How Simio Objects Differ from Other Object-Oriented Modeling Tools

C. Dennis Pegden, Ph.D.

Simio makes modeling dramatically easier and faster by providing a new object-based paradigm that radically changes the way objects are built and used. The purpose of this paper is to describe how the object-oriented modeling framework of Simio differs from other object-oriented modeling tools. To arrange an issue specific demonstration at WSC 2009, email dennis_pegden@simio.biz.

The general idea of object-oriented modeling is not new – in fact the first object oriented modeling tool – Simula – was introduced over 50 years ago and provided the core ideas in use today in object-oriented modeling tools. There are a number of object-oriented modeling tools on the market day. When using these tools a user selects objects from a library and places them in a model. A general consensus from many users of these tools is that they work well on simple models, but for complex applications they are difficult to use because they either lack flexibility (users cannot add or modify objects), or achieve flexibility by requiring the user to write complex code in a programming language such as C++ or Java. Either option is a barrier to the user in terms of rapidly modeling complex systems. This has been a key constraint on the widespread acceptance of object-oriented modeling tools.

Simio differs from other object-oriented modeling tools in that Simio objects are ***process-based*** rather than ***code-based***. A Simio object is defined by creating a set of graphical process flows that describe the object's behavior. A process is a flowchart that describes a sequence of activities and decisions that are made by the object. A process may span time and may be constrained by limited resources. A simple example of a process that is familiar to many users of process-oriented modeling tools is: SEIZE-DELAY-RELEASE. In this process the object waits to seize a resource, delays by an activity time, and then releases the resource. Note that this activity spans time and the time to execute the process is dependent on both the availability of the resource and the specified delay time.

In other tools the objects are code-based and implemented in a programming language. If the tool supports user-defined objects, then the user must implement any new objects in the same programming language. The user must have mastered the basic concepts of object-orientation (e.g. encapsulation, inheritance, polymorphism, etc.), and also be skilled in the required programming language. As a result the creation of a new object requires an expert programmer in a specific programming language.

The process-based objects in Simio have a number of important advantages over the code-based objects in other tools. The first and most obvious advantage is that objects are much easier to create since they do not require programming skills in a specific language. In addition since the logic for a Simio object is defined by graphical process flows and is visible to the user they are easier to understand and to modify. Most importantly, however, the object behavior in Simio is defined using high-level process modeling constructs that span time. This greatly simplifies the task of building objects.

In most object-oriented modeling tools the user is able to embellish the provided objects with custom logic for a specific application. For example it might be necessary to count the number times that a customer completes service on a specific server and then use this count in some way within the model logic. This type of user-added logic is very important to be able to flexibly model a wide range of systems. Tools will typically provide a way to add such logic at predefined points in the objects that are provided in their standard library.

The process-based objects in Simio again have some unique and important advantages over code-based objects when it comes to adding custom logic to existing objects. A code-based object will have logical "hooks" to make calls to a user-supplied function that is executed at selected points in the object.   The user must code this function in the specified programming language (e.g. C++ or Java). In some cases a simplified scripting tool is provide as an alternative to do simple things such as assignments without coding, however the flexibility and power of the scripting tools are very limited. In either case, however, the inserted logic must fully execute at that specific point in simulated time. The logic cannot delay for a specified time, wait for a resource to become available or relocate, wait for a tank to reach a specified fill level, or perform other types of complex logic that spans simulated time. In contrast the process-based objects in Simio provide a feature called "add-on" processes that are executed at specific logical points in the object. These are the counterparts to the function calls in code-based objects but are much more powerful since they have the full power of Simio processes and can span time as necessary. For example there is an add-on process for the Server object in Simio that is run whenever the Server fails. This process could be created by the user to include logic to wait to seize the repairman, and also wait for the repairman to arrive at the Server before returning control back to the object. This is an extremely powerful capability.

In summary, the process-based objects in Simio provide a number of important advantages over the more traditional code-based objects found in other object-oriented modeling tools. These advantages include both ease-of-use by eliminating the requirement to be an expert programmer, as well as modeling flexibility by allowing objects to be defined and embellished using processes that span time as opposed to coded functions that must execute without a simulated time-advance. In short, the process-based objects in Simio (patent pending) are both simpler and more powerful than the code-based objects in other modeling tools.

About The Author

Simio® LLC was founded by Dennis Pegden, Ph.D. to reinvigorate the field of simulation modeling. A 33-year veteran of the simulation industry, Pegden's career highlights include leading the development of SLAM (marketed by Pritsker and Associates) and founding Systems Modeling Corporation, now part of Rockwell Automation. Pegden also led the creation of the simulation products SIMAN and Arena. Simio's management team brings one hundred five years of combined experience in the simulation market. To arrange an issue specific demonstration at WSC 2009, email dennis_pegden@simio.biz.