Simulation-based design for infrastructure system simulation

Fumarola, M; Huang, Y; Tekinay, C; Seck, MD

Publication date
2010

Document Version
Accepted author manuscript

Published in
Proceedings of The 2010 European Simulation and Modelling Conference

Citation (APA)
Fumarola, M., Huang, Y., Tekinay, C., & Seck, MD. (2010). Simulation-based design for infrastructure system simulation. In G. K. Janssen, K. Ramaekers, & A. Caris (Eds.), *Proceedings of The 2010 European Simulation and Modelling Conference* (pp. 288-293). Eurosis-ETI.

SIMULATION-BASED DESIGN FOR INFRASTRUCTURE SYSTEM SIMULATION

Michele Fumarola, Yilin Huang, Çagri Tekinay and Mamadou D. Seck
Systems Engineering Group
Faculty of Technology, Policy and Management
Delft University of Technology
PO Box 5015, NL-2600GA Delft
The Netherlands
E-mail: {m.fumarola,y.huang,c.tekinay,m.d.seck}@tudelft.nl

KEYWORDS
Simulation Library, DEVS, Infrastructure, Logistics

ABSTRACT

Simulation models are often used to analyze the behavior and performance of infrastructure systems. The use of simulation models in multi-actor design processes is restricted to the analysis phase after conceptual designs have been completed. To use simulation models throughout the design process, simulation environments need to be adapted to support the interactive and iterative nature of design processes, making the shift from a 'hard' to a 'soft' systems perspective. By using the Discrete Event Systems specification, modular and hierarchical component libraries can be constructed that can be used in participatory design processes. In this paper, we present our experiences on developing and using simulation environments in design processes for container terminals and rail systems. We present the cases, the architecture of the design environments and the use of them in actual design processes. Hereafter, we will discuss the enhancement we are studying to support the multiple perspectives present in multi-actor environments.

INTRODUCTION

The challenges in designing modern infrastructure systems are abundant. Multiple actors are involved that have to keep track of investments, technical feasibility, and, the nowadays increasingly important, societal and environmental impact. Modeling and simulation (M&S) can be used to support these multi-actor environments. However, constructing simulation models requires specialized skills and a lot of time and effort. This does not match well with the interactive and flexible character of a multi-actor design process. The use of M&S in a multi-actor design process has to adhere to certain requirements before it can be successfully employed.
Modern M&S environments stem from a 'hard systems' paradigm that is based on the conception that "there is a desired state, S(1), and a present state S(0), and alternative ways of getting from S(0) to S(1); 'problem solving,' according to this view, consists of defining S(1) and S(0) and selecting the best means of reducing the difference between them" (Checkland 1978). This paradigm suffers from a lack of recognition of different interests in a decision making process. According to Rosenhead and Mingers (2001) this means that in current simulation environments

- problem formulation is in terms of a single objective;
- there are overwhelming data demands;
- people are treated as passive objects;
- there is a single decision-maker with abstract objectives from which concrete actions can be deduced;
- and there is an attempt to abolish future uncertainty, pre-taking future decisions.

'Soft systems' (Checkland 1978) have been presented as a methodology that is closer to reality. It acknowledges multiple actors and mostly diverging opinions, and views the decision making process as a learning process wherein the various actors engage to understand the problem and each other's opinion.
Although the soft systems methodology is not intended to be used with simulation models, this could be a fruitful combination. Aughenbaugh and Paredis (2004) provide a very thorough and to-the-point explanation as to what simulation can bring to design and decision making:

> Without modeling and simulation, design relies on implicit knowledge. Implicit knowledge is unreliable in that designers do not know the assumptions and uncertainty in the knowledge explicitly. When decisions are coupled and require input from several experts, there is no way to make tradeoffs using only implicit knowledge about uncertainties.

Robinson (2001) reports on the use of simulation as a way of understanding the complexity of the problem and to facilitate the discussion among stakeholders. den Hengst et al. (2007) further elaborates on Robinson (2001) by discussing what can be done but what are still the limitations on using simulation models to foster decision making processes. The major shortcomings of modern simulation environments in multi-actor settings are

- the expertise needed to build simulation models;
- the complex code to build the simulation model, verify and validate it;
- the lack of knowledge of the actors concerning M&S results in a hard to accept model;
- and finally, running the experiment takes a long period (multiple hours) of time.

To use simulation models in design processes, it is important to shift from a hard systems perspective to a multimethodological approach as presented by Robinson (2001) and den Hengst et al. (2007). This shift sets specific requirements on the design environment and changes the use of simulation models: whereas now simulation models are purely used as an analysis tool after the conceptual design phase has been completed, it will become a discussion platform throughout the entire design process.

Designing multi-actor design environments

According to Simon (1996), design processes follow a path of first structuring the problem, followed by a formulation of alternative solutions based upon selected criteria and finally by a selection of the best alternative. To allow this, design environments need to provide sufficient flexibility in constructing the alternatives. This calls for a modular framework that supports the construction of alternative designs. den Hengst et al. (2007) reminds us that the complex code can be a burden: hiding this is necessary to use simulation models in design processes. A feasible approach to allow modularity and hide internal code would be to use component-based modeling. Indeed, components provide the major advantage of being self-contained, reusable, replaceable and customizable (Verbraeck 2004).
Hu et al. (2005) discuss the suitability of the Discrete Event Systems Specification (DEVS) (Zeigler and Praehofer 2000) for component based-modeling and simulation. To facilitate the easy development of simulation models using existing components, they suggest the use of variable structure in DEVS, which means dynamically adding and removing components and couplings. Three reasons are given for using variable structure in DEVS: (1) to model systems that exhibit structure and behavior changes, (2) to design and analyze a system under development, and (3) to be able to load only a subset of the system's component when simulating very large models. As the modularity and hierarchical structure of model components provided by the DEVS formalism could benefit model construction in general, recent work has been focused on developing the Event-Scheduling DEVS library (ESDEVS) (Seck and Verbraeck 2009). ESDEVS implements the parallel DEVS formalism on top of the DSOL library (Jacobs 2005). The ESDEVS is based on the event-scheduling worldview wherein executions of the internal transition function are scheduled according to the specified time advance function and unscheduled at the reception of external events. The confluent transition function handles the coincidence between internal and external events. Dynamic structure DEVS is also implemented in the ESDEVS library so that components and coupling relations can be added and removed dynamically during simulation runtime.
A component-based framework hides complex details of simulation models while allowing an easy way of constructing alternative designs. This fits perfectly the idea of having a design environment build for supporting an interactive and iterative multi-actor design process. A multi-actor design environment should support participatory design: an engineering design process is seldom performed by a single person or an actor. Many actors are involved and they all try to achieve their own objectives. A design process should support a certain convergence of interests of all actors.

Outline of the paper

In this introduction, we have motivated the need of requiring a shift in the use of simulation to support design processes in multi-actor environments. In our research, we are particularly interested in supporting the design of logistics systems. In the next section, we are going to present a case study involving the design of automated container terminals. For this case study, we have developed a simulation components library and accompanying tools to use this library in interactive design sessions. After this, we presentd a similar approach for the development of rail transport systems. In both case studies, we have studied how simulation models can be used interactively in the design process, instead of purely as an analysis tool after the conceptual design is concluded. Hereafter, we will discuss what else is needed in a design process to better support decision makers. We will present some concepts that are part of our future work and finally conclude the paper.

AUTOMATED CONTAINER TERMINAL OPERATIONS

Container terminals have become essential in today's supply chain of goods. There is a steady increase in the use of automated equipment to improve the effiency

of loading and stacking containers in terminals. Equipment for stacking containers can be automated as well as equipment that is used for transporting the containers from the stacks to the quay cranes. Exploring alternative types of equipment and design options is a time consuming and challenging task. A simulation component library coupled to an AutoCAD design environment can fasten the assessment of design decisions.

Architecture of the design environment

The 3D visualization tool serves as a platform for communicating design decisions to all actors involved in the process. In Fumarola and Versteegt (2011) we have shown that 3D visualization environments enhance communication among actors by having an indisputable and understandable presentation platform. Technically less inclined people tend to prefer 3D visualizations above technical 2D CAD drawings. For this reason, the 3D visualization tool, which in diagram 1 is called 'Virtual Terminal', is an integral part of the design environment.
The simulation components library has been developed in DEVS (Zeigler and Praehofer 2000) using DSOL ES-DEVS (Seck and Verbraeck 2009). It has been discussed extensively in Fumarola et al. (2010): it contains a set of components for each type of equipment in a common automated container terminal. The components library contains components for quay cranes, automated guided vehicles, and rail mounted gantries. To control the equipment, is has an implementation of terminal operating system containing algorithms for path finding and scheduling.
Both the visualization and simulation models are instantiated using an XML file that is generated based on CAD drawings. The XML contains the elements extracted from the CAD design that are important for the visualization and simulation model. The visualization tool uses a component library of 3D meshes to use the appropriate 3D mesh for the given equipment. Likewise, the simulation environment queries for the right simulation component using the right name and parameters. Using this structure, one single CAD drawing can be used to fully instantiate the visualization and simulation components. This is reflected in Figure 1.

Use of the design environment

The design environment is made to support the multiple actors present in the design process for automated container terminals. In Hu (2008) and Derksen (2009), whose work were part of the same case study discussed here, an analysis of the actors involved in the decision making process has been carried out. The major identified actors are involved with

- innovation as a way to deploy novel equipment to increase productivity;
- business as to keep costs and revenues in balance;
- engineering as to design a technically feasible terminal;
- and, finally, environment and safety.

The design process of container terminals can be partitioned roughly in five phases: project acquisition phase, global engineering phase, detailed engineering phase, implementation phase, and operation phase (Fumarola and Versteegt 2011). The design environment can support actors in the global engineering phase and the detailed engineering phase. In the global engineering phase, conceptual designs are made that are based on best practices and rules of thumb. Simulation is mostly not used in this phase because of the quick and highly iterative character of this phase. The design environment discussed here is highly suitable for the global engineering phase. It can be used to quickly assess different designs using its easy to use interface. The same models can later be used in the detailed design phase, which serves to study in depth the behavior of the system using extensive scenarios.

RAIL SIMULATION LIBRARY

Modern rail transport planning and design are complex and time-consuming. Many stakeholders such as the transport authorities and operators are involved in decision making, each to their own objectives and responsibilities. To design and develop adequate simulation tools that support these domain-experts in the decision making process is a challenge. Different aspects of the railway system, e.g. the infrastructure, control measures and timetables, shall be combined in a self-contained simulation package to allow for comprehensive experimental analysis by multiple actors, and to provide them with a platform that enhances common understanding to reach well-informed decisions. Due to the long life span of rail infrastructure and services, changes often come up which lead to new issues to study. Thus rail-bounded modeling particularly requires malleable composition and configuration in order to promote model reusability. The design requirements mentioned earlier embrace these needs, and therefore they are used as the guidelines to design our rail simulation library.

Rail Model Hierarchy

Similar to the first case, rail models in the library are defined following the DEVS formalism (Zeigler and Praehofer 2000), which benefits the model structure with modularity and hierarchical composition (or decomposition). The library is built as an extension of the ES-DEVS library (Seck and Verbraeck 2009). Two classes of models are distinguished, the *rail element*s and the *rail component*s, as shown in Fig. 2, which are in line with

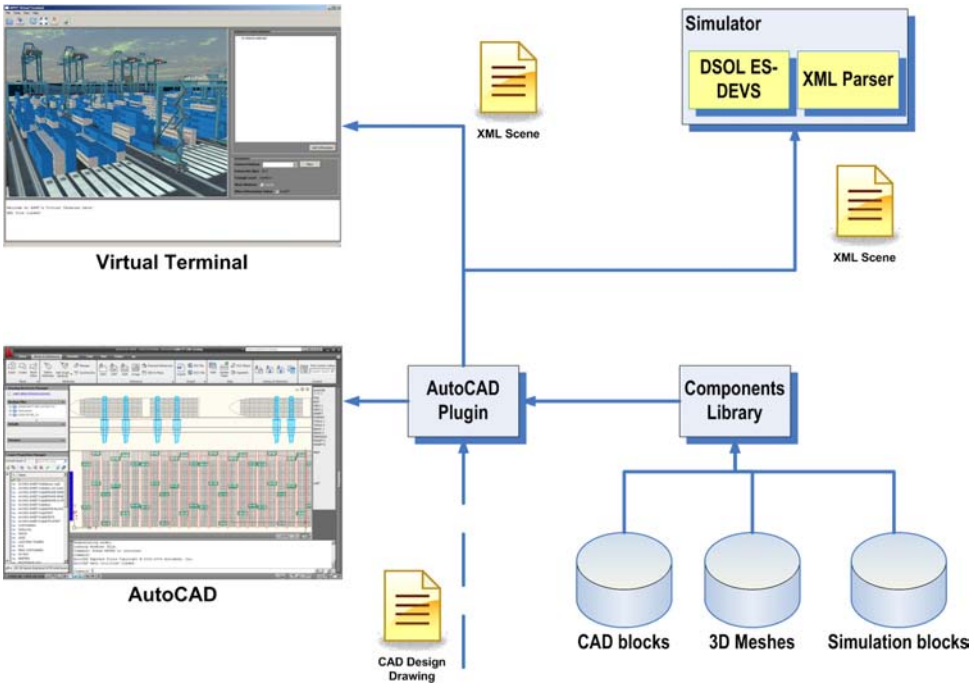

Figure 1: Architecture of the automated container terminal design environment

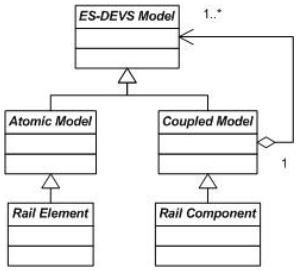

Figure 2: (Simplified) Model Composite Relation

the concept of atomic and coupled models in DEVS. Rail elements are the irreducible models that can not be decomposed, e.g. a rail vehicle, sensor, signal or a piece of track segment; more see (Huang et al. 2010). Rail components are resultants of composition that may contain rail elements and rail components. The rail elements and components can be composed into more complex components, which in turn can be composed, and so further recursively. As such, users can build complex rail network with the components without knowing the complex internal code.

Fig. 3 illustrates a simple example of the rail network model hierarchy. A cloud shape denotes a rail component, a rectangle denotes a rail element, and a one-to-more relation is represented by a triple-line. In general, a top level rail model (component) contains one or more sources and sinks, which respectively generates and removes vehicles in the simulation. (Each vehicle generator in a source can schedule the vehicle generation according to a different timetable.) The rail model

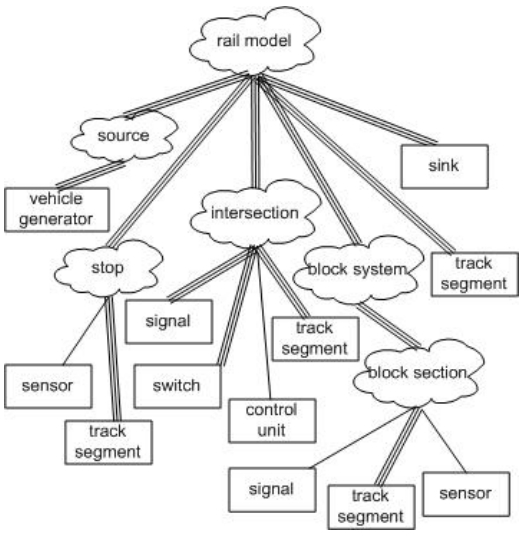

Figure 3: An Example of the Rail Model Hierarchy

composition is flexible. It may contain stops, intersections, block systems, and other model composition. The intersection, e.g., has signals to control the accessibility of the track sections. Switches allow vehicles to move from one track over another. The control unit computes the signalling logic depending on the occupancy of the tracks and switches. Each of those rail elements models one functionality of the rail infrastructure, and the their aggregation forms a higher level component that performs more complex tasks.

A rail vehicle (model element) is generated in a source. As the vehicle drives in the rail network, it is moved

from one rail component to another (dynamic structure DEVS). The rail network model, at its lowest description level, is a directed graph of linked rail infrastructure elements (track segments, sensors, switches and signals). A vehicle model is linked directly with an infrastructure element, each of which is capable of *message propagation*, the principle object-to-object communication mechanism used in the library. Its base concept is the DEVS message passing through paired I/O ports.

Message Propagation

The message propagation can be along the traffic current or in the opposite direction. When a vehicle lacks information about its next infrastructure or the preceding vehicle, it sends a *request-message* forward. The infrastructure element that gets the message propagates the message until the next infrastructure of interest and/or a preceding vehicle is found. The found element (e.g., a track segment that has speed limit change, a signal or a vehicle) replies with a *response-message* which is propagated backward until it reaches the original sender of the request-message. If an element changes state, it also sends an *update-message* backward to inform its potential succeeding vehicle. A message contains information of the sender, the contemplated receiver (if necessary), and the distance between the sender and the receiver. According to this information, a vehicle's movement is computed until its next infrastructure or its preceding vehicle (if any). Once the vehicle reaches the next infrastructure (it will be linked to the next infrastructure) or its preceding vehicle, a new round of request and response message exchange is performed and new movement is computed. Message propagation is a communication mechanism that is transparent to model users. The users only need to setup the rail network. The message sending, forwarding, receiving and the dynamic linking of the vehicles are accomplished by the model elements autonomously.

Rail Model Builder

To simplify model construction and configuration, we have developed the CAD Rail Model Builder (CRMB) as one of the possible approaches to interface with the rail library. Many railway companies and authorities use CAD and GIS applications for infrastructure design. The resulting design files can be used for model generation. However, as these files often do not contain information about the network topology in a hierarchical way, a couple steps of data cleaning, preprocessing and transformation are necessary to make the data useful for DEVS model generation. For example, when the orientation of the track drawing entities is not in accord with the traffic current, the track orientation needs to be corrected first. The positions of switches and crossovers can be detected where more than two track segments connect. Based on the position and orientation of the connecting tracks, the switch type is determined whether it is converging or diverging. In order to identify certain model components, pattern recognition technics are applied. The identified elements, e.g. tracks, switches and crossovers, that belong to one component are grouped and indexed for model generation. Some infrastructure components do not have a determinate pattern of its composite elements. These elements are defined by users in additional data sources (e.g., a configuration file) for model generation.

Enhancements of the library

The rail simulation library is still in development. We plan to separate the rail components into light-rail and heavy-rail specific packages. Both based on the generic rail simulation core. Such restructuring could reduce the library core to a minimum. A graphical modeling interface is also in design to provide a model configuration alternative for users. In a recent paper (Huang and Verbraeck 2009), we proposed a dynamic data-driven approach for rail simulation. The idea is to automate model calibration and validation by comparing model output with rail operation data. In this context, the DEVS based library design would also benefit our development.

FUTURE WORK

In the light of the first and second case studies, we show that DEVS model libraries can provide a feasible framework for modeling infrastructure systems. Supporting component-based modeling, developed DEVS libraries offer modularity to the designer to customize and reuse the model components. Yet, the conducted case studies reveal that several issues need to be addressed to fully support decision making in infrastructure systems.
The design process of infrastructure systems is multi-actor by nature in which every actor has its own interests and perceives the system in his own way. Therefore, the models should support different perspectives from various actors. What is more, each unique perspective can be decomposed into various levels of abstraction, so called resolutions. This will allows the actors to change the resolution of their models to specify the inputs at different levels of precision, analyze the output and reason about the cause-effect relationships. However, the existing designs lack a common consistent framework resulting in a single perspective and statically defined models at different resolutions. A framework with a set of rules, design guidelines and constraints need to be introduced in the existing design environment to allow a problem to be studied inside the whole assumptions space, instead of having a base case and trying to have some intuitions from that. This is important when dealing with the design of infrastructure systems. In a recent paper

(Tekinay et al. 2010), we discussed the key issues and preliminary design ideas to provide a multi-resolution and multi-perspective modeling in multi-actor environments.

CONCLUSIONS

The purpose of this paper is to present our experiences when designing and using the DEVS model libraries to support decision making in infrastructure systems. Two different model libraries including the library for automated container terminals and light-railway systems is given a baseline case to discuss the challenges throughout the design process. DEVS model libraries allow designer to build component-based models which have the advantage of being modular, extensible and reusable. Such capabilities provided by DEVS model libraries are arguably essential for designing infrastructure systems where vast number of components and control mechanisms involved. For the further studies, DEVS model libraries will be a solid backbone for us to deal with the next challenges of providing multiple perspectives at different levels of abstractions to support multiple actors.

REFERENCES

Aughenbaugh J. and Paredis C., 2004. *The Role and Limitations of Modeling and Simulation in Systems Design.* In *ASME 2004 International Mechanical Engineering Congress and Exposition.* American Society Of Mechanical Engineers, 13–22.

Checkland P., 1978. *The origins and nature of hard systems thinking. Journal of applied systems analysis*, 5, 99–110.

den Hengst M.; de Vreede G.; and Maghnouji R., 2007. *Using soft OR principles for collaborative simulation: a case study in the Dutch airline industry. Journal of the Operational Research Society*, 58, 669–682.

Derksen T., 2009. *Modelling Modes of Operation: A DSS and a decision making process on the selection of the mode of operation for APM Terminals.* Master's thesis, Delft University of Technology, the Netherlands.

Fumarola M.; Seck M.; and Verbraeck A., 2010. *A DEVS component library for simulation-based design of automated container terminals.* In *Proceedings of the 3rd International ICST Conference on Simulation Tools and Techniques.* Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering.

Fumarola M. and Versteegt C., 2011. *Supporting automated container terminal design processes with 3D virtual environments.* In H. Yand and S. Yuen (Eds.), *Handbook of Research on Practices and Outcomes in Virtual Worlds and Environment*, IGI Global.

Hu H., 2008. *Choosing the Optimal Mode of Operation for Marine Container Terminals.* Master's thesis, Delft University of Technology, the Netherlands.

Hu X.; Hu X.; Zeigler B.; and Mittal S., 2005. *Variable Structure in DEVS Component-based Modeling and Simulation. Simulation*, 81, no. 2, 91–102.

Huang Y.; Seck M.D.; and Verbraeck A., 2010. *LIBROS-II: Railway Modelling with DEVS.* In B. Johansson; S. Jain; J. Montoya-Torres; J. Hugan; and E. Yücesan (Eds.), *Proceedings of the 2010 Winter Simulation Conference.* IEEE. To be appear.

Huang Y. and Verbraeck A., 2009. *A Dynamic Data-Driven Approach For Rail Transport System Simulation.* In M.D. Rossetti; R.R. Hill; B. Johansson; A. Dunkin; and R.G. Ingalls (Eds.), *Proceedings of the 2009 Winter Simulation Conference.* IEEE, 2553–2562.

Jacobs P., 2005. *The DSOL simulation suite - Enabling multi-formalism simulation in a distributed context.* Ph.D. thesis, Delft University of Technology.

Robinson S., 2001. *Soft with a Hard Centre: Discrete-Event Simulation in Facilitation. The Journal of the Operational Research Society*, 52, 905–915.

Rosenhead J. and Mingers J., 2001. *Rational Analysis for a Problematic World Revisited: Problem Structuring Methods for Complexity, Uncertainty and Conflict.* John Wiley & Sons.

Seck M.D. and Verbraeck A., 2009. *DEVS in DSOL: Adding DEVS operational semantics to a generic Event-Scheduling Simulation Environment.* In *Proceedings of the 2009 Summer Computer Simulation Conference.*

Simon H., 1996. *The Sciences of the Artificial.* The MIT Press, third ed.

Tekinay C.; Seck M.D.; Fumarola M.; and Verbraeck A., 2010. *A Context-Based Multiresolution Multiperspective Modeling Framework.* In B. Johansson; S. Jain; J. Montoya-Torres; J. Hugan; and E. Yücesan (Eds.), *Proceedings of the 2010 Winter Simulation Conference.* IEEE. To be appear.

Verbraeck A., 2004. *Component-based Distributed Simulations. The Way Forward?* In *Proceedings of the 18th Workshop on Parallel and Distributed Computer Simulation.* IEEE Computer Society Press, Los Alamitos, CA, 141–148.

Zeigler B.P. and Praehofer H., 2000. *Theory of Modeling and Simulation.* Academic Press.