Preview

Thursday, June 29, 2023 9:56 PM

ModelingComplexSystems.net
Collection: Model Construction
Title: Methodology
Section: Drug Levels Model.Preview

I'll have grounds more relative than this. The play's the thing wherein I'll catch the conscience of the king.

William Shakespeare
(Hamlet)

The process of building a dynamic model has several ordered phases. We will briefly describe several of those phases here. The following sections will provide a complete example of the finished product for each of these.

Problem Statement
State the problem you would like to solve. Add any pertinent information that will help define the problem.

Causal Inference Diagram
Construct a diagram — called a *Causal Inference Diagram* — that illustrates a logical description of the system wherein the problem has arisen. Be sure to illustrate any feedback loops that may exist in the system.

Flow Diagram
Construct a diagram — called a *Flow Diagram* — that illustrates the network of relations that comprise the system. Pay particular attention to those relations that determine the state variables of the system. Typically state variables are noticeable for their ability to influence the values of other more transitory variables scattered throughout the system. An example of a state variable is the oil level associated with a four cycle internal combustion engine. In fact we give the name *Level* to any of the variables we have identified as state variables in a system.

Code
Write the code that links all the variables in your system together based on your flow diagram. Each variable identified in your flow diagram will define a class (and an object) in an object oriented paradigm. As an additional phase, you could in fact draw what is called a *UML class diagram*[1] to help define the code properties and the relationships between the variables. In later versions of our *Methodology* wib we may add the drawing of a UML class diagram to our ordered phases, if it helps

improve the ability of the user to write the code.

The code we show at this stage is Java, and we don't show all the classes involved. When we pursue the actual construction of the code in the following sections, we will work through each class both in Java and in Python.

We would like to make the point here that you are writing code for your own personal use. You are not designing it to be handed off to anyone. Thus it need not be "bulletproof" (in the vernacular of computer science). You should, however, take the necessary cautions to protect yourself from yourself. Once you have solved the problem of interest, you can hand the code off to someone who knows how to write code intended for users who are only running the executable, and do not have access to the code files themselves. This will cut your coding time in half, at least. And you can concentrate on learning code basics, like *functional programming* techniques, rather than dealing with all the other intricacies of code development for release.

Data Generated

You can generate a lot of data with a dynamic model. We thought you would like to see an example. Sorry about that.

Data Analysis

We show the results of an analysis of a small fraction of our data, which is a plot of the drug levels over the time horizon specified. Just what the doctor ordered.

This plot was done using Microsoft Excel. But you can create the same type of plot with any data hungry app you have handy. "Oh", but you say with dismay, "no graphics built in?" If you can match the graphics capability of Excel or MATLAB in zero time with your own code, go write ahead.

We skip the verification of our results at this time because, well, we don't want to give away the plot. In any case that's how Hamlet trapped Claudius (the king): by exposing him to a plot he couldn't resist.

References

1. Russ Miles & Kim Hamilton
Learning UML 2.0