Machine and Business Modeling and Simulation for Workflow Integration

IOAN SALOMIE, TUDOR CIOARA, IONUT ANGHEL,
MIHAELA DINSOREANU, TUDOR IOAN SALOMIE
Department of Computer Science
Technical University of Cluj-Napoca
15 C.Daicoviciu street, Cluj-Napoca, 400020
ROMANIA

Abstract: This paper addresses the problems of modeling and simulating complex industrial production lines. The paper proposes a methodology for building business models, organized on layers of increasing complexity, starting from production line elementary machines and sensors up to complex business workflows. The resulted models could be translated and executed by any workflow execution engines. For workflow testing purposes, a simulation framework for production line elementary machines is proposed. The machine simulator is based on probabilistic state machines, uses standards for describing business machine functionality and is exposed as a Web Service. The proposed methodology and simulator were used for modeling and simulating a meat processing line developed in the context of the Food Trace project.

Key-Words: Workflow, Simulation, Business process, State machine, Machine modeling, Web service

1 Introduction

Business services recently became the fundamental element of large scale industrial systems, therefore developing a significant market. Business processes are represented as workflows and should be able to collaborate and to be executed. Currently, a number of methodologies and specific languages for defining process interactions and collaborations were developed. Business processes (BP) are defined following business domain rules and can be classified in two types: internal BP and external BP. Internal BP are based on company specific information, modeling complex company-specific processes such as industrial workflows. External BP usually implies collaboration of specific partners (organizations) from a business domain, based on specific rules. Working directly with internal or external business processes, before testing them, usually can cause errors which may lead to improper operation of the industrial system.

In the business services domain, activities like simulation and online monitoring play a very important role. The use of process simulation leads to detecting errors in the process design such as structural errors due to improper workflow and uncertainty errors deriving from business process representation. After successful detection of these errors we can use BP reengineering to remodel and correct the BP.

For processes simulation, a model that reproduces the real situation has to be created. Models can only be used for simulation if they precisely follow the original system. Modeling real systems is not always a simple task because they are usually too complex to be accurately described with mathematic models.

In the context of business process modeling and execution, many simulation approaches have been proposed. In [1] the authors present a process simulation system based on interactive events. The system simulates the interactions between composite services and uses a Service Oriented Architecture. The proposed system architecture is suited for the loosely coupled service computing environments and is based on an extension of the XPDL [2] meta-model. The architecture incorporates interactive event flows between individual workflows, explicitly modeled at design time, while the event interactions with data correlation are implemented at run time.

Another approach is SQMA (Situation-based Qualitative Modeling and Analysis) model described in [3]. The SQMA authors a model for representing and simulating industrial systems using Rough Set Intervals. The proposed model uses interval-based representation for qualitative models for implementing the behavior of real systems. The SQMA model hierarchically structures the whole system and decomposes system's levels into components. After that, component variables are modeled using intervals and characteristic values represented as a one-value interval. Physical rules that are used for the model verification are formulated using interval arithmetic to complete the description of each component. Using Rough Intervals and physical rules, a transition matrix between components is constructed and used in simulation. The main disadvantages of this

approach consist of inaccurate representation of machines business logic and difficulties in model management. Another disadvantage is that using Rough Set Intervals it is difficult to model complex business scenarios involving more cooperating machines.

In [4], a framework for the simulation of business process workflow models is discussed. The approach uses BPEL language for modeling the workflow which is transformed in a dataflow network model. A model checker for verifying the correctness of the workflow properties is used. Using graph theory and Petri Nets, the authors have developed a framework for fault simulation by inspecting the dataflow model.

Our approach on modeling and simulation of business services is presented in the context of the FOOD-TRACE research project [5]. The FOOD-TRACE project aims to study and design an integrated IT system for food industry processing organizations, in response to the EU requirements regarding food traceability and quality assurance. The system models the production lines using internal business processes.

The objective of this paper is to develop business modeling and simulation methods for workflow integration and testing. This objective was achieved by: (i) defining a methodology for the construction of workflow models which follow specific business rules; (ii) presenting a method for physical machines modeling using probabilistic state machines; (iii) designing and developing a simulator as a web service which uses the state machine model to simulate the execution of workflow models.

The rest of the paper is organized as follows. In section 2, we present the layered architecture used for workflow modeling and execution. In section 3, we present the physical machine modeling through state machines and the simulator design as a web service. To illustrate the simulator's functionality, a scenario is presented in section 4. Section 5 gives conclusions and promising future work.

2 A Layered Approach to Business Process Modeling

One of the best ways to present high-level business collaborations among different heterogeneous and autonomous business processes is by using workflows. Mapping real processes onto workflows is an open research problem. Usually, this mapping is done in two steps:

- the real processes are divided into simple processes having basic functionality;
- the simple processes are represented as web services interconnected by a workflow model.

There are several ways of representing workflows. The main idea is to move business process modeling closer to the user knowledge. Currently, two approaches are used for describing business processes and their internal collaboration and execution. The first one involves using a visual modeling language that generates an intermediary representation (for example BPMN [6]) which is then converted into an executable language such as BPEL [7]. The second approach describes the processes directly in BPEL.

We have identified the following requirements that should be addressed during workflow model design:

1. The need to abstract the business process concept by eliminating workflow model irrelevant details;
2. The need to represent real processes into workflow activities including traceability features. The model should allow both upstream and downstream traceability. Upstream traceability starts from row materials and concludes to the final product. Downstream traceability takes the product and decomposes it into sub-products and traces them down until the row materials.
3. The need to associate web services to workflow activities.

The resulted workflow can be executed by different BPEL Servers such as Oracle BPEL [8], Microsoft BizTalk [9] or IBM Web Sphere [10].

Our approach uses BPEL and Microsoft BizTalk Server for process modeling and workflow representation and execution. Although BizTalk Server is a friendly environment for designing organization specific workflows, there are some problems that arise from mapping the workflow to BPEL. An important problem when using BizTalk Server is that not all the elements used to model the workflow can be converted into BPEL elements thus leading to incomplete workflow-BPEL mapping. For example, BizTalk Server workflow element Transform, that associates two complex messages, doesn't have a BPEL correspondent.

Following the identified requirements we propose a layered architecture (Figure 1) based on service orchestration [11] in which the services communicate only with simple messages. The advantages of using this architecture consist in (i) reusing organization specific services, (ii) allowing modeling of a wide range of business domains and (iii) eliminating the incomplete workflow-BPEL mapping.

We have used an incremental methodology for layer construction. We start from an initial layer that contains physical or simulated machines of the production line on which simple services from the ARR layer are mapped. The rest of the layers are incrementally generated, each increment generating a new layer. A new layer is created if both of the following two conditions hold:

- at least two processes could be identified on top of the existing layers;
- there is at least one specific business rule that leads to the interaction of the processes identified on the topmost layer.

Business rules are derived from business domain or company specific standards, policies and rules. Using the set of business rules and process orchestration, new business processes can be obtained. The layer construction methodology formalism is given below:

(1) (L_n is created) $\Leftrightarrow$ ($\exists$ M={ P_1 , P_2…, P_k | k>1}) and (($\exists$ N$\subset$M, N,M$\in L_{n-1}$) or (N,M $\in L_{n-1} \cup L_{n-2} \ldots \cup L_1$)) and (||N||$\geq$2) and ($\exists$ BR | ORCS(N,BR)-> P$\in L_n$) and ($L_1 \equiv$ ARR)

where P_i are the L_{n-1} level process and ORCS(N,BR) represents the orchestration of L_{n-1} processes into a process P on L_n level based on specific business rule.

In the context of the FOOD-TRACE project, following the layer construction methodology, we have identified four layers and we have developed the system model workflow which follows the food industry business rules. The four specific architectural layers are described below (see Figure 1).

The ARR (Atomic Request/Reply) layer, specifies the atomic services that use a request/reply message exchange pattern. These services interact with the physical level (real or simulated sensors or simple machines), such as those responsible for acquiring the production line parameters (temperature, humidity, etc.).

The SP (Simple Processes) layer is on top of the ARR layer. This layer contains simple processes that are obtained by composing or orchestrating the atomic processes from the ARR layer using specific business rules. A process is part of the SP layer if the following holds:

(2) (P$\in$SP) $\Leftrightarrow$ ($\exists$ M={P_1, P_2…, P_k | k>1}M$\in$ARR), (||M||$\geq$2) and ($\exists$ BR | ORCS(M,BR)->P $\in$SP)

where ORCS(M,BR) represents the orchestration of the set M of ARR layer processes into a process P on SP layer based on business rule BR.
Processes which correspond to a single machine from the product line are included in this layer. For example, the process of "meat cutting" corresponds to the meat cutting machine. According to the business rules, the "meat cutting" process orchestrates temperature acquisition and machine starting from the ARR layer.

The CP (Complex Process) layer, defines complex processes that involve a set of machines working together for achieving a complex task. CP process definition is given below:

(3) (P$\in$CP) $\Leftrightarrow$ ($\exists$ M={P_1, P_2…, P_k | k>1} M$\in$SP) or (M$\in$SP$\cup$ARR), (||M||$\geq$2)) and ($\exists$ BR | ORCS(M,BR) ->P$\in$CP)

where ORCS(M,BR) represents the orchestration of the set M of SP and ARR layer processes into a process P on CP layer based on a specific business rule BR.
For example, in the FOOD-TRACE project, consider the process of mixing the meat with ingredients. This is a complex process, which is executed by two machines: the "add-ingredients" and the "mixing" machine.

The WF (Workflow) layer, is the topmost layer representing the workflow which models a specific product line. A workflow W is defined as follows:

(4) (W$\in$WF) $\Leftrightarrow$ (($\exists$ M={P_1, P_2…, P_k | k>1}M$\in$CP) or (M$\in$CP$\cup$SP$\cup$ ARR), (||M||$\geq$2)) and ($\exists$ BR | ORCS(M,BR) ->W)

where ORCS(M,BR) represents the orchestration of the set M of CP,SP and ARR layer processes into a process P on CP layer based on a specific business rule BR.
The results of the workflow model execution are stored in an internal repository and exposed through web services to organization business partners such as the Consumer Protection.

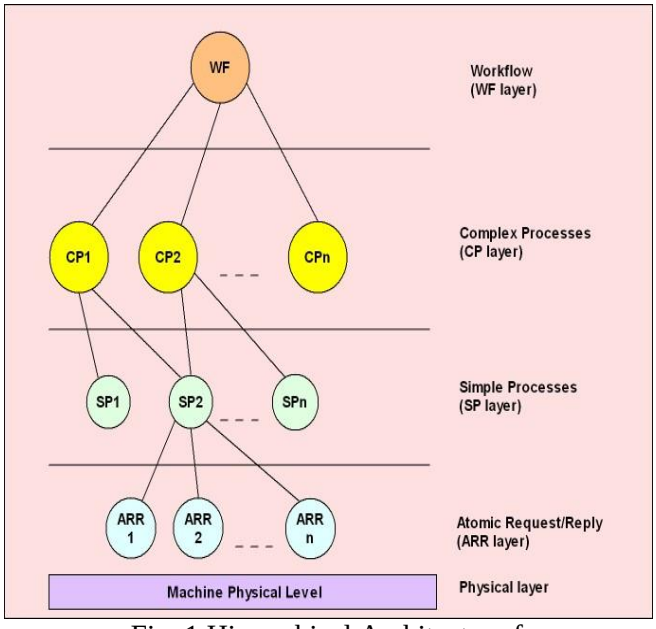

Fig. 1 Hierarchical Architecture for Workflows

3 Physical Machine Modeling

In order to simulate the execution of the proposed workflow model we have designed and implemented a simulator based on nondeterministic, probability-based, state machines. All physical machines

from a product line are modeled by state machines having an associated graph representation. The states and state-transitions of a physical machine are mapped in the state machine model onto vertexes and edges, respectively. The events that trigger state transitions of physical machines are simulated by messages passed to the state machine simulator. Each transition has an associated probability which expresses the chance of changing the current state with a next state. Also, the transitions contain two timing constants which specify how much time to wait before and after the transition. The timing constants can be used to simulate processes which need a given amount of time to complete. An example of a real machine modeled as a state machine is the meat cutting machine presented below.

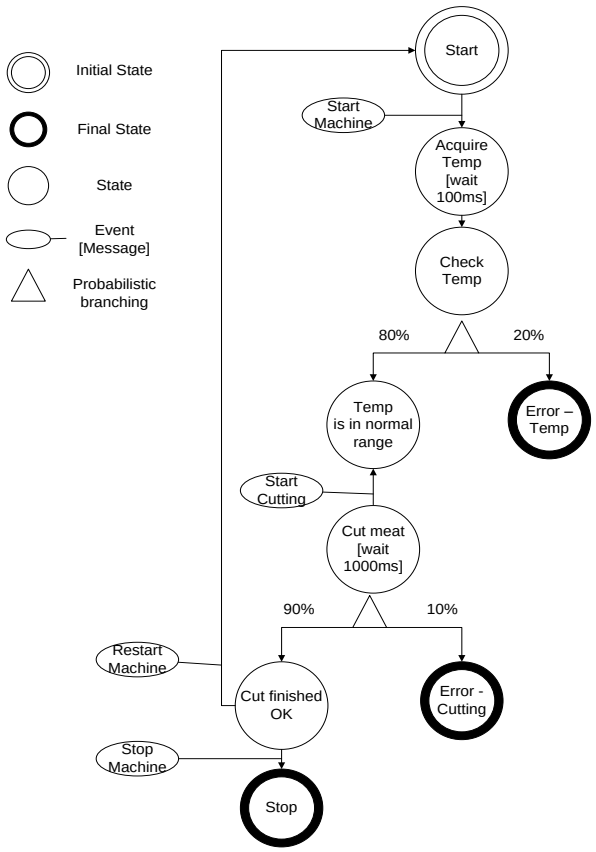

Fig. 2 Meat cutting state machine

Every physical machine, modeled as a state machine, has a corresponding XML description having the following elements:

- the root node is "stateMachine". The "initialState" parameter specifies the initial state of the state machine;
- in the messages section all the possible messages (identified by name) used for event simulation of the physical machine are defined;
- the list of states, each state being identified by an integer value. The states can be of four types: auto, message, error and final. The auto states automatically advance from the current state to the next state, depending on the contained "action" elements. The message states wait for a message to advance. The error states represent logical or physical errors the simulated machine reached while the final states mark the end of the state machine execution. The value of the state identifier is a positive integer for message and auto type states, negative for error states and zero for final states;

```
<stateMachine initialState="Start">
  <messages>
    <message name="StartMachine" />
    <message name="StartCutting" />
    <message name="RestartMachine" />
    <message name="StopMachine" />
  </messages>
  <state name="Start" type="message" id="0">
    <message name="StartMachine">
      <action nextState="AcquireTemp"/>
    </message>
  </state>
  <state name="AcquireTemp" type="auto" id="1">
    <action nextState="CheckTemp" />
  </state>
  <state name="CheckTemp" type="auto" id="2">
    <action probability="0.8" nextState="TempOK"
                                  waitBefore="100" />
    <action probability="0.2"
            nextState="ErrCheckingTemp" />
  </state>
  <state name="TempOK" type="message" id="3">
    <message name="StartCutting">
      <action nextState="Cut" />
    </message>
  </state>
  <state name="Cut" type="auto" id="4">
    <action probability="0.9" nextState="CutOK"
                                  waitBefore="1000" />
    <action probability="0.1" nextState="ErrCutting" />
  </state>
  <state name="CutOK" type="message" id="5">
    <message name="StopMachine">
      <action nextState="Stop" />
    </message>
    <message name="RestartMachine">
      <action nextState="Start" />
    </message>
  </state>
  <state name="ErrCheckingTemp" type="error" id="-1"
/>
  <state name="ErrCutting" type="error" id="-2" />
  <state name="Stop" type="final" id="0" />
```

Fig. 3 XML meat cutting state machine

- the "action" elements contain the attributes "nextState", "waitBefore" and "waitAfter". "nextState" attribute is mandatory, and represents the name of the next state if this action is chosen. The "waitBefore" and "waitAfter" attributes are optional and express the

amount of time to pause before and after executing the actions, in milliseconds (if omitted, zero is assumed);
- "probability" is a real number in the interval [0.0 .. 1.0]. It determines the probability of a specific action to be chosen. The sum of probabilities for a group (state element for auto states and message element for message states) must be 1.0. If some probabilities are omitted, the remaining probability is distributed amongst them (e.g. if we have 5 action elements and the first has a probability of 0.1, the second one of 0.3 and the rest are omitted, then the last three will each have a equal probability of 0.2).

Figure 3 presents the XML definition of the state machine which models the "meat cutting" state machine from Figure 2.

The simulated execution of the proposed model requires message passing between the workflow and the simulator. This leads to developing the simulator frontend as a web service. The web service simulator communicates with the web services that model the workflow through SOAP messages. The main simulator functionality is exposed through the following web service operations:

1. "InitializeStateMachine(stateMachineName: string)" is used to create a new session of the web service that will be used for the simulation of the state machine specified by the parameter string. No other methods can be invoked on the web service prior to successfully (true is returned) executing this method.
2. "PostMessage(message: string)" is used to post a message to the previously initialized state machine. The method returns an integer value representing the state in which the simulated machine is. The returned integer value should be checked in order to see if it is a normal state (message or auto) or an error or final state.
3. "GetStateNameForID(id: int)" returns a string representing the name for the state represented by the parameter id.
4. "ResetMachine()" is used to reset the machine to its initial state and returns a Boolean value indicating the operation success.

4 Simulation Scenarios

A workflow of a sausage preparing product line was proposed as a simulation scenario (Figure 4). The workflow model was constructed using the layered construction methodology presented in Section 2.

Based on the scenario represented in Figure 4, we have identified the following atomic request/reply processes: getTemperature, getTime, getHumidity, getOxidation, getWeight and machineStart/Stop. They are represented as web services based on the request/reply paradigm, which interacts directly with the simulated or real machines. The simple processes of the SP layer such as "meat-cutting", "mixing" or "filling" are constructed by orchestrating the atomic request/reply web services.

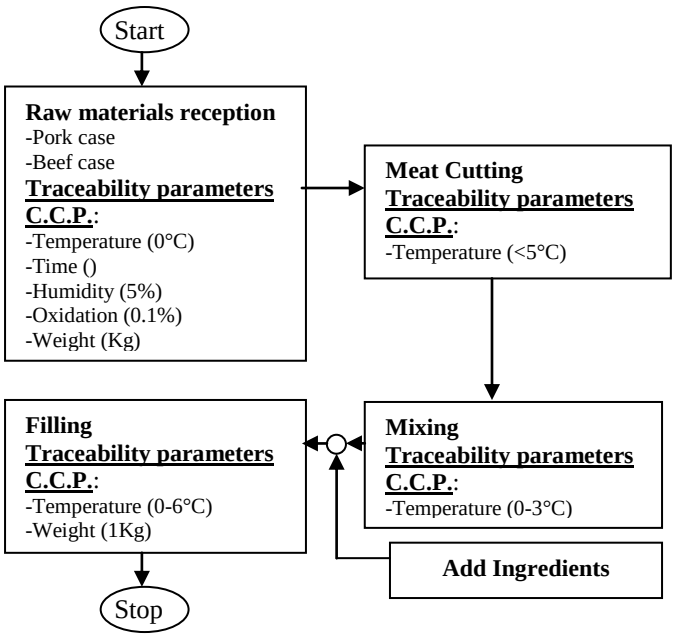

Fig. 4 Workflow scenario

Using Microsoft BizTalk Server Orchestrator, the simple services are represented as BizTalk workflows, exported as BPEL processes and saved in a database for a later use. For the complex processes level in the layered architecture, we have identified the process of "Mixing and Add-Ingredients". Next, we describe the simulation of the simple "Meat Cutting" process (see process model in figures 2 and 3). The workflow model for the "Meat Cutting" machine is designed in Microsoft BizTalk Orchestrator.

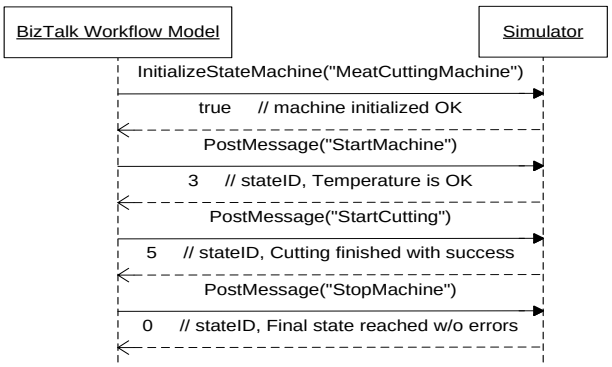

Fig. 5 "Meat Cutting Machine" Simulation Sequence Diagram

The proper simulation is achieved by message passing between the simulator web-service and the BizTalk orchestration.

The simulated execution is conducted by the BizTalk representation of the workflow model. The sequence diagram presented in Figure 5 describes a successful simulation scenario for the "Meat Cutting Machine". The workflow model for the "Meat Cutting Machine" described in BizTalk is showed in Figure 6.

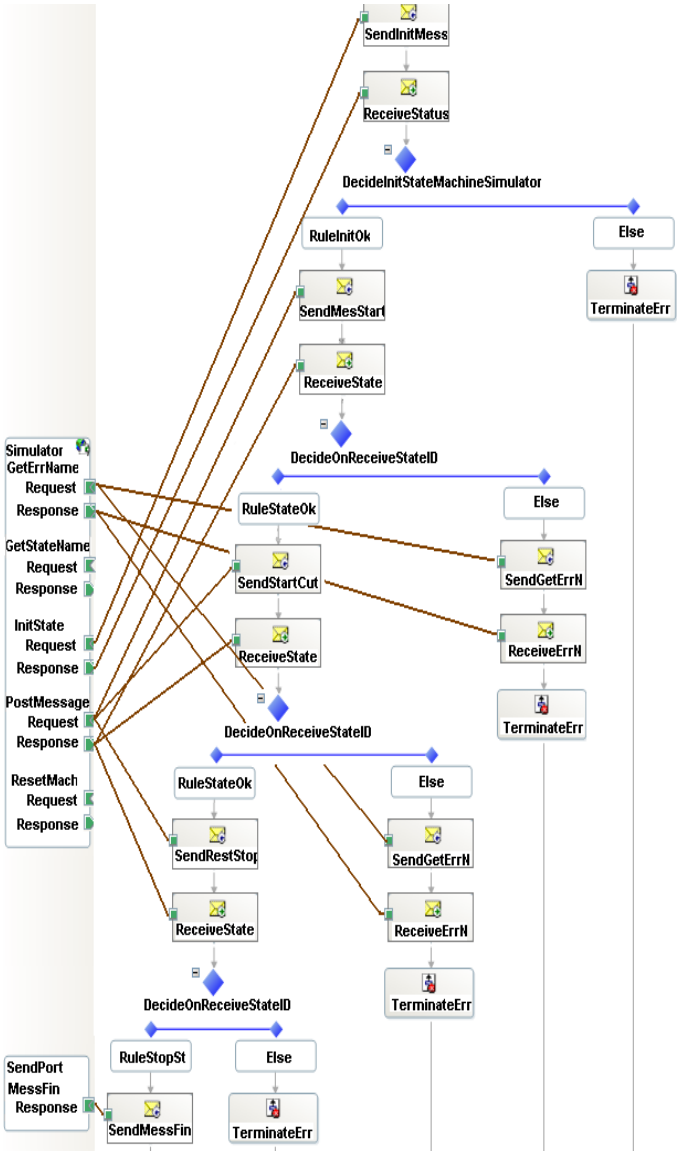

Fig. 6 BizTalk workflow model

5 Conclusion and Future Development

This paper proposes a methodology for building business models organized on layers of increasing complexity from production lines elementary machines and sensors to complex business workflows. The paper also presents a method to simulate the execution of business processes modeled as workflows. The simulation process involves the following phases: (i) the construction of workflow models which follow specific business rules using the presented methodology; (ii) modeling physical machines using probabilistic state machines model; (iii) the simulated execution of workflow models. In the context of the FOOD-TRACE research project, for modeling and testing purposes, a workflow model of a sausage preparing product line was used as a simulation scenario. For future work, we intend to improve the simulator by changing the state machine runtime with the Microsoft Workflow Foundation runtime [12]. This will allow the use of a standard state machine XML based representation. Also an inter business approach on simulation, which will consider the collaboration among different business partners, is a future enhancement of the simulator.

References:

[1] Yanchong Zheng, Yushun Fan, Wei Tan, Interactive-Event-Based Workflow Simulation in Service Oriented Computing, *Fifth International Conference on Grid and Cooperative Computing (GCC'06)*, 2006.
[2] XPDL specifications, *http://www.wfmc.org/ standards/ xpdl.htm*.
[3] M. Rebolledo, *Rough intervals—enhancing intervals for qualitative modeling of technical systems*, Artificial Intelligence 170, 667–685, 2006.
[4] Mate Kovacs, Laszlo Gonczy, *Simulation and Formal Analysis of Workflow Models*, Electronic Notes in Theoretical Computer Science, *www.elsevier.nl/ locate/entcs*.
[5] FOOD-TRACE project, *http://www.coned.utcluj.ro/ FoodTrace/*.
[6] BPMN (Business Process Modeling Notation), *http://www.bpmn.org/*.
[7] BPEL4WS Specifications, *http://www-128.ibm.com/ developerworks/library/specification/ws-bpel/*.
[8] Oracle BPEL, *www.oracle.com/technology/bpel*
[9] Microsoft Biztalk Server 2006, *http://www.micro soft.com/biztalk/default.mspx*.
[10] Web Sphere, *www.ibm.com/software/ websphere*.
[11] Service Orchestration, *www.serviceoriented.org*.
[12] Microsoft Workflow Foundation, *wf.netfx3.com*
[13] Moscato, F., Mazzocca, N., Vittorini, *Workflow Pattern Analysis in Web Services Orchestration: The BPEL4WS Example*, 1st International Conference on High Performance Computing and Communications 2005, LNCS 3726, 395-400.
[14] Li, H., Lu, Z, *Decentralized Workflow Modeling and Execution in Service-Oriented Computing Environment*, IEEE International Workshop on Service-Oriented System Engineering, 2005.