

17

A Simio™ job data set allows a list of jobs to be externally defined for processing by the simulation model. The jobs are defined in a data set containing one or more tables, with relations defined between table columns. The specific schema for holding the job data is arbitrary. And it can be user defined to match the data schema for the manufacturing data (e.g. an ERP system). The job data typically includes release and due date, job routings, setup and processing times, material requirements, as well as other properties that are relevant to the system of interest. The objects in Simio™ can directly reference values specified in the job data set (e.g. processing time) without knowing the schema that was implemented to store the data.

The resource features built into Simio™ objects provide direct support for modeling complex resource selection and dynamic routing logic. Objects in Simio™ have a user-defined capacity and can be seized, released, and preempted by other objects. Objects can follow work shifts that can alter the time spent by a job being processed thereby. Objects can also model complex changeover logic for jobs that utilize the object (e.g. change-dependent or sequence-dependent changeovers). Objects can be placed in multiple lists, and selection of an object from a list can be based on flexible rules such as minimum changeover time or longest idle time. Jobs can also be dynamically routed between objects based on the state of an object (e.g. a machine). Objects also support very flexible rules (earliest due date, least remaining slack, critical ratio, etc) for selecting between competing jobs that are waiting to seize the object. Finally, the job usage history for objects can be displayed on an interactive Gantt chart.

The Materials element in Simio™ provides direct support to model things that can be consumed and produced during the execution of the model. Materials can also be defined hierarchically to model a traditional Bill of Materials (BOM) for manufacturing applications. Hence, a manufacturing step can be modeled as the consumption of a specific list of materials within the hierarchical BOM.

The present invention may be embodied in other specific forms without departing from its spirit or essential characteristics. As will be apparent to one skilled in the art, the functionality of the invention described herein is implemented by computer instructions which execute on a computer, in other words on a physical computing device. In a preferred embodiment, the computer instructions (software) are written in the C# programming language, and run on a Microsoft Windows operating system.

The described embodiments are to be considered in all respects only as illustrative, not restrictive. The scope of the invention is, therefore, indicated by the appended claims rather than by the foregoing description. All changes that come within the meaning and range of equivalency of the claims are to be embraced within their scope.

What is claimed is:

1. A computer-based system for developing simulation models on a physical computing device, the system comprising:

one or more graphical processes;  
one or more base objects created from the one or more graphical processes,

18

wherein a new object is created from a base object of the one or more base objects by a user by assigning the one or more graphical processes to the base object of the one or more base objects;

wherein the new object is implemented in a 3-tier structure comprising:

an object definition, wherein the object definition includes a behavior,

one or more object instances related to the object definition, and

one or more object realizations related to the one or more object instances;

wherein the behavior of the object definition is shared by the one or more object instances and the one or more object realizations; and

an executable process to add a new behavior directly to an object instance of the one or more object instances without changing the object definition and the added new behavior is executed only for that one instance of the object.

2. The computer-based system of claim 1, wherein the new object is created without the need for methods or programming.

3. The computer-based system of claim 1, wherein a derived object is created from an existing object by overriding one or more graphical processes assigned to the existing object and/or assigning additional graphical processes to the existing object.

4. The computer-based system of claim 3, wherein a composite object is created by combining two or more base objects, derived objects and/or objects that are themselves composite objects.

5. The computer-based system of claim 1, wherein types of base objects include 1) fixed objects, 2) agent objects, 3) entity objects, 4) transporter objects, 5) link objects, and 6) node objects.

6. The computer-based system of claim 1, wherein the graphical process is a sequence of process steps that is triggered by an event.

7. The computer-based system of claim 6, wherein the sequence of steps in a graphical process include a Begin step and an End step, and additional steps are added by a user from a collection of steps.

8. The computer-based system of claim 6, wherein event types include 1) time events, 2) logic events, 3) change events and 4) cross events.

9. The computer-based system of claim 1, wherein all the objects have standard processes that are triggered by model logic, and the standard processes define behavior all for the objects.

10. The computer-based system of claim 1, wherein a model is built by graphically combining one or more base, derived, and/or composite objects that represent physical components of a system being modeled.

11. The computer-based system of claim 10, wherein the model is a finite capacity scheduler.

12. The computer-based system of claim 10, wherein the system being modeled is a discrete system.

13. The computer-based system of claim 10, wherein the system being modeled is a continuous system.

\* \* \* \* \*