

THE AUTOMOD PRODUCT SUITE TUTORIAL

Brian Stanley

Division Headquarters:
Brooks Automation, AutoSimulations Division
655 Medical Drive
Bountiful, UT 84010 U.S.A.

ABSTRACT

Whether designing a new system or modifying an existing one, engineers want to take the guesswork out of finding the best possible solution. While there are many analysis methods for designing industrial systems, simulation remains the method that provides the highest level of confidence that a system will perform. A well-written simulation model can be a valuable tool in the design, analysis, and operation of manufacturing and other complex systems. The *AutoMod* Product Suite from Brooks Automation, AutoSimulations Division has been used on 1000s of projects to help engineers and managers make the best decisions possible.

1 BACKGROUND

The *AutoMod* Product Suite combines the ease of use of a simulator-type tool with the power and flexibility of a simulation language. As shown in Figure 1, 3-D visualization has been an integral part of the *AutoMod* product since its inception in 1984.

The main focus of the *AutoMod* simulation software is on manufacturing and material handling systems. *AutoMod* is flexible enough, though, to use in other non-manufacturing applications, for example, fast food restaurants, rental car lots, airport ticket counters, and container loading in ports.

With the release of *AutoMod* version 10.0 in 2001, graphics in *AutoMod* are even more realistic than before. *AutoMod's* new VR Graphics allows modelers to create visually accurate 3D scenes in their models, making them more convincing. VR Graphics in *AutoMod* allows VRML and Open Inventor format files to be imported for entities in the model. Additional graphical enhancements include templates for conveyors and path movers that provide realistic graphics for widely used material handling systems.

The *AutoMod* Product Suite includes:

- AutoMod – discrete and continuous simulation
- Simulator – manufacturing-oriented interface for fast and easy modeling
- AutoStat – statistical analysis including optimization
- AutoView – dynamic walk-through of animation with AVI support
- Model Communications Module (MCM) – protocols for linking to third party software, notably for PLC emulation

This tutorial gives an overview of the *AutoMod* product Suite, explaining the technical details of the products and how they interrelate to each other.

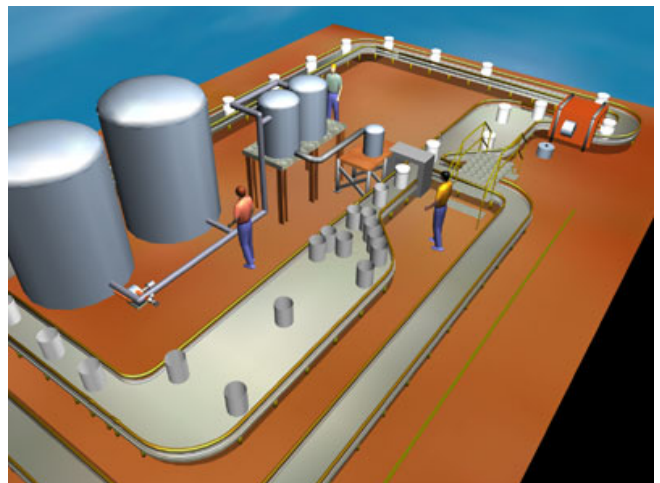


Figure 1: *AutoMod* VR Graphics in a Filling Operation

2 INTRODUCTION

AutoMod's strength is that it combines the ease of use of a “simulator” with the power and flexibility of a simulation

language. Simulators usually address specific real-world problems with predefined constructs, making model definition quick and easy. *AutoMod's* movement systems aid users in defining the movement of material either manually or by automated equipment. Material movement systems include:

- Path Mover (path/vehicle systems such as lift trucks, AGVS, and human movers)
- Conveyors (including belt and roller types)
- Automated Storage and Retrieval Systems (ASRS)
- Robots
- Bridge Cranes
- Power and Free Chain Conveyors
- Tanks and Pipes

To define material movement systems, the user simply defines movement elements, such as paths and stations, and then inputs the operating parameters, such as velocity and acceleration. *AutoMod* then automatically creates the corresponding model logic. 3-D animation is created automatically as well, providing a realistic picture of how a facility will look and operate. Model animation can be viewed from any angle or perspective in real time, providing visualization capabilities unmatched in other simulation tools.

AutoMod's key strengths are:

- Flexibility to model complex systems accurately
- Unlimited size of models
- High performance simulation engine
- Best in class statistical analysis features
- Graphic environment for creation of geometry
- True to scale import from CAD tools

The animation provided in *AutoMod* models is concurrent, meaning that the graphic pictures are running real-time with the simulation model. The model execution environment is very interactive, allowing the user to stop and start the simulation, run without animation in an accelerated time scale, and select objects from the animation screen to gather detailed statistics about the system being modeled. Statistics can be viewed at any time during a simulation run. These model execution features make it easier to verify and validate models of complex systems and provide a forum for communication about system performance among project team members.

3 AUTOMOD INTERFACE

An *AutoMod* model consists of one or more systems. A system can either be a process system, in which flow and con-

trol logic are defined, or a material movement system. Each model must contain one process system and may contain any number of movement systems. Processes can contain complex logic to control the flow of either manufacturing materials or control messages, to contend for resources, or to wait for user-specified times. Loads can move between processes with or without using movement systems.

All inter-arrival and event times can be represented by deterministic values or be derived randomly from one of many statistical distributions. *AutoMod's* interface is window-oriented, utilizing pop-up and pull-down menus, dialog boxes, selection lists, and an editor for developing process logic.

4 AUTOMOD'S WORLD VIEW

Any number of movement systems can be defined in an *AutoMod* model, and a process system connects the movement systems to the logical flow of products. In the process system, loads (products, parts, etc.) move between processes (locations) and compete for resources (equipment, operators, and queues). The load is the active entity, executing action statements that are connected to the processes. Typical action statements give users the ability to:

- Use machines/operators
- Move into queues
- Clone new loads
- Change load types
- Wait on user-defined delay lists
- Increment/decrement counters
- Set variable values
- Read from data files
- Send to other processes
- Make conditional tests

Figure 2 shows the process system pallet for *AutoMod*. The pallet is organized in a "top-down" manner with the most commonly defined elements at the top.

4.1 Processes

The process system is the backbone of an *AutoMod* model, providing the general-purpose simulation features required for modeling a wide range of real-world problems. While material movement is important, it is not always critical in manufacturing. No value is added to product while it is being moved around a manufacturing facility. Machines and people perform value-added operations. The *AutoMod* process system is where the value-added operations and control logic are defined.



Figure 2: AutoMod's Process System Pallet

AutoMod's process system includes a simulation language based on action statements. Action statements combine the power of a structured language with the ease of use of English-like, manufacturing-oriented syntax. *AutoMod* models are not limited in any way, so model logic can be of any size and complexity. The power and flexibility of the *AutoMod* language makes it easy to model almost any real-world situation.

Processes in *AutoMod* are the places where actions are performed or decisions are made. For example, an inspection operation could be represented as a process in *AutoMod*. From the final assembly process, parts would be "sent" to the inspection process, where an inspector (resource) would look at the parts. Processes can have a physical location, but it is not a requirement.

4.2 Loads

Loads are the active entities in *AutoMod* and can be created in many ways, including deterministic or probabilistic generation. The inter-arrival rate for loads can be read from an external data file or attached to a statistical distribution. *AutoMod* has predefined random distributions that can be used to fit most real-world random events. Loads are

given types, such as "RedCar" or "PartA," and they can have attributes such as color, stock keeping unit (SKU), priority, and time in the system. Attributes can be accessed and modified in *AutoMod's* action statements. Loads have 3-D shapes and dimensions like most other entities in *AutoMod*. Loads can also be changed on the fly in *AutoMod* through actions like:

```
scale by x 2
/*scales load in x by 2 times */

set color to brown
/*sets load color to brown*/
```

4.3 Resources

Resources in *AutoMod* are used to represent machines, operators, fixtures, containers, and any other finite capacity objects. There are two default categories for a resource's state. The first is the working category (busy or idle), and the second is the availability category (up or down). During the animation, state colors indicate the status of each resource. Statistics are automatically collected for every resource in a model.

Loads use resources for specified processing times. These times can be deterministic or probabilistic, using *AutoMod's* built-in statistical distributions to simulate randomness. Processing times are either part specific or are applied to all parts using a particular resource. Resources can also be preempted if a higher priority load needs immediate attention.

Resources can have downtimes, which are defined by MTTF (mean time to fail) and MTTR (mean time to repair). These times can use *AutoMod's* statistical distributions to represent random failures. Though MTTF is calculated by model-simulated time by default, it can also be based on parts processed or machine running time.

Resource cycles can be created and attached to specific resources to indicate when or how often events such as random failures or PM's occur for a resource.

4.4 States

The user can define states other than the default states. These states might be used to represent conditions such as blocked, starved, PM, offline, etc. The state of a resource can then be changed using *AutoMod's* actions so that statistics can be tracked for each state. In addition, state monitors can be defined and used to track states for entities other than resources, such as vehicles, conveyors, or particular areas of a facility.

4.5 Queues and Order Lists

Queues in *AutoMod* are both graphical and statistical elements, and they can have any user-defined capacity. When a queue has reached its capacity, the next load trying to enter that queue must wait until there is space available. Queue contents can be shown dynamically in the animation, and loads can be stacked in any direction in the queue.

Loads that are at a queue or process may be sorted or delayed until they are explicitly ordered to leave. To determine which action should take place next, the loads must place themselves on order lists. An order list is not a physical entity like a queue, but a logical element that provides a way to sort loads that have been delayed for any reason. To remove a load from an order list, another load must execute an order action. Loads can be ordered to move to another process or order list, or to simply continue where they left off in their processing. Order lists can be sorted by load priority or other load attributes, either in ascending or descending order.

4.6 Blocks

Blocks control the number of entities occupying a physical space, making them very useful for controlling path mover vehicles. Blocks may have any capacity from one to infinity. Path mover vehicles (lift trucks, AGVs, Electrified Monorails, etc.) and loads increment blocks automatically when moving through the physical space defined by the block. Loads can also claim blocks in process logic, as directed by the user. Blocks can have any shape, including combinations of the *AutoMod*-supported shapes (discussed in section 5).

4.7 Variables and Counters

Data may be stored in an *AutoMod* model using variables. Variable values are changed using the “set” action as follows:

```
set Setup_time to 123.456
```

Variables can be used in calculations or can be compared to other variables. In addition to integer, real, and string types, variables can also be used to store references to other process system entities, such as processes, queues, resources, order lists, counters, loads, and locations. Storing these references gives the users more power and flexibility to expand and extend models to match modifications

of the actual system. *AutoMod* also supports the notion of arrayed entities, making it easier to model real-world systems that have some dimensionality, for example a warehouse with similar operations on two floors.

Counters are similar to variables; except they can have finite capacity and they can only be positive integer values. Counters have a maximum capacity, making them useful for traffic control. When a load tries to increment a counter that is at its capacity, the load will be delayed until another load decrements the counter. Statistics for counters are collected automatically.

4.8 Functions, Subroutines, and Source Files

Functions and subroutines are used to create modular models. This allows the models to be extended more easily. Users can define their own functions in the *AutoMod* or C languages, and these functions can be called from anywhere in the *AutoMod* model. Subroutines help eliminate duplication of action statements in the model.

Source files contain the logic for the model. Any number of source files may be defined, and they are processed differently depending on their file extension. For example, “.m” files contain *AutoMod* logic and will be checked by *AutoMod* for correctness when edited. Files with a “.c” extension should contain C language and will be compiled with the model. Users may also define other source files to contain input data or documentation for the model.

4.9 Labels

AutoMod provides the ability to use text labels to enhance model understanding. Labels may be added to any location in the model’s physical space, and they can either be static or dynamic during the simulation. Labels can rotate when the animation view changes or can be attached to a fixed position on the screen.

4.9.1 Tables

Tables in *AutoMod* supply the user with the means to collect statistics on any model entity and to classify those statistics for better understanding of their distribution. Tables automatically provide average, standard deviation, maximum, and minimum for all values entered in the table. Users can define the number of “bins” or categories and *AutoMod* adds values to the appropriate bin when directed by the user in the tabulate action statement.

4.10 Types and Random Streams

Types in *AutoMod* provide users with a powerful tool for creating, among other things, lists of entities such as resources or stations, and then making complex decisions based on the contents of the lists. Some of the default types are integer, real, resource, location, etc. But with user-defined types, it is possible to create, for example, a variable of type *ResourceList*, whose data can then be manipulated in any number of ways to facilitate complex decision-making.

Random streams may be defined to give each element of randomness independence from other elements. The number of different random streams used by *AutoMod* models is without limit. The current version of the software (10.0) supports the Linear Congruential Generator (LCG). Commencing with the upcoming maintenance release (Version 10.5), *AutoMod* will use the Combined Multiple Recursive Generator described by L'Ecuyer (1999).

4.11 Run Control

Run control in *AutoMod* allows users to define the warm-up and steady-state periods for the model by resetting time-persistent statistics. Reports can be printed for any run control period, or “snap.” Business graph output can be automatically created. Post-processed animation periods used by *AutoView*, an animation extension (discussed in section 4), are also defined in the run control. Run control also provides an entity tracing capability that gives the model builder an event-by-event account of the model run. This information is useful in verifying and validating a model.

4.12 Business Graphics

Graphs in *AutoMod* are easy to define and they update in real time with the animation. Graph types include bar charts, pie charts, and timelines. Figure 3 shows a typical timeline business graph. Any model entity can be attached to a graph, including transporter vehicle velocity, number of loads on a conveyor section, or average utilization of a machine. Graphs can be printed or plotted to a variety of supported output devices, and graph displays can be controlled using the *AutoMod* language.

5 AUTOVIEW

AutoSimulations' post-processed animation extension can be used to view the animation records created by running an *AutoMod* model. The *AutoView* product allows the user to pre-define all views and time periods in the model animation, and then play them back to generate presentation-quality animation files.

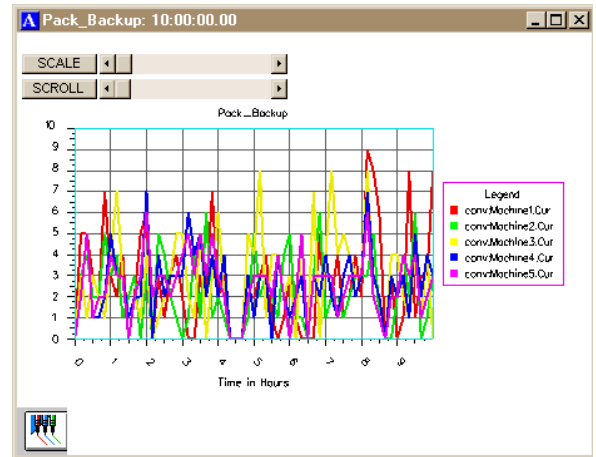


Figure 3: AutoMod Timeline Business Graph

AutoView is a free program, so users can create detailed animation walk through graphics and can distribute the *AutoView* files freely to their customers.

AutoView also has single-frame capture capability to AVI and MPEG format files. This feature allows users to create animations that can be shared with others without the need for any additional software. Also, the single-frame capability helps smooth the animation on larger models where hardware constraints may affect animation performance.

6 GRAPHICS IN 3-D

Both dynamic and static objects can be displayed during model execution. Figure 1 shows a screen shot of a typical *AutoMod* model during a model run. Dynamic objects represent loads, vehicles, resources, queues, and statistics. The static layout is the background graphics of the plant, such as columns, aisle markings, and walls. Labels can identify specific areas in the facility.

There are several ways to create a layout of the system to be modeled. *AutoMod* comes with a three-dimensional graphics editor that allows the user to construct objects from standard graphics primitives. Cone, Box, Hemisphere, Trapezoid, Frustum, Cylinder, Arc, Vector (list), Set, Text, and Triad are primitives that can be selected, placed, and scaled to create any static entity in the facility.

AutoMod supports the true-to-scale import of CAD format files under the IGES standard.

Additionally, with the release of version 10.0, Virtual Reality Modeling Language (VRML) and Open Inventor graphic format files can be imported directly into *AutoMod*. Most CAD systems and other solid geometry creation programs support VRML, and there are many translators available that can convert from other file formats, like IGES, STEP, and DXF, to VRML. VRML format files can be included as either static or dynamic shapes in the *AutoMod* model.

7 RUN-TIME ENVIRONMENT

In keeping with *AutoMod's* interactive features, the user has complete control of the model in the run-time environment. The model can be viewed with the animation on or run with the animation off. *AutoMod* uses concurrent animation; the simulation progresses as the animation picture is being updated. With animation off, the simulation does not render the animation, but it performs all simulation calculations. The user can suspend the simulation at any instant to review statistics through pop-up windows, to take resources down, to set break points or alarms, or to control the view of the animation without constraint.

7.1 View Control

AutoMod provides a comprehensive, flexible, and easy-to-use method of interacting with a model during model execution. If the simulation project is in the experimentation phase where only parameter changes are made and the model needs to be re-run several times *AutoMod* provides the ability to run in batch mode without animation.

7.2 User Interaction

AutoMod provides advanced debugging and trace facilities. A model can be single-stepped at any time during the animation. Also, the ability to set breakpoints and alarms allows the user to suspend the simulation when a certain event occurs or when a specific clock time is reached.

AutoMod also provides comprehensive reports. The reports can be displayed on request at any time during the simulation. Printed versions of the reports can also be specified in the Model Development Environment. Reports are configurable by the user, who can choose between standard, full, and no report for all *AutoMod* entities.

AutoMod automatically keeps track of many statistics. These automatic reports are linked to specific entity types, such as:

- Movement systems
- Processes
- Queues
- Resources
- Order Lists, etc.

Vehicle states are tracked during the entire model run, and reports are generated automatically. Reports can be sorted alphabetically or numerically for easier analysis. The user can also develop and generate custom reports from within process procedures.

8 AUTOMOD'S SIMULATOR INTERFACE

In addition to *AutoMod's* standard interface already described, *AutoMod* also includes a manufacturing template called the Simulator. The Simulator can be used to model problems where:

- Machines are grouped by capability. For example, in a machine shop one might have sets of lathes, mills, and drill presses that can perform the same operation.
- Parts have complex routings, including alternate steps, setups, and batching.
- Product is manufactured to customer order.

Figure 4 shows the main menu for the Simulator.



Figure 4: Simulator Main Menu

8.1 Factory Resources

Factory resources include the equipment, operators, fixtures, and any other item required for the manufacturing of products. Machines are grouped into families that can perform similar operations. When a machine finishes the current operation, it looks at all the products available to process, and makes a decision about the next thing to do. This decision is called a scheduling rule. The Simulator allows the users to select from among many standard rules, like first-in-first-out, highest priority, or same setup.

8.2 Products

Products are the items that are being produced in the Simulator model. Products have a routing that defines the operations required to manufacture them. The routing defines families of machines to visit, and can also include alternated steps, setup requirements, processing time, and batching requirements. A bill of materials can also be defined for sub-assemblies, which can be either produced or purchased.

8.3 Demand

Demand in the Simulator represents the customer orders. In manufacturing, many decisions need to be made, including how to allocate resources, what each operation should do next, and when to “launch” new orders into the facility. The demand file in the Simulator defines each customer order, start date and time, and expected due date. Reports from the simulator indicate which orders met their due dates and which were late.

The Simulator in AutoMod provides a quick and easy way to define models of a certain class of manufacturing problem. Most Simulator models can be created without writing any code. The spreadsheet interface to the simulator is intuitive and very easy to use. Data can be imported directly from other data sources, like an MRP or ERP system, to facilitate even faster model building.

9 AUTOSTAT

The AutoStat product is the statistical analysis tool in the *AutoMod* product family. AutoStat does extensive output analysis on an *AutoMod* model, including:

- Confidence intervals
- Warm-up determination
- Sensitivity analysis
- Factor-response analysis
- Design of experiments
- Optimization using evolution strategies

AutoStat also has support for running scenarios across a network of machines. Support for multiple machines and CPUs give users the ability to make many more runs of the simulation than was possible before. Figure 5 shows an example graph from AutoStat.

The evolution strategies algorithm used by AutoStat is well suited to finding the optimum solution without getting “trapped” at a local optimal value. Figure 6 shows the output from an optimization run. Across the X axis are the number of generations required to find the optimum, and the Y axis gives the values for the user-defined fitness function. The fitness function can be defined as any com-

bination of model inputs and outputs, with user-defined weighting factors applied to each value.

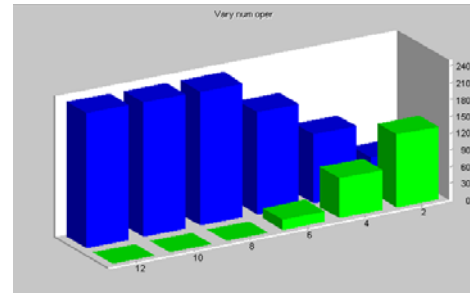


Figure 5: Example Graph From Auto Stat

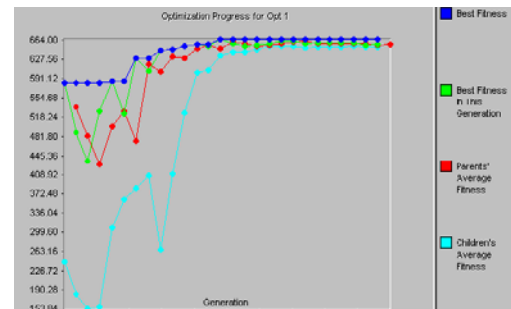


Figure 6: Optimization Run Graph

10 MODEL COMMUNICATIONS MODULE

The Model Communications Module (MCM) is an extension to *AutoMod* that allows simulation models to communicate with other programs locally or on a network of machines. Some of the applications of MCM include communications between:

- Two or more *AutoMod* models
- Control systems and *AutoMod* models
- Other applications and *AutoMod* models

Communication between two or more models allows for parallel and distributed simulation. MCM also includes Multi-Model Sync (MMS), which keeps the event lists of the linked models synchronized.

The fastest growing application of communications is emulation, where an *AutoMod* model is used to replace a real AMHS in order to test and debug an industrial control system. This allows control system designers to test the control logic using the simulation prior to going into the field, which can result in considerable savings in terms of time spent on site performing routine logic testing and debugging. Much on site work involves waiting for problems to arise during ramp, followed by a difficult and slow investigation into the origins of the incident. The resulting logic modification is then hard to verify under a range of conditions without major disruption to production. Emulation provides a re-

liable way of verifying control code functionality offline, training operators in a safe environment, and of testing modifications to a control system before they get put into effect. Furthermore, the emulation model serves as a testbed for any further control system modifications throughout the life of the system, so production is not disrupted.

In order to facilitate emulation and other linked-application requirements, MCM enables sockets, Object Linking and Embedding for Process Control (OPC) and Dynamic Data Exchange (DDE) technologies. Connecting across different platforms is achieved using sockets technology, which allows communication via strings or C structures. OPC is specifically designed to facilitate the convenient and efficient connection of Programmable Logic Controllers (PLCs) to other software through a standard protocol that is non-proprietary. Models may be easily linked to other applications using DDE. Examples of this include Excel spreadsheets, Access databases, ergonomics programs, as well as control programs.

11 TEACHING AUTOMOD

Teaching AutoMod has been greatly facilitated with the publication of the text *Getting Started with AutoMod* (Banks 2000). PowerPoint slides have been prepared to accompany the text. Additionally, the electronic files are available for all of the example models in the text. These are all downloadable from www.autosim.com as is the Student Version of AutoMod 9.1. Additionally, professors can receive a file with the solutions to all of the exercises in the text.

12 SUMMARY

AutoMod is a state-of-the-art simulation system that provides the ability to define the physical elements of a system using true-to-scale CAD-like graphics and the logical portion of the system using a powerful procedural language. The results are that a typical user can be three to ten times more productive using *AutoMod* in comparison to using any other simulation tool. The accuracy and degree of detail with respect to model development is unequalled.

AutoMod allows the construction of very large, complex models. In fact, *AutoMod* provides an architecture that has proven that the larger the project, the more benefits *AutoMod* provides over other simulation systems.

AutoMod provides realistic, 3-D visualization. There are no limits to the views or the size of the picture to be shown. The degree of animation realism is also unmatched, as *AutoMod* provides animation that facilitates faster model building and better communication of model results.

The *AutoMod* product suite, including the AutoStat, AutoView, and MCM extensions, provides a comprehensive solution to the needs of the simulation user. Through the use of AutoSimulations' technologies, models can be-

come more than design tools, being used to test controls, operate the facility, and plan for expansion.

REFERENCES

- AutoMod Beginning Class Notes*. 2001. Bountiful, UT: Brooks Automation, AutoSimulations Division.
- AutoMod User's Manual*. 2001. Bountiful, UT: Brooks Automation, AutoSimulations Division.
- Banks, J. 2000. *Getting Started with AutoMod*. Bountiful, UT: Brooks Automation, AutoSimulations Division.
- Bowden, R. O. and J. D. Hall. 1998. Simulation Optimization Research and Development. In *Proceedings of the 1998 Winter Simulation Conference*, ed. D. J. Medeiros, Edward F. Watson, John S. Carson, Mani S. Manivannan, 1693-1698. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers.
- L'Ecuyer, P. 1999. Good parameter sets for combined multiple recursive random number generators. *Oper. Res.* 47, 159-164.

AUTHOR BIOGRAPHY

BRIAN STANLEY, Eastern Regional Account Manager, joined Brooks Automation, AutoSimulations Division in 1998. Prior to Mr. Stanley's account management responsibilities, he completed simulation projects in distribution, manufacturing, and material handling. Mr. Stanley received a Bachelor of Industrial Engineering degree from Georgia Institute of Technology. His e-mail address is brian.stanley@brooks.com.