



# Guidelines for Simulation Design



## Guidelines for the Design of Military Simulations



Roger Smith

smithr@modelbenders.com



**1999 Modeling, Simulation, and Gaming of Warfare Course**



## Notes

This presentation tries to capture some of the universal guidelines and principles that are necessary for constructing a simulation system. The ideas are extracted from the experience and writings of about a dozen simulation professionals.

This presentation is available electronically at:

<http://www.modelbenders.com/>

# Guidelines for Simulation Design

## Creating a Digital World



## Notes

Simulations are a digital version of some aspects of the real world. We seek to capture only those details that are important for the purposes of the simulation, and to represent these with models that interoperate with each other in a consistent manner.

The digital world is always a sub-set of the real world.

The digital world is always an approximation of the real world.

A perfect model of the real world would be another real world.

A simulation contains variables that can be set many ways, only some of which match the real world. Therefore, the simulation itself is more complex than the real world for those variables that it includes. Though the simulation is a sub-set of the real world, it is a more complex sub-set.

# Guidelines for Simulation Design

## Simulation is an Art Form



## Notes

Creating a simulation is much like creating large works of art. It requires the participation of many people with different capabilities.

Military people provide the real system.

Archeologists dig out details of the real system.

Mathematicians and researchers discover algorithms for representing the system.

Designers provide the vision and direction for the model.

Artists create the best arrangements of the pieces to create or display the model.

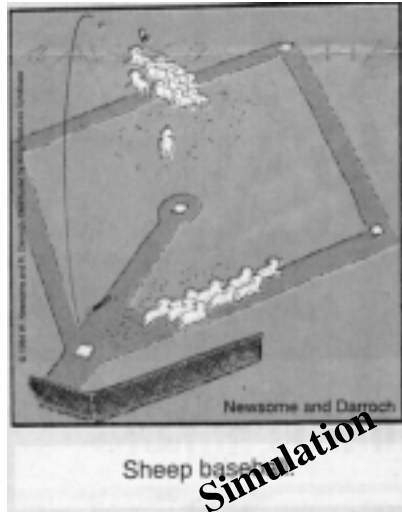
Programmers create the software that drives the whole thing.

The dynamics that occur between these people are unpredictable and uncontrollable. As a result the finished product is always a combination of the artistic abilities of all of these people.

You never know exactly how it is going to turn out.

# Guidelines for Simulation Design

## Don't Be a Sheep



You do not have to behave like sheep when designing and building a simulation.

Innovation and artistic license are valuable skills in this field.

## Notes

There is a tendency in all fields to chase the latest, newest, most widely talked about technologies. However, most fields require a great deal of imagination, creativity, and self confidence to make really useful progress. Even though everyone in your project or office is chasing the latest buzzword there are still many interesting and valuable problems to solve that no one is talking about - yet.

We want to encourage you to think for yourself and believe in your own abilities to identify important technologies and to think critically about the items that are attracting the masses. Don't be a sheep.

# Guidelines for Simulation Design

## Models



1945 Monogram Balsa Ships



1903 Ford Model A



Role Model



Clothing Model



Architectural Model



“I had been active in modeling the bill and securing its passage.”  
- Benjamin Franklin

## Notes

The term “model” can be used for so many ideas in the English language. The breadth of its application is a statement about how active our society is in different forms of modeling. But, it also makes communication on the topic difficult to standardize.

As children most of us were introduced to toy models that we put together with glue.

Every car on the road is defined by the model name just as this Model A Ford automobile was in 1903.

History is defined by people who are seen as role models - those whom we are encouraged to emulate.

Clothing, sunglasses, and hundreds of other items are presented through the use of models - these being people with the ideal body, hair, or features for using the product being sold.

Architects create models of buildings before constructing them.

# Guidelines for Simulation Design

## Models



Artists Model



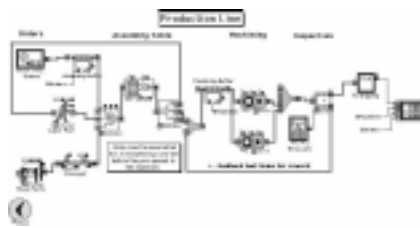
Toy Model



Mathematical Model



Software Model



Process Model



Vehicle Models

## Notes

When artists create pictures, sculptures, and other artifacts they refer to the copied object as a model. This emphasizes the desire to copy or be like the original object.

Hundreds of varieties of toys are models of real world items - soldiers, cars, trains, etc.

Mathematicians have created graphics that illustrate the relationships between variables in an equation. This picture displays an equation of many different variables in the form of a work of art.

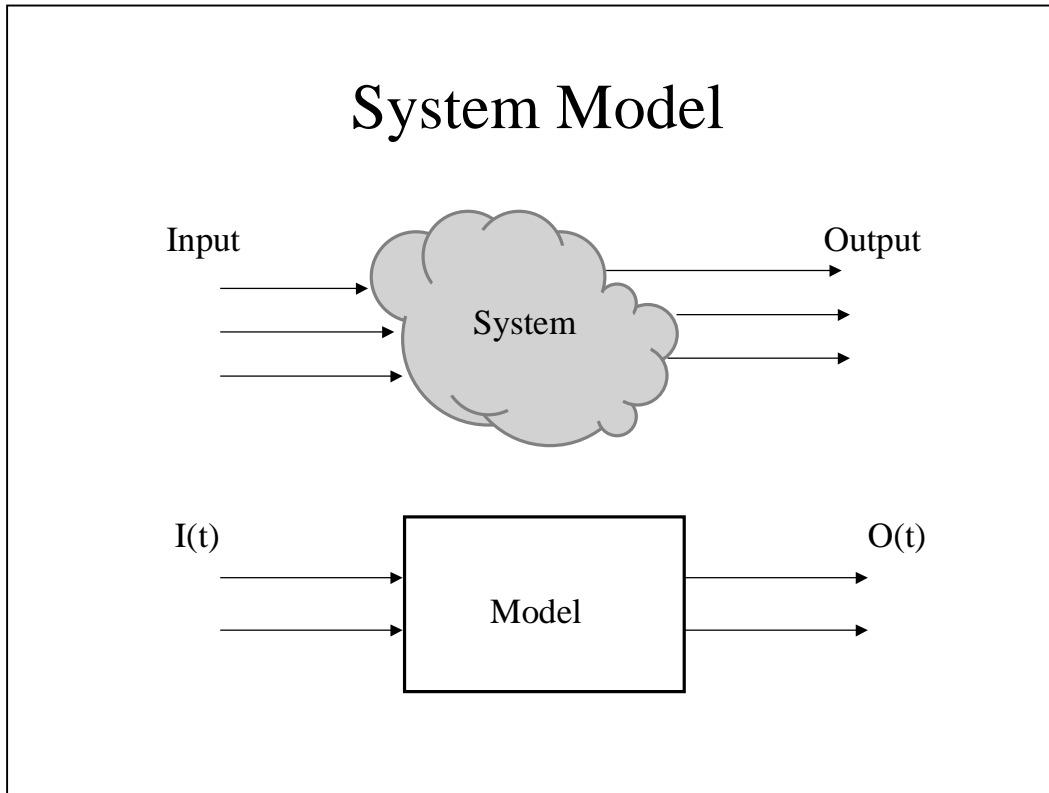
Software is now created through the use of modeling tools which allow designers to capture and manipulate ideas graphically rather than as a programming language.

Manufacturing models represent the performance and capabilities of a production line.

The computer graphics world refers to the 3D objects they display on the screen as models.

Discussions of models may refer to any one of these applications - or others that have not been given here.

# Guidelines for Simulation Design



## Notes

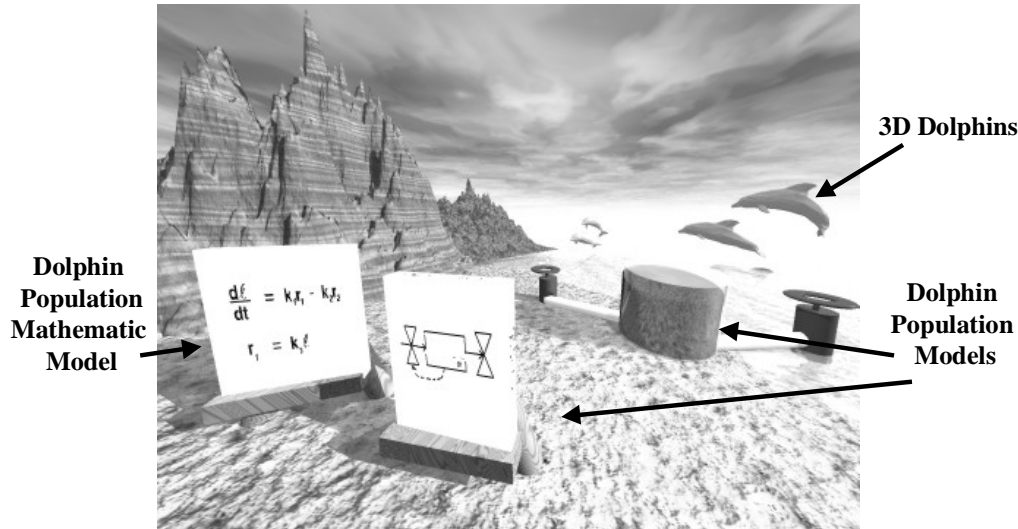
A model is an abstraction that behaves somewhat like a defined “system”. In the real world a system is a set of transformations that convert input data into output data. This general definition covers everything from the operations of a computer to the process of driving nails. The key being that it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

If those algorithms are defined they can be captured in some abstract form as a “model”. By definition the model is less exact, 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 the Outputs match the outputs of the real system - given corresponding input values.

# Guidelines for Simulation Design

## Model Abstraction



Picture courtesy of Paul Fishwick, University of Florida

## Notes

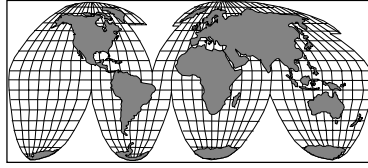
Any system can be represented at multiple levels of abstraction. The most natural and accessible is a 3D visual representation as shown by the models of jumping dolphins. This is most commonly found in television animation and other visual media.

However, some dolphin behaviors can not be captured with a jumping animation, like the birth and death rates of the species. The Tank-and-Valve on the beach is a representation of the birth and death rates of dolphins. This governs the size of the dolphin population without representing individual reproductive behaviors. The paper easel on the right is a static model diagram of the tank-and-valve model.

The paper easel on the left is the mathematical definition of the birth and death rates and their effect of the population over time. This mathematical model is more abstract, but can also capture behaviors at a more detailed level than can be represented in 3D.



## Mapping ... Modeling



“Even now...maps are not true pictures of reality. Each map is a product of compromises, omissions, and interpretations. Even a good map tells a multitude of little white lies.”

- Mark Monmonier, Syracuse University  
*National Geographic*, February 1998

## Notes

Mapping is a form of modeling. Professor Mark Monmonier emphasizes the compromises that are made in a map when he says

“Even now maps are not true pictures of reality. Each map is a product of compromises. Omissions, and interpretations. Even a good map tells a multitude of little white lies.”

Like a map - no model is a perfect representation of the system it models. However, these imperfections are completely normal, reasonable, acceptable, and intentional. Each model has a purpose which requires some details from the real world, but does not require others.

## Software ... Modeling

“Software development is not a ‘true’ engineering discipline based on well understood physical principles; good software has a large component of artistry and craftsmanship behind it.”

- Karen Mackey  
*IEEE Software*, January 1998

### Notes

Professor Karen Mackey emphasizes that software is not JUST an engineering discipline but is also a work of art. She says:

“Software development is not a ‘true’ engineering discipline based on well understood physical principles; good software has a large component of artistry and craftsmanship behind it.”

Similarly, modeling is as much an art form as it is a science. Every problem can be abstracted in infinitely many ways to create a model. Determining which abstraction is best comes from experience and artistic preference.

## Programming ... Modeling

“Universities should not teach students how to program in Java, C++, Pascal, or any other language. We should provide our students with a set of core, transferable programming skills that they will be able to adapt, as necessary, to the new programming languages that will inevitably appear during their careers.”

- Michael Pont  
*IEEE Software*, January 1998

## Notes

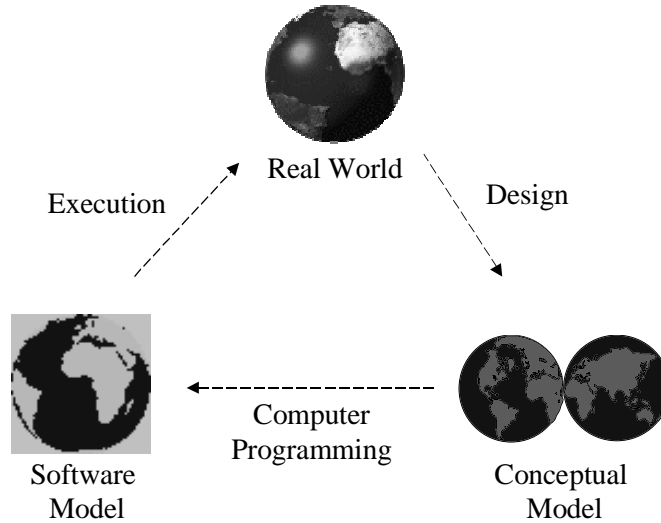
Professor Michael Pont expresses a point that is as true of modeling as it is of programming:

“Universities should not teach students how to program in Java, C++, Pascal, or any other language. We should provide our students with a set of core, transferable programming skills that they will be able to adapt, as necessary, to the new programming languages that will inevitably appear during their careers.”

Good modelers possess the skills of abstraction, defining limits, and identifying relationships. These skills are far more universally applicable than the ability to build models with specific tools like UML, OMT, MODSIM, GPSS, and SLAM.

# Guidelines for Simulation Design

## Modeling is a 3 Step Process



## Notes

### **Modeling is a three step process.**

It begins with some subset of the real world which is defined as the system to be modeled. This subset is transformed through the process of Design in a Conceptual Model of the system. This Conceptual Model captures the characteristics and behaviors of the original system that are essential for the study or exercise to which the model will be applied.

The Conceptual Model is transformed through the process of computer programming into a Software Model. The software model must accurately capture and reproduce the details captured in the Conceptual Model.

The process of execution generates the outputs or behaviors that we hoped to capture from the real world. These outputs should therefore correspond to some known set of behaviors in the real world.

[NOTE: This was first proposed in 1976 by Bernard Zeigler in his book *Theory of Modelling and Simulation*. The concept was later used by Bob Sargent to organize the processes of VV&A. ]

## 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 ...”



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

## Notes

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

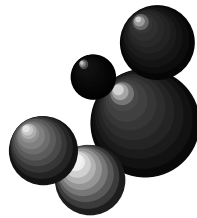
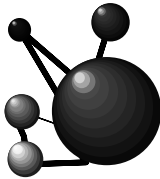
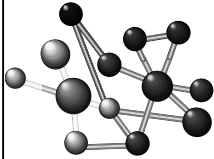
In 1975 Robert Shannon published a fact that has haunted every novice model builder:

“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.

# Guidelines for Simulation Design

## Appropriate Detail



- Virtual Simulator
- Semi-Automated Forces
- Constructive Simulation
- Board Wargame
- Spreadsheet Model
- Computer Game

## Notes

Military combat has been modeled in hundreds of different ways. Each approach focuses on a different problem and therefore arrives at a different level and form of abstraction. Any one of these may be the right solution to the problem you have. The level of detail required is driven by the problem to be solved, not by the computer power available or the budget of the program.

## Golden Rule of Modeling™

A model has no inherent value of its own.™  
The value of a model is based entirely upon the degree to which it solves someone's real-world problem.



© Copyright 1997, Roger Smith

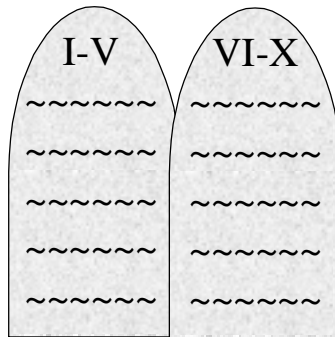
## Notes

**A model has no inherent value of its own. The value of a model is based entirely upon the degree to which it solves someone's real-world problem.**

This is one of the most important concepts you will encounter in this class. When creating a Software Model you must keep your eye on the Conceptual Model. When creating the Conceptual Model you must keep your eye on the problem to be solved. Hopefully, the problem has been accurately bounded and identified in the subset of the real world that you are extracting.

# Guidelines for Simulation Design

## Ten Commandments of Modeling



### Notes

We are about to present 10 Commandments of Modeling. These are rules-of-thumb or wise practices that have been gathered from a large audience of experienced model developers. These commandments illustrate practices that should be a part of any modeling project.



## 10 Commandments

### ① Problems First, Answers Second

- > Do not accept any predefined solutions to the problem
- > It is impossible to get the right answer before you know what the question is
- > Ask questions, identify the problem that needs to be solved
- > Design an efficient answer



© Copyright 1997, Roger Smith

## Notes

Many projects come equipped with pre-formulated answers. This is the fastest way to start down the wrong path. The purpose for building a model is often to answer a complex problem. Understanding that problem involves considerable study and mental investment. If you already have the answer, then either it is a simple problem, or you do not know what the problem is.

A quote from *Alice's Adventures in Wonderland*, by Lewis Carroll is very appropriate here:

**Alice:** "Would you tell me, please, which way I ought to go from here?"

**Cheshire Cat:** "That depends a good deal on where you want to get to."

**Alice:** "I don't much care where - so long as I get somewhere."

**Cheshire Cat:** "Then it doesn't matter which way you go."

# Guidelines for Simulation Design

## 10 Commandments

### ② Learn from the Legacy

- Know and use existing theory of modeling
- Study old models of the same or similar systems
- Understand why previous generations selected specific solutions - follow their yellow brick road
- Learn what they did wrong - avoid their pit of despair



© Copyright 1997, Roger Smith

## Notes

The first wargame was built in 1664, the first flight simulator in 1930, the first virtual world in the 1960's. The simulation community has wrestled with problems similar to yours and arrived at very valuable solutions. You need to learn from all of the mental and physical work that has gone into previous systems.

“Those who do not know the past are destined to repeat it.”

## 10 Commandments

### ③ Create a Conceptual Model

- Use analytical model to identify important factors
- Identify system interactions and dependencies
- Completely traverse the problem in design, not in code
- Locate computational black holes



© Copyright 1997, Roger Smith

## Notes

A virtual world is a very complex thing. It is difficult to envision all of the pieces and interactions that will exist in it. A formal conceptual model captures the elements, events, modifiers, and functions that will operate in the system. Without this, the model will turn out lop-sided at best and inoperable at the worst.

## 10 Commandments

### ④ Build a Prototype

- > “Large successful systems come from small successful systems.”
  - Bill Joy, Sun Microsystems
- > Aim prototype at the 80:20 rule
  - 80% of the functionality is 20% of the code



© Copyright 1997, Roger Smith

## Notes

It is extremely difficult for any team to build a large system that works right. The level of complexity is just too large to control. A prototype can delve into this complexity in the most essential places and allow you to understand and control it.

Since any model spends 80% of its time executing 20% of the code. The prototype should dig deeply into this core 20%.

## 10 Commandments

### ⑤ Push the User's Hot Buttons

- Involve users in development
- Create an intuitive and seductive user interface
- Some features are essential for user acceptance
- Imbue the model with "face validity"



© Copyright 1997, Roger Smith

## Notes

Every user or customer has special Hot Buttons that have to be pushed. You must uncover these and highlight them in your model and your prototype.

To find those Hot Buttons you must spend time listening to the users. Do not rely solely on their written specifications. Listen to their stories about past experiences, successes, and failures. Come to understand WHY they are asking for certain features. Recognize the deadly flaws of previous models and do not repeat them.

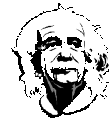
# Guidelines for Simulation Design

## 10 Commandments

### ⑥ Simplify, Simplify

- > “essentia non sunt multiplicanda praeter necessitatem”
- > “hypotheses are not to be complicated without necessity”
  - Occam’s Razor, 1320AD - Sir William of Occam
- > “Everything should be made as simple as possible - but no simpler!”

□ Albert Einstein



© Copyright 1997, Roger Smith

## Notes

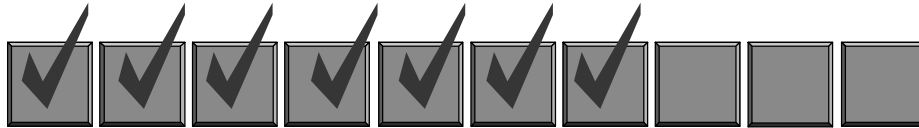
Everyone from the customer to the programmer is eager to add as many interesting and impressive details as possible. Each will profess eloquently that their entail is essential to the simulation. The model has a mission and a goal - stick to it.

# Guidelines for Simulation Design

## 10 Commandments

### ⑦ Start with the Objective, Mission, and User

- > Before describing what you will build, you have to describe why you are building it.
- > Who will use it and what will they do with it?
- > What experience must it convey?
- > What lessons must it teach?



© Copyright 1997, Roger Smith

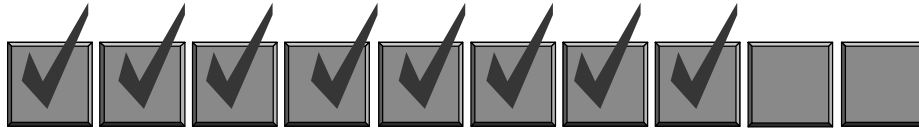
## Notes

When you get to the software design phase, it is always tempting to define the objects first. However, each simulation provides a unique approach to the problem. The objective of the simulation, the missions performed by the scenario, and the interactions with the user will drive any list of objects you are tempted to create. You must first define why the simulation is to exist, before you can define what it will include.

## 10 Commandments

### ⑧ Interactions Before Objects

- To achieve the mission for the simulation, what interactions must occur within the simulation?
- Define the interactions with the users and the implications those make on simulation events.
- Do not define your objects until you have defined primary interactions with the user and secondary interactions within the simulation.



© Copyright 1997, Roger Smith

## Notes

Once you know who the users are and what the mission of the simulation is, you can identify the interactions that the user will have with the simulation - these are primary. Interactions with the user lead to the internal interactions which support those - these are secondary.

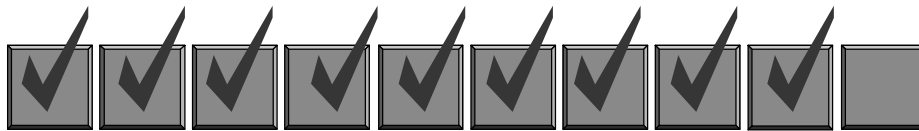
After the external and internal interactions are defined, then you can decide what objects are necessary to accomplish those interactions.



## 10 Commandments

### ⑨ Algorithms and Attributes Last

- > The defined interactions can be modeled with many different algorithms
- > The algorithms will drive the attributes needed to support them



© Copyright 1997, Roger Smith

## Notes

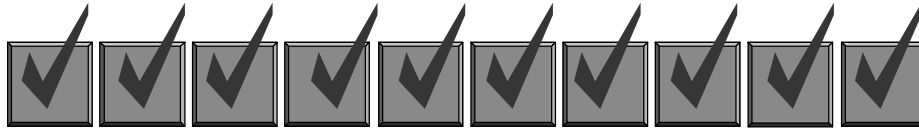
Each interaction can be modeled with many different algorithms. The mission, interactions the the user, and internal interactions drive the design of a specific algorithm to support them. Algorithm selection will also define the attributes assigned to an object.

Every real world object has thousands or millions of attributes. It is impossible to include all of those in a simulation. The algorithms define the attributes required.

## 10 Commandments

### ⑩ Trust Your Creative Juices

- Creativity and energy are primary ingredients of a simulation system
- Creating models and simulations is a combination of science and art - discipline and imagination
- Not following your creativity will stagnate a project



© Copyright 1997, Roger Smith

## Notes

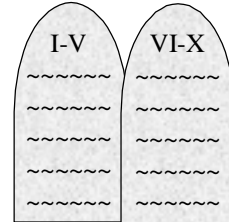
When working with a new team that has not created a simulation before, they are afraid to move forward without explicit direction on what they should build. They are afraid that they will head off in the wrong direction and create a product that others will criticize. The fear of making mistakes results in constant repetitive requirements analysis, organizational alignment, product research, process definition, and a hundred other non-modeling tasks. The team avoids making concrete decisions about the design of the product.

Experienced members of the team must demonstrate, instill, and encourage the brave act of trusting your own creative juices. The team leaders must provide the vision for the entire product, but each programmer, designer, and artists must have the freedom and confidence to express what they see in the product.

# Guidelines for Simulation Design

## 10 Commandments of Modeling

- ✓ Problems First, Answers Second
- ✓ Learn from the Legacy
- ✓ Create a Conceptual Model
- ✓ Built a Prototype
- ✓ Push the User's Hot Buttons
- ✓ Simplify, Simplify
- ✓ Start with the Objective, Mission, and User
- ✓ Interactions Before Objects
- ✓ Attributes and Algorithms Last
- ✓ Trust Your Creative Juices



© Copyright 1997, Roger Smith

## Notes

The 10 Commandments of Modeling have been derived for the experiences of a large family of simulation developers. Following them will increase your productivity and improve your chances of creating a successful model. Violating them will give you the opportunity to relearn these lessons the hard way.

## Modeling Checklist

### 1. Describe It

What is it for? Why build it?

### 2. Touch a Real One

Size, color, connections, etc.

### 3. Use a Real One

Feel, motion, process

### 4. Define the Interactions

Input, output, dependencies

### 5. Design Algorithms

Physical, behavioral,  
environmental

### 6. Define Hot Buttons

What sells?

### 7. Code the Interface

Isolate methods, data

### 8. Build Core Code

Capture breadth, stub details

### 9. Dig Deeper

Code details, modular structure

### 10. Rework

Throw out bad ideas, add  
detail

### 11. Field Test

Take it to the field

### 12. Better Next Time

Learn lessons, move on

## Notes

This is a twelve-step checklist followed by Roger Smith to produce several successful simulation systems. It describes his personal process for creating and fielding a successful simulation system.

In some cases this checklist overlaps with the principles and commandments already presented.

# Guidelines for Simulation Design

## Laws of Data

- ❑ First Law: You can never get all of the data you need.
- ❑ Second Law: You can not use all of the data you can get.
- ❑ Third Law: You always have to collect some of the data yourself.
- ❑ Fourth Law: You always have to synthesize data to meet the needs of the model.



© Copyright Evans & Sutherland

## Notes

It is notoriously difficult to get good data about military combat. The combat environment is not conducive to the scientific method or scientific inquiry. Data collectors do their best to piece together what happened later based on anecdotes and evidence left behind. Data is also drawn from experimentation with weapons.

Given these limitations, the Laws of Data are:

- 1) You can never get all of the data you need.
- 2) You can not use all of the data you can get.
- 3) You always have to collect some of the data yourself.
- 4) You always have to synthesize data to meet the needs of the model.

The 4th Law has important ramifications on the VV&A process and the ability to get a model Accredited.

## Golden Rule of Modeling™

A model has no inherent value of its own.™  
The value of a model is based entirely upon the degree to which it solves someone's real-world problem.



© Copyright 1997, Roger Smith

## Notes

This is the most important concept in modeling and one of the most valuable lessons you can take away from this course.