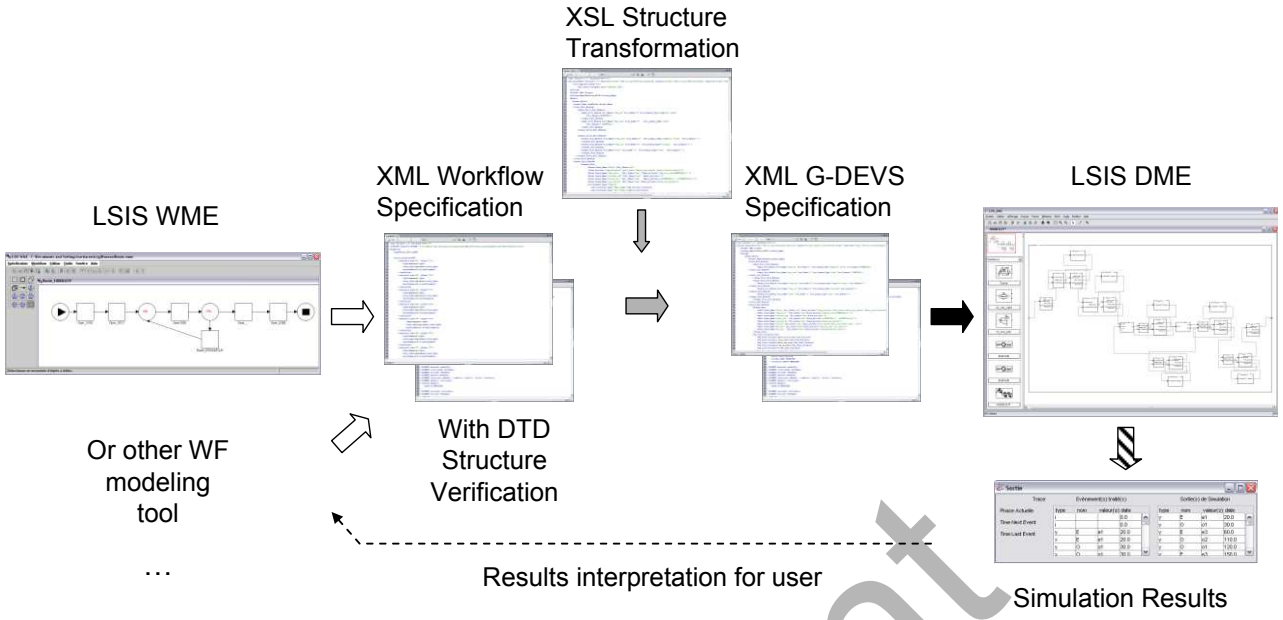


Figure 6. Workflow model to G-DEVS coupled model

3.2. Steps from Workflow model to G-DEVS model

In a first step, the Workflow graphic model is exported from LSIS_WME (or another Workflow editor) in the Workflow XML format presented above (cf. Figure 6 white arrows). Some extracts of the XML format of Route H80XX are given in Figure 7**.** They define the basic components: task OPer_1010, resource R1, arc between OPer_1010 and OPer_1015, controller OR join and input item wafer1. The correctness of the structure of this XML Workflow is verified by a Workflow XML DTD.

```
<workflow> <nomWF>H80XX</nomWF>
   <List_ressourcesWF>
        <ressource name="R1" taches="Oper1010">
        <type>human</type> <sous_type>operator</sous_type>
        <performance>0.8</performance>
        </ressource>
    ...
    <List_ taskWF>
        <task> <task_A>
        <TA nameTA="Oper1010" stockTA="st1" ressources="R1">
        <t_exec_moyen>120</t_exec_moyen
        <actionTA>wafer inspect</actionTA>
        </TA>
    ...
    <List_controllersWF>
        <controller>
        <AndJoin nameAJ="or_join1">
        <inputsAJ>
        <inputAJ nameinputAJ="OJ_in1"> </inputAJ>
        <inputAJ nameinputAJ="OJ_in2"> </inputAJ>
        <fjointureAJ>wafer_jonction</fjointureAJ>
        </AndJoin>
    ...
    <List_arcsWF>
        <arc reference="1"
        source ="Oper1015" destination ="or_join1"
        portdest="OJ_in1">
        </arc>
    ...
    <List_itemsWF>
        <item name="wafer1"> <typeit>symbolic value</typeit>
        <valueit>row_wafer</valueit> </item>
    ...
```

Figure 7. XML Workflow

In a second step (cf. Figure 6 grey arrows), we apply an Extensible Stylesheet Language (XSL) on the XML structure of the Workflow in order to modify its tree presentation to a XML G-DEVS coupled model structure. The algorithm defines the coupling between atomic models that represent Workflow tasks and controllers. It adds also in the XML G-DEVS implicit information from Workflow. For instance, the coupling of resources on tasks (for their synchronisation) is defined as G-DEVS coupling relation. Stock models are also coupled with tasks. The Figure 8 illustrates extracts of XML G-DEVS coupled model corresponding to extracts of XML Workflow of Figure 7. It presents an input port definition, some atomic models included with the link to their XML definition and some G-DEVS coupling relations.

```
<Library_Name>Workflow_DEVS</Library_Name>
<Model> <Coupled_Model>
    <Coupled_Name>model_WF_H80XX</Coupled_Name>
    <Ports_Unit_ModelC>
        <Input_Ports_Unit_ModelC>
        <Input_Port_ModelC
            Port_Name=" arriv_st1"
            Port_Rank="0" Port_Domain_Type="Symbolic value"
            Port_Domain=""/>
    ....
    <Included_Models_Unit>
        <Included_Model
            Model_Name="Oper1010" Model_Type="TA"
            Model_Style="atomic"
            File_Reference="Oper1010.XML"
        <Included_Model
            Model_Name="R1" Model_Type="ressource"
            Model_Style="atomic"
            File_Reference="R1.XML"
        <Included_Model
            Model_Name=" or_join1" Model_Type="controller"
            Model_Style="atomic"
            File_Reference="or_join1.XML"
    ...
    <Coupling_IC
        Included_Model_Src="Oper1010"
        Output_Port_Included_Model_Src="Item_out"
        Included_Model_Trg="or_join1"
```

Pre Print

```
        Input_Port_Included_Model_Trg=" OJ_in1"/>
    ...
    <Coupling_EOC
        Included_Model_Src="End_Task"
        Output_Port_Included_Model_Src="item_out"
        Current_Model="model_WF_H80XX"
        Output_Port_Model_Trg="global_item_out"/>
    ...
    <Coupling_EIC
        Current_Model="model_WF_H80XX"
        Intput_Port_Model_Src="global_item_in"
        Included_Model_Trg="Begin_Task"
        Input_Port_Included_Model_Trg="item_in"/>
    ...
```

Figure 8. XML G-DEVS Coupled Model

The XSL algorithm also defines a G-DEVS atomic model for each resource, task, stock and controller by a XML G-DEVS description. In order to illustrate the XML representation of G-DEVS atomic models, we present in Figure 9 the atomic model of the OR-join controller of the Route H80XX. This simple atomic model possesses three phases, its behavior is designed to route items by transmitting one item on its output port from one or two received on its input ports regarding to condition on items.

```
<Library_Name> Workflow_DEVS</Library_Name>
<Model> <Atomic_Model>
    <Atomic_Name>or_join1</Atomic_Name>
    <Ports_Unit_ModelA>
        <Input_Ports_Unit_ModelA>
        <Input_Port_ModelA Port_Name="OJ_in1" Port_Rank="0"
        Port_Domain_Type="Symbolic value" Port_Domain=""/>
        <Input_Port_ModelA Port_Name="OJ_in2" Port_Rank="0"
        Port_Domain_Type="Symbolic value" Port_Domain=""/>
        </Input_Ports_Unit_ModelA>
        <Output_Ports_Unit_ModelA>
        <Output_Port_ModelA Port_Name="item_out" Port_Rank="0"
        Port_Domain_Type="Symbolic value" Port_Domain=""/>
        </Output_Ports_Unit_ModelA>
    </Ports_Unit_ModelA>
    <States_Unit_ModelA>
        <Phases_Unit>
        <Phase Phase_Name="wait" Init_Phase="y" Actions=""/>
        <Phase Phase_Name="S_OJ_in1" Init_Phase="n"/>
        <Phase Phase_Name="S_OJ_in2" Init_Phase="n"/>
        </Phases_Unit>
    </States_Unit_ModelA>
    <Functions_Unit_ModelA>
        <Transition_Functions_Unit>
        <Internal_Transitions_Unit>
        <Internal_Transition Transition_Src="S_OJ_in1"
        Transition_Trg="wait"
        <Internal_Transition Transition_Src="S_OJ_in2"
        Transition_Trg="wait"
        </Internal_Transitions_Unit>
        <External_Transitions_Unit>
        Transition_Src="wait"
        Transition_Trg="S_OJ_in1" Event_Variable_Name="OJ_in1"
        Event_Value="EVENTVAL" Transition_Elapsed_Condition=""
        Transition_Actions=" var_OJ_in1=EVENTVAL[0];" Transi-
        tion_Conditions=" EVENTVAL[1] > 10"/>
        <External_Transition Transition_Src="wait"
        Transition_Trg="S_OJ_in2" Event_Variable_Name="OJ_in2"
        Event_Value="EVENTVAL" Transition_Elapsed_Condition=""
        Transition_Actions=" var_OJ_in2=EVENTVAL[0];" Transi-
        tion_Conditions=" EVENTVAL[1] > 30"/>
        </External_Transitions_Unit>
        </Transition_Functions_Unit>
        <Out_Functions_Unit>
        <Out_Function Phase_Src="S_OJ_in1"
        Event_Variable_Name="item_out"
        Event_Value="var_OJ_in1" Out_Function_Conditions=""/>
        <Out_Function Phase_Src="S_OJ_in2"
        Event_Variable_Name="item_out"
        Event_Value="var_OJ_in2" Out_Function_Conditions=""/>
        </Out_Functions_Unit>
        <Timelife_Functions_Unit>
        <Timelife_Function Phase_Src="wait" Ta="Infinite"/>
        <Timelife_Function Phase_Src="S_OJ_in1" Ta="0"/>
        <Timelife_Function Phase_Src="S_OJ_in2" Ta="0"/>
        </Timelife_Functions_Unit>
    </Functions_Unit_ModelA>
</Atomic_Model> </Model>
```

Figure 9. OR JOIN XML G-DEVS Atomic Model

In a third step (c.f. Figure 6 black arrows), LSIS_DME loads data from the XML description for each G-DEVS atomic models (e.g. task duration functions, required resources, stocks capacities...) to instantiate atomic templates of G-DEVS models (eg. Figure 10).

3.3. Atomic G-DEVS models templates

Basic components of Workflow allow instantiation by parameterization of G-DEVS atomic model templates.

The atomic G-DEVS template model of task is presented Figure 10 conforming to graphical representation of [19]. This model is initialized in a "waiting of item to treat" state. Further to the reception of an item, the task has to make a demand of allocation to a resource model. If this resource is available, the task can process the work on the item and then release it. If the resource is not available, the item is put in wait, and a new demand of allocation is asked to the resource after the duration of occupation communicated by this last one.

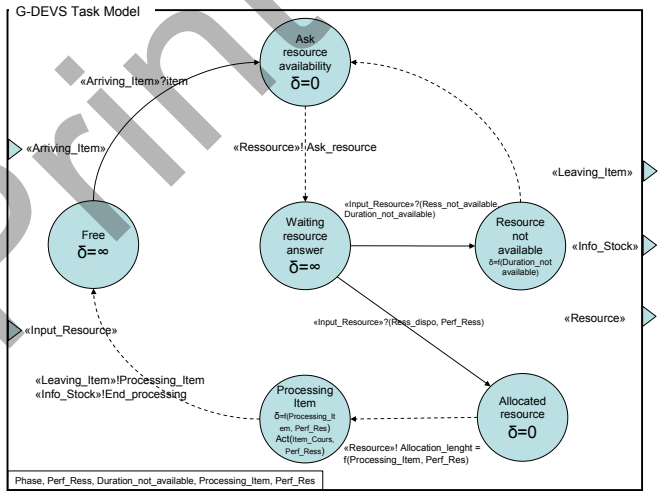


Figure 10. Task G-DEVS Atomic Model

The atomic G-DEVS template model of component Stock (cf. Figure 11) has for function to pile and to depilate the working items that arrive on a busy task.

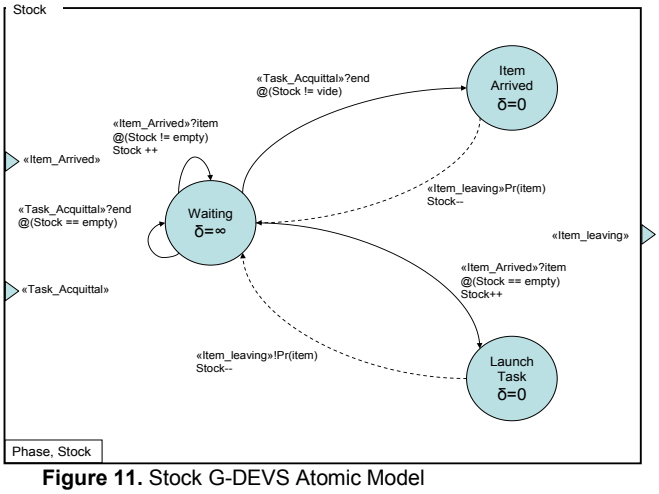


Figure 11. Stock G-DEVS Atomic Model

Pre Print

The G-DEVS Resource model template (cf. Figure 12) contains an initial "Free" state. When a demand is received from a task in this state, the resource answers its availability. Then, it communicates with the Task the duration of allocation and transit to a "Resource Busy" state, and cannot thus any more, be assigned to another task for a fixed duration. As a consequence, the other Task models making a demand of allocation will receive a negative answer to their request. When the duration of allocation is elapsed, the state of the model becomes again free and thus available on the allocation by a task.

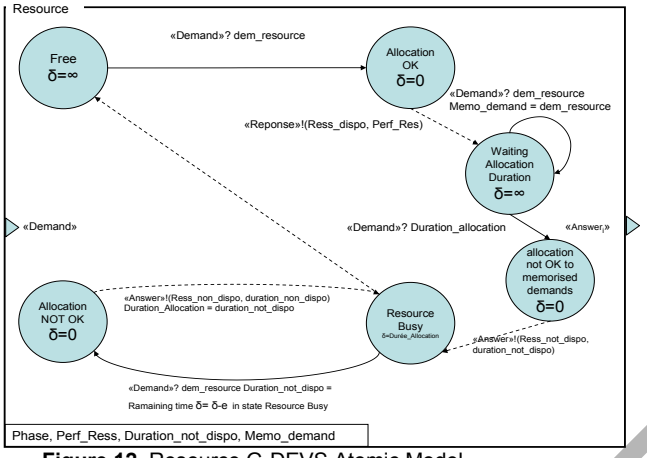


Figure 12. Resource G-DEVS Atomic Model

3.4. Generating coupled G-DEVS models

In addition to atomic models instantiation, LSIS_DME reads the XML description of included G-DEVS models coupling, to generate a (graphically editable) G-DEVS coupled model representing the global Workflow process. The Figure 13 presents the G-DEVS coupled model of the route H80XX. We can notably remark that Workflow implicit components: Resources and Stock have been added in this model in order to define a model able to be simulated (i.e. Corresponding blocks groups of same color are identified in the Figure 13 between Workflow model and G-DEVS model).

Furthermore, Workflow items are mapped into G-DEVS coefficient events with the list of values containing: item reference, values, time of processing and routing information. These items, planned in the Workflow, are inserted in the event scheduler of the G-DEVS simulator.

Finally, the simulation of the Workflow coupled G-DEVS model is run. The result of the simulation gives a log report containing the output events generated (cf. Figure 6 striped arrows). The results of the simulation are interpreted and sent back to LSIS_WME in order to be understandable by the user of the Workflow modeling tool. It characterizes item processing time, item accumulation in stocks, bottlenecks, resource allocation and synchronization.

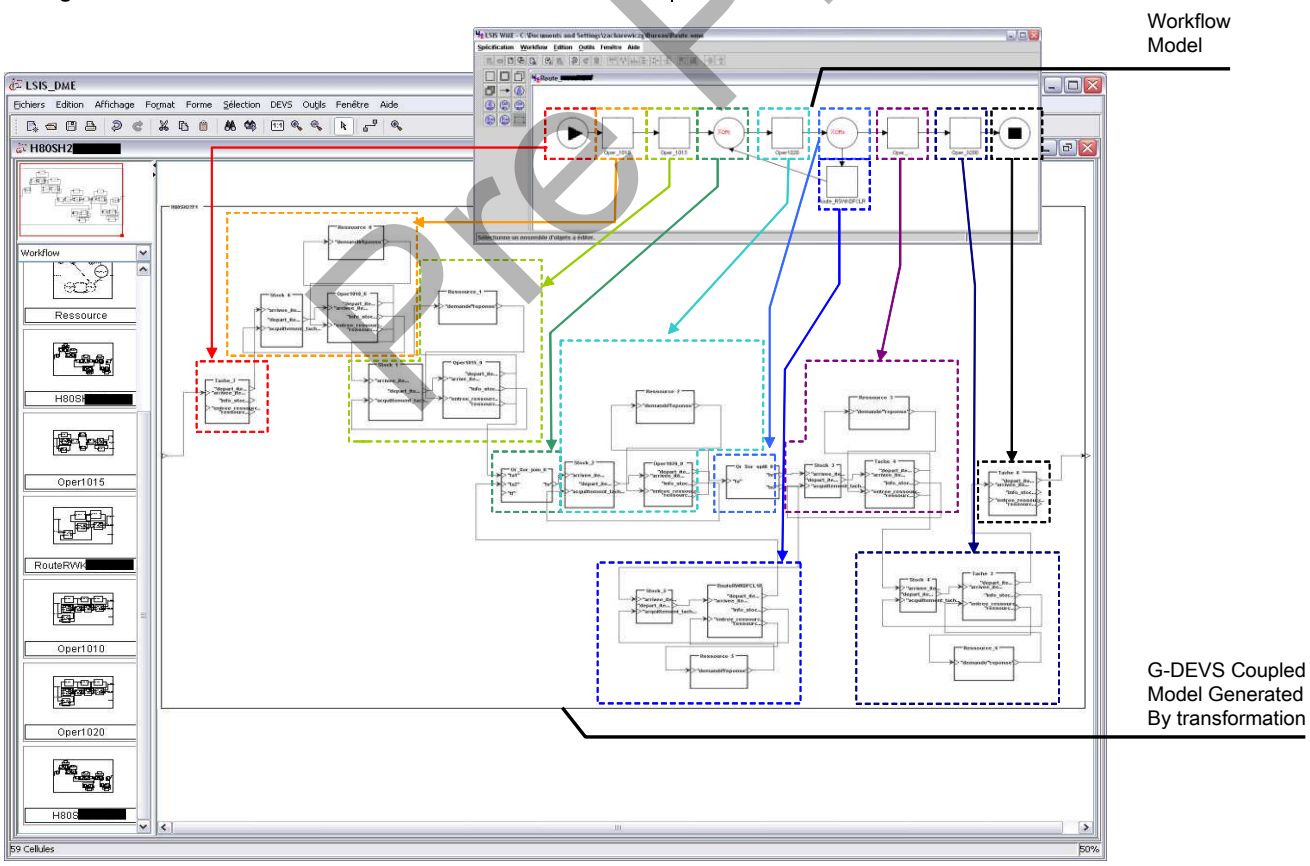


Figure 13. Route H80XX G-DEVS coupled Model

Pre Print

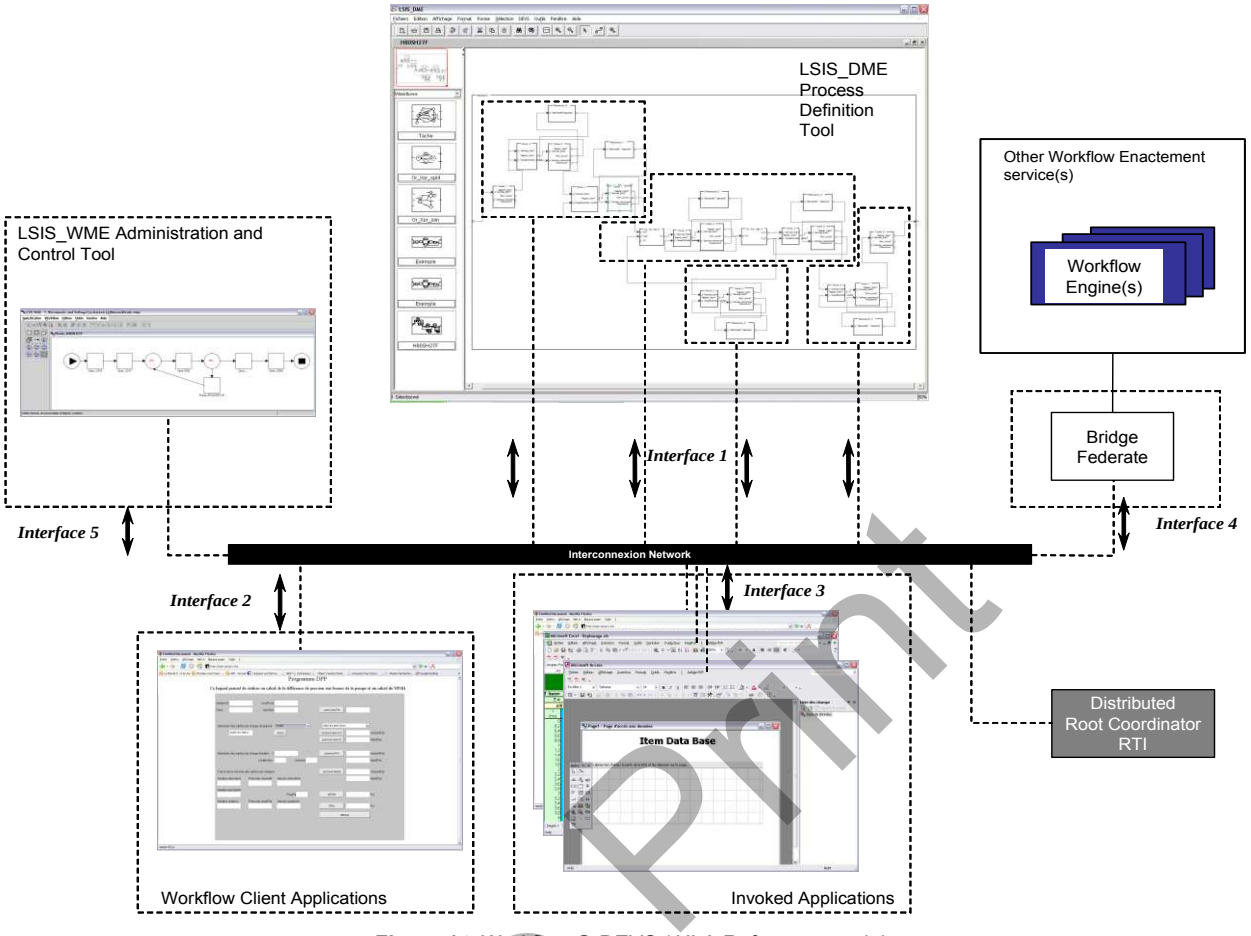


Figure 14. Workflow G-DEVS / HLA Reference model

4. G-DEVS Workflow Environment HLA Compliant

We presented, in § 2.4, that G-DEVS models can be run from several computers thanks to the capability of LSIS_DME to create HLA federates. This capability matches with the distribution requirements of actual industrial processes [20]. Indeed, actual real industrial processes required the use of human decision and multiple softwares that interact with the process at the different steps of its execution. These softwares are heterogeneous and need to cooperate. A key to these requirements is to use the HLA standard as a common way to share synchronized data between them. Thus, we propose to use LSIS_DME as the engine of a distributed Workflow environment and to generalize the HLA compliance to the whole Workflow environment by adding other federates to the federation in order to define a Distributed Workflow Reference Model. This innovative architecture is presented in Figure 14.

4.1. Distributed Workflow M&S Environment

Therefore, we included the modeling tool LSIS_WME into a federate. The models defined in XML generated by this federate are integrated into HLA objects and shared with LSIS_DME. In detail, LSIS_WME publishes to HLA objects that represent the components of the Workflow model and LSIS_DME subscribes to. The updates of information are routed by the RTI. If the Workflow model is modified by the user of LSIS_WME, LSIS_DME is informed of these changes; it could take them into account in its G-DEVS model and reruns the simulation with the new model structure and atomic models settings.

During the simulation, LSIS_DME updates, in a HLA object, the log of events resulting of the simulation. These results are subscribed by LSIS_WME to give to the users the simulation animation and results. For this reason, this software can be seen as the modeling, control and administration tool of the Workflow environment.

Moreover, in the Workflow definitions [18], client and invoked applications can be called during the run time in order to process computation not tackled by the models and their simulators.

On the one hand, we propose to integrate human in the loop, to do qualitative choice during the simulation. For that purpose, we implement web interfaces called during the simulation, by the Workflow engine, in order to specify for instance some routing of items in the process. Data exchanged during the call are defined in HLA objects.

On the other hand, some complex mathematical computation of data treatment, are not taken into account in

Pre Print

transition/output functions of the G-DEVS model described with LSIS_DME. In that case the simulation is interrupted and the data are transferred to specific software by publishing to an object. This software computes and sends back data to the process definition tool by publishing to a HLA object or interaction.

Finally, we also intend the possibility of interfacing with other Workflow environments using the concept of bridges federates [22]. In this last interfacing, it will remain to define what data to share between Workflows, concretely defining HLA objects to be exchanged between federates.

4.2. Workflow Federation Object Model HLA tables

We detail in this part the data shared between the different distributed components of the Workflow.

Creating HLA Object Class Table

The HLA classes of shared object specified for the Workflow federation must be specialized according to the example considered, namely in our case the Route H80XX. All of these classes are presented Figure 15.

The interface 1 possesses here three child classes. The fist and the third get back the information of the current state of the model to display it on the animated monitoring tool. The status of the Items which circulate by event exchange in the G-DEVS model during the simulation is also registered.

The interface 2 possesses two sub classes that get back the information resulting of a human-machine interface. In the considered route, it defines, on one hand the possibility to add information in items for routing before injecting them in the XORj controller component, and on the other hand by modifying the Item manually in a not automatic operation.

The interface 3 manages the information generated by the applications involved automatically in Workflow. For instance automatic decisions made from drawing of a curve of productivity and storage / retrieving of data in a data base.

The interface 4 allows sharing information with another Workflow federation. To connect the Workflow, we propose to use the works of [22] that register a common federate (considered as a bridge federate) in the two federations that intend to exchange data. This solution enhances security and confidentiality of data in and between the federations.

Finally, the interface 5 allows taking into account the modifications on the process (in term of structure and parameter setting) which can be brought by a user using the tool of monitoring and management.

Creating HLA Object Attributes Table

The attributes of the objects shared in the Workflow federation detail data shared in the Workflow federation.

These attributes contain the information concerning the exchanged items, the information exchanged with the called applications and the client applications and finally the information concerning the modifications of Workflow process structure.

Pre Print

Object Class Structure Table				
HLA object Root (N)	Interface (S)	Interface 1 (PS)	Contained Events (PS)	Contained Item (PS)
				Exit (Released) Item (PS)
			Table of models current state(PS)	Stocks Contents (PS)
			Start / End information of simulation (PS)	Indication of Start / End Oper 1015 (PS)
				Indication of Start / End Oper … (PS)
				Indication of Start / End Route_RWKDFCL1R (PS)
		Interface 2 (S)	Modification of Item contents for routing (PS)	Modification of Item contents for routing Xoj after 1015 (PS)
			Modification of Item contents by operation not handled by Workflow (PS)	Modification of Item contents by operation not handled by Workflow (PS)
		Interface 3 (S)	Information resulting from the called aplication (PS)	Automatic Treatment of a subpart of Road RWKDFLC1R (PS)
		Interface 4 (S)	Information towards federate to bridge several Workflow Federations (PS)	(to be completed)
		Interface 5 (S)	Information to administrate modification of the Workflow model (PS)	Modification Oper 1010 (PS)
				Modification Oper 1015 (PS)
				Modification Oper … (PS)
				Modification Oper 9020 (PS)
				Modification Oper 1020 (PS)
				Modification Routes RWKDFCL1R (PS)
				Modification Coupling H80SH27F (PS)
			Administration Information to set Clients/Invoked Applications (PS)	Excel Parameter tunning for treatment of Road RWKDFCL1R (PS)

Figure 15. HLA object table of Workflow environment federation

4.3. Mechanism of Publication / Subscription

Application of Monitoring and Management

A Workflow environment has to contain a tool of monitoring and management (i.e. [2]). We chose to integrate the tool of modelling LSIS_WME developed in our lab. Indeed, this tool for graphical representation of workflow can be used to display an animation, in order to follow the evolution of the simulation. By using this tool a user can also choose to stop the simulation, to modify the structure of the model of the Workflow process and to restart the simulation from a given point (a state, and timestamp) by taking into account structure modifications.

The HLA standard is usually employed to integrate the functions of remote monitoring on diverse applications. Numerous examples are presented in the literature to illustrate this kind of application; we can refer to the application of naval Fleet modelling and distributed simulation of the Italian modelling group SIREN [21].

The federate of monitoring and management embedding a dynamic displayable version of LSIS_WME application that subscribes to information supplied by the federate that run the simulation. On its side, it will assure the broadcasting of modifications of the structure of Workflow by publishing an object describing the new structure of the model. This mechanism is illustrated in Figure 16.

Invoked Applications and Client Applications

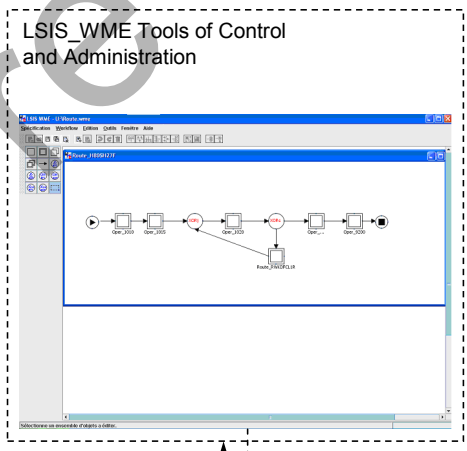
A Workflow environment has to be interfaced with applications and tools which will be called to achieve not modeled processing on items transiting during the Workflow simulation (i.e. [18]). A human user can also be involved to treat data with a graphic interface, during the simulation. For example to make choices of paths in the workflow process not fully specified in the model, or to make a qualitative choice on the processed data not fully specified in the model simulated by the environment. In addition specific applications from the industrial field (to complex to be integrated in the model because of a black box behavior) can be called to make automatic processing on the data. In our case the processing of productivity measurements and analysis are made and the storage of data in a data base will be made by external specific applications that exchange data with the environment trough Interface 3 as depicted in Figure 17.

The federates embedding these applications will thus subscribe to information supplied by federate assuring the simulation. On their side, they will process the received information (automatically or with human in the loop) in order to produce data to reinsert in the simulation. Then, they will guarantee the distribution of the information resulting from a human interface use or from an invoked program by publishing to an object describing the new information to be taken into account during the next steps of the simulation of the model.

Communication with other Workflow Environment

In the specification of this distributed environment, we also consider the possibility to have connections with others Workflow environments through bridges federate. This solution is based on the works of communications between federations proposed by [22]. This solution is considered but not yet implemented in the environment.

Pre Print



Interface 1 (PS)	Contained Events (PS)	Contained Item (PS)
		Exit (Released) Item (PS)
	Table of models current state(PS)	Stocks Contents (PS)
	Start / End information of simulation (PS)	Indication of Start / End Oper 1015 (PS)
		Indication of Start / End Oper … (PS)
		Indication of Start / End Route_RWKDFCL1R (PS)

Interface 5 (S)	Information to administrate modification of the Workflow model (PS)	Modification Oper 1010 (PS)
		Modification Oper 1015 (PS)
		Modification Oper … (PS)
		Modification Oper 9020 (PS)
		Modification Oper 1020 (PS)
		Modification Routes RWKDFCL1R (PS)
		Modification Coupling H80SH27F (PS)
	Administration Information to set Clients/Invoked Applications (PS)	Excel Parameter tunning for treatment of Road RWKDFCL1R (PS)

Figure 16. Interfacing with a Graphical Monitoring component

Interface 1 (PS)	Contained Events (PS)	Contained Item (PS)
		Exit (Released) Item (PS)
	Table of models current state(PS)	Stocks Contents (PS)
	Start / End information of simulation (PS)	Indication of Start / End Oper 1015 (PS)
		Indication of Start / End Oper … (PS)
		Indication of Start / End Route_RWKDFCL1R (PS)

Interface 2 (S)	Modification of Item contents for routing (PS)	Modification of Item contents for routing Xoj after 1015 (PS)
	Modification of Item contents by operation not handled by Workflow (PS)	Modification of Item contents by operation not handled by Workflow (PS)

Interface 3 (S)	Information resulting from the called aplication (PS)	Automatic Treatment of a subpart of Road RWKDFLC1R (PS)

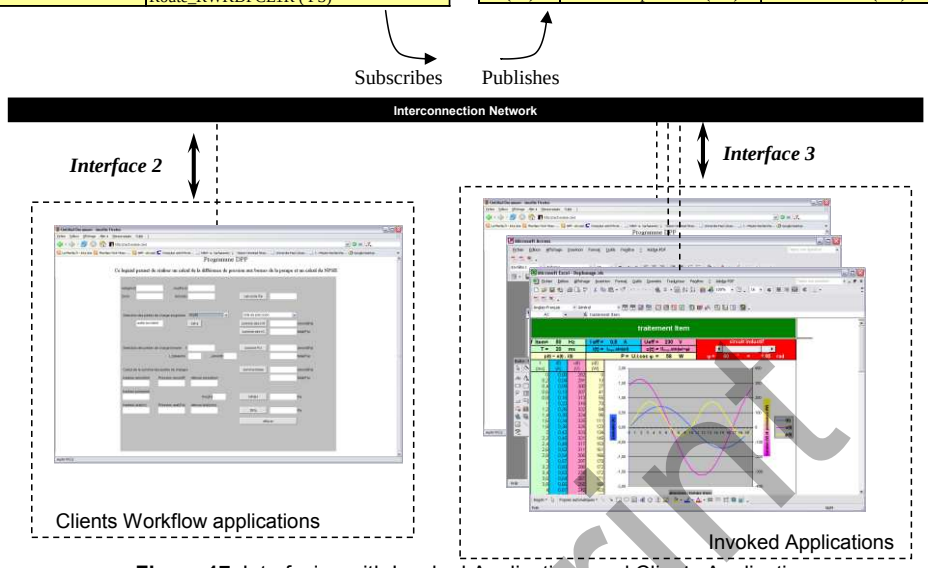


Figure 17. Interfacing with Invoked Applications and Clients Applications

5. M&S of Route H80XX case

The Route H80XX (presented in Figure 5) was modeled in the environment LSIS_DME. We created several sets of events planned in input of the process and several parameters settings of the atomic models, which characterize the tasks processing, to test the limits of the procedures H80XXconfiguration. For confidentiality reasons, these sets of data and the parameter settings of the models do not result directly from a real case. The input events (representing the wafer product named FDXX) were initialized with values allowing defining relevant characteristics. For example, a given percentage of FDXX was defined as defective in output of certain operations.

5.1. Simulation Results of Route H80XX

We present in Figure 18 some simulation results of 4 sets of 100 items FDXX planned in the input scheduler of the model H80XX with different percentage of defective items in Oper 1020.

We observe, in particular, the average duration of items transit. These data are determined from the log of simulation results containing the successful, processed and generated events (items), of the coupled model G-DEVS of the Route H80XX.

Despite the simplicity of this Workflow, this information allows determining a sensible economic planning of items FDXX and can indicate a questioning of the chains of tasks. In particular, the analysis of the Stock of Oper 1020 indicates us an important mean-level, in the case of a number of defective item equal to 30 percents.

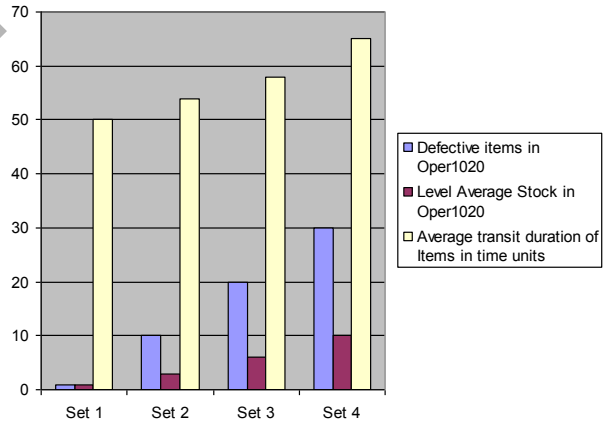


Figure 18. Simulation Results of coupled G-DEVS Route H80XX with LSIS_DME

From this observation of process, using this simple log of output events and in the hypothesis of 30 percents of defective products, we suggest to modify the Workflow process. In concrete terms, we propose to parallelise Oper 1020 into Oper1 1020 and Oper2 1020. We obtain a quasi identical Route to H80XX, except ORj controller is now connected to a new Ors controller that splits item regarding to availability of the Oper1 1020 and Oper2 1020. In

Pre Print

addition, another ORj controller is added between Oper1 1020 and Oper2 1020 output and the existing ORs.

The modified Workflow model indicates, by simulation a reduction of the items transit durations to approximately 56 units of time, still in the case of 30 percents defective Items referenced FDXX in output of task Oper1 1020 and Oper2 1020.

5.2. Experience acquired on Route H80XX case

The modeling and simulation of a concrete industrial case (i.e. the Route H80XX), in the distributed Workflow environment developed in this STMicroElectronics partnership project, have allowed us to validate the structure coherence of the proposed distributed Workflow benchmark presented Figure 14.

Indeed, the results of simulation has supplied us rational information notably on the duration of items transit, the levels of stocks during run time, the bottlenecks which can occur on the production chain, etc. We have also measured the run time performance of the simulation and keep in mind that this Workflow environment employs HLA for an interoperability purpose rather than for the runtime performance obtained.

These results will be employed to assist the human expert decision-making involved within the framework of the modification of the flow of procedures in the production lines of STMicroelectronics. They will allow anticipating the errors that can occur resulting of a wrong modification of a flow of procedures. It will also allow comparing quantitatively several modifications of a procedure to determine the most efficient.

6. CONCLUSION

This paper presented a new transformation algorithm from Workflow XML specification to G-DEVS models. The use of the G-DEVS formalism to represent Workflow allows bringing safety of a formal specification. In addition, we proposed a new Workflow environment HLA compliant using G-DEVS. The use of the HLA specification has facilitated connecting new HLA compliant components in the Workflow environment.

The development of the proposed environment has been validated by software tests and experts of the domain; however it still requires the addition of some improvement that are under the scope of our studies at the moment.

First improvement will be the integration of other client and invoked applications into the Workflow environment by turning them into HLA federates. Furthermore, Workflow is built by collecting information resulting from domain experts; who define more often temporal information using mean values of task duration. Thus, it would be interesting to envisage a modeling based on Min-Max DEVS [23] and [24] that might allow a more realistic simulation. In addition, the capability of the environment to modify the structure of a process during the simulation could be facilitated by using the dynamic structure of DEVS simulator referenced in [9]. Finally, we envisage using the environment as a plug-on real process of Workflow to be able to manage “on line” the data flow that follows the flow of real materials. The application will stand in that case for a real time environment. This last improvement will require operating sensors on the real system, and the simulation will be constrained to be as fast as wall clock time [26].

As a final point, the environment has produced sound simulation results of real processes models that represent STMicroelectronics company electronic wafers assembly lines. This models based on real cases has permitted us to corroborate the interest and the efficiency of the Workflow environment concepts presented in this article. These workflow environment concepts can be, from that postulate, abstracted to the level of generic production Workflow.

Pre Print

ACKNOWLEDGE

This work is financially supported by the “Communauté du Pays d’Aix”, “Conseil Régional Provence Alpes Côte d’Azur”, “Conseil Général Des Bouches-du-Rhône” and STMicroelectronics – Zone industrielle de Rousset 13106 ROUSSET cedex, France.

REFERENCES

[1] Courtois T. 1996. “Workflow: la gestion globale des processus” Logiciels & Systèmes no11, (Sept) 46-50.

[2] Workflow Management Coalition. 1999. *Terminology & Glossary.* WfMC-TC-1011, 3.0, Feb.

[3] Hollingsworth D., "The Workflow Reference Model," Workflow Management Coalition Spec., WfMC-TC-1003, Jan. 1995.

[4] Staffware. Staffware, Palo Alto, California, 2001, http://www.staffware.com, http://www.tibco/com.

[5] C-Log. Workey, C-Log International. 480, Place de la Courtine 93194 Noisy le Grand, 2006, http://www.c-log.com/040_Workey.

[6] Van Hee K., R. Post and L. Somers. 2005. “Yet Another Smart Process EditoR” *ESM* 2005, (Porto 24-26 Oct)

[7] Yawl. Yawl, 2005, http://www.yawl-system.com/.

[8] Van der Aalst W.M.P., and K.M. van Hee. *Workflow Management: Models, Methods, and Systems*. MIT press, Cambridge, MA, 2002.

[9] Zeigler B.P.; H. Praehofer; T. G. Kim. 2000. *Theory of Modeling and Simulation*. 2nd Edition, Academic Press, New York, USA.

[10] Giambiasi N., B. Escude and S. Ghosh. 2000. “G-DEVS A Generalized Discrete Event Specification for Accurate Modeling of Dynamic Systems”, Transac-

tions of the SCS International, 17(3):120-134.
[11]Institute of Electrical and Electronic Engineers 2001. *High Level Architecture (HLA) - Federate Interface Specification* IEEE Standard for Modeling and Simulation (M&S). std 1516.2-2000, March.
[12]Defense Modeling and Simulation Office (DMSO), U.S. Dept of Defense. 1998. *High Level Architecture Rules* Version 1.3. IEEE P1516.2, Standard for M & S, April.
[13]Fujimoto R. M., 1998, "Time Management in the High Level Architecture". Transaction of the SCS, Simulation, vol. 71, no.6, (December): 388-400.
[14]Zacharewicz G., N. Giambiasi, C. Frydman "Lookahead Computation in G-DEVS/HLA Environment", in: *Simulation News Europe Journal (SNE) special issue 1 "Parallel and Distributed Simulation Methods and Environments"*, vol. 16, n° 2, pp. 15 - 24, September 2006. ISSN, 0929-2268.
[15]Zacharewicz G., N. Giambiasi, C. Frydman ``Improving the Lookahead Computation in G-DEVS/HLA Environment", in: *9th IEEE International Symposium on Distributed Simulartion and Real-Time Applications (IEEE DS-RT 2005)*, IEEE, pp. 273 - 282, Montreal, Qc., Canada, 10-12 October 2005. ISBN 0-7695-2462-1
[16]Zeigler B.P., G. Ball, H.J. Cho and J.S. Lee, "Implementation of the DEVS formalism over the HLA/RTI: Problems and solutions", In *Simulation Interoperation Workshop (SIW)*, number 99S-SIW-065, Orlando, FL, 1999.
[17]Lake T., B.P. Zeigler, H.S. Sarjoughian and J. Nutaro, "DEVS Simulation and HLA Lookahead" In *Simulation Interoperability Workshop (SIW)*, number 00S-SIW-160, Orlando, FL, 2000.
[18]Workflow Management Coalition. 2005. *Workflow Process Definition Interface -- XML Process Definition Language (XPDL)*. WFMC-TC-1025, Oct.
[19]Song, H.S. and Kim, T.G., "The DEVS framework for discrete event systems control", in *5th Annual Conference on AI, Simulation and Planning in High Autonomous Systems*, (Gainesville, FL, USA, 1994), 228 - 234.
[20]Zacharewicz G., C. Frydman and C. Zanni. 2002. "Workflow/HLA Distributed Simulation Environment", in: Harbour, Maritime & Multimodal Logistics M&S, HMS 2002, (Bergeggi, Italy, Oct 3-5).
[21]Revetria, R., NAVI: Learning HLA from an Interactive Exercise Tutorial. in *SCSC 2003* (Montreal, Canada, 2003).
[22]Bréholée B. and P. Siron. 2003. "Design and Implementation of a HLA Inter-federation Bridge" In Proceedings of the EUROSIW, (Stockholm, Sweden, 16-19 June)
[23]Giambiasi N. and S. Ghosh. 2001. "Min-Max Devs: A New Formalism for the Specification of Discrete Event Models With Min-Max Delays". European Simulation Symposium, (Marseille, France, Oct 18-20), 616-621
[24]Hamri M.A., N. Giambiasi, C. Frydman ``Min-Max DEVS Modeling and Simulation", in: Simulation Modelling Practice and Theory (SIMPAT), vol. 14, n° 7, pp. 909-929, October 2006. Ed. Elsevier, ISSN 1569-190X.
[25]Baati L., C. Frydman, N. Giambiasi "Simulation Semantics For Dynamic Hierarchical Structure DEVS Model", in: DEVS'06, DEVS Integrative M&S Symposium, Part of the 2006 SCS Spring Simulation Multiconference, SpringSim'06, Huntsville, Alabama, USA, 2-7 April 2006.
[26]Fujimoto R. M. 2000 *Parallel and Distributed Simulation System*, Wiley Interscience, NY.

Gregory Zacharewicz is an Associate Professor in Bordeaux 1 University (IUT MP). His research interests include DEVS, G-DEVS, Distributed Simulation, HLA, and Workflow. He recently focused on Enterprise Modelling and Interoperability.
www.laps.u-bordeaux1.fr/

Claudia Frydman is a full Professor in Paul Cézanne University of Marseille. She has been active for many years in knowledge management and currently her research is focusing especially on researches on knowledge based simulation.
http://www.lsis.org/~claudia_frydman.html

Norbert Giambiasi is a full Professor in Paul Cézanne University of Marseille and Director of LSIS (Laboratory of Information Sciences and Systems). He has been active for many years in simulation and currently his research is focusing especially on DEVS and relative developments.
http://www.lsis.org/~norbert_giambiasi.html

Pre Print