TEACHING AND RESEARCH NEWS: Integrating Data Flow Models and System Dynamics in Information Systems Design Courses

Don M. M. Booker, Pace University, 150 Nassau St, N.Y., N.Y., 10038 212-285-3486, June 28, 1982.

1. INTRODUCTION

A large body of recent work in information system design models has encouraged the use of structured data flow models (Yourdon and Constantine, 1978; Gane and Sarson, 1977; Jackson, 1975). For the past several years, the computing and information systems curriculum at Pace University has emphasized incorporating the data flow approach to systems analysis and design in the two term introductory management information system design course, the on-line system design course, the data base design course, and to a lesser extent the decision support system design course.

The data flow model was originally conceived to achieve the goals common to other structured programming methodologies – primarily the eventual production of correct (e.g. meeting defined requirements), complete and error free program code and operating procedures constrained to exhibit only the three primary control structures of sequence, iteration and choice, and organized by heirarchical invocation.

While the data flow model has served well in teaching information system design at a detail or program design level, (Bohl, 1978), and as a method for requirements definition and analysis, (Bergland 1981; King, 1981), several weaknesses have emerged. It is difficult to relate to the wider functional requirements and behaviour of the business firm and its environment. It is difficult to incorporate less well defined, non-computerized, information processes and decisions, both on an operational level and at a policy level. It is particularly difficult to integrate policy level decision support systems design methods with lower level operational information systems methods and models at a quantitative level.

While the structured data flow models are intended to be "dynamic" models of information flow, they do not in fact lend themselves easily to quantitative estimates and analysis of actual dynamic data flows. This becomes evident at a fairly basic design level when estimation of physical design parameters such as data volume and frequency is attempted rather than logical description and decomposition of data elements and structures. In more advanced courses such as on-line systems design and distributed systems design, this weakness becomes more apparent. It becomes extremely difficult to effect a smooth instructional transition from structured data flow design models and methods to quantitative techniques needed for cost-benefit analysis, alternative design evaluation, and performance monitoring and measurement.

In an effort to overcome these weaknesses, current practice has been to introduce some system dynamics models and techniques (Forrester, 1961, 1968; Goodman, 1974). These models are introduced as a natural extension of the structured data flow models we have been using. This allows us to introduce a more quantitative approach and a dynamic simulation capability to the design process. The integration of detail level information system design models and wider models of the firm is facilitated (Roberts, 1978). An earlier and more detailed model and analysis of alternative physical systems design is possible. Simulation of on-line and data base performance models can be included along with case studies and smaller problems.

Recent efforts at improving the auditability of large information systems (Weiss, 1982) have suggested the possibility of using simulation as a technique for auditing of complex information systems. Since auditability and improved management control are primary goals for all our design courses, the incorporation of systems dynamics simulation models provides yet another advantage to design courses for actual design practice.

Our instructional approach emphasizes the use of business cases (Kutscher, 1981; Copeland, 1954) for information system design rather than smaller problems. When the micro version of Dynamo is released we plan to complement the individual design cases assigned to students with more extensive interactive simulations.

2. EDUCATIONAL GOALS OF INTEGRATING DATA FLOW MODELS AND SYSTEM DYNAMICS MODELS

The MIS CIS program at Pace University has several educational goals. Primary among them is the integration of professional practice and theory (Hoare, 1981). Our aim is to prepare our students for leadership positions upon graduation as systems analysts and design in the large corporate environment, as well as, for graduate study in management, information systems or computing success. We attempt to integrate theory with contemporary practice in the design and installation of very large information systems.

However, university resource limitations and student time constraints severely limit the scope of projects which can be assigned in the undergraduate curriculum. We have attempted to cope with these limited resources by aiming for a high degree of integration in the case material used in different courses in the MIS program. Our aim is for the student to develop a series of projects and case studies which deal with increasing depth and sophistication while utilizing substantially the same problem material. First the case problems are approached as a simple batch design, then as an on-line system, next as a data base problem and finally as a decision support system and business policy problem. At each point appropriate theoretical, conceptual and applied programming and language tools and techniques are introduced while expanding, deepening and changing the emphasis and complexity of the case framework.

This approach is particularly suitable for the introduction of system dynamics based models of the firm to the case material and the use of Dynamo to model and simulate the effects of alternative emphases (on-line systems, distributed systems,

data base design, etc) on the problem focus and factual nature of the case materials.

Our aim is not only for the student to be capable of detail program, algorithmn and file design, but also to be able to integrate smaller individual system components, designed in individual courses, into the broader management contenxt of the firm and correctly, completely and consistantly define business requirements which can be embodied in an economically achievable system design and a technically and organizationally feasible working system.

2.1 Curriculum Objectives

These educational goals requires several (frequency disparate, not to say dysfunctional), curricular objectives (compare DPMA Educational Foundation, 1982):

(1) Mastery of programming skills in Assembler and COBOL languages.

(2) Mastery of computing science design concepts and criteria for operating systems, network design, data base design and programming language design.

(3) Mastery of system analysis and design tools and techniques, particularly the use of data flow models for business requirements definition.

(4) The development of the ability to integrate subsystems of programs, devices, procedures and data bases into a functional whole which meets management requirements. Familiarity with specific application systems requirements for accounting and financial systems, MIS, DSS, order entry and production control systems, auditing, and managerial control systems.

(5) The development of a systems view of computing, information systems and the business enterprise as a whole. The ability to apply and relate a systems perspective to wider social and economic processes and problems and view them from a systems perspective.

In order to accomplish these goals within the resources available, each course must build upon and be integrated with the other courses in the curriculum, reinforcing a structured systems perspective, while introducing the appropriate new concepts, techniques and language and design tools appropriate to the individual course objectives.

3. TEACHING TECHNIQUES AND COURSE CONCEPTUAL CONTEXT: DATA FLOW DIAGRAMS AND SYSTEM DYNAMICS MODELS

In addition to providing a framework which encourages the analysis of complex processes as a whole, the use of a system dynamics provides several unique techniques and advantages for an integrated information systems curriculum.

3.1 Emphasis on Control: Structures, Processes and Data

One of the weaknesses of most traditional system design approaches is the lack of an *integrated* emphasis on system control. By controls I mean the entire range of techniques from management controls, through project controls, design controls, down to program and data controls including even the algorithmic control structures and data elements required for the use correctiveness assertion proof techniques.

I would even expand the classical HIPO (hierachial input process output) (IBM, 1972, model to encompass an explicit feedback and a control "box". Such a "HI-POCIT" model of the basic system Design Process would emphasize a Hierarchical Input-Process- *Control and Interactive Testing Structure*, similar to ORR's HIPO+DATA BASE model (ORR, 1978; 1980).

A system dynamics model allows us to easily illustrate with casual diagrams positive and negative feedback effects, to trace the effect of cascaded error propagation and to illustrate the importance and effect of varying types and quality of management controls.

The conventional data flow models either treat control structure and processes "in passing" or specifically exclude them (Demarco, 1979). By introducing casual diagrams and Dynamo language features and statements such as rate and level, we can retain a data flow network to Dynamo model isomorphism while emphasising and adding specific *control structures and elements* to the data flow *models.*

3.2 Introducing Dynamic Effects to the System Design Course in Order to Encourage Interactive Psychological Feedback

Perhaps the greatest advantage in using Dynamo is the ability to allow the student to actually model a system design as a *dynamic functioning process.* Many students are first attracted to programming by the immediate positive (and occasionally negative) psychological feedback provided by the computer programming and debugging process – particularly that provided by interactive computing environments (Couger and Zawacki, 1981, pp. 24-25). Dynamo allows the system design course to take advantage of the same positive reinforcement and interactive satisfaction the student receives in debugging smaller programming exercises for large system design problems and exercises. (Schneiderman, 1980; Weinberg, 1971; Martin, 1973).

The "Gestalt" apprehension of complex structures still seems to be aided by some form of graphical representation. A picture, even a poor one, is still worth a thousand words. I am attempting to integrate the use of the commonly recommended graphical system design tools, (working "upward" from a basic directed graph network model) *and* to incorporate the use of simulation in order to encourage the student to achieve the psychological advantages of an ability to *interactively manipulate* the geographical model. I would like students eventually to be able to "turn over" and "play" with the system model on the CRT tube, as well as, eventually in their mind and on paper.

3.3 Graphics Ismorphism, and Concept Assimulation

One basic assumption in my course design is that the student will more easily assimulate complex conceptual models and analysis techniques if (1) a visual graphical or geometrical representation is available and used, and (2) if different techniques are related to previously learned structures, either as special cases or more advanced generalizations.

Thus, the basic mathematical model to which I return again and again is the directed graph, and its various permutations into detail program flow charts, system flow charts, data flow diagrams, activity flow designs, data structure diagrams, hierarchy charts, linked lists, module linkage charts, casual diagrams and Dynamo flow diagrams.

I would like the students to be able to expand their analysis and design skills from a simple directed graph, through an equivalent matrix model, on to a stochastic model as appropriate to the problem requirements and the measures of performance available.

The student should be able to easily convert from a program or algorithmic perspective to a structural or graphical model of a system and back again while applying the appropriate quantitative or analytical tool to the solution of those aspects of the system design which require (and are amenable) to precise, rigorous solutions. I believe experience with system simulation eases the students' efforts to make this transition from qualitative to quantitative systems modeling and analysis.

4. CONCLUSION

By combining simulation with various graphical structure and models, I hope to eventually aid the student in developing the ability to mentally visualize the structure and process isomorphism inherent in all systems, (Alexander, 1973; deRosnay, 1979) and to design from the perspective of either a data, decision or a process/procedure model.

REFERENCES

1. ALEXANDER, C. *Notes on the Synthesis of Form.* Harvard U.P., Cambridge, Mass. (1973).
2. BERGLAND, G.D. "A Guided Tour of Program Design Methodologies", *Computer,* October 1981, pp. 13-34 (1981).
3. BOHL, M. *Tools for Structured Design,* S.R.A., Palo Alto, Ca. (1978).
4. COPELAND, M.T. "The Genesis of the Case Method in Business Instruction", in M.P. McNair (editor). *The Case Method at the Harvard Business School.* McGraw Hill, N.Y. (1954).
5. COUGAR, J.D. and ZAWACKI, R.A. *Motivating and Managing Computer Personnel.* Wiley, N.Y., (1980).
6. DEMARCO, T. *Structured Analysis and Systems Specification,* Yourdon, N.Y., (1978).
7. D.P.M.A. Educational Foundation, *DPMA Model Cirriculum for Undergraduate Computer Information Systems Education.* South-Western.
8. DEROSNAY, J. *The Macroscope: A New World Scientific System.* (trans. by R. Edwards) Harper and Row, N.Y., (1979).
9. FORRESTER, J.W. *Principles of Systems,* M.I.T. Press, Cambridge, Mass. (1968).
10. FORRESTER, J.W. *Industrial Dynamics,* M.I.T. Press, Cambridge, Mass. (1961).
11. GANE, C. and SARSOW, T. *Structured Systems Analysis: Tools and Techniques.* Prentice-Hall, N.J. (1979).
12. GOODMAN, M.R. *Study Notes in System Dynamics.* M.I.T. Press, Cambridge, Mass. (1974).
13. HOARE, C.A.R. "Professionalism", *Computer Bulletin,* September, 1981, pp. 2-4, Vol 27 (1981).
14. IBM Corp. *HIPO – A Design and Documentation Technique.* (Tech. Manual, GC-20-185), IBM Corp. (1972).
15. JACKSON, M.A. *Principles of Program Design,* Academic Press, N.Y., (1975).
16. KING, D. "Current Methodologies in Structured Design", *Computerworld,* (In Depth section, pp. 1-44) (1981).
17. KUTSCHER, R.I. "Purpose and Potentials of the Case Method in Policy/Strategy Instruction", Presented at the Academy of Management (1981).
18. MARTIN, J. *Design of Man-Computer Dialogues.* Prentice-Hall, N.J. (1973).
19. ORR, K. *Structured Systems Development.* Yourdon, N.Y. (1978).
20. ORR, K. *Structured Requirements Definition.* Q.E.D. Information Sciences, Wellesley, Mass. (1980).
21. ROBERTS, E.B. (editor). *Managerial Applications of System Dynamics.* M.I.T. Press, Cambridge, Mass. (1978).
22. Schneiderman, B. *Software Psychology: Human Factors in Computer and Information Systems.* Winthrop, Cambridge, Mass. (1980).
23. WEINBERG, G.M. *The Psychology of Computer Programming.* Van Nostrand Reinhold, N.Y. (1971).
24. WEISS, H. "Innovative Uses of the Computer as an Audit Tool", Presented to the EDP Auditors Association, N.Y. Chapter, June 14, 1982. Automation Center, Reston, Va. (1982).
25. YOURDON, E. and CONSTANTINE, L. *Structured Design.* Yourdon Press, N.Y. (1975).