Principles of Modeling

The First Crucial Step in Creating a Virtual World

Roger Smith
ModelBenders LLC

http://www.modelbenders.com/

Notes

The process of modeling is an exercise in abstraction, setting limits, uncovering and defining relationships, and creating information where none is available. This process can be defined by some universal principles that are used by all modelers and that can be learned in the classroom much more economically than they can be learned through trial-and-error.

This course exposes some of the Principles of Modeling that have been learned and shared by practitioners experienced in the art and science of modeling and simulation over the last 30 years.

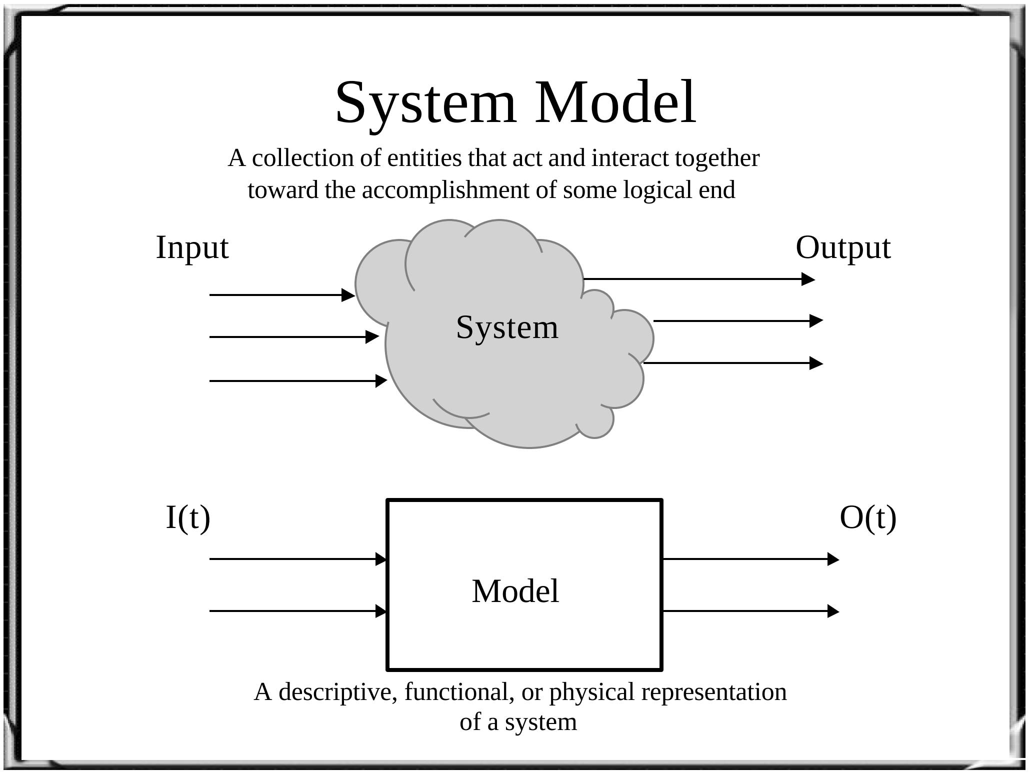
Course Outline

- Introduction
- The Philosophy
- The Principles
- The Products
- The Process
- The Practice
- The Programming
- Conclusion

Notes

This course is divided into several sections. We will begin with an introduction of the concepts, terms, and ideas that are essential to modeling. We will then explain the basic philosophy and mid-set behind modeling. This is followed by a collection of modeling principles that have been collected from experts. Once you understand the principles we will describe the different parts of a system model and go through the process of creating a model. Finally, we will provide a complete example of a model product.

At the conclusion of this course you will have a clear picture of what is involved in creating a model.



Notes

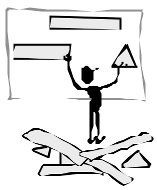
A model is an abstraction that behaves somewhat like a defined "system". In the real world, a system is a set of transformations that convert input states into output states and events. This general definition covers everything from the operations of a computer to the process of driving nails. The key is that it converts Input into Output through some defined set of algorithms.

If the real system algorithms are defined, then they can be captured in an abstract form as a "model". By definition the model is less exact and less all-encompassing than the original system (otherwise the model would be the system). The model accepts a set of Input values and transforms them into Output values.

The accuracy of the model is measured by the degree to which model Outputs match system Outputs - given corresponding Input values.

Goal of Modeling

"Modeling is a way of thinking and reasoning about systems. The goal of modeling is to come up with a representation that is easy to use in describing systems in a mathematically consistent manner."



Paul Fishwick,
Simulation Model Design and Execution

Notes

Creating a model of a single object or event is not very difficult. However, creating a model of an entire virtual world in which all of the objects and events are represented in a consistent and complimentary way requires a great deal of mental and creative effort, as well as hard-earned expertise.

Most people begin modeling by creating representations of systems that are inconsistent or inefficient. Through trial-and-error they learn better techniques. Unfortunately, this trial-and-error process is extremely expensive and time consuming. One of the objectives of this class is to allow you to collect the expertise of others rather than paying for every lesson learned yourself.

Avoid Too Much Detail

"The tendency is nearly always to simulate too much detail rather than too little. Thus, one should always design the model around the questions to be answered rather than imitate the real system exactly …"

Robert E. Shannon,
Systems Simulation: The Art and Science,
1975

Notes

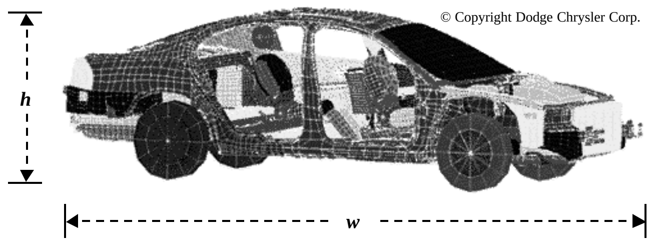
Design a model to answer the question, not to mimic the real world.

In 1975 Robert Shannon published a fact that has haunted every novice model builder:

> "The tendency is nearly always to simulate too much detail rather than too little. Thus, one should always design the model around the questions to be answered rather than imitate the real system exactly"

Detail in the model is an attractive and seductive thing. The model is enticed by the intricacy and beauty of the creation itself and loses sight of the purpose for which the model is being built. Additional detail requires additional software, data, debugging, CPU time, and storage space. It can also make you forget why you are creating the model to begin with.

Physical Model



Sample Attributes
Height Width Mass Shape Composition Tensil Strength

ACTION: **Capture physical attributes of the objects to be modeled. Usually involves measurement and reference manuals. The goal is to gather all attributes that can contribute to events and interactions within the simulation or that are impacted by those events and interactions in a way that is significant to the user of the model.**

ADVICE: **Physical model attributes are not to be gathered until after Logical and Dynamic models have been defined.**

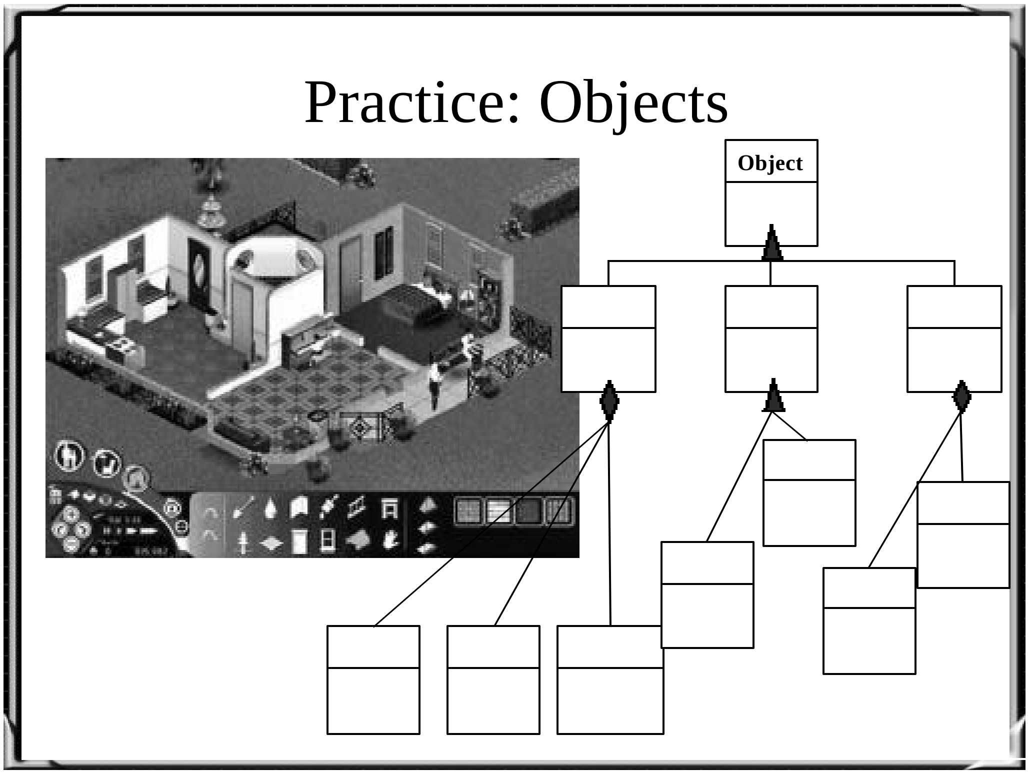
WARNING: **The biggest danger in this phase of modeling is the collection of far more attributes than will actually be used in the simulation system.**

Notes

The Physical Model captures details about the physical existence and capabilities of objects and phenomena. To illustrate this, and several other models we are going to explore the process of modeling an automobile.

The physical model of an automobile may define its length, width, height, and mass. Information about its shape may be captured as well as the composition of key parts of the vehicle. The physical model may totally omit information about the color of the paint and upholstery. It may not speak to the quality of the stereo system or any of the other characteristics that are essential to selling the vehicle. This model is focusing on its physical handling and crash-worthiness.

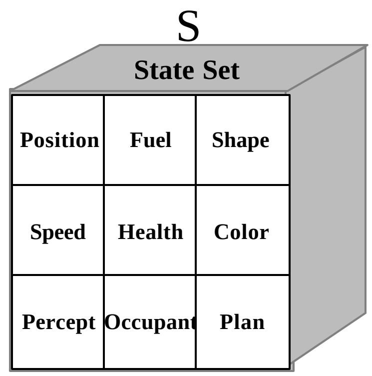
The details required for the physical model are not known because you know what the real vehicle is like. Rather the details are necessary because of the specific purpose of the model as expressed in the problem statement of the system. This problem statement is also expressed in the logical and dynamic models of the system.



Notes

The blank framework for an initial object decomposition is shown in this diagram. Identify the object specializations and decompositions that would exist for this scene.

State Set



<u>Discrete Event Simulation</u>
State changes only in response to an event

<u>Continuous Simulation</u>
State changes constantly over time

<u>Combined Simulation</u>
Contains state variables and algorithms of both types

Every simulation has a state set that defines the condition of the virtual world at a specific point in time. Each state variable contributes a small piece of the virtual world. These variables are included in the state set and are valuable only if they meet one of two criteria. (1) They must make some contribution to the output of the model. Or, (2) they must generate an event that contributes to the output of the simulation. Since state variables are essentially "invisible", their existence is only revealed when they contribute to the simulation output.

Notes

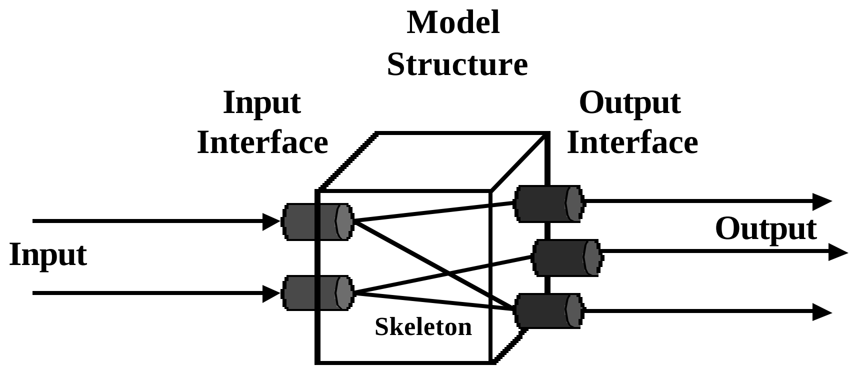
Every simulation has a "state set". This is the accumulation of every variable contained within each model and each object in the system. The "state" of the simulation at any time is then the accumulation of the settings of each of these variables at that time.

In strict discrete event simulations, the state of the simulation changes only at the point at which a specific event is processed. If an event is not processed, then the state of the system (including the time) is not changing.

In a continuous simulation, the algorithms define a continuous function for the changing of a model. This algorithm may be consulted at any time to know what value is appropriate, but the variables are constantly changing even if they are not being consulted.

Most simulations combine both of these concepts to create a system that addresses a specific problem. This is often referred to as a hybrid simulation.

Code the Interfaces



Notes

The interfaces between modules are what allow the entire system to pass and process information. It is a good idea to create the software from “the outside-in”. This allows you to settle issues of data exchange with other components while the software is till simple. Once the interfaces are created, the components can be connected together.