Proceedings of the 1993 Winter Simulation Conference
G.W. Evans, M. Mollaghasemi, E.C. Russell, W.E. Biles (eds.)

INTRODUCTION TO ARENA™

Norene Collins
Christine M. Watson

Systems Modeling Corporation
The Park Building
504 Beaver Street
Sewickley, PA 15143, U.S.A.

ABSTRACT

This paper presents an overview of Arena--an extendible modeling system that is built on SIMAN/Cinema. A key idea behind Arena is the concept of tailorability to a specific application domain through the use of a template.

1 INTRODUCTION

During the past decade, there has been tremendous progress in the development of new simulation technology and related software. Most modern simulation tools now provide an interactive graphical interface for model definition as well as real-time graphical animation. These new capabilities represent a significant improvement over earlier nongraphical, batch-oriented modeling tools.

The many recent advances in simulation technology have created a greater awareness and use of simulation by industry. Managers are now more aware of the potential benefits of simulation. However, even with the many important advances that have been made over the past several years, there are still many cases where complex systems are being designed and implemented without the benefit of simulation. In our view, a very small percentage of systems that could benefit from simulation are actually simulated, and the primary reason for this is the high level of effort required to employ simulation technology successfully. We believe that the key to making simulation technology more widely used is to make the tools significantly easier to learn and use.

This paper describes a SIMAN/Cinema-based modeling/animation system, named Arena, developed by Systems Modeling Corporation. In our view, this system represents a major advance in simulation technology by combining the modeling power and flexibility of the SIMAN/Cinema system with the ease-of-use of application-focused packages. Arena offers a high level of modeling flexibility across a wide range of problem domains, yet is very simple to learn and use.

Arena also provides a complete simulation environment that supports all basic steps in a simulation study. The Arena system includes integrated support for input data analysis, model building, interactive execution, animation, execution tracing and verification, and output analysis. In this paper, we restrict our focus to the model building functionality of Arena.

The key idea behind Arena is the concept of tailorability to a specific application area. The Arena system is not restricted to a specific set of predefined modeling primitives, but can be easily tailored to a domain-specific application area by means of an application template. Hence a user of Arena who is modeling health care systems could employ a health care template that would contain modeling primitives focused on health care systems (doctors, nurses, beds, X-ray, etc.). Likewise, a person modeling traffic flow in a city could employ Arena with a traffic flow template that would contain modeling primitives focused on traffic flow (roads, exit ramps, cloverleafs, stoplights, etc.). Unlike conventional simulation systems that have their modeling primitives "hard-coded" into the software by the vendor, Arena allows the modeling primitives to be "soft-coded" by the end user by means of the template.

The application template concept is fundamental to the flexibility and ease-of-use provided by Arena. This mechanism makes it possible to provide the end user of the product with a tool that closely matches the real system being modeled--hence the user is presented with concepts and terminology that are focused on his or her problem. This dramatically reduces the level of modeling abstraction required by the user. However, the user is not limited to the constructs provided by a single domain-restricted construct set. The user can combine domain-restricted constructs from one or more application-focused templates with the full modeling power of the SIMAN simulation language and thereby avoid the

modeling "brick wall" encountered with traditional hard-coded, domain-restricted packages. The modeling power of SIMAN is made available to the user as simply one additional template.

Arena is a graphical modeling/animation system that is based on concepts from object-oriented programming and hierarchical modeling. A discussion of these concepts and their relationship to simulation languages and modeling can be found in the 1992 Winter Simulation Conference Proceedings [Pegden, 1992].

2 ARENA BASICS

Since the Arena system blends together concepts from SIMAN/Cinema, object-oriented programming, hierarchical modular modeling, as well as a number of new concepts, it is helpful to define some basic terminology used to discuss the hierarchical modeling concepts in Arena.

2.1 Main Menu Bar and Work Area

The first thing you will see when you start Arena is the *Main Menu Bar*. It is from here that all windows within Arena are accessed. The space underneath the main menu bar is the *Work Area*. All windows are displayed in the work area.

2.2 The Model Window

All model building and model execution takes place in a special window in Arena called the *Model Window*. Like all windows in Arena, the model window can be moved, resized, enlarged to full screen, or minimized to an icon. You can also have multiple model windows open in the work area at the same time.

The model window is where new models are created, existing models are modified, animations are developed and models are executed. To open a new model window, you click on *Model* in the main menu bar, and then click on *New* in the pop-down menu.

The model window is divided into three main regions: the bar region, the panel region, and the workspace region.

2.3 Objects

Objects in Arena are components that make up your simulation model and animation. These objects include static graphics such as lines, circles, and text; animation components; and application modules. Objects are added to the model window via the panels.

2.4 Panels and Panel Tabs

The *Panel Tabs* located to the left of a model window resemble file folder tabs with each one representing a different group of application functions, referred to as *Panels*. To select a panel, click on the desired tab.

All Arena model windows contain a Run panel, a Draw panel, and an Animate panel. The *Run* panel is used to execute your model. The *Draw* panel permits you to add static graphics to your model window. The *Animate* panel is used to add Cinema V animation objects to your model. All other panels are used to attach application-focused templates.

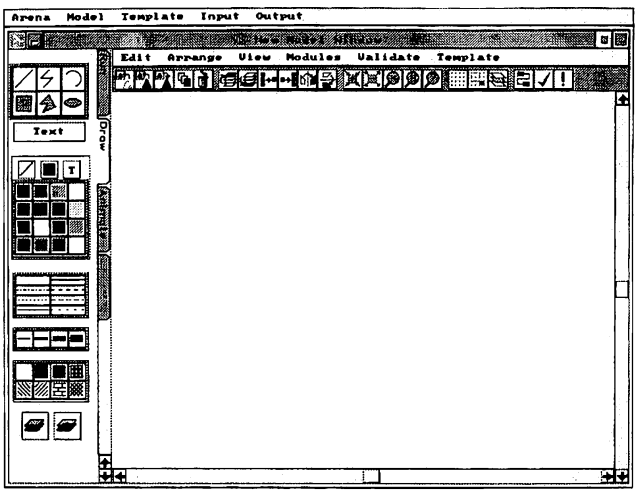

Figure 1 New Model Window

2.5 Modules and Templates

You build a model in Arena by placing and interconnecting *Modules* in the workspace. A module is simply a modeling construct that you use to represent some component of a system. Although the appearance and function of modules may differ substantially, all modules share some basic underlying constructs. The modules that you use for modeling an emergency room in a hospital may look and function quite differently from the modules that you might use to model a computer network; none-the-less the methods that you use for selecting, placing, interconnecting, and entering associated data remains the same across these modules.

Modules are organized into useful groups called *Templates* (or Application-Specific Templates, ASTs). The modules in a template are stored in one or more template panel files. All template panel files have a file extension .tpo, and these may be attached to the empty panels in the model window. To attach a new template panel file to your model window, click on a blank panel tab.

2.6 The SIMAN Template

The SIMAN Template contains the base modules that correspond directly to the SIMAN blocks/elements. The SIMAN Template is made up of two template panel files: blocks.tpo and elements.tpo. Figure 2 illustrates a model built using modules from the SIMAN Template.

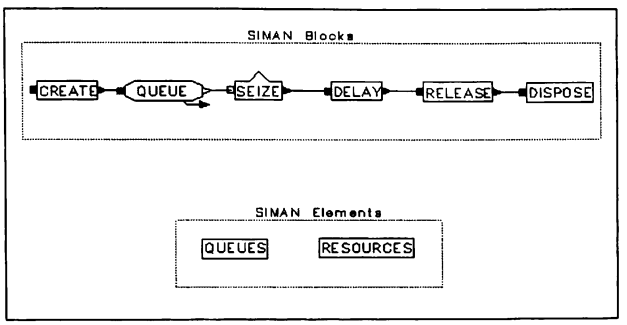

Figure 2 SIMAN Model

2.7 Elements and Properties

The term SIMAN *Element* is used to denote the constructs used by SIMAN to represent physical objects in the real system. Examples of SIMAN elements include Resources, Conveyors, Transporters, Queues, and Variables. There is a one-to-one correspondence between a base module in the elements.tpo template panel file, the elements in the conventional SIMAN experiment, and the objects being modeled in the real system.

The characteristics of an element are referred to as its *Properties*. For example, the speed, acceleration, deceleration, and turning velocity factor are all properties of a transporter element. In the elements.tpo template panel file, the properties of the base modules correspond to the fields defining each element in the SIMAN experiment.

3 MODULE BASICS

An Arena user builds a model by placing and interconnecting modules in the workspace of a model window. For example, a model of a simple service system might be built by interconnecting modules from a general-purpose modeling template representing the arrival process, service process, and departure process. A model of an emergency room might be built using modules from a health care template representing the reception/admission process, X-ray, and so forth. The modeler may freely mix modules from multiple application templates--including the base SIMAN template.

3.1 Base and Derived Modules

There are two basic types of modules: base modules and derived modules. *Base* modules are the lowest level modules in Arena and correspond directly to the SIMAN blocks and elements. Hence a QUEUE module in the SIMAN Template (blocks.tpo panel file) corresponds directly to the QUEUE block in the conventional SIMAN model file. Likewise, a QUEUES module in the SIMAN Template (elements.tpo panel file) corresponds directly to the QUEUES element in the conventional SIMAN experiment file.

Derived modules are built up from one or more base and/or derived modules. To illustrate the concept of a derived module, consider the SIMAN block sequence QUEUE-SEIZE-DELAY-RELEASE shown in Figure 3. In Arena, we could build a module named Serve to represent this sequence of four SIMAN blocks. The corresponding elements, QUEUES and RESOURCES, may also be included in the Serve module to define the appropriate objects. Note that the Serve module can then be used in building additional derived modules.

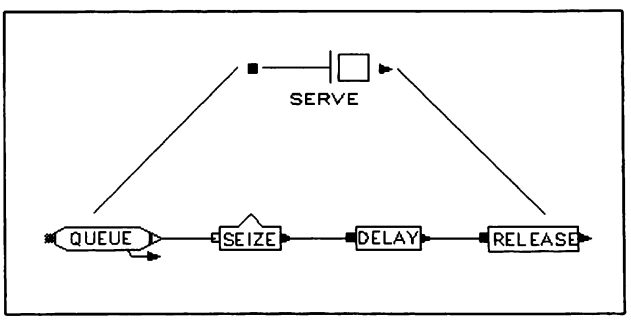

Figure 3 The Serve Module

3.2 Operands

A module has operands that define values associated with the module. For example, the Serve module might have operands defining the queuing discipline, server name, and processing time. The module creator defines the characteristics of each operand including the positioning in the dialog box, user prompt, default value, permissible values, and user interface method (text boxes, toggles, selection buttons, etc.). The user may view/edit the operands of a module by double-clicking the mouse on the module in the layout--this causes the dialog box for the module to appear.

The operands of a derived module may supply values for operands of its component modules from which it is constructed. For example, the text string supplied for the server name could be used to supply part or all of the queue identifier in the QUEUE module and/or the

resource identifier in the SEIZE and RELEASE modules. Note that operand values are not inherited up from below--but are instead defined at higher-level modules and then passed down to lower-level modules. This is analogous to the passing of arguments from a main program down to a function in normal programming.

3.3 Dialog Boxes

Dialog Boxes are forms that display parameter choices for you and solicit and accept input from you. Six of the different controls in the Arena User Interface are shown in Figure 4. Selection buttons are small diamond-shaped buttons that allow you to choose a single value for a parameter that has many choices. Toggle buttons permit you to alternate between a yes and no answer. Text boxes are used to enter and edit text strings. With a text and list box you can either enter a value in the text box or click on the down arrow to reveal a pull down scrollable list of available options. A repeat group allows you to insert, change, and delete entries within a scroll list. Action buttons permit you to perform a function.

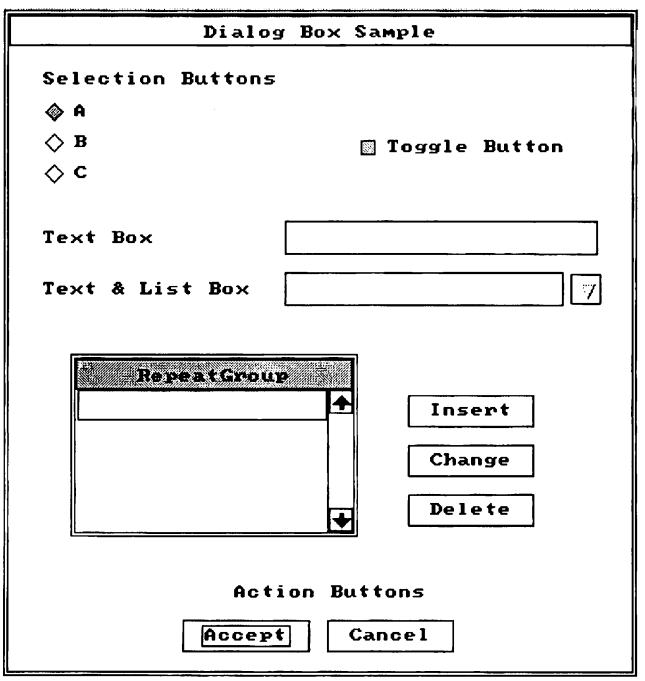

Figure 4 User Interface Controls

3.4 Entry/Exit Points and Connectors

Entities in a model move from module to module and activate functions that act upon the SIMAN elements and thereby change the state of the system. For example, an entity moving through the Serve module may change the state of the queue and resource elements referenced by the module. Entities enter modules at locations called *Entry Points* and exit modules at locations called *Exit Points*. The exit point of one module may be connected to the entry point of a second module by means of a graphical connector. An entity passes through a connector in zero simulated time.

3.5 Derived Modules

A derived module is created by:

1) Defining the operands for the module,
2) Building a submodel from existing base/derived modules to represent the logic of the module,
3) Drawing a panel icon to represent the module graphically in the panel, and
4) Defining the user view of the module.

A module's user view is the view of the module seen by the end user who places the module into the workspace of the model window. The user view can contain static components from the Draw panel as well as animation components from the Animate panel.

User views may also contain one or more entry/exit points. These entry/exit points are used by the end user of the module to interconnect the derived module with other modules in the layout. They may be placed in any position in the user view. Figure 5 shows the user view for the Arrive, Server, and Depart modules from the Common panel of the Arena Template. Figure 6 depicts the dialog box associated with the Server module.

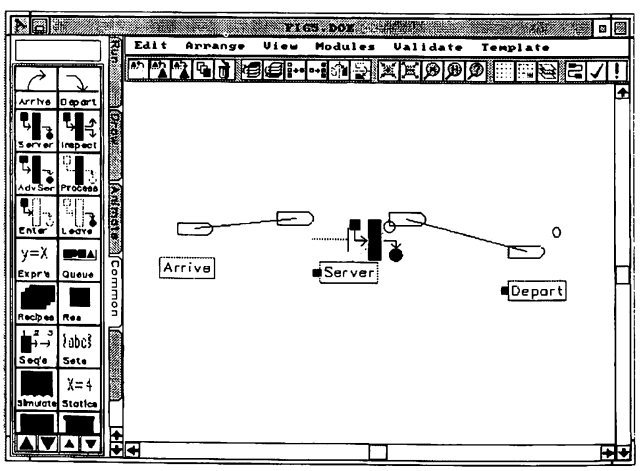

Figure 5 Arena Template Modules

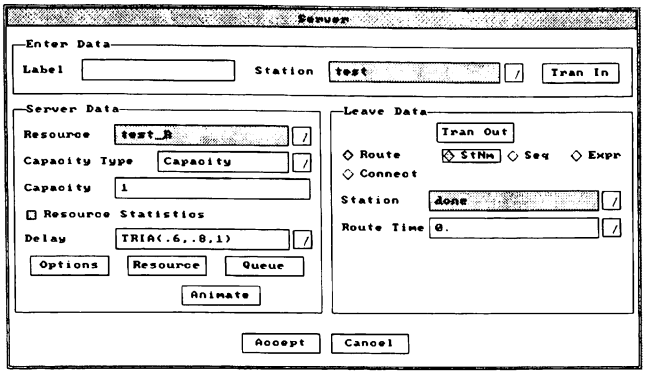

Figure 6 Server Dialog Box

3.6 Cinema Animation

One of the important features of Arena is the full integration of animation as a fundamental part of the system. Arena contains Cinema-based animation constructs including queues, levels, variables, resources, plots, histograms, transporters, conveyors, and paths. These constructs dynamically change position, shape, or color during the execution of the model.

Animation can be incorporated into an Arena model in two ways. First, if Cinema animation constructs are included in the user view of a derived module, then these animation constructs come along with the module when it is placed in the layout. Second, animation constructs can be freely added anywhere in the model window to embellish any animation that is included in the modules' user views.

4 TEMPLATES

The real power of Arena is realized through its support for application-specific templates (ASTs). Numerous application areas have been discussed for possible templates including computer systems, communication systems, traffic flow, airport design, fast food restaurants, high-speed packaging, health care, process industry, package sorting, warehousing, and many more. Each new application template brings simulation technology much closer to a large class of potential users.

In addition to general application areas, the opportunity exists for developing company-specific templates that contain modules focused at a specific company--thereby sharing the effort and expertise of a few people. For example, an automotive manufacturer could develop a template containing manufacturing equipment and/or workcenter designs specific to automotive assembly within the company. By tailoring the modeling primitives to a specific company, such templates can help make simulation technology a much simpler and more widely used methodology throughout the company.

We have already briefly discussed the SIMAN Template. This template contains the base modules that correspond directly to SIMAN blocks/elements. In this section, we will briefly discuss three other templates. The first is the Arena Template, which is a general modeling template that builds on SIMAN and provides useful higher-level modules across a broad range of applications. The second is a template for modeling general manufacturing processes. The third is a template for modeling semiconductor wafer fabrication.

4.1 The Arena Template

The Arena template contains derived modules that provide general-purpose constructs that are at a higher modeling level than the standard SIMAN blocks/elements. The objective of this template is to provide a comprehensive set of high-level modeling constructs for modeling simple systems across a broad range of applications.

For the new user, the Arena template provides an easier entree into modeling using Arena. By providing the modeler with higher-level modules such as Arrive, Server, and Depart, the new user can build simple models with less modeling abstraction--and therefore, with greater ease and less learning. The new user can then transition into using the base SIMAN primitives as necessary to expand on the modeling flexibility of the Arena Template.

For the experienced user, the Arena Template allows the user to leverage the predefined modules in the Arena Template to develop large models more rapidly. Since a single module in the Arena Template typically corresponds to several base SIMAN Template modules, large models can be built more rapidly and with fewer errors. The modules in the Arena Template can be mixed with SIMAN Template modules to provide a detailed modeling capability for complex systems.

The Arena Template is made up of three panel files; the Common panel, the Support panel, and the Transfer panel.

4.1.1 The Common Panel

The Common panel contains two types of modules, logic modules and data modules. Logic modules are used to describe the logical process flow of the system, while data modules describe individual elements of the system.

The Common panel's eight logic modules are derived modules that can be used to model common processes such as entity arrival, service, and departure. These modules also create most, if not all, of the elements

corresponding to the logic module. The logic modules in the Common panel are:

Arrive	Depart
Server	Inspect
AdvServer	Process
Enter	Leave

As the name implies, these modules are the more "Common" modules used when modeling with the ARENA Template. For many applications, the Common panel logic modules, with some assistance from the Common panel's data modules, include sufficient functionality and flexibility to describe the system being modeled. You may choose to only use modules from the Common panel when building your simulation model.

The Common panel's eight logic modules are all based around the concept of an entity entering a station (the Enter module), an entity being processed at a station (the Process module), and an entity leaving a station (the Leave module). Further, the functionality of the Enter, Process and Leave modules is combined into one module in the Advanced Server (AdvServer) module.

The Arrive module is used to create entities. The Depart module is used to collect statistics and dispose of entities. The Server and Inspect modules are scaled-down versions of the Advanced Server module, except that the Inspect module also determines whether an entity passes or fails inspection based on a user-defined probability.

4.1.2 The Support Panel

All of the modules in the Support panel are logic modules. While the logic modules in the Common panel consist of several functions combined to represent a higher-level process, the modules in the Support panel are a representation of very specific functions. These modules contain the individual components which make up the high level logic modules in the Common panel.

The SIMAN template contains two panels, the Blocks panel which contains logic modules, and the Elements panel which contains data modules. Support panel modules are similar to the modules in the Blocks panel of the SIMAN template except that they provide more functionality. In fact, many of the modules in these two templates have the same name, only with different capitalization. For example, the Support panel includes Create, Station, Seize, Delay and Release modules while the Blocks panel includes CREATE, STATION, SEIZE, DELAY and RELEASE modules.

4.1.3 The Transfer Panel

The Transfer panel contains modules used to describe the transfer of entities from one area of the system (a station) to another area. There are six data modules and approximately a dozen logic modules in the Transfer panel. Two of the six data modules are used to define transporters and conveyors, respectively. The remaining four modules are used to define data related to the path that a transferred entity follows.

All of the logic modules in the Transfer panel are similar to modules in the Blocks panel, and like the modules in the Support panel, the module names are the same. For example, the Transfer panel includes Request, Transport and Free modules, while the Blocks panel includes REQUEST, TRANSPORT and FREE modules. Modules in the Transfer panel provide more functionality than their counterparts in the Blocks panel.

4.2 The Manufacturing Template

The Manufacturing Template is a collection of modules that may be combined to describe the process flow of a manufacturing system. The template was designed to support the majority of discrete manufacturing applications; however, it is not limited to that application domain. Each of the modules within the template is made up of one or more SIMAN blocks and/or elements. The modules have been designed to allow for flexibility in modeling, while ensuring a user-friendly modeling environment.

The Manufacturing Template supports various types of process flow including unconstrained push, constrained push and pull. Unconstrained push may be utilized if there are no work-in-process (WIP) constraints. Constrained push is used when blocking occurs due to either physical space or WIP constraints. The use of production and transfer authorizations enable the template to provide just-in-time (JIT) type pulling capability. Any of these process flow methods or a combination of them may be used in a single simulation model.

Various types of workcenter processing activities are supported within the template. The processes supported include single part, production, assembly and batch processing. Single-part processing implies that one part enters a workcenter and the same part exits the workcenter upon completion of processing. Production-type processing enables one part to enter a workcenter and multiple parts of the same or different part types to exit the workcenter. Assembly operations are accomplished by combining all specified component parts into an assembled part. Finally, both temporary and permanent batching is available at the workcenters. If parts require batch processing, a number of parts can be grouped for the processing activities and unbatched following completion.

In incorporating the features of the SIMAN language into the template, material handling capability is strongly

supported. Both free path and guided transporters, as well as accumulating and nonaccumulating conveyors, can be used to move parts from one workcenter to the next. For less detailed control of material handling, operator resources or simple delays may be used for transfers.

The Manufacturing Template consists of three types of modules: workcenter modules, component modules, and data modules. Both the workcenter and component modules consist of the logic portion of a manufacturing system, including such functions as entering a workcenter, exiting material handling, processing and moving to the next workcenter. Workcenter and component modules support modeling at multiple levels of detail. Workcenter modules include sufficient functionality and flexibility to describe many manufacturing systems. Component modules are simply components of workcenter modules that allow a user to incorporate detailed logic to represent a system. Data modules allow the definition of information related to objects used in the workcenter and component modules, such as operators and parts.

Given that the Manufacturing Template consists of two levels of logic modules (workcenter and component modules), it supports both machine-based and jobstep-based modeling orientations. In a machine-based orientation, the logic defining what is to occur, though not necessarily all of the data, is specified in conjunction with the machine (or workcenter). Workcenter modules may be used to define processing logic and the default data associated with each workcenter. In a jobstep-based orientation, the logic defining what is to occur is specified with the jobstep itself. By utilizing component modules, the processing logic and data may be incorporated directly into the process plan for a part. Combinations of the two modeling orientations can be achieved by intermixing the levels of logic modules.

4.3 The Wafer Fabrication Template

The Wafer Fabrication Template is designed to support modeling of wafer fabrication operations in the semiconductor industry. (This template is based on prototype research performed by D. Phillips, G. Curry, and B. Deuermeyer at Texas A&M University, and it was developed with partial funding from SEMATECH. John Fowler managed the development of this template for SEMATECH and played a key role in the design of the template.) While not restricted to this application domain, the model structure and terminology used in this template are consistent with that found in wafer fabrication applications.

Models that are built using this template consists of two types of user input, model data and model rules. Model data definitions include products, technologies and jobsteps (process plans), recipes, workcenters, resources, operators, and production schedules. Model rules define the model logic to be used by a particular workcenter or recipe--i.e., model rules define the logic by which manufacturing lots seize and release sets of resources, are batched and split apart, and undergo processing delays. Given the large number of jobsteps in a typical wafer fabrication model, model rules are defined separately from the model data so that they may be re-used.

A unique aspect of the Wafer Fabrication Template is that is it can generate either a standard simulation model or a special model that is used to perform analytical flow and queue analysis of the system. Either of these models can be generated from the same user description of the system--i.e., the user builds a single model of the system and then selects which form of the model (standard or flow and queue) they would like to execute. Flow and queue analysis, as the name implies, involves performing two types of analysis: a flow analysis and a queuing analysis. First, flow analysis is performed to determine the rate at which products (manufacturing lots of wafers) flow through individual components of the system. Second, based on the calculated flow rates, an interactive queuing analysis is used to calculate resource utilizations and queue times.

The flow and queue analysis provides the same basic performance measures for the system as does the standard simulation model, but uses queuing approximation formulas to obtain the results. The advantage of the flow and queue analysis is that it executes much faster than a simulation model of the same system. The advantage of the simulation model is that it provides more accurate results.

5 SUMMARY

Arena is a new hierarchical modeling system based on SIMAN and Cinema. The key feature of this system is that it allows users to define new modeling constructs (modules) that can be tailored to a specific problem domain. These domain-focused modules can be placed into libraries called templates. The development of application-specific templates for use with Arena creates the exciting opportunity to bring simulation technology to a large cross section of people who currently do not benefit from this technology. Arena brings simulation technology much closer to the application specialists--and does so across an unlimited range of problem domains.

REFERENCES

Glavach, M., and D. Sturrock (1993) *"Introduction to SIMAN/Cinema"* in Proceedings of the 1993 Winter Simulation Conference, IEEE, Piscataway, NJ.

Pegden, C.D., R.E. Shannon, and R.P. Sadowski (1990), *Introduction to Simulation Using SIMAN*, McGraw-Hill, New York, NY.

Pegden, C.D., D. Davis (1992), *"Arena: A SIMAN/Cinema-Based Hierarchical Modeling System"* in Proceedings of the 1992 Winter Simulation Conference, J.J. Swain, D. Goldsman, R.C. Crain, J.R. Wilson, Eds. IEEE, Piscataway, NJ.

Systems Modeling Corporation (1993), *Arena Template Reference Guide*, Sewickley, PA.

Systems Modeling Corporation (1993), *Arena User's Guide*, Sewickley, PA.

Systems Modeling Corporation (1993), *SIMAN V Reference Guide*, Sewickley, PA.

AUTHOR BIOGRAPHIES

NORENE COLLINS received her B.S. in Industrial Distribution and M.S. in Management Systems from Clarkson University. She is currently a quality assurance engineer at Systems Modeling. Her responsibilities include product testing and documentation. She is a member of IIE.

CHRISTINE M. WATSON received her B.S. in Industrial Engineering from Pennsylvania State University and her M.S. in Industrial Engineering from the University of Pittsburgh. She is a project engineer at Systems Modeling Corporation where she is responsible for training, consulting and template development. Her current interests include the simulation and improvement of service industry facilities. Prior to Systems Modeling in 1989, she worked for Westinghouse Electric Corporation. She is the Exhibits Chair for the 1993 WSC and is a member of IIE.