Simulation of Large Scale Software Systems: Implementation, Methodology and the Utilization of APL

By:

Miklos Antal Vasarhelyi

and

John Clark Fellingham

Address:

Accounting-Information Systems Research Program
Graduate School of Management
University of California--Los Angeles

ABSTRACT

This paper describes the implementation features of a system proposed in the 1971 Winter Simulation Conference.* It also describes the experimental design and implementation of the "Online Planning Experiments" which utilized the above mentioned system to examine the behavioral features of man-machine interaction.

Emphasis is given to the discussion of APL as a simulation language. Issues around the tradeoffs between online and batch data processing are discussed and followed by a discussion of the positive and negative features of APL as a simulation language. The paper is concluded with some normative statements concerning when APL should and should not be used in simulation.

*Simulation: A tool for design and pre-implementation testing of large scale software systems, Proceedings of the 1971 Winter Simulation Conference, pp:261-267

I--Introduction

In this paper the implementation features of a system methodology proposed in the 1971 Winter Simulation Conference (8) are discussed.

This methodology proposed the utilization of surrogate software systems to simulate the main features of a large scale software system. This simulation would allow improved scheduling, cost estimates and would give an early warning of possible developmental problems. Also the utilization of surrogate software systems would allow users to more fully participate in the software development process. Figure I describes a proposed organization-wide interactive planning system of which the surrogate simulator is the central topic of this paper. In this section the main features of the paper are introduced and related to earlier work.

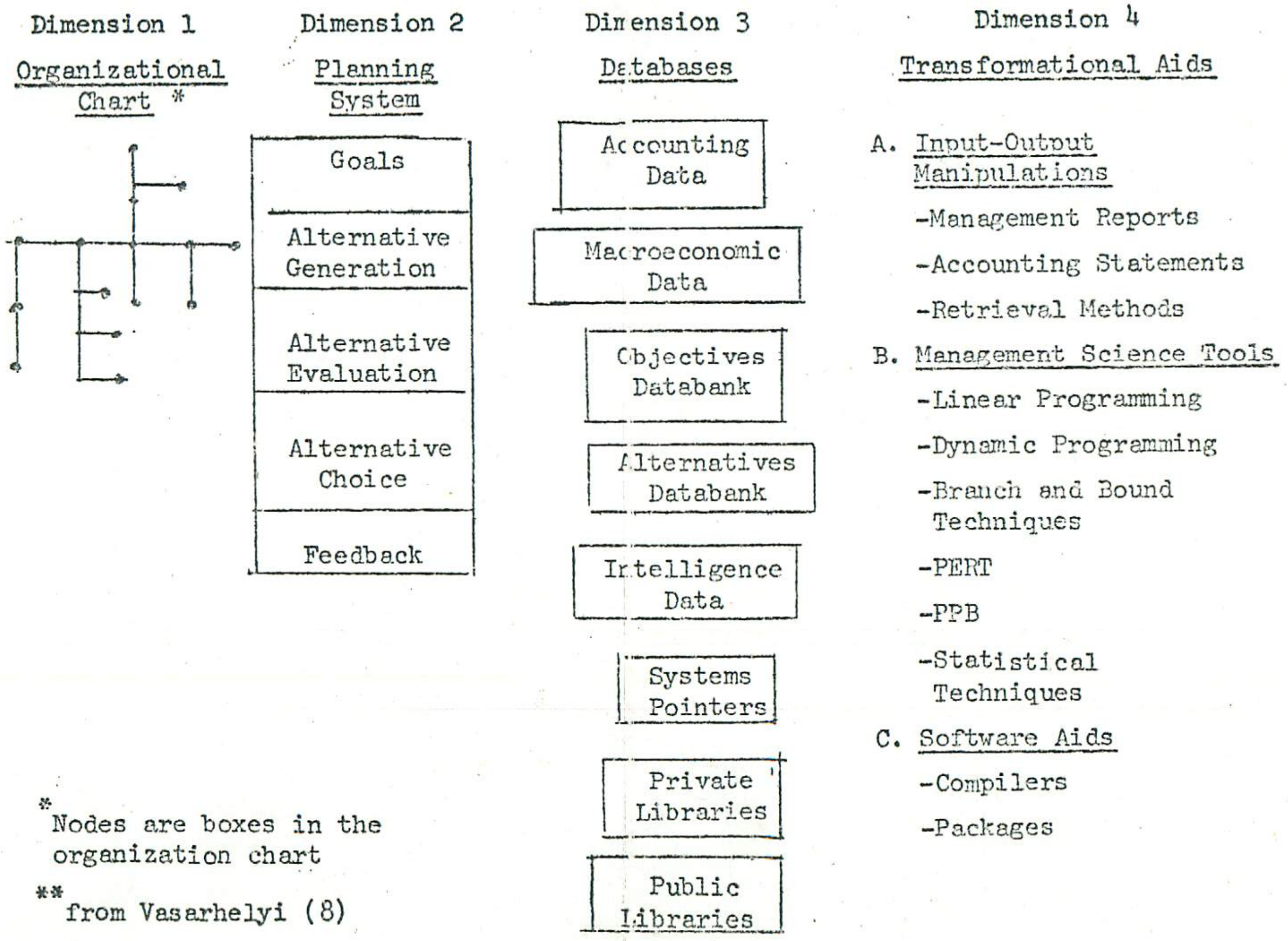
In Section II the main features of the "interactive planning simulator" (IPS) represented in Figure II are presented. Also in Section II is a description of the "online planning experiments" which utilize the IPS for examining behavioral factors around man-machine interaction. These experiments served also to partially test the IPS and to provide quite extensive insights into the utilization of APL as a simulation language.

The concentration in Section III is on the methodological issues surrounding the utilization of APL within the context of a simulation language. This is done by extending the design experience acquired with the IPS by relating special issues to the utilization of APL for simulation.

Considerations related to strengths and weaknesses of APL, online versus batch tradeoffs and the APL features espec-

Figure I

The Interactive Planner **

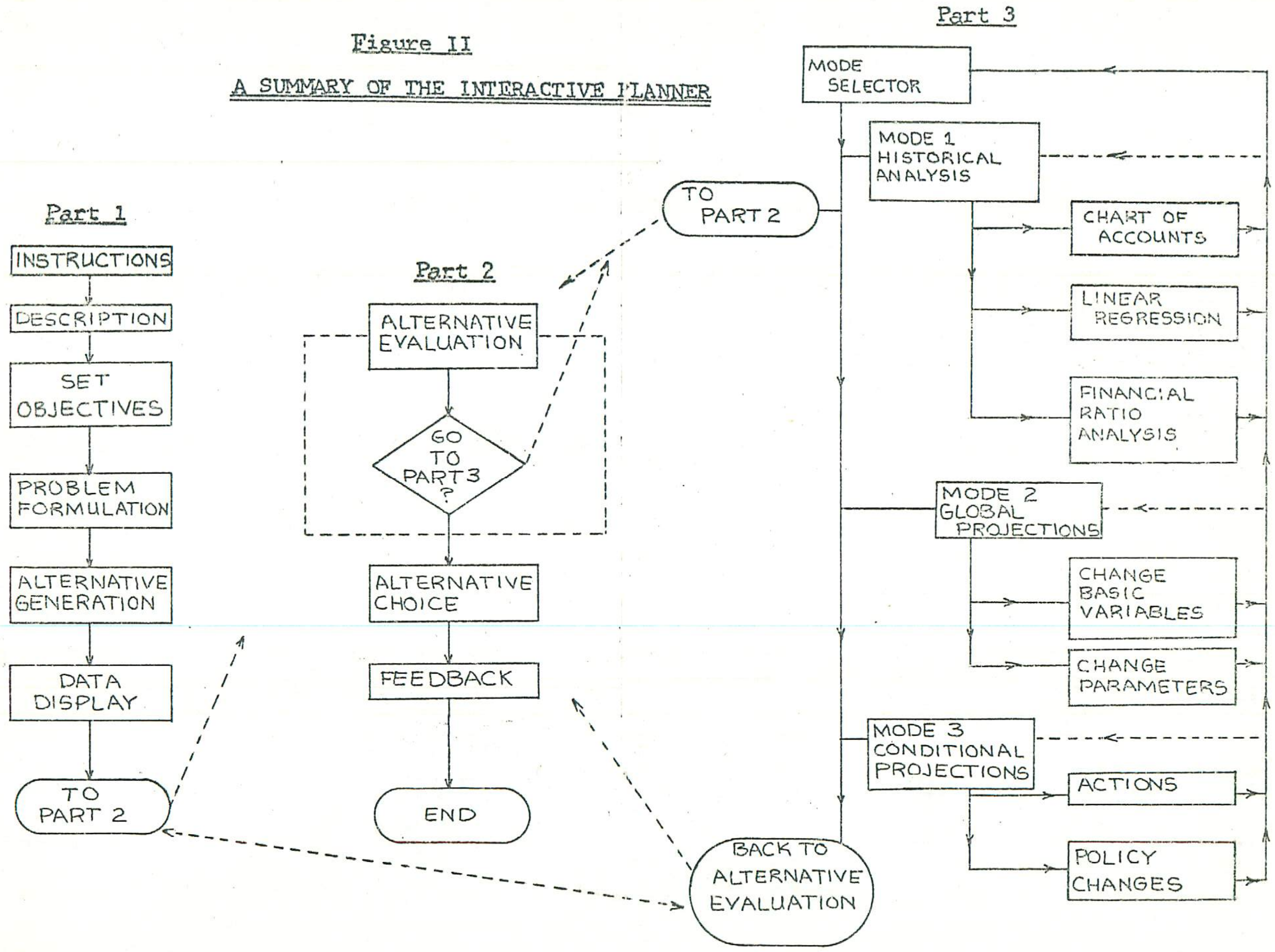


*Nodes are boxes in the organization chart

**from Vasarhelyi (8)

Figure II

A SUMMARY OF THE INTERACTIVE PLANNER



ially suitable to simulation studies are considered.

In the last section, the summary ties the IPS to those considerations presented in section III.

II--Simulating large-scale software systems

Planning is an especially appropriate area of application for online computing technology. Sackman (7) states the following:

> The fundamental justification for going to online planning, as far as the available experimental literature is concerned, lies in the quality of the problem-solving experience and the excellence of the solution.

Also, Benton (2) testifies on the trend towards the utilization of online systems:

> With the installation of on-line real-time systems designed to handle more and more of the every day aspects of business, the future looks much more exciting for managers interested in asking "What if...".

These "What if?" questions will be responded to by a careful planning process utilizing the online real-time systems.

In online planning, the planner can spend some of his time exploring possible courses of action by utilizing simulation which will facilitate his decisions about the future. The interactive planning system simulator was designed with these three basic objectives:

(1) to demonstrate that online planning was not only feasible but also desirable.

(2) to test and explore man-machine behavior with special stress on human behavioral factors.

(3) to simulate a proposed large scale-software system that would in a real situation perform organization wide planning function.

The experimental design allowed extensive analysis and hypothesis testing on the first two areas described. The third area, validation, requires that it be compared with a real-life equivalent and examination of the predictive ability of such a surrogate sytem. Up to the present only generalizations and theoretical inferences can be made on this matter.

This section is addressed to the questions of the first two areas described above. This is done by a description of the main features of the IPS, by a description of some technical features of the simulation performed (cost, design and testing) and finally a discussion of the experimental design that addresses these questions.

II.A System description

The interactive planning simulator was designed with three main parts. These parts were complementary and an effort was made to give real symbiotic characteristics to the man-machine system. Emery (2) calls for exactly such characteristics, "In order to make a major improvement in the planning process, we will have to draw upon the best features of both man and machine."

The two first parts draw mainly on an individual's ability to create unprogrammed settings and alternatives as well as to attribute numbers to conceptual frameworks. In part three, the full computational power of the machine is used and upon the individual's prompting performs a simulation of the firm's financial policies and financial reporting. Figure II summarizes the main features of the interactive planning simulator. This system is directive in nature but its flexibility increases considerably when the user masters some of its main features. In Part I the user is presented with instructions as to the essentials of man-machine interaction. These essentials are basically how to respond to a question (by answering YES or NO or selecting some of the machine-indicated answers) and how to make the system more flexible. After this is completed the system inquires of the user whether or not he wants a summary of the features of the system available during the utilization. From this point on, Part I acts as a thinking aid to the planner. It directs him through the processes of objective setting, problem formulation and alternative generation. The final step in this process is the data display module where the user can request to see the data contained in the database of the firm and obtain independent or aggregate plots of the data.

In Part II the user learns of those features intended to help him in making his final alternative choice. He is directed to Part III when he is asked if he would like to numerically evaluate some of his alternatives. In that case he can opt to receive descriptions of any of the features available in Part III and then is given instructions on how to transfer to that section. On the other hand, if the user does not choose to go to Part III he is asked to assign values to his original alternatives and weight the desirability of the objectives. These values are presented in a matrix format to the user who can re-enter different values until he is satisfied with the results. Most users only reach this point after completing the numerical analysis in Part III and after gaining a deeper knowledge of the problem than they had initially. At several points of Part II the user is asked if he wants to change either his unstructured decisions (objectives, alternatives etc.) or his analysis. If he does so he is sent to Part III. Part III is the numerical evaluation of alternatives. It has three basic modes:

(1) Historical analysis--historical data, trend analysis, linear fits and a financial ratio analysis module.

(2) Global projections--the user is given N years of projections based on historical performance (a linear fit is found for historical performance and projections are made accordingly) and is allowed to change the equation and basic variables of his projections.

(3) "What if" mode--the user can test different possible accounting actions or policies by the utilization of a financial behavior simulator. This could be described as a simulator within the interactive planning simulator as it serves as a simulation model for strategy testing. Reports are supplied in the form of financial statements such as balance sheets, income statements and fund flows. If the user enters no strategy change or accounting actions the simulator would give the projections that mode II of this section would supply.

Once the capabilities of the system are understood the attention of this paper should be directed to some of the simulation implementation features such as costs, language selection, computer support, experimental design, a summary of results and some additional features of interest to the simulation buffs.

II.B Costs

The development of the IPS software took approximately one year of one man's effort as his Doctoral dissertation (10). In terms of computer costs the total effort came to approximately $3,500.00 of which $2,900.00 were spent on APL. The remainder was spent on batch runs of utility packages and standard statistical packages on the data such as SPSS. Some of these costs were incurred by obtaining a backup for the APL system in the case of total computer catastrophe. Of the APL costs approximately 60% were spent directly on connect time (400 hours), 35% on cpu time and 5% on disk storage costs.

Of the connect time approximately 200 hours were spent on straight software development. About 150 APL programs were created of which only 100 were utilized in the final version of the IPS. Another 40 hours were spent in performing the experiment (mean of 3.5 hours connect time per subject) and 20 in data analysis.

II.C Interactive design of interactive systems

The first four subjects that participated in the "online-planning experiment" were part of a pilot run the objective of which was to clear the system of final "bugs". What actually happened was that few program bugs were incountered but a sizeable series of utilizational problems were present. From the feedback sheets of one pilot run the designer would restructure the problem areas overnight and have the next pilot run the experiment with the new features. These changes, initially large, became quite negligible after the last pilot run. However, the efficacy of having this "interactive problem clearance" where the interaction is between the system designer, the pilot subject and again the designer seemed to show quite an interesting potential for future system design.

II.D Technical Features

A detailed discussion of APL as a simulation language is presented in Section III. The main reason for the selection of APL was its availability at UCLA and the prerequisite that the language to be used be interactive in nature with strong facilities for designing conversational programs.

UCLA's computer system is an IBM 360/91 which normally has excellent response time and availability. One of the major problems in the experimental part of this project was system unstability. It is difficult to convince a subject that he should wait periods of 10 to 50 minutes under the tenuous promise that the system will come up again. UCLA's APL is enriched by the APL Plus-file system* which provides excellent file-retrieval capabilities decreasing the very serious restriction of workspace size limitations. However at this stage plus-file is still not able to handle the storage of programs and therefore quite cumbersome workspace transfers were required from the user for passing from one part of the system to the other parts or vice-versa.

The version of the Plus-file system used was quite

*APL Plus-file subsystem is an enlargement of APL giving it file access capabilities designed by Scientific Time Sharing Corp.

expensive in terms of cpu time and somewhat slow in retrieval time. This caused cpu costs to go from 10% of the total as estimated to approximately 35%. On the other hand the cost of APL linkup per hour decreased by 50% which allowed practically twice as many subjects to be run than had been initially estimated to be possible. The interactive planner traced and timed all subject actions for research purposes which somewhat slowed down the system. Also, as APL has strong space restrictions in any workspace all texts and data were stored in independent files and had to be moved in and out for most user interactions.

Further details would be out of the context of this paper, however some of the features of the experimental testing of the IPS should be discussed.

II.E Experimental design

The experimental design for the examination of behavioral factors around man-machine interaction was done in 9 steps as shown in Figure III. This design not only explored decision styles of individuals but also man-machine performance and human emotional factors* in man-machine interaction.

Subject screening involved obtaining a representative sample of the general population of man-machine planning systems users. This population was somewhat familiar with business terminology with a wide range of possible backgrounds. The actual subject population was quite heterogeneous in nature; among them businessmen with many years of working experience, MBA students of planning and control, PH.D.

*see Argyris(1)

Figure III

Experimental Design

Step	Description	Explanation
1	Subject Screening	Subject Screening and Selection
2	Instructions	Instructions on System Utiliz. and features available
3	Cognitive Style test	Test to determine decision approach of the subject
4	Case	Reading the case
5	Pre-Question-naire	Examination of attitudes and expectations before the utilization of the IPS
6	Experiment Using the IPS	Subject utilizes the IPS to plan for its firm
7	Memorandum	Subject fills out a memo describing his planning recommendations to top management
8	Post and Open-ended Quests.	Examination of attitudes, feelings and perceptions about the experiment and about the IPS
9	Debriefing	Subject is debriefed by having the objectives and the main features of the research explained to him. His feelings about the research are discussed.

students in several fields and undergraduate students in accounting.

The computer backgrounds of the subjects differed considerably. There were veteran system analysts and managers that have never seen or interacted with computers. There were students who had, had one computer class only; students with no computer experience at all and very sophisticated students with extensive experience in APL software design.

Subjects received a business planning case and were asked to utilize the IPS to solve it. Before and after the experiment questionnaires were administered and attitudes measured. The case contained both qualitative (personalities, opinions, and anecdotes) and quantitative data. Financial statements and other indices were also available through the IPS.

Very early in the experimental stage, it became evident that the IPS was a very powerful educational device. The subjects finished the experiment exhausted but gratified. Most felt that they had learned a great deal and had enjoyed the experience.

The final analysis of the experiment involved three main areas of inquiry:

(1) questions related to the decision-styles of the subjects.

(2) questions related to man-machine performance.

(3) questions related to "human-emotional problems" in the use of computers.

Judges evaluated the quality of computer planning, while the IPS provided a series of traces and time-measurements for subject evaluation.

The next section discusses the main methodological issues related to the utilization of APL as the simulation language.

III APL as a simulation language

The IPS is a deterministic non-time sequential and basically interactive system. Discussions limited in scope to IPS would not enable general conclusions on APL as a simulation language to be drawn. Therefore in this section a few examples of the utilization of APL for basic simulation functions are presented. Discussion of these, added to the IPS experience will then allow more general conclusions to be drawn.

First of all, APL is an interactive conversational programming language for use in the time-sharing mode.

III.A The tradeoffs between batch and time-sharing

The advantages and disadvantages of programming in the time-sharing mode have been extensively discussed in the literature and are briefly summarized here. They basically reduce to a trade-off between efficient use of programmer time v.s. efficient use of machine time. The online mode is significantly faster in terms of time spent programming (see (5),(6),(7) and (9)). This appears to be due primarily to more efficient debugging in the on-line mode. In projects where debugging occupies a significant portion of the software development process, time-sharing mode appears more attractive. However, there is a tendency to be more extravagant with machine time.

In the specific case of simulation, the time-sharing mode has the advantage of monitoring and interacting. The simulator can examine intermediate output, and can terminate or modify the run based on this information.* This capability is not available with batch processing unless specific controls are built into the simulation program.

III.B Features of APL

Specialized simulation languages, in general, were formulated to deal with a specific type of problem, e.g. the flow of items through processing stations. APL has the flexibility to handle more general types of problems since its subroutines are simple and efficient. An APL subroutine does not require the link-editing as do other general languages. An APL subroutine acts as a newly defined operator. For example, a program to calculate the mean of a string of numbers looks like this.

```
        ∇M← MEAN X
     [1] M←+/X÷ρX
```

Now the statement *MEAN X* can be used in another program to yield the mean value of a vector X.

Emshoff and Sisson(4) have identified several features that are necessary for any simulation language.

*This would enable simulation applications such as interactive job-shop scheduling. Where the user would monitor expected workloads in his shop and decide on changes on strategies throughout the simulation.

(1) Create random numbers. The operator '?' is the basis for generating random numbers.. The statement ?9 picks a number from 1-9. The statement *V*←?*N* ρ 9 generates a vector of length N composed of numbers drawn at random from 1-9 with replacement.

(2) Create random variables. The examples given can serve as the basis for random variates. The following program generates random variates from the exponential probability distribution functions using the method of inverse transformation.

```
        ∇T← EXRANDOM N
     [1] T← -⍟1-((?Nρ10001)-1)÷10000
```

It should be noted that this function can be used as a defined operator within a larger APL simulation program just as the *MEAN X* example. *MEAN EXRANDOM X* would give the mean of N random numbers drawn from an exponential function.

(3) Advance time. APL has very strong vector and array processing capabilities. Time can be specified by position in an array. A unit time advance would correspond to a unit increase in the index. Compression and expansion features allow the deletion or expansion of arrays.

(4) Record data for output. Data ouput on the "2741" terminal is slow relative to a line printer, so extensive output would be cumbersome. Often an entire set of figures is not needed; the mean response and the standard deviation may be adequate for the problem under consideration. Built-in statistical routines in APL can provide these and other summary

figures. The APL plus-file system can store significant amounts of data for selected analysis or retieval of the entire file. Some extensions of APL allow output to be printed on an offline printer.

(5) Perform statistical analysis on recorded data. Statistical packages, such as STATPAK 2, are also available in APL public libraries. This makes statistical analysis a simple task.

(6) Arrange outputs in specified formats.* Due to the ease with which operators can be defined by writing programs to perform the desired functions, there exists extensive public libraries of commonly used procedures. Formatting routines are generally available which will output data with flexibility and clarity.

(7) Detect and report logical inconsistencies and other errors. Vasarhelyi and Mock (9) point out the power of APL for debugging. Besides the original APL error messages which are generally meaningful, the programmer has available the TRACE and SUSPEND functions.

At this point, an example of a simple simulation written in APL may be instructive.

III.C An example

Problem: A bread manufacturer makes a profit of $ .20 per loaf for each loaf sold on the day it is baked. Excess production is disposed of at $ .05 per loaf loss. The demand

*unfortunately the basic features of the APL language make formatting cumbersome at times.

for fresh bread has a normal distribution with mean 1500 and standard deviation of 200.

The simulation program is merely two lines of code:

```
    ∇ PROFIT← D BREAD P
[1] PROFIT←((P⌊D)×0.2)-((P-D)×0.05)×P>D
```

For the sake of comparison we asked colleagues to write this simulation in FORTRAN and PL/C. The Fortran program was 64 statements alone, the PL/C 25 statements. The costs in terms of machine seconds was cheaper in APL as it took only 2/60 of one second of cpu time to process 100 observations. On the other hand it took approximately 1/4 of a second in PL/C and nearly 1/2 of a second in Fortran--all done on UCLA's IBM 360/91 system.

There are some other factors which should be noted about this program. The variable D is a vector representing the distribution of demand for fresh bread. A vector of normally distributed random variates with a certain mean and standard deviation can be generated using a routine in the public library. Generating 100 variates with parameters 1500 and 200 required 2/60 of a second cpu time. Saving this vector for use in several different runs is a simple and inexpensive way to achieve correlated sampling for the purpose of variance reduction. In this way the machine does not have to regenerate the same string of random variates for separate runs.

Another aspect of this example that should be noted is that the observations are independent. This type of problem lends itself very nicely to vector processing, which is to be preferred to reiterative statements.

III.D Further considerations

APL is interpretive; that is, each statement, or line of code is compiled and executed separately. This entails high compile and execution costs for long programs and also programs which have extensive loops.

The chief disadvantage of APL as a simulation language is the large interpreting costs. Often simulation programs are long with significant amounts of looping as activities are simulated in different time periods and different situations with updated information. Such a program would be expensive, perhaps prohibitively so, to compile and execute in APL.

There are two basic ways whereby the cost of the simulation can be reduced to a reasonable level:

(1) Reduce the number of statements to be interpreted. The powerful operators, along with the capability for multiple specifications make this a feasible alternative.

(2) Vector and array processing. If V is a vector, the expression V+3 will add 3 to every element of V. Another example, which shows the power of vector processing when observations are not independent is as follows: Suppose we want to compare (subtract) each element of a vector from every other element. The statement $M \leftarrow V \circ . - V$ will result in a matrix M where $M[I;J]=V[I]-V[J]$. In general $M \leftarrow A \circ . g B$ results in $M[I;J]=A[I]gB[J]$ where g is a standard scalar function--such as: $+, \times, -, \div$.

Matrices and higher order arrays can be operated upon in this manner. For example, if M and N are matrices of appropriate dimensions, $M+.\times N$ results in the ordinary matrix product.

A Programmer using APL who thinks in terms of typical Fortran-like loops will encounter high interpretive costs. However, before APL is abandoned for this reason, the possibility of vector processing should be explored; APL offers several avenues for imaginative programming in this regard.

The above considerations involved a series of typical simulation applications not explored in the IPS. The conclusions of this paper will try to integrate the learnings provided by the IPS with the above considerations.

IV. Conclusions

The APL language, with file-extension capabilities proved itself to be not only adequate but extremely appropriate for the simulation of large scale software systems. Findings in the utilization of the IPS not only contributed to the better understanding of behavioral factors in man-machine interaction but indicated the adequacy of simulating large scale software systems.

As an outgrowth of the IPS software development effort, interest focused on APL as a simulation language. Section III discussed APL vis-a-vis different typical simulation features. Figure IV summarizes the advantages and disadvantages of APL for simulation. These factors indicate that APL would be an appropriate language for simulation studies with the following characteristics:

Figure IV

APL for simulation

ADVANTAGES	SHORTCOMMINGS
1. Powerful conversational features	1. Workspace limitations
2. Modularity	2. Interpretative costs
3. Data retrieval characteristics	3. Difficulties in "meta-execution"
4. Compactness of code	4. Compactness of code (documentation)
5. Operator definition capabilities	5. High looping costs
6. Character manipulation features	6. Data input and output features
7. Debugging features	7. Formatting
8. Real-time measurements	
9. Real-time monitoring	
10. Real-time user intervention	
11. Random number generation	
12. Vactor and array manipulation	
13. Public libraries available	
14. Insignificant turnaround time	

(1) Interactive or monitoring capabilities required

(2) Large development costs compared with "utilization" costs

(3) Dynamic modular structures

(4) Fine tuning required

(5) Intense matrix manipulation required

Those characteristics for which APL would not be an adequate language are listed as follows:

(1) Many unavoidable loops

(2) Small software development costs relative to "utilization" costs

(3) Large scale problems

(4) Problems needing large amounts of input or output

(5) Applications for which there exists a specialized simulation language

Concluding, in view of the above stated factors, APL should be taken seriously as an attractive simulation language.

Bibliography

1. Argyris, Chris; "Management Information Systems: The Challenge to Rationality and Emotionality," <u>Management Science</u>, Vol. 17,6; February 1971, pp: B275-292.
2. Benton, William; <u>The Use of the Computer in Planning</u>, Addison-Wesley Publication Company; Reading, Mass; 1971.
3. Emery, James; <u>Organizational Planning and Control Systems</u>; MacMillan; New York; 1969.
4. Emshoff, James R. and Sisson, Roger L.; <u>Design and Use of Computer Simulation Models</u>; The Macmillan Company; London; 1970.
5. Gold, Michael M.; "Time-Sharing and Batch Processing: An Experimental Comparison of Their Values in a Problem-Solving Situation." <u>Communications of the ACM</u>; Vol. 12,5; May 1969.
6. Martin, James Thomas; <u>Design of Real-Time Computer Systems</u>; Prentice Hall; Englewood Cliffs, New Jersey; 1967, pp: 46.
7. Sackman, Harold; "Advanced Research on On-line Planning: Critique and Recommendations;" <u>Systems Development Corporation</u>; SP-3480; Santa Monica, Ca; April 1970.
8. Vasarhelyi, M. A; "Simulation: A tool for design and pre-implementation testing of large-scale software systems;" <u>Proceedings of the 1971 Winter Simulation Conference</u>; pp: 261-267.
9. Vasarhelyi, M. A. and Mock T. J; "The Art of online Debugging;" <u>Proceedings of the Online 1972 Conference</u>; Brunel University, England; September 1972.

10. Vasarhelyi, M. A; "Man-Machine Planning System: a Cognitive Style Examination of man-machine interaction;" unpublished doctoral dissertation, UCLA, January 1973.