

6 Developing a Simulation-Implementation

After the model has been conceptualized, coded, verified, and validated, it is ready to provide information through experimentation. Experimentation is the process of initializing key parameters in the model and setting up production runs to make inferences about the behavior of the system under study. The experimentation process consists of three steps:



Figure 6.1 Experimentation Process

www.sylvania.com

We do not reinvent the wheel we reinvent light.

Fascinating lighting offers an infinite spectrum of possibilities: Innovative technologies and new markets provide both opportunities and challenges. An environment in which your expertise is in high demand. Enjoy the supportive working atmosphere within our global group and benefit from international career paths. Implement sustainable ideas in close cooperation with other specialists and contribute to influencing our future. Come and join us in reinventing light every day.

Light is OSRAM

OSRAM SYLVANIA



6.1 Experimental Design

The first step in the experimentation process is experimental design. The term experimental design implies that an experiment is being designed or set up. This is done by identifying key parameters and initializing these parameters to values of interest. An example of an input parameter that could be the focus of an experiment is the number of doormen needed at the 21-Club (See section 6.3). The simulation could be initialized with either one or two doormen. Other issues that need to be resolved during the experimental design stage follow:

Length of Simulated Run Time: Simulation run time must be determined during the experimental design stage. Simulations can be either terminating or steady-state. Terminating simulations end at a predetermined time. For example, if a barber shop opens at 9:00 AM and closes at 5:00 PM each day, a terminating model could be set up with run times that end specifically after eight hours of operation. A steady state simulation on the other hand is set up to measure the long run performance of a system. A model running in the steady-state mode is normally run for a period of time and then statistically "reset" to remove any bias caused by unusual start-up conditions. The run is then continued and steady state results are tabulated.

Replications: Since simulation is an approximation, it is important to treat it as such. This means that one run of the model will not usually provide an absolute answer. The random inputs to a simulation will produce outputs exhibiting variability. The model should be run enough times to produce a sample size from which a mean and standard deviation with a satisfactory confidence level can be calculated.

Reseeding Random Number Streams: Each replication should either re-seed the random number streams being used, or start in the stream where the last run left off. If pseudo-random numbers are being used and each run starts with the same seed value, no variability will occur between runs. Each run would be using the same number stream.

Initial Conditions: In some models, it is possible to force the system to a particular state upon start-up. This action may be desirable in the case of a manufacturing system model that will either never naturally cycle to an unusual condition or may cycle to that condition but only after hours of run time. By setting the initial conditions of a model to certain values, experimentation can be facilitated.

Variables of Interest: The analyst will need to know what values are of interest. For instance, are two systems being compared in terms of throughput? Or are the numbers of workstations being determined? Experimentation will need to be constructed considering the values being examined.

6.1.1 Random and Pseudo-Random Number Streams

A variable Z which can assume any value in the range (x,y) with equal probability is defined as being random (Figure 6.7). Random variables can be uniformly or non-uniformly distributed. They may be continuous or discrete. Random variables are selected by chance and are not influenced in any way by past values. Random numbers are widely used in experiments that are dependent upon chance. Examples of these are machine breakdowns, occurrences of rejected parts coming off an assembly line, and service times at an automatic teller machine.

Random numbers can be generated in many different ways. The flip of a coin, drawing numbers from a hat, roulette wheels, and playing cards are examples of physical generation methods. The most common method of obtaining random numbers for use in the computer simulation field is through algebra. A number called a seed is selected and used to produce a sequence of numbers in the following manner (Figure 6.2):

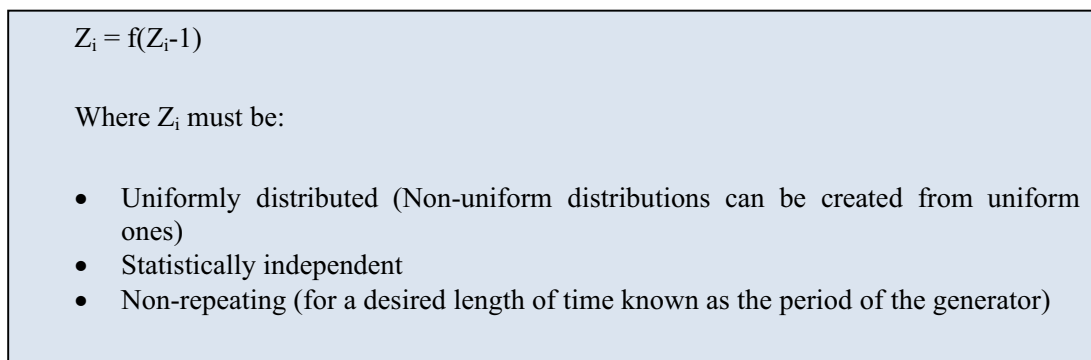


Figure 6.2 Method for Obtaining Random Numbers for Use in Simulation

Although the resulting number stream from the preceding example is not truly random, if created properly, it will pass most tests for statistical randomness. Number streams generated through algebraic means are known as pseudo-random numbers. The use of pseudo-random number streams has several advantages over the use of true random numbers including:

1. The sequence of numbers is reproducible. This means different versions of a program can be tested using the same input data.
2. The streams can be generated quickly and efficiently for use in a simulation program.

Although it is important to understand the concepts of random number generation and pseudo-random number streams, a simulation analyst will rarely have to create his or her own random number generator. Most simulation software products already have a built in means of creating pseudo-random numbers. As long as care in selecting a seed value (this is usually done for the analyst as well) is taken, very few problems should occur. However, this is not a topic to be taken too lightly. Without a reliable stream of random numbers, simulation results would be rendered invalid.

6.1.2 Special Concerns for Terminating Simulations

Terminating simulations are models that represent a system that starts in a particular state and then terminates or ends after a fixed period of time or when a predetermined condition is reached. Terminating simulations are used to model particular manufacturing operations where the characteristics of the system startup are important to include or in situations where a specific period of time is being analyzed, like weekend ticket sales for a newly released movie. Terminating simulations study a system over a predefined time period like a shift, a day, or a week. An analyst using a terminating simulation makes no attempt to bring the model to a steady state or stable operating condition. Instead, model replications include all startup biases that might exist (as do most real world systems).

It is important to remember that terminating simulations are still dependent on random samples from input distributions and each run must be treated as a single observation. This means replications using different random number seeds must be generated and used to compute a confidence interval or range likely to include the true value of the mean. Since terminating simulation runs produce output data exhibiting statistical independence, the analysis becomes easier.



360°
thinking.

Deloitte.
© Deloitte & Touche LLP and affiliated entities.

Discover the truth at www.deloitte.ca/careers



6.1.3 Special Concerns for Steady State Simulations

Steady state simulations are intended to examine a system from which startup and ending biases are removed. In other words, the analyst seeks to achieve a state in the model where conditions remain stable. While in many instances this might be less realistic than a terminating simulation, it can be useful in determining underlying characteristics of the system and filtering out the statistical noise. Two main challenges exist when developing a steady state simulation. The first is determining whether a steady state condition has been achieved. The second deals with statistical independence of the derived samples.

Several techniques are used to deal with the first challenge. Almost every system experiences a period of time upon startup where resources are being acquired, customers are entering, or the initial stages of the system are in use while subsequent stages have not yet filled. These forces can result in a skewing of output statistics often called initial bias (See Figure 6.3). Several practical techniques are used to avoid the impact of startup bias. These include:

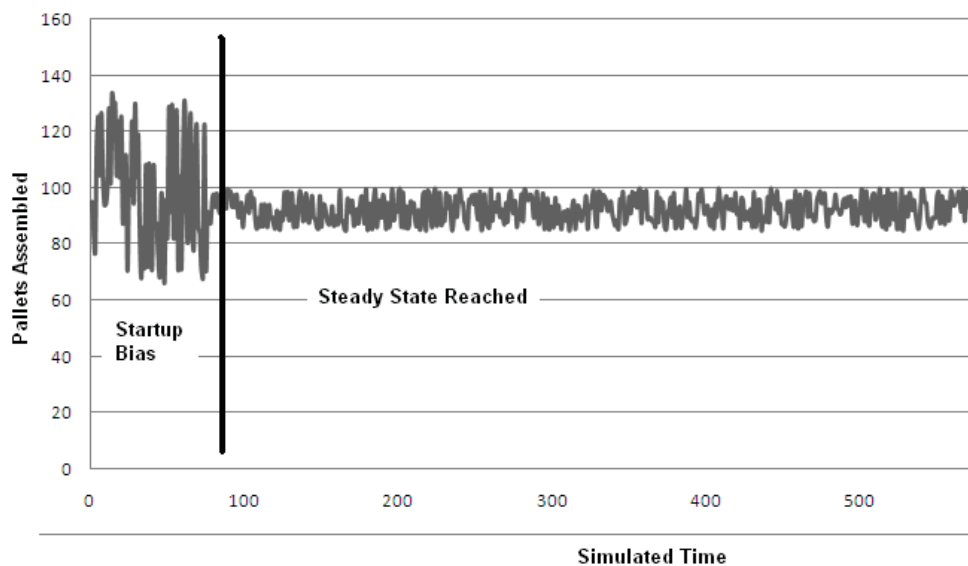


Figure 6.3 Initial Bias in Steady State Simulation

1. A warm-up period is determined after which model statistics collection is restarted. Most simulation software packages have built-in the capability of resetting statistics. In a sense, this would represent throwing out all data collected to the left of the dark line in Figure 6.3 and only considering data after that point in time. The analyst may have to review the simulation results and decide when the model appears to have entered steady state.
2. Initializing the model with a realistic, steady state condition of resource utilization and customers or other entities strategically placed within the model. This technique can work but must be verified as accurate and free of other forms of startup bias. Some researchers call this 'preloading' and others call it priming the model.
3. Data analysis can be used to remove the bias but this technique is rarely used.

The second challenge is to ensure collected samples from the simulation are independent. In a steady state simulation, the current state of the model is dependent on the previous state. Therefore, independence cannot be assumed. In order to correct this problem, a series of simulation runs (often called replications) can be constructed in a way that collection of statistics is suspended between replications while the model runs in steady state. The resources and entities are not cleared, just the statistics. Each replication can be collected with a sufficient run period (almost like the initial bias period) between to ensure independence. Figure 6.4 illustrates. Of course, the analyst would have to ensure the periods selected for statistical collection and those discarded were of sufficient length to ensure the goal of independence is achieved.

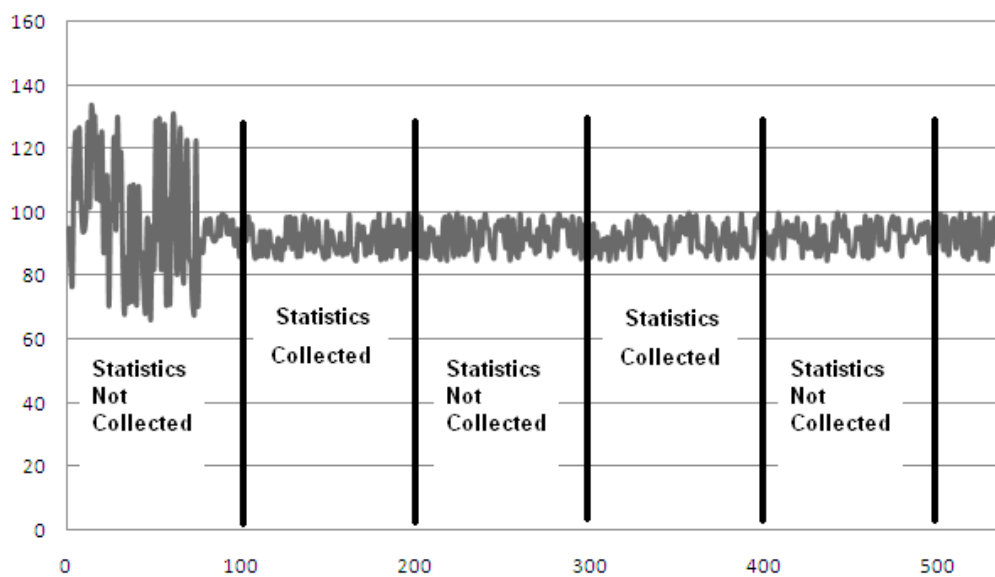


Figure 6.4 Removal of Startup Bias in Steady State Simulation

6.2 Production Runs

After the desired experiments have been identified and the input parameters properly set, production runs can be made. Production runs are no more than executing the simulation code and creating output reports for each desired scenario. Production runs must consider the simulation type (terminating vs. steady state) and the variable(s) to be studied. Enough runs must be completed to construct a confidence interval and ensure collected data is representative of the variation in the system. If a confidence interval is constructed using the output data, the likelihood that the model's true characteristics are included is a probability of $1-\alpha$. Three main factors influence the width of a model's confidence interval:

1. Desired confidence level (e.g. α)
2. Replication count – Higher numbers of replications can produce a better (narrower) confidence interval.
3. Variation of parameter being collected (s) – Greater variation produces a less desirable confidence interval and must be offset with more replications.

It is important to note that the collected data represents the model's characteristics. If the model has not been constructed properly, or contains invalid input data or misguided logical assumptions, all the statistical analysis in the world cannot force accurate results to emerge from its operation.

6.3 Output Analysis


The final phase of experimentation involves analysis of model outputs. If the output produced satisfies the simulation project's objectives, the experimentation stage is over. If problems or new questions are raised, the model may be returned to an earlier phase for modification or for production of additional runs.

6.3.1 Statistical Output Analysis of a Single Model


As stated in section 6.2, when a simulation has any type of stochastic behavior incorporated into its structure, it becomes necessary to view the results of a production run as an experimental sample. Enough samples must be taken to ensure that output variability can be accounted for in terms of a mean and standard deviation. Confidence intervals should be created to give the model users a range within which a parameter's true value is likely to fall. The following case study illustrates:

SIMPLY CLEVER

ŠKODA



We will turn your CV into an opportunity of a lifetime



Do you like cars? Would you like to be a part of a successful brand? We will appreciate and reward both your enthusiasm and talent. Send us your CV. You will be surprised where it can take you.

Send us your CV on www.employerforlife.com



Oslen AGV System

Oslen Manufacturing uses a small AGV system to move material in their warehouse. With a recent increase in orders, they need to coax more productivity out of their aging system. Their goal is to develop an accurate model of the existing system, then later use it to compare various options for improvement. After developing the model, they generated the following set of data representing 100 eight hour shifts. The model uses a terminating approach since it was determined that startup conditions needed to be included for the model to be accurate. The following data was collected (see

44	46	39	52	38
30	45	60	19	35
40	30	40	52	45
65	26	33	45	47
16	32	35	12	29
30	37	14	58	70
47	64	43	37	50
39	60	30	49	23
37	25	48	23	74
49	83	29	18	56
59	49	32	39	38
57	55	51	39	40
14	46	43	61	34
65	19	11	57	31
52	52	55	45	23
31	62	65	54	33
29	56	63	45	29
35	46	57	59	66
59	49	29	33	39
55	54	35	20	10

Table 6.1 Oslen's Data Set

Simulation analysts at Oslen used the following criteria:

1. Desired confidence level (e.g. α) = .05
2. Replication count = 100

In order to determine if the sample size was large enough, Oslen's used SAS software to develop a confidence interval with a 95% chance that the mean for their model's performance had been captured. First they conducted a test for normality to ensure the confidence interval could be constructed with standard statistical practice. SAS provides a variety of statistics for testing normality as shown in Figure 6.5.

Tests for Normality

Test	--Statistic--	-----p Value-----
Shapiro-Wilk	W 0.988172	Pr < W 0.5210
Kolmogorov-Smirnov	D 0.055116	Pr > D >0.1500
Cramer-von Mises	W-Sq 0.055786	Pr > W-Sq >0.2500
Anderson-Darling	A-Sq 0.356909	Pr > A-Sq >0.2500

Figure 6.5 Test Statistics Used to Check Normality

The Shapiro-Wilk W of .988 does not reject the null hypothesis that the variable is normally distributed ($p < .521$). Likewise, Kolmogorov-Smirnov, Cramer-von Mises, and Anderson-Darling tests do not reject the null hypothesis. Additionally, the plots (shown in Figure 6.6) indicate normality cannot be rejected.

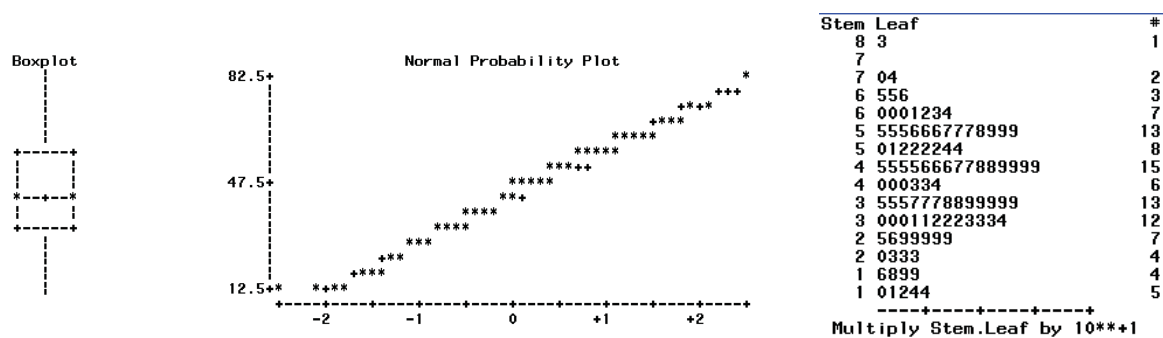


Figure 6.6 Plots Demonstrating Normality of Collected Data

Given that the data is assumed normally distributed, a 95% confidence interval can be constructed to represent the model's mean loads moved per eight hour shift. SAS indicated a mean of 42.49 with a 95% confidence interval (39.45,45.53) as shown in Figure 6.7.

Basic Statistical Measures				The SAS System	
Location		Variability		The MEANS Procedure	
				Analysis Variable : SIMRUN	
				Lower 95% CL for Mean	Upper 95% CL for Mean
Mean	42.49000	Std Deviation	15.30927	39.4523094	45.5276906
Median	43.50000	Variance	234.37364		
Mode	29.00000	Range	73.00000		
		Interquartile Range	23.50000		

Figure 6.7 Basic Statistical Measures and Confidence Intervals

6.3.2 Statistical Output Analysis Used to Compare Models

It is often desirable to use computer simulation to compare alternatives. Several considerations are required for this use. First, it is important to retain independence within each data set (techniques for doing this were discussed in the terminating and steady state sections of this chapter). It is also important that independence is maintained between data sets being compared. This means, in general, that different random number seeds should be used when running the different model scenarios. Other possible correlations can be reduced by using large sample sizes or by ensuring each data point is the average of multiple observations. The Oslen Manufacturing case is continued to illustrate comparing models.

Oslen AGV System (Continued)

After developing a baseline model of existing operations and ensuring output matched the real-world system, Oslen simulation analysts made modifications to the model to represent a new computer system. This effectively scheduled and routed the AGVs according to more sophisticated algorithms that use artificial intelligence to anticipate future load movements based on past patterns. By purchasing new software, the expense of redeveloping the entire system could be avoided. After close consultation with the software vendors, the model was modified and the following data was collected (Table 6.2):

I joined MITAS because
I wanted **real responsibility**

The Graduate Programme
for Engineers and Geoscientists
www.discovermitas.com



Month 16
I was a construction supervisor in the North Sea advising and helping foremen solve problems

Real work
International opportunities
Three work placements







42	56	44	64	45
31	51	60	21	32
38	29	39	64	51
78	32	33	52	53
19	35	37	13	36
62	45	16	57	79
46	61	43	36	57
38	57	34	49	25
42	30	55	27	85
48	89	33	20	70
65	55	34	41	46
55	63	56	40	41
16	56	41	59	34
68	22	11	59	33
51	65	62	60	27
33	68	66	61	33
33	62	73	45	31
35	49	65	68	75
64	50	20	40	40
30	33	40	19	11

Table 6.2 Oslen's New Data Set

The new data set was also found to be normally distributed (see Figure 6.8).

Tests for Normality

Test	--Statistic--	-----p Value-----
Shapiro-Wilk	W 0.986916	Pr < W 0.4320
Kolmogorov-Smirnov	D 0.051112	Pr > D >0.1500
Cramer-von Mises	W-Sq 0.043771	Pr > W-Sq >0.2500
Anderson-Darling	A-Sq 0.319491	Pr > A-Sq >0.2500

Figure 6.8 Normality Statistics for New Data Set

Since data from both runs are normally distributed, a simple t-test can be used to test for differences. Had the data sets not been normally distributed, then a Mann-Whitney test would have been used. The data analysis features of an Excel spreadsheet were used to run the t-test. Figure 6.9 illustrates the outcome.

	Data Set 1	Data Set 2
Mean	42.48	47.35
Variance	234.19	307.04
Observations	100	100
Pooled Variance	270.62	
Hypothesized Mean Difference	0	
Df	198	
t Stat	-2.09	
P(T<=t) two-tail	0.04	
t Critical two-tail	1.97	

Figure 6.9 t-Test Statistics

The mean of the modified simulation is 47.35. This compares to the original mean of 42.48. The t-stat of 2.09 provides a probability less than 4% that the means are not different ($p < .04$). The simulation analysts at Oslen's Manufacturing concluded the new computer algorithms for their AGV system could result in an approximate 11% gain in productivity. A cost/benefit analysis then could be conducted with these simulation results.

6.4 Output Reporting

Knowing what statistics to gather from a simulation is a very important part of the simulation process. Most models have the capability of providing much more information than is required. The simulation analyst must be able to discern between the summary data that will be useful to the model's customers and the detailed supporting data which verifies that the model is performing as desired. When the statistical work has been completed, results that suit the consumer of the information must be constructed.

6.4.1 Simulation Report

A simulation report is a document intended to communicate the results of a simulation study. A typical simulation report should contain the following elements:

1. Title page – The title page should contain the name of the system being simulated, the date of the report, a revision number or letter, the party for whom the report is being generated, and the name and/or company of the simulation analyst.
2. A short executive summary might be included directly following the title page to provide the important information in a quick, easily understood way.
3. Table of contents – If the report is more than a few pages long, a table of contents should be included.

4. Statement of objectives – The goals and objectives for the simulation study including any specific questions that were to be answered should be summarized in this section of the report.
5. Methodology – This section should include simulation input assumptions and a brief statement describing the general methods used to develop the model. It should be kept in non-technical terminology. The limitations of the model should also be discussed here.
6. Conclusions and recommendations – A summary of the major findings and any recommendations are included in this part of the output report document.
7. Findings – This section of the report should provide data that backs up the “Conclusions and Recommendations”. Graphs, tables, and charts that make the data easier to understand and more readable should be used. All data needs to be fully explained.
8. Appendix – The appendix should contain the actual simulation code, all input data, raw output data, reference materials, and calculations.

ie business school

#1 EUROPEAN BUSINESS SCHOOL
FINANCIAL TIMES 2013

#gobeyond

MASTER IN MANAGEMENT

Because achieving your dreams is your greatest challenge. IE Business School's Master in Management taught in English, Spanish or bilingually, trains young high performance professionals at the beginning of their career through an innovative and stimulating program that will help them reach their full potential.

- Choose your area of specialization.
- Customize your master through the different options offered.
- Global Immersion Weeks in locations such as London, Silicon Valley or Shanghai.

Because you change, we change with you.

www.ie.edu/master-management | mim.admissions@ie.edu |

Download free eBooks at bookboon.com



Click on the ad to read more

6.4.2 Quick Reports

Many simulation languages provide special output report generation facilities. These tools can be used to produce graphs and tables, as well as perform statistical analysis. For routine simulation runs that are used by the same customer time after time, a standard format can be developed. The simulation can then produce a report automatically after each production run. An example demonstrating the usefulness of this “quick report generation” can be found in a job shop environment. Each time a new order is received, a simulation is performed to determine how long certain production machines will be in use. The same model is used for each new order with its input parameters set to reflect the quantities and types of parts being built. A standard report is generated by the simulation and copied to the appropriate shop personnel.

6.4.3 Logic Transfer

In some applications, more than an output report is needed to effectively communicate the results of a simulation study. In order to ensure all subtle aspects of a complex system’s model are implemented in the real world, a logic transfer should be performed. Logic transfer involves moving information, data, or code from a computer simulation into the real world computer that will be controlling the actual system.

Simulations requiring logic transfer often are being used as design tools. They are precursors to the actual system being built. The simulation team has a goal to identify all logic necessary to implement a controller or other complex logic to be used in system development. This means the simulation team must effectively communicate its developed and debugged algorithms to the system design team. This ensures that the required logic is incorporated into the actual controller for the real system to operate just like the model did. Four methods of facilitating logic transfer have been devised: Philosophic Transfer, Pseudocode Transfer, Database Transfer, and Actual Code Transfer.

Philosophic Transfer: The most common and perhaps least efficient method of transporting simulation logic into actual system control logic is the philosophic transfer. Under this scenario, logic for the real-world system is based on the key ideas and assumptions used in the simulation. This information is presented to the system design team either verbally or in the form of a written report. The design team then evaluates the simulation findings and uses them as a guide or criterion in implementing the actual system software.

In many simple cases, the philosophic transfer works quite well. However, there are some inherent shortcomings that can cause inefficiencies. One of these inefficiencies is duplication of effort. The system software is designed, coded, debugged, and tested. This same process has already been used in the development of the simulation. Another potential shortcoming is the possibility that a piece of information was not communicated properly resulting in either a misinterpreted or omitted portion of logic. A subtle difference in logic can produce a discrepancy that renders the simulation results invalid. The philosophic transfer method tends to make simulation validation (Carson, 1986) a more complex task. Since the two software systems were developed in a somewhat independent fashion, the differences become harder to identify. Whether the simulation accurately depicts the actual system becomes a difficult question with an answer that is hard to prove.

The best way to ensure that these problems don't occur when employing the philosophic transfer method is to form a system design team consisting of both simulators and software engineers. A common design can be arrived at and ideas can be tested with a simulation. When the design is finalized and the simulation is complete, the system software can be written. If the project has emphasized communication and structure, the transfer of the philosophy contained in the simulation should be successful. Duplication of effort has been minimized by doing the initial design work collectively. The ideal philosophic transfer approach to implementing a simulation in the real world is characterized by team work and communication.

Pseudocode Transfer: The second method of transferring data from a simulated controller to an actual controller is the pseudocode transfer method. This approach is very similar to the philosophic transfer but it takes the level of detail a step further. It evolved from the need to ensure that all assumptions incorporated into a simulation were addressed and made apparent to the actual system design team (Figure 6.10). In this scenario, pseudocode is generated by simulation personnel as a method of documenting what has been implemented. This pseudocode is analyzed and rewritten for the actual controller by system software engineers.



Figure 6.10 Pseudocode Transfer

The pseudocode method of transfer offers the advantage of being more detailed. Therefore, the number of omissions and oversights occurring are expected to be fewer than would be seen when using the philosophic transfer method. Simulation validation is also made easier. The logic rules used in the control system can be examined and verified as identical to those in the simulation. Differences are identified readily and the pseudocode provides a common record for both the simulation and system software package.

A disadvantage associated with using this method of transfer is duplication of programming effort. The pseudocode has to be rewritten in the language of the controller, debugged, and tested. Some of the simulation team's time is also required to rewrite their logic in the form of pseudocode.

Communication between the simulation team and the system design team is important under this transfer scenario; but not nearly as crucial as when using the philosophic transfer method. By placing a structured and detailed pseudocode document in the hands of the design team, little additional interaction will be required of the simulators. It is not until the verification and validation phase of the project, when the completed system software is compared to the simulation, that further communication will be required.

Database Transfer: The third method of transferring simulation logic to an actual system is known as a data base transfer. The essence of this transfer method is to transport a data base, developed in the simulation, and use it to drive the actual system controller. It is important to note that this method of data transfer is contingent upon software existing in both the simulation and controller. This requires that an initial development project must be undertaken to define the database and delineate how it will be used. After this has been established and the simulation and controller software have been developed, the data base can be transferred and utilized. This type of transfer will most commonly be used in cases where many similar systems are being designed using a generic simulation and a generic controller.

The advantages to using this transfer method are quite apparent. The debugging of the data base and its testing are all done in the simulation phase. Nearly all duplication of programming effort is eliminated (after the initial system has been created). Changes to the simulation and actual system can be done easily and consistently. Simulation validation and verification becomes much easier. In addition, communication and the passing of abstract concepts becomes less of an issue.



"I studied English for 16 years but...
...I finally learned to speak it in just six lessons"
Jane, Chinese architect

ENGLISH OUT THERE

Click to hear me talking before and after my unique course download



A potential drawback encountered when using the data base transfer method is the loss of flexibility. Although using a data base simplifies logic and provides the system engineers with a standard, unusual cases and exceptions become more difficult to incorporate. For this reason it is very important that the data bases be designed in a comprehensive manner.

Actual Code Transfer: The final method of data transfer to be examined is transporting actual code from simulation to system controller. This can be done either when the simulation has been written in a conventional language such as C# or when using a specialized simulation language such as GPSS/H or GPSS/PC which has the capability of calling external subroutines. These subroutines contain code that will be used in the system controller. The FORTRAN subroutines can be written either by the simulators or by the system design team prior to implementation in the simulation. Duplicated effort is reduced because the control logic is written only once in the simulation phase and then reused in the actual system.

Using this actual code transfer method requires that the simulators and system designers work very closely. It may be advantageous to form one team consisting of both simulation and design personnel. A disadvantage inherent when using this method is the requirement of a more concerted initial effort and additional time to debug the model. The simulator may not be familiar with the control algorithm and the system designer may not be familiar with the rest of the simulation. When the model is run and debugged, some learning time might be required. However, testing and debugging the simulation is also testing and debugging the actual system.

6.5 Post Processing Output

All models must provide some type of output. This output can be communicated or presented to the end user in many different ways. Traditionally, simulation output data took the form of hard to understand stacks of numerical data. Some simulation analysts thrive on reams of paper covered with statistics. Coupled with this overkill tendency of the analysts, is the ease with which most simulation languages allow data to be gathered and reported. Nearly all simulation software products automatically make an abundance of data available at the end of each run.

Knowing what data to report and what data to omit are challenges for the simulation analyst. A good rule of thumb is to make the data as readable as possible and limit the quantity to the essentials. Two types of output data are usually available from a simulation. The first type describes the model itself and will help the analyst verify the model. This does not need to appear in a simulation report. The second type of data is the output that describes characteristics of the system being modeled. This data will be of interest to the end user and should appear in some type of output report. Tables, graphs, and bar charts can all be used to make the raw data more readable.

Although many simulation products contain data analysis tools, it may still be desirable to explore some commercially available software. The reason for this is two-fold. First, the function of a simulation software product is usually modeling, not data presentation. Second, excellent data presentation software is available. This software has been developed specifically for putting data into a form that can be easily understood and communicated. The simulation analyst can use these types of packages to great advantage. Many different data analysis and presentation software packages are currently available. Two of these are briefly profiled in sections. 6.5.1 and 6.5.2.

6.5.1 MS-Office Excel

A very popular method of transforming raw simulation data into an easy-to-read report is through the use of spreadsheet software. Many spreadsheet programs are currently available from a wide variety of sources. The most popular of these is MS-Office Excel. Spreadsheets are programs which allow the user to manipulate data that has been entered into a series of cells organized into rows and columns. Although it is possible to enter the data into the spreadsheet from the computer keyboard, the simulation analyst will probably import a raw simulation output data file into the spreadsheet. Excel provides the capability of performing this function in a variety of ways with various data import tools. Many of these can be located on the data ribbon (see Figure 6.11).

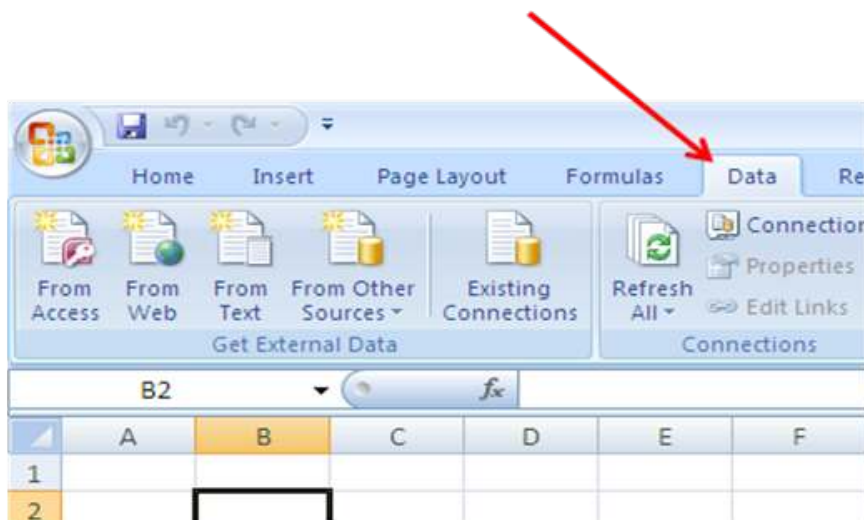


Figure 6.11 Data Ribbon in MS-Excel

The first three icons on the data ribbon provide capability to import data from MS-Access, Web pages, and text files. Many simulation products create output files that can be imported using the text option. This involves browsing out to the folder where the file is located, then selecting it for import.

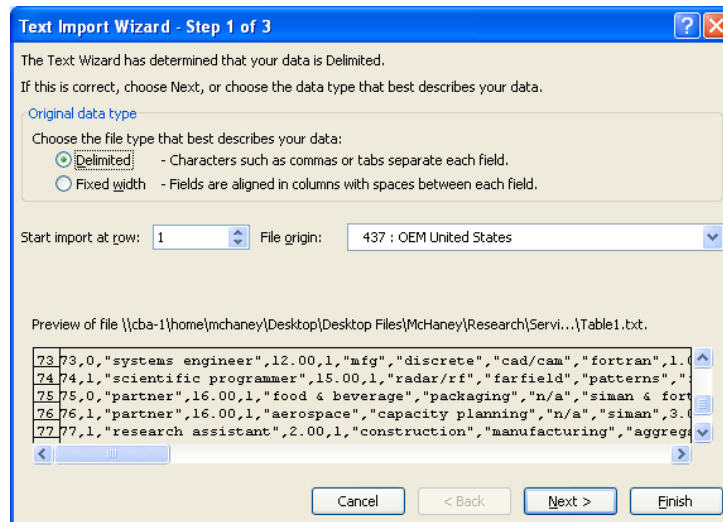


Figure 6.12 Importing Data in MS-Excel

Excel uses a wizard that will enable the analyst to select the delimiting character (space, comma, et cetera) and then import the data for analysis (see Figure 6.12). Excel also allows the import of data tables from Web pages. Using the 'From Web' button launches a browser window. The analyst can paste the URL where the data resides and a list of eligible tables will be displayed (See Figure 6.13). Clicking on a table will import it into Excel (see Figure 6.14).

Excellent Economics and Business programmes at:



**university of
 groningen**





**“The perfect start
 of a successful,
 international career.”**

www.rug.nl/feb/education

CLICK HERE
 to discover why both socially
 and academically the University
 of Groningen is one of the best
 places for a student to be



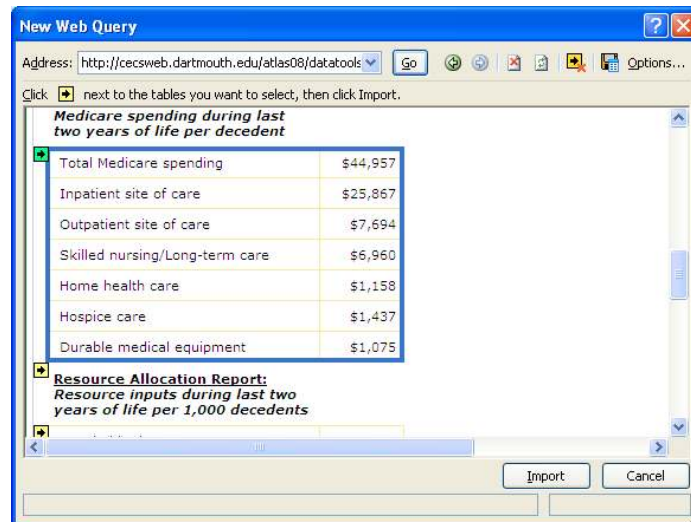


Figure 6.13 Web Query in MS-Excel

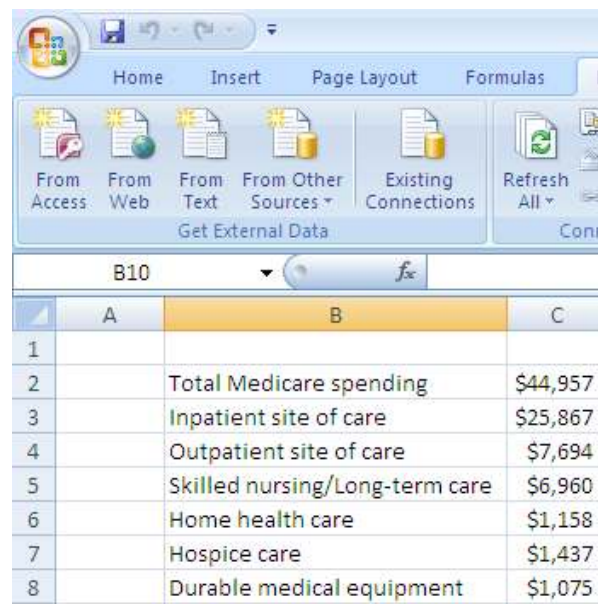


Figure 6.14 Data from a Web Query in MS-Excel

Once the data has been transported into the spreadsheet, any of the data manipulations offered by the software may be performed. For instance, Excel provides a wide variety of statistical analysis tools. Figure 6.15 looks at the data analysis menu showing some of the selections.

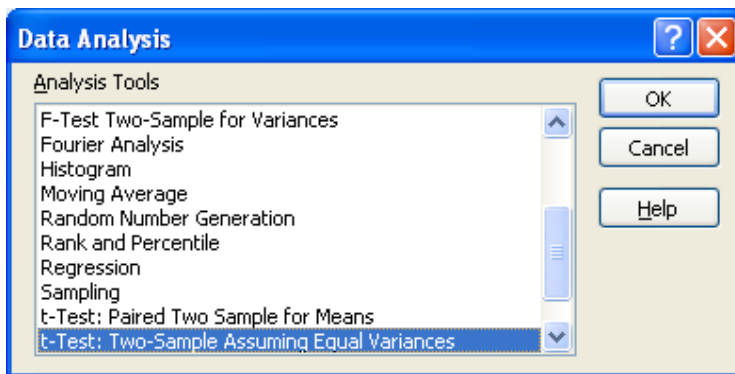


Figure 6.15 Data Analysis Menu in MS-Excel

6.5.2 SAS

Another commercially available software package that can be used as a simulation output data post processor is SAS 9.2 Software from SAS. SAS 9.2 is a powerful data analysis environment that permits statistical experiments to be conducted on sets of data. Fully capable of performing traditional statistical tests and the nonparametric tests often required in simulation analysis, SAS can take most simulation output beyond what is possible with the simulation software packages. Figure 6.16 provides a look at SAS running an analysis.

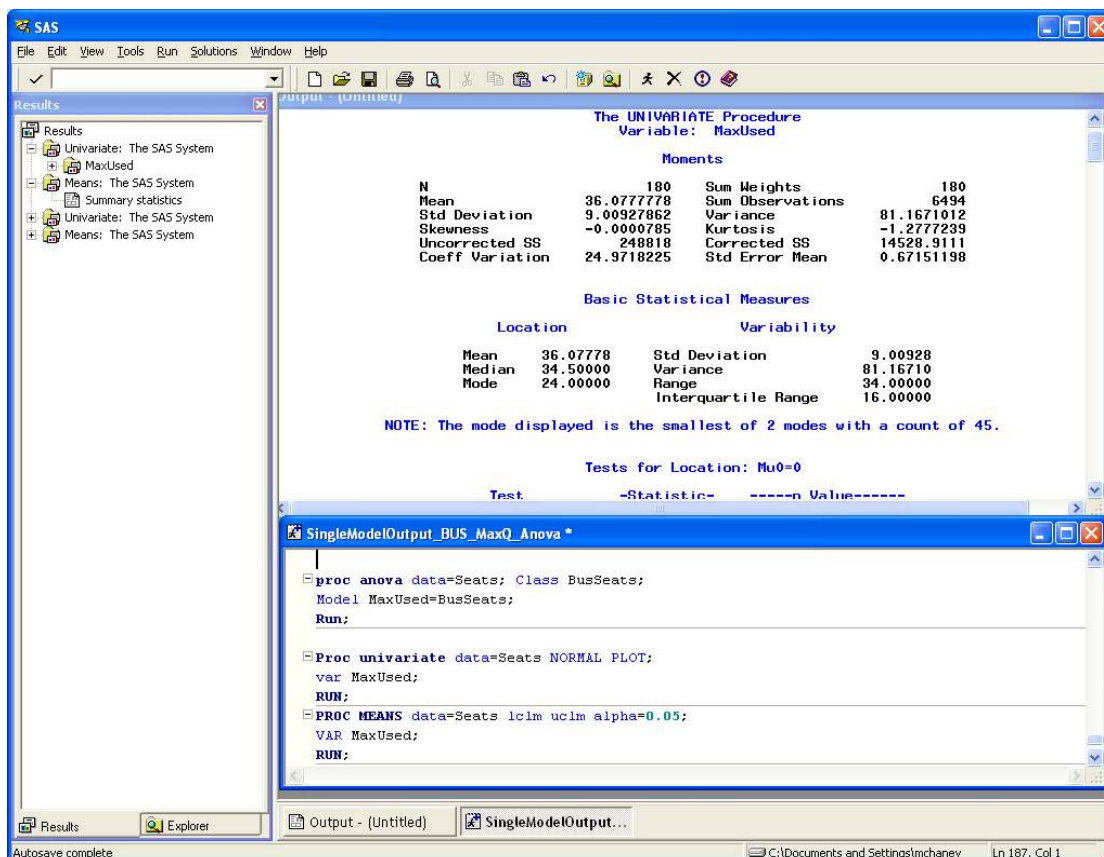


Figure 6.16 SAS Program and Output

6.5.3 Custom Generated Animation

Using animation software with computer simulation is discussed in Chapter Two. Most simulation products include animation software or third party software (such as Proof), are available to take full advantage of current graphics technology and produce exciting high resolution animation to accompany simulation program output.

6.6 Operations, Maintenance and Archival Phase

This phase involves storing the model with consideration of its potential future use. Some models are not stored but rather are used in decision support. For instance, a model may be used for staffing questions and for daily use to determine the expected impact of changing conditions. In other cases, the model's current need is fulfilled and it should be stored, maintained, and the project team disbanded.

Another common activity at this point in time is conversion. This may not apply to all situations but it may involve turning the model over to the customer or end-user. These end-users may adopt the simulation as a tool requiring little or no additional intervention from the simulation team. The storage component of the archive phase reflects the investment in the simulation project. Documentation and physical simulation code, as well as data and other records from the project should be stored in a secure, safe place for future access. Simulations are becoming important repositories for corporate knowledge and can provide important inputs to future organizational decision making.



LIGS University
based in Hawaii, USA

is currently enrolling in the
Interactive Online **BBA, MBA, MSc,**
DBA and PhD programs:

- ▶ enroll **by October 31st, 2014** and
- ▶ **save up to 11%** on the tuition!
- ▶ pay in 10 installments / 2 years
- ▶ Interactive Online education
- ▶ visit www.ligsuniversity.com to find out more!

Note: LIGS University is not accredited by any nationally recognized accrediting agency listed by the US Secretary of Education. More info [here](#).



The third component in the archive phase is maintenance. Here the simulation is monitored and adjusted to reflect environmental changes or newly available organizational information. Often, a member of the simulation project team will be appointed as the contact person for maintenance issues.

The final component in the archive phase is the dissolution of the project team. When the full team is no longer warranted, it is dissolved and the team's members freed for other duties.

6.7 Bibliography

J.M. Bloodgood and R.W. McHaney, "DSS-Supported Knowledge Acquisition and Transfer: An Exploration," *Journal of Information and Knowledge Management*. Vol. 2, No. 3, 1–10 (2003).

C.W. Emory, *Business Research Methods*, Third Edition. Richard D. Irwin, Inc., Homewood, Illinois (1985).

J.O. Henriksen and R.C. Crain, *GPSS/H User's Manual*, Wolverine Software Corporation, Annandale, Virginia (1983).

F.N. Kerlinger, *Foundations of Behavioral Research*, Harcourt, Brace, Jovanovich College Publishers, Fort Worth, Texas (1986).

J. Kleijnen, *Statistical Tools for Simulation Practitioners*. Marcel Drekker, New York (1987).

A.M. Law and W.D. Kelton, *Simulation Modeling and Analysis*, 3rd Edition, McGraw-Hill Book Company (2000).

R.W. McHaney, *Computer Simulation: A Practical Perspective*, Academic Press, San Diego (1991).

R.W. McHaney, "Reusing Simulation Logic in System Development Projects," *Progress in Simulation*, Volume I (George Zobrist and James V. Leonard, editors). ABLEX Publishing Corp. Norwood, New Jersey, 159–185 (1991).

R.W. McHaney, *Simulation*, In Miriam Drake (Ed.), *Encyclopedia of Library and Information Science*, Second Edition, Marcel Dekker, New York, 2643–2655 (2003).

R.W. McHaney, "Success Surrogates in Representational Decision Support Systems," *Encyclopedia of Information Science and Technology*, Vol 1 (Mehdi Khosrow-Pour, ed.), Idea Group Publishing, Hershey, PA, 2672–2677 (2005).

R.W. McHaney and D. White, Discrete Event Simulation Software Selection: An Empirical Framework” Simulation & Gaming, 29(2) 228–250 (1998).

R.W. McHaney, D. White and G. Heilman, “Simulation Project Success and Failure: Survey Findings,” Simulation & Gaming, 33(1), 49–66 (2002).

J. McLeod, Computer Modeling and Simulation: Principles of Good Practice 10(2). Simulation Councils, Inc., La Jolla, California (1982).

M. Page-Jones, The Practical Guide to Structured Systems Design, Yourdon Press, New York (1980).

E.J. Pedhazur and L.P. Schmelkin, Measurement, Design, and Analysis: An Integrated Approach, Erlbaum, Hillsdale, NJ (1991).

S. Robinson, R.E. Nance, R.J. Paul, M. Pidd and S.J.E. Taylor, “Simulation Model Reuse: Definitions, Benefits and Obstacles,” Simulation Modelling Practice and Theory, Elsevier, 12, 479–494 (2004).

D. White and R.W. McHaney, Simulation Steps, Working Paper (2009).



.....Alcatel-Lucent 

www.alcatel-lucent.com/careers

What if you could build your future and create the future?

One generation's transformation is the next's status quo. In the near future, people may soon think it's strange that devices ever had to be "plugged in." To obtain that status, there needs to be "The Shift".

