Proceedings of the 2012 Winter Simulation Conference
C. Laroque, J. Himmelspach, R. Pasupathy, O. Rose, and A.M. Uhrmacher, eds

RECENT INNOVATIONS IN SIMIO

David T. Sturrock
C. Dennis Pegden

Simio LLC
504 Beaver St.
Sewickley, PA, 15143, USA

ABSTRACT

This paper briefly describes Simio™ simulation software, a ***sim***ulation modeling framework based on ***i***ntelligent ***o***bjects. It then describes a few of the many recent enhancements and innovations including SMORE charts that allow unprecedented insight into your simulation output, sophisticated built-in experimentation that incorporates multi-processor support, optimization, and Risk-based Planning and Scheduling (RPS).

1 SIMIO BASIC CONCEPTS

The Simio modeling software lets you build and run dynamic 3D animated models of a wide range of systems – e.g., factories, supply chains, emergency departments, airports, and service systems. Simio employs an object approach to modeling, whereby models are built by combining objects that represent the physical components of the systems.

An object has its own custom behavior as defined by its internal model that responds to events in the system. For example, a production line model is built by placing objects that represent machines, conveyors, forklift trucks, aisles, etc. You can build your models using the objects provided in the Standard Object Library, which is a general purpose set of objects that comes standard with Simio. You can also build your own libraries of objects that are customized for specific application areas. As you will see shortly, you can also modify and extend the Standard Library object behavior using process logic.

One of the powerful features of Simio is that whenever you build a model of a system, you can turn that model into an object definition by simply adding some input properties and an external view. Your model can then be placed as a sub-model within a higher-level model. Hence, hierarchical modeling is very natural in Simio.

2 SIMIO OBJECTS

Simio is a ***sim***ulation modeling framework based on ***i***ntelligent ***o***bjects. The intelligent objects are built by modelers and then may be reused in multiple modeling projects. Objects can be stored in libraries and easily shared. A beginning modeler may prefer to use pre-built objects from libraries, however the system is designed to make it easy for even beginning modelers to build their own intelligent objects for use in building hierarchical models.

An object might be a machine, robot, airplane, customer, doctor, tank, bus, ship, or any other thing that you might encounter in your system. A model is built by combining objects that represent the physical components of the system. A Simio model looks like the real system. The model logic and animation is built as a single step. Further, you might typically work in 2D mode for ease of modeling, but the fully automatic compelling 3D animation is just a single keystroke away (Figure 1).

978-1-4673-4781-5/12/$31.00 ©2012 IEEE

Objects are built using the concepts of object-orientation. However unlike other object-oriented simulation systems, the process of building an object is very simple and completely graphical. There is no need to write programming code to create new objects.

The activity of building an object in Simio is identical to the activity of building a model. Whenever you build a model it is by definition an object that can be instantiated into another model. For example, if you combine two machines and a robot into a model of a work cell, the work cell model is itself an object that can then be instantiated any number of times into other models. The work cell is an object just like the machines and robot are objects. In Simio there is no way to separate the idea of building a model from the concept of building an object. Every model that is built in Simio is automatically a building block that can be used in building higher level models.

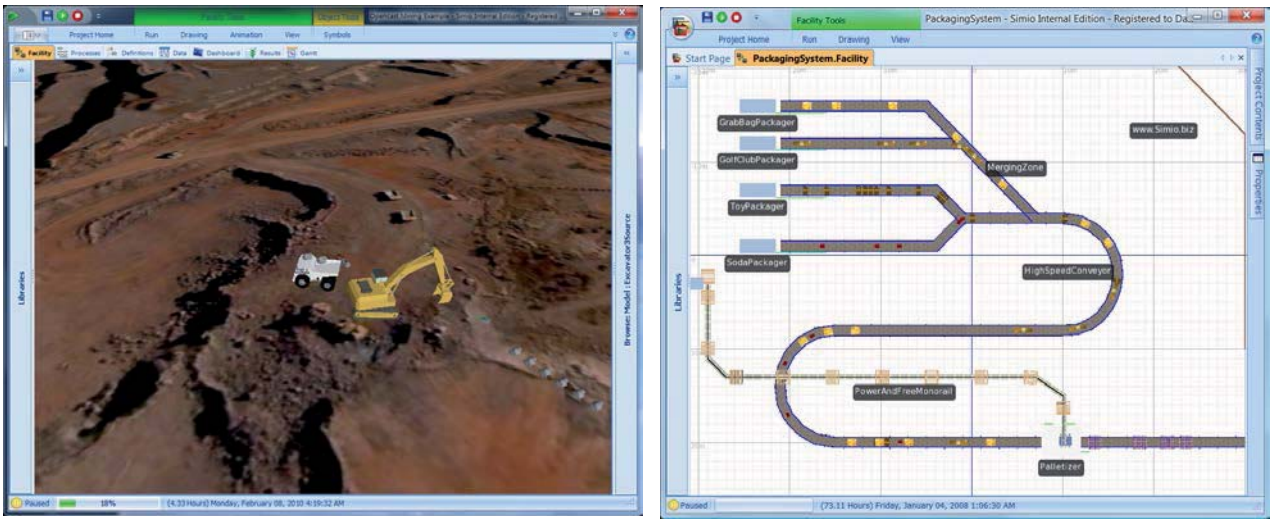

Figure 1: Ease of 2D modeling, with compelling 3D presentations

3 SIMIO USER INTERFACE

When you first open Simio you see a Start Page which includes links to the free *Introduction to Simio* e-book, the *Simio Reference Guide*, training videos, example models, and SimBits. The SimBits are small models that illustrate how to approach common modeling situations. From the Start Page you can click on New Model in the ribbon, or the Create New Model link to open a new model.

Models are defined within a project. A project may contain any number of models and associated experiments (discussed later). A project will typically contain a main model and an entity model. When you open up a new project, Simio automatically adds the main model (default name Model) and entity model (default name ModelEntity) to the project. You can rename the project and these models by right clicking on them in the project navigation tree. You can also add additional models to the project by right clicking on the project name. This is typically done to create sub-models that are then used in building the main model.

The entity model is used to define the behavior of the entities that move through the system. In older simulation systems entities cannot have behavior, and therefore there is no mechanism for building an entity model. However, in Simio, entities can have behaviors that are defined by their own internal model. The default model entity is “dumb” in that it has no explicit behavior, however as you will see later, you can modify the entity model to take specific actions in response to events. You can also have multiple types of entity models in your project, with each having its own behavior. For example in a model of an emergency department you could have different entities representing patients, nurses, and doctors.

As we will see later, a project can also be loaded into Simio as modeling library. Hence, some projects contain a collection of models for a specific application, and other projects contain models that are primarily used as building blocks for other models. The same project can either be opened for editing or be loaded as library.

The initial view of your Simio project is shown in Figure 2. The key areas in this screen include the ribbons across the top (currently showing the Run ribbon), the tabbed panel views with the Facility highlighted just below the ribbons, the libraries on the left, the browse panel on the right, and the Facility window in the center.

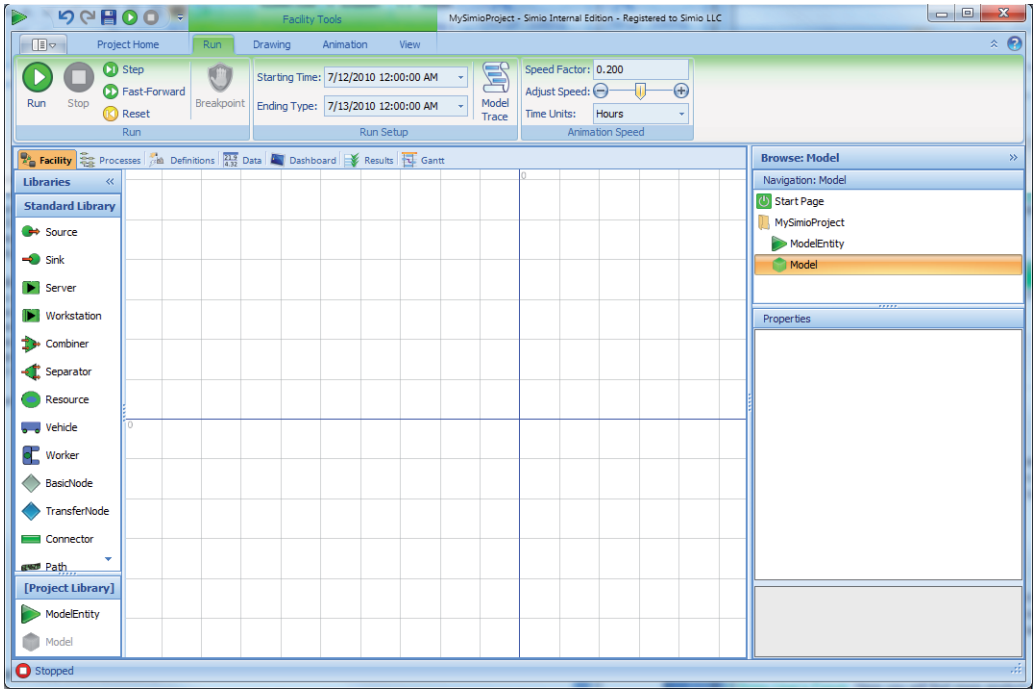

Figure 2: Initial view of your Simio project

The ribbons are tabbed UI panels that provide a convenient way to quickly access the many functions available to you for building, animating, and running models. You can manually switch between ribbons by clicking on the ribbon tabs. You can also condense the ribbons to just the tabs by double-clicking on any tab. In this mode clicking on the tab will expand the ribbon until you click in the model, and then the ribbon will then shrink back to just tabs. Double-clicking again on any ribbon tab will restore the ribbons to their regular size.

The Browse panel on the right provides for project navigation and property editing. The upper navigation window is used to switch between the Start Page, Project view, and the associated models and experiments. You can switch between models by simply clicking on the model in the navigation tree. For example clicking on the main model will make this the active model and display the Facility window for that model. Likewise clicking on the ModelEntity will make the default entity the active model, and display the initially empty set of processes that defines entities' behavior. The properties window that is located in the Browse panel immediately below the navigation window is used for editing the properties of objects.

Whenever the Facility window of a model is selected the Libraries panel on the left displays the libraries that are open and available for modeling within the facility. The libraries will include the Standard Library, the Project Library, and any additional projects that have been loaded as libraries from the Project Home ribbon. The Standard Library is a general purpose library of objects that is provided with

Simio for modeling a wide range of systems. The Project Library is a library of objects corresponding to the current models in your project. This lets you use your project models as sub-models that can be placed multiple times within a model. Note that the active model is grayed out in the Project Library since a model cannot be placed inside itself.

The Facility window that is shown in the center is drawing space for building your object-based model. This panel is shown whenever the Facility tab is selected for a model. This space is used to create both the object-based logic and the animation for the model in a single step. The other panel views associated with a model include Process, Definition, Data, Dashboard, and Results. The Process panel is used for defining custom process logic for your models. The ability to mix object-based and process modeling within the same model is one of the unique and very powerful features of Simio – this combines the rapid modeling capabilities of objects with the modeling flexibility of processes. The Definitions panel is used to define different aspects of the model, such as its external view and the properties, states, and events that are associated with the model. The Data panel is used to define data that may be used by the model, and imported/exported to external data sources. The Results panel displays the output from the model in the form of both a pivot grid as well as traditional reports. Note that you can view multiple model windows at the same time by dragging a window tab and dropping it on one of the layout targets.

4 3D ANIMATION

By default Simio provides a top-down 2-D view of the facility model. This is often a very convenient view for creating and editing your model. However facility models in Simio are defined in 3D. You can switch between the 2D and 3D views of the facility using the View section of the View ribbon, or by using the 2 and 3 keys on the keyboard. The View section also lets you set the facility model to auto-rotate in 3D – click anywhere in the facility to stop auto-rotation. You can also change the background color of the 3D space. Click anywhere in the facility and drag to pan the view. Press the right mouse button and move left and right to rotate the view, or up and down to zoom in and out. You can also zoom using the mouse wheel.

An object may be animated in 3D to reflect the changing state of the object. For example a forklift truck raises and lowers its lift, a robot opens and closes its gripper, and a battle tank turns its turret. The animated model provides a moving picture of the system in operation. Simio provides a direct link to Google Warehouse (a free massive online library of graphic symbols) for animating 3D objects. At the instant you are looking for a 3D symbol, from within Simio you can search for, download, and incorporate that symbol into your model (Figure 3).

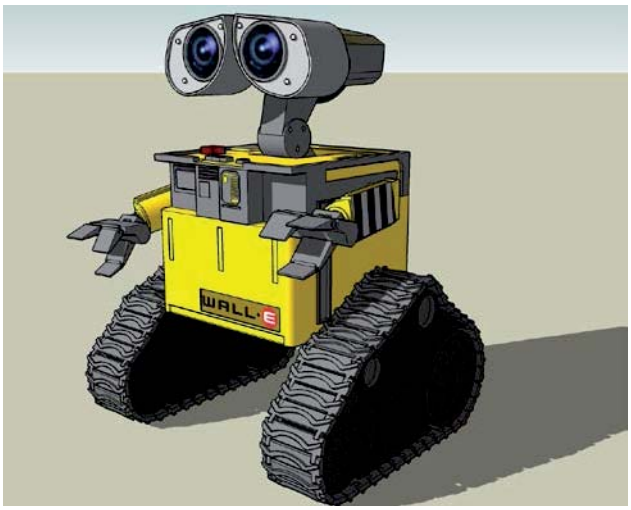

Figure 3: Google Symbol Warehouse 3D objects import directly into Simio

5 STANDARD LIBRARY

The Standard Library included with Simio provides a rich and customizable set of modeling features. They can be used by dragging an object into the facility view and connecting it to other objects. Each object has a comprehensive set of properties to allow customizing its behavior. In addition, the behavior of each object can be extended by taking advantage of add-on processes to define extra logic custom to your application. This avoids the "brick walls" of application often encountered using modules and pre-built objects in other products. Finally, all of the objects in the standard library have been defined using processes. These object definitions are open – you can view, copy, and even change the object definitions. The Standard Library objects are illustrated in Table 1.

Table 1: Simio Standard Library objects

Object	Description
Source	Generates entity objects of a specified type and arrival pattern.
Sink	Destroys entities that have completed processing in the model.
Server	Represents a capacitated process such as a machine or service operation.
Workstation	Includes setup, processing, and teardown and secondary resource and material requirements.
Combiner	Combines multiple entities together with a parent entity (e.g. a pallet).
Separator	Splits a batched group of entities or makes copies of a single entity.
Resource	A generic object that can be seized and released by other objects.
Worker	A moveable resource that may be seized for tasks as well as used to transport entities.
Vehicle	A transporter that can follow a fixed route or perform on demand pickups/drop offs.
BasicNode	Models a simple intersection between multiple links.
TransferNode	Models a complex intersection for changing destination and travel mode.
Connector	A simple zero-time travel link between two nodes.
Path	A link over which entities may independently move at their own speeds.
TimePath	A link that has a specified travel time for all entities.
Conveyor	A link that models both accumulating and non-accumulating conveyor devices.

6 RESOURCE CONCEPT, STATES & STATISTICS

Every object has built in resource behavior. Any object may seize capacity of another object and use it as a "resource." Objects have a resource capacity that can be fixed or follow a schedule. Objects maintain a ranked queue of other objects that are waiting to seize capacity. When an object is released it is reallocated by either selecting the first in the queue, or dynamically selecting from all waiting objects based on a rule.

Objects can intelligently interact with each other and the environment. A resource may reject a seize request if it is for example, batching up work of a certain type, or too close to a break to start new work. Resources have automatic state-based and overall statistics collection as well as optional animation as illustrated in Figure 4.

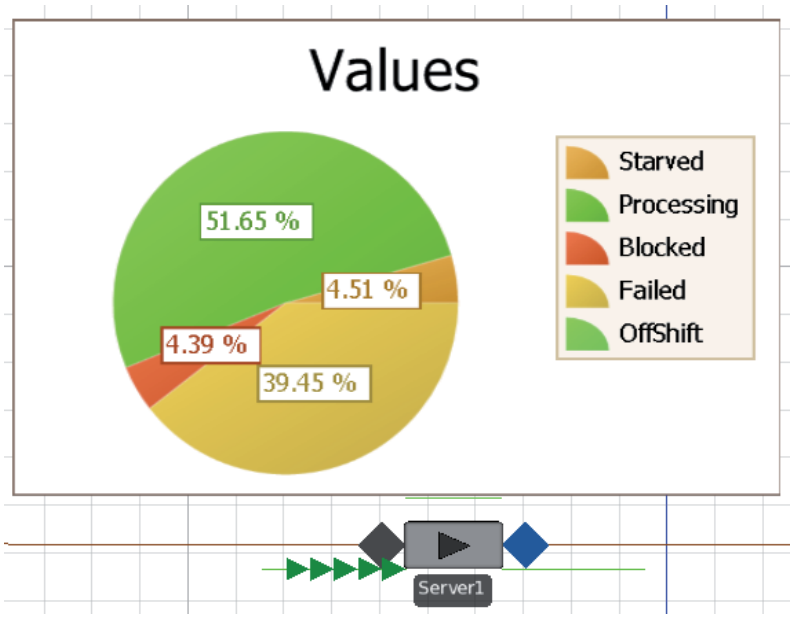

Figure 4 – Resource State Animation

7 PROCESSES

Object-based tools such Simio are very good for rapidly building models. You simply drag objects into the workspace, set the properties for those objects, and your model is ready to run. However the traditional problem with this approach is modeling flexibility. It's extremely difficult to design a set of objects that work in all situations across multiple and disparate application areas without making the objects overly complicated and difficult to learn and use.

The Simio Standard Library addresses this problem through the concept of add-on processes. An add-on process is a small piece of logic that can be inserted into the Standard Library objects at selected points to perform some custom logic. This custom logic can be used to seize/release resources, make assignments to variables, change travel networks, evaluate alternatives, etc. The processes are created as graphical flowcharts without the need for programming. Hence Simio combines the benefits of object-based modeling (i.e., ease of learning and rapid modeling) with the power and flexibility of graphical process modeling.

Process logic can be inserted into an object on an instance by instance basis, without modifying or changing the main object definition. For example one Server instance might incorporate process logic to seize and move a secondary resource during processing, while another instance for the same Server definition incorporates special logic to calculate the processing time based on a learning curve, and a third instance of the same Server incorporates special process logic for modeling a complex repair process. All these instances share the same Server definition but customize each instance as needed.

Another thing that is particularly powerful about add-on processes is that – unlike a programming insert - processes can span simulated time. For example a process can wait for tank to fill, a resource to become idle, or a queue to reduce to a specific size. Hence processes are not only easier to learn, create, and understand, they are also significantly more powerful than programming inserts.

You build and edit processes using the Process window shown in Figure 5. The left side of this window has a fly-out panel for selecting steps, and the right hand side has the navigation panel and properties editor. The steps are categorized by Common Steps, All Steps, and User Defined Steps. The Common steps include the steps that are most frequently used in models. The All Steps include all of the built-in

steps for Simio. The User-Defined Steps include user-coded steps that have been added to Simio to extend the core functionality of the product (for details see the Simio User-Extensions Reference Guide).

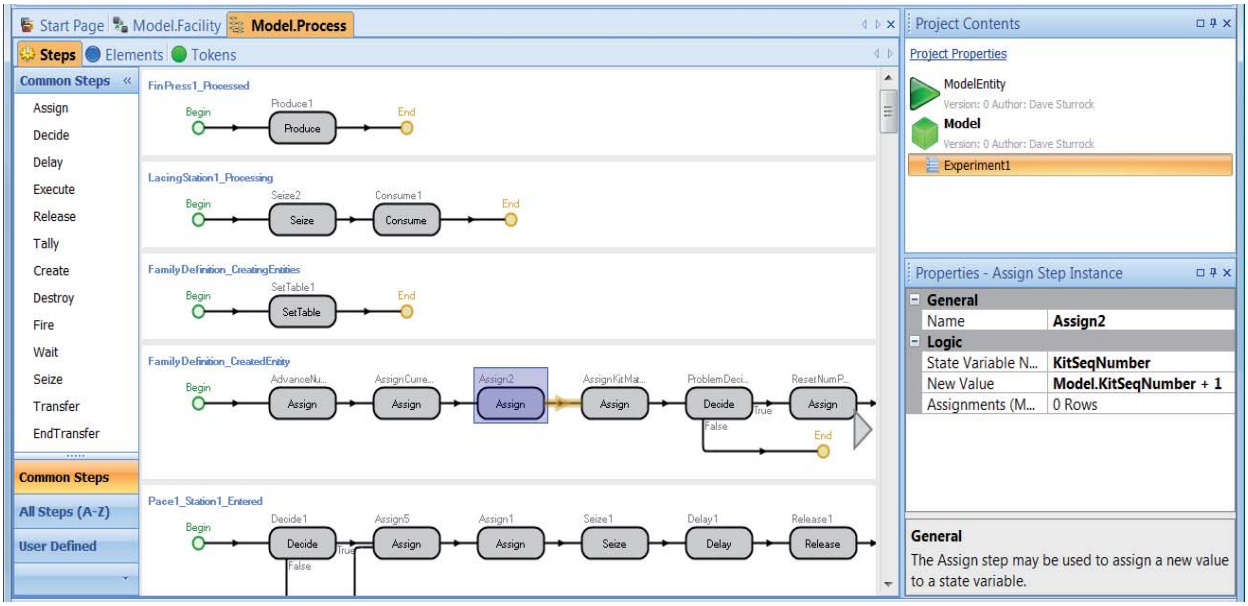

Figure 5: Graphical process definition

8 BUILDING OBJECTS

One way to build a new object definition is simply by combining other objects. Essentially as you are building a model, you are also building an object. This type of object is called a *composed object* because we create this object by combining two or more *component* objects. This object building approach is fully hierarchical, i.e., a composed object can be used as a component object in building higher level objects. And you can now select which components are visible and animated when others use your object. You would typically add a custom user view and properties in order to make this type of object customizable by others. This is only one way of building objects in Simio - there are two other important methods.

The most basic method for creating objects in Simio is by defining the logical processes that alter their state in response to events. For example, a machine object might be built by defining the processes that alter the machine state as events occur such as part arrival, tool breakdown, etc. This type of modeling is similar to the process modeling done in traditional modeling systems in use today such as Arena or GPSS. An object that is defined by describing its native processes is called a *base object*. A base object can in turn be used as a component object for building higher level objects.

The final method for building objects in Simio is based on the concept of inheritance. In this case we create an object from an existing object by overriding (i.e., replacing) one or more processes within the object, or adding additional processes to extend its behavior. In other words we start with an object that is almost what we want, and then we modify and extend it as necessary to make it serve our own purpose. For example we might build a specialized drill object from a generalized machine object by adding additional processes to handle the failure and replacement of the drill bit. An object that is built in this way is referred to as a *derived object* because it is sub-classed from an existing object.

Regardless which method is used to create an object, once created it is used in exactly the same way. An object can be instantiated any number of times into a model. You simply select the object of interest and place it (instantiate it) into your model.

9 EXPERIMENT WINDOW

There are two basic modes for executing models in Simio. The first mode is **interactive mode** where you can watch the animated model execute and view dynamic charts and plots that summarize the system behavior. This is useful for building and verifying/validating the model as well as getting general insight into how the system will perform. Once the model has been validated the next step is typically to define specific scenarios to test with the model. In this case we have no interest in the animation and we would like to replicate each scenario to account for the underlying variability in the system and to reach statistically valid conclusions from the model.

In the **experimentation mode** we define one or more properties on the model that we can change to see the impact on the system performance. These properties, exposed in the experiment as Controls, might be used to vary things like conveyor speeds, the number of operators available, or the decision rule for selection of the next customer to process. These model properties are then referenced by one or more objects in the model. You may also add Responses. These would generally be your Key Performance Indicators (KPIs) on which you make the primary decision on "goodness" of the scenario. You may dynamically sort on any column, for example to display the highest profit scenarios first. You can also add Constraints that will automatically be applied before or after a run to prevent running or to later discard a scenario that violates an input or output constraint. When you run an experiment, it takes full advantage of all processors available. For example, Figure 6 illustrates four scenarios being executed concurrently on an inexpensive quad-core machine.

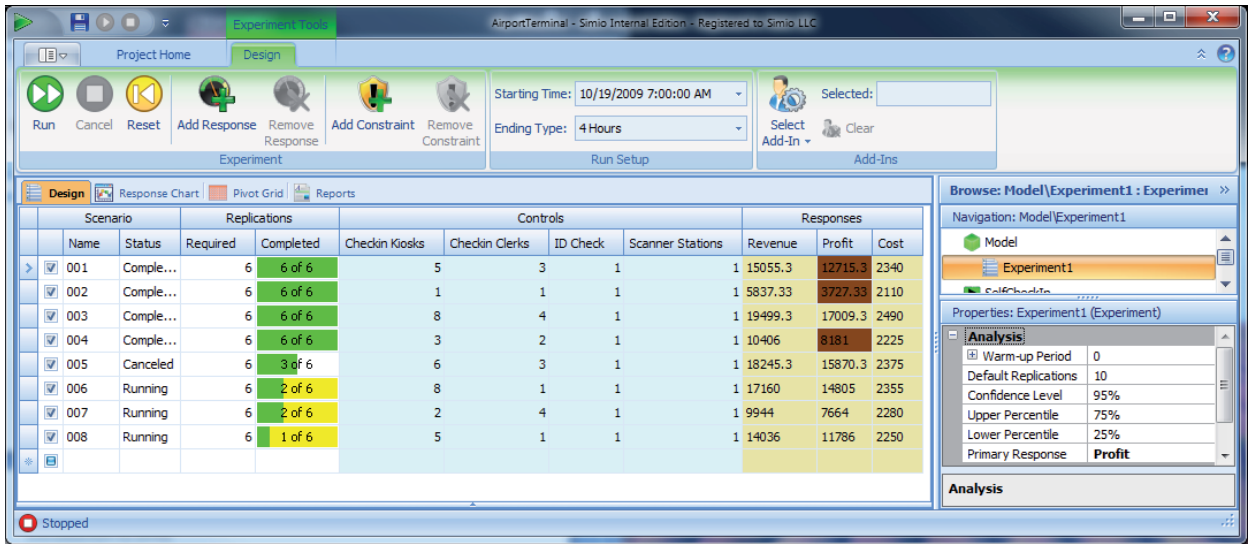

Figure 6: Simio Experiment Window for airport application

Simio experiments can also run user-defined add-ins. These are written in any .NET language to apply additional analysis to an experiment. Because they can be added to and take full advantage of the foundations described above, they are relatively easy to write – primarily focusing on the analytical task and not the "housekeeping" tasks.

Two add-ins were added to Simio Version 3. A **Ranking & Selection** add-in is supplied that uses algorithms based on Dr. Barry Nelson's research to automatically take a set of "good" scenarios and provide analysis and possibly additional replications to discern the best. **OptQuest** has also been made available as an add-in to provide comprehensive optimization capabilities (including full support of the multi-processor capabilities). OptQuest is an extra-cost add-in for commercial use, but it is provided in an eval-

uation mode. OptQuest is also included free, in unlimited mode, with our Academic and Student licenses (but is limited to non-commercial applications as is our academic software).

10 SMORE PLOTS

Simio Pivot Tables and Reports provide an estimate of the population mean and confidence interval based on multiple replications. While this is exactly what is needed in some situations, in others it provides an inadequate amount of information required to make a decision taking into account risk. As Dr. Nelson puts it in his paper discussing the concepts of Measure of Risk & Error (MORE) plots (Nelson 2008), "A baseball player's batting average is a meaningful historical statistic. However, a simulation is not trying to create history; instead it is trying to say something about what will happen in the future and whether we can live with it."

The Response Results window of an Experiment creates a Simio Measure of Risk & Error (SMORE) plot using the Response value that is selected in the Response pull down menu. A SMORE plot displays both the estimated expected value of a scenario and multiple levels of variability behind the expected value. The plot displays results across replications, for each scenario.

A SMORE plot consists of a Mean, Confidence Interval for the Mean, Upper Percentile Value, Confidence Interval for the Upper Percentile Value, Lower Percentile Value, Confidence Interval for the Lower Percentile Value, Median, Maximum Value, and Minimum Value (Figure 7).

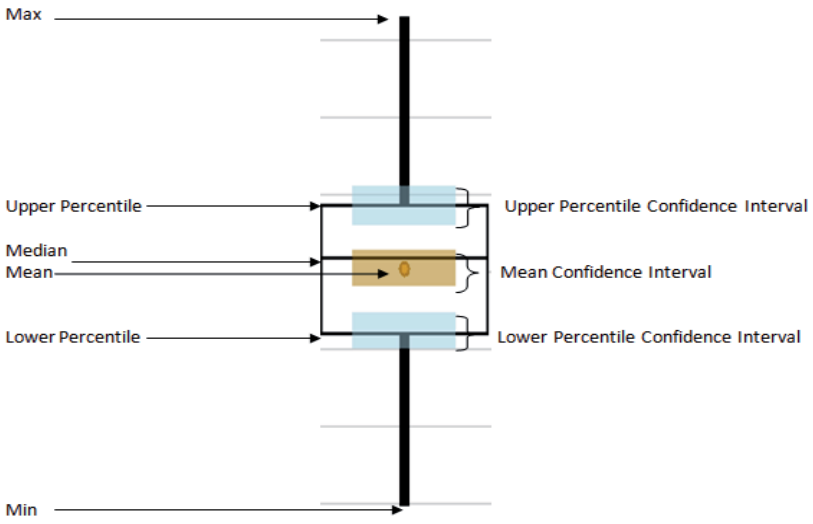

Figure 7: Major components of a SMORE Plot Item

The Response Results tab can be moved so that the graph can be seen plotting as the Design Window steps through replications and scenarios, in real time. In addition you can enable/disable optional graphic features like the Confidence Intervals, Connecting lines, Constraint limits, Individual Observations, and Histograms. Using the Raw Data tab, you can view and export the details for additional analysis. You can also use this in combination with experiment add-ins. For example, Figure 8 shows concurrent real-time display of the Design window and the Response Chart for a OptQuest optimization analysis.

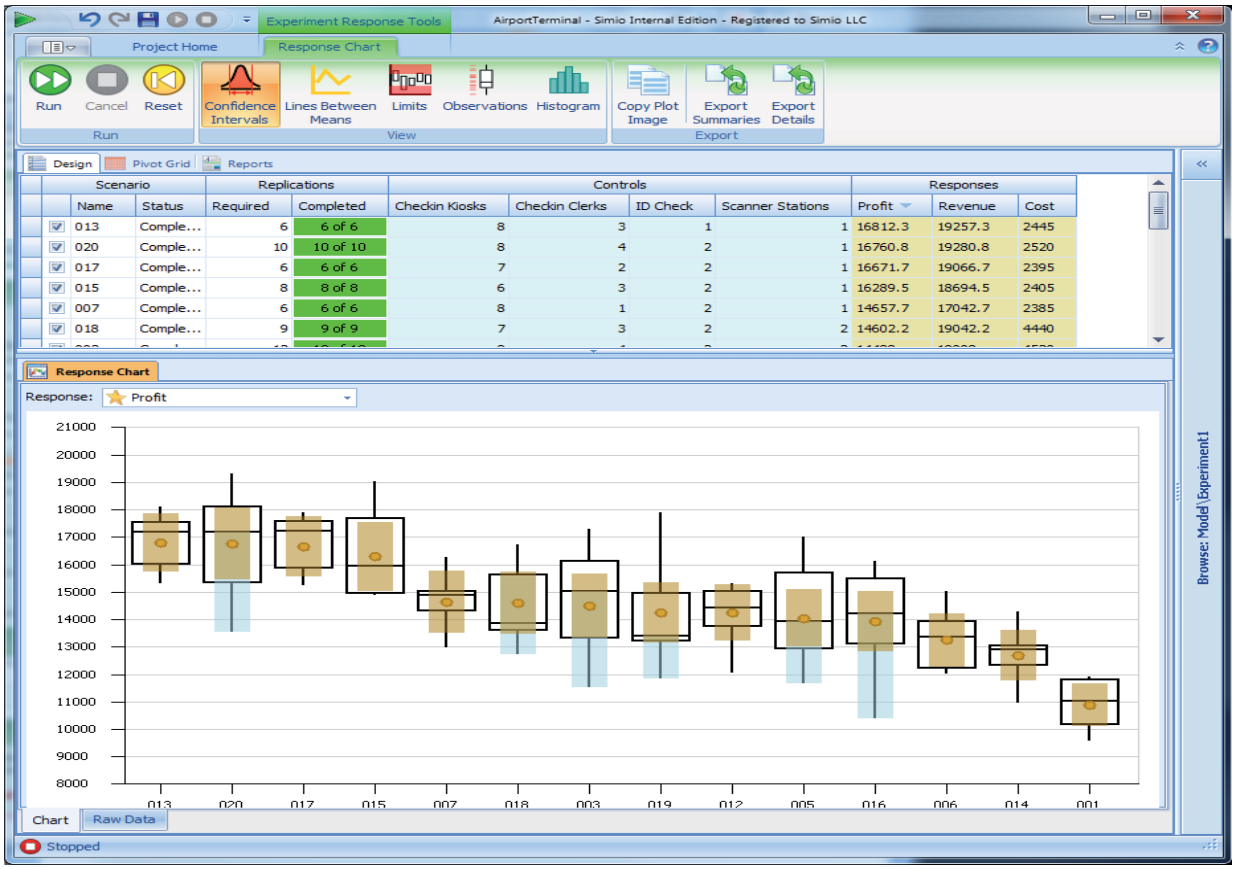

Figure 8: SMORE Plot of airport optimization example

11 RISK-BASED PLANNING AND SCHEDULING (RPS)

Traditional planning and scheduling methods are unable to account for unavoidable variations, making it difficult for them to remain of value in the long term. In contrast, simulation-based planning and scheduling can incorporate variation into the model, providing long-term value. However, this requires a new set of simulation tools specifically designed to focus on planning and scheduling, including the ability to account for the underlying risk imposed by variations in the system. These new tools go beyond the traditional use of simulation for comparing alternative designs and directly support the use of models within an operational setting to improve the everyday production, operational, and business level decisions that are key drivers to the overall success of an organization.

Risk-based Planning and Scheduling is the application of simulation methodology to operational planning and scheduling. We can incorporate variation and unplanned events into the same base model that we use to generate the plan to also generate risk measures for each transaction relative to its targets. Hence a given plan can be judged by the decision-maker not just on its feasibility at the time that the plan is generated, but also the robustness of the plan over time in terms of the underlying risk associated with hitting each target that has been defined for each individual transaction that we are planning. This provides the decision maker with the ability to plan critical operations while fully accounting for the underlying risk imposed by variations in the system.

12 SUMMARY OF OTHER RECENT CHANGES

There has been a plethora of other new features recently added to Simio. Many vendors are still locked to the archaic waterfall development process which results in software updates perhaps once every 12-18 months. Simio follows an agile development methodology allows you to choose to receive new features when it's good for you, (they come out about monthly) rather than wait until the next major release. This process also dramatically improves our productivity – often introducing more features in a month than you might see in the annual releases of others. Over 40 pages was required to briefly summarize the changes from Simio Version 4 to Version 5. In addition to the features already discussed, we have introduced new capabilities in the areas of flow processing of all types (continuous, discrete, batch, mass, and mixed flows), experimentation, distributed processing, searching, detailed control logic, support of alternate units (length, volume, mass), optimization, relational database support, sophisticated attached animation (that can even travel with entities), worker and vehicle enhancements, improved SimBits and documentation, add-in hooks, object and add-in sharing, API improvements, Risk-based Planning and Scheduling, and much more. You can find the full list on-line at http://www.simio.com/resources/release-notes/.

13 SUMMARY

Simio is a new modeling framework based on the core principles of object oriented modeling. It is unique in the following ways:

1. The Simio framework is a graphical object-oriented modeling framework as opposed to simply a set of classes in an object-oriented programming language that are useful for simulation modeling. The graphical modeling framework of Simio fully supports the core principles of object oriented modeling without requiring programming skills to add new objects to the system.
2. Simio supports 3D animation as a natural part of the modeling process. Simio provides a direct interface to Google Warehouse to make it easy to incorporate realistic 3D symbols into your model.
3. The Simio framework is domain neutral, and allows objects to be built that support many different application areas. The process modeling features in Simio make it possible to create new objects with complex behavior without programming (no C#, Java or custom language use is required).
4. The Simio framework supports multiple modeling paradigms. The framework supports the modeling of both discrete and continuous systems, and supports an event, process, object, systems dynamics, and agent modeling views.
5. The Simio framework provides specialized features to directly support applications in emulation and finite capacity scheduling that fully leverage the general modeling capabilities of Simio.
6. Risk-based Planning and Scheduling (RPS) extends traditional APS to fully account for the variation that is present in nearly any production system, and provides the necessary information to the scheduler to allow the upfront mitigation of risk and uncertainty.

REFERENCES AND ADDITIONAL READING

Kelton, W. D., J. S. Smith, and D. T. Sturrock. 2011. *Simio and Simulation: Modeling, Analysis, Applications.* 2nd ed. New York: McGraw-Hill, Inc

Nelson, B. L. 2008. "The MORE Plot: Displaying Measures of Risk & Error From Simulation Output." In *Proceedings of the 2008 Winter Simulation Conference*, edited by S. J. Mason, R. R. Hill, L. Mönch, O. Rose, T. Jefferson, J. W. Fowler, Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.

Nygaard, K and O-J Dahl. 1962. "SIMULA -- An Extension of ALGOL to the Description of Discrete-Event Networks." Presented at the *Second International Conference on Information Processing*.

Sturrock, D. T. 2011. Success in Simulation, Ongoing blog and discussion. Accessed July 15. http://simio.com/blog/about-this-blog/.

AUTHOR BIOGRAPHIES

DAVID T. STURROCK is vice-president of Products for Simio LLC. He graduated from the Pennsylvania State University in Industrial Engineering. He has over 25 years of experience in the simulation field and has applied simulation techniques in the areas of transportation systems, scheduling, plant layout, call centers, capacity analysis, process design, health care, packaging systems and real-time control. He is co-author of two simulation textbooks and teaches simulation at the University of Pittsburgh. In his present role for Simio he is responsible for development, support and services for their simulation and scheduling product suite. His email is dsturrock@simio.com.

C. DENNIS PEGDEN is the founder and CEO of Simio LLC. He was the founder and CEO of Systems Modeling Corporation, now part of Rockwell Software, Inc. He has held faculty positions at the University of Alabama in Huntsville and The Pennsylvania State University. He led in the development of the SLAM, SIMAN, Arena, and Simio simulation tools. He is the author/co-author of three textbooks in simulation and has published papers in a number of fields including mathematical programming, queuing, computer arithmetic, scheduling, and simulation. His email is cdpegden@simio.com. Additional company information can be found at www.simio.com.