SIMULATION: A TOOL FOR DESIGN AND PRE-IMPLEMENTATION TESTING OF LARGE SCALE SOFTWARE SYSTEMS

Miklos Antal Vasarhelyi
University of California at Los Angeles

ABSTRACT (*)

This paper discusses the application of simplified, surrogate (simulated) software systems in the pre-implementation, design and evaluation of large scale software systems.

In earlier research a theoretical framework for an interactive computer language for financial planning has been postulated.[10] This paper discusses the utilization of a surrogate computer system as a tool to test the feasibility of such an interactive planning system. This surrogate system is programmed in APL and serves both as a validation of a planning theory and as an ex-ante device for the prediction and evaluation of expected system performance.

Surrogate systems are held to be a most useful methodology in developing and analyzing large scale systems.

Introduction

This paper has the main objective of proposing a methodology for pre-implementation testing of large scale software systems. The examples used to clarify this methodology throughout the paper are drawn from a research project still in progress at UCLA.[10] Therefore, these examples should not be considered as conclusive evidence but simply as a device to better illustrate the process being proposed.

The Problem of Large Scale Software System Development

Large-scale software development is one of the most serious management problems that exists in the usage of modern data processing systems.

Enormous costs overruns, large time delays and minimal control over the quality of the projects being developed are common problems in most software development projects.[5]

Several methods have been developed to apply Management Science tools for better planning, control and management of software development. Unfortunately none of these methods has been very successful in terms of the above mentioned task.

In addition, budgets have also been unreliable indicators of resources to be dedicated for software development. However, the main shortcoming lies, as in many other R & D efforts, in the difficulty of early identification and abandonment of low payoff projects or projects with low probability of success.

This is clearly stated by Jones and McLean:

> "There is a natural unwillingness to abandon a part of a system or a program in which considerable time and effort have been invested, even when subsequent developments indicate that another approach would be clearly superior"[5]

As a partial solution to these kinds of planning difficulties this paper proposes simulation as a way to:

1) Test possible software systems
2) Identify problems before considerable investments are made
3) Develop realistic budgets and schedules for the development of large scale systems.

The Problem from a Management Perspective

Managers seem to exhibit two extremes of computer utilization. Either they become deeply involved in the development of the EDP system and have thorough knowledge of its structure, or they are ignorant of its functioning and even distrust most computer related activities.

This phenomenon comes from the intrinsic nature of computer systems. They are complex, specialized and require large investments of time for learning. To narrow this gap system designers have been trying to develop systems that do not require comprehension of data processing phenomena. Conversational systems were basically developed for this purpose.

The simulation approach of this paper

(*) The author would like to thank Prof. T. J. Mock for his suggestions on this paper.

provides a basic framework to be used by the neophite manager and concurrently makes it possible for the sophisticated user to add his own options to the public libraries and databases. It advocates the generalization of the methodology illustrated in an academic utilization to utilizations in industry and government.

An Interactive Planning System: simulating a proposed large scale software system

An interactive planning system (IPS) is being designed and simulated as part of the doctoral dissertation of the author. Its main features can be summarized as follows:

1) The IPS is simulated by the planning system simulator (PSS)
2) The corporational environment is simulated by the usage of a case
3) The whole PSS is interactive in nature, therefore, allows the user to experiment with different alternatives
4) Planning situations are staged and subjects solve them with the aid of the PSS for one set of subjects and without for the other
5) Configurations of the PSS differing in terms of modules and features are given to the subjects for their usage
6) Analysis is conducted both on the performance results and on the planning process performed by the subjects.

The main question that arises from this description is: Why should one use simulation for the development of an IPS?

Simulation was defined by Mize and Cox as:

> "...the process of conducting experiments on a model of a system in lieu of either:
>
> 1) Direct experimentation with the system itself
> 2) Direct analytical solution of the problem associated with the system"[7]

In the context of the illustration a total development of the system would not be compatible with the nature of a doctoral dissertation or feasible given the scope of available resources. On the other hand, the issues around formalization of the planning effort, the performance of planning in an interactive mode, and the application of management science tools to planning were of deep research value and interest. Thus the dilemma of learning about a large scale system without actually implementing it confronted the researcher as it confronts all system designers.

Simulation became the method through which the large scale software development would be avoided but its desirability, principles and features could still be tested.

In the next section, the actual method of building surrogate systems and environments is proposed.

Simulation of Large Scale Software Systems

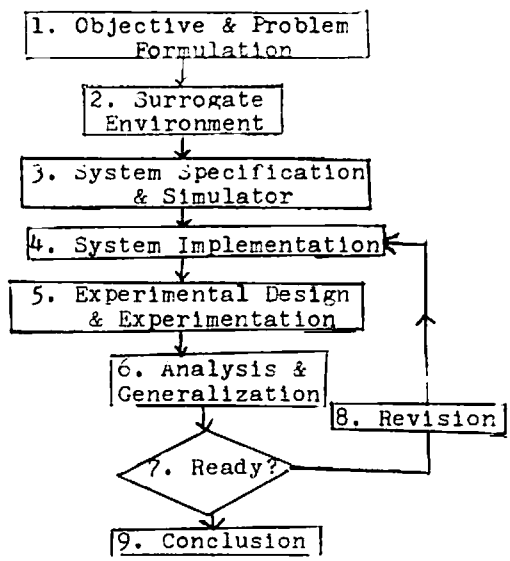
The usage of simulation points out dramatically the need for clear statement of objectives in a software development project.

These main objectives that encompass both the philosophy of the project and the basic user needs are tested in the simulator. This proposed methodology does not examine marginal or peripheral features nor design details.

The main objectives of the IPS were to build a system, conversational in nature, usable by the neophite manager, with the characteristics of what Gerrity calls a man-machine decision system.[4]

The main steps for simulating a large-scale software system are flowcharted in Figure I.

Figure I
Flowchart of the proposed steps for the Simulation of a large-scale software system



In order to clarify Figure I a step-by-step discussion of the proposed procedure follows:

Step 1. Establish the main objectives and problem specifications of the system being proposed.
This step is very often neglected in systems design. But an accurate problem

statement can generate considerable savings in time and effort. Several techniques can be used to improve the problem specification. Churchman suggests looking at the whole system and identifying its goals, resources, environment, components and management.[2]

Step 2. Design a surrogate environment in which the system will be tested.
This should encompass a scenario within which the system will be used, the main variables affecting the utilization of the proposed system and typical decision problems that the system will help to solve. Techniques such as questionnaires, direct observation and system analysis are very useful in pin pointing the main relevant features of the environment.

Step 3. Design the basics of the proposed system and choose the simplifications to be adopted in the simulator.
The features and simplifications to be adopted in the simulator should be based on the objectives of both building the system and of performing the simulation. The simulator should:

1) Have the main features of the system being proposed and described in step1
2) Consider the power and limitations of the tools to be used
3) Consider the scope of user needs
4) Consider the limited nature of the simulation process.

Step 4. Construct the surrogate system.
This is the software development stage of the simulation. Design, programming and testing are to be performed. Always keep in mind throughout this stage that what is being developed is not the system but simply a simulator. Valuable insights of the design problem for the larger system can be drawn. Therefore, it is important to keep careful documentation of the development of the simulator.

Step 5. Experimental design and experimentation.
Typical utilization situations are devised within the simulated environment. Subjects are used to test the system with different configurations and problems. Outcomes and processes performed are measured.

Step 6. Analysis and generalization.
Measurements are analyzed and different configurations compared. Cost - Benefit analysis is performed on the incremental (non-essential) modules. Generalization from the results of the simulation to the proposed system must be considered including analysis of its consequences related to the proposed system.

Step 7.. Decision.
Decision on the adequacy of the systems tested. Should other configurations be tested? Should other specifications be added? Is the simulated environment adequate?

Step 8. Revision.
The shortcomings and inadequacies found in the earlier step are revised and return to step 3 of this description.

Step 9. Conclusion.
Final system design. Preparation of the budget and schedule for systems development.

Figure II illustrates these steps by associating them to actual steps of a simulation study. Figure III on the other hand shows how the main features of a larger proposed system can be reduced to a smaller but representative system simulator.

Methodological Issues Around the Planning System Simulator

A brief description of features of the PSS are discussed in this section. The objective is to clarify the proposed methodology by means of an example.

A case was used to simulate the environment. Another alternative to perform this task in the development of an MIS is to simply have managers describe past situations and use the simulator to solve these. During the construction of the PSS, the data from the case was placed in a large matrix to be accessed by the planning routine. This matrix was the main quantitative element of the database which also includes data on macroeconomic events, objectives of the different levels of the corporation and possible alternatives to solve the problems in question.

These objectives were used as standards to evaluate the quality of the outcomes (generated plans).

The problem of measurement of the quality and desirability of a software system is a rather complex one. Obviously, the first criterion is how well a system fulfills the objectives of its designers and of the organization implementing it. On the other hand, techniques will have to be used on the evaluation of the results of the simulation. The intangibles associated with a large-scale software system are great: How to quantify the advantages to a corporation of an error-free system? How to consider customers satisfaction with a better service time? How to assign figures to the advantages accruing to a corporation from a better decision system?

On the other hand, it is important to evaluate the competing configurations by

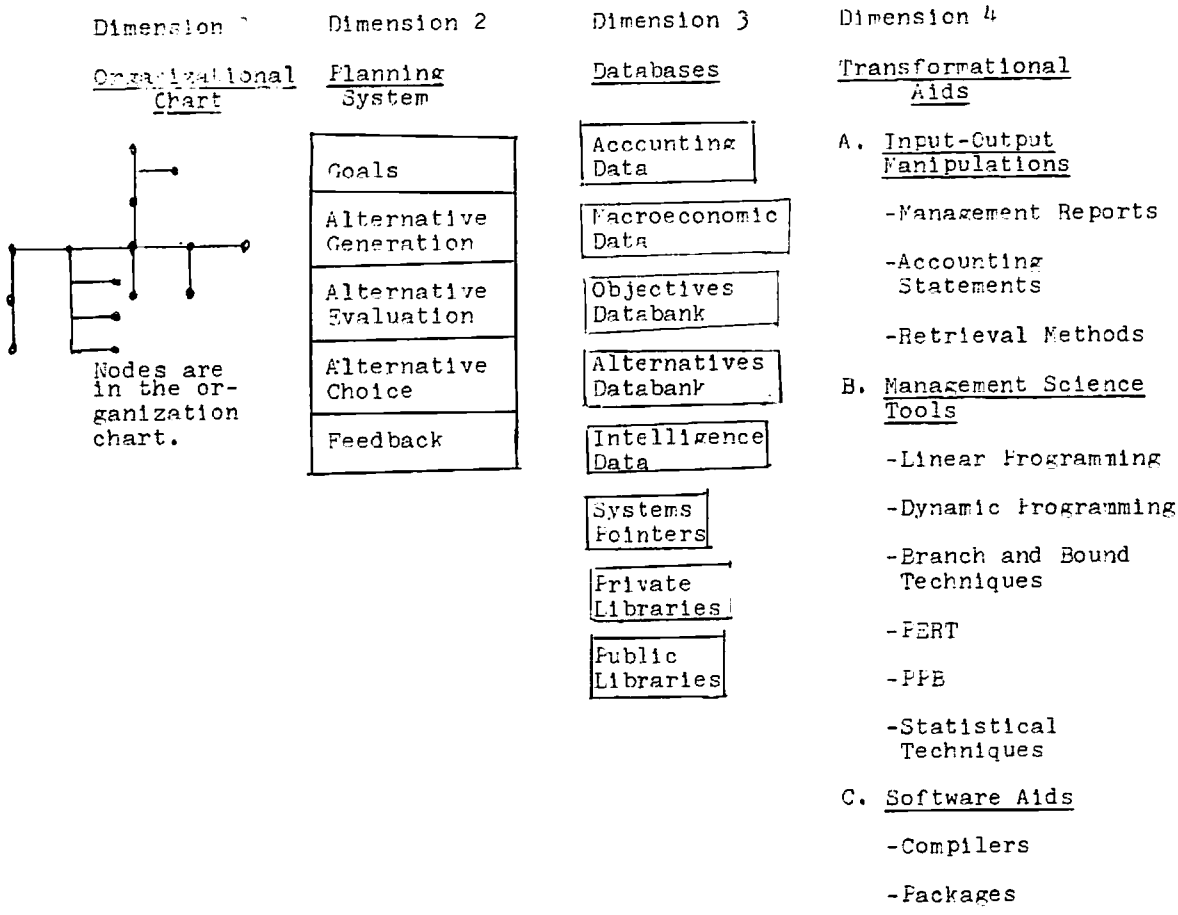
Figure II

Simulation of the IPS

1)	Objective & Problem Formulation	-Directive planning system -Vertical and horizontal integration of planning systems -Real-time access to databases -Interactive nature of the planning system -Availability of Management Science tools for neophites
2)	Surrogate Environment	-Organization-wide system -Manufacturing concern -Accounting data -Non-computer specialist management
3)	System Specification & Simulator	-Environment to be represented by a case -Retrieval system simply manipulating a matrix with the principle data -Database for the corporation but only one decision-maker at a time -Management Science tools to be made available: -linear programming -financial analysis packages -forecast and projection package
4)	System Implementation	-Build a matrix of data form -Use APL for the programming of the main features -Build four dimensions for the PSS: -organizational -directive planning system -databases -computational aids -Use successive APL workspaces for simulating a larger core
5)	Experimental Design & Experimentation	-Create some "typical" utilizations and have managers utilize the surrogate system for their solution -Measure time taken for decision: costs of utilization, system development and partial costs: measurement of "quality" of decision made with the system
6)	Analysis & Generalization	-Analysis of the results in terms of cost and benefits -Analysis of the "non-quantifiable" features of the system, e.g. user satisfaction, public image improvements due to speed, accuracy acquired, etc. -Analysis of the system breaking down into essential modules and incremental modules: ranking of incremental modules: by usefulness -Generalization: place these results within context of the IPS, evaluate how desirable is each feature -Estimate difficulties, main problems, and resources required in terms of money, time and manpower for the IPS
7)	Decision	
8)	Revision	-Add or subtract features of the system, revise steps 2 and 3, and start again on step 4
9)	Conclusion	

Figure III.A

The IPS



some kind of criterion. A cost-benefit analysis is hereby suggested. Criteria such as time taken to make decisions, cost of computer time, estimated costs of software development, implementability and evaluation of the quality of the plans generated are being used in the PSS.

Techniques can be borrowed from social science research in order to measure the less quantifiable features. A panel of judges is being used to evaluate the quality of plans.

The choice of computer languages is also an important aspect to be considered. The main points to be considered are: availability of software, adequacy for the task in question, core requirements, utilization costs, compiler accuracy, programming skills, hardware considerations, etc.

APL was chosen for the PSS due to the following reasons: its interactive nature, the facility of designing conversational programs, its availability at UCLA, its strong computational features, the ease to acess public libraries, and the competence of the researcher in APL programming.

Some of the shortcomings of APL for the PSS are: memory limitations due to the compiler, cumbersome output formatting, and the serious limitations on system manipulations within programs.

Finally, on the issue of resource requirements for simulation of large scale software systems, it is important to consider the following points:

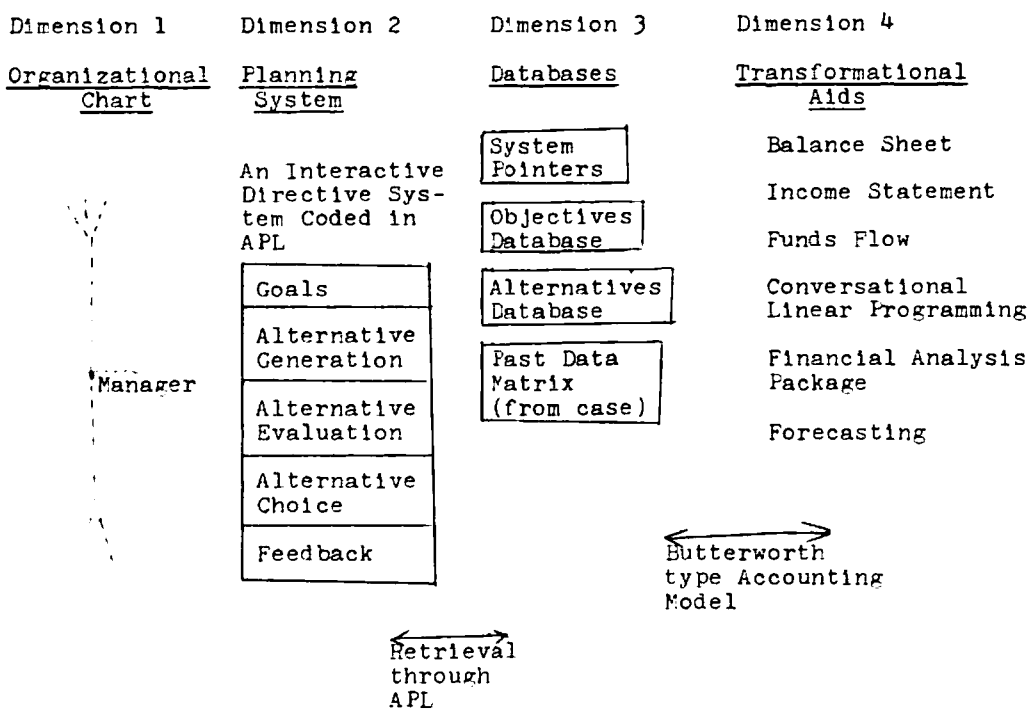
1) Simulations are costly but part of this cost is recovered through learning

2) Risks of total system development failure or inadequacy of the proposed system are decreased considerably

3) Simulation can be used as a tool to get management involvement in the development of MISs.[1]

Figure III.B

The PSS



4) Management help and time will be required to improve proposed systems.

The total amount of resources required for the development of the PSS is not known yet. Up to the present time, half of the effort went into research design and into the development of the methodology. Programming took approximately 25% of the effort and analysis of the results 15%.

The total effort until now has taken approximately a year and a half of the researcher's full-time effort, $1,500.00 in computer time, around 100 hours of faculty supervision and a considerable number of hours of cooperation of fellow students acting as subjects for the experiment and helping programming, debugging and designing the research.

It is likely that a non-academic simulation would have a much smaller percentage of the effort concentrated on design.

Conclusions

The simulation of a large-scale software system is only worthwhile if the findings about the simulator are generalizable to the system being analyzed.

If this is true, the approach presents innumerable features that will be useful for the user, including:

1) Pre-implementation testing of proposed software systems

2) Testing of alternate configurations for such systems

3) Considerable improvements in budgetary accuracy in terms of money, time and manpower

4) Better management decisions about system development in Management Information Systems

5) Early identification of key problems.

On the other hand, there are disadvantageous aspects that should be considered such as:

1) The cost of the simulator

2) Experimental costs and time

3) The possible non-validity (non-representativeness) of such a simulator

4) Dispersion of manpower by having programmers and system analysts dedicated first to the construction of the simulator and then to the MIS development.