

Proceedings of the
**Twenty-Fifth Hawaii International
Conference on System Sciences**

Volume III: Information Systems

Decision Support and
Knowledge-Based Systems

Edited by Jay F. Nunamaker, Jr.

Sponsored by the University of Hawaii

In cooperation with ACM, the IEEE Computer Society, and
the Pacific Research Institute for Information Systems and Management



IEEE Computer Society Press
Los Alamitos, California

Washington • Brussels • Tokyo

A Prototype for Including Simulation of IS Dynamics in CASE Environments

James R. Warren, Jack W. Stott, and A. F. Norcio

Department of Information Systems
University of Maryland, Baltimore County
Baltimore, MD 21228

(301) 455-3206

Internet: jwarren@umbc1.umbc.edu, stott@umbc1.umbc.edu,
norcio@afn.ifsm.umbc.edu

Abstract

This paper describes a method to integrate information systems (IS) design performance evaluation with the IS development process. The nature and cost of an information system is shaped by decisions about where to provide computer support, the hardware platform, and the database management system (DBMS) architecture -- decisions influenced largely by how quickly work must be done. A prototype system has been developed which produces simulation results automatically from data flow diagrams (DFDs) augmented with information regarding the performance of system components. The objective is to make the evaluation of IS design dynamics a common and integral part of the development process by producing simulation results directly from computer-aided software engineering (CASE) tool data dictionaries. The prototype reads DFDs from a custom DFD-drawing tool and formulates a corresponding simulation model. The prototype provides model-based expert advice in the use of the simulation model and in the interpretation of its output. The use of the prototype is illustrated through its application to a proposed research clinic information system.

1: Introduction

The research described in this paper is predicated upon two assumptions.

1. Analysts should analyze system dynamics while developing Information Systems (IS), and
2. The only way this will happen is if the analysis of IS dynamics is made an integral part of the CASE environment.

Concerning the first assumption, we define IS dynamics on several levels.

- Physical level. At this level the developer is interested in assessing differences in performance of hardware or system/operating software alternatives. Typical tools used to assess the physical level dynamics are benchmarks and simulations of different hardware configurations. Typical problems addressed at this level include the choice of hardware and data communications platforms. For a further discussion, see [1].
- Data Dynamics level. At this level, the developer is interested in assessing changes in overall system performance with alternative *data handlers*. For instance, when the relational data base management systems first appeared there was concern that they did not have the required performance level. Developers, and hence users, had to forego the increased capabilities that the RDBMS offered for the sake of performance. However, most of these decisions were made without basis in fact or much analysis. For a more in depth discussion of database dynamics, see [1].
- Single-Design Evaluation level. The developer is interested in assessing whether or not a particular design is feasible. Simulation is used to assess whether the work flow can be handled by the number of work servers in the system [2, 3].
- Design-Alternative Selection level. The developer uses simulation of system alternatives to choose "best" design alternatives. This also includes assessing the *focus* of automation. With the growing importance and understanding of organizational re-engineering [4] it is important for the developer to have tools with which to evaluate different organizational structures and IS structures.

For the physical level, it is common practice to use simulation, especially for data communications designs, and there are several tools available [5, 6]. The use of simulation to assess the performance of alternative database architectures is also an established practice [7]. For this research, however, we are concerned with the last two levels, the evaluation of logical IS designs.

Concerning the second assumption, we believe that there are three fundamental problems that must be overcome and that they can only be overcome if the analysis of IS dynamics is made an integral part of a CASE tool. These problems are:

- Developing the simulation models. Not only is it time consuming, but it takes skills that most IS developers lack. Also, developing programs in any language, including simulation languages, is error prone, making validation a difficult issue.
- Running the models. If developers use common simulation environments, they are faced with another tool to understand, and also they must understand the technical aspects of running simulations (See section 2 for more on this.)
- Interpreting the results. The developer must be well versed in the interpretation of statistical data in order to understand the results of stochastic simulation.

Stochastic discrete-event-based simulation of IS designs can be implemented using simulation 4GLs such as GPSS/H and SIMSCRIPT II.5, but most IS developers lack the time and expertise to write a simulation program every time a question arises about the dynamics of an IS design, or to modify the simulation for each design alternative. Simulation environments aid in the use of simulation, but still require IS professionals to express problems in notations outside of their accustomed paradigm [8, 9].

Two efforts [10, 11] show that there is some interest in the integration of simulation with CASE, yet no system has delivered the kind of useful environment necessary. The efforts to add system dynamics evaluation capabilities to IS development environments have typically ignored the three fundamental problems that must be overcome in order to make the assessment of dynamics common-place.

This paper describes part of an ongoing research project with the goal of delivering the analysis of IS dynamics to the IS developer. The second goal is to test the assumptions stated in the opening paragraph and evaluate how well the technology improves the IS developers' product. Section 2 gives a quick overview of using simulation to make decisions and some of the difficulties

encountered. Section 3 describes a prototype system developed to deliver simulation to IS developers. Section 4 then describes a case study using the tool, and finally section 5 offers some conclusions, insights and future research directions.

2: Using stochastic simulation to make decisions

Stochastic (i.e., pseudo-random-number-based) simulation has been widely used for system performance evaluation, but is often presented without attention to the statistical nature of simulation output [12]. The results of a single stochastic simulation run represent merely an observation from a population of possible simulation results. Stochastic simulation can never yield the outcome that a given parameter of the simulated system is exactly x . One must be content with a *confidence interval*; that is, "We are $p\%$ confident that the true value lies between $x - \delta$ and $x + \delta$." One of the simplest methods of generating confidence intervals is the method of independent replications, where a simulation run is repeated a number of times to obtain a set of independent observations. For an unknown parameter μ_x of the simulated system, $P(\mu_x = \overline{X}(n) \pm t_{n-1, 1-\alpha/2} s) = 1 - \alpha$, where $\overline{X}(n)$ and s are the mean and standard deviation of the independent observations of μ_x , respectively, and n is the number of replications. The important simulation run parameters in the method of replications are the *simulation duration*, the *warm-up period* (the amount of time the simulation is run and observations are discarded on the suspicion that the system has not yet reached a steady-state), and, of course, the *number of replications*.

Expert systems have been developed to aid simulationists in the statistical issues of simulation [13]. Expert system support can include aid in determining simulation duration and number of replications, assessing the impact of initial conditions, constructing confidence intervals, comparing output from alternative designs, and estimating response/input variable relationships. Knowledge-based support can also be provided through on-line help embedded in a simulation environment. The help system for the simulation model development environment (SMDE) includes: (a) a general assistance manager, offering an introduction to the SMDE, tutorials on usage of the tools in the environment, and a glossary of terms; and (b) help routines local to specific tools [14].

3: A prototype IS design dynamics simulation environment

A prototype system has been developed to support the IS developer in the analysis of IS design dynamics with stochastic simulation. The prototype runs under the X Windows System on a Hewlett-Packard HP 9000 series 370 workstation. The environment presents the developer with three windows (see figure 1). In the upper-left the developer has a DFD-drawing utility. This allows the specification of system structure and dynamic attributes. In a true integrated-CASE environment, DFD-drawing should be performed using a middle-level CASE tool, where the DFD-components are integrated with the IS-

design's data dictionary. In the lower-center portion of the screen is a simulation run parameter input window. This window allows the user to input the parameters to specify a method-of-replications simulation experiment. The prototype automatically interprets the DFD as a network of queues and conducts a simulation under the specified parameters. The user can receive help in formulating the simulation run parameters by clicking on the parameter for which assistance is desired. The upper-right window presents the results of the most recent simulation run. Standard performance statistics (utilization, throughput, waiting time, etc.) are reported for each DFD process and for DFD data stores that are central to the dynamics of the IS design. The developer can receive help in understanding the simulation output, confidence interval

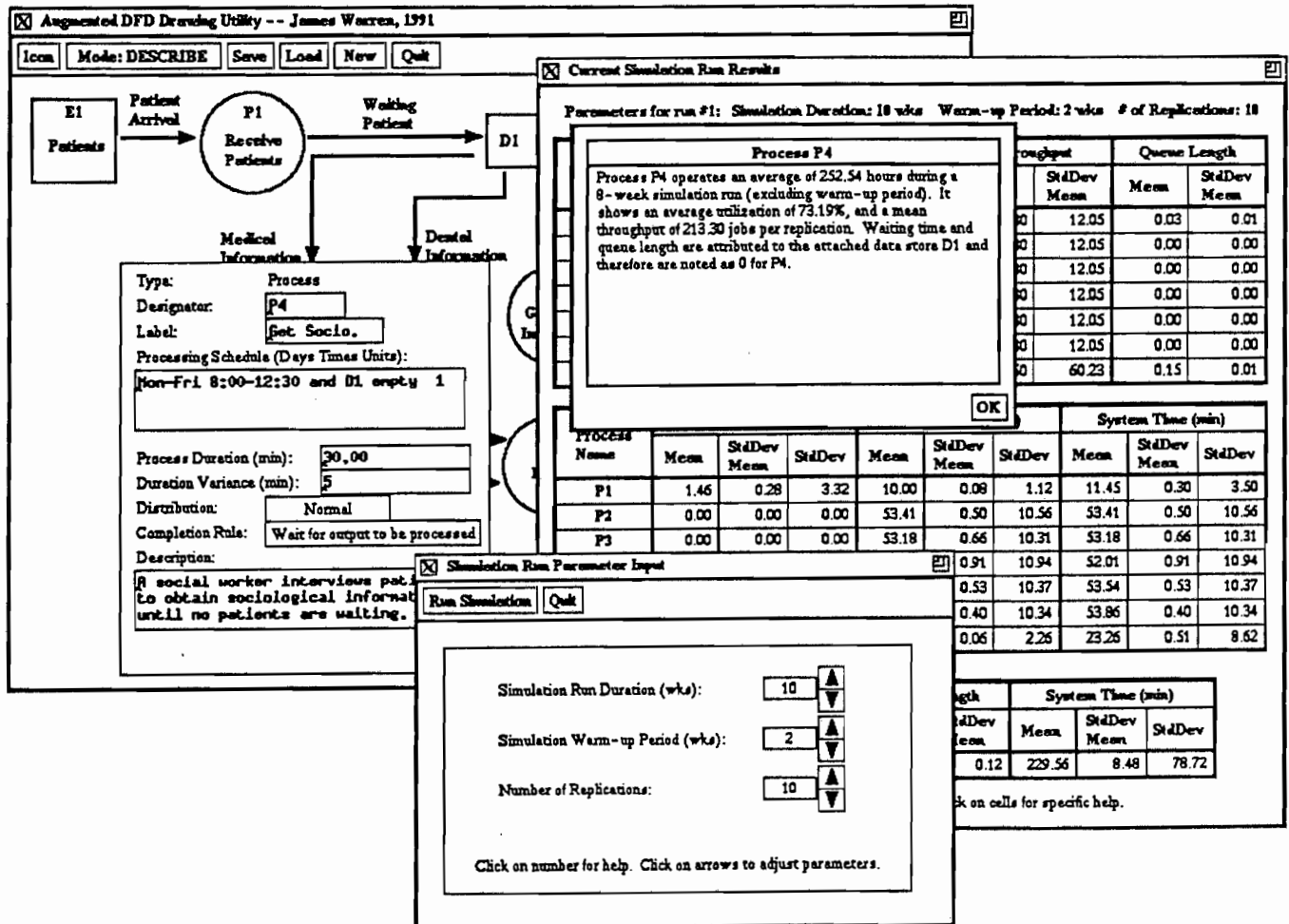


Figure 1. The prototype IS design dynamics simulator runs under X11 on an HP9000 series 370 workstation. The environment presents three windows: (a) a DFD-drawing utility with pop-up windows to specify attributes of DFD components, (b) a simulation parameter input window with parameter-specification help, and (c) a simulation results window which explains the output, generates confidence intervals, and provides heuristic warnings of processing bottlenecks.

generation, and heuristic recommendations regarding potential processing bottlenecks from the simulation results window.

We first describe our method of basing simulation models on DFDs, then describe the simulation usage support provided by the prototype.

3.1: Automatic simulation from Data Flow Diagrams

The prototype IS simulator produces simulation results directly from DFDs drawn using an augmented DFD-drawing utility. The mapping from DFDs to simulation is accomplished using the model that each DFD *process* represent a server/queue system; that is, each DFD process represents a line of "jobs" and someone/something that accomplishes those jobs, sending them on to whatever process, data store, or external entity is the output of the process. The conventional DFD notation shows the direction and contents of flows through the system, but lacks several items relevant to an IS design simulation. Ordinarily, DFDs do not indicate the *rate* at which jobs are processed, nor do they indicate the amount of time that servers are available to process jobs. The augmentation of DFDs with such dynamics attributes is straight-forward. Figure 2 shows the pop-up editing window associated

with a DFD process in the augmented DFD-drawing utility. The window in figure 2 allows the user to specify:

1. A designator and label for the process, as normally appears on a DFD.
2. A processing schedule (indicating that the process functions on Mondays through Fridays, from 8AM until 12:05PM, but does not shut down until no jobs are awaiting service).
3. A processing duration, variance, and distribution (5 minutes per job with a standard deviation of 2 minutes, following the normal distribution).
4. A processing completion rule (which could indicate that the process server should wait for a completed job to be received or completed at a subsequent process before continuing to the next job in the queue).
5. A textual description of the process.

When the developer selects the "Run Simulation" button on the simulation run parameter input window, the discrete event-based simulator of the prototype reads the internal representation of the augmented DFD. The simulator maps the DFD to a network of queues, where the DFD attributes define the service rates and distributions for the queuing systems. Note that it would constitute a minor modification to a middle-level CASE tool such as Index Technologies' Excelsior to allow attachment of attributes to the DFD components, allowing the tool to serve as the front-end for a simulation environment such as this prototype.

3.2: Simulation usage support

The simulation usage support of the prototype is implemented as an on-line help system rather than in a more obtrusive format. The objective is not to impose a model on the developer's decision-making style, but only to offer assistance in the proper use and interpretation of simulation when the developer requests assistance (by clicking the mouse on the object of interest). Assistance takes the forms of: (a) definitions, (b) explanations and calculations, and (c) recommendations.

On the simulation run parameter input window the developer indicates the need for help by clicking on a parameter. The prototype will offer a definition of the indicated parameter and give the developer the option of asking for a recommendation. When the developer requests a recommendation, the prototype analyzes the current simulation results to ensure that they are in keeping with its model of proper simulation usage. (If no simulation run has yet been conducted, the prototype offers a heuristic default recommendation and advises that it will be able formulate a more precise recommendation

Type:	Process
Designator:	P1
Label:	Receive
Processing Schedule (Days Times Units):	Mon-Fri 8:00-12:05 and idle 1
Process Duration (min):	10
Duration Variance (min):	1
Distribution:	Normal
Completion Rule:	Do not wait
Description:	Receptionist receives patients, explains procedure, and directs them to wait for the specialists.
OK	

Figure 2. Pop-up editing window for DFD process attributes.

once a simulation has been run.) The simulation usage model contains three major principles:

1. The difference between simulation duration and warm-up period should be adequate to allow a sufficiently high number of observations per simulation run to secure normality of the replication means (based on analysis of process throughput).

2. The warm-up period should not end until the running mean utilization of each process approaches a constant level with a given accuracy δ^7 .

3. Increase in the number of replications or the simulation duration should narrow the confidence intervals of output parameters, but lengthening of the simulation duration is the more efficient of the two methods.

The prototype uses the first two of these principles when making recommendations about simulation duration and warm-up period. Either the parameter is indicated to be adequate or a larger value is recommended for one or both of the parameters. The third principle is implemented in the prototype by responding to the developer's request for a recommendation on number of

replications with an explanation of the relationship between confidence intervals and the simulation run parameters.

The simulation run results window offers output-interpretation help in three forms, depending on the area of the simulation output indicated by the developer:

1. Column headers provide a definition of the indicated output parameter.

2. Row headers provide a summary explanation of the all the output shown for the indicated process or data store. A heuristic recommendation is made if the process utilization is sufficiently close to 100% to indicate a possible bottleneck.

3. Cells in the table body provide a confidence interval of the mean of the indicated parameter.

4: Example use of the prototype

We will now illustrate the use of the prototype via application to an IS-component for a research clinic to support an inter-disciplinary health-care team for geriatrics

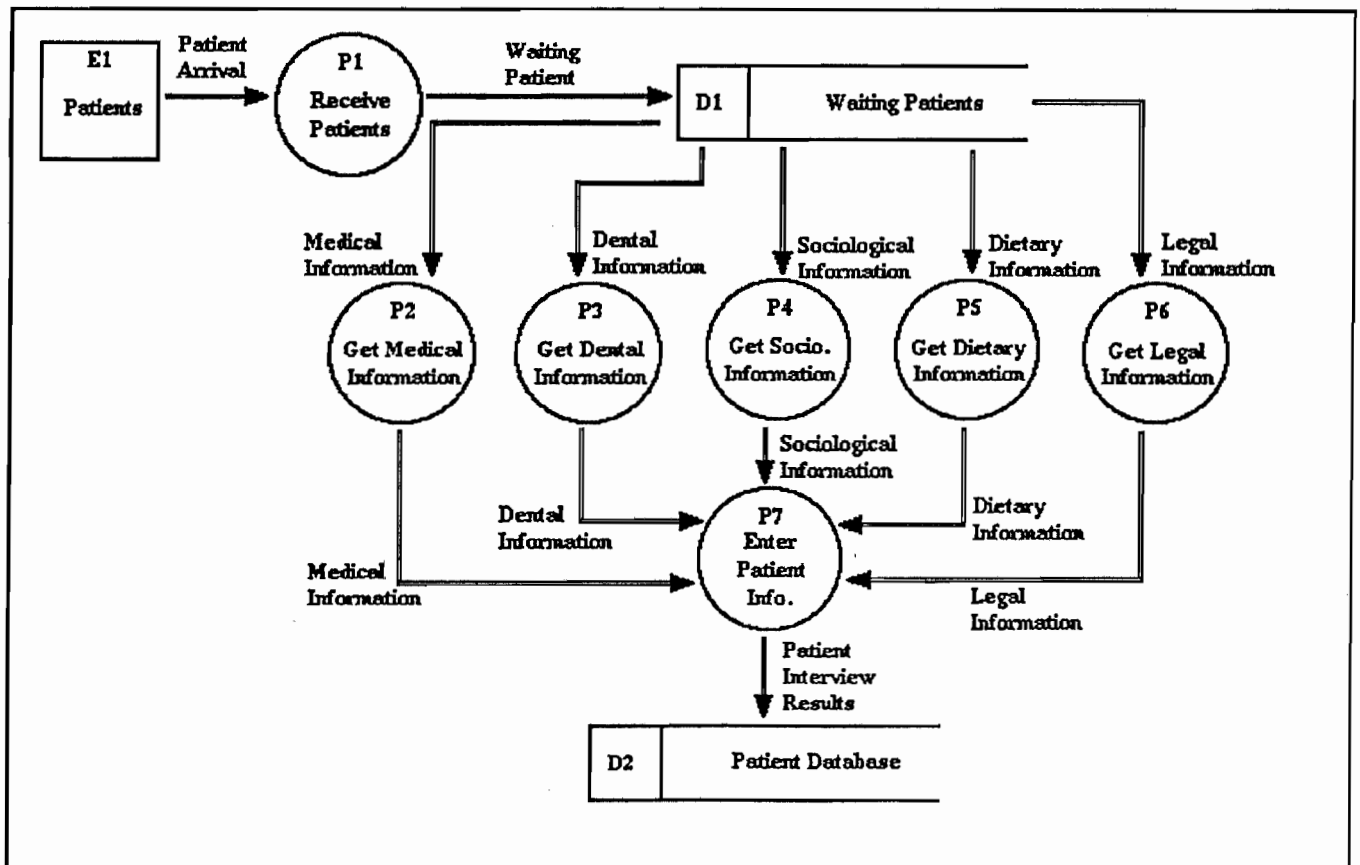


Figure 3. Data flow diagram of the data acquisition function of a research clinic IS to support an Inter-disciplinary health-care team for geriatrics and gerontology.

After each interview, the specialist enters the information at a terminal (process P7), waiting for the other specialists to finish at the terminal if necessary. Note that the designers want to implement this system using only one terminal (probably a microcomputer) to keep implementation costs low. The data entry requires 40 percent of the duration of the interview. The specialists remain at work each day until the waiting room is empty**.

Type:	Process
Designator:	P2
Label:	Get Medical
Processing Schedule (Days Times Units):	Mon-Fri 8:00-12:30 and D1 empty 1
Process Duration (min):	30.00
Duration Variance (min):	5
Distribution:	Normal
Completion Rule:	Wait for output to be processed
Description:	Physician interviews and examines patients to get their medical history until no patients are waiting.
OK	

Figure 4. Annotation for process P2 of the research clinic.

and gerontology*. One component of the proposed research clinic IS involves the collection of interdisciplinary data from geriatrics patients for the purpose of later analysis. Figure 3 shows the DFD of the data acquisition component of the research clinic. Patients arrive Monday through Friday, 8AM to 12PM at an average rate of one every 45 minutes. A receptionist receives the patients (process P1) and directs them to the waiting room (data store D1). The waiting patients visit each of five specialists once: a doctor (or nurse practitioner), a dentist, a social worker, a dietician, and a lawyer. Each of the specialists spends an average of 30 minutes (normally distributed with a standard deviation of 5 minutes) interviewing and/or examining the patient to acquire information (processes P2 through P6).

* The clinic project is being jointly funded by the Geriatric Gerontology Education and Research Program, University of Maryland at Baltimore, and the Loch Raven Baltimore Veterans Affairs Hospital.

Designator:	F3
Label:	Medical
Starting at Data Store:	D1
Terminating at Process:	P2
Delay Duration (min):	0
Delay Variance (min):	0
Distribution:	Uniform
Imports jobs with:	
Absent	Medical Information
Don't Care	Dental Information
Don't Care	Sociological Information
Don't Care	Dietary Information
Don't Care	Legal Information
Description:	Patient provides physician with their medical history.
OK	

Figure 5. Annotation for the data flow between data store D1 and process P2 of the research clinic.

The system dynamics are encoded in the attributes of the DFD components using the augmented DFD-drawing utility of the prototype. For example, figure 4 shows the annotation for the data flow between D1 and P2. The window specifies that P2 services (imports) jobs with the "Medical Information" attribute absent regardless of the state of the other four attributes. The attributes are defined in the annotation of D1 and represent the five types of information acquired by the specialists. Service at P2 causes the "Absent" attribute "Medical Information" to become "Present" in the serviced job. When all five

** This is a considerable simplification of an actual research clinic, where, for example, first-time patients always visit the doctor and dentist before the other specialists and returning patients may visit any subset of the five specialists.

attributes are present, then job (i.e., patient) is done and exits the data store. Figure 5 shows the annotation for process P2, "Get Medical Info." The processing schedule indicates that one processing unit is available Monday through Friday, from 8AM to at least 12:30PM, and that unit does not become unavailable until no jobs are awaiting service in data store D1. The completion rule "Wait for output to be processed" directs the simulator to wait for a job to be processed at P7 (the output destination of P2), before continuing to process the next job (i.e., conducting the next patient interview).

Figure 6 shows the simulation output results for the clinic with a simulation duration of 10 weeks, warm-up period of 2 weeks (which is actually unnecessary since the

system starts from empty each day), and 10 replications. This simulation requires about 45 seconds of real time. A cursory view of the utilization figures indicates that the system dynamics are not out of control. Note that because jobs at D1 wait simultaneously on processes P2 through P6 (waiting for an available specialist that has not yet been visited), the queueing statistics (queue length and waiting time) for these processes are zero. The system time at data store D1 indicates that patients complete the interviewing process in less than four hours on the average (estimated mean of 229.56 minutes), which is acceptable to the developer.

Viewing the performance of process P7, however, reveals a possible problem with the design dynamics.

Current Simulation Run Results								
Parameters for run #1: Simulation Duration: 10 wks Warm-up Period: 2 wks # of Replications: 10								
Process Name	Utilization (%)		Operating Time (hr)		Throughput		Queue Length	
	Mean	StdDev Mean	Mean	StdDev Mean	Mean	StdDev Mean	Mean	StdDev Mean
P1	21.71	1.28	163.74	0.15	213.30	12.05	0.03	0.01
P2	71.02	1.96	267.26	11.24	213.30	12.05	0.00	0.00
P3	68.36	2.17	276.45	11.06	213.30	12.05	0.00	0.00
P4	73.19	2.53	252.54	10.09	213.30	12.05	0.00	0.00
P5	66.04	2.42	288.16	10.52	213.30	12.05	0.00	0.00
P6	63.69	2.07	300.48	10.23	213.30	12.05	0.00	0.00
P7	15.84	0.86	1344.00	0.00	1066.50	60.23	0.15	0.01

Process Name	Waiting Time (min)			Service Time (min)			System Time (min)		
	Mean	StdDev Mean	StdDev	Mean	StdDev Mean	StdDev	Mean	StdDev Mean	StdDev
P1	1.46	0.28	3.32	10.00	0.08	1.12	11.45	0.30	3.50
P2	0.00	0.00	0.00	53.41	0.50	10.56	53.41	0.50	10.56
P3	0.00	0.00	0.00	53.18	0.66	10.31	53.18	0.66	10.31
P4	0.00	0.00	0.00	52.01	0.91	10.94	52.01	0.91	10.94
P5	0.00	0.00	0.00	53.54	0.53	10.37	53.54	0.53	10.37
P6	0.00	0.00	0.00	53.86	0.40	10.34	53.86	0.40	10.34
P7	11.28	0.53	8.81	11.98	0.06	2.26	23.26	0.51	8.62

Data Store Name	Throughput		Queue Length		System Time (min)		
	Mean	StdDev Mean	Mean	StdDev Mean	Mean	StdDev Mean	StdDev
D1	213.30	12.05	0.94	0.12	229.56	8.48	78.72

Click on table header for definitions. Click on cells for specific help.

Figure 6. Simulation output for the research clinic.

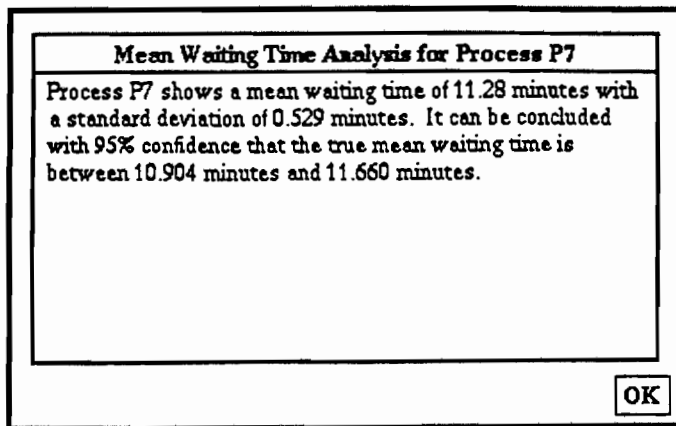


Figure 7. Confidence Interval generation for mean waiting time at process P7 (enter-patient-info.) of the research clinic.

Clicking on the "Mean Waiting Time" cell of process P7 causes the prototype to pop-up the window shown in figure 7. We are 95% confident that specialists wait an average of 10.904 to 11.660 minutes in line to enter their interview data for each patient. When this is coupled with the time required for data entry itself, the service times of the specialists are around 53 minutes per patient (rather than the 30 minutes per patient required for the interview). Note that the simulation environment merely describes the scenario; it is up to the developer to assess the acceptability of the design dynamics and to formulate alternatives to unacceptable outcomes.

In fact, having specialists wait for a terminal an average of approximately 11.3 minutes is unacceptable. The developer investigates two candidate design modifications that may offer solutions to this problem:

1. Increase the number of terminals to two. This reduces the 95% confidence interval of the mean waiting time at P7 to between 0.658 and 0.746 minutes, resulting in service times for P2 through P6 lowering to around 42.5 minutes per patient.

2. Use only one terminal, but formulate the data entry task at P7 such that it requires only 20 percent (rather than 40 percent) as long as the patient interview. This design results in a mean waiting time at P7 with a 95% confidence interval between 1.592 and 1.732 minutes, and service times for P2 through P6 around 37.5 minutes per patient due to the reduction in both queuing and service time at P7.

The dynamics of each of these design modifications is easily investigated by modifying the annotation to process P7, and each modification results in a much more acceptable system dynamics scenario. The implementation of these modifications, however, is the more difficult matter. Can a system with two terminals

be delivered within budgetary constraints? Can a human-computer interface be designed which allows the specialists to enter the results of a patient interview in only 20 percent of the duration of the interview itself? The simulation environment allows the developer to quickly quantify the value of these modifications to the system dynamics. At this point, it should be clear how this tool promotes the use of system dynamics simulation as an integral part of the IS development process; assessing designs, considering modifications, assessing the modifications.

5: Conclusions

Direct production of IS design dynamics simulations from DFDs allows the developer to gain the benefits of simulation without side-tracking to code a simulation in a 4GL or to re-encode the design structure into a specialized notation. This paper describes a prototype simulation environment to support the dynamic modeling of information systems designs from DFD-based system descriptions. The prototype produces simulations automatically from DFDs, provides a model for simulation usage, and aids in the interpretation of simulation results. The prototype uses a custom DFD-drawing utility; however, the most beneficial configuration would be one in which the DFD-drawing capability and data dictionary of an integrated CASE environment were used to supply the system structure for simulation, as this would allow immediate assessment of the dynamics of proposed designs directly from the IS design tool.

The simulation environment answers "what-if" questions about the IS design. The prototype must be provided with processing rates for each of the DFD processes; rates based on both machine and human factors. The developer may formulate measures of IS-component performance from benchmarking experiments, published reports, systematic observations of task performance, and/or past experience. Once the developer has specified the dynamics of system components, the prototype can be asked, "If the components perform like this, *what* is the overall state of the system dynamics?" This can be thought of as providing evidence for the confirmation (or denial) of the acceptability of the dynamics of the IS design. It can also be thought of as producing a powerful set of dynamics requirements for the IS-component designers; *if* each component performs at least as well as specified, *then* the dynamic performance of the IS should be at least as good as the simulation results indicated. In the difficult, often mind-boggling task of IS development, there is a reassuring feeling about anything as straightforward as "If I do this, *then* I'll get that."

References

1. J. Warren and J. Stott, Database Dynamics: Why and How, *Proceedings, 24th Hawaii International Conference on Systems Sciences*. Kauai, HI, January, (1991)
2. J. Warren and J. Stott, "CASE/Simulation: Making performance evaluation a normal part of information systems development," *Proceedings of the Second International Working Conference on Dynamic Modeling of Information Systems*, July, Washington D.C., pp 155-176. (1991)
3. R. Crosslin, R. Freedman, and D. Sutherland, A Dynamic Approach to Systems Analysis, Design, and Evaluation. Center for Research in Information Systems Working Paper No. 88-1. (1988)
4. M. Hammer, Reengineering Work: Don't Automate, Obliterate, *Harvard Business Review*, July-August, 104-112. (1990)
5. A. Gore, QASE to Configure Huge Systems, *MACWEEK*, November 13, 20. (1990)
6. A. Schwartz, Y. Yemini, and D. Bacon, NEST: A Network Prototyping Testbed, *CACM*, 33(10), 63-74. (1990)
7. M. Eich, C. Fan, W. Sun, and S. Rafiqi, A Methodology for Simulation of Database Systems. *Simulation*, June, 241-254. (1989).
8. H. Watson and J. Blackstone, *Computer Simulation*, 2nd ed, John Wiley & Sons, New York. (1989).
9. O. Balci and R. Nance, Simulation Model Development Environments: A Research Prototype, *J. Opl. Res. Soc.*, 38(8), 753-763. (1987).
10. L. Boydstun, D. Teichroew, S. Spewak, Y. Yamamoto, and G. Starner, Computer Aided Modeling of Information Systems, *Proceedings, IEEE COMPSAC80*, 37-41, Chicago, October. (1980).
11. J. Pallatto, Cadre to Expand Suite of CASE Workstation Tools, *PC Week*. November 26. (1990).
12. K. Pawlikowski, Steady-State Simulation of Queuing Processes: A Survey of Problems and Solutions, *ACM Computing Surveys* 22, June, 123-170. (1990).
13. Y. Park and J. Mellichamp, A Statistical Expert System for Simulation Analysis, *Proceedings, Summer Computer Simulation Conference*, 611-616, Calgary, July. (1990).
14. V. Frankel and O. Balci, An On-line Assistance System for the Simulation Model Development Environment, *Int. J. Man-Machine Studies* 31, 699-716. (1989).

Acknowledgements

Special thanks are owed to Dr. "Kip" Canfield (Department of Information Systems, University of Maryland Baltimore County) for his helpful suggestions at several stages in the development of the prototype IS simulator.