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

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. We then introduce a new distributed Workflow Reference Model with HLA-compliant Workflow components. We detail the HLA objects shared between Workflow federates and 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

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). However, the drawback is that there is no clear simulation semantics associated to these Workflow engines. Almost all of these engines are ad hoc. This fact may lead to errors that are difficult to detect.

Discrete Event System Specification (DEVS), Statecharts and 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. We therefore propose a set of rules (grouped in the form of an algorithm) that automatically transforms 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 using an industrial application.

SIMULATION, Vol. 84, Issue 5, May 2008 197–213

DOI: 10.1177/0037549708092833
Figures 1, 4–18 appear in color online: http://sim.sagepub.com

2. Recalls

2.1 Workflow

According to [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.

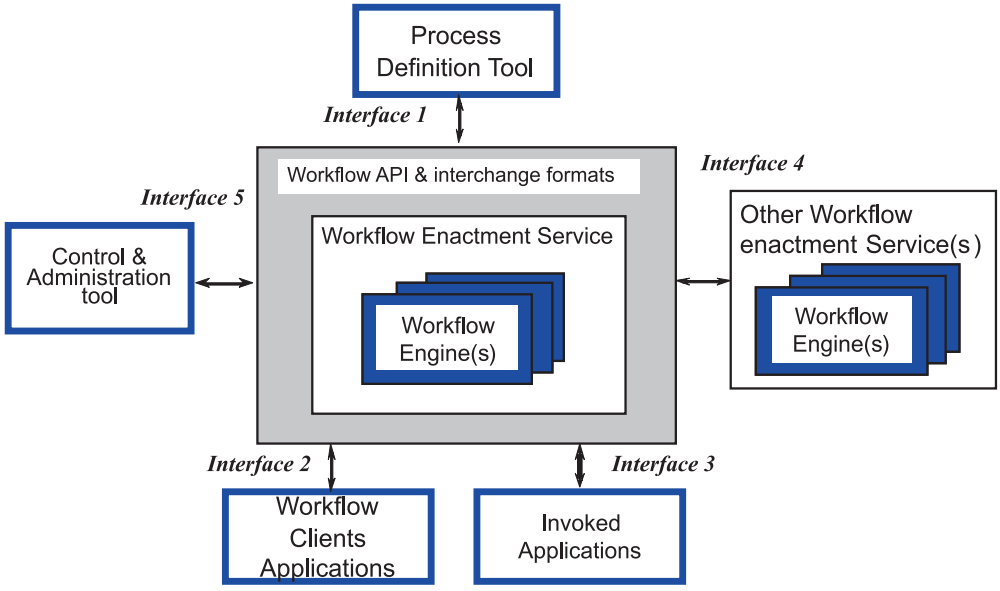

Figure 1. Workflow reference model: components and interfaces

The WfMC develops standards in the field of Workflow [2, 3]. In particular, it defines an architectural representation of a workflow management system, identifying the most important system interfaces, mostly adopted in the Workflow management field (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.

A Workflow consists of procedures also named 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 by rectangles and controllers by 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 professional tools (e.g. Staffware [4] 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 modeling and simulation theory. For example, this theory separates the modeling phase from the simulation phase allowing the reuse of the validated specifications to other applications.

In Yasper [7] and Yawl [8], developed respectively by the University of TU/Eindhoven and the University of Queensland, 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 for using Petri nets because of formal semantics despite the graphical nature, state-based instead of event-based and an abundance of analysis techniques.

We believe that a simulation tool based on DEVS formalism [9] can enhance the simulation and validation processes. Zeigler [9] discusses how DEVS modeling is more accurate than Petri nets modeling due to the following facts.

- 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 of the simulator.

2.2 G-DEVS

Traditional discrete event abstraction (e.g. DEVS) approximates observed input-output signals as piecewise constant trajectories. Generalized-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

	Tibco [4]	Workey [5]	Yasper [7]	Yawl [8]
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

DEVS model is a zero-order G-DEVS model (the input-output trajectories are piecewise constants).

G-DEVS possesses the concept of coupled model previously introduced [8]. 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 carried out using a hierarchical approach.

The concept of abstract simulator [8] 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 ensure the simulation of the atomic models, Coordinators which ensure the routing of messages between coupled models and the Root Coordinator which ensures the global management of the simulation. The simulation runs by exchanging specific messages (corresponding to different kinds of events) between the different processors. The specificity of the 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)

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 facilitated the creation of the standard 1.3 in 1996, which then evolved to HLA 1516 in 2000 [11].

The interface specification of HLA describes how to communicate within the federation through 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 of 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 former is persistent during the simulation and the latter is just transmitted between two federates. These objects are implemented with Extensible Markup Language (XML) format. Further details on RTI services and information distributed in HLA are presented in [11, 12].

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

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 and Root Coordinator into RTI. Thus, the 'global distributed' model (i.e. the federation) comprises federates intercommunicating. Figure 3 illustrates the decomposition into federates AB and ACD (Figure 3b) of a G-DEVS coupled model A (Figure 3a).

The G-DEVS models federates intercommunicating by publishing and subscribing to HLA interactions (Figure 4) that map the coupling relations of the global distributed coupled model. The first attribute is the type of message that qualifies, i.e. if the message is an external event or an init message. The second identifies the model which emits the message. The third model identifies precisely which

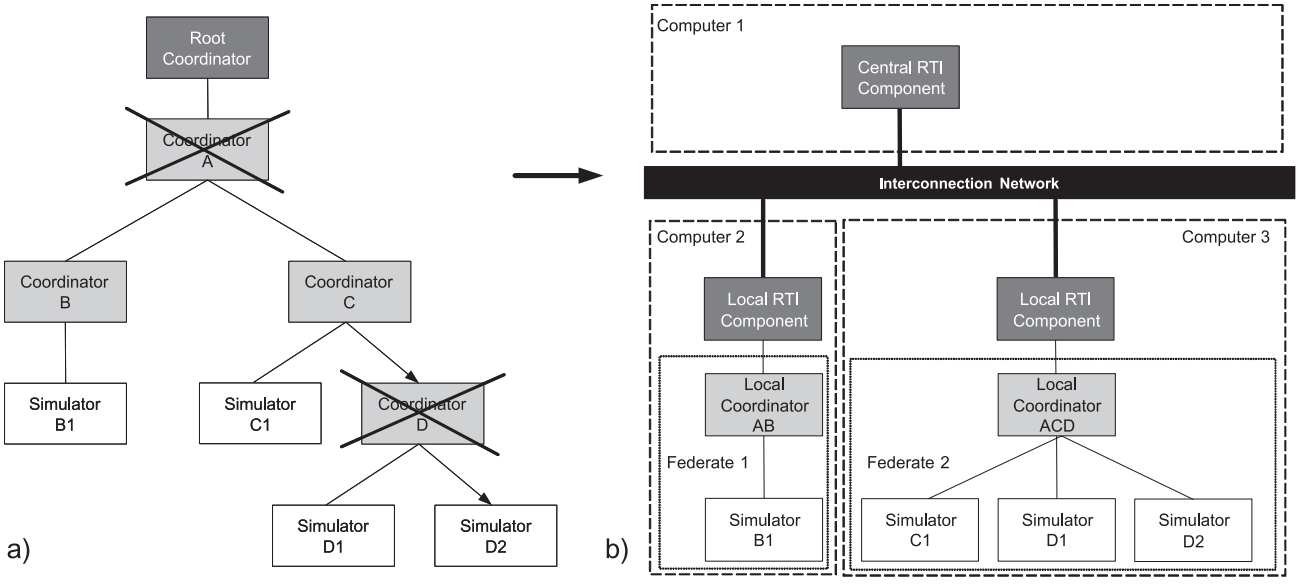

Figure 3. G-DEVS distributed simulation structure

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 4. HLA parameter table

port is concerned. The fourth defines the degree of the polynomial function in the case of G-DEVS models. The last attribute is the event values (or list of event values).

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

3. Transformation of Workflow Specifications into G-DEVS Model

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

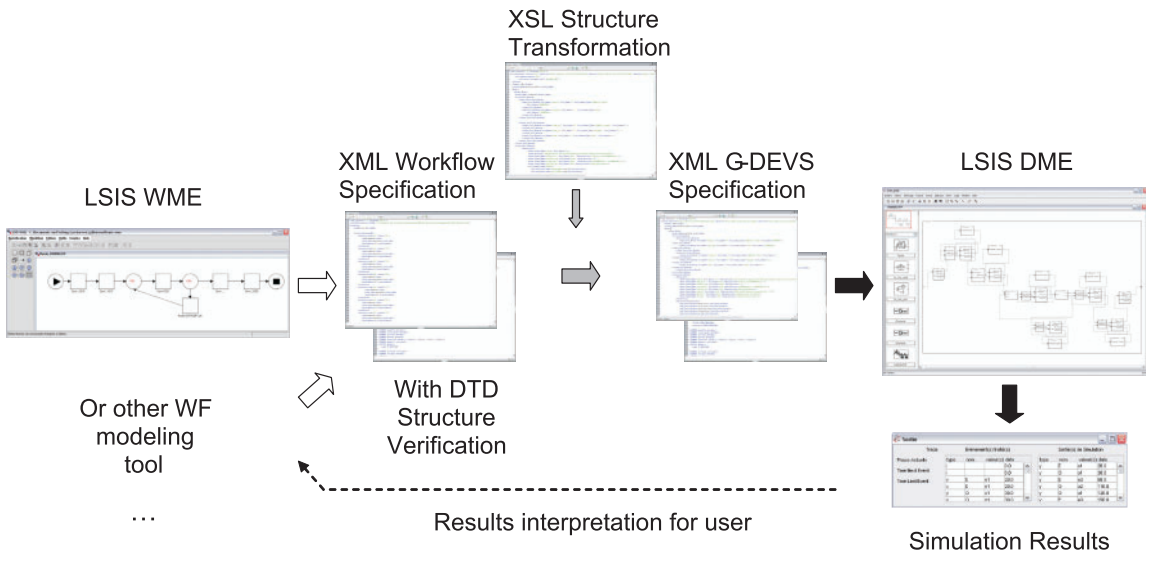

Figure 5. Workflow model to G-DEVS coupled model

the model. Furthermore, these models are often simulated by ad hoc engines that could not be compared in terms of correctness and efficiency with regards 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 standard in the 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.

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. On the other hand, some definitions of this DTD are relative to administrative Workflow, are not required for the kind of Workflow under our scope and overcast the description for non-Workflow expert users. We therefore 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, composed of the basic components <Name, RESS, A, Ct, L, IT, ST> where

- Name is the 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; and
- ST is the list of the stocks of the Workflow.

A XML Workflow process model is composed of task components that treat items and controller 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 announced in Section 2.1 and can be simulated.

We propose a general method in three steps, described in Figure 5, 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 Rousset production site.

In the following, we detail the transformation of a simple model from this industrial process. This model is depicted in Figure 6 with the graphical Workflow Model Editor developed at LSIS (LSIS_WME). It represents the high-level Workflow model of an assembly line of chips

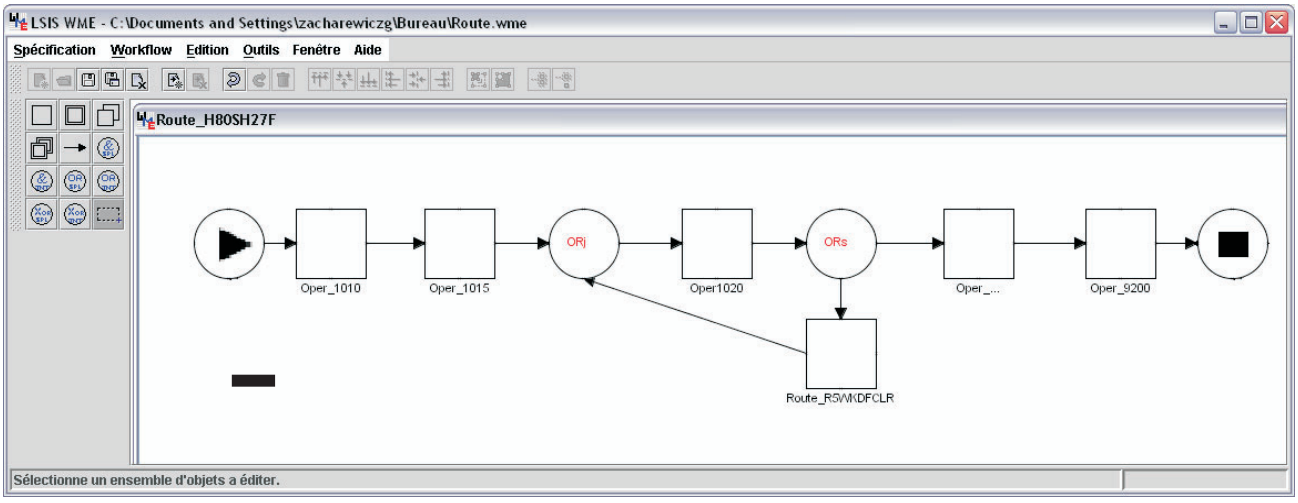

Figure 6. Route H80XX Workflow model on LSIS_WME

named Route H80XX. This model consists of an 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.

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 5, 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.

In a second step (cf. Figure 5, 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 also adds implicit information from Workflow in the XML G-DEVS. For instance, the coupling of resources on tasks (for their synchronization) is defined as G-DEVS coupling relation. Stock models are also coupled with tasks. 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.

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, depending on conditions on items.

In a third step (cf. Figure 5, 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, etc.) to instantiate atomic templates of G-DEVS models (e.g. Figure 10).

3.3 Atomic G-DEVS Model Templates

Basic components of Workflow allow instantiation by parameterization of G-DEVS atomic model templates. The atomic G-DEVS template model of task presented in Figure 10 conforms to the graphical representation in [19]. This model is initialized in a 'waiting for item to treat' state. Further to the reception of an item, the task has to make an allocation demand 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 allocation demand is made of the resource after the duration of occupation communicated by the resource.

The atomic G-DEVS model of component Stock (Figure 11) is a gueving model. It has for function to pile and to depilate items that arrive on a Task regarding to its status (busy/free).

The G-DEVS Resource model template (Figure 12) contains an initial 'Free' state. When a demand is received from a task in this state, the resource communicates its availability. It then communicates with the Task the duration of allocation and transit to a 'Resource Busy' state,

```
<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

and therefore cannot 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 has elapsed, the state of the model becomes free again and therefore available on the allocation by a task.

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. 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 block groups of the same color are identified in 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 5**,** 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.

4. G-DEVS HLA-Compliant Workflow Environment

In Section 2.4 it was seen that G-DEVS models can be run from several computers due to the capability of LSIS_DME to create HLA federates. This capability matches the distribution requirements of actual industrial processes [20]. Indeed, actual real industrial processes required the use of human decision and multiple software which interacts with the process at the different steps of its execution. These software are heterogeneous and need

```
<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"
      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

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 using LSIS_DME as the engine of a distributed Workflow environment and generalizing 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 13.

4.1 Distributed Workflow M&S Environment

We included the modeling tool LSIS_WME with 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 which 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 rerun 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 from 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 [2, 3], client and invoked applications can be called during the runtime in

```
<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];" Transition_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];" Transition_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

order to process computations not tackled by the models and their simulators.

We propose to integrate humans into the loop, to make qualitative choices during the simulation. For that purpose, we implement web interfaces called by the Workflow engine during the simulation, in order to specify e.g. some routing of items in the process. Data exchanged during the call are defined in HLA objects.

However, some complex mathematical computations are not taken into account in transition/output functions of the G-DEVS model described with LSIS_DME. In this case, the simulation is interrupted and the data are trans-

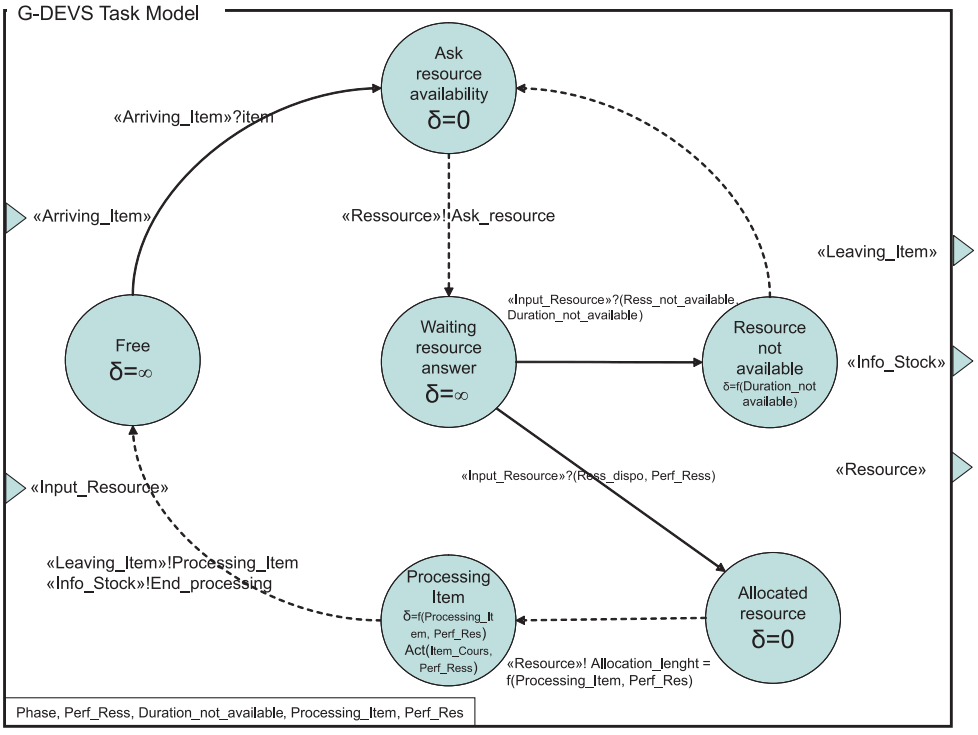

Figure 10. Task G-DEVS atomic model

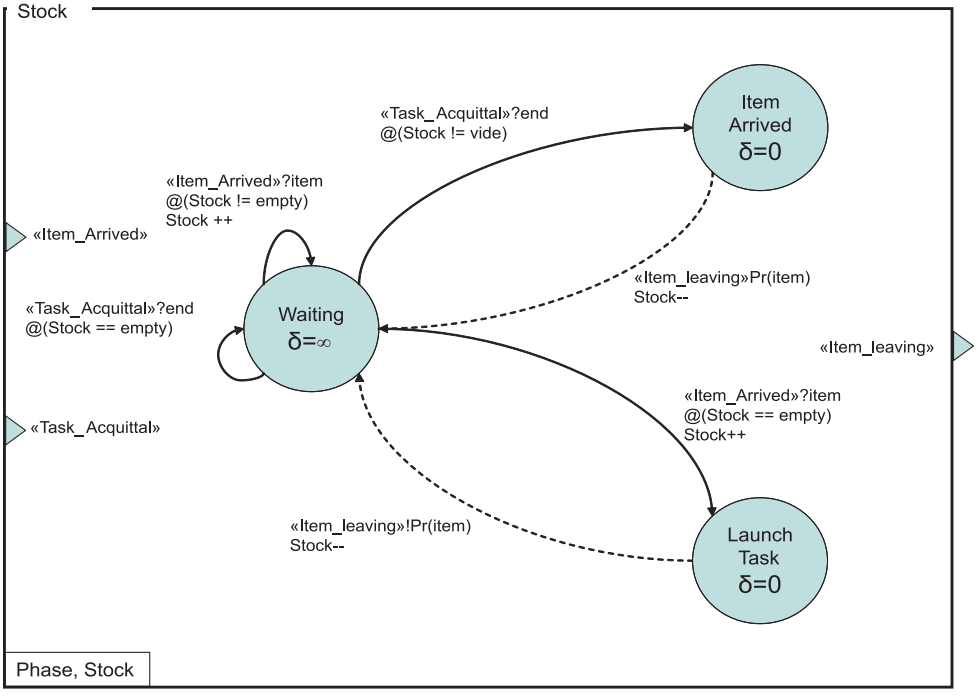

Figure 11. Stock G-DEVS atomic model

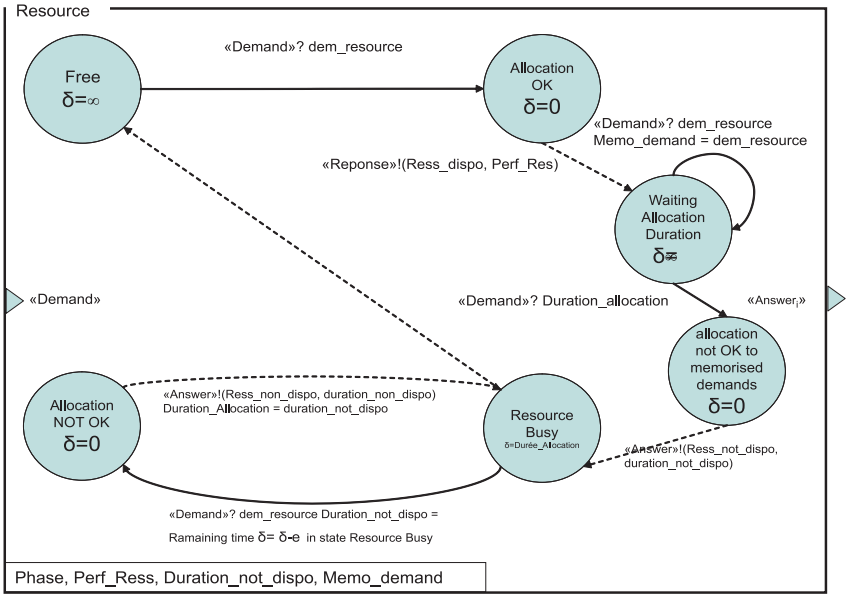

Figure 12. Resource G-DEVS atomic model

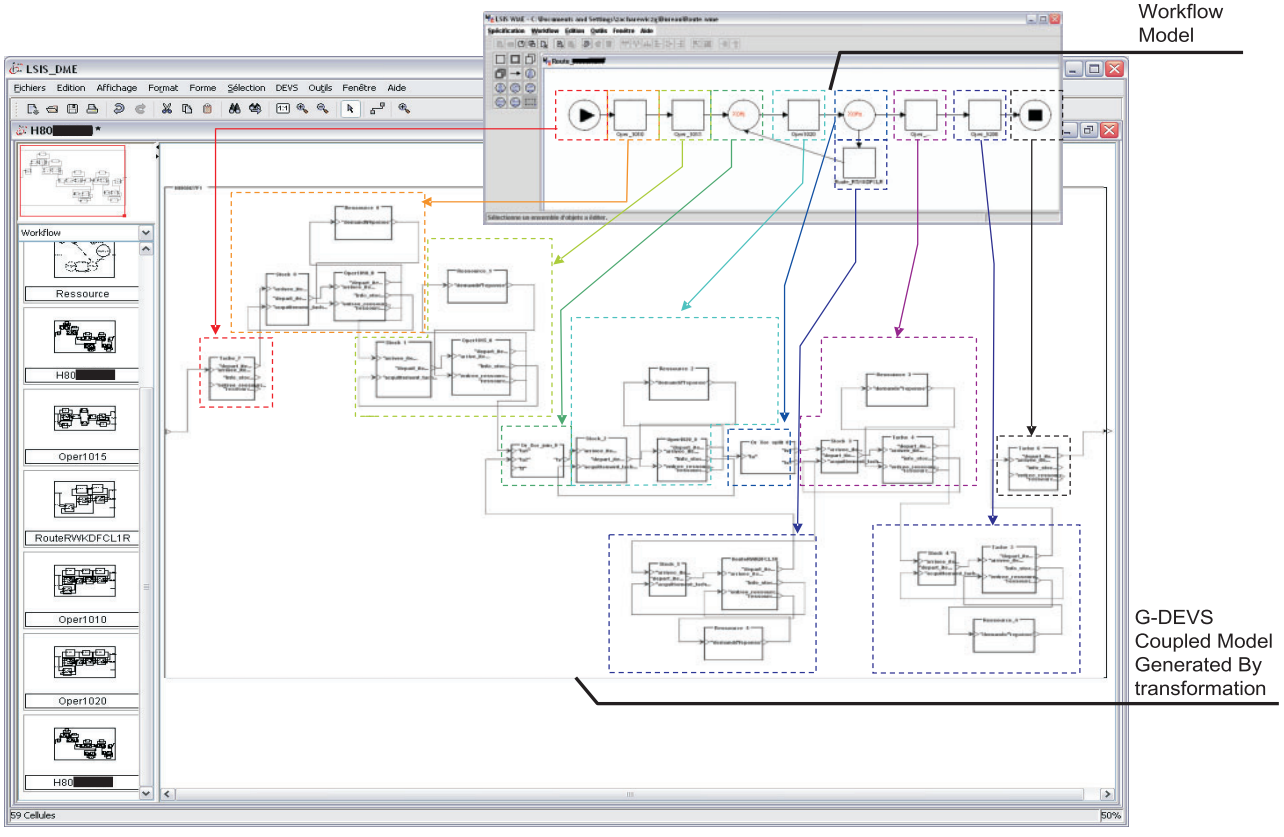

Figure 13. Route H80XX G-DEVS coupled model

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 101:
				Indication of Start / End Oper … (
				Indication of Start / End Route_RWKDFCL1R (PS)
		Interface 2 (S)	Modification of Item contents for routing (PS)	Modification of Item contents for r after 1015 (PS)
			Modification of Item contents by operation not handled by Workflow (PS)	Modification of Item contents by o not handled by Workflow (PS)
		Interface 3 (S)	Information resulting from the called aplication (PS)	Automatic Treatment of a subpart 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 RWKDFCL1
				Modification Coupling H80SH27F
			Administration Information to set Clients/Invoked Applications (PS)	Excel Parameter tunning for treatn Road RWKDFCL1R (PS)

Figure 14. HLA object table of Workflow environment federation

ferred 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 introduce the possibility of interfacing with other Workflow environments using the concept of bridges federates [21]. 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 here the data shared between the different distributed components of the Workflow.

4.2.1 Creating HLA Object Class Table

The HLA classes of shared objects specified for the Workflow federation must be specialized according to the example considered, namely in our case Route H80XX. All of these classes are depicted in Figure 14.

Interface 1 possesses three child classes. The fist and the third retrieve the information about 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.

Interface 2 possesses two sub-classes that retrieve the information resulting from a human-machine interface. In the considered route, it defines the possibility of adding information to items for routing before injecting them into the XORj controller component or by modifying the item manually in a manual operation.

Interface 3 manages the information generated by the applications involved automatically in Workflow e.g. automatic decisions made from drawing a curve of productivity and storage/retrieving of data from a database.

Interface 4 allows sharing information with another Workflow federation. To connect the Workflow, we propose to use previous work [21] which registers 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, Interface 5 allows us to consider the modifications on the process (in terms of structure and parameter setting) which can be made by a user using the tool of monitoring and management.

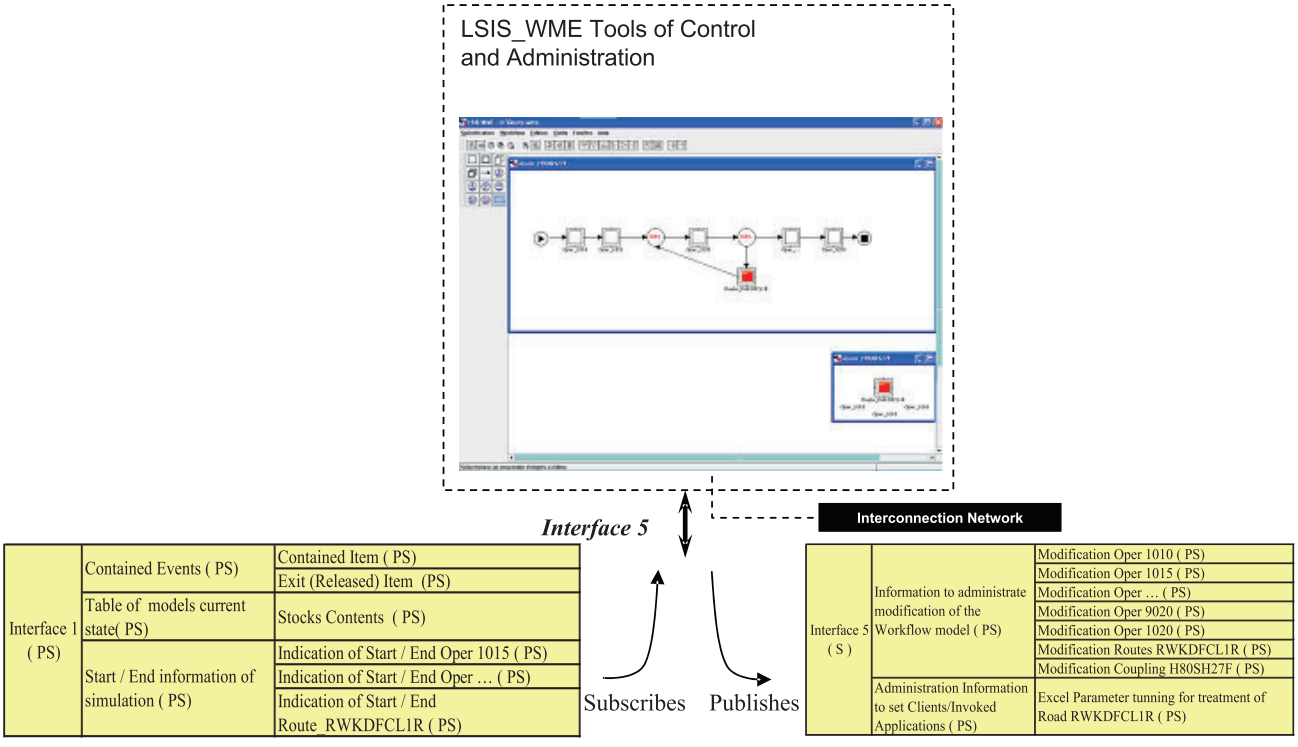

Figure 15. Interfacing with a graphical monitoring component

4.2.2 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.

4.3 Mechanism of Publication/Subscription

4.3.1 Application of Monitoring and Management

A Workflow environment has to contain a tool for monitoring and management [2]. We chose to integrate the tool for modeling 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 over diverse applications. Numerous examples are presented in the literature to illustrate this kind of application; we can refer to the application of naval Fleet modeling and distributed simulation of the Italian modeling group SIREN [22].

The federate for monitoring and managing is a dynamic ally actualized displayable version of LSIS_WME, this last subscribes to information supplied by the federate that run the simulation execution to display a graphical status of the Workflow. On its side, it will ensure 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 15.

4.3.2 Invoked Applications and Client Applications

A Workflow environment has to be interfaced with applications and tools which will be called to achieve a part of the process not automated in the model (i.e. [2]). A user can also treat data with a graphical interface during the simulation e.g. to make choices of paths in the workflow process 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 (too complex to be integrated in the model because of a black box behavior) can be applied to process data automatically. In our case, productivity measurements are processed and analyzed and

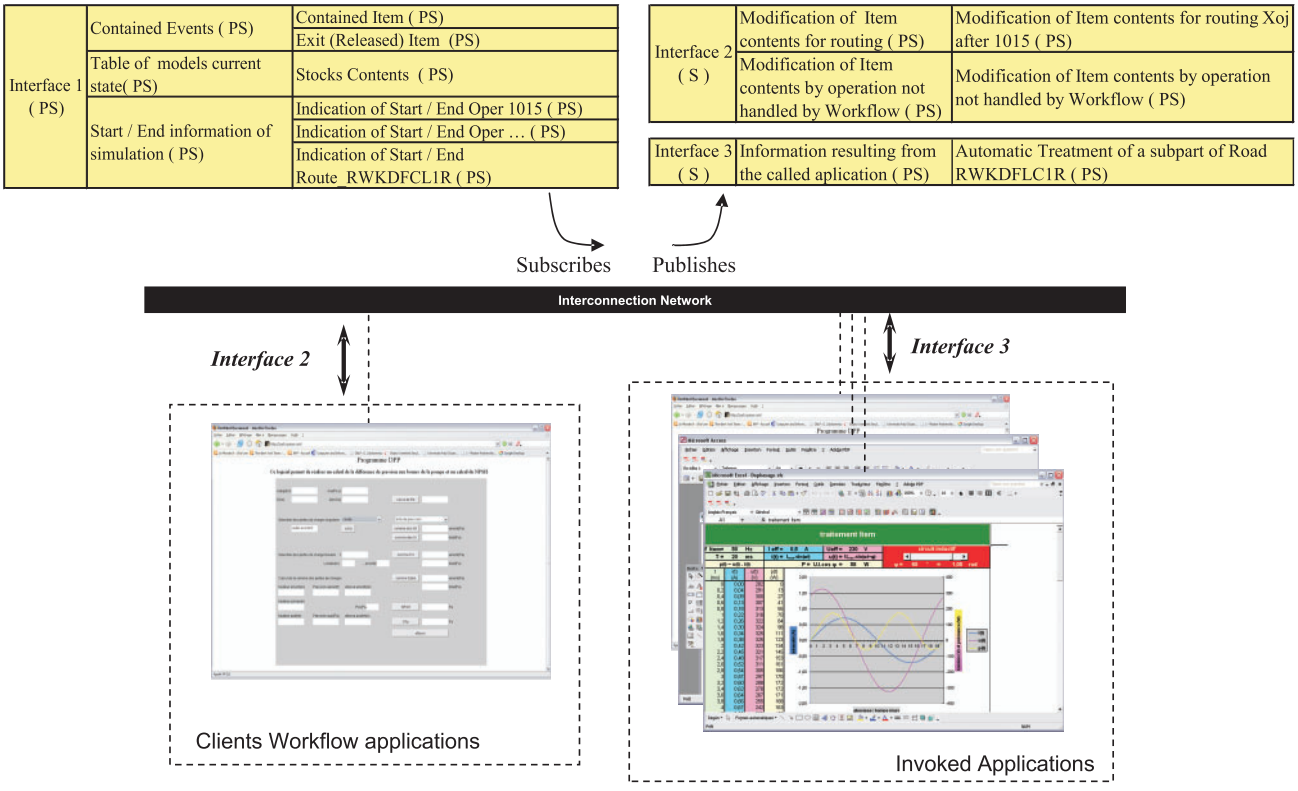

Figure 16. Interfacing with invoked applications and clients applications

data stored in a database by external specific applications that exchange data with the environment through Interface 3, as depicted in Figure 16.

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

4.3.3 Communication with other Workflow Environments

In the specification of this distributed environment, we also consider the possibility of having connections with others Workflow environments through bridges federate. This solution is based on communications between federations [20]. The solution is considered but not yet implemented in the environment.

5. M&S of Route H80XX Case

The Route H80XX (presented in Figure 6) was modeled in the environment LSIS_DME. We created several sets of events planned in the input of the process and several parameter 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 parameter settings are not related to 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 the output of certain operations.

5.1 Simulation Results for Route H80XX

We present in Figure 17 some simulation results of 4 sets of 100 items FDXX planned in the input scheduler of the model H80XX with a 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 successfully processed and generated events (items) of the coupled model G-DEVS of Route H80XX.

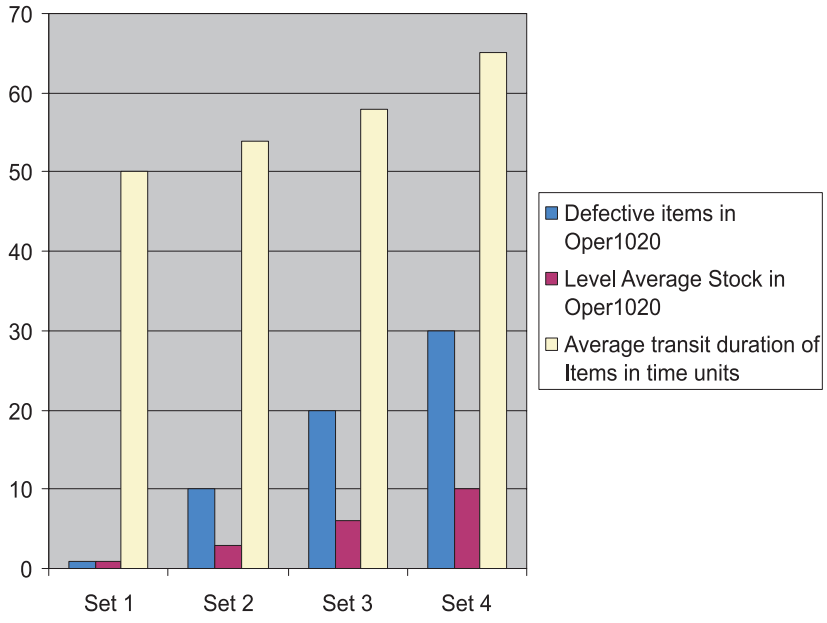

Figure 17. Simulation results of coupled G-DEVS Route H80XX with LSIS_DME

Despite the simplicity of this Workflow, this information allows the determination of 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 an important mean-level, in this case of a number of defective items amounting to 30%.

From this observation of process, using this simple log of output events and assuming 30% defective products, we suggest a modification of the Workflow process. In concrete terms, we propose to parallelize Oper 1020 into Oper1 1020 and Oper2 1020. We obtain a quasi-identical Route to Route H80XX, with the exception that the ORj controller is now connected to a new Ors controller that splits items regarding availability of Oper1 1020 and Oper2 1020. In 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 duration to approximately 56 units of time, within 30% defective items of 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. 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 in Figure 18. Indeed, the simulation results have supplied us with rational information, notably on the duration of items transit, the levels of stocks during runtime and the bottlenecks which can occur on the production chain. We have also measured the runtime performance of the simulation and note that this Workflow environment employs HLA for an interoperability purpose rather than for the runtime performance obtained.

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

6. Conclusion

We have presented a new transformation algorithm from Workflow XML specification to G-DEVS model. The use of the G-DEVS formalism to represent Workflow allows us to bring safety to a formal specification. In addition, we proposed a new HLA-compliant Workflow environment using G-DEVS. Use of the HLA specification has facilitated the connection of new HLA-compliant components in the Workflow environment.

The development of the proposed environment has been validated by tests; however, it still requires the addition of some improvements that are under development. The first improvement will be the integration of other

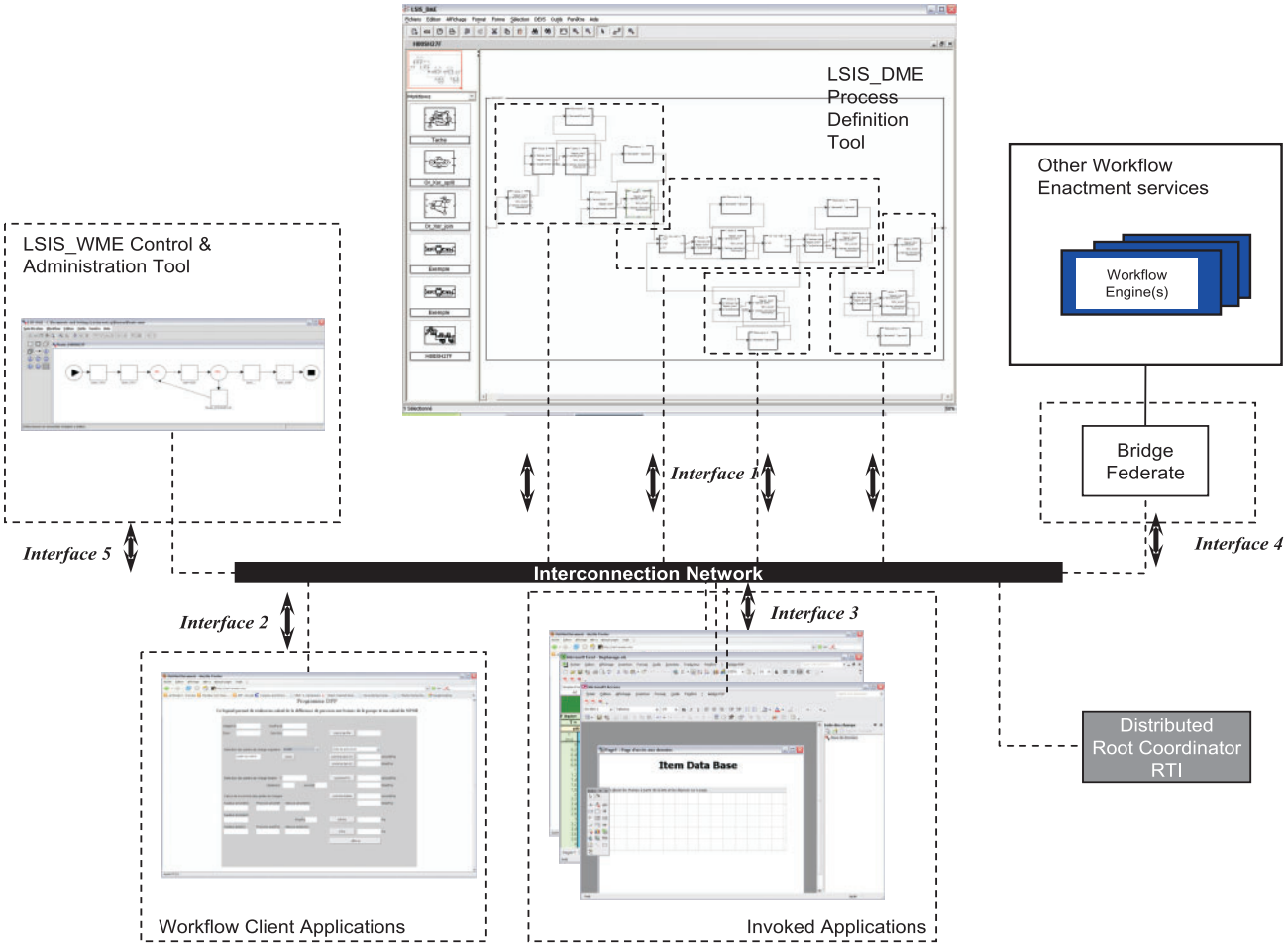

Figure 18. Workflow G-DEVS/HLA reference model

client and invoked applications into the Workflow environment, by converting them to HLA federates. Furthermore, Workflow is built by collecting information resulting from domain experts, who often define temporal information using mean values of task duration. It would therefore be interesting to envisage a modeling based on Min-Max DEVS [23, 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 [25]. Finally, we envisage using the environment as a plug-on real process of Workflow to be able to manage online the data flow that follows the flow of real materials. The application would, in that case, be 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].

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

7. Acknowledgements

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.

8. References

[1] Courtois T. 1996. Workflow: la gestion globale des processus. *Logiciels & Systèmes* 11, 46–50.
[2] Workflow Management Coalition. 1999. Terminology and Glossary. WfMC-TC-1011, 3.0.

[3] Hollingsworth D. 1995. The Workflow Reference Model. Workflow Management Coalition Spec., WfMC-TC-1003.
[4] Staffware. 2001. Staffware, Palo Alto, California. http://www.staffware.com, http://www.tibco/com.
[5] C-Log. 2006. Workey, C-Log International. 480, Place de la Courtine 93194 Noisy le Grand. http://www.c-log.com/040_Workey.
[6] Van der Aalst, W.M.P. and K.M. van Hee. 2002. *Workflow Management: Models, Methods, and Systems*. MIT press: Cambridge, MA.
[7] Van Hee K., R. Post and L. Somers. 2005. Yet Another Smart Process EditoR. In *ESM'2005, The 2005 European Simulation and Modelling Conference*, Porto, Portugal, 24–26 October 2005.
[8] Yawl. 2005. Yawl. http://www.yawl-system.com/.
[9] Zeigler, B.P., H. Praehofer and 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. *Transactions 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.
[12] Defense Modeling and Simulation Office (DMSO), US Dept of Defense. 1998. *High Level Architecture Rules* Version 1.3. IEEE P1516.2, Standard for M&S.
[13] Fujimoto, R.M. 1998. Time Management in the High Level Architecture. *Transactions of the SCS, Simulation* 71(6), 388–400.
[14] Zacharewicz, G., N. Giambiasi, C. Frydman. 2006. Lookahead Computation in G-DEVS/HLA Environment. *Simulation News Europe Journal (SNE) special issue 1 'Parallel and Distributed Simulation Methods and Environments'* 16(2), 15–24.
[15] Zacharewicz, G., N. Giambiasi, C. Frydman. 2005. Improving the Lookahead Computation in G-DEVS/HLA Environment. In *Proceedings of 9th IEEE International Symposium on Distributed Simulation and Real-Time Applications (IEEE DS-RT 2005)*, Montreal, Canada, 10–12 October 2005: IEEE; pp. 273–282.
[16] Zeigler B.P., G. Ball, H.J. Cho and J.S. Lee. 1999. Implementation of the DEVS formalism over the HLA/RTI: Problems and solutions. In *Proceedings of Simulation Interoperation Workshop (SIW)*, Orlando, FL: number 99S-SIW-065.
[17] Lake T., B.P. Zeigler, H.S. Sarjoughian and J. Nutaro. 2000. DEVS Simulation and HLA Lookahead. In *Proceedings of Simulation Interoperability Workshop (SIW)*, Orlando, FL: number 00S-SIW-160.
[18] Workflow Management Coalition. 2005. *Workflow Process Definition Interface: XML Process Definition Language (XPDL).* WFMC-TC-1025.
[19] Song, H.S. and T.G. Kim. 1994. The DEVS framework for discrete event systems control. In *Proceedings of 5th Annual Conference on AI, Simulation and Planning in High Autonomous Systems*, Gainesville, FL: pp. 228–234.
[20] Zacharewicz G., C. Frydman and C. Zanni. 2002. Workflow/HLA Distributed Simulation Environment. In *Proceedings of Harbour, Maritime & Multimodal Logistics M&S*, Bergeggi, Italy, Oct 3–5.
[21] Bréholée, B. and P. Siron. 2003. Design and Implementation of a HLA Inter-federation Bridge. In *Proceedings of the EUROSIW*, Stockholm, Sweden, June 16–19.
[22] Revetria, R. 2003. NAVI: Learning HLA from an Interactive Exercise Tutorial. *In Proceedings of Summer Computer Simulation Conference 2003*, Montreal, Canada.
[23] Giambiasi, N. and S. Ghosh. 2001. Min-Max Devs: A New Formalism for the Specification of Discrete Event Models With Min-Max Delays. In *Proceedings of European Simulation Symposium*, Marseille, France, Oct 18–20: pp. 616–621.
[24] Hamri, M.A., N. Giambiasi, C. Frydman. 2006. Min-Max DEVS Modeling and Simulation. *Simulation Modelling Practice and Theory* 14(7), 909–929.
[25] Baati, L., C. Frydman, N. Giambiasi. 2006. Simulation Semantics For Dynamic Hierarchical Structure DEVS Model. In *DEVS'06, DEVS Integrative M&S Symposium*, Huntsville, Alabama, April 2–7.
[26] Fujimoto, R.M. 2000. *Parallel and Distributed Simulation System*. Wiley Interscience: New York.

Gregory Zacharewicz *is an Associate Professor at Bordeaux 1 University (IUT MP). His research interests include DEVS, G-DEVS, distributed simulation, HLA and Workflow. He has recently been focused on Enterprise Modeling and Interoperability.*

Claudia Frydman *is a full Professor in Paul Cézanne University of Marseille. She has been active for many years in knowledge management and her research interests include knowledge-based simulation.*

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 focused on DEVS and relative developments.*