JOHN D. BERNARD, MS
D. HENRY PATE, ED.D.

TECHNICAL EDUCATION MODELING AND SIMULATION

TEMS

CLEMSON UNIVERSITY - INDUSTRIAL EDUCATION
CLEMSON, SOUTH CAROLINA

THE INSTRUMENT

Abstract

Initial testing is now complete on the **TEMS** instrument, the Technical Education Modeling and Simulation system. This research in the Industrial Education Department at Clemson University is a three phased project. The phases include developing an instrument (TEMS) similar to DYNAMO II (Pugh, 1970) for the modeling of socio-econ-educational systems, reducing world model concepts to a regional model for the state of South Carolina, and integrating technical education attributes and effects into the classical capital sector for this regional model.

The **TEMS** system is developed in dBASE IV and 'C'. It has all the model definition building features and run characteristics of DYNAMO II. Written for an IBM AT class of equipment, **TEMS** will replicate the WORLD2 model (Forrester, 1971) results in 40 minutes for a 100 year run. **TEMS** supports both the real time graphic mappings of selected variables and post analysis graphics. It has both an integrated statistical interface to SPSS statistics and a reporting system for model runs, definitions, user created functions, and run time statistics.

Experimentation is in progress to calculate a CHAOS mapping for the class of level variable equations. Using this Verhulst equation mapping, **TEMS** should then dampen any wild ramping and explosiveness for these selected variables during the simulation.

Table of Contents

Introduction

The Industrial Education Department at Clemson University is conducting research with technical education and its socio-economic impact. This paper focuses on the first phase of a three phase research project which is investigating the social and economic impact of technical education on aggregate households (people) and on the statewide economy.

The research question which we are investigating is: In the regional socio-economic model for South Carolina there exists a significant relationship between technical education and socio-economic quality of life.

System dynamics modeling and simulation are needed in these circumstances to study the effects of economic changeovers on the quality of life for the individual and of the economic health of the state. States in either growth mode or poor economic conditions need tools to analyze their predicament, to forecast their outcomes, and to help design their plans.

In order to successfully conduct this research we needed both an instrument to describe our model and its intricate interactions, and a regional socio-economic simulator for South Carolina. Statistics alone (time series analysis, ANOVA, ANCOVA . . .) is not sufficient to describe and map complex variable interactions over time.

The major reasons for selecting system dynamics modeling and simulation were because there existed proven instruments of analysis, a bed of literature on the subject, and a track record of research for this non-linear dynamics science. We are not exactly creating data and doing experimental research, but rather, applying meta-analysis to extract known relationships and quantifiable values for these relationships. We then integrate these relationships and produce (hopefully) new research and new insights into the technical education system and it relationship with the industrial community.

The three phases of this research include developing an instrument (TEMS) similar to DYNAMO II (Pugh, 1970) for the modeling of socio-econ-educational systems, reducing World model concepts to a regional model for the state of South Carolina, and integrating technical education attributes and effects into the classical capital sector of this regional model. As stated in **Dynamics of Growth in a Finite World**, "DYNAMO, the computer language most often used to express system dynamics models, is not absolutely essential to the method - other flow diagrams and computer languages could be used." (Meadows et al, 1974) We feel that we have achieved an alternative with **TEMS**.

This paper summarizes our modeling and simulator instrument development. We conclude this work with both the statement and evidence to support that we now have a verifiable tool for model development and research specifically in the area of technical education.

Instrument Specifications

The instrument or tool for the research is very important to us for several reasons. A proper instrument (the computer implementation) provides a pathway for the verification of a lumped model. The results of the instrument must be repeatable, and reproducible using other established instruments.

DYNAMO II (Pugh, 1973) is an excellent continuous simulation language for model building and simulation. We wanted feedback characteristics of a continuous simulation language, the programmatic mechanics of DYNAMO II, the functions found in large scale social systems, and these additional modeling and simulation system characteristics:

- Chaos analysis and interaction with the simulator.
- On-line integration with graphics and statistics for both real-time and post processing.
- Interactive and available on multiple desktops with an ease of use requiring minimal training.
- User interface which would stimulate usage by researchers in education, sociology, and labor economics (not only system dynamics or operations research).
- A system tool customized for general purpose socio-economic modeling.

It is essential that our modeling system deal with disequilibrium and chaotic behavior. As Sterman points out: most economic models are in a state of disequilibrium as a rule. (Sterman, 1986) The science of chaos has advanced to the point that new understandings of chaotic processes often give rise to order. We want our modeling system to help identify and employ dampening techniques to our non-linear social systems in various states of turbulence. We must adapt these processes for the "higher rule of order." (Killian, 1989).

After our initial Smalltalk/V development we realized that real-time interaction rather than post processing interaction was a time saver and gave us a more dynamic thinking process. Real-time graphics with a built-in statistical interface can magnify research productivity and validation.

We decided to develop our own continuous simulation system to meet our requirements for a research tool. We started with a Smalltalk/V (Digitalk Inc., 1988) development, abandoned it, and redeveloped a dBASE IV (Ashton-Tate, 1988) system. The description of the Smalltalk/V system follows.

Development of the SmallTalk/V Prototype

In this phase we had a goal to design and build a dynamic interactive tool to build models and simulate the model running. Object-Oriented Language Systems (OOPS) seemed like an excellent place to start for doing model building and simulation. The inherent concepts that OOPS performed operations and functions in a fashion more closely to the human mind convinced us that this was the design tool for the future.

The basic premise is that objects are created which have attributes, operations (methods), and communicative messages which they may pass to other objects or classes. The human mind is said to cluster similarly. As system dynamics researchers are aware, models resemble networks or webs with nodes linked together, so that a programming language designed around this concept has to be "natural."

As an example the equation to drive the integration of quality of life (qol) through the first forward difference was:

```
| dial |
dial := ((dials at: #qol) reading).
(dials at: #qol) changeReading:
(dial + (dial * capInvToQualOfLife) +
(dial * pollToQualOfLife))!
```

In this code **reading** and **changeReading** are methods found in subclasses created for this application. They read the screen dial and update the screen dial for qol. The values of the individual equation for qol are actually stored in the screen image. The needles are read from the screen image and converted to current values. From a dynamics point of view, this concept added new meaning to the phrase "what you see is what you get."

For a simple experimental prototype we used this simple network relationship to tie some of the basic level variables of WORLD2 (Forrester, 1971) together for testing purposes:

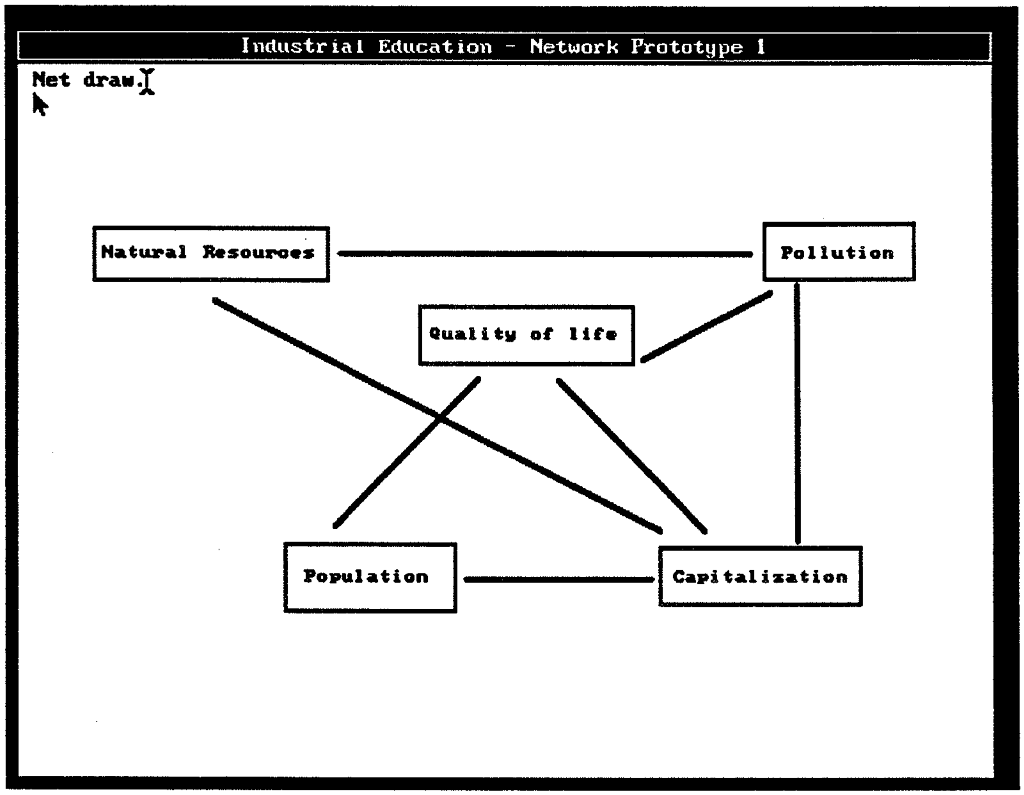

Figure 1

For prototypes we used this network with simple equations to create a real system of data relationships which gave a characterization of a lumped model (a somewhat arbitrary test model which was not validated, only logically linked).

Using a Smalltalk/V sample program developed by Digitalk Inc., we adapted it for our use. The original program was the real-time interaction of someone with the controls and dashboard gauges of an automobile. After modification, if the capital investment rate was a little sluggish then we could do things like step on the accelerator or throw the capital transmission into a lower gear for more torque. If population was growing out of hand, we simply could step on the population brakes and slow it down. If we wanted to play "what-if" games with the pollution gauge, we could just change the pollution needle. The entire model could even be thrown into reverse.

We designed a simple simulator with the prototype model hard wired into Smalltalk/V code so that we could get the feel for the user interface and the speed/space performance of the computing aspects of the system. Our prototype looked like this:

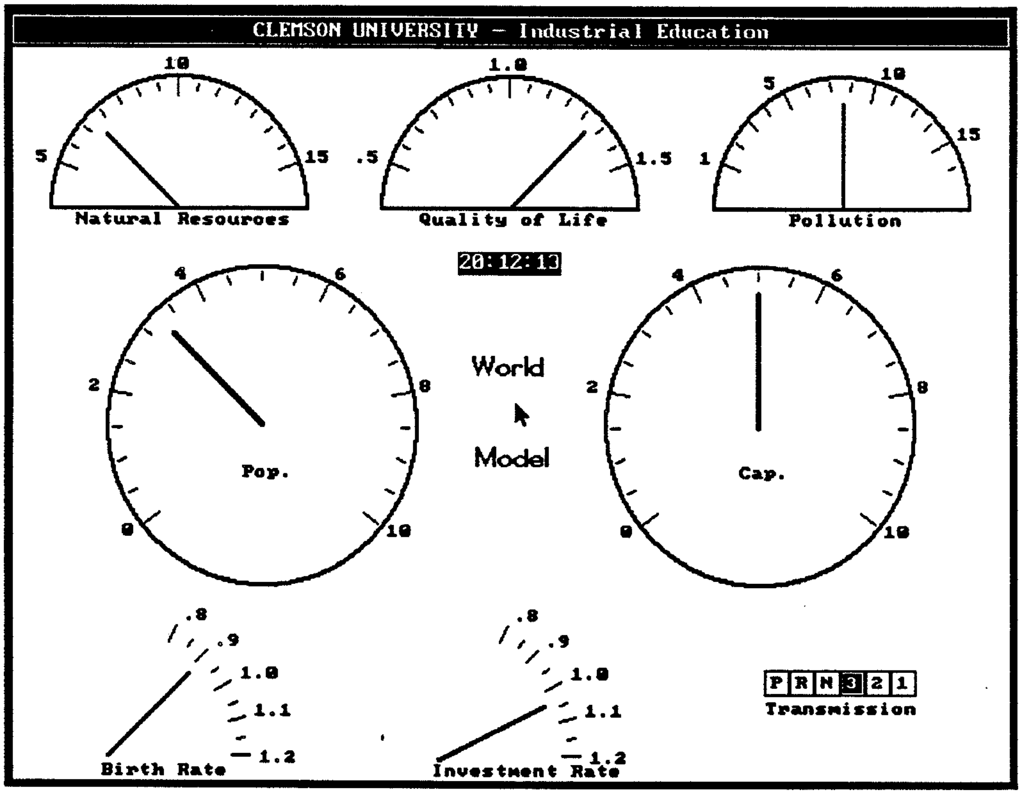

Figure 2

We included warning indicators for when natural resources were too low or pollution was too high. Even here we began to set the conditions for monitoring serious turbulence or chaos. The researcher, in the next figure as an example, could reduce the overload or force the model to continue in its turbulent state, similar to what Ed Lorenz, Peter Scott, and Robert Shaw do with meteorology and fluid models. (WGBH Educational Foundations, 1989)

The out of balance conditions which were hard-wired into the Smalltalk/V simulator would be brought back to their initial state values if the researcher selected "reduce" in the pop-up window, as seen in the next figure. Alternatively, one could select "continue" and then pick up the needle on the screen and place it at any value.

The next logical development would have been to analyze the change in variable levels or rates rather than using critical set points for chaos. We incorporated this in the dBASE IV simulator, described later.

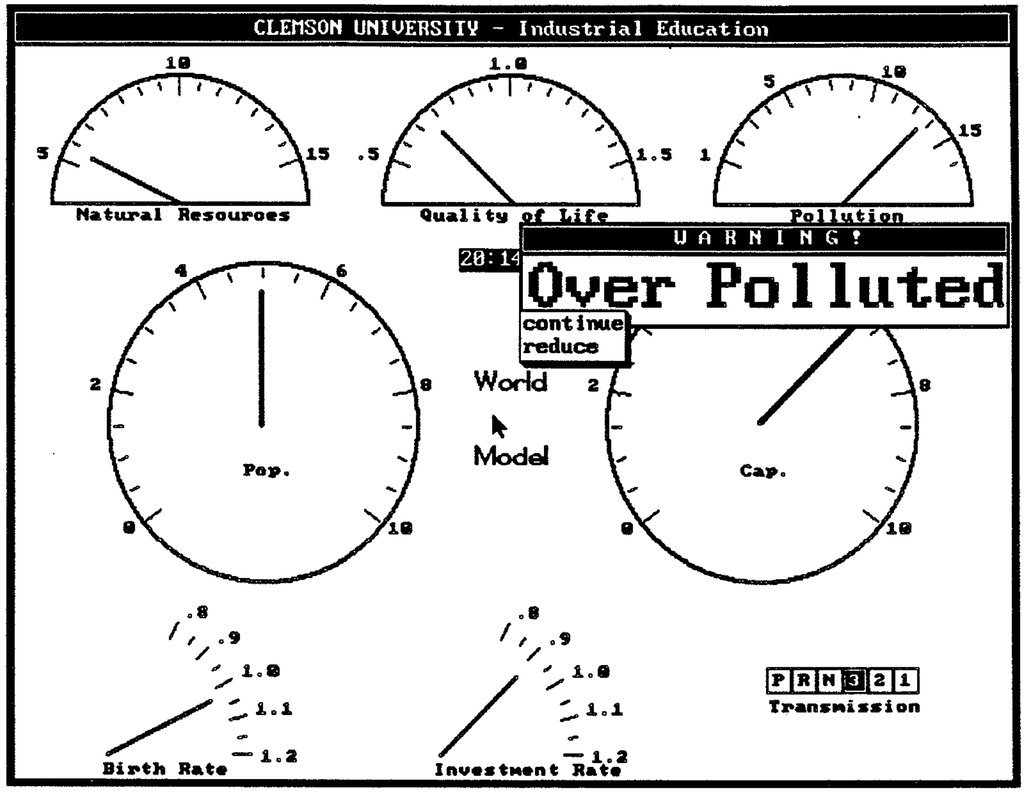

Figure 3

SmallTalk/V Summary Conclusions

We abandoned this prototype for three primary reasons.

The small simulator ran very slowly, and as time went on its computation for each year (with DT = 1) increased 20 percent. The model would compute Year 1 in five seconds and Year 7 was taking 20 seconds. Even after adding floating point capability and optimizing the space/speed options the simulation speed did not improve very much.

Smalltalk/V is a more complex programming language than most programmers care to admit. This language comes with over two thousand methods and one hundred classes. Thirty five pages of cross referencing these methods and classes only compounds Digitalk's observation that " . . . interim versions of methods are an excellent way to 'divide and conquer' . . ." and that programming tips are "survival tips." (Digitalk Inc., 1988).

Finally, the language may be too powerful. The concept of networking helps us understand the complex, but the logic may not be intuitive. Each time we made a simple change to the model, like stepping on the capital investment accelerator, could be a complete dissertation within itself. Simulators with this kind of flexibility may only cause outside critics of this process to say that the simulator is only an out-of-control results producer.

These reasons should not be conceived, however, as criticisms of Smalltalk/V because in large research projects with many researchers and with INTEL 386/486 microcomputers with megabytes of memory, this flexible language could be an excellent research and development tool.

Development of the dBASE IV/C TEMS System

We settled on developing our second prototype in dBASE IV and 'C' for several reasons. These software languages had important ancillary support packages such as graphics and statistics. When the computations were intense (floating point or graphic computations) the load and call to an external program was there to expedite processing. These software products both have automatic application generators which significantly reduced instrument development time. Finally, the simulation produces voluminous data which needs to be cataloged, managed, and evaluated. A data management system with a built in procedural programming language was very desirable.

The prototype network illustrated in Figure 1 was again used for initial testing, and after our successful testing, we then built WORLD2 and simulated it producing statistically identical results (described later).

TEMS System Description

Our simulator is classically constructed: the software evaluates each node of a linked set of equations and pulls in each input value from either other nodes, constants, or tables. The equations are evaluated in such an order so as to avoid simultaneous equation solving. The clock is then incremented and the process repeated. (Aburdene, 1988)

To support the simulator we have a module to build the equations and to set their initial values. Another module establishes the criteria for the individual run (times, DT, etc.). Because these modules integrate with a data management system, the researcher can conveniently track different runs, models, and results. Secondly, an integrated interface to both a graphics and a statistics subsystem (SPSS) permit the results to be analyzed immediately without any intermediate steps.

Our system is interactive with selection windows that move the researcher from major screen to minor screen. By simply using the arrow keys on the keyboard, a selection could be highlighted and then picked by pressing the enter key. The Control Menu carries the researcher to the other master menus for building the model, setting up the runs, actually running the model, and reviewing the results. In the review of the results, bar chart and line graphics may be selected for four variables at a time to be displayed on the screen, printer, or plotter. Also from this submenu the statistics package is automatically set on the stats database. The researcher may start selecting data samples, the type of statistics, and the output specifications.

This figure gives an overall insight into our model builder and simulator:

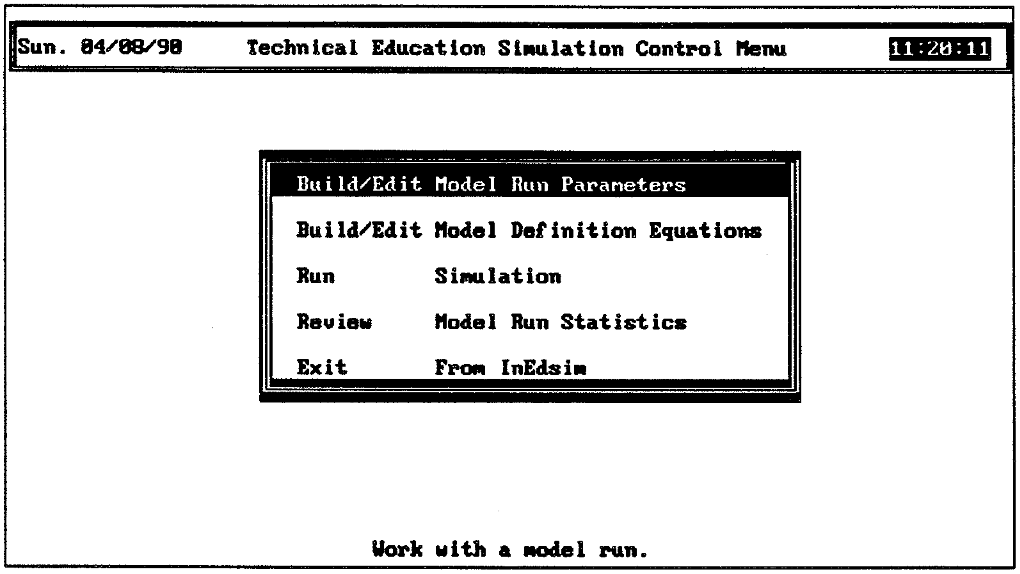

Figure 4

Here one can see the Control Menu which permits the user to select one of five activities: building the model, setting up a run of the model, running the model, reviewing the results of the run, and exiting the system.

The next submenu for building the model and setting up the run for the model are similar because they permit the researcher to add new elements, change elements, browse through all the elements, reorganize the elements, and exit the submenu.

The Definition submenu, Figure 5, seen on the next page permits the researcher to develop a set of equations, similar to DYNAMO II, to describe the model. Because this is a specialized simulator, most of the mathematical equations typically found in a social simulator are predefined as functions. The researcher only has to name the equation (also specifying an abbreviation) and give it a relative number and equation type: level, auxiliary, or rate. Further, the other nodes of the model which input to the current node are specified as well as any constants or table values to feed the predefined mathematical equations. After the model has been defined then **TEMS** sorts the equations to ferret out simultaneous equation relationships and reassigns the relative equation numbers.

Other functions may be built by the researcher to define relationships which are not routinely provided in the simulator. If the equation under development is a level equation then the

Chaos Function may be set to "Y" to indicate to the simulator that the results of this equation should be monitored for dampening in the event that this level is in a wild collapse or ramping mode.

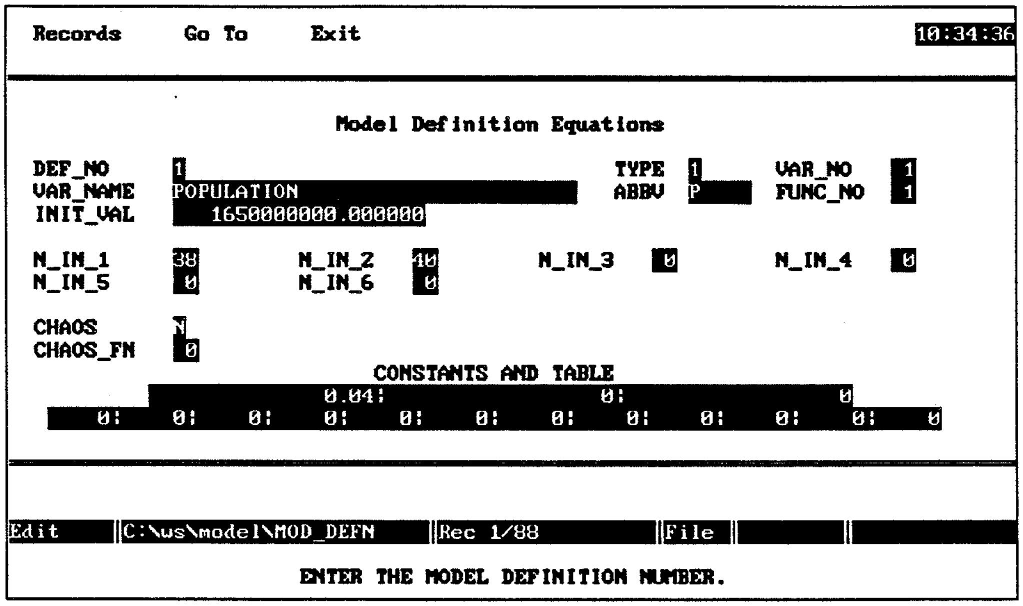

Figure 5

The model building and run options have a common attribute: Verhulst equation tracking. We have incorporated an experimental mechanism to control and suppress model collapse modes.

As seen in the Model Definition and Run submenus (Figures 5 and 6), Chaos can be selected at both menus. When the model is running and Chaos = 'Y' for both the run and the specific equation then the simulator checks the five previous values to see if there exists a rapid increase or decrease in the change of the level, and if the values are moving away from equilibrium.

The Verhulst equation:

$$Y_{t+1} = Y_t(1 + R) - RY_t^2,$$

where Y is the level variable under consideration and R is the rate of change. The first portion of the right hand side of the equation is a classic first forward difference, but the second portion of the equation, $- RY_t^2$, dampens large changes.

On any level equation (population as an example) created, the researcher may elect to automatically have the calculated values for that variable tracked against the known equilibrium paths of the Verhulst equation.

The Run submenu on the next page, Figure 6, permits the researcher to specify which model to run, the purpose of the run, and comments about the run. The beginning and ending times are chosen, as well as the DT interval, the date, time, and the type of graphics (line or bar charts) dynamically displayed during the run. Four variables can be graphically monitored.

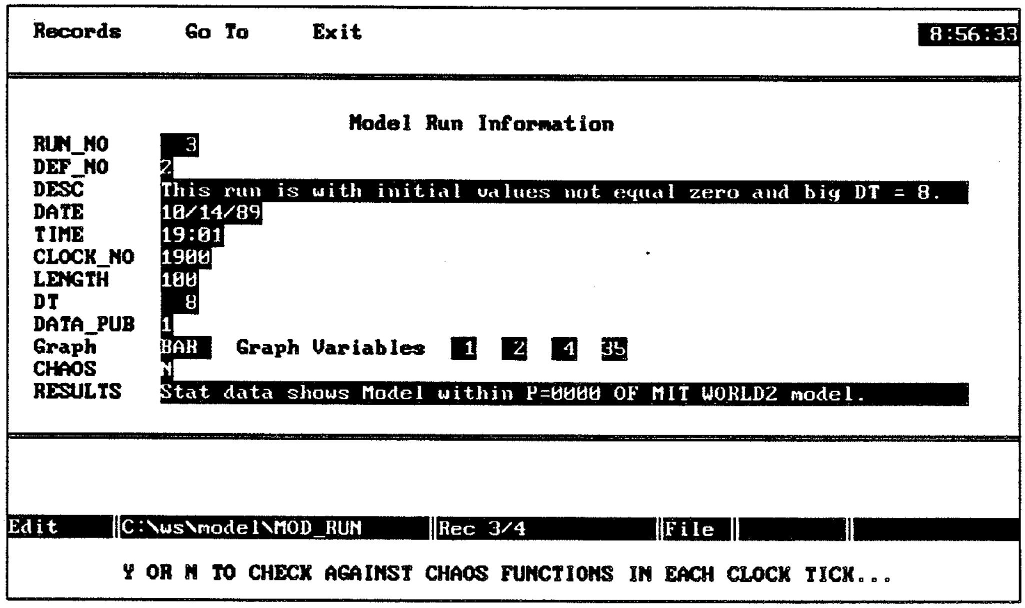

Figure 6

We can see from the next figure the Verhulst equation plotted without transients paths removed.

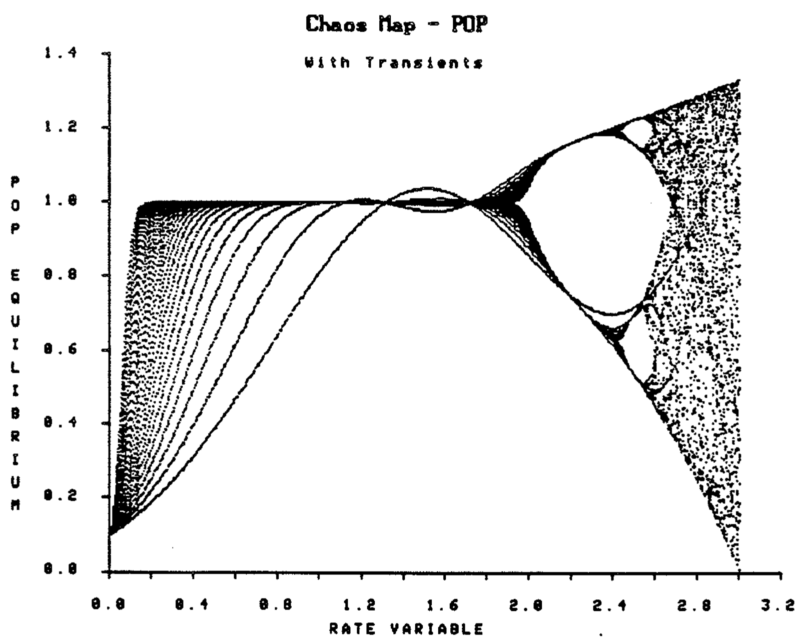

Figure 7

Transients are defined as an arithmetic series of points which are plotted at every R. At any R one can have an infinite number of points which represent the current value of the population. The population Y in this plot goes from .1 to a stable equilibrium value of 1. As seen on the previous figure some points may actually map above 1. (Dewdney, 1987)

To paraphrase Pierre-Francois Verhulst, an eighteenth century Belgian researcher in population growth, "An absolute shortage of food when there are lots of rabbits enables different control mechanisms in nature than when there are just a few rabbits." It is interesting to note from the transient equation graph (Figure 7) that along several transient paths both R decline and R incline can produce an extinction of a species.

The equation further describes the effect of a large population on equilibrium: the brakes are applied more significantly at higher population levels. As seen in Figure 8, an R less than 2.0 and a small initial population evolves to a stable population of 1. What follows are oscillation paths (attractors) where a population may map into an equilibrium path.

Figure 8 shows the plotting of the equation with the first 6000 transients removed. Now we can clearly see the equilibrium path and the attractors (alternate equilibrium paths). These temporary equilibrium values, have been logged into an internal table in the simulator. The sea of chaos is evident at R > 2.6, and once the level variable collapses in this area, there presently is no recovery in TEMS.

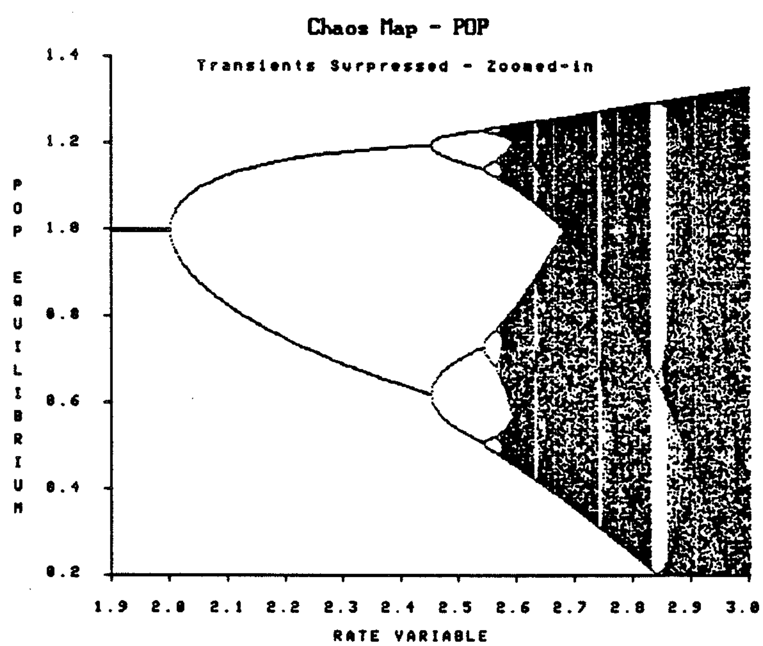

Figure 8

Again, if there has been a sudden change, the simulator looks for the nearest equilibrium or attractor point in the Verhulst table and forces the new current value in that direction. The current value is "dampened" or "clipped" to emulate exogenous control found in physical and social phenomena.

We have a screen display of the simulator while running WORLD2. Back in the Run submenu, Figure 6, there is a factor called the Data Publish factor which permits one to specify the interval that the results are to be put in the stats database. When they are published, the screen graphics update to reflect the last five data values for the four selected variables.

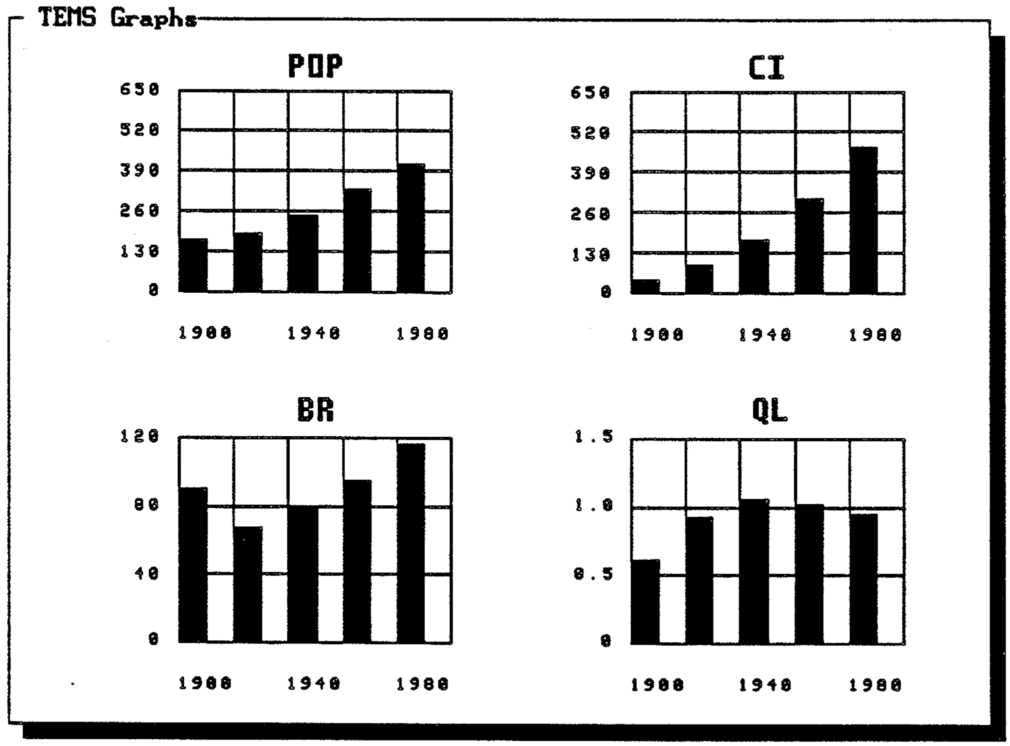

Figure 9

Finally in addition to the actual running of the model there is the review of the results by examining individual variables or by browsing all variables or by printing the entire set of run data or by graphing it. The researcher may look at four different variables at a time on the screen or pass them all to the printer or plotter.

TEMS Results and Verification

Statistically our simulator produces repeatable and reproducible results when compared to another verifiable simulator using a validated model, WORLD2. Additionally, we have conducted the necessary correctness proofs for the program modules, reprogrammed critical modules for a comparison of results, and traced input-output relationships as suggested in the literature. (Sargent, 1984)

During our WORLD2 simulation run, using **TEMS** we produced a data set of results which we used to compare to the DYNAMO II WORLD2 results. A matching of the two data sets revealed:

Sun. 04/08/90 **Review of Model Run Statistics** 11:49:16

Paired samples t-test: INEDCIAF WLD2CIAF

Variable	Number of Cases	Mean	Standard Deviation	Standard Error
INEDCIAF	11	.2502	.043	.013
WLD2CIAF	11	.2500	.043	.013

(Difference) Mean	Standard Deviation	Standard Error	Corr.	2-Tail Prob.	t Value	Degrees of Freedom	2-Tail Prob.
.0002	.000	.000	1.000	.000	1.49	10	.167

Figure 10

The five levels and the QL auxiliary values from WORLD2 were t tested using pairing of the values in DYNAMO II and **TEMS**. As seen in Figure 11 above, a high correlation value indicated a near perfect degree of association and the t value was 1.49. All equation variables were equally matched between the two data sets. We can accept the null hypothesis that there exists no significant difference between the values of each data set. (Best, 1988)

The very small differences were attributed to differences in the floating point chip and floating point routines of the two different computers. We are able to maintain an 18 significant digit precision with dBASE IV so we did not attribute any slight differences to numeric precision.

Authors' Note

Program listings of the Smalltalk/V prototype described in this paper will be made available at the conference for those who wish to have a copy for further development.

References

1. Aburdene, Maurice F. **Computer Simulation of Dynamic Systems.** Dubuque, Iowa: Wm. C. Brown Publishers, 1988.

2. Ashton-Tate Corporation. **dBASE IV Reference Manual.** Torrance, California, 1988.

3. Best, John W. and Kahn, James K. **Research in Education.** Englewood Cliffs: Prentice-Hall, Sixth Edition, 1988.

4. Bronson, Richard. "Computer Simulation: What it is and how it is done." Byte Magazine, (March, 1984): 95-102.

5. Dewdney, A. K. "Probing the strange attractions of chaos." Scientific American 257: 108-111, July, 1987.

6. Digitalk Inc. **Smalltalk/V Programmers Handbook.** Los Angeles, May, 1988.

7. Engle, Charles B. "Chaos on computers." Science (D 1, 1986): 246:1172.

8. Forrester, Jay W. **World Dynamics.** Cambridge: Wright-Allen Press, Inc., 1971.

9. Killian, Anita M. "Playing Dice with the Solar System." (an MIT treatise on chaos), Sky & Telescope, Vol. 78, No. 2, Belmont, Massachusetts: Sky Publishing Corporation, August, 1989.

10. Meadows, Dennis. L. et al. **Dynamics of Growth in a Finite World.** Cambridge: Wright-Allen Press, Inc., 1974.

11. Pugh, Alexander L. **DYNAMO II User's Manual.** Cambridge: The MIT Press, Fourth Edition, 1973.

12. Sargent, Robert G. "Verification and Validation of Simulation Models." 1984 Winter Simulation Conference Proceedings (1984): 115-121.

13. Sterman, John D. "The Economic Long Wave: Theory and Evidence." Systems Dynamics Review Vol. 2, No. 2: 88 - 125, Summer, 1986.

14. WGBH Educational Foundation. "The Strange New Science of Chaos." #1603, New York: Journal Graphics Inc., 1989.