Proceedings of the 1994 Winter Simulation Conference
ed. J. D. Tew, S. Manivannan, D. A. Sadowski, and A. F. Seila

ISSUES AND REQUIREMENTS FOR BUILDING A GENERIC ANIMATION

D. Michelle Benjamin
Barbara Werner Mazziotti

Textile/Clothing Technology Corporation
211 Gregson Drive
Cary, North Carolina 27511, U.S.A.

F. Bradley Armstrong

Simulation Engineering Associates
7317 Laketree Drive
Raleigh, North Carolina 27615, U.S.A.

ABSTRACT

Since the main function of animation is to serve as a communication tool (Law and McComas 1992), an animation needs to be visually accurate to a user. When presenting to managers, a visually accurate animation can increase the credibility of a simulation model and analysis. Currently it is feasible to build a generic simulation model in a reasonable time period that gives a user the capability of changing the system scope, the characteristics of the whole system or specific objects, and object names. The animation associated with the generic simulation must be able to display all of the changes made by a user in the simulation. This paper presents the issues and requirements for developing generic animations and points out the deficiencies of currently available tools.

Keywords and Phrases

Generic animation, data-driven animation, visually accurate animation, generic model, Focused Applicator Simulator & Trainer (FAST), non-programming data entry.

1 INTRODUCTION

If a specific simulation model is built, it is easy to represent that model with an accurate animation. Typically, a CAD drawing and representative icons are used in the animation to mimic the appearance of the actual facility. However, if a simulation model is a generic model, the animation needs to be generic as well. Generic models and animations would be developed for an audience primarily of trained problem solvers who are not model builders. The user could be a decision maker that does not have the time or expertise to develop models or a student who is using simulation and animation as a training tool. The generic animation must be able to represent any scenario that a user could simulate with the generic model, therefore, it must take into account the ranges for all of the parameters that could define a user's system. Therefore, the animation needs to be data-driven and visually accurate.

If an animation is not representative of a user's system, credibility in the program could be lost or confusion could arise. However, it is very challenging to develop a generic animation that can provide acceptable representations of specific systems. For example, when building a generic animation for manufacturing facilities, the animation must represent *every* manufacturing facility that could be simulated by the generic model. A manufacturing facility could be defined by the number of stations, the number and type of machines per station, the number and classification of people (labor), and the type and sequence of parts flowing through the facility. Therefore, each facility could have a different appearance.

A Focused Application Simulator and Trainer (FAST) program (Armstrong et al. 1994) will allow a user to quickly analyze his/her system and answer various "What if" questions. The components of a FAST program which make this feasible are a data entry environment, a generic simulation model, a generic (interactive) animation, and a customized output report. A FAST program will allow a user to define the scenario he/she would like to analyze, run the simulation, view an animation that is automatically configured, and review the output results. To start the analysis, users will define their systems in the non-programming data entry environment of the program. Then they can run the simulation and animation to quickly analyze and compare the results generated for multiple scenarios of their systems. For example, they can quickly change various parameters such as, the type of material handling system, the manufacturing process (push vs. pull), or an operator's task assignment (Mazziotti and Armstrong 1994) to determine an alternative to meet their objectives.

The animation provides a training tool for a user. The user can watch the animation to gain understanding about the dynamics of his/her system. An animation that allows runtime interactions gives the user the capability to make changes to the system while the simulation is running. The interaction in the animation will immediately show the user the consequences of the change to the system. This will allow the user to experiment with different techniques that can be used in the system. For example, the user may have the objective of minimizing work-in-process. The interaction could allow the user to move operators or parts to a different station to decrease the work-in-process at a station. This could train the user on how to manage work-in-process. In order to allow the user to change data in the data entry environment and in an interactive animation, the simulation model and animation must be data-driven. The data-driven model and animation must be able to read the data at initialization to set up the system and update parameters whenever the user interacts with the program.

In order to accommodate the interaction feature of a FAST model, there are restrictions on the animation packages that can be used. The animation needs to be interrupted while the simulation is running for the user to change the data. Post-processed animation packages can not be interrupted to change the system conditions. The animation packages discussed in this paper do not include post-processed animation packages because they could not provide the capability of user interaction.

This paper will address the requirements and issues of developing the generic animation required in FAST. Section 2 presents the requirements of the simulation model and the corresponding requirements of the animation. Section 3 presents some issues in building a generic animation.

2 REQUIREMENTS FOR BUILDING A GENERIC ANIMATION

When building a generic model, issues and requirements must be addressed before and concurrent to the model development. The same is true with building a generic animation. Below is a summary of the requirements for a generic model and the corresponding requirement for the generic animation.

Table 1: Generic Model and Animation Requirements

Generic Simulation	Generic Animation
Change system scope - # machines, # operators	Only the defined objects will appear
Characteristics of whole system - pull vs. push, material handling system	Visual indicator of option selected
Change logic &/or classification of objects - machine or part type, operator class	Icons will display the requested type and/or classification
Names & labels, current system status variables	User-defined names and labels appear at appropriate location, variables for defined parameters appear
Adjust statistics report to the scope of user-defined system	Display statistics to the scope of the user-defined system

2.1 Change System Scope

Simulation is commonly used to determine machine and manpower requirements. An initial setup is sometimes calculated with a spreadsheet then analyzed with simulation. The randomness of the system in a simulation could demonstrate the need for more or less manpower or machines. If the machine and operator utilization were low, an alternative to increase the utilization is to add another part type to the production line. When a user increases/decreases the number of machines, operators or parts, an animation needs to display the appropriate increase/decrease in objects. In some animation packages, the number of machines displayed is based on the machine capacity. The machines are placed at a predefined location and if the capacity is greater than zero, the machine icon is displayed. The number of operators and parts displayed can be based on entity identification--an entity is created for each operator and part defined. For most animation packages the number of resources and entities is data-driven.

The resource icons are placed at the development stage of an animation based on the scope of the generic model. Therefore, the scope of a generic model could be artificially defined based on the display capabilities of the animation package. Since an animation needs to be visually accurate, the animation needs to display all of the machines that could be defined in a user's system. Flexibility in displaying combinations of machines per station is lost with the predefined resource locations. A user could be analyzing three machine at ten stations vs. ten machines at three station and the animation may or may not indicate this change upon initialization.

Having the machines (resources) placed at predefined locations prevents the visual accuracy for the relative distance between machines. The distance between machines determines how far the part or operator will travel when moving between machines. The operators and parts (defined as entities) will move faster or slower depending on the relative distance between the machines. For the animation to be visually accurate, the machines need to be located at relative distances from each other based on the user-defined distances. Currently, in some animation packages, the relative location of the resource symbols can not be data-driven in an animation.

2.2 Change Characteristics of the Whole System

Simulation is also used to test different manufacturing processes and material handling systems. A FAST user can define the manufacturing process or material handling system to be tested in the data input. When a user changes a characteristic of the whole system, a visual indicator needs to appear in the animation displaying the new characteristic. For example, if the material handling was changed from carts to a conveyor system, to be visually accurate, the icon representing a cart would disappear and the parts would move on an object representing a conveyor system.

Different layout configurations could be analyzed to determine which provides the best process flow. A different layout could be required depending upon the manufacturing process or material handling system implemented. For example, a conveyor system may need a circular layout whereas the carts may need an L-shaped layout. Currently, a user would not be given the option to change the layout configuration in the data entry for an animation that could not display the various layouts. As stated earlier, the placement of the machine icons is determined at the development stage of an animation based on the capability of a generic model.

2.3 Change Logic and\or Classification of Objects

Some animation packages can display multiple parts flowing through the system with different routings, the requested amount of people at their requested location, and the requested number of machines. When entering data, a user could specify (from a predefined library of icons) which icon to display for each part type (see Figure 1). In an animation, the different icons will flow through the production line according to their routing. An animation will also help a user identify errors in data entry of the routing and sometimes processing times (for example, if a user put the decimal in the wrong place for the processing time, the part will be processed slower or faster than expected at that machine).

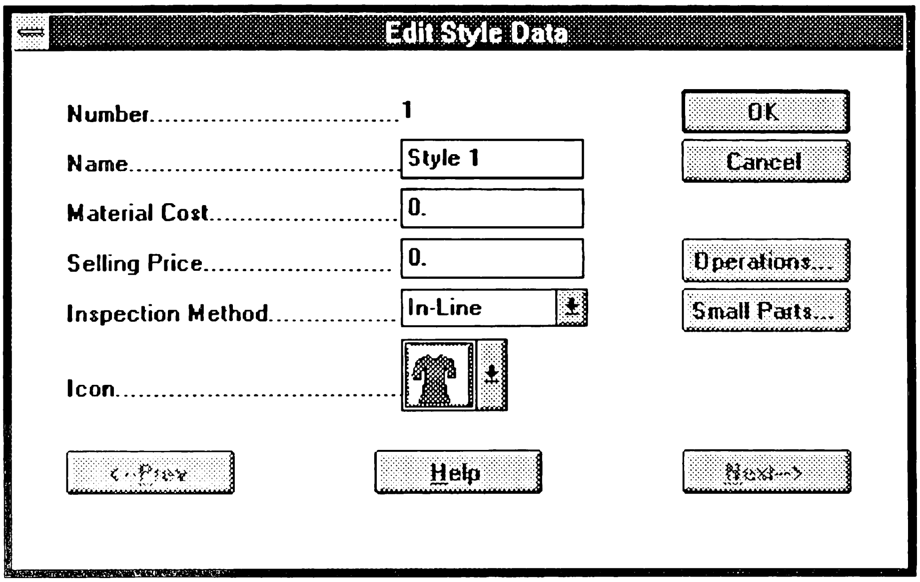

Figure 1: Selection of Part Icon

Typically a manufacturing facility will have various machine types required for the production of various part types. When entering data, a user can specify the machine types required. However, in some animation packages an animation can not display the appropriate symbol. For example, in the apparel industry, the last operation in the production line usually is fold and inspect. An animation should display a table for this operation. If an animation package does not provide the capability of displaying user-defined machine symbols, the symbol displayed would be a sewing machine icon that was a predefined default resource symbol. Different machine icons could be placed at each station, but this would not solve the problem for production lines with the same machine type at different stations. If a user is not given the capability to select a resource symbol, a generic machine symbol has to be picked carefully because it must represent every machine type that could be found in a user's manufacturing facility. In SIMAN/CINEMA V, defining the machine as a resource eliminates the user's ability to choose the machine symbol or location at initialization.

Labor is typically required to operate or load the machines in the production of the parts. Different skill levels or classification of labor can be defined. The skill level or classification determines where the operator is capable of working. In an animation, the different skill levels can be displayed by using different colored labor symbols. However, if the operators have unique training, the operator icons will need to have an identification. This identification could be the operator's name or a number. With some of the current animation packages, the operator's identification has to be a number which is hard coded on the labor icon and a user must remember each operator's identification. Classification or skill level for labor can be data-driven and visually accurate, but unique training is not visually accurate. (The first operator icon could be a female while the first user-defined operator could be a male--definitely not visually accurate!)

2.4 Define Names or Labels and Current System Status Variables

If a user were able to pick a labor icon (from a predefined library) according to sex and/or race and a user-defined operator's name was to be displayed above the icon, an animation would be visually accurate for each operator. However, some animation packages do not seem to have the capability of reading and displaying the character variables defined by a user. If the character variables can not be displayed appropriately (the user-defined name or no name if not defined), an animation could become cluttered with extra labels for most scenarios. Labels would have to be hard coded in an animation. For example, a FAST program could be developed for a set of assembly processes. The differences in the assembly processes include the number of operations, the processing times, and the number of required machines available per station. Some of the assemblies will require less operations than others. The assemblies requiring less operations may not use all of the stations available in the generic model. Therefore, no symbols would show up at those stations in the animation because no machines were required at those stations for this process. However, if labels or variables were placed at a station that is not required, they would still show up (the variable would have a value of 0), even though that station does not exist in a user's system (see Figure 2). The animation package should allow the various labels and variable to be located at a predefined offset from a user-defined symbol. Therefore, if there were no machines defined at a station, the station labels and variables would not be displayed. This approach of labeling would be data-driven and visually accurate. Using character variables would also eliminate the need for cross-referencing. Currently a user would have to print out the data input to see what Station 1 and Operator 1 represent in the real system to understand the dynamics in an animation. If the station and operator name were data-driven, there would be no confusion or uncertainty.

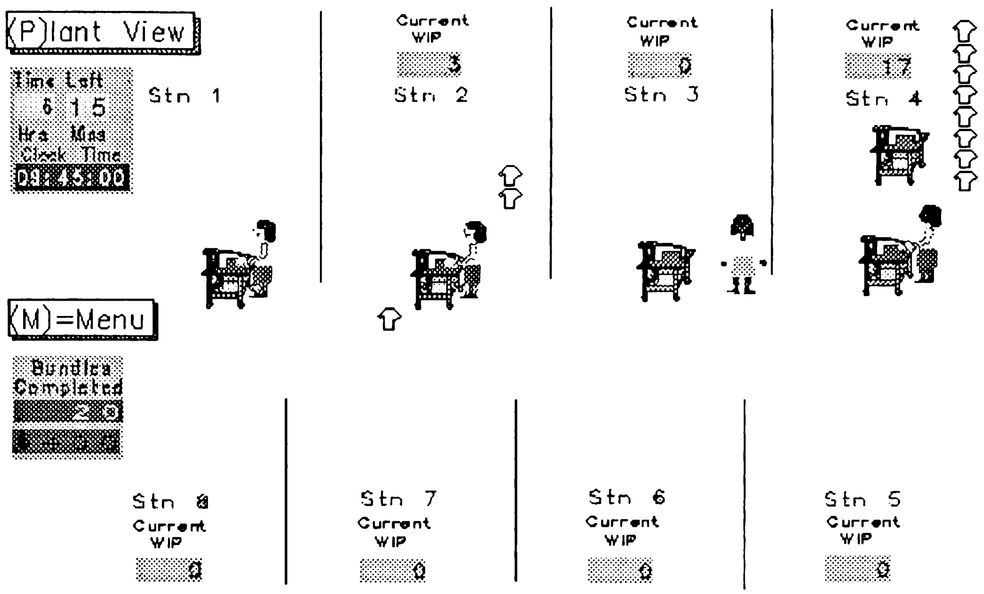

Figure 2: Names and Labels

2.5 Adjust Statistic Reports to the Scope of the User Defined System

While running a simulation, statistics are updated informing a user of the current status of the system. The statistics to be displayed are determined at the development stage of an animation. If a chart format is used to display the statistics, the chart size is based on the maximum number of entries. If a scenario being simulated does not require this maximum, values of zero are displayed throughout the chart. The variables and labels are still displayed. An animation should only display the statistics for the scope of a user's system. An output report from a simulation can be customized to print out the statistics for the operators, machines and parts defined by a user. The output report can also print out the appropriate headings (ie., operator name, machine type, part type). Therefore, a user should have the same features in an animation.

3 ISSUES IN BUILDING A GENERIC ANIMATION

Another challenge in building an animation is fitting all of the desired information in the display window. Some animation packages solve this problem by having multiple windows/screens to display the information. Simple++ is object oriented allowing a user to view the desired level of detail in the manufacturing facility animation. Non-object oriented or single window animation packages allow a user to view close-up views of different section of the same layout. The entire layout could be the main view. However, if a user's system was large, a lot of the detail would be lost because the icons, labels and variables would become too small to read. To regain the detail, different windows could display close-up views of the various sections of the manufacturing facility and the statistics. This would allow a user to view the overall dynamics and zoom in on a specific area of interest when needed.

Determining the views for the different windows becomes an issue at the development stage of an animation. The area of interest for each user needs to be displayed in the windows. One user could have a facility with a few stations and many machines per station while another user could have many stations with a few machines per station. Therefore, two windows could be needed to display the same number of machines.

User interaction is an important training tool in FAST. If interaction is used in a FAST, the method of interaction must be determined at the development stage of the FAST. Some methods that could be used include a dialogue box, a text window, customized buttons or a mouse driven technique. If a dialogue box or a fixed size window is used, caution must be taken to not obscure information in the animation that a user would be using to make a decision. Therefore, the best alternative would be a graphical interaction window. This interaction window would have the capability of displaying icons, customized buttons, multiple fonts, and colors. In addition, the interaction could be data driven in displaying the appropriate user-defined names, labels and icons. Important information would not be obstructed because a user could minimize the window, re-evaluate the system, and then maximize the interaction window when changing the data. However, the method used could be limited by the animation package used. Most animation packages do not provide the capability to use all interaction techniques.

4 SUMMARY

Currently the available animation packages reviewed by the authors do not meet the needs of generic animation. For the most part, they do NOT facilitate data-driven, visually accurate animations. A user could customize each specific animation by picking different icons from an icon library, changing the labels, and removing unnecessary variables, but this could be time consuming. In addition, if the icons were changed to the correct representation and the labeling and variables were corrected, this does not guarantee that the icons will be placed in the appropriate location according to the user-defined system. The animation layout could also be changed by the user, but that entails moving all misplaced stations (the resources, seize points, queues, etc.) to a new location which could be intimidating to a non-technical user. Changing the animation would lead to the development of multiple animations to represent the various scenarios that the user would simulate. Having multiple animations is an unacceptable solution and strays from the main point of fast problem solving and training.

A FAST program needs to be built in a reasonable period of time at a reasonable price to an end user. In building a FAST program, the authors encountered difficulties in finding a single simulation/animation package that met all of the requirements. The animation packages reviewed by the authors in varying degrees of detail include SIMSCRIPT/SIMGRAPHICS, SLAM SYSTEM, SIMAN/CINEMA, QUEST, AUTOMOD, ithink, and WITNESS. Please note that an exhaustive research was not completed on all of the packages because the packages were deficient in one or more of the issues and requirements stated in this paper.

REFERENCES

Armstrong, F. B., K. Powell, B. W. Mazziotti, and D. M. Benjamin. 1994. Focused Analysis and Training Environments. *Proceedings of the 1994 Winter Simulation Conference.*

Law, A. M. and M. G. McComas. 1992. How to Select Simulation Software For Manufacturing Application. *Industrial Engineering* Volume 24, No. 7: 29-35.

Mazziotti, B. W., and F. B. Armstrong. 1994. Creating a Focused Application Simulator with Flexible Decision Making Capabilities. *Proceedings of the 1994 Winter Simulation Conference.*

AUTHOR BIOGRAPHIES

D. MICHELLE BENJAMIN is currently a Simulation Analyst at the Textile/Clothing Technology Corporation ($[TC]^2$). During the past year at $[TC]^2$, she has completed consulting projects that address issues in warehousing and production assembly systems. She has helped in the development of flexible, data-driven simulation models and animations. In addition, she has taught seminars about simulation and its use in the textile and apparel industries. She received her B.S. in Industrial Engineering and M.S. in Industrial Engineering and Operations Research from Pennsylvania State University. She is a member of IIE and ASQC. Her interests include user interfaces, generic modeling, and output analysis.

BARBARA WERNER MAZZIOTTI is currently the Manager of Simulation Services at the Textile/Clothing Technology Corporation. For the past 4 years, Mrs. Mazziotti has been challenged with bringing the technology of process simulation to the apparel and textile industries. In this position, she has focused on creating data-driven, flexible simulation models and animations for non-expert users. In addition to conducting seminars on simulation and production systems, she has completed consulting projects with more than 20 retail, apparel, and textile companies. Prior to $[TC]^2$, Barbara was on the consulting staff at Systems Modeling Corporation, teaching SIMAN and CINEMA, managing projects and doing analysis for a variety of industries and applications. Barbara began her career at General Motors as a Simulation Project Engineer for a group of metal stamping and automotive assembly plants. Barbara has a B.S. in Operations Research and Industrial Engineering from Cornell University and is completing an M.S. in Industrial Engineering at North Carolina State University. She is a member of SCS and senior member of IIE.

F. BRADLEY ARMSTRONG is a Fellow Engineer for Asea, Brown, Boveri in the Engineering and Manufacturing Productivity Center where he is an internal consultant for simulation and world class manufacturing methods. He also is the owner of Simulation Engineering Associates (SEA). Prior to forming his own company, he was a Senior Staff Engineer at Hughes Aircraft company where he worked as an internal simulation consultant. He also worked as a Senior Systems Analyst for Pritsker and Associates, and as an Operations Analyst for General Dynamics. He received a B.S. in Mechanical Engineering from the University of Texas at Austin in 1981, an M.S. degree in Industrial Engineering from Purdue University in 1986, and is completing an M.S. degree in Integrated Manufacturing and Systems Engineering. He is a professionally registered engineer in North Carolina, California, and Indiana. He is a member of IIE and SCS. His current interest is in using simulation, statistics, and GUI tools to build Focused Analysis and Training Environments.