Universitätsbibliothek

Aktive Fernleihe

UNIVERSITÄT PASSAU

Universität Passau · 94030 Passau

University of Arizona
Library
Interlibrary Loan
1510 E University
Tucson AZ 85721-0055 USA

Auskunft erteilt: Fr. Antesberger, 0851 509-1627
Telefax: 0851 509-1677
e-mail: ubafl@uni-passau.de

Zeichen:
Datum: 17.6.11

Rechnung für Ihre Fernleihbestellung
calculation for your borrowing order

AIRMAIL

Sehr geehrte Kolleginnen und Kollegen, Dear Colleagues!

Bitte senden Sie diesen Schein mit dem Buch bzw. den Büchern und

_____ **Internationalen Antwortscheinen** oder _____ Voucher an uns zurück.

Please return this slip with [redacted]

5 **Coupon-Réponse International** (CRI) **or** 1 **Voucher.**

Best.-Nr.:/Order-Nr.:	Signatur:
951826	as copy

Mit freundlichen Grüßen
Yours sincerely
Antesberger

TUCSON/USA

AIRMAIL

Von: "University of Arizona Library (AZU)" <askddt@u.library.arizona.edu>
An: <Utta.Materny@uni-passau.de>
CC: <Inge.Hopper@uni-passau.de>
Datum: 14.06.2011 18:28
Betreff: Interlibrary Loan Request - TN#: 951826

Dear Colleagues,

We would like to obtain an article or book chapter from your library.

Please send the SCAN/COPY to us by email, fax, or air mail.
Email is preferred.

Request Type: SCAN/COPY

Lender's Call number: 80/SS 2003 R565
Lender's Location: UB/SB Passau / IM, Lesesaal [80/]
OR
ISBN/ISSN:
OR
OCLC #: 634891278

Article Title: Complex Systems Design: A Simulation Modelling Approach
Article Author: Rozenblit, J.W.
Title: The international workshop on Harbour, maritime and multimodal logistics modelling & simulation : HMS 2003, September 18-20, 2003, Riga, Latvia /
Author:
Volume:
Issue:
Month:
Year: 2003
Publisher:
Pages: 17- 20

4 S.

TN#: 951826
Patron: Napalkova, Liana, Faculty, Electrical/Computer Engr

Please contact us if your fee exceeds $50.00 USD
We can pay by VISA, IFLA Vouchers or cheque.

Request Complies with the following provisions of U.S. copyright law or 108(g) (2) Guidelines:
CCL = older than 5 years

Regards,
Dana Von Berg

University of Arizona Library
Interlibrary Loan
1510 E University
Tucson AZ 85721-0055
USA

OCLC: AZU
Email: askddt@u.library.arizona.edu
ARIEL: 150.135.238.50
ODYSSEY: 150.135.238.6
Phone: 520-621-6438
Fax: 520-621-9868

COMPLEX SYSTEMS DESIGN: A SIMULATION MODELING APPROACH

Jerzy W. Rozenblit
Department of Electrical and Computer Engineering
The University of Arizona
Tucson, Arizona 85721-0104, U.S.A.
E_mail: jr@ece.arizona.edu

ABSTRACT

This paper provides an overview of the plenary presentation. It addresses the complexity of modern engineering systems and discusses simulation modeling techniques that can assists in the design of such systems. An approach for constructing and testing virtual design models prior to realization commitments is shown. This approach leverages from object-oriented and discrete-event simulation methods. Concepts for system level automation of the design process will be given and examples that range from embedded systems applications to high level symbolic visualizations will be discussed to support the theory-based concepts.

CONTEXT - DESIGN PROBLEM

Modern engineering design is a highly complex process involving a multiplicity of objectives, constraints, materials, and configurations. Despite great strides in computational tools such as high performance workstations and CAD systems intended to help to cope with this rising complexity, the design process remains error prone. Given the often severe constraints imposed by cost, environmental impacts, safety regulations, etc., designers are forced to make compromises that would not be necessary in an ideal world. Simulation modeling is recognized as a useful tool in assessing the quality of design choices and arriving at acceptable trade-offs. However, computer simulation and other advanced computational tools are of limited effectiveness without a methodology to induce a systematic handling of the multitude of goals and constraints impinging on the design process. Therefore, in our research, we use modeling and simulation concepts to unify engineering design activities and systematically construct and evaluate complex systems' design models.

In the presentation, we will discuss a model-based design methodology (Rozenblit 2001) and its tenets. We will describe the requirements specification, the structural and behavioral model development process as well as model mapping, i.e., the process of allocating virtual design components to hardware and software realizations. Then, we will focus on requirements-based test sequence generation and specification of simulation experimental conditions.

Our framework targets primarily embedded systems. However, applications that extend beyond that domain will also be presented. Namely, selected examples that interface symbolic visualization software architectures with autonomous real-time agents will be shown.

MODEL BASED DESIGN

The model-based process, as shown in Figure 1, uses stepwise refinement of simulateable models and offers the opportunity to abstract system components at multiple levels of representation. In this methodology, a set of requirements and constraints is obtained for the system to be modeled. The system is then described as an abstract model that is a combination of its structural and associated behavioral specifications.

Given a set of design objectives, requirements and constrains, we first build a simulateable model of the system under design (SUD). Modeling entails the specification of structure (object model) and behavior (dynamics). Object modeling (i.e., model structuring) typically leads to a specification of a structure instance. This is commonly done in a graphical language such as the Unified Modeling Language (UML), which has become a *de facto* tool for object modeling. However, rather than generating a *single* instance of an object model, we advocate the development of a generative object representation that underlies the entire family of possible design configurations for a problem domain at hand. Indeed, UML allows us to capture the multiplicity of design views and taxonomies (specializations) of components through its decomposition and specialization relationships. An enormous variety of decompositions and specializations in large scale systems leads to a combinatorial explosion of design choices. To harness this complexity, procedures are needed that prune out instances of design which best fit design objectives and requirements. Thus, we use heuristic search methods that convert design requirements into selection (for choices from among alternatives offered by taxonomic relationships) and synthesis (for aggregations from

among decompositions) into production rules. Then, we search design spaces for best alternatives. The outcome of the search is a set of sub-optimal instances of design object models (Rozenblit and Huang 1991).

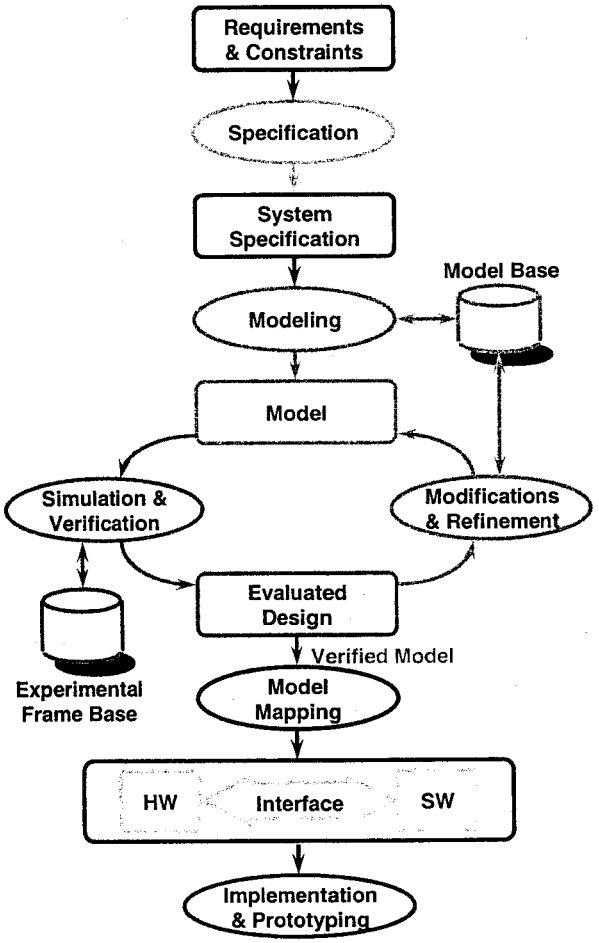

Figure 1. Model-Based Design

The dynamics (behaviour) of model components is specified using various modeling formalisms such as the discrete event system specification (DEVS) (Zeigler 1984), finite state machines, Petri nets, etc. The choice of the specification formalism is based on the system's domain. Both the structural and behavioral specifications constitute a virtual representation of the system under design (SUD). This is a "design blueprint" from which a system will be realized. Model components remain implementation and realization (i.e., hardware or software) independent.

We verify correctness of models through computer simulation. A simulation test setup is called an *experimental frame* (Zeigler 1984). It is associated with the system's model during simulation. A frame specifies conditions under which the model of the system is observed. Simulation is then executed according to the run conditions prescribed by the frames. At the end of the simulation process the "best" (polyoptimal) virtual system prototype is obtained. The design is then partitioned into hardware, software and corresponding interfaces using a process that we call *model mapping* (Schulz et al. 1998).

TESTING

Testing at the model level involves the creation of a set of test scenarios based upon the system requirements. These test scenarios are used to create a suite of experimental frames. The experimental frames are subsequently translated into scripts for real-time testing of the prototype physical system.

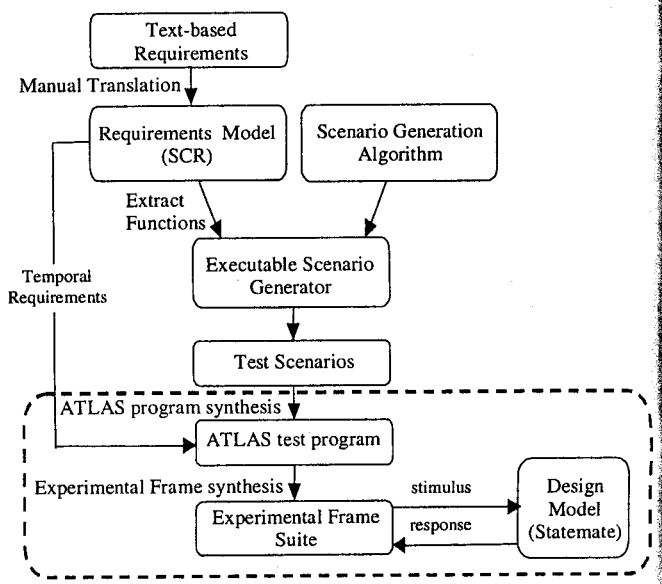

Figure 2. Testing Framework

Requirements-based testing

The requirements-based testing problem can be defined as follows: given as input a set of requirements for the SUD, produce as output a set of scenarios that adequately test the system. More specifically, as depicted in Figure 2, we map the requirements onto a requirements model using the Software Cost Reduction (SCR) (Heitmeyer et al. 1996) technique. Then, we generate test scenarios that exercise at least once each requirement identified in the requirements model. This is, in essence, a set covering problem, i.e., the problem of finding the minimum test sequence that covers all the requirements. To solve this problem, we use the SCR formalism (FSM-like specification), carry out the state space exploration through a controlled simulation of the requirements model, and thus generate a scenario tree that defines a set of inputs to the model. The test scenarios are then translated into experimental frames.

The experimental frames are used to apply stimuli to the design model and to gather the design model's responses. The results of the simulation are stored for further analysis. In (Cunning 2000), we have developed the test scenario generation algorithms.

Experimental-frame specification

Execution of design models is carried out using the experimental frame paradigm. Experimental frames define conditions under which models can be observed and experimented with. An experimental frame reflects modeling objectives. The statement of objective is translated into specific performance measures. Necessary output, input and control variables are defined so that such measures can be obtained through simulation experiments. An experimental frame a) subjects a model to input stimuli (which represent potential interventions into the model's operation), b) observes the model's reactions to the input stimuli and collects the data about such reactions (output data), and c) controls the experimentation by placing relevant constraints on values of the designated model state variables and by monitoring these constraints. Figure 3 illustrates the separation of a model and its experimental frame. A model can be executed in various experimental frames, each reflecting a specific objective of a simulation study.

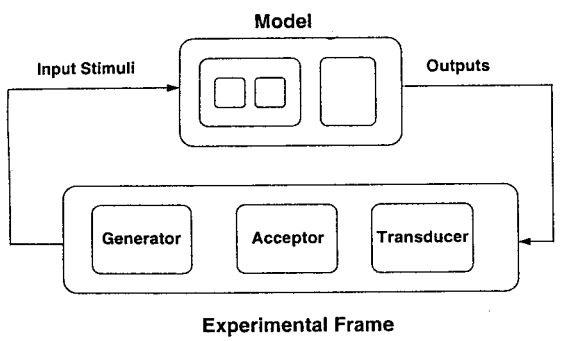

Figure 3. Model Experimental Frame Coupling

Experimental frames provide us with with a quantifiable means of trading off design solutions (in the form of simulation models) with respect to the set of design objectives. Such trade-off decision can be made by team members using multiple criteria decision making (MCDM) techniques as depicted in Figure 4. In the scenario shown in the figure, alternative design solutions are given as models $M_1,\ldots,M_n$. The experimental frames, EF_1 and EF_2 reflect two performance objectives, e.g., component utilization (EF_1) and task throughput (EF_2). The models can now be cross-evaluated in both frames and the tradeoff solution (the model M*) can be selected.

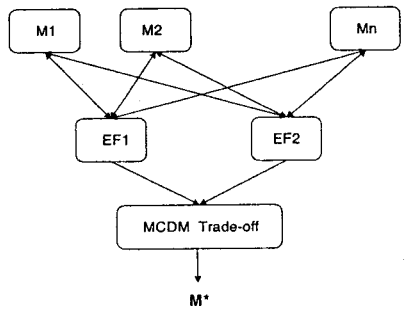

Figure 4. Trading-off Design Solutions using Experimental Frames and MCDM

Experimental frames are given concrete form. Employing the concepts of automata theory, a frame can be defined as a composition of a generator which produces the input segments sent to a model, an acceptor, that is a device that continually monitors the simulation run for satisfaction of constraints, and a transducer which collects the input/output data and computes summary performance measures. Experimental frame template specifications can be stored in an experimental frame base (recall Figure 1) for reuse and rapid simulation run setup.

PRACTICAL EXPERIENCES

Our laboratory conducts research in systems design and analysis, engineering of complex systems, software engineering, and embedded systems. The model-based design framework has been applied in design of real-time devices (examples include: autonomous cruise controllers, a safety injection system, elevator controllers, and robotic agents).

In addition to embedded system applications, we have been building a large scale object-oriented system for decision making in complex, information rich situations (such as military, peacekeeping, or disaster relief operations). The purpose of this work is to provide visualization capabilities to decision makers using advanced computer technology that symbolically abstracts the most important features of the information space. The technology facilitates rapid creation of tailored, low resolution, high semantic content visualizations of complex operations. Recent extensions (Peng et al. 2003) include a hybrid software/hardware that builds on the symbolic, object-oriented visualization software called Advanced Tactical Architecture for Combat Knowledge System (ATACKS) (Suantak et al. 2001). The extension is the design of a real-time robot agent layer that interacts wirelessly with ATACKS. This layer enacts decisions made by software agents, continuously relays the execution states back to ATACKS, and updates its actions as advocated by

replanning algorithms. The design and implementation will be presented with a small example that illustrates the hybrid system's operation.

CONCLUSIONS

The presentation will focus on providing the overview and technical details of our design approach. Several examples will illustrate the major tenets of the methodology. We will also discuss open issues such as the need to develop test generation algorithms that take into account timing relations as specified in the design requirements. To accomplish this, we will adopt explicit timing into the SCR requirements specification formalism, extend the scenario generation algorithms to classes of systems other than discrete event. We will also present ideas for *model compilation,* that is a process of transforming executable models into physical realizations. Our design framework produces virtual design blueprints in the form of simulateable models. They are implementation independent, i.e., no commitment to realization technology is made at this point. The ultimate goal is to map the model onto a prototype realization that combines hardware, software, and interface elements. A good analogy here would be compilation of the source code in a programming language (say C++) into executable code. Thus, our objective in model compilation is to take a simulateable model of an SUD and "compile" it into a format that lends itself to software and hardware prototyping.

REFERENCES

Cunning, S.J. 2000. "Automating Test Generation for Discrete Event Oriented Real-Time Embedded Systems." PhD Dissertation, The University of Arizona, USA.

Heitmeyer, C.L., Jeffords, R.D. and B. G. Labaw. 1996. "Automated Consistency Checking of Requirements Specifications." *ACM Transactions on Software Engineering and Methodology*, 5(3), (July), 231-261.

Peng, J., Rozenblit, J.W. and L. Suantak. 2003. "A Hybrid Architecture for Visualization and Decision Making in Battlespace Environments", *The Tenth IEEE Conference on Engineering of Computer-Based Systems*, 207-213.

Rozenblit, J.W. and Y.M. Huang. 1991. "Rule-Based Generation of Model Structures in Multifacetted Modeling and System Design," *ORSA Journal on Computing,* 3(4), 330-344.

Rozenblit, J.W. 2001. "Systems Design: A Simulation-Based Modeling Framework." In Discrete Event Modeling and Simulation: A Tapestry of Systems and AI-based *Theories and Methodologies.* (Eds. F. Cellier and H. Sarjoughian), Springer Verlag, 107-127.

Schulz, S., Rozenblit, J.W., Mrva, M. and K. Buchenrieder. 1998. "Model-Based Codesign." *IEEE Computer*, 32(8), 60-68

Suantak, L., Momen, F., Rozenblit, J.W., Barnes, M. and T. Fichtl. 2001. "Intelligent Decision Support of Support and Stability Operations (SASO) through Symbolic Visualization." In *Proceedings of the 2001 IEEE International Conference on Systems, Man, and Cybernetics*, 2927-2931.

Zeigler, B.P. 1984. "Multifacetted Modeling and Discrete Event Simulation." Academic Press, 1984.

BIOGRAPHY

Jerzy W. Rozenblit is Professor of Electrical and Computer Engineering at The University of Arizona, Tucson. He holds the PhD and MS degrees in Computer Science from Wayne State University, Michigan and the MSc in Computer Engineering from the Technical University of Wroclaw, Poland. His research and teaching are in the areas of complex systems design and simulation modeling. His research in design has been supported by the National Science Foundation, Siemens AG, Semiconductor Research Corporation, McDonnell Douglas, and the US Army Research Laboratories.

Dr. Rozenblit serves as Associate Editor of ACM Transactions on Modeling and Computer Simulation, Associate Editor of IEEE Transactions on Systems, Man and Cybernetics, and Executive Board Member of IEEE Technical Committee on Engineering of Computer Based Systems. He was Fulbright Senior Scholar and Visiting Professor at the Institute of Systems Science, Johannes Kepler University, Austria and has held visiting professorship appointments at the Technical University of Munich, Central Research Laboratories of Siemens AG, and Infineon Technologies AG, in Munich. His e-mail is <jr@ece.arizona.edu>.