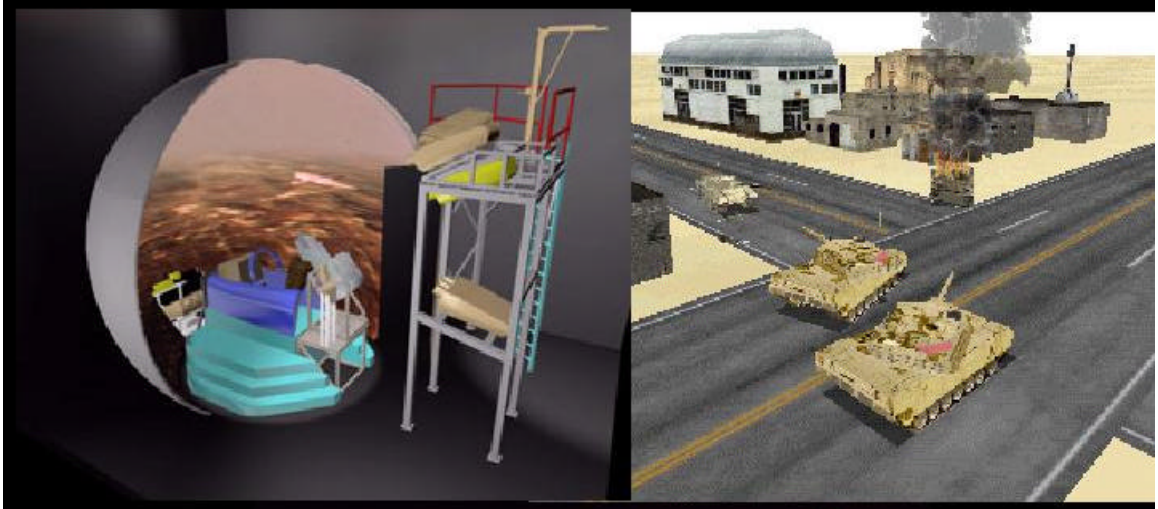
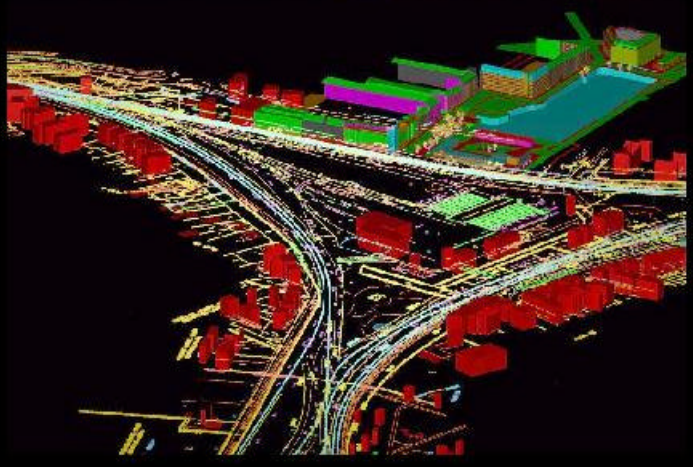
SIMULATION
The Engine Behind The Virtual World

Volume 1 in the *Simulation 2000* Series





Roger Smith

Title:	**Simulation: The Engine Behind The Virtual World**
Author:	**Roger D. Smith** Chief Scientist, ModelBenders LLC http://www.modelbenders.com/
Summary:	This article describes the science, technology, and tools of computer simulation in a language and style that average people can grasp very quickly. Simulation is the engine or brains behind all forms of believable computer generated worlds or business analysis tools. It allows virtual worlds to be more interactive, computer games to represent smarter opponents, business analysis tools to extrapolate information more accurately, and elaborate special effects to be driven by software. The article is an excellent reference for news stories, classroom lectures, term papers, research projects, and career guidance. Volume 1 in the ***Simulation 2000*** series. **Intended Audience:** • News Reporters and Story Researchers • High School and College Students • Classroom Teachers • Managers, Administrators, and Scientists Entering the Field **Outline of Material:** • Background • Common Applications of Simulation • The Simulation Development Process • Fundamental Techniques for Model Building • Essential Computer Technologies • Simulation Languages and Tools • Conclusion • Excellent References • Appendix A: 35 Simulation Web Sites • Appendix B: 23 Universities with Simulation Curricula

BACKGROUND

Definition. *Simulation* is the process of designing a model of a real or imagined system and conducting experiments with that model. The purpose of simulation experiments is to understand the behavior of the system or evaluate strategies for the operation of the system. Assumptions are made about this system and mathematical algorithms and relationships are

derived to describe these assumptions - this constitutes a "model" that can reveal how the system works. If the system is simple, the model may be represented and solved analytically. A single equation such as DISTANCE = (RATE * TIME) is an analytical solution representing the distance traveled by an object at constant rate for a given period of time.

However, problems of interest in the real world are usually much more complex than this. In fact, they may be so complex that a simple mathematical model can not be constructed to represent them. In this case, the behavior of the system must be estimated with a simulation. Exact representation is seldom possible in a model, constraining us to approximations to a degree of fidelity that is acceptable for the purposes of the study. Models have been constructed for almost every system imaginable, to include factories, communications and computer networks, integrated circuits, highway systems, flight dynamics, national economies, social interactions, and imaginary worlds. In each of these environments, a model of the system has proved to be more cost effective, less dangerous, faster, or otherwise more practical than experimenting with real system.

For example, a business may be interested in building a new factory to replace an old one, but is unsure whether the increased productivity will justify the investment. In this case, simulation would be used to evaluate a model of the new factory. The model would describe the floor space required, number of machines, number of employees, placement of equipment, production capacity of each machine, and the waiting time between machines. Simulation runs would then evaluate the system and provide an estimate of the production capacity and the costs of a new factory. This type of information is invaluable in making decisions without having to build an actual factory to arrive at an answer.

Simulations are usually referred to as either discrete event or continuous, based on the manner in which the state variables change. Discrete event refers to the fact that state variables change instantaneously at distinct points in time. In a continuous simulation, variables change continuously, usually through a function in which time is a variable. In practice, most simulations use both discrete and continuous state variables, but one of these is predominant and drives the classification of the entire simulation.

History. One of the pioneers of simulation concepts was John von Neumann. In the late 1940's he conceived of the idea of running multiple repetitions of a model, gathering statistical data, and deriving behaviors of the real system based on these models. This came to be known as the Monte Carlo method because of the use of randomly generated variates to represent behaviors that could not be modeled exactly, but could be characterized statistically. von Neumann used this method to study the random actions of neutrons and the effectiveness of aircraft bombing missions. These methods wee first used in industry to determine the maximum potential productivity of factories.

Concepts for Discrete Event Simulations (DES) were developed in the late 1950's. The first DES-specific language was developed at General Electric by K.D. Tocher and D.G. Owen.

The General Simulation Program (GSP) was created to study manufacturing problems at General Electric and was shared with the rest of the world at the Second International Conference on Operations Research.

Purpose. Simulation allows the analysis of a system's capabilities, capacities, and behaviors without requiring the construction of or experimentation with the real system. Since it is extremely expensive to experiment with an entire factory to determine its best configuration, a simulation of the factory can be extremely valuable. There are also systems, like nuclear reactions and warfare, which are too dangerous to carry out for the sake of analysis, but which can be usefully analyzed through simulation.

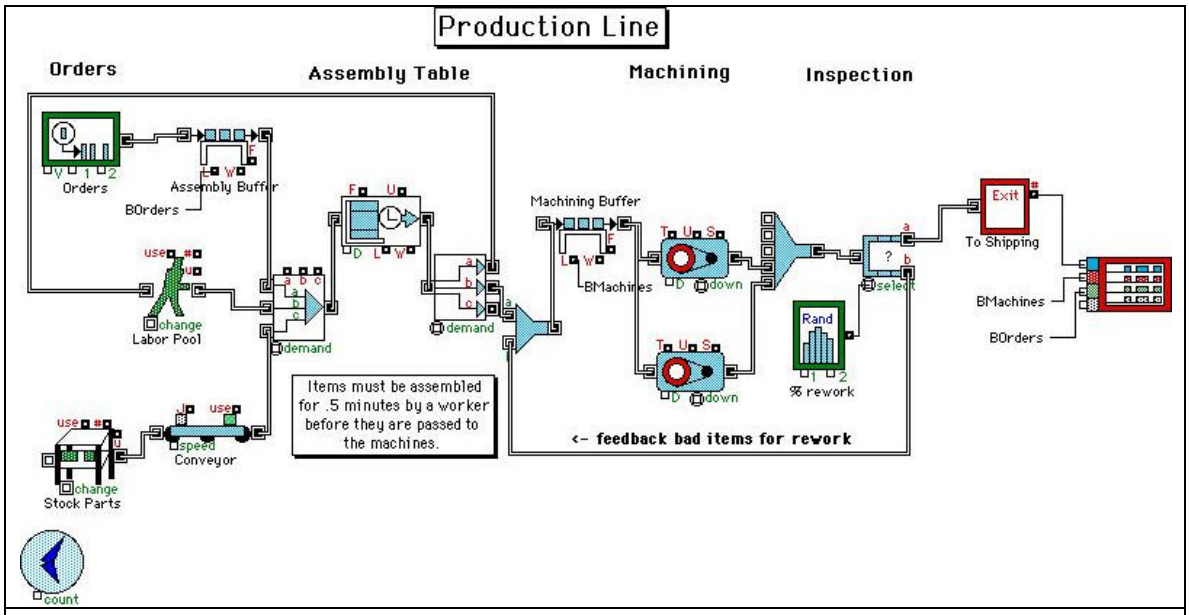
Advantages and Disadvantages. When conducting a simulation or contriving a model, certain limitations must be acknowledged. Primary among these is the ability to create a model that accurately represents the system to be simulated. Real systems are extremely complex and a determination must be made about the details that will be captured in the model. Some details must be omitted and their effects lost or aggregated into other variables that are included in the model. In both cases, an inaccuracy has been introduced and the ramifications of this must be evaluated and accepted by the model developers. Another limitation is the availability of data for describing the behavior of the system. It is common for a model to require input data that is scarce or unavailable. This issue must be addressed prior to the design of the model to minimize its impact once the model is completed.

Both of the limitations listed above lead to a simulation that provides approximate results or that describes system behavior statistically. For this reason, simulation usually provides measurements of general trends, rather than exact data for specific situations or individuals. A simulation would be hard pressed to determine which piece of material will actually be ruined by a milling machine. But, it would be an excellent tool for determining the impacts of machine failure on factory productivity, using known statistical distributions for the failures of many machines.

COMMON APPLICATIONS OF SIMULATION

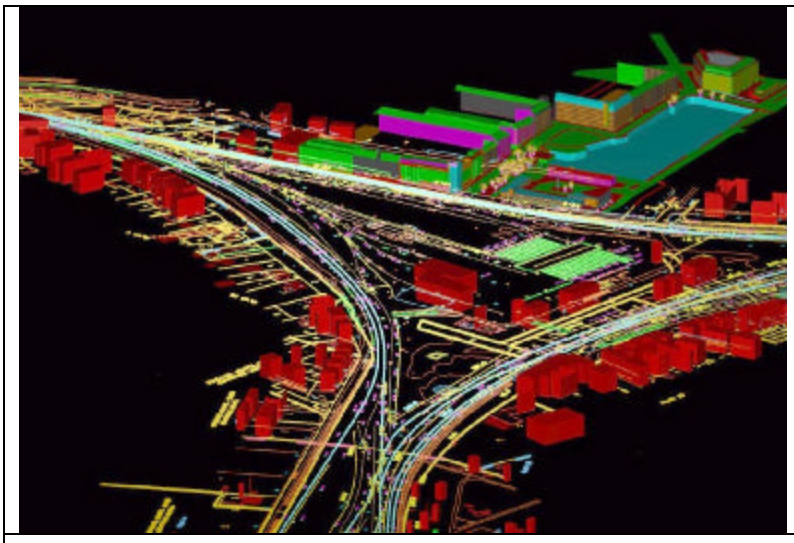
Simulation is used in nearly every engineering, scientific, and technological discipline. In the fifty years since its formal definition it has been adapted for a wide variety of applications. Today, the techniques are employed in the design of new systems, the analysis of existing systems, training for all types of activities, and as a form of interactive entertainment.

Design. Designers turn to simulation to allow them to characterize or visualize a system that does not yet exist and for which they wish to achieve the optimum solution. Manufacturing models may describe the capacities of individual machines, the time to prepare material for operation, time to transfer materials from one machine to another, the effects of human operators, and the capacities of waiting queues and storage bins. Simulations of new pieces of equipment may evaluate their performance, stress points, transportability, human interfaces, and potential hazards to the environment. Business process models may evaluate the flow of paperwork through a company to determine where redundancies or unnecessary operations are located, allowing them to redesign operations such that the same work can be performed with a fraction of the labor and time that has evolved into the process. Major airlines use simulations to study complex routing patterns for large numbers of aircraft traveling around the world. The intention is to identify routes that serve the most passengers and use the fewest assets most efficiently. Factors such as aircraft capacity, ground time, flight time, scheduled maintenance, crew availability, weather effects, and unscheduled downtimes are all considered in such models.



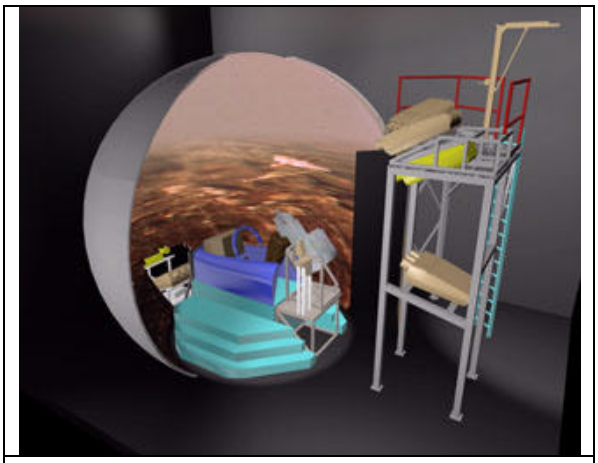
Simulation of a Manufacturing Production Line
Courtesy of Imagine That Inc.

Analysis. Analysis refers to the process of determining the behavior or capability of a system that is currently in operation. Unlike design, analysis may be supported by the collection of data from the actual system to establish model behaviors. The model can then be modified to determine the optimum configuration or implementation of the real system. A computer network can be described by the volume of traffic carried, the capacity of the lines and switches, performance of a router, and the path taken from sender to receiver. Based on measured message patterns, the network can be configured to deliver the most information using the shortest or most reliable paths available. In the health care industry, the models are used to schedule doctors, staff, equipment, and patients in an effort to improve service times and reduce costs. Social trends can be simulated to determine what services or goods will be needed at a given time by a specific sector of society. The impacts of aging, health, family composition, and a host of other factors can be predicted from an appropriate social model.



Traffic Flow Simulation
Courtesy of Intergraph Computers Inc.

Training. Training simulations recreate situations that people face on the job and stimulate the trainee to react to the situation until the correct responses are learned. These devices produce experienced personnel without the expense of making mistakes on the job. Perhaps the best known of these are flight simulators, which model dangerous environments where life threatening situations can be mitigated through learning in a non-lethal environment. Military simulators replicate the performance characteristics of the aircraft, instruments in the cockpit, effects of weapons, support from other combat systems, communications with other pilots, and terrain over which the events occur. Similar systems are used to train the captains of large ocean-going ships to dock without destroying both a real ship and a real dock. Entire mock-ups are made of nuclear power control centers to teach operators how to respond to emergency situations and to identify potential hazards before a crisis occurs. Modern medical equipment is so expensive and scarce that simulations have been constructed to allow interns and nurses to practice, develop, and certify their skills without having to schedule training time on the real equipment, competing for its use by real patients.



Flight Simulator
Courtesy of SEOS International Ltd.

Entertainment. The entertainment industry makes wide use of simulation to create games that are enjoyable and exciting to play. These contain many, but usually not all of the components of simulation described in this article. Arcade games, computer games, board wargames, and role playing games all require the creation of a consistent model of an imaginary world and devices for interacting with that world. These simulations often appear very similar to training simulations, but differ in that their purpose is entertainment rather than practice for real-world events. This fact allows game developers the freedom to modify the laws of physics and other behaviors, rather than accurately capturing their real world equivalents. Advances in these simulations, together with the prevalence of the Internet, are allowing the creation of multi-player on-line games that pit players against multiple opponents around the world. Though the purpose of these simulations is entertainment, the technical challenges faced by their developers are just as daunting as those in the other categories.

Tank Simulator Computer Game
Courtesy of MaK Technologies, Interactive Magic, and Zombie Studios

All of the areas listed here allow systems to be understood without incurring the expenses or dangers of working with the actual system. As the benefits of simulation become more widely understood, and the complexity of modern problems increases, the user base for simulation will grow rapidly.

THE SIMULATION DEVELOPMENT PROCESS

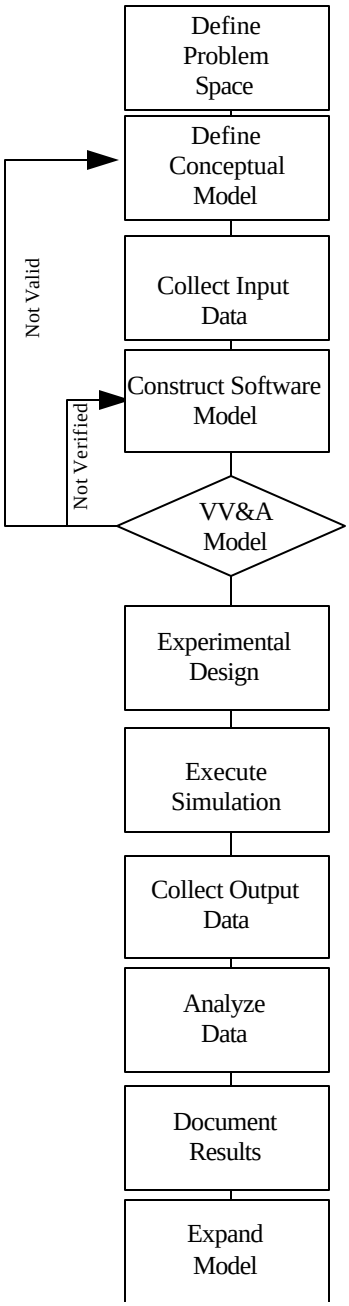


Figure 1. Simulation Development Process

The creation and operation of a simulation was once a black art in which only experienced practitioners could claim competence and understanding. However, over the last several decades a definite process has evolved for developing, validating, operating, and analyzing the results of simulations. In this section we will describe the process illustrated in Figure 1.

Define Problem Space. The first step in developing a simulation is to explicitly define the problem that must be addressed by the model. The objectives and requirements of the project must be stated along with the required accuracy of the results. Boundaries must be defined between the problem of interest and the surrounding environment. Interfaces must be defined for crossing these boundaries to achieve interoperability with external systems. A model can not be built based on vague definitions of hoped for results.

Define Conceptual Model. Once the problem has been defined, one or more appropriate conceptual models can be defined. These include the algorithms to be used to describe the system, input required, and outputs generated. Assumptions made about the system are documented in this phase, along with the potential effects of these assumptions on the results or accuracy of the simulation. Limitations based on the model, data, and assumptions, are clearly defined so that appropriate uses of the simulation can be determined.

The conceptual model includes a description of the amount of time, number of personnel, and equipment assets that will be required to produce and operate the model. All potential models are compared and trade-offs made until a single solution is defined that meets the objectives and requirements of the problem and for which algorithms can be constructed and input data acquired.

Collect Input Data. Once the solution space has been determined, the data required to operate and define the model must be collected. This includes information that will serve as input parameters, aid in the development of algorithms, and be used to evaluate the performance of the simulation runs. This data includes known behaviors of working systems and information

on the statistical distributions of the random variates to be used. Collecting accurate input data is one of the most difficult phases in the simulation process, and the most prone to error and misapplication.

Construct Software Model. The simulation model is constructed based on the solution defined and data collected. Mathematical and logical descriptions of the real system are encoded in a form that can be executed by a computer. The creation of a computer simulation, as with any other software product, should be governed by the principles of software engineering.

Verify, Validate, and Accredit the Model. Verification, validation, and accreditation (VV&A), is an essential phase in ensuring that the model algorithms, input data, and design assumptions are correct and solve the problem identified at the beginning of the process. Since a simulation model and its data are the encoding of concepts that are difficult to completely define, it is easy to create a model that is either inaccurate or which solves a problem other than the one specified. The VV&A process is designed to identify these problems before the model is put into operation.

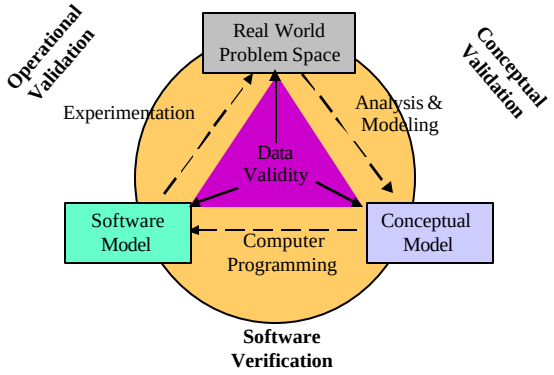


Figure 2. VV&A Process

For the purposes of VV&A the simulation development process is divided into the problem space, conceptual model, and software model with definite transitions and quality evaluations between these stages as shown in Figure 2. Validation is the process of determining that the conceptual model reflects the aspects of the problem space that need to be addressed and does so such that the requirements of the study can be met. Validation is also used to determine whether the operations of the final software model are consistent with the real world, usually through experimentation and comparison with a know data set. Verification is the process of determining that the software model accurately reflects the conceptual model. Accreditation is the official acceptance of the software model for a specified purpose. A software model accredited for one purpose may not be acceptable for another, though it is no less valid based on its original design.

Design Experiments. This phase identifies the most productive and accurate methods for running the simulation to generate the desired answers. Statistical techniques can be used to design experiments that yield the most accurate and uncompromised data with the fewest number of simulation runs. When simulation runs are expensive and difficult to schedule, experimental design can ensure answers at the lowest cost and on the shortest schedules.

Execute Simulation. This is the actual execution of the designed, constructed, and validated model according to the experimental design. The simulation runs generate the output data required to answer the problem initially proposed. In the case of Monte Carlo models, many hundreds or thousands of replications may be required to arrive at statistically reliable results.

Collect Output Data. Concurrent with the execution of the model, output data is collected, organized, and stored. This is sometimes viewed as an integral part of the model, but should be distinctly separated since it is possible to change the data collected without changing the model algorithms or design.

Analyze Data. Data collected during the execution of a simulation can be voluminous and distributed through time. Detailed analyses must be performed to extract long-term trends and to quantify answers to the driving questions that motivated the construction of the simulation. Analysis may produce information in tabular, graphic, map, animation, and textual summary forms. Modern user interfaces have greatly enhanced this phase by displaying data in forms that can be easily understood by diverse audiences.

Document Results. The results of the simulation study or training session must be documented and disseminated to interested parties. These parties identify the degree to which the simulation has answered specific questions and areas for future improvements.

Expand Model. Simulation models are expensive and difficult to build. As a result, once a model is built, it will be modified for use on many related projects. New requirements will be levied, new users will adopt it, and the entire development process will be conducted many times over.

FUNDAMENTAL TECHNIQUES FOR MODEL BUILDING

Though simulations vary widely in their design and implementation, most share a few common features to achieve their goals.

Event Management. A simulation is made up of states, events, and entities. States are groups of variables that describe the system at a specific time. Events are activities that change the state of the system. Entities are the objects represented in the simulation, the things described by the state variables and to which events occur. Events are the key items that make transformations in the model and drive it through its operations. These may include the arrival of a piece of material at a milling machine, the departure of an aircraft from an airport, the delivery of a message in a network, or an engagement between a missile and a fighter aircraft. These are typically managed through the use of multiple lists or queues in the model. The queues identify which events are ready to be processed, which are waiting until a specified time, and which must be triggered by specific conditions.

Queues manage events by ordering and releasing them according to specified criteria. The most common type of queues are the First In First Out (FIFO), Last In First Out (LIFO), Ordered, and Random. Each of these releases events into the simulation using a different method and each is uniquely useful for representing specific situations. The FIFO queue may contain a plan in the form of events to be executed. The LIFO queue may handle object reactions that interrupt and supersede planned events. An Ordered queue is widely used in training simulations, in which the time an event occurs drives its insertion into the simulation. A random queue assigns no order to the events in the queue, processing any one without regard for its priority or arrival order.

Time Management. In a simulation, the advancement of time is performed using a variable that can be controlled as any other and need not be tied to the advancement of real time or the internal computer clock. Typically, simulations move forward through the use of event based time (event stepped) or incremental time advancement (time stepped). An event-stepped simulation recognizes that in the model of the system, the only changes occur at the points at which events occur. Therefore, the model jumps from one scheduled event to the next, omitting the representation of intermediate times and speeding up the execution of the simulation by eliminating operations that do not effect simulation state. Time stepped simulations, on the other hand, are used when there are a large number of interactions between entities based on shared events. Training simulations use this method because of the need to present a consistent flow of time and events to a person that is interacting with the simulation.

A great deal of work has been done in the area of managing time stepped simulations. These simulations are discrete, stepping from one time to another using a specified increment. This increment may be fixed such that each step is one minute, one second, or one microsecond long, or it may be variable such that the size of the step is determined by the activity being simulated. It may be necessary, for example, to represent time in sub-seconds during a combat engagement and in days during the political negotiations prior to the beginning of hostilities.

As simulations have grown to operate across networks of computers or on parallel computers, the models have been separated into pieces that represent a portion of the problem, and exist as multiple software programs on multiple machines. This has resulted in the need to maintain consistency among programs, which can not be done effectively with the simple queuing lists used within a single program. Parallel and distributed time management was initially achieved through the use of a shared clock to which all programs would refer. However, research has lead to the use of algorithms that can ensure time synchronization without the use of a central shared clock.

Parallel and distributed time management can be accomplished through conservative or optimistic synchronization. Both methods use a mechanism to understand the time of each of the processes. Conservative synchronization then chooses to maintain consistency among all processes as the simulation executes. Optimistic synchronization, on the other hand, allows each process to move ahead as fast as computationally possible, putting each process at a

different point in time. When an event is received from another process that affects past events in the local process, the simulation reverses its operations and "rolls back" in time to include the new event interaction. The conservative method assumes that interactions between processes are common enough that constant synchronization is the most efficient method of proceeding into the future. The optimistic method assumes that interactions are scarce and the problem can be solved most efficiently by working as fast as possible. These roll back only when interactions are received in the past.

Random Number Generation. Many models require the use of random numbers to introduce the variability caused by statistical rather than deterministic representations of events. Random number generators are used in the computers to replicate the creation of a series of numbers that are random and independent of each other. These algorithms are actually deterministic and merely provide the impression of randomness. Algorithms are typically required to be repeatable, fast, use little storage space, and usually generate Uniformly distributed numbers in the range of (0,1). To create variates from other distributions the Uniform random number becomes the input to a second algorithm that generates Normal, Exponential, Poisson, Gamma, Weibull, Lognormal, Beta, Binomial, or other distributed numbers.

Physical Modeling. Traditionally, models have represented the capabilities of machinery and systems based on their physical characteristics and the basic laws of physics. The focus has been on understanding and representing the physical environment - distance, rate, weight, density, etc. In manufacturing systems, models represent entities entering a system in which events are generated by statistical distributions buffered by waiting queues. In analytical physics simulation, the models represent the specific behaviors of particles or chemicals under specified conditions. In training simulations, the models reproduce the physical world allowing people to interact with terrain, buildings, and other entities. Each of these takes a unique view of the essential variables and algorithms needed to represent the physical behavior of the system. The impact of human decision-making and process variance is handled through the use of statistical distributions that represent variation with the aid of random number inputs to these distributions.

Behavioral Modeling. As the role of simulation has grown, the need to more accurately represent human and group behaviors has increased. To accommodate these needs, simulation developers have turned to the artificial intelligence community for assistance. Models now contain finite state machines, expert systems, neural networks, and case-based reasoning to represent human behavior in finer detail. This acknowledges that specific physical conditions trigger human behaviors that must be explicitly modeled to achieve the appropriate results. Behavioral modeling has been particularly useful in training applications where computer controlled adversaries with challenging and realistic behaviors are required.

Model Management. A computer simulation is a system of software and hardware that must be developed and managed in accordance with the same principles of systems and software engineering that govern other applications. Issues that are not germane to the science of simulation are very important to the business of simulation. An attractive and friendly user

interface is important. Systems that have provided only textual input/output are giving way to those with graphical user interfaces and multi-dimensional representations of the simulated world. Configuration management of the models provides stability to a simulation program, ensuring that it can control its own evolution. Documentation provides permanence of expertise by recording model assumptions, algorithms, data collection, and validation results. This establishes a foundation that can extend the useful life of a model beyond the tenure of its original developers. Finally, domain architectures are being developed which attempt to capture the essential components and interactions of entire families of simulations. The intent is to create a structure that eliminates redundant development and promotes the reuse of modules or components that provide common functionality and interfaces across an entire family of simulations.

ESSENTIAL COMPUTER TECHNOLOGIES

Simulations, like all other applications, leverage technologies from other areas of science. The algorithms and information required to create a very complex model usually exceed the power of the available computer hardware and software necessary to run it. However, simulation programs are growing larger and more useful as a direct result of advancements in computer science. A few of the most useful technologies are described here.

Networks. The ability to distribute a simulation across a network of computers leads to more detailed, scaleable, complex, and accessible models. Distributed message passing and event synchronization allow a single problem to be addressed with a large number of traditional computers on a network. The proliferation of standardized networks between computerized machinery, communications systems, decision aids, and other tools has created an environment in which simulations can drive "real world" computers directly and extract data from them in real time. This has blurred the boundary between real and simulated worlds.

Parallel Computing. Parallel computing provides many of the advantages of networked computers, but adds the characteristic of close coupling. Some problems can be divided into many thousands of separate processes, but the interactions between them are so frequent that a general-purpose network for delivering messages introduces delays that greatly extend the execution time of the simulation. In these cases, parallel computers can provide the close coupling between processors and memory that allow the simulation to execute much more quickly.

Artificial Intelligence. The representation of human and group behavior has become essential in some parts of the simulation community. Techniques developed under the umbrella of artificial intelligence and cognitive modeling can solve some of these problems. Simulations are including more finite state machines, expert systems, neural networks, case based reasoning, and genetic algorithms in an attempt to represent human behavior with more fidelity and realism.

Computer Graphics. Simulation data lends itself very well to graphic displays. Factories and battlefields can be represented in full 3D animation using virtual reality techniques and hardware devices. Graphical user interfaces provide easy model construction, operation, data analysis, and data presentation. These tools place a new and more attractive face on simulations that previously relied on the mind's eye for visualization. This often leads to greater acceptance of the models and their results by the engineering and business communities.

Databases. Simulations can generate a large amount of data to be analyzed and often require large volumes input data to drive the models. The availability of relational and object oriented databases has made the task of organizing and accessing this information much more efficient and accessible. Previously, model developers were required to build their own storage constructs and query languages, a distraction from the real focus of the simulation study.

Systems Architecture. Simulations can be grouped into families, or domains, where the same software architectures can be used to model entire classes of problems. This recognition in transaction-based simulation has lead to the creation of a host of simulation products that encapsulate functionality used to model everything from factory operations to aircraft routing schedules.

World Wide Web. The expansion of the Internet and the World Wide Web has led to experiments with simulations that are either distributed through the Internet or accessible from it. These simulations make use of standard protocols and allow the distribution of a simulation across multiple computers that are not directly controlled on a dedicated network. Simulation users do not necessarily need to own the computers that run the simulation. Instead, the user may access a simulation-specific machine connected to the Web, provide input values, control model execution, and receive the results without ever having their own copy of the simulation software or the computers necessary to run it.

SIMULATION LANGUAGES AND TOOLS

A number of simulation languages and tools have been developed specifically to assist developers in constructing models of their systems. These languages are intended to serve a specific problem domain, rather than support general purpose programming as do FORTRAN, C, Pascal, and Ada. However, general purpose languages are still widely used to construct simulations in domains for which simulation specific languages or packages do not yet exist or where the problem is so unique that simulation tools can not be created economically.

Some of the more popular languages and tools are described below and a sample comparison of three provided in the table.

Discrete Event Simulation. Discrete event simulation includes a wide array of both problems and commercial tools for solving them. The descriptions below are separated into those that use an actual programming language and those that are simulation applications or toolkits.

Languages

Simula, the first simulation-specific programming language, was developed by O.J. Dahl and K. Nygaard of the Norwegian Computer Center. In 1961 they created Simula I by making extensions to ALGOL 60. The language was a general purpose programming language that included specific extensions to support simulation applications. Simula 67 was the first object oriented programming language, providing support for objects, classes, inheritance, encapsulation, multi-threading, and garbage collection. It was the motivation for the later development of the C++ language.

GPSS/H from Wolverine Software is a block programming language improved from the original GPSS developed at IBM in 1969. This language provides an interactive debugging environment, a floating-point clock, and built in mathematic, trigonometric, and statistical functions. It automatically collects basic simulation output data and supports the extension of this collection by the programmer.

SIMSCRIPT II.5 from CACI Products is an event-oriented and process-oriented language that evolved from the original SIMSCRIPT developed at the Rand Corporation in 1962. The language is actually a complete general programming language that can be used to build discrete-event, continuous, and combination simulations. It is supplemented by SIMGRAPHICS that allows the user to develop input forms, output displays, and interactive controls for the simulation.

SIMAN/Cinema from Systems Modeling is a combined simulation language and animation system. SIMAN models are constructed graphically using the Cinema package and automatically converted into code. The language includes built-in functions for manufacturing and material handling systems, an interactive debugger, and analyzers for input and output data.

SLAM II from Pritsker Associates is predominantly used for process-oriented simulation, with extensions that support event-oriented simulation and combinations of the two. The language represents models in a network-like structure that includes nodes and branches. Support packages allow the developer to draw a network, which is then converted into the simulation code.

MODSIM from CACI is an object-oriented programming language with graphic extensions to support data input, execution monitoring and control, and output analysis. The language includes built-in routines for statistical distributions and simulation management operations. Interaction with the language is through a development environment that includes a compiler, object manager, and debugger.

Language Comparison

To illustrate the syntax of some of the more common simulation languages, Table 1 provides the code for the same problem in GPSS/H, SIMAN, and SLAM II. This code represents a simple single-server queue with exponential inter-arrival and service times, such as a barbershop with one barber and a waiting queue. Though these languages are designed to efficiently serve the needs of model builders, their syntax is often equal or more complex than general purpose programming languages. This fact has motivated the creation of the simpler graphic packages and toolkits described in the next section.

Table 1. Simulation Language Comparison

GPSS/H	SIMAN	SLAM II

SIMULATE
GENERATE RVEXPO(1, 1, 0)
QUEUE SERVERQ
SEIZE SERVER
LVEQ DEPART SERVERQ
TEST L N$LVEQ, 1000, STOP
ADVANCE RVEXPO(2, 0.5)
STOP RELEASE SERVER
TERMINATE 1
 	
START 1000
END
	BEGIN;
	CREATE,,EX(1,1) :EX(1,1)
	:MARK(1);
	QUEUE, 1;
	SEIZE :SERVER;
	TALLY :1, INT(1);
	COUNT :1,1;
	DELAY :EX(2,2);
	RELEASE :SERVER :DISPOSE;
	END;
		GEN, 1,,,,,,72;
		LIM,1,1,100;
		NETWORK;
		RESOURCE/SERVER(1),1;
		CREATE,EXPON(1.0,1),1,1;
		AWAIT(1),SERVER;
		COLCT,INT(1),DELAY IN
		QUEUE,,2;
		ACTIVITY,EXPON(0.5,2),,DONE;
		ACTIVITY,,,CNTTR;
		DONE FREE,SERVER;
		TERM;
		CNTR TERM,1000;
		END;
		;
		INIT;
		FIN;

Tools

Extend from Imagine That is a visual, interactive simulation package for discrete event and continuous modeling that allows users to build models and user interfaces graphically. Model execution is carried out interactively on the graphic model representation. The package can accept data input through the interfaces or from separate files. Extend provides built-in mathematic and statistical functions and can be customized through the addition of C and FORTRAN routines. It also supports a client/server relationship with spreadsheet programs.

Workbench from SES is a visual simulation environment that allows models of complex systems to be built and executed graphically for performance analysis and functional verification. A model is specified graphically, as a hierarchy of directed graphs; declaratively, by filling in forms attached to each node in a graph; and procedurally, by specifying procedural methods attached to the nodes where desired in an internal language that is a superset of C.

TAYLOR II from F&H Simulations is a graphic model building package based on four fundamental entities - elements, jobs, routings, and products. These are manufacturing oriented where elements can represent machines, buffers, conveyors, transport, paths, warehouses, and

reservoirs. The three basic operations supported are processing, transport, and storage. During simulation execution, graphic interfaces provide 2D and 3D views of factory activities.

COMNET III from CACI Products is designed to simulate communications networks. It provides a graphic interface for model building, execution, and data analysis. It specifically provides statistical distributions and control data for communications and computer networks as used by telephone companies, cable television broadcasters, and computer networks.

BONeS Designer from the Alta Group models the protocol and messaging layers of computer architectures and communications systems. The tool provides graphical user interfaces for defining data structures, analyzing results, generating finite state machines, and directing interactive simulation runs.

CSIM18 from Mesquite Software is a library of classes, functions, procedures, and header files that describe the activities and statistical distributions of communications, transportation, microprocessors, and manufacturing systems. Library components can be combined with developed software in C and C++ to create a simulation model that has fast execution.

SimPack from the University of Florida is a toolkit written in C and C++ to support the development of simulation programs by the user. It contains routines to support basic simulation operations and management of declarative, functional constraint, and combination models. The software can be combined with software written by the user.

CPSim from BoyanTech provides an execution kernel that manages synchronization, scheduling, deadlock prevention, and message passing, as well as a library of C functions that can be used to build an application. CPSim represents the system being modeled as a directed graph of communicating objects that are categorized as sources, nodes, and sinks. The tool supports portable models across single and multi-processor computers.

Continuous Simulation. The Advanced Continuous Simulation Language (ACSL) from MGA Software was developed specifically for modeling time-dependent, nonlinear differential equations and transfer functions. The language allows the user to create code from block diagrams, mathematical equations, and FORTRAN statements

The Continuous System Modeling Program (CSMP) is constructed from three general types of statements - structural, which define the model, data, which assign numerical values to parameters, and control, which manage the execution of the model.

Interactive Simulation. In the interactive training arena a number of simulation products have emerged, particularly with military domain applications.

VRLink from MAK Technologies supports network protocols and simulation management for distributed military simulations. This package provides routines that format messages according

to defined standards and manage the delivery and receipt of these messages across a number of computer platforms.

ITEMS from CAE Electronics provides a graphic environment for constructing simulated virtual worlds and the entities that populate them. The tool allows the creation of vehicles, aircraft, and humans and the specification of the physical characteristics and behavioral patterns. Terrain and weather data can be imported from standard formats or generated internally to create an operational environment. It provides a graphical user interface for executing and managing simulation runs.

MultiGenII from Multigen Paradigm is a three dimensional modeling tool for generating the visual representations of simulated objects, terrain, and cultural features for a complete synthetic environment to support training simulations. The tool simplifies the creation of the visual objects, allowing simulation developers to focus on more specific physical and behavioral models within the simulation.

CONCLUSION

Simulation Expansion and Transformation. Like all computer applications, modeling and simulation is expanding as a result of improvements in computer hardware and software technologies. There was a time when simulation was performed entirely by dedicated personnel using expensive, dedicated computer systems. We have reached a point where significant simulations can be performed on personal computers by experts in a specific field, without the need for a staff of simulation specialists.

Research in simulation itself is leading to an array of new technologies and methods for constructing and using models. Innovations include formalisms for defining models, interoperability of a diverse set of interactive simulations, metamodeling, human behavior modeling, and concurrent simulation.

Future. The manufacturing, research, planning, and training communities have discovered that answers to their questions and insights into their problems can be obtained economically and quickly from simulation models. As the world evolves into an information society, more and more business, recreation, and government activities will be defined in the form of digital data which can be organized, analyzed, and predicted using simulation. This power will drive the wide adoption of simulation by all forms of business and government.

EXCELLENT REFERENCES

APPLICATION	REFERENCE
Manufacturing, Service Lines, and Health Care	Banks, J. Editor. 1998. *Handbook of Simulation: Principles, Methodology, Advances, Applications, and Practice*. New York: Wiley-Interscience. Law, A. and Kelton W. 1991. *Simulation Modeling and Analysis*. New York: McGraw Hill.
Interactive Training and Entertainment	Kuhl, F., Weatherley, R., and Dahmann, J. 2000. *Creating Computer Simulation Systems: An Introduction to the High Level Architecture*. Upper Saddle River, NJ: Prentice Hall PTR. Singhal, S. and Zyda, M. 1999. *Networked Virtual Environments: Design and Implementation*. New York: Addison Wesley.
Simulation System Design and Tools	Fishwick, P. 1995. *Simulation Model Design and Execution*. Englewood Cliffs, NJ: Prentice Hall. Fujimoto, R. 2000. *Parallel and Distributed Simulation Systems.* New York: Wiley-Interscience. Nance, R. 1996. "A History of Discrete Event Simulation Programming Languages". *The History of Programming Languages - II*. New York: Association of Computing Machinery. Schriber, T. 1991. *An Introduction to Simulation Using GPSS/H*. New York: John Wiley.
Analysis of Military Problems	Hughes, W.P. Editor. 1997. *Military Modeling for Decision Making*. Alexandria, Virginia: Military Operations Research Society. Knepell, P. and Arangno, D. 1993. *Simulation Validation: A Confidence Assessment Methodology*. Los Alamitos, CA: IEEE Press. Morse, P.M. and Kimball, G.E. 1998. *Methods of Operations Research*. Alexandria, Virginia: Military Operations Research Society.

APPENDIX A: 35 SIMULATION WEB SITES

A great deal of information on simulation is available on the World Wide Web. The addresses of some of the more useful and dynamic organizations are provided below.

Conferences and Discussions

Winter Simulation Conference
http://www.wintersim.org/
Simulation Interoperability Workshop
http://www.sisostds.org/
Electronic Simulation Conference
http://www.scs.org/confernc/elecsim/elecsim.html
International Journal of Computer Simulation, Modeling, and Analysis
http://tebbit.eng.umd.edu/simulation/
On-line Executable Simulations
http://www.cis.ufl.edu/~fishwick/websim.html
http://ms.ie.org/websim/survey/survey.html
Discrete Event Simulation Server
http://masg1.epfl.ch/roso.mosaic/nino/devs.html
Systems Understanding Server
http://www.radix.net/~crbnblu/welcome.html
USENET News Group - "comp.simulation" archive
http://tebbit.eng.umd.edu/simulation/comp.simulation.archive.html

Military Offices

Defense Modeling and Simulation Office
http://www.dmso.mil/
Joint Modeling & Simulation System
http://www.jmass.wpafb.af.mil/
Joint Simulation System
http://www.jsims.mil/
Joint Training, Analysis, and Simulation Center
http://www.jtasc.jfcom.mil/
Joint Warfare Simulation
http://www.dtic.mil/defenselink/jwars/
Joint Warfighting Center
http://www.jwfc.js.mil/

Professional Organizations

ACM Special Interest Group on Simulation

http://www.acm.org/sigsim

IEEE Computer Society - Technical Committee on Simulation
http://www.ieee.org/

Society for Computer Simulation
http://www.scs.org/

Institute for Operations Research and Management Science
http://www.informs.org/

INFORMS College on Simulation
http://www.isye.gatech.edu/profOrg/informs/informs-sim

Military Operations Research Society
http://www.mors.org/

Companies

CACI
http://www.caci.com/

AutoSimulations
http://www.autosim.com/

Taylor II
http://www.taylorii.com/

Imagine That
http://www.imaginethatinc.com/

MultiGen
http://www.multigen.com/

Mak Technologies
http://www.mak.com/

Evans & Sutherland
http://www.es.com/

BoyanTech Inc.
http://www.wdn.com/bti-sim

The Alta Group
http://www.altagroup.com

SES Inc.
http://www.ses.com/

MGA Software
http://www.mga.com/

Abstraction Software
http://www.abstraction.com/abstraction

High Performance Systems
http://www.hps-inc.com/

Virtual Prototypes
http://www.virtualprototypes.ca/

APPENDIX B: 23 UNIVERSITIES WITH SIMULATION CURRICULA

Arizona State University
California Institute of Technology
California State University, Chico
Carnegie Mellon University
George Mason University
Georgia Institute of Technology
Johns Hopkins University
Naval Postgraduate School
North Carolina State University
Texas A&M University
University of Arizona
University of California at Los Angeles
University of Central Florida
University of Colorado
University of Florida
University of Illinois Urbana-Champaign
University of Michigan
University of Pennsylvania
University of Southern California
University of Texas
University of Virginia
University of Washington
Virginia Polytechnic University

Trademarked products referenced in the "Simulation Languages and Tools" section of the paper:

Trademark	Trademark Holder
GPSS/H	Wolverine Software
SIMSCRIPT II.5	CACI
SIMGRAPHICS	CACI
SIMAN/Cinema	Systems Modeling Corp.
SLAM II	Pritsker Associates
MODSIM	CACI
TAYLOR II	F&H Simulations
COMNET III	CACI
BONeS Designer	The Alta Group
CSIM18	Mesquite Software
CPSim	BoyanTech Inc.
ACSL	MGA Software
VRLink	MAK Technologies

ITEMS	CAE Electronics
MultiGenII	MultiGen Corp.