BUSINESS PROCESS SIMULATION FOR MANAGEMENT CONSULTANTS: A DEVS-BASED SIMPLIFIED BUSINESS PROCESS MODELLING LIBRARY

Igor Rust[(a)]; Deniz Cetinkaya[(b)]; Mamadou Seck[(c)]; Ivo Wenzler[(d)]

[(a)(b)(c)]Systems Engineering Group; Faculty of Technology, Policy, Management
Delft University of Technology; Jaffalaan 5, 2628BX, Delft, THE NETHERLANDS
[(d)]Accenture Nederland; Gustav Mahlerplein 90, 1082 MA, Amsterdam, THE NETHERLANDS

[(a)]i.j.rust@student.tudelft.nl, [(b)]d.cetinkaya@tudelft.nl, [(c)]m.d.seck@tudelft.nl, [(d)]ivo.wenzler@accenture.com

ABSTRACT

Business process simulation enables analysis of business processes over time and allows to experiment with scenarios (like for instance redesigning business processes) before implementing them into an organization. This research aims at providing an easy way of business process modelling and simulation for management consultants whose core competence is not simulation model development. During the design and development process, management consultants are actively involved following a user-centred design approach. The outcome of this research is a library of DEVS-based business process modelling elements implemented in Java and using the DSOL simulation library to provide the simulation capabilities.

Keywords: business process modelling, business process simulation, component based modelling, DEVS

1. INTRODUCTION

To stay competitive and to operate effectively, an organization needs to improve its process efficiency and its quality by adapting its strategy, structure, management and operations to its changing business environment. Management consultants provide expertise and recommendations to improve their clients' business performance. To support organizational decisions, a good understanding of the business processes is essential.

A business process is a series of activities that produces a product or service for a customer. Business Process Modelling (BPM) is the activity resulting in a representation of an organisation's business processes so that they may be analyzed and improved (Weske 2007).

A distinction can be made between static modelling and dynamic modelling of business processes (Bosilj-Vuksic, Ceric and Hlupic 2007). Static modelling tools often provide a graphical process representation, for example simple flowcharts, IDEF0 or BPMN diagrams to depict business processes. Business Process Simulation (BPS) tools, provides the possibility to simulate and evaluate the dynamic behaviour of business processes.

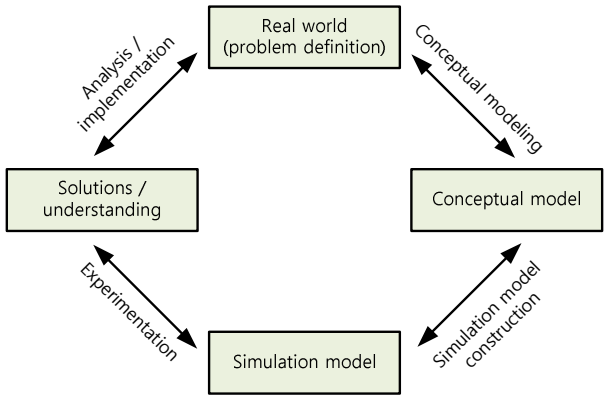

Figure 1: Simulation Project Life Cycle.

Figure 1 depicts the main phases and products of a simulation study. The organization, for which an analysis is undertaken, is part of the "real world". First, a conceptual model is developed, often in a graphical form, which contains the essential aspects of the problem situation (Banks 1998). A conceptual model helps to build credibility for and guides the development of a simulation model. Next, a simulation model of the business process is developed based on this conceptual model. The executable model can be manually programmed by the modeller, or constructed through a visual interface (Pidd 2004a).

After the development of the simulation model, experiments can be set up and the simulation model is executed by simulation software to analyse the output. A simulation software generally consists of a simulation engine (or simulation executive) and an application program (Pidd 2004b). The engine keeps track of the state changes which occur at some moment in time and reminds the application when a state change is due. How the experiment is set-up and which output parameters are of interest, depend on the business case. Based on the presented outputs, more experiments can be performed or changes may be implemented in business processes.

Although the usefulness of business process simulation was proven by many authors and various simulation tools are available, still many consultants

and business analysts rely on simple static process mapping methods (Bosilj-Vuksic et al. 2007; Melão and Pidd 2003). Some reasons for the lack of adoption are that much experience is needed to develop valid simulation models and simulation model development is time consuming and costly (Van Eijck and De Vreede 1998). More specifically, there is a lack of business process simulation tools which supports an easy and quick approach of modelling and analysis of business process by consultants and business analysts.

This paper presents a business process simulation method to support management consultants to model, simulate and analyze business processes in a well defined manner. Next section provides background information about business process consultancy which is extracted from the interviews with the consultants. Section 3 discusses the design process of our research which is based on a user-centred design approach. Section 4 and 5 present a DEVS component library for business process modelling and its usage. Finally, conclusions and future work are given in Section 6.

2. SUPPORT FOR BPS BY CONSULTANTS

2.1. The Consultants' View on Business Processes

Harington (1991) defines a business process as a group of logically related tasks that use the resources of the organization to provide defined results in support of the organization's objective. Consultants see business processes more specifically as "*a series of activities and decisions which are performed by resources and which influences the flow and state of the entities"*.

An entity (or passive entity) is an abstract object which can represent anything that undergoes activities in a business process. The entity arrives at an organization, "flows" through the business process(-es), and then leaves the organization. What the entity actually represents is called the entity type. Examples are an order that is received by a company and needs to be processed, an insurance claim or a contract cancellation e-mail. During a business process, an entity undergoes state changes as a result of the activities that are performed by resources. The state of an entity may for instance be "received", "processed" or "finished".

A resource is responsible for making decisions and performing activities on entities and is considered as the leading part of a business process. A resource can be a human or a machine. Resources are typified by their capability, role, capacity, availability and state. The capability of a resource depicts whether a resource is able or allowed to perform a certain activity. Based on for instance the experience that a resource has, the resource may be able to perform more complicated activities. The collection of all capabilities of a resource is called the resource's role. It is possible that within an organization, multiple resources have the same capabilities and thus share the same role. In that case, a role has a certain capacity: the number of resources that are available to perform a determined set of activities.

The availability of a resource depicts when or for how long a resource is available to commence activities. For instance, a full time employee has an availability of 1 FTE (Full Time Equivalent), which depicts that during a complete working day the resource is available to perform activities. The state of a resource relates to whether a resource is currently busy (or active) with performing a certain activity, or is available for new work.

An activity is a piece of work performed by one or more resources which requires a certain amount of time. An activity can be a task or a grouping of task, called a sub-process. A task is a piece of work that cannot be subdivided into smaller pieces of work to be performed by a resource, or is not crucial for the purpose of describing and analyzing a business process. There are three options how an activity can be performed, namely 1) one resource starts and finishes an activity on its own; 2) a resource hands over the entity to another resource who will perform one or more activities; or 3) the amount of work is divided over two or more resources. In the first case, the flow of an entity through activities is called sequential. In the second case, a resource will hand over the entity to another resource that will perform his activities. The third case is called a parallel activity. After the work is split up, two or more resources will perform their activities independently. At some moment the work is synchronized again and some resource will continue performing activities.

In a business process decisions are made that influence the choice and order of activities to be undertaken by a resource. A decision can depend on the attribute of an entity (e.g. entity type or state), or the state of the system (e.g. what are other resources doing, how many entities are waiting to be processed, etc.).

2.2. Conceptual Modelling

Pidd (1996) defined a model in the context of operations research and management sciences as "*an external explicit representation of part of reality as seen by the people who wish to use that model to understand, to change, to manage and to control that part of reality*". In other words, a model can be used as a representation of reality (like for instance an object, idea or an organization), to support someone who wants to understand that part of reality, and possibly wants to make decisions which will influence reality.

There exist various modelling languages that support the representation of business processes in a model in a standard and consistent way. Some examples are BPMN, Flow Chart, Gantt Chart, IDEF0, IDEF3, and UML (Aguilar-Saven 2004). Each of these languages has different characteristics (semantics, representation notation, ability to include decomposition and hierarchy of processes, etc.).

In order to support consultants with a new modelling approach, a conceptual modelling language should be chosen or developed that is able to represent the consultants view on business processes as described in the previous section. The notation should also be

understandable to enable correct interpretation by other consultants, as well as domain experts of the modelled organization.

2.3. Simulation Model Development

As mentioned in the previous section, various languages exist to represent conceptual business process models. However, most of these languages present an abstract way of thinking and don't provide the possibility to include all details needed to enable direct translation into an executable simulation model, nor direct execution of these models by a simulation engine. Due to the lack of possibilities for a conceptual modeller to include all details in a conceptual model, different simulation models can be developed based on the same conceptual model. If for instance the simulation model is developed by someone who was not part of the conceptual modelling stage, the risk increases that a final simulation model does not represent the business processes as was intended by the conceptual modeller. Cetinkaya, Verbraeck and Seck (2010) concluded in recent research that there is a large semantic gap between the conceptual modelling stage and the simulation model construction stage.

With regard to the actual development of an executable simulation model, various formalisms exist to support the formalization of simulation models, like for instance Discrete Event System Specification (DEVS) (Zeigler, Praehofer and Kim 2000) and Petri Nets (Peterson 1981). Developing an executable simulation model using one of these formalisms requires a deep understanding of the underlying concepts, as well as programming experience.

Reusing parts of simulation models or reusing even complete simulation models is suggested to decrease the complexity and time needed to develop models. Different forms of reuse are: code scavenging (reusing existing code of a simulation model), function reuse (reusing predefined functions that provide specific functionalities), model reuse (reusing a complete simulation model for a different situation) and component reuse (reusing an encapsulated simulation module with well-defined interfaces) (Pidd 2002). Usage of components for simulation modelling is considered to be fruitful concept to increase the efficiency of hierarchical model construction (Cetinkaya, Verbraeck and Seck 2010).

3. DESIGN PROCESS

When we consider business process simulation from a consultant's point of view, the three main activities that he/she is interested in performing, are: 1) developing the business process model (conceptual model) and specifying the model parameters; 2) experimenting with a simulation model; and 3) interpreting the results. How the translation of a conceptual model into an executable simulation model can be done is important, but also irrelevant to the consultant (as long as the simulation model leads to results as how the consultant intends it to do). Because our goal is to support consultants with business process modelling and simulation, we follow a user-centred design (UCD) approach.

The main goal of a UCD approach is to increase the likelihood that a designed and developed artefact is found usable by its end-users. User-centred design approach is concerned with incorporating the end-users perspective during the design and development process to achieve a usable system (Maguire 2001). Because management consultants are the end-user of our new business process modelling method, they are placed at the centre of the design process. To incorporate consultants in this research, a series of design and evaluation rounds are held (in the form of workshops) with management consultants of a large international management consultancy firm. These workshops are intended to get (among other things) an understanding of the consultant and his/her view on business processes (as it is already discussed in Section 2.1); to decide upon and evaluate an understandable modelling language of the consultants view and to evaluate the usability of the proposed modelling approach.

4. A DEVS COMPONENT LIBRARY FOR BPM

4.1. Modelling Elements

The modelling representation that was the outcome of the design research process is based on the Business Process Modelling Notation (BPMN). BPMN is an industry-wide standard for modelling of business processes and was chosen because of various reasons, like: 1) the way consultants see business processes, corresponds closely to how business processes are described in the BPMN specification; 2) many management consultants have experience with modelling of business processes through the use of "swimlane diagrams" (a method that resembles BPMN to some extent, but more simplified); 3) BPMN was used for conceptual modelling of business processes in past simulation projects and was found to be useful by the involved consultants; and 4) the BPMN is becoming a standard language to model business processes throughout industries, which increases the likelihood that clients are already familiar with the notation and the models.

A set of modelling elements were determined (see Figure 2), which allow modelling of business processes by consultants as how they actually perceive business processes (as mentioned in Section 2.1). A resource (or group of resources which share the same capabilities) is represented by a *Swimlane*, which is a graphical modelling element that can contain activities (like tasks and sub processes) and decision elements (gateways).

The arrival of entities in a business process is represented by a *Start Event*. The Start Event is placed outside a Swimlane, to depict that arriving entities are coming from outside the organization (or business unit) and no resources are busy at that arrival time. The *End Event* represents the end of a business process, namely when an entity leaves the organization, and is also placed outside a Swimlane.

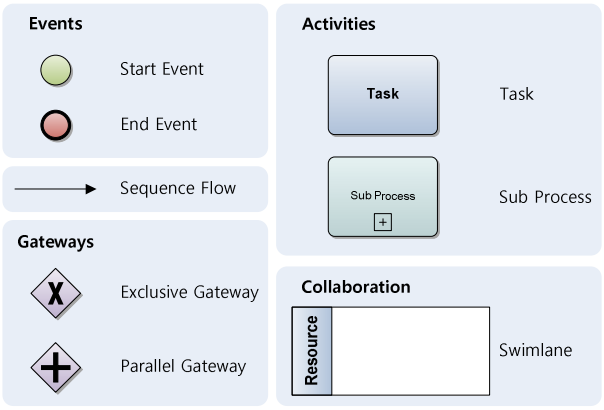

Figure 2: Overview of the BPM Elements.

After the entity enters through a Sequence Flow a Swimlane, a resource "picks up" the entity in a pre-specified manner (e.g. on a random basis, following some pattern or based on a certain priority rule) and performs one or more activities. A *Task* modelling element is an activity which represents work that is performed by a resource and consumes a certain amount of time. *Sub-process* modelling element is included to support hierarchical modelling.

Parallel Gateways are included to enable activities to be performed in parallel by multiple resources and are used in pairs: one gateway is used at the start of a parallel activity. It duplicates an entity and forwards the entities to the activities that are performed by different resources in parallel. A second Parallel Gateway is used to synchronize a parallel activity again after both activities are completed.

Exclusive Gateways provide the functionality to resemble business decisions. The flow of an Entity is directed based on the evaluation of a condition. This condition can be either the evaluation of an entity-specific attribute (e.g., entities of a specified type/state move in one specified direction, other entities move in another direction), or based on probability (e.g., 70% of the arriving entities move in one direction, the other 30% move in the other direction).

4.2. Formalization of Modelling Elements

We use DEVS to specify our simulation models. In DEVS, a system can be represented by two types of models: atomic and coupled models. Atomic DEVS models describe the behaviour of a system, whereas coupled models describe the composed structure of a system. Atomic models can be integrated into coupled models, and coupled models can be integrated in higher level coupled models. This way, a model is decomposed in a hierarchical manner.

The suggested BPM elements and some supplementary elements are matched to DEVS components. For every element a state-diagram was developed and validated. Figure 3 shows the state diagram for Task element. Since a Swimlane represents either a resource or a group of resources, a Task needs to check for waiting entities.

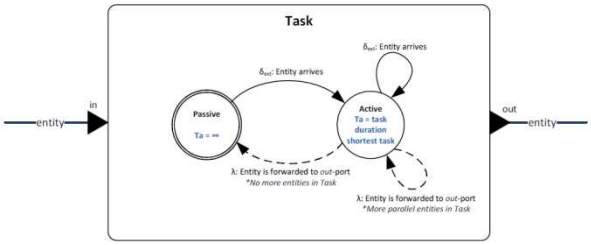

Figure 3: State Diagram for Task Element.

The supplementary components are developed to support some of the needed simulation functionality as discussed by the consultants. For instance, when an entity enters a swimlane, the entity is placed in a queue and a resource is requested. When available, the resource is attached to the entity after which the entity leaves the swimlane entry. For this purpose, the *Swimlane Entry* component was designed. When an entity leaves a swimlane, the resource occupied with the entity should be made available for other (possibly already waiting) entities. This is done by the *Swimlane Exit* component. To organize the assignment of resources to waiting entities, a partial de-central approach was chosen, namely by implementing a *Resource Manager* component which is part of each Swimlane.

The Resource Manager (RM) receives signals from amongst other the Swimlane Entry and Swimlane Exit that a new entity arrived and was placed in an entry queue, or that an entity is leaving a Swimlane. Based on a certain rule as specified by the modeller, the RM evaluates the states of all resources and queues within a Swimlane, and directs if possible a resource to a queue (by sending a signal to the appropriate queue containing information about the available resource and the destination queue).

4.3. Implementation with DEVSDSOL

DSOL, which stands for "Distributed Simulation Object Library", was selected to provide the simulation and execution functionalities (Jacobs 2005). DSOL is a proven multi-formalism simulation library which can be considered as a generic purpose simulation tool. It is written in the Java programming language and has been used effectively in various simulation projects. DSOL also supports execution of simulation models based on the DEVS formalism through the DEVSDSOL library (which is compatible with hierarchical DEVS) (Seck and Verbraeck 2009). The choice for DSOL was made because of its flexibility and functionality regarding simulation, and its support for the DEVS formalism.

Each DEVS component has been implemented in Java and these components are executable with DEVSDSOL simulation library. Some implemented components are BPMNStartEvent, BPMNEndEvent, BPMNTask, BPMNExclusiveGateway, BPMN-ParallelGateway, etc. Figure 4 shows the DEVS internal transition function of the Task element.

```java
@Override
protected void deltaInternal() {
    if (this.phase.equals(active)){

        if (this.parallelTaskList.isEmpty()){
            prt("[Task " + myID + "]  dINT, phase = " + this.phase + ",
            parallelTaskList() == 0; no more entities in the taskList");
            this.phase = this.passive;
            this.sigma = this.phase.getLifeTime();
            prt("dInt - new sigma: " + sigma);
            return;
        }
        else if (this.parallelTaskList.size() > 0){
            prt("[Task " + myID + "]  dINT, phase = active, parallelTaskList() >
            0; (some) entities are still in the task list, namely " +
                    this.parallelTaskList.size() + " entities with ID " +  this.
                    parallelTaskList.get(0).getID());
            this.phase = this.active;
            this.sigma = this.parallelTaskList.get(0).getTimeRemainingInTask();
            prt("dInt - sigma is set to: " + sigma);
            return;
        }
        else {
            prt("dInt-EXCEPTION");
        }
    }
}
```

Figure 4: Sample Code from Task Component

In order to provide a higher level abstraction mechanism to our library, we applied the model driven development framework presented in (Cetinkaya, Verbraeck and Seck 2011). Next section gives brief information about the framework and then explains how it is performed in this work.

5. APPLYING THE MDD4MS FRAMEWORK

MDD4MS is a model driven development framework for modelling and simulation. The framework suggests an M&S life cycle with five stages (Problem Definition, Conceptual Modelling, Specification, Implementation and Experimentation), metamodel definitions for different stages, model to model (M2M) and model to text (M2T) transformations for the metamodels and a tool architecture for the overall process. MDD4MS presents a sample prototype implementation which is developed in Eclipse. The MDD4MS prototype provides:

- a BPMN metamodel and a BPMN editor,
- a DEVS metamodel and a DEVS editor,
- a DEVSDSOL metamodel and a DEVSDSOL editor,
- a BPMN 2 DEVS M2M transformation,
- a DEVS 2 DEVSDSOL M2M transformation,
- a DEVSDSOL 2 Java M2T transformation.

In this study, we used the BPMN editor to draw our business process models. A sample business process model is shown in Figure 5. Since the MDD4MS prototype provides generic model transformation rules for BPMN, we rewrote some rules for BPMN2DEVS M2M transformation. In this way, we directly transformed the visual modelling elements to the components that we implemented in our library.

For example, Figure 6 shows the rule to transform a Swimlane to a coupled model. We added the part that generates a Resource Manager with one input and one output port for each Swimlane.

The auto generated DEVS model via model transformations is shown in Figure 7. Once we have the DEVS model, the MDD4MS prototype automatically generates the DEVSDSOL model and the java code for

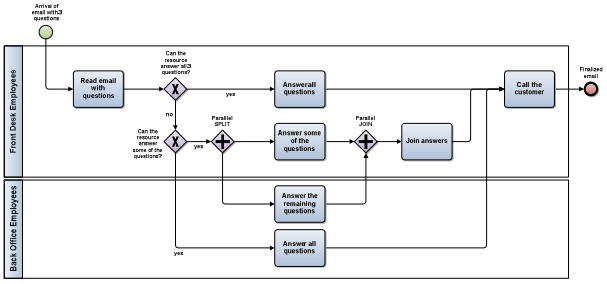

Figure 5: Sample Business Process Model.

```
rule BPMNSwimlaneToDEVSCoupled_inRoot {
    from
        s: CM_Metamodel!BPMNSwimlane (s.isInRoot())
    to
        t: SM_Metamodel!DEVSCoupledComp  (
            SMParentModel <- s.CMParentModel,
            DEVSComponents <- s.BPMNActivities,
            ...
        ),
        resourceM : SM_Metamodel!DEVSAtomicComp (
            DEVSParentComp <- t,
            Name <- 'RM_' + s.Name,
            DEVSComponentType <- 'ResourceManager',
            ImplementationLink <- 'this, "ResourceManager", ' + s.getQueueMode(
        ),
        outRM: SM_Metamodel!DEVSOutputPort (
            Name <- 'out',
            DEVSPortParent <- resourceM
        ),
        inRM : SM_Metamodel!DEVSInputPort (
            Name <- 'in',
            DEVSPortParent <- resourceM
        )
    do {
        thisModule.DefineResourceManager(t, resourceM, Sequence{outRM, inRM});
    }
}
```

Figure 6: A sample rule from BPMN_2_DEVS.atl.

coupled components that uses the implemented classes for BPM modeling elements in our library. In other words, visual business process models, drawn by the BPMN editor, are transformed to executable Java code and they can be simulated with DSOL.

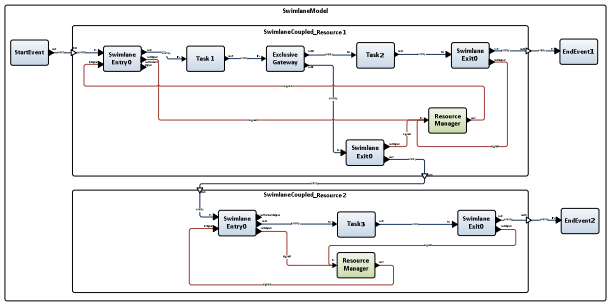

Figure 7: Auto generated DEVS Model.

6. DISCUSSION

This work proposed a new modelling approach for consultants to model and analyse business processes based on a proven theory, industry-wide standards and active end-user involvement during the design process.

A library of DEVS-based BPMN modelling elements is implemented with Java that uses the DSOL simulation library to provide the simulation capabilities. It provides an easy way of dynamic modelling for consultants with limited knowledge of simulation model development. As a future work, the credibility of our approach will be evaluated.

REFERENCES

Aguilar-Savén, R.S., 2004. Business Process Modelling: Review and Framework. *International Journal of Production Economics*, 90(2), 129-149.

Banks, J., 1998. *Handbook of Simulation - Principles, Methodology, Advances, Applications, and Practise* (p. 849). New York: John Wiley & Sons, Inc.

Bosilj-Vuksic, V., Ceric, V. and Hlupic, V., 2007. Criteria for the evaluation of business process simulation tools. *Interdisciplinary Journal of Information, Knowledge, and Management*, 2, 73–88.

Cetinkaya, D., Verbraeck, A. and Seck, M. D., 2010. Applying a Model Driven Approach to Component Based Modeling and Simulation. *Proceedings of the 2010 Winter Simulation Conference* (pp. 546-553). Baltimore, MD.

Cetinkaya, D., Verbraeck, A. and Seck, M. D., 2011. MDD4MS: A Model Driven Development Framework for Modeling and Simulation. *Proceedings of the Summer Computer Simulation Conference 2011*. Den Haag, NL.

Harrington, H., 1991. *Business Process Improvement: The Breakthrough Strategy for Total Quality Productivity*. New-York: McGraw Hill.

Jacobs, P. H. M., 2005. *The DSOL Simulation Suite*. Thesis (PhD). Delft University of Technology.

Maguire, M., 2001. Methods to support human-centred design. *International Journal of Human-Computer Studies*, 55(4), 587-634.

Melão, N. and Pidd, M., 2003. Use of business process simulation: A survey of practitioners. *Journal of the Operational Research Society*, 54(1), 2-10.

Peterson, J. L., 1981. *Petri Net Theory and the Modeling of Systems* (p. 290). New Jersey: Prentice Hall.

Pidd, M., 1996. *Tools for Thinking: Modelling in Management Science*. Chichester, John Wiley & Sons, Inc.

Pidd, M., 2002. Simulation Software and Model Reuse: A Polemic. *Proceedings of the 2002 Winter Simulation Conference*. (p. 772-775)

Pidd, M., 2004a. Computer Simulation in Management Science (5th ed., p. 311). West Sussex, England: John Wiley & Sons, Inc.

Pidd, M., 2004b. Simulation Worldviews – So What?. *Proceedings of the 2004 Winter Simulation Conference, 2004*. England: John Wiley & Sons, Inc.

Seck, M.D. and Verbraeck, A., 2009. DEVS in DSOL: Adding DEVS Operational Semantics to a Generic Event-scheduling Simulation Environment. *Proceedings of the Summer Simulation Multiconference 2009*. Istanbul, Turkey

Van Eijck, D. T. T. and De Vreede, G.-J., 1998. Simulation support for organizational coordination. *Proceedings of the 1998 Hawaiian Conference of Systems Sciences* (Vol. 1, p. 633–642). Los Alamitos: IEEE Computer Society.

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

Zeigler, B. P., Praehofer, H. and Kim, T. G., 2000. *Theory of Modeling and Simulation* (Second Edi.). Academic Press.

AUTHORS BIOGRAPHY

IGOR J. RUST is an M.Sc. graduate in Systems Engineering, Policy Analysis and Management and received his degree in 2011 from Delft University of Technology. In 2011 he received his B.Sc. at the faculty of Technology, Policy and Management, also from Delft University of Technology. His B.Sc. thesis was nominated for the Dutch Logistics Thesis Award 2011. His interests include discrete event simulation and mathematical programming. His e-mail address is <i.j.rust@student.tudelft.nl>

DENIZ CETINKAYA is a Ph.D. student at Delft University of Technology. She is in the Systems Engineering Group of the Faculty of Technology, Policy and Management. She received her M.Sc. in Computer Engineering from the Middle East Technical University, Turkey in 2005. She received her B.Sc. with honours in Computer Engineering from the Hacettepe University, Turkey in 2002. Her research focuses on component based modelling and simulation. Her e-mail address is <d.cetinkaya@tudelft.nl>.

MAMADOU D. SECK is an Assistant Professor in the Systems Engineering Group of the Faculty of Technology, Policy, and Management of Delft University of Technology. He received his Ph.D. degree from the Paul Cezanne University of Marseille and his M.Sc. and M.Eng. degrees from Polytech Marseille, France. His research interests include modelling and simulation formalisms, dynamic data driven simulation, human behaviour representation and social simulation, and agent directed simulation. His e-mail address is <m.d.seck@tudelft.nl>.

IVO WENZLER is part of Accenture's Global Consulting Experts Group and was until recently a Senior Executive within Accenture's Talent and Organization Service Line. For one day per week he also holds a position of Associate Professor at Delft University of Technology where he teaches a master's course in simulation game design. His e-mail address is <ivo.wenzler@accenture.com>.