MCA 504
Modelling and Simulation

Index

Subject : System Simulation and Modeling **Author : Jagat Kumar**
Paper Code: MCA 504 **Vetter : Dr. Pradeep Bhatia**
Lesson : System Models and System Simulation
Lesson No. : 01

Structure

1.0 Objective

The main objective of this module to gain the knowledge about system and its behavior so that a person can transform the physical behavior of a system into a mathematical model that can in turn transform into a efficient algorithm for simulation purpose.

1.1 Introduction

Computer simulation is a powerful methodology for design and analysis and complex systems. The overall approach in computer simulation is to represent the dynamic characteristics of a real world system in a computer model. The model is subjected to experiments to obtain predictive information useful in making informed decision making about the characteristics of the real system. Simulations are suitable for problems in which there are no closed-form analytical solutions. Since most dynamic problems in practice can not be represented and solved fully using mathematical equations, computer simulation is a powerful and flexible methodology in complex systems analysis.
Simulations can be classified into continuous and discrete simulations. In continuous simulations, the state variables, i.e., the collection of variables needed to describe the system, change continuously over time and the behavior of the system is typically described by differential equations. Examples of continuous systems include the modeling of thermal or hydraulic systems. Discrete simulations are event-driven where the state variables change at discrete time points. Examples of discrete-event simulations include service industry applications such as queues in a grocery store and manufacturing applications involving material flow analysis. In general we have three different methods as shown in Figure 1 to study a real system

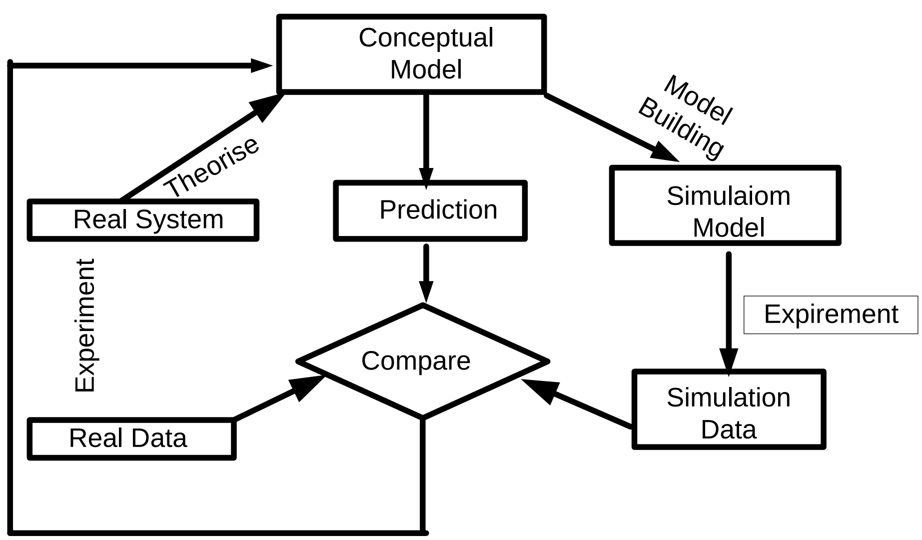

Figure 1 : Three Methods of Science

Briefly we can say that Simulation is
• Simulated system imitates operation of actual system over time
• Artificial history of system can be generated and observed
• Internal (perhaps unobservable) behavior of system can be studied
• Time scale can be altered as needed
• Conclusions about actual system characteristics can be inferred
in Figure 2 , actual system (real system) is compared with simulation

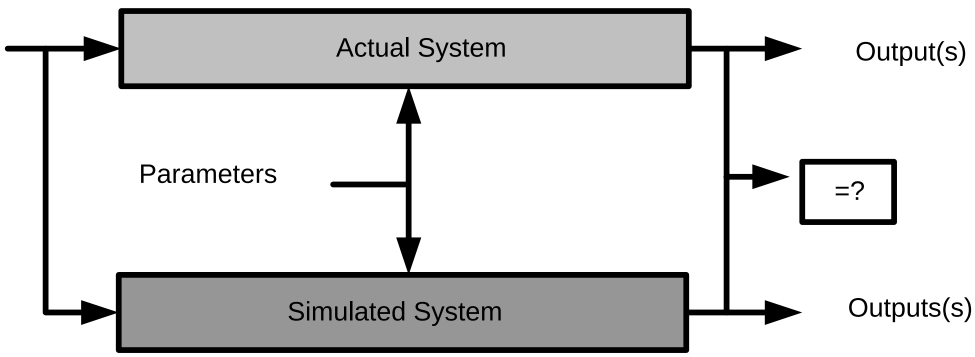

Figure 2: Simulation vs Actual System

1.1.1 Formal Definition(s)

Simulation can be broadly defined as *a technique for studying real-world dynamical systems by imitating their behavior using a mathematical model of the system implemented on a digital computer.*

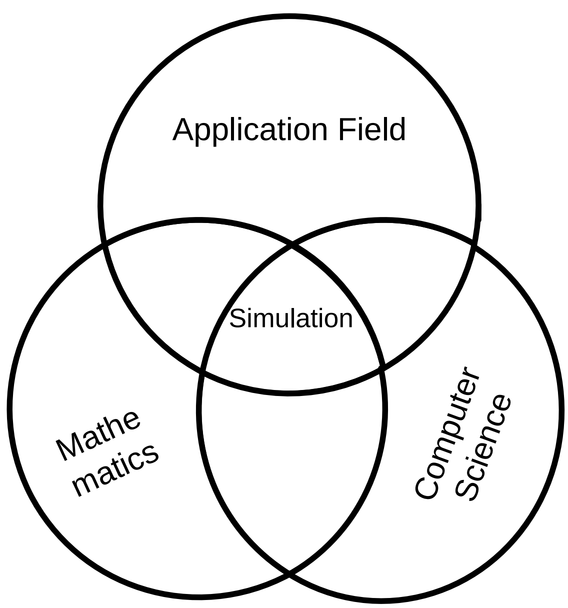

Figure 3: Simulation is Interdisciplinary

Simulation can also be viewed as *a numerical technique for solving complicated probability models*, ordinary differential equation and partial differential equation, analogously to the way in which we can use a computer to numerically evaluate the integral of a complicated function. That's why science of simulation is considered as an interdisciplinary subject a shown in Figure 3.

1.1.2 A Brief History of Simulation

1940's: Monte Carlo method is developed by physicists working on Manhattan project to study neutron scattering. Researchers   include John von Neumann, Stanis law Ulan, Edward Teller, Herman Kahn

1950's: First special-purpose simulation languages developed (e.g. IMSCRIPT by Harry Markowitz at RAND Institute)
1970's: Research initiated on mathematical foundations of simulation
1980's: PC-based simulation software developed, graphical user interfaces, object-oriented programming
1990's: Web-based simulation, fancy animated graphics, simulation-based optimization, Markov-chain Monte Carlo methods Simulation has become ever more prominent as a method for studying complex systems in which uncertainty is present. In various surveys, simulation has been found to be the most frequently used tool of Operation Research practitioners. Simulation is an interdisciplinary subject, using ideas and techniques from Statistics, Probability, Number Theory, and Computer Science.

1.1.3 Application Areas of Simulation

- Manufacturing
- Computer Systems
- E-business/workflow systems
- Finance
- Telecommunications
- Transportation
- Military

1.1.4 Advantages and Disadvantages of Simulation

Advantages: Simulation arbitrary model complexity, circumvents analytically intractable models, facilitates what-if and sensitivity analyses, building a model can lead to system improvements and greater understanding can be used to verify analytic solutions

Disadvantages: Simulation provides only estimates of solution, only solves one parameter at a time, can take a large amount of development and/or computer time ("simulation as a last resort"). Don't use computer simulation if a common-sense or analytical solution is available, or if resources are insufficient, or if simulation costs outweigh benefits.

1.1.5 Difficulties of Simulation

- Provides only individual, not general solutions
- Manpower and time-consuming
- Computing memory and time-intensive
- Difficult so experts are required
- Hard to interpret results
- Expensive

1.1.6 When to Use Simulation?

- Study internals of a complex system e.g. biological system
- Optimise an existing design e.g. routing algorithms, assembly line
- Examine effect of environmental changes e.g. weather forecasting
- System is dangerous or destructive e.g. atom bomb, atomic reactor, missile launching
- Study importance of variables
- Verify analytic solutions (theories)
- Test new designs or policies
- Impossible to observe/influence/build the system
- When it allows inspection of system internals that might not otherwise be observable
- Observation of the simulation gives insights into system behavior
- System parameters can be adjusted in the simulation model allowing assessment of their sensitivity (scale of impact on overall system behavior)
- Simulation verifies analysis of a complex system, or can be used as a teaching tool to provide insight into analytical techniques
- A simulator can be used for instruction, avoiding tying up or damaging an expensive, actual system (e.g., a flight simulation vs. use of multimillion dollar aircraft)

1.2 Modelling Concepts

There are several concepts underlying simulation. These include system and model, events, system state variables, entities and attributes, list processing, activities and delays, and finally the definition of discrete-event simulation.
The process of making and testing hypotheses about models and then revising designs or theories has its foundation in the experimental sciences. Similarly, computational scientists use **modeling** to analyze complex, real-world problems in order to predict what might happen with some course of action. For example, Dr. Jerrold Marsden, a computational physicist at CalTech, models space mission trajectory design (Marsden). Dr. Julianne Collins, a genetic epidemiologist (statistical genetics) at the Greenwood Genetics Center, runs genetic analysis programs and analyzes epidemiological studies using the Statistical Analysis Software (SAS) (Greenwood Genetics Center). Some of the projects on which she has worked involve analyzing data from a genome scan of

Alzheimer's disease, performing linkage analyses of X-linked mental retardation families, determining the recurrence risk in nonsyndromic mental retardation, analyzing folic acid levels from a nutritional survey of Honduran women, and researching new methods to detect genes or risk factors involved in autism. Scientists in areas such as cognitive psychology and social psychology at the Human-Technology Interaction Center of The University of Oklahoma perform research on the interaction of people with modern technologies (Human-Technology Interaction Center). Some of the studies involve "strategic planning in air traffic control" and "designing interfaces for effective information retrieval from collections of multimedia." Buried land mines are a serious danger in many areas of the world (Weldon et al. 2001). Scientists are using a combination of mathematics, signal processing, and scientific visualization to model, image, and discover land mines. Lourdes Esteva, Cristobal Vargas, and Jorge Velasco-Hernandez have modeled the oscillating patterns of the disease dengue fever, for which an estimated 50 to 100 million cases occur globally each year (Esteva and Vargas 1999).

1.2.1 System, Model and Events

A model is a representation of an actual system (Figure 4) and Figure 5 presents modelling and simulation concepts as introduced by Zeigler [2].

- A model is an abstraction of the real system
- Simplifying assumptions are used to capture (only) important behaviors
- Linearization, time-bound behaviors, etc., may make analysis tractable

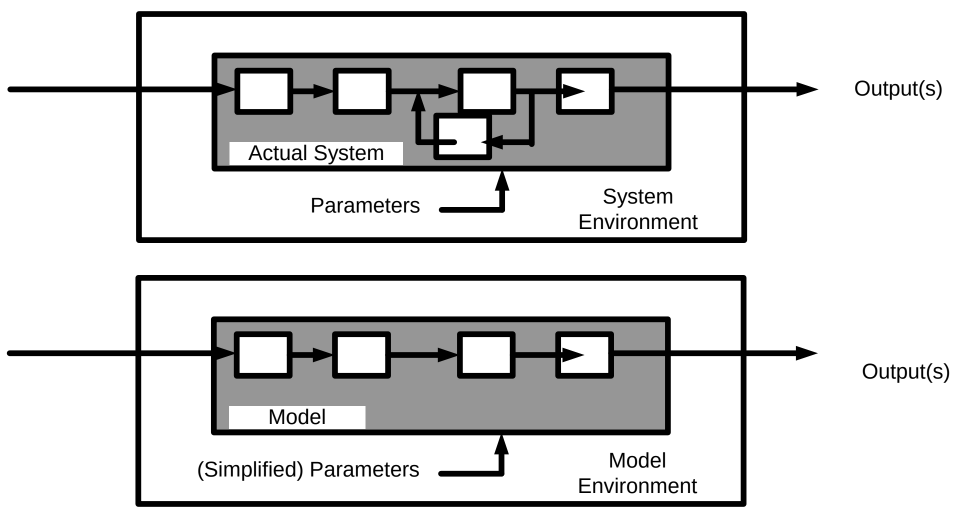

Figure 4 : Pictorial Representation of System Model

Formally we can define, Modeling is the application of methods to analyze complex, real-world problems in order to make predictions about what might happen with various actions.

Object is some entity in the Real World. Such an object can exhibit widely varying behavior depending on the context in which it is studied, as well as the aspects of its behavior which are under study.

Base Model is a hypothetical, abstract representation of the object's properties, in particular, its behavior, which is valid in *all* possible contexts, and describes all the object's facets. A base model is hypothetical as we will never in practice be able to construct/represent such a total model. The question whether a base model exists at all is a philosophical one.

System is a well defined object in the Real World under specific conditions, only considering specific aspects of its structure and behaviour.

Experimental Frame When one studies a system in the real world, the experimental frame (EF) describes experimental conditions (context), aspects, within which that system and corresponding models will be used. As such, the Experimental Frame reflects the *objectives* of the experimenter who performs experiments on a real system or, through simulation, on a model.

Immediately, there is a concern about the limits or boundaries of the model that supposedly represent the system. The model should be complex enough to answer the questions raised, but not too complex. Consider an event as an occurrence that changes the state of the system. In the example, events include the arrival of a customer for service at the bank, the beginning of service for a customer, and the completion of a service. There are both internal and external events, also called endogenous and exogenous events, respectively. For example, an endogenous event in the example is the beginning of service of the customer since that is within the system being simulated. An exogenous event is the arrival of a customer for service since that occurrence is outside of the simulation. However, the arrival of a customer for service impinges on the system, and must be taken into consideration.

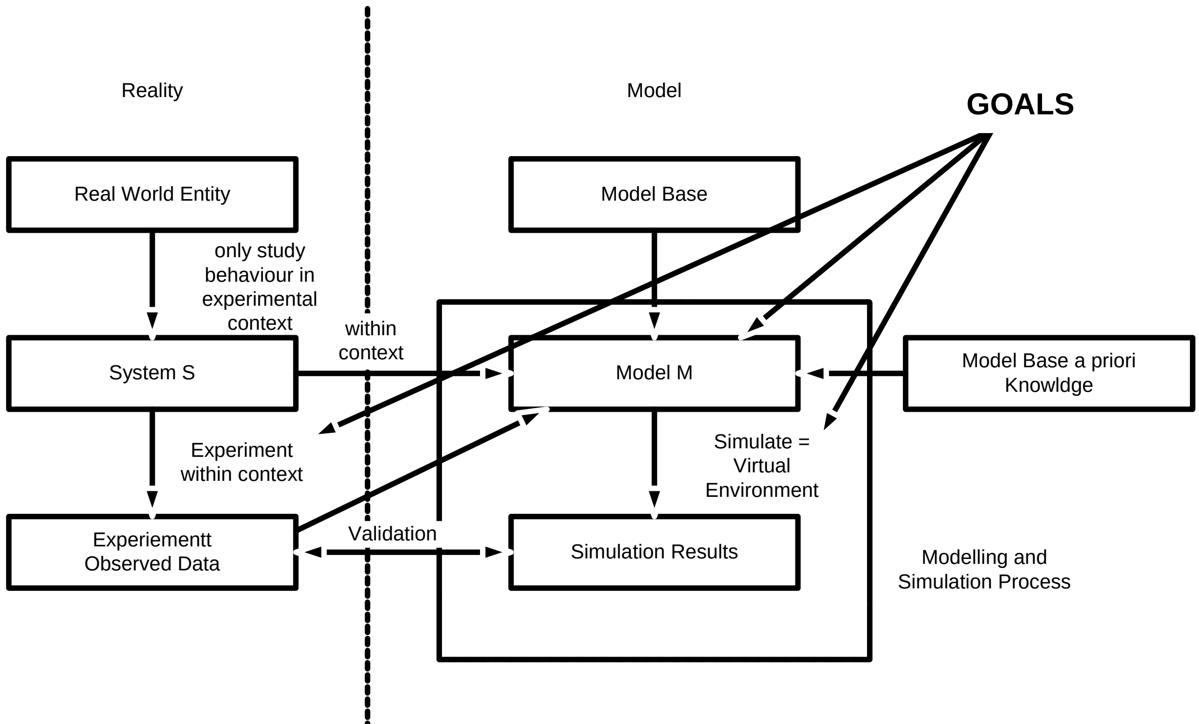

Figure 5: Modeling and Simulation

Discrete-event simulation models are contrasted with other types of models such as mathematical models, descriptive models, statistical models, and input-output models. A discrete-event model attempts to represent the components of a system and their interactions to such an extent that the objectives of the study are met. Most mathematical, statistical, and input output models represent a system's inputs and outputs explicitly, but represent the internals of the model with mathematical or statistical relationships. An example is the mathematical model from physics,

Force = Mass × Acceleration

based on theory. Discrete-event simulation models include a detailed representation of the actual internals.

Discrete-event models are dynamic, i.e., the passage of time plays a crucial role. Most mathematical and statistical models are static in that they represent a system at a fixed point of time. Consider the annual budget of a firm. This budget resides in a spreadsheet. Changes can be made in the budget and the spreadsheet can be recalculated, but the passage of time is usually not a critical issue. Further comments will be made about discrete-event models after several additional concepts are presented.

Models have Many Uses, Typically

- To understand the behaviour of an existing system (why does my network performance die when more than 10 people are at work?)

- To predict the effect of changes or upgrades to the system (will spending 100,000 on a new switch cure the problem?)
- To study new or imaginary systems (let's bin the Ethernet and design our own scalable custom routing network)

1.2.2 System State Variables

The system state variables are the collection of all information needed to define what is happening within the system to a sufficient level (i.e., to attain the desired output) at a given point in time. The determination of system state variables is a function of the purposes of the investigation, so what may be the system state variables in one case may not be the same in another case even though the physical system is the same.
Determining the system state variables is as much an art as a science. However, during the modeling process, any omissions will readily come to light. (And, on the other hand, unnecessary state variables may be eliminated.) Having defined system state variables, a contrast can be made between discrete-event models and continuous models based on the variables needed to track the system state. The system state variables in a discrete-event model remain constant over intervals of time and change value only at certain well-defined points called event times. Continuous models have system state variables defined by differential or difference equations giving rise to variables that may change continuously over time.
Some models are mixed discrete-event and continuous. There are also continuous models that are treated as discrete-event models after some re-interpretation of system state variables, and vice versa.

1.2.2.1 Entities and Attributes

An entity represents an object that requires explicit definition. An entity can be dynamic in that it "moves" through the system, or it can be static in that it serves other entities. In the example, the customer is a dynamic entity, whereas the bank teller is a static entity. An entity may have attributes that pertain to that entity alone. Thus, attributes should be considered as local values. In the example, an attribute of the entity could be the time of arrival. Attributes of interest in one investigation may not be of interest in another investigation. Thus, if red parts and blue parts are being manufactured, the color could be an attribute. However, if the time in the system for all parts is of concern, the attribute of color may not be of importance. From this example, it can be seen that many entities can have the same attribute or attributes (i.e., more than one part may have the attribute "red").

1.2.2.2 Resources

A resource is an entity that provides service to dynamic entities. The resource can serve one or more than one dynamic entity at the same time, i.e., operate as a parallel server. A dynamic entity can request one or more units of a resource. If denied, the requesting entity joins a queue, or takes some other action (i.e. diverted to another resource, ejected from the system). (Other terms for queues include files, chains, buffers, and waiting

lines.) If permitted to capture the resource, the entity remains for a time, then releases the resource.
There are many possible states of the resource. Minimally, these states are idle and busy. But other possibilities exist including failed, blocked, or starved.

1.2.2.3 List Processing

Entities are managed by allocating them to resources that provide service, by attaching them to event notices thereby suspending their activity into the future, or by placing them into an ordered list. Lists are used to represent queues. Lists are often processed according to FIFO (first-in first-out), but there are many other possibilities. For example, the list could be processed by LIFO (last-in-first out),according to the value of an attribute, or randomly, to mention a few. An example where the value of an attribute may be important is in SPT (shortest process time) scheduling. In this case, the processing time may be stored as an attribute of each entity. The entities are ordered according to the value of that attribute with the lowest value at the head or front of the queue.

1.2.2.4 Activities and Delays

An activity is duration of time whose duration is known prior to commencement of the activity. Thus, when the duration begins, its end can be scheduled. The duration can be a constant, a random value from a statistical distribution, the result of an equation, input from a file, or computed based on the event state. For example, a service time may be a constant 10 minutes for each entity, it may be a random value from an exponential distribution with a mean of 10 minutes, it could be 0.9 times a constant value from clock time 0 to clock time 4 hours, and 1.1 times the standard value after clock time 4 hours, or it could be 10 minutes when the preceding queue contains at most four entities and 8 minutes when there are five or more in the preceding queue. A delay is an indefinite duration that is caused by some combination of system conditions. When an entity joins a queue for a resource, the time that it will remain in the queue may be unknown initially since that time may depend on other events that may occur. An example of another event would be the arrival of a rush order that preempts the resource. When the preempt occurs, the entity using the resource relinquishes its control instantaneously. Another example is a failure necessitating repair of the resource. Discrete-event simulations contain activities that cause time to advance. Most discrete-event simulations also contain delays as entities wait. The beginning and ending of an activity or delay is an event.

1.2.3 Model Classifications

Several classification categories for models exist. A system we are modeling exhibits **probabilistic** or **stochastic behavior** if an element of chance exists. For example, the path of a hurricane is probabilistic. In contrast, a behavior can be **deterministic**, such as the position of a falling object in a vacuum. Similarly, models can be deterministic or probabilistic. A **probabilistic** or **stochastic model** exhibits random effects, while a **deterministic model** does not. The results of a deterministic model depend on the initial conditions; and in the case of computer implementation with particular input, the output is the same for each program execution. As we studied this and other modules, we can have a probabilistic model for a deterministic situation, such as a model that uses random numbers to estimate the area under a curve .Figure 6 is depicted the classification of different kinds of models.

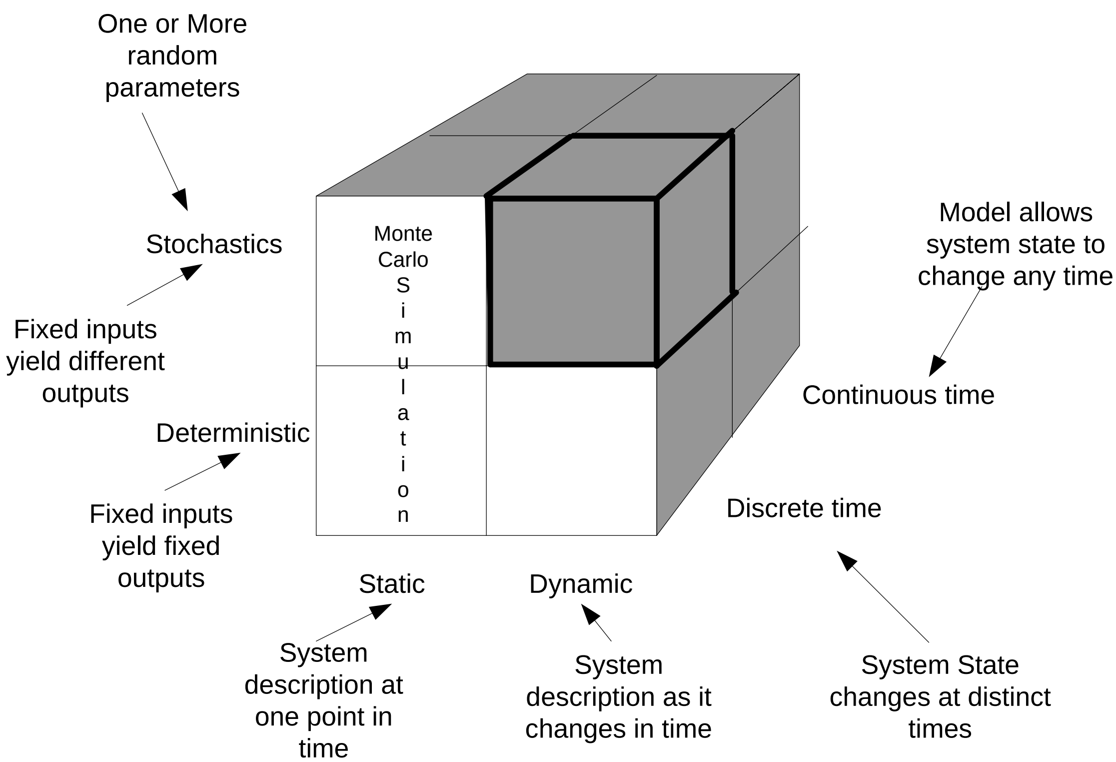

Figure 6: Classification of Different Types of Model

1.2.3.1 Discrete-Event Simulation Model

Sufficient modeling concepts have been defined so that a discrete event simulation model can be defined as one in which the state variables change only at those discrete points in time at which events occur. Events occur as a consequence of activity times and delays. Entities may compete for system resources, possibly joining queues while waiting for an available resource. Activity and delay times may "hold" entities for durations of time. A discrete-event simulation model is conducted over time ("run") by a mechanism that moves simulated time forward. The system state is updated at each event along with capturing and freeing of resources that may occur at that time.

1.2.3.2 Stochastic and Deterministic Systems

Definitions A system exhibits **probabilistic** or **stochastic behavior** if an element of chance exists. Otherwise, it exhibits **deterministic behavior**. A **probabilistic** or **stochastic model** exhibits random effects, while a **deterministic model** does not.

Deterministic: Randomness does not affect the behaviour of the system. The output of the system is not a random variable.
Stochastic: Randomness affects the behaviour of the system. The output of the system is a random variable.

1.2.3.3 Static and Dynamic Simulations

We can also classify models as static or dynamic. In a **static model**, we do not consider time, so that the model is comparable to a snapshot or a map. For example, a model of the weight of a salamander as being proportional to the cube of its length has variables for weight and length, but not for time. By contrast, in a **dynamic model**, time changes, so that such a model is comparable to an animated cartoon or a movie. For example, the number of salamanders in an area undergoing development changes with time; and, hence, a model of such a population is dynamic. Many of the models we consider in this text are dynamic and employ a static component as part of the dynamic model.
Definitions A **static model** does not consider time, while a **dynamic model** changes with time.

Static: A simulation of a system at one specific time, or a simulation in which time is not a relevant parameter for example, Monte Carlo & steady-state simulations.
Dynamic: A simulation representing a system evolving over time for examples, the majority of simulation problems.

1.2.3.4 Discrete vs. Continuous Systems

When time changes continuously and smoothly, the model is **continuous**. If time changes in incremental steps, the model is **discrete**. A discrete model is analogous to a movie. A sequence of frames moves so quickly that the viewer perceives motion. However, in a live play, the action is continuous. Just as a discrete sequence of movie frames represents the continuous motion of actors, we often develop discrete computer models of continuous situations .
Definitions In a **continuous model**, time changes continuously, while in a **discrete model** time changes in incremental steps.

Continuous: State variables change continuously as a function of time (Figure 7) and generally analytical method like deductive mathematical reasoning is used to define and solve the system.

State Variable (S.V.) = f (t)

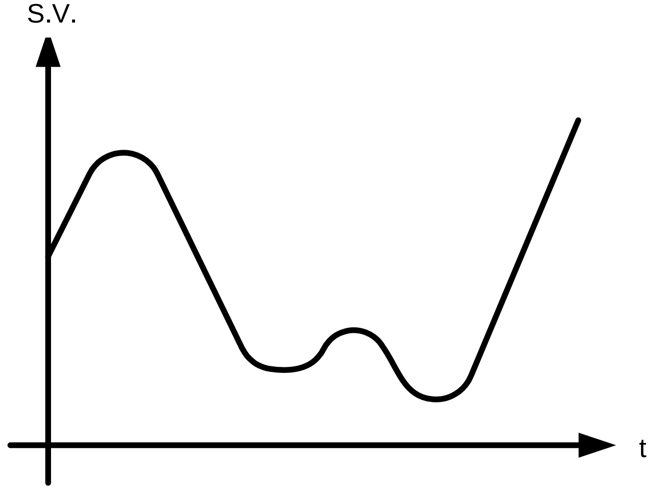

Figure 7:Continous Model behavior is shown

Discrete: State variables change at discrete points in time (Figure 8) and generally numerical method like computational procedures is used to solve mathematical models.

State Variable(S.V.) = f(n t)

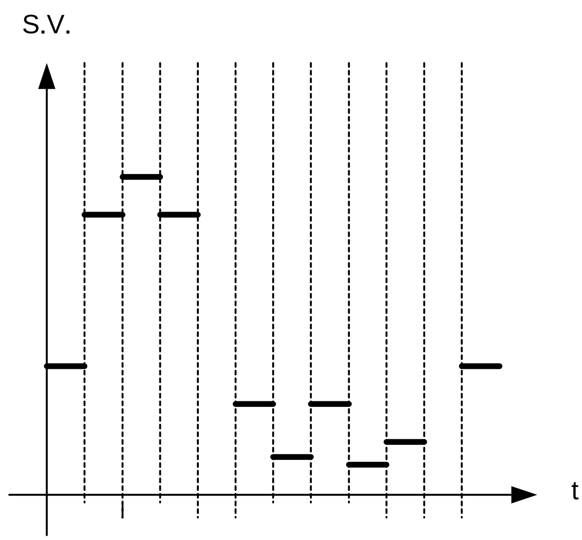

Figure 8: Discrete Model behavior is shown

Examples of Different Systems

- Queue length at a cash machine: Stochastic, Discrete Time, Discrete System
- The motion of the planets: Deterministic, Continuous Time, Discrete System
- Logic circuit in a computer: Deterministic, Discrete Time, Discrete System

- Flow of air around a car: Deterministic, Continuous Time, Continuous System
- Closing prices of the 30 DAX shares: Stochastic, Discrete Time, Discrete System

1.2.3.5 A Classic Example of Queue at Bank Counter

We see queues at everywhere. Queues are buffers to smooth out differences in arrival rates and service times. Queue Theory is well understood. Closed-form queue-theoretic models can be used to speed up simulations. Deriving results from such models requires simulation. Here are we given a example of queue formed at bank counter (Concept of queue is discussed in more detail in unit IV) .At bank counter customers arrive at random intervals and suppose there is only one cashier .Customers must wait in a queue. Service times at the cashier are also random Measured inter-arrival times (seconds):25, 111, 56, 232, 97, 452, 153, 45,...Measured service times (seconds): 45, 32, 11, 61, 93, 56, 30,...

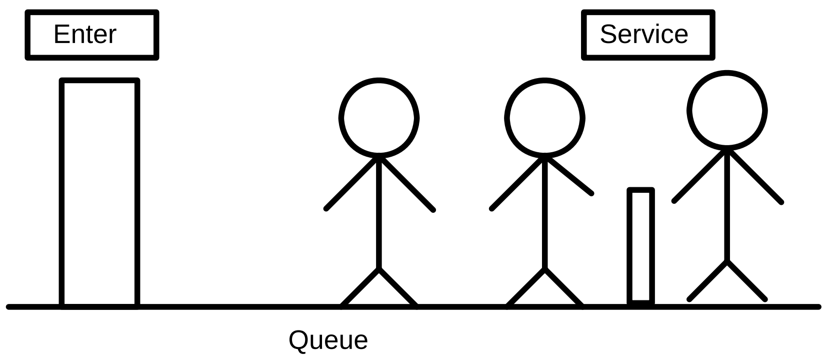

Figure 7: Classical Example of Queue

Now compute the average length of the queue and the probability that the cashier is busy.

1.3 Computer Workload and Preparation of its Models

Figure 8 outlines the phases of computer simulation development. All simulation studies begin with a specification of objectives and problem formulation. Model development including conceptualization and implementation follows the formulation. Data are collected from the real world early during model development to adequately provide the parameters of the model entities. Decisions need to be made during the implementation phase on the choice of the platform, language, analysis methods, etc. The implemented model must be verified for accuracy and validated for correspondence to the real-world system being represented. The simulation is run several times and statistical analyses of the output data are conducted before a modeler provides recommendations based on the simulation study. The processes involved in simulation modeling and analysis are not strictly linear. An analyst may iterate between different stages in computer simulation development including problem formulation, model abstraction, implementation, verification and validation, and output analysis. The different stages are detailed below

1.3.1 Steps of the Modeling Process

The modeling process is cyclic and closely parallels the scientific method and the software development life cycle (SDLC) for the development of a major software project. The process is cyclic because at any step we might return to an earlier stage to make revisions and continue the process from that point.

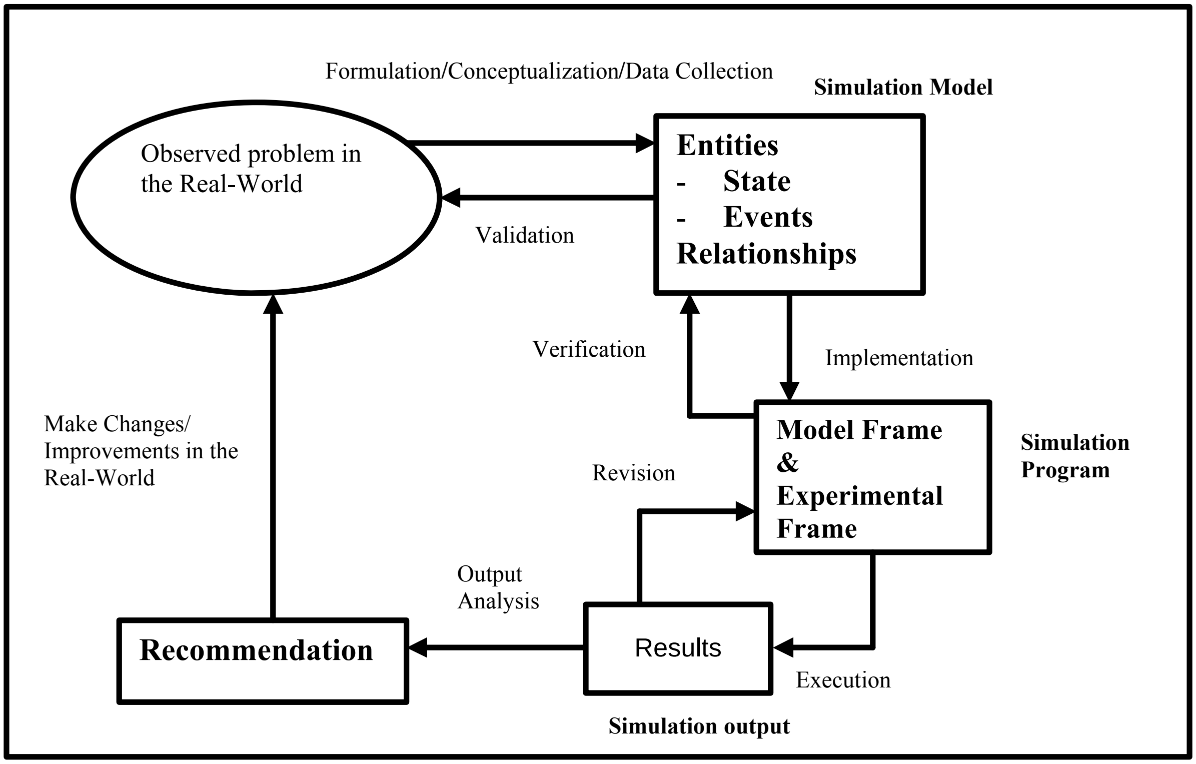

Figure 8. Phases of Computer Simulation Development and Analysis

The steps of the modeling process are as follows:

1. Analyze the Problem

We must first study the situation sufficiently to identify the problem precisely and understand its fundamental questions clearly. At this stage, we determine the problem's objective and decide on the problem's classification, such as deterministic or stochastic. Only with clear, precise problem identification can we translate the problem into mathematical symbols and develop and solve the model.

2. Formulate a Model

A specific set of goals initiate a simulation study. The goals could be for *what-if* analysis of a system being designed or for evaluation of variety of prototypical scenarios of an existing system. For example, in a service industry application, simulations of banking systems can be used to evaluate the tradeoffs associated with having an additional teller. The performance measures of the system being studied must be made explicit during the

problem formulation stage. In the banking system example, performance measures include average customer waiting time, teller utilization, etc. Decisions on whether simulation is an appropriate methodology must also be made carefully during this stage.
In this stage, we design the model, forming an abstraction of the system we are modeling. Some of the tasks of this step are as follows:

a. Gather Data

We collect relevant data to gain information about the system's behavior.

b. Make Simplifying Assumptions and Document them

In formulating a model, we should attempt to be as simple as reasonably possible. Thus, frequently we decide to simplify some of the factors and to ignore other factors that do not seem as important. Most problems are entirely too complex to consider every detail, and doing so would only make the model impossible to solve or to run in a reasonable amount of time on a computer. Moreover, factors often exist that do not appreciably affect outcomes. Besides simplifying factors, we may decide to return to Step 1 to restrict further the problem under investigation.

3. Model Abstraction

Following problem formulation, the analyst abstracts relevant features of the system to be represented in the model. Depending on the goals of the simulation study, the analyst decides on the appropriate level of detail or granularity of the model. Aspects of the system relevant to the simulation must be specified. Once the scope has been sufficiently limited, the system boundaries can be drawn. Bounding the system is a very important analytical step because it defines internal and external factors as well as inputs and outputs. The process of bounding the system can help to reduce the level of complexity by reducing the size of the system in some cases. By drawing a boundary around the system, attention is drawn to the impacts on and by other systems interacting with the system of interest. Several valuable questions can be raised in the process of delineating the problem system. The problem focus is sharpened and some attention is drawn to environmental factors outside the system. Frequently, diagramming techniques such as flow charts are helpful in pictorially representing the system in terms of entities, their behavior, and the interactions between entities. The diagrams can be very useful in communicating about the system and can sometimes reveal additional insights about the problem.
It is very helpful when conceptualizing a system to have a consistent set of questions to answer for each element of the system. These questions provide a parallel structure for the various elements of the system and ensure that all the relevant information is obtained. Forming these general questions stimulates a broader view of the elements of the system and stimulates comparisons and contrasts among them. In answering these general questions about the system, some simplifying assumptions can be made initially. There are two basic types of assumptions about systems: structural assumptions and data assumptions. Structural assumptions are made regarding the internal operation of the system. They concern differences between the way a system is designed to work and the

way it actually operates in practice under a variety of conditions. Some steps may be shortened or bypassed completely under some circumstances for the sake of expediency. Alternate routing can occur in cases of blockage or excessive queuing. More realistic factors can be added to the model gradually as the development proceeds. Data assumptions are made with respect to the entities being processed. Inputs may cycle through peak and slack periods. Individual times may vary for different entities in the same process. These can be included later when they have been more accurately determined. Assumptions made in the problem formulation should be explicitly listed so they can be addressed eventually in a meaningful manner.

4. Determine Variables and Units

We must determine and name the variables. An **independent variable** is the variable on which others depend. In many applications, time is an independent variable. The model will try to explain the **dependent variables**. For example, in simulating the trajectory of a ball, time is an independent variable; and the height and the horizontal distance from the initial position are dependent variables whose values depend on the time. To simplify the model, we may decide to neglect some variables (such as air resistance), treat certain variables as constants, or aggregate several variables into one. While deciding on the variables, we must also establish their units, such as days as the unit for time.

a. Establish Relationships Among Variables and Submodels

If possible, we should draw a diagram of the model, breaking it into submodels and indicating relationships among variables. To simplify the model, we may assume that some of the relationships are simpler than they really are. For example, we might assume that two variables are related in a linear manner instead of in a more complex way.

b. Determine Equations and Functions

While establishing relationships between variables, we determine equations and functions for these variables. For example, we might decide that two variables are proportional to each other, or we might establish that a known scientific formula or equation applies to the model. Many computational science models involve differential equations, or equations involving a derivative.

5. Solve the Model

This stage implements the model. It is important not to jump to this step before thoroughly understanding the problem and designing the model. Otherwise, we might waste much time, which can be most frustrating. Some of the techniques and tools that the solution might employ are algebra, calculus, graphs, computer programs, and computer packages. Our solution might produce an exact answer or might simulate the situation. If the model is too
complex to solve, we must return to Step 2 to make additional simplifying assumptions or to Step 1 to reformulate the problem.

6. Model Implementation

The conceptual model generated during the earlier phase must be implemented in the form of a simulation program. During implementation, an analyst can use a simulation language such as SIMAN or GPSS, or standard programming languages such as C, C++, or Java, or special simulators tailored for specific applications. Typically, most simulation languages provide software constructs to represent entities, to perform queuing utilities, and standard statistical analysis. The SIMAN simulation language also orients the analyst to make decomposition between the model frame, which is used to represent the structure and overall behavior of the system and experimental frame, which is used to store data used by the model. Some languages have a graphical front end to configure a system simulation. The graphical environment is intended for analysts without programming background to rapidly simulate and analyze a specific system. It takes longer to use standard programming languages to apply for simulation, but they provide greater levels of control in representing complex decision making behavior. The specific simulation software or language selected is dependent on the nature of the application, expertise of the analysts, and availability of appropriate hardware and software. The fundamental problem in model implementation is to translate the conceptual model to a simulation program using the constructs of a simulation or software package. Model verification and validation typically follow the model implementation phase.

7. Verify and Interpret the Model's Solution

Once we have a solution, and the model's solution is used, it may be necessary or desirable to make corrections, improvements, or enhancements. In this case, the modeler again cycles through the modeling process to develop a revised solution. We should carefully examine the results to make sure that they make sense (verification) and that the solution solves the original problem (validation) and is usable. The process of **verification** determines if the solution works correctly, while the process of **validation** establishes if the system satisfies the problem's requirements. Thus, verification concerns "solving the problem right," and validation concerns "solving the right problem." Testing the solution to see if predictions agree with real data is important for verification. We must be careful to apply our model only in the appropriate ranges for the independent data. For example, our model might be accurate for time periods of a few days but grossly inaccurate when applied to time periods of several years. We should analyze the model's solution to determine its implications. If the model solution shows weaknesses, we should return to Step 1 or 2 to determine if it is feasible to refine the model. If so, we cycle back through the process. Hence, the cyclic modeling process is a trade-off between **simplification** and **refinement**. For refinement, we may need to extend the scope of the problem in Step 1. In Step 2, while refining, we often need to reconsider our simplifying assumptions, include more variables, assume more complex relationships among the variables and submodels, and use more sophisticated techniques.
Although we described the modeling process as a sequence or series of steps, we may be developing two or more steps simultaneously. For example, it is advisable to be compiling the report from the beginning. Otherwise, we can forget to mention significant

points, such as reasons for making certain simplifying assumptions or for needing particular refinements. Moreover, within modeling teams, individuals or groups frequently work on different submodels simultaneously. Having completed a submodule, a team member might be verifying the submodule while others are still working on solving theirs.
The modeling process is a creative, scientific endeavor. As such, a problem we are modeling usually does not have one correct answer. The problems are complex, and many models provide good, although different, solutions. Thus, modeling is a challenging, open-ended, and exciting venture.
Validation is the process of assuring that the conceptual model accurately represents the behavior of the real system. Verification is the process of assuring that the implemented model accurately represents the conceptual model. These two processes are theoretically distinct but are closely related in practice. The initial conceptual model should have high *face validity*. Input should be sought from as wide a range as possible of people knowledgeable about the system. There are at least three reasons for this. Primarily, people who work with the system in different ways have different knowledge about the system. Some of this knowledge overlaps with others, but some are unique to a particular perspective. The unique perspectives complete the system concept and correct misconceptions. Secondly, the overlapping areas of knowledge provide crosschecks of the various inputs for consistency. Thirdly, participation in the modeling process reinforces confidence in the simulation. It provides the users the opportunity to question and critique the conceptual model. Involvement enhances acceptance and understanding among the users of the real system being modeled. Without end-user involvement, user skepticism and resistance can thwart improvements.

After the face validity of the conceptual model has been established the assumptions made about the model must be examined. Estimates by those experienced with the system are the best initial values for both types of assumptions. However, careful data collection and statistical analyses are required for refinements. The first step in the input data analysis is the identification of the appropriate statistical distribution for the data. Second, the parameters relevant to that distribution must be estimated from the sampled data. Finally, the fit of the selected distribution to the data can be verified by applying appropriate statistical tests such as the Kolmorgorov – Smirnov (K-S) test or chi-squared test.

Ultimately, the model must accurately predict the system performance for a range of input conditions. The simulation should be robust enough to yield accurate results when assumptions and parameters are varied. The most conclusive test for the model is the simulation of the system under conditions where the outputs of the real system are known. The focus during this process is on the overall transformation of the inputs into outputs. The outputs of the simulation can then be compared with the historical data. This testing can be only made for situations where historical data are available.

8. Execution

Most simulations represent random or stochastic characteristics where different quantitative values are obtained for the same model with varying random values. For such simulations, called non-deterministic systems, the simulation model must be executed several times and statistical analysis must be performed on the simulation output to assess variability of the simulation. From an output analysis perspective, there are two types of execution modes: terminating and steady state. The terminating mode is appropriate for systems that start and run for a time then stops for some period. An example of this kind of operation is a store that closes overnight. A steady state simulation is suitable for a system that operates continuously, such a power generating plant or a hospital emergency room. The widest possible set of initial conditions should be tried in both cases.

Initial conditions can have a profound effect on the results of simulations. Initial conditions include such things as whether or not the system resources start up in the busy or idle state. Not only should all possible sets of initial conditions be tried for a given set of conditions, but also the simulations should be run several times for each set of conditions. For terminating simulations, this means that several runs are required. In the case of a steady state simulation, data samples should be taken from several subintervals in each run. The subintervals should be of equal length in all runs for valid statistical comparisons to be made.

9. Output Analysis

A simulation is a statistical experiment that imitates the real system. Appropriate statistical methods should be applied to determine the degree of correspondence between the outputs of the model and those of the real system. These methods will vary depending upon whether a terminating or steady state simulation is being analyzed. The basic goal here is to determine the variability in the estimators chosen to evaluate the system. In general steady state simulations are more difficult to analyze due to the effects of initial conditions and choice of run time. For steady-state systems, data should not be used if it is collected before statistical equilibrium is reached. Initial condition bias can lead to erroneous conclusions, even for long run times and multiple runs. The estimators may have smaller variation, but they may converge to the wrong value. Typically, a type of hypothesis testing is conducted using confidence intervals as a function of variability of the output data from multiple simulation runs.

10. Report on the Model

Reporting on a model is important for its utility. Perhaps the scientific report will be written for colleagues at a laboratory or will be presented at a scientific conference. A report contains the following components, which parallel the steps of the modeling process:

a. Analysis of the problem

Usually, assuming that the audience is intelligent but not aware of the situation, we need to describe the circumstances in which the problem arises. Then, we must clearly explain the problem and the objectives of the study.

b. Model design

The amount of detail with which we explain the model depends on the situation. In a comprehensive technical report, we can incorporate much more detail than in a conference talk. For example, in the former case, we often include the source code for our programs. In either case, we should state the simplifying assumptions and the rationale for employing them. Usually, we will present some of the data in tables or graphs. Such Figures should contain titles, sources, and labels for columns and axes. Clearly labeled diagrams of the relationships among variables and submodels are usually very helpful in understanding the model.

c. Model solution

In this section, we describe the techniques for solving the problem and the solution. We should give as much detail as necessary for the audience to understand the material without becoming mired in technical minutia. For a written report, appendices may contain more detail, such as source code of programs and additional information about the solutions of equations.

d. Results and conclusions

Our report should include results, interpretations, implications, recommendations, and conclusions of the model's solution. We may also include suggestions for future work.

11. Recommendation

Ultimately the results are used to decide what changes to make to the real system, if any. Naturally, there is a cost associated with changing a system. Any changes will be judged in the long run by the savings or additional revenue generated. Other systems that interact with the system being simulated may be affected by changes. After the simulation has been done and the data has been thoroughly studied, the analyst must look beyond the system on which the attention has been focused. In some cases a less costly change in some other system may yield more cost-effective improvements. This possibility should be detected in the problem analysis phase preceding simulation, but sometimes elements in the systems environment are overlooked. Finally, although planning and implementing the changes may not be the responsibility of the analysts, any recommendations for change should be described in as much detail as possible. Cost and impact analysis along with a detailed description forms a solid proposal for change. Any changes made to the real world must be monitored and data must be collected from the real system to feed into the simulation model. The usefulness of the model does not end when the analysis is concluded. The model represents an investment of resources and can continue to provide useful data when changes to the system are proposed again.

1.4 Summary

Simulation is very powerful, problem solving technique. Its applicability is so general that it would be hard to point out disciplines or system to which simulation has not been applied. The basics idea behind simulation is simple, namely model the given system by means of some equations and then determine its time dependent behaviour. In Simulation we makes a model of conceptual model and then results are compared with real system. Normally simulation is used when either an exact analytic expression for the behaviour of the system under investigation is not available or the analytic solution is too time consuming. Simulation is considered as interdisciplinary subject because it uses concepts from mathematics computer science and application field. A model is a representation of an actual system. Models can be of different kinds. Discrete-Event Simulation Model is defined as one in which the state variables change only at discrete points in time at which events occur. A deterministic system is defined as in which randomness does not affect the behaviour of the system. A stochastic system is defined as in which randomness affects the behaviour of the system. In order to process a simulation of an event first we have to perform formulation of the problem, and then we have to implement the defined model in any suitable programming language. Finally we have to perform verification and validation and then analysis of output results.

1.5 Key Words

System, Model, Base Model, Simulation, Continuous, Discrete, State Variables, Dynamic, Digital Computer, Statistical Sampling, Stochastic Systems, Numerical Techniques, Monte Carlo Method, Graphical User Interfaces, Object- Oriented Programming, Optimization, Entities, Attributes, FIFO, LIFO, Statistical Analysis Software (SAS), Software Development Life Cycle (SDLC), Probability, Differential Equation, Face Validity.

1.6 Self Assessment Questions

Q1. Give an example of a nonstochastic simulation?

Q.2 Give an example of each

a. Stochastic Model
b. Continuous Model
c. Discrete Model
d. Static Model
e. Dynamic Model

Q.3 What is Model? Define some model of surrounding events.
Q.4 What is a system? Define kinds of system.
Q.5 What is the difference between static and dynamic model?
Q.6. What is the difference between continuous and discrete model?
Q.7 What is simulation? Give some advantage and disadvantage of simulation.
Q.8 Write some application of simulation.
Q.9 How to Build and Apply Computer Simulations? Explain.
Q.10 What is meant by the "System State" in a simulation.
Q.11 Compare and contrast the modeling process with the scientific method: Make observations; formulate a hypothesis; develop a testing method for the hypothesis; collect data for the test; using the data, test the hypothesis; accept or reject the hypothesis.
Q.12 Compare and contrast the modeling process with the software life cycle: Analysis, design, implementation, testing, documentation, and maintenance.
Q.13 What do you understand by the term face Validity of a conceptual model?

1.7 Reference /Suggested Reading

1. *Proceedings of the 1999 Winter Simulation Conference,* Jerry Banks, **Introduction to Simulation**
2. Bernard P. Zeigler, Herbert Praehofer, and Tag Gon Kim. *Theory of Modelling and Simulation: Integrating Discrete Event and Continuous Complex Dynamic Systems*. Academic Press, second edition.
3. Banks, Carson, Nelson & Nichol, *Discrete Event System Simulation*, Prentice Hall.
4. G.Gorden, "System Simulation",PHI.
5. N. Deo , " System Simulation", PHI.
6. Giordano, Frank R., Maurice D. Weir, and William P. Fox. 2003. *A First Course in Mathematical Modeling*. 3rd ed. Pacific Grove, Calif.: Brooks/Cole-Thompson Learning.

Subject : System Simulation and Modeling
Paper Code: MCA 504
Lesson : Verification and Validation of Models
Lesson No. : 02

Author : Jagat Kumar
Vetter : Dr. Pradeep Bhatia

Structure

2.0 Objective

The area of experimentation and results analysis for simulation models is briefly introduced here. By the end of this module you will learn the verification and validation techniques to compare the defined model with real system's data.

2.1 Introduction

How do we know that the model we have used, is an accurate representation of the system being simulated? This is an important question and must be answered satisfactorily before a simulation study can be made use of. The area of experimentation and results analysis for simulation models is well developed and a range of rules and guiding methods can be found in the literature, e.g. are available. Many of the techniques developed are here to ensure that dangerous mistakes are not made when analyzing and interpreting the results. In fact , without establishing the validity of the model , if we accept the(erroneous) simulation results the consequences may be disastrous. Put simply the power of modern simulation software to generate large quantities of data can leave the user with the false sense of security that the results generated are credible and truly representative of the system under study. Like all modelling techniques care needs to be exercised.
What is a valid model? Since no simulation model will duplicate the given system in every detail it is not an appropriate question to ask if a simulator is a 'true' model of a real system. We should only ask if the model is a 'reasonable' approximation of the real

system. The acceptable levels of reasonableness and approximation will vary from system to system and simulation to simulation. There is no universally acceptable criteria for accepting a simulation model as a valid representation. There are only guidelines that aid in establishing confidence in the model.

There are a number of phases as shown in Figure 1, for checking a simulation model prior to experimental analysis:

- **Verification** - the accuracy of transforming a problem formulation into a model (specification) deals with building the **model right**
- **Validation** - model behaves with satisfactory accuracy consistent with the study objectives deals with building the **right model**

The above are essential checks performed prior to analysis of the model and are used to establish what is known as model credibility. Verification and validation is shown in Figure 2, as cause - effect and input – output terms.

A simulation run typically starts in the empty and idle state. The run is therefore characterised by a "run-in" phase followed by a "steady state" phase, see Figure 1. The run-in phase is generally ignored and is only used for investigating the effects of transient conditions such as starting up a new factory or performing radical changes within an existing facility.

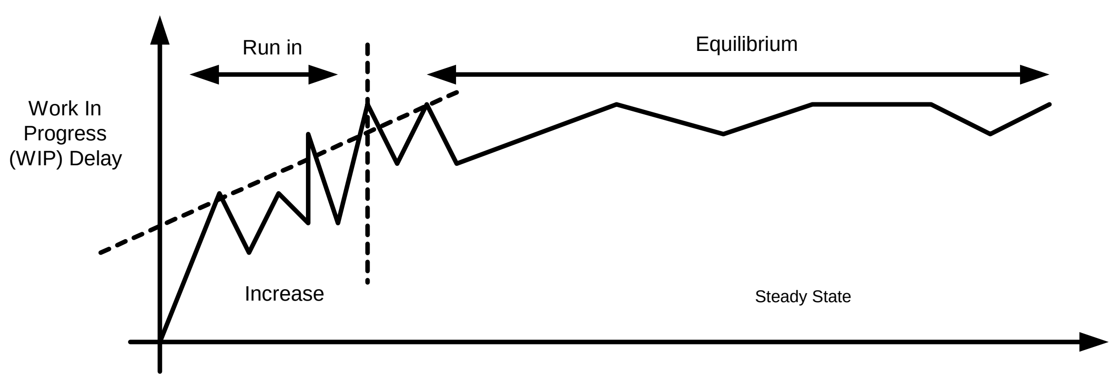

Figure 1. The two key phases of a simulation run

Typically the steady state phase is of greater interest. At this stage checks must be made to ensure no long term trends exist, such as continual build up of stock in the factory, that suggest the model (hence the real system) will be unstable and unworkable.
Generally what is known as multiple replications is performed. This is where the model is run several times. Each time the random number generators are set to provide different sequences of random numbers, e.g. in case of manufacturing process the breakdown patterns of machines are different and the points at which material is scrapped is different. This allows confidence that the results being compiled represent the average and the range of conditions that are likely and therefore play down ‘freak’ or ‘unusual’ behaviour.

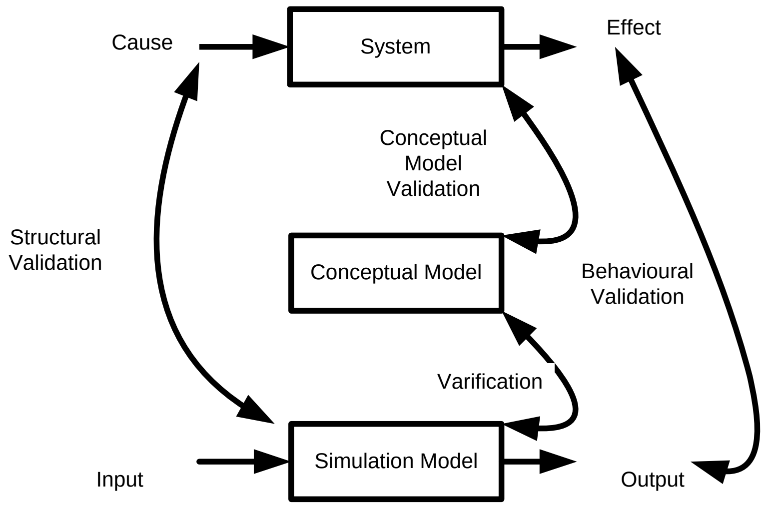

Figure 2 Verification and Validation Activities

2.2 Verification and Validation

The validation efforts can be grouped into two parts

1. validation of the abstract model itself
2. validation of its implementation

The first part consists of examining all assumptions, which transform the real world system into the conceptual model. A great deal of judgment and an intimate knowledge of the real system are involved in this step. The validation of the abstract model is often highly subjective. Testing the validity of an implementation is a more objective and easier task. It consists of checking the logic, the flowchart, and the computer program to ensure that the model has been correctly implemented.

The presentation of an experimental frame, which is shown in Figure 2, enables a rigorous definition of model *validity*. Let us first postulate the existence of a unique *Base Model*. This model is assumed to accurately represent the behavior of the Real System under *all* possible experimental conditions. This model is *universally valid* as the data $\mathbf{D}_{\mathbf{RealSystem}}$ obtainable from the Real System is always equal **(symbol ≡ is used for equality)** to the data $\mathbf{D}_{\mathbf{BaseModel}}$ obtainable from the model.

$$\mathbf{D}_{\mathbf{BaseModel}} \equiv \mathbf{D}_{\mathbf{RealSystem}} \qquad \text{--------------------(i)}$$

A Base Model is distinguished from a **Lumped Model** by the limited experimental context within which the latter accurately represents Real System behavior.
A particular **experimental frame** ***E*** may be applicable to a real system or to a model. In the first case, the data potentially obtainable within the context of *E* are denoted by

$\boldsymbol{D_{RealSystem}} \| \boldsymbol{E}$. In the second case, obtainable data are denote by **Dmodel** $\|$ **E**. With this notation, a model is valid for a real system within Experimental Frame *E* if

$$D_{LumpedModel} \| \; E \equiv D_{RealSystem} \| \; E \qquad \text{--------------------(ii)}$$

The data equality $\equiv$ *m*ust be interpreted as equal to a certain degree of accuracy. It shows how the concept of validity is not absolute, but is related to the experimental *context* within which Model and Real System *behavior* are compared and to the *accuracy metric* used.
One typically distinguishes between the following types of model validity.

Replicative Validity concerns the ability of the Lumped Model to *replicate* the input/output data of the Real System. With the definition of a Base Model, a **Lumped Model** is replicatively valid in Experimental Frame *E* for a Real System if

$$\mathbf{D_{LumpedModel}} \| \; \mathbf{E} \equiv \mathbf{D_{BaseModel}} \| \mathbf{E} \qquad \text{--------------------(iii)}$$

Predictive Validity concerns the ability to identify the *state* a model should be set into to allow *prediction* of the response of the Real System to *any* (not only the ones used to identify the model) input segment. A Lumped Model is predicatively valid in Experimental Frame *E* for a Real System if it is replicatively valid and

$$F_{LumpedModel} \| \; E \subseteq \mathbf{F}_{BaseModel} \; \| E \qquad \text{--------------------(iv)}$$

where F_S is the set of I/O functions of system *S* within Experimental Frame *E*. An I/O function identifies a *functional relationship* between Input and Output, as opposed to a general non-functional *relation* in the case of replicative validity.

Structural Validity concerns the *structural relationship* between the Real System and the Lumped Model. A Lumped Model is structurally valid in Experimental Frame *E* for a Real System if it is predictively valid and there exists a **morphism** Δ from Base Model to Lumped Model within frame *E*.

$$\mathbf{LumpedModel} \| \; \mathbf{E} \; \Delta \; \mathbf{BaseModel} \| \mathbf{E} \qquad \text{--------------------(v)}$$

When trying to assess model validity, one must bear in mind that one only observes, at any time *t*, $\mathbf{D_{RealSystem}}$ **(t)** , a subset of the potentially observable data $D_{RealSystem}$. This obviously does not simplify the model validation enterprise.
Whereas assessing model validity is intrinsically impossible, the *verification* of a *model implementation* can be done rigorously. A *simulator* implements a lumped model and is thus a source of obtainable data $\boldsymbol{D}_{\mathbf{Simulator}}$. If it is possible to prove (often by design) a

structural realtionship (morphism) between Lumped model and Simulator, the following will hold unconditionally

$$D_{\mathbf{Simulator}} \equiv \mathbf{D}_{\mathbf{LumpedModel}} \qquad \text{-------------------(vi)}$$

Before we go deeper into predictive validity, the relationship between different *refinements* of both Experimental Frames and models is elaborated. In Figure 3, the *derived from* relationship for Experimental Frames and the *homomorphism.*

Relationship for Models is depicted. If we think of an Experimental Frame as a formal representation of the context within which the model is a valid representation of the dynamics of the system, a more restricted Experimental Frame means a more specific behaviour. It is obvious that such a restricted Experimental Frame will match far more models than a more general Experimental Frame. Few models are elaborate enough to be valid in a very general input/parameter/performance range. Hence, the large number of applies to (*i.e.,* match) lines emanating from a restricted Experimental Frame. The homomorphism between models means that, when modifying/transforming a model (*e.g.,* adding some non-linear term to a model), the simulation results (*i.e.,* the behaviour) within the same experimental frame must remain the same.

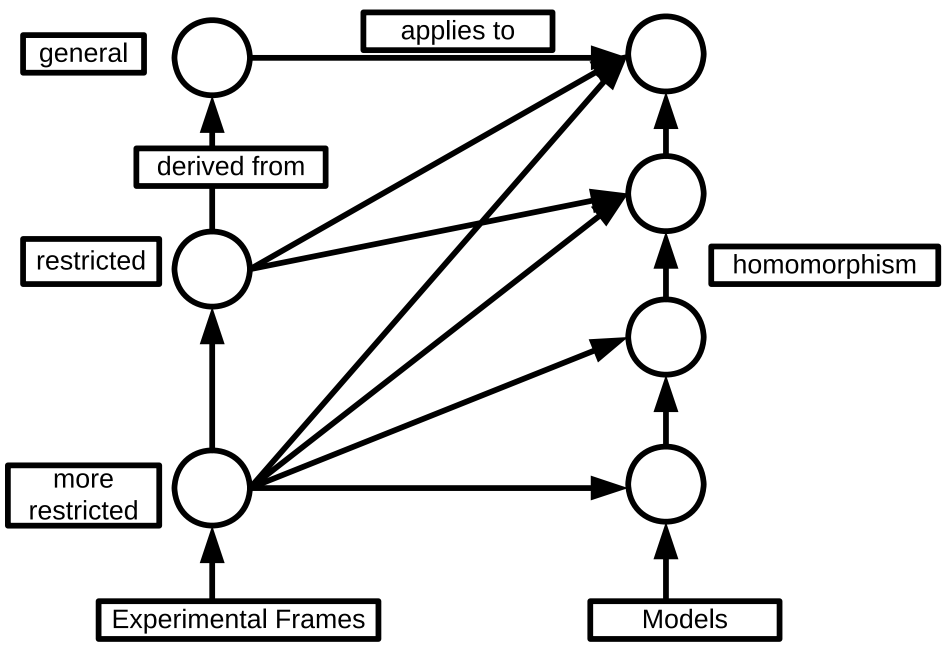

Figure 3 : Experimental Frame – Model Relationship

Though it is meaningful to keep the above in mind during model development and use, the highly non-linear nature of many continuous models makes it very difficult to automate the management of information depicted in Figure 3. Non-linear behaviour makes it almost impossible, based on a model or experimental frame symbolic representation, to make a statement about the area in state-space, which will be covered (*i.e.,* behaviour). A pragmatic approach is to

1. Let an expert indicate what the different relations are. This is based on some insight into the nonlinear dynamics. Such expert knowledge can be built from a large number of conducted experiments.
2. Constantly with each experiment validate the expert information.

A crucial question is whether a model has predictive validity, is it capable not only of reproducing data which was used to choose the model and parameters but also of predicting new behavior? The predictive validity of a model is usually substantiated by comparing new experimental data sets to those produced by simulation, an activity known as model validation. Due to its special importance in the communication between model builders and users, model validation has received considerable attention in the past few decades. The comparison of the experimental and simulation data are accomplished either subjectively, such as through graphical comparison, Turing test, or statistically, such as through analysis of the mean and variance of the residual signal employing the standard F statistics, multivariate analysis of variance regression analysis, spectral analysis, autoregressive analysis, autocorrelation function testing, error analysis, and some non-parametric methods.

The above-mentioned methods are designed to determine, through comparison of measured and simulated data, the validity of a model. As one might intuitively expect, different modelling errors usually cause the behavior of the model to deviate in different ways from that of the real system or, in other words, different modelling errors correspond to different pattern in the error signal, the difference between experimental data and simulated data. These patterns if extractable, can obviously be used to identify the modelling errors.

2.3 Comparing Model Data with Real System Data

After development of suitable model of defined problem and simulation of defined model, we have to perform the comparison of output data with real system data.

2.3.1 Validating Existing Systems

When the simulated system exists in real life, then the most obvious and the best approach is to use the real world inputs to the model and compare its outputs with that of the real world inputs to the model and compare its output with that of the real system. This process of validation is straightforward enough in principal but may present some difficulties when carried out. Firstly, it may not always be easy to obtain input and output data from a real life system without distributing it. Secondly, even if we could get actual input output of an existing model it would not generally be for very long periods. Since the data are usually probabilistic. For small lengths of simulation runs the variability of the model output would be large. Therefore designing test that work with small samples is difficult. What usually is done is to simulate the model several times (replicate) with different sequences of random numbers and obtain the range of variation amongst these. Then, if the model is valid, the real output should lie somewhere in the middle of the range of model output. The third problem is to establish that the model output and the real system outputs are ‘practically’ from the same population.

If the outputs to be compared are sample means (e.g., average queue length, waiting time, idle time), one could use any of a number of statistical tests **(called 'goodness' of fit tests)** available to measure the discrepancy between the two outputs (i.e., model output and the real system output). One such test is **chi-square test**. Others are **Kolmogorov-Smirnov test, Cramer - von Mises test** and the **Moments test**. One could also use hypothesis testing to determine if there is any significant difference between, say the average of the independent set of observations.

2.3.2 Validating Fist Time Model

If a model is intended to describe a proposed or hypothethetical system (which does not exist at present or did not exist in the past) then the task of validation is even more difficult. There are no historical data available to compare its performance with. Since hypothetical system are , by their very nature , based upon assumptions it is the validity of these assumptions the simulation model is dependent on. A number of guidelines for testing validity of such system have been found useful. These are as

1. **Subsystem Validity:** A model itself may not have any existing system to compare it with, but it may consist of known subsystem each of that validity can be tested separately.
2. **Internal validity:** One tends to rejects a model if it has a high degree of internal variability. A stochastic system with high variance due to its internal processes will obscure changes in the output due to input changes. The test can be performed by replicating a simulation run with several different random number sequences and then computing the variance of the outputs. If the variance is too high we reject the model.
3. **Sensitivity Analysis:** Sensitivity analysis consists of systematically varying the values of parameters or the input variables one at a time (while keeping all others constant) over some range of interest and observing the effect upon the model's response. Sensitivity analysis will tell to which parameters the system is more sensitive. The parameters to which the system response is relatively insensitive, we need not pay very close attention to. The knowledge how far the assumed parameters values could be from the true one without significantly affecting the response helps building our confidence in the model.
4. **Face Validity:** If the model goes against the common sense and logic, it should be rejected (even if it behaves like the real system). If those with experience and insight into similar system do not judge the model as reasonable, it has to be rejected.

   These and other validation tests do not completely validate a model. While failure to pass a validation test would results in rejection of a model, passing these tests does not guarantee that the model is valid. It only builds up our confidence in the model.

   Ideally errors of a modeling should be separated from the errors in its implementation (programming errors, etc.) by first validating the abstract mathematical model before writing a simulator for it. In practice, however, it is really possible to check the validity of the mathematical model without examining its computer version. This is because of the mathematical intractability of a model, which was the reason for simulating it in the first place.

Although validation is often messy, expensive and time consuming, involves subjective ness and judgments and is rarely conclusive, it must always be attempted. However inconclusive, it does provide a check against grosser errors and gives us confidence to use the simulation results for decision making.

2.4 Summary

Verification and validation of defined system is very critical and it provides the testability to defined system against the real system. Verification is defined as the accuracy of transforming a problem formulation into a model (specification) deals with building the model right while Validation is defined as the defined model behaves with satisfactory accuracy consistent with the study objectives deals with building the right model. Without establishing the validity of the model, simulation results can be erroneous and their consequences may be disastrous.

2.5 Keywords

Verification, Validation, System, Model, Conceptual Model, Base Model, Lumped Model, Real System , Kolmogorov- Smirnov test (KS Test), Cramer -von Mises test, Moments test , Goodness of Fit, Face Validity, Replicate Validity, Predictive Validity, Structural Validity, Work-in-Progress (WIP).

2.6 Self Assessment Questions

Q.1 Two similar terms used in the steps of a simulation study are “verification” and “validation.” One of them refers to the debugging of the simulation code itself. Which one refers to the process of insuring that the model is a correct representation of the system?
Q.2 What do you understand the term “Model Validation and Verification”? Explain.
Q.3 What do you understand by the term Face Validity of a Conceptual Model?
Q.4 What is the difference between Validation and Verification?
Q.5 Why Validation is so important in Modelling and Simulation?
Q.6 Give some advantages and disadvantage of Validation in Simulation.

2.7 Reference /Suggested Reading

7. *Proceedings of the 1999 Winter Simulation Conference,* Jerry Banks, Introduction to Simulation.
8. Bernard P. Zeigler, Herbert Praehofer, and Tag Gon Kim. *Theory of Modelling and Simulation: Integrating Discrete Event and Continuous Complex Dynamic Systems*. Academic Press, second edition, 2000.

9. Banks, Carson, Nelson & Nichol, *Discrete Event System Simulation*, Prentice Hall.
10. N. Deo , " System Simulation", Prentice Hall of India.

Subject : System Simulation and Modeling
Paper Code: MCA 504
Author : Jagat Kumar
Vetter : Dr. Pradeep Bhatia
Lesson : Differential Equations In Simulation
Lesson No. : 03

Structure

3.0 Objective

The main objective of this unit to introduce the concepts of ordinary and partial differential equations in continuous system modeling. By the end of this module we will learn about different kinds of differential equations and their application in modeling. We will also learn the uses of simulation in education and training.

3.1 Introduction

Modeling of a system whether system is continuous or discrete heavily used the concepts of Ordinary Differential Equation (ODE), Partial Differential Equation (PDE) and probability and statistics. Even we can say that without these two branch of mathematics modeling is just impossible.
Continuous processes occur everywhere as we will learn in unit V. Here, we are interested in cases with discrete variables, some examples of continuous process are

- An object falling to the ground
- The motion of the planets orbiting the sun
- The current and voltage in an electrical circuit
- The level of alcohol in my blood on January 1st, 2005
- The populations of a predator and its prey

In almost all above cases, the relationships between the variables and its rate of change i.e. its derivative are defined by an Ordinary Differential Equations (ODEs) are very important in all branches of Science and Engineering. ODEs form the basis for the simulation of almost all continuous phenomena. Understanding ODEs is essential for understanding natural and technical processes.

3.2 Ordinary Differential Equations

A **differential equation** is an equation involving an unknown function and its derivatives. The **order** of the differential equation is the order of the highest derivative of the unknown function involved in the equation.

$$\frac{dy}{dx} = f(y,t) \qquad \text{----------------------------(i)}$$

In addition, we usually know the value of y (0) is $y(0) = y_0$.

A **linear differential equation** of order *n* is a differential equation written in the following form

$$a_n(x)\frac{d^n y}{dx^n} + a_{n-1}(x)\frac{d^{n-1}y}{dx^{n-1}} + \dots\dots\dots + a_1(x)\frac{d\ y}{dx} + a_0(x)y = f(x) \qquad \text{------------------(ii)}$$

where $a_n(x)$ is not the zero function. Note that some may use the notation $y', y'', y''', y^{(4)}, \ldots$ for the derivatives. A linear equation obliges the unknown function y to have some restrictions. Indeed, the only operations which are accepted for the variable y are:

(i) Differentiating y
(ii) Multiplying y and its derivatives by a function of the variable x
(iii) Adding what you obtained in (ii) and let it is equal to a function of x.

There are several issues related to differential equation solution whether a differential equation have a solution? Does a differential equation have more than one solution? If yes, how can we find a solution, which satisfies particular conditions? A problem in which we are looking for the unknown function of a differential equation where the values of the unknown function and its derivatives at some point are known is called an **initial value problem** (in short IVP). If no initial conditions are given, we call the description of all solutions to the differential equation the **general solution**.

3.2.1 Modeling via Differential Equations

One of the most difficult problems that a scientist deals with in his everyday research is "How do I translate a physical phenomenon into a set of equations which describes it?"

It is usually impossible to describe a phenomenon **totally**, so one usually strives for a set of equations which describes the physical system **approximately** and **adequately**.

In general, once we have built a set of equations, we compare the data generated by the equations with real data collected from the system (by measurement). If the two sets of data "agree" (or are close), then we gain confidence that the set of equations will lead to a good description of the real-world system. For example, we may use the equations to make predictions about the long-term behavior of the system. It is also important to keep in mind that the set of equations stays only "valid" as long as the two sets of data are close. If a prediction from the equations leads to some conclusions which are by no means close to the real-world future behavior, then we should modify and "correct" the underlying equations. As you can see, the problem of generating "good" equations is not an easy exercise.

Note that the set of equations is called a **Model** for the system.

How do we build a Model?

The basic steps in building a model are as

Step 1: Clearly state the assumptions on which the model will be based. These assumptions should describe the relationships among the quantities to be studied.
Step 2: Completely describe the parameters and variables to be used in the model.

Step 3: Use the assumptions (from Step 1) to derive mathematical equations relating the parameters and variables (from Step 2).

The best example of mathematical modeling is the one related to population growth problems given below (Example 2). Keep in mind that this problem has many ramifications ranging from population explosion to extinction phenomena.

If there is no analytic solution for systems of ODEs, we are forced to integrate using numerical methods. The numerical integration of ODEs forms the most important part of continuous simulation.

Example 1: Balance Equations

Most ODEs are balance equations. A balance equation basically says

Change = Increase – Decrease

For example an ODE for the amount of water x in a tank

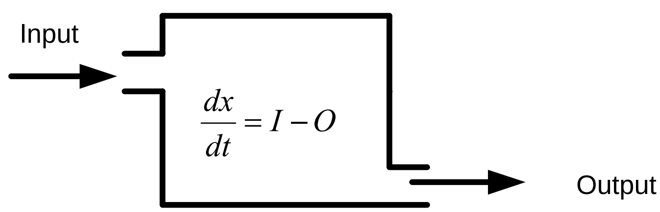

Figure 1: Water tank Problem Model by Balance Differential Equation

Example 2: Population Dynamics

Here are some natural questions related to population problems:

- What will the population of a certain country be in ten years?
- How are we protecting the resources from extinction?

More can be said about the problem but, in this brief discussion we will not discuss them in detail. In order to illustrate the use of differential equations with regard to this problem we consider the easiest mathematical model offered to govern the population dynamics of a certain species. It is commonly called **the exponential model**, that is, the rate of change of the population is proportional to the existing population. In other words, if $P(t)$ measures the population, we have

$$\frac{dP}{dt} = kP$$

where the rate k is constant. It is fairly easy to see that if $k > 0$, we have growth, and if $k < 0$, we have decay. This is a linear equation which solves into

$$P(t) = P_0 e^{kt}$$

where P_0 is the initial population, i.e. $P(0) = P_0$. Therefore, we conclude the following:

- if $k>0$, then the population grows and continues to expand to infinity, that is,

$$\lim_{t \to +\infty} P(t) = +\infty$$

- if $k<0$, then the population will shrink and tend to 0. In other words we are facing extinction.

Clearly, the first case, $k>0$, is not adequate and the model can be dropped. The main argument for this has to do with environmental limitations. The complication is that population growth is eventually limited by some factor, usually one from among many essential resources. When a population is far from its limits of growth it can grow exponentially. However, when nearing its limits the population size can fluctuate, even chaotically. Another model was proposed to remedy this flaw in the exponential model. It is called the logistic model (also called Verhulst-Pearl model). The differential equation for this model is

$$\frac{dP}{dt} = kP\left(1 - \frac{P}{M}\right)$$

where M is a limiting size for the population (also called the **carrying capacity**). Clearly, when P is small compared to M, the equation reduces to the exponential one. In order to solve this equation we recognize a nonlinear equation, which is separable. The constant solutions are $P = 0$ and $P = M$. The non-constant solutions may obtained by separating the variables.

$$\frac{dP}{P\left(1 - \frac{P}{M}\right)} = kdt$$

and integration

$$\int \frac{dP}{P\left(1 - \frac{P}{M}\right)} = \int kdt$$