1986

A MODEL FOR EVALUATING THE PERFORMANCE OF OPERATIONAL LEVEL INFORMATION HANDLING ACTIVITIES

Steven M. Miller
Carnegie Mellon University

Diane M. Strong
Carnegie Mellon University

A MODEL FOR EVALUATING THE PERFORMANCE OF OPERATIONAL LEVEL INFORMATION HANDLING ACTIVITIES

Steven M. Miller and Diane M. Strong
Graduate School of Industrial Administration
Carnegie Mellon University

ABSTRACT

This paper describes a modeling framework that will allow a manager to simulate and evaluate the performance of alternative designs of an operational level information handling process. Performance is measured in terms of the quality of the outputs of the process, the total flow time through the process, and the human resource time required to produce an output. The two major design options represented in the model are capabilities of computerized information systems used, and characteristics of quality control mechanisms within the information handling process. Based on the field study, we elaborate on why the problem of designing information flows in an office is difficult and we describe some of the complexities that need to be considered in a performance evaluation model. This paper presents a summary of the methodology for representing an information handling process, and an example of how the methodology can be used to address a design problem at the field site.

INTRODUCTION

Planned performance benefits from using computerized information systems are more likely to be achieved if better planning and decision making about the design of the computerized process can be done. This can only happen if design choices are clearly understood. This paper describes ongoing research that seeks to increase the understanding of tradeoffs made at the design stage of an information system. The design choices involve simultaneous consideration of the capabilities of a computer system and the work process into which the computer system will be embedded.

This research is motivated by an observed "real world" problem. In a particular firm, management plans to introduce a new computer-based information system into a work group within an office. The computer-based information system is designed to perform some tasks currently performed by people. It is evident that using the new system will change the way work is currently performed, but the details of how work will change and the implications of these changes are not readily apparent. In order to implement the new system with a minimum of disturbances to operations, management believes it is important to anticipate changes in how work will be performed.

The following section elaborates on the purpose of the model. The conceptual framework for the model is then presented. This includes a discussion on the difficulty of the design problem. Information handling process complexities that need to be considered to model the performance impacts of information system changes are also discussed. Following that, a summary of the methodology for representing the information handling process is presented. This is the formal way of describing information handling activities so that design choices can be specified and performance measures calculated. Then, the use of this methodology for an example design problem is described. Finally, the research efforts to date are summarized and the planned activities to complete this research are briefly described.

RESEARCH FOCUS

This research focuses on understanding real world information handling activities in the organization, and on developing a modeling framework that will facilitate analysis of design tradeoffs. The ultimate goal of this research is to develop a modeling framework that can be used by a manager to simulate and evaluate the performance of alternative information system designs. The simulation of alternatives is designed to be used before a new computer-based information system is implemented. Thus, the research takes a proactive or prospective focus.

The modeling framework will also be used as a research tool to investigate impacts of alternative designs for using computer-based information systems in an information handling process. As Cohen (1984) suggested, the development of computerized methods for assessing the performance of alternative designs in organizations is a good way to increase the rate of progress in the development of theories of how organizations operate. This paper focuses on describing the modeling of an information handling process and the practical use of the modeling framework as a decision aid to managers planning installations of computer systems into information handling processes.

Offices to be Modeled

This research concentrates on offices in which goals are well defined and there is a substantial degree of routinization in the activities required to meet the goals. While activities in such offices are often prescribed by well identified procedures, judgment and decision making are still required on a frequent basis during the execution of these activities (Suchman, 1983). The offices studied here support operational level decision making within a manufacturing enterprise. This particular field site performs order processing functions to provide manufacturing with the information needed to produce the product requested by a customer.

The information handling systems within these offices already make substantial use of computers for database management, record keeping, communications and other standard office functions. This research focuses on the usage of computers to automate decision making that heretofore has not been automated in the office (or the company). In this particular case, the new computer applications are expert systems. Expert systems will be used to structure parts of the information handling task that were difficult to perform with other available information technologies. The application of AI technologies expands the options available to system designers to improve the performance of the information system.

Performance Evaluation

The outputs of the model are performance measures for evaluating and comparing alternative designs of an information handling process. Performance is conceptualized in terms of the quality of the outputs, the time to produce an output, and the human resources required to produce an output from the system. Quality is measured as the percentage of acceptable outputs produced by the information system. This is a simplified view of quality, but it is a reasonable approximation. Flow time is measured as the average time to move an information object, such as an invoice, through the system. Human resource requirements are measured in terms of the number of labor hours required to process the information objects.

These three variables, quality levels, flow time, and labor requirements, were chosen as performance measures because they represent commonly cited expected benefits of automating, and they are also benefits that often fail to materialize (Markus, 1984). Thus, the focus of the evaluation is on expected benefits. The use of advanced computer systems is expected to improve the quality of the outputs of the process and to produce these outputs in less time. The use of advanced computer systems is also expected to reduce the direct labor resources required to operate the process.

Quality is measured in terms of the percentage of acceptable outputs and the other two are measured in terms of time. In principle, these could be converted into cost figures so that tradeoffs between the three benefits could be easily evaluated. However, the focus of this research is on understanding the basic performance characteristics of an information process and the changes in performance when advanced computer systems are incorporated into the process. When the functioning of an information handling process is understood well enough to achieve planned performance improvements from using computer systems, tradeoffs between different benefits and the costs of achieving these benefits can then be evaluated. Therefore, the initial research efforts will use direct measures of information process performance.

Design Choices

The design options represented in the model are capabilities of computerized information systems used in the process and characteristics of quality control mechanisms within the process. The alternative capabilities considered for computer systems are the comprehensiveness of the decision rules included in the system and the assistance provided to people performing quality control functions. The alternative characteristics considered for quality control mechanisms are their placement in the information handling process and their capabilities for generating, detecting, and correcting exceptions.

The focus of these design options is on quality control of exceptions. Exceptions are problems with objects processed in an information system that will become unacceptable outputs if not fixed. A major focus of this research is understanding the generation, detection, and handling of exceptions in the current information process at the field site, and how changes in the generation, detection, and handling of exceptions affect the performance of the information system. Efforts are focused here for two reasons. The first is that people performing information handling activities at the operational level spend a large portion of their time doing exception detection and correction. In a work group studied in detail at the field site, approximately two thirds of the people's time is spent in this way. Secondly, alternative capabilities of computerized information systems are expected to significantly affect the number of exceptions generated and the amount of time required to detect and handle an exception. The combination of these two reasons means that proposed changes to the design of an information process affecting the generation, detection, and correction of exceptions will have a significant impact on performance.

CONCEPTUAL FRAMEWORK FOR THE MODEL

The model described in the next section is based on a particular view of an information handling system, and the complexities of achieving performance improvements in such a system. This view and the literature supporting it are discussed in this section in terms of the following three elements.

- a network of information handling activities
- manual and computerized processors
- the design of a quality control system

A Network of Information Handling Activities

A typical operational level information handling process consists of multiple interdependent activities. The focus is on the inputs and outputs of information handling activities, not on the procedures or rules for performing an activity. This perspective of an information handling process is one that has emerged from the literature on organizations as information processing systems developed from the early work by March and Simon (1958) and Cyert and March (1963), extended by Galbraith (1973, 1977). Information outputs of one activity are inputs to another activity. Based on the information dependencies between activities, the activities form a network which is a representation of the information handling system.

When a computer system is embedded into a work process composed of a network of multiple interdependent activities, the effect of performance improvements at the activity in which a new computer system is used may not be apparent. Performance improvements for a single activity (local effects) do not necessarily translate into performance improvements for the entire work process (global effects).

Whether global improvements result depends on the relationship between the local activity and the outputs of the entire process. For example, if quality problems for the process are caused by quality problems at the local activity level, then increased computer use at that level is likely to yield improved quality for the entire process. However, other activities may cause quality problems that offset the improvements from increased computer use at the local level, then quality improvements for the entire activity are unlikely. A similar "local vs. global" argument applies to the reduction in flow time. To reduce flow time for the entire process, the activity for which the computer system is used must be on a critical path through the information handling process. That is, it must be a bottleneck activity in terms of flow time. Otherwise, reduced flow time at the local activity will not translate into reduced flow time for the entire process.

The local vs. global argument does not directly apply to human resource requirements. If fewer human resources are required for the local activity, then fewer human resources will be required for the entire process. This is because human resource requirements for the entire process are assumed to be the sum of the human resource requirements for each activity in the process.

Manual and Computerized Processors

Each activity comprising the information handling system is performed by a processor. Two types of processors are considered in this research, people and computers. The view of human behavior underlying this work is that a person is an information processor that is part of a complex information processing and decision making system within the organization. Therefore, an information handling process can also be thought of as a network comprised of both people and computer systems.

Although more computer systems continue to be incorporated into information handling systems, the view of an information handling system as a combination of manual and computerized processors continues to be relevant because people are still an important part of even highly computerized systems. The information flow between activities performed by computers and activities performed by people is an important aspect of the performance of the information handling process.

A Network of Quality Control Processes

The purpose of some of the activities in the information handling process is to perform quality control functions. These controls can be classified into three types; preventive controls, detective controls, and corrective controls (Mair, Wood, and Davis, 1978). Preventive controls reduce the frequency with which exceptions occur. Exceptions are problems within an information handling process that become unacceptable outputs if they are not fixed. Preventive controls are generally so embedded in the process that they are viewed as a normal part of the process. They are not separate activities within the process, but are part of the rules for performing activities.

Detective controls (exception detection) monitor the process and indicate when exceptions have occurred. Corrective controls (exception handling) fix the problems found by detective controls. Exceptions are generated by normal processing activities performed by either people or computers. Exceptions may also be generated by exception handling activities. Fixing exceptions is often a complex process that may introduce new exceptions into the system.

One motivation for explicitly representing activities that control the quality of the information flowing through the system comes from organizational literature. Organizational theorists, such as Carroll (1967) and Kickert (1980), use the model of closed loop feedback control as a framework for modeling the process of decision making in organizations. Another motivation comes from the accounting literature. Johnson, Leitch, and Neter (1981) explain that auditors, accountants, and system designers require information about the relative frequency of errors in an account, the size of these errors, and the distribution of these errors to conduct accounting audits. The authors elaborate:

> "System designers also need this knowledge for incorporating administrative and accounting controls into management information systems. The location and sophistication of error detection and correction procedures are a function of the expected frequency, magnitude, distribution, and likely causes of errors at each juncture in the transaction

processing cycle. Knowledge of frequency, magnitude, distributions, and possible causes of errors is therefore needed for the design of effective information systems."

A quality control system is a subsystem of the information handling system. As narrowly defined, a quality control system is the set of exception detection and exception handling activities throughout the entire information handling process that determine the quality of the final output of the process. A broader definition also includes activities that generate exceptions as part of the quality control system. When designing a quality control system, the broader definition helps to focus attention on designing procedures that reduce the number of exceptions generated, rather than focusing entirely on designing procedures for detecting and fixing exceptions.

A good process design matches the exception detection and handling activities to the exception generation characteristics of the computerized and manual processes. The lack of an appropriate match is hypothesized to be a major reason why expected computerization benefits are often not achieved. Increasing the use of computers changes the exceptions that are generated, both in terms of the number of exceptions generated and the type of controls that can detect these exceptions. Therefore, existing exception detection and handling activities must be adjusted when the use of computers is increased to maintain a match with the generation of exceptions.

A good design for a quality control system is one that produces a high percentage of acceptable outputs from the entire process, but does not require much extra flow time or human resources. A tradeoff exists between achieving increased quality and reduced flow time for a given technology. If generated exceptions are not detected and handled, there are quality problems. Therefore, to remove these quality problems, time must be spent in detection and handling activities which increases flow time. If the design of the technology and the work process can be altered, there are two possibilities for avoiding this tradeoff. One possibility is not to take the generated exceptions as given. For example, computer systems can be designed to accept a greater portion of objects with little or no increase in processing time. The other way to avoid the tradeoff is based on the local vs. global effects in a complex process. It may be possible to find places in the process to locally increase flow time without increasing the flow time through the entire process. At these "slack" points in the process, exception detection and handling can be done to increase quality without incurring a flow time penalty.

The flow time and human resource effects of a quality control system depend primarily on how much assistance the computer system provides with exception detection and handling. For exception detection activities, there must be some way to recognize whether the output produced by the computer or the manual process is acceptable. If this is difficult to determine, the detection activity will take longer to perform and will probably fail to detect some exceptions. A well designed computer system should provide some assistance in detecting exceptions. For example, some computer systems produce reports listing objects for which there are problems, whereas other computer systems simply list everything that was processed and people must search for exceptions. It is possible for manual detection and handling activities of computerized processes to take longer to perform than when doing the process manually. In this case, computerization may decrease flow time and increase human resource requirements, just the opposite of what was expected.

MODELING AN INFORMATION HANDLING PROCESS

Models of information handling processes have been developed in both the office information systems literature and the accounting literature. In the office information systems literature, offices have been modeled as information flow networks (Ellis and Nutt, 1980) and as Petri nets (Zisman, 1977). These research efforts have focused on the development of specification languages and software tools to facilitate the construction and analysis of models (See Ellis and Nutt (1980) and Bracchi and Pernici (1984) for a review of this literature.) The time to process information activities is often included in these models so that queueing theory or simulation can be used to analyze the flow of information (Nutt and Ricci, 1981). Exception handling is not a focus of these models. Feedback loops, rework, and mechanisms to monitor and control the quality of the outputs are not emphasized or explicitly considered.

In the accounting literature, information systems have been modeled as reliability networks (Cushing, 1974, Bodnar, 1975) and as stochastic processes (Yu and Neter, 1973). These models have given explicit consideration to representing the quality of the information flowing through the system since the goal of auditing is to determine whether quality outputs are being produced. (See Knechel (1983) for a review of this literature.) Cushing uses reliability theory to calculate the probability of a good output. Alternatively, Yu and Neter view the movement of an object, such as an order, through the information handling process as a Markov chain. Knechel (1985) combines characteristics of both of these approaches in his simulation modeling approach to evaluating quality controls. Little emphasis is placed on the time or resources required to produce quality outputs, although it is not completely ignored (Cushing, 1974).

The model described below borrows from both of these literatures. In particular, the core of the model is an information flow network similar in nature to models found in the office information systems literature. In addition to this basic model structure, changes in the quality of information as it flows through the network is modeled as a stochastic process. The description of an information handling process represented in the model is then simulated to compute performance measures for the process.

MODELING AN ACTIVITY NETWORK

The core of the model is a network of activities. The network is an acyclic directed graph[1] with a single starting node and one or more terminating nodes. For the operational level processes being modeled, this means that exception handling or re-work is not modeled as a feedback loop, but rather as a separate forward process. For many practical applications, modeling exception handling as a repetition of a previous computerized or manual process is not realistic. Although it may appear from observations that the process is repeated, the performance parameters of the process are probably different the second time through the process. This restriction to an acyclic directed graph greatly simplifies calculation of the performance measures and is not expected to restrict the capability to model exception detection and handling activities.

Information objects, such as orders or vouchers, flow through the network as they are processed by the activities. Objects are not necessarily processed by all of the activities in the network. The path of any object through the network may differ from other objects. As objects are processed by activities they change states. The state of an object is a summary of the effects of information processing on the object. Object states are represented as a vector of values. The state of an object is the only information stored about the object. Knowing the state of the object and its location in the network is sufficient to determine the next activity for that object. The history of the object in terms of previous activities is not needed. This memory-less property permits the processing of an object to be modeled as a Markov process.[2]

Three activity structures are the basic components of the model:[3]

- Transformation: one input arc and one output arc
- OR-split: one input arc, select one of several possible output arcs
- OR-join: several inputs arcs from an OR-split, one output arc

The simplest type of activity structure, called a transformation, is an activity with one input arc and one output arc. It models normal or exception handling processing of objects by people or computers. A branching structure, called an OR-split, has one input arc and multiple output arcs and models an exception detection activity. One output arc is selected for each object from

[1]An acyclic directed graph contains no cycles or loopes. It is not possible in an acyclic directed graph to start at a node and follow the directed arcs back to the same node. All arcs point from the starting node toward the terminating node(s).

[2]The memory-less property depends on how the states are defined. It is not a property inherent in a system being modeled. Defining the states so that the memory-less property applies will be easier in some systems than in others. For many information handling systems, a person can determine the next activity to perform from information that is part of the object. Thus, the states of the object is defined from information that is part of the object.

[3]The terms OR-split, OR-join, AND-split, and AND-join are taken from Ellis (1983).

several possible output arcs. Corresponding to the branching structure is a merging structure,[4] called an OR-join, which joins together paths that branched from an OR-split activity.

In the simple example shown in Figure 1, transformations are represented as rectangles and branches, and merges are represented by diamonds. These activity structures are labeled by capital letters. Between activities is a circle representing objects after one activity and before the next activity. These object states are labeled numerically.

In the example, a computer system performs a transformation (at A). The output of the computer system is manually checked for exceptions. The process then splits into two branches, one for outputs from the computer system that are detected as acceptable by the exception detection activity, and one for outputs that are detected to be unacceptable. An object follows one of the two paths, depending on whether an exception was found. For detected exceptions, there is an exception handling activity (at C) to fix the problem. Finally, the two paths are merged together to produce the final object state.[5] This example is purposely kept simple to illustrate concepts and is not intended to demonstrate all of the complexities of the information handling processes the model is designed to analyze. A more realistic example is discussed later in the paper.

MODELING QUALITY

The method for modeling quality is based on the earlier work of Yu and Neter (1973), which views each activity in an activity network as having some propensity for introducing errors or not eliminating them. Changes in the quality of an object as it moves through the activity network are viewed as a stochastic process using a Markov model. In the modeling framework described in this paper, each transformation activity, such as processing or exception handling, may change the quality of the object. Branches and merges do not change the quality. However, branching may be done based on the quality of the object, i.e. exception detection.

For the example shown in Figure 1, two quality values are considered: the information is acceptable or it is unacceptable. Suppose the process starts (at $state_0$) with unacceptable information. For example, the information has not yet been included in the order. The computer process (at A) is supposed to supply acceptable information. If it always supplied acceptable information for every object, the exception detection and handling processes (at B and C) would not be needed, and the final output state ($state_5$) would be acceptable outputs. However, such a perfect process is unlikely. The final output quality depends on the changes in quality at all of the transformations and on correct functioning of the exception detection activities.

Modeling Quality Changes (Transformations)

Changes in the quality values at each transformation are modeled by a matrix of probabilities. The probabilities in the matrix, labeled p_{ij}, represent the probability that an input of quality value i is transformed into an output of quality value j. For the example, with the two quality values acceptable and unacceptable, a transformation probability matrix looks like:

	To Quality Value	
From Quality Value	Acceptable	Unacceptable
Acceptable	p_{11}	p_{12}
Unacceptable	p_{21}	p_{22}

For example, p_{21} represents the probability that an object entering with quality value 2, unacceptable, is transformed into quality value 1, acceptable. The sum of the probabilities for each input quality value must be 1. That is, $\sum_j p_{ij} = 1$ *for each i.* This means that each input quality value must become some output quality value.

The numbers in the transition matrix represent the capabilities of the processor performing the activity. For example, p_{21} and p_{22} represent the capabilities of a processor to handle objects with

[4]A branching and merging structure pair to model the start and end of parallelism will be part of the full model, but will no be discussed in this paper. In this case, all output arcs are selected to model parallel activities operating on an object. The parallelism branching structure is called and AND-split and the merging structure is called an AND-join. Information technologies, such as database systems, support parallel activities within an information handling process.

[5]According to the model specification, the example could have terminated with two termination states (states 2 and 4) rather than merging into one state. Which is chosen depends on the information processing system being modeled and the desired calculations from the model.

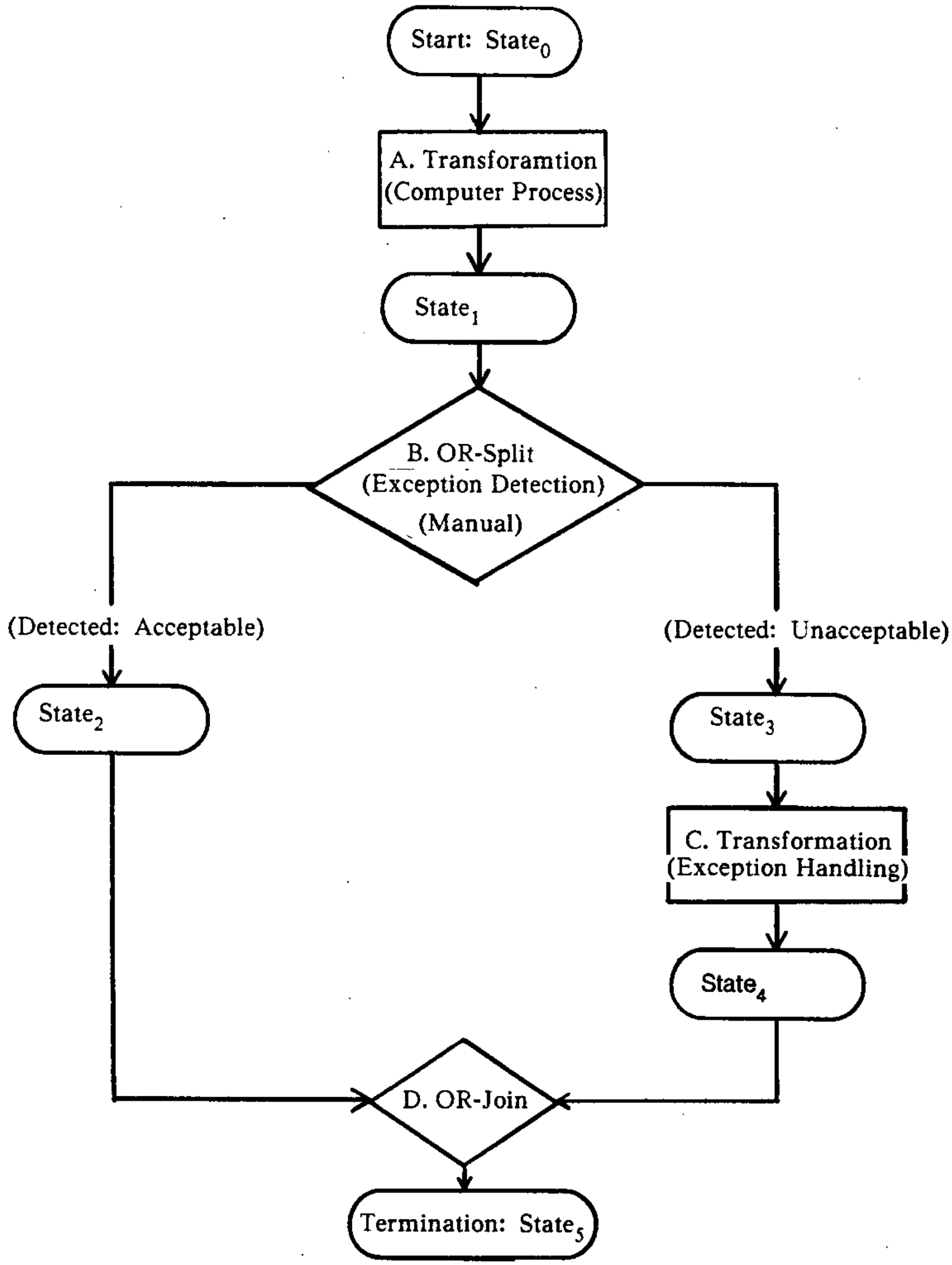

Figure 1: Activity Network

unacceptable quality values as inputs. The larger p_{21} is, the more capable the processor is at transforming unacceptable inputs into acceptable outputs. If the characteristics of the processor change, such as a switch from a person to a computer or from a traditional computer system to an expert system, the probabilities in the matrix should change.

Example transition matrices for the transformations in Figure 1, the computer process at A and the exception handling activity for unacceptable information at C, are shown in Table 2. The probabilities shown in the matrices are hypothetical, and are chosen only to demonstrate the type of phenomenon that can be represented.

Suppose the quality transformations for A, the computer process, are as follows:

	To Quality Value	
From Quality Value	Acceptable	Unacceptable
Acceptable	0.10	0.90
Unacceptable	0.10	0.90

The numbers mean that 10% of the outputs have information of acceptable quality and 90% have unacceptable information. Since each row is identical, the output of the computer process is independent of the quality value of the input to the process. (If the transformation matrix represented a manual process, it is unlikely that all of the rows in the matrix would be identical, since people would be expected to look at the input to check whether additional processing is required.)

Although it may seem unlikely that a computer system would produce acceptable outputs only 10% of the time, the situation is not uncommon in practice. For example, databases and decision rules in computer systems are often not kept up-to-date with the changing environment. Also, in any firm, the official policy modeled by the computer system may not represent actual practice. A typical case is the difference between standard lead times stored in a database and the actual time required to produce a product. A computer process producing only 10% acceptable outputs is a likely candidate for replacement by more advanced technology, such as an expert system that could make decisions based on rules that captured many of the contingencies to be considered.

Suppose the quality transformations at C, the exception handling activity for objects detected as having unacceptable information, are as follows:

	To Quality Value	
From Quality Value	Acceptable	Unacceptable
Acceptable	0.95	0.05
Unacceptable	0.90	0.10

The second row indicates that even if the detection process always functioned correctly, i.e. only unacceptable objects were sent on this path, this exception handling activity only fixes 90% of the unacceptable objects. The first row indicates that if acceptable objects are mistakenly sent on this path, this exception handling activity will convert 5% of these into unacceptable outputs, that is, the processor "fixes" problems that do not exist.

A perfectly functioning exception handling activity would have all ones in the first column and all zeros in other columns. This means that for any incoming quality value, the outgoing quality value is always acceptable. Since column 2 is not all zeros in the above matrix, there is a non-zero probability that some outputs of the exception handling activity will still not have acceptable quality.

Modeling Quality Detection (OR-split)

An exception detection activity, modeled by an OR-split activity structure, does not change the quality of an object. It sends each object to an exception handling activity or to the next process. However, an exception detection activity may not function correctly. That is, an incorrect output arc may be selected. In the example, the exception detection activity (at B) may fail to catch an exception and select the acceptable arc when the object actually had unacceptable information. Therefore, the quality transformation for an OR-split models type I and type II errors in the choice of output arcs, not quality changes in the object. The following table shows the type I and type II errors for an exception detection activities.

	Evaluated or Detected State	
True State	No Exception	Exception Exists
No Exception	Correct	Type I error
Exception Exists	Type II Error	Correct

The information in the above table is represented in the quality transformation matrix for an exception detection activity. The general form of the matrix, using the two quality values in the example, is as follows:

Quality Value	Arc Selected	
	Acceptable	Unacceptable
Acceptable	q_{11}	q_{12}
Unacceptable	q_{21}	q_{22}

The first subscript is the incoming quality value and the second subscript is the arc selected. Since the quality value does not change during exception detection, the outgoing quality value is not explicitly represented in the matrix.

The sum of the probabilities along any row must be one. That is, one arc must be selected for each incoming object. For example, any object arriving at the detection activity with acceptable quality will leave with acceptable quality along either the "detected acceptable" or the "detected unacceptable" arc. In the above matrix, q_{11} and q_{22}, are probabilities that the exception detection activity makes the correct choice. The probabilities, q_{12} and q_{21}, are the probabilities of type I and type II errors, respectively. A perfect detection activity is modeled by a matrix for which the diagonal elements are equal to unity and all other elements are zero, i.e., no type I and type II errors.

Suppose that for the example exception detection activity (at B), the following values are used:

Quality Value	Arc Selected	
	Acceptable	Unacceptable
Acceptable	0.90	0.10
Unacceptable	0.05	0.95

For acceptable inputs, 90% are recognized as acceptable while 10% are mistakenly classified as unacceptable (type I errors). The 10% of the acceptable inputs sent to the exception handling activity because of a less than perfect exception detection activity may be "fixed" and possibly become unacceptable. For unacceptable inputs, 5% of these exceptions are not caught (are classified as acceptable) and 95% are correctly classified as unacceptable. The 5% of the unacceptable inputs not sent to an exception handling activity will complete the activity with an unacceptable quality. These 5% are type II errors from the exception detection activity.

In summary, changes in, and detection of, the quality of information objects at activities are modeled by two types of probability matrices. For each transformation activity structure in the model (i.e., for each computerized or manual process and for each exception handling activity), a transition probability matrix represents the probabilities that the quality of the information object changes from particular input quality values to particular output quality values. For each OR-split activity structure in the model, a probability matrix represents the probabilities that information objects with particular input quality values are detected by an exception detection activity as having those quality values, or are mistakenly detected as having other quality values. Given these probability matrices for each activity in the network, matrix multiplication can be used to compute the probability of acceptable and unacceptable outputs at the termination state(s). This computation will be explained in more detail in the section on computing the performance measures.

Modeling Time

Two time figures are associated with each activity. One represents the time required to process a single information object through the activity. This is the elapsed time to perform the activity. The other represents the human resource time required to perform the activity. If the activity is performed by a computer system, the human resource time is likely to be zero, although some human resource time may be required. These two time figures for each activity provide information to compute the total human resource time required and the total processing time in the activity network.

In addition to this time information for each activity, information about the scheduling of activities is included in the model. Scheduling information, along with the processing time for activities, is used to compute the idle time between activities which is needed to compute flow time. Idle time is the time information objects wait in queues for the next activity to be performed.

The scheduling information needed in the model is information about when activities should be performed. For example, a computer system may only run as an overnight batch activity. This scheduling information must consider the processors used to perform the activities, since an appropriate processor, either human or computer, must be available to perform the activity when it is scheduled. For the particular information handling system being modeled in the initial version of the simulation, there is one human processor and multiple computer processors, one for each computerized activity. This models a scenario in which a person is responsible for all activities for a set of information objects. There are separate computer systems for each computerized activity. This is a typical way of assigning responsibility to people, although other ways such as specializing by activity are also used. Assuming separate computer systems means either that dedicated computer systems are used or multi-processing systems are used. One human processor and multiple computer processors means that activities performed by people cannot be scheduled in parallel, but computerized activities can be scheduled in parallel with each other and with manually performed activities. In this particular information handling system, each activity is assigned a particular day of the week when it is usually performed. This information is included in the model for each activity.

In summary, to capture timing information in the model so that flow time and human resource time performance measures can be computed, the following information is included in the model for each activity in the activity network:

- processing time for the activity
- human resource time for the activity
- time during the week when the activity should be scheduled

Computing the Performance Measures

The three performance measures, the quality of the outputs from the activity network, the average flow time through the activity network, and the average human resource time for manual processing in the activity network, will be computed given the input information. These inputs are:

- the activity network
- the quality transformation matrix for each activity
- the processing time for each activity
- the human resource time for each activity
- scheduling information for the activities

Simulation of the activity network will be used to compute the three performance measures because it is a general and flexible technique that will not restrict extensions to the modeling framework. In particular, computation of the flow time through the network using queueing theory methods is not analytically tractable for general scheduling rules. Simulation has been recommended by other researchers for analyzing the performance of information handling systems (see, for example, Knechel (1985) in the auditing literature and Nutt and Ricci (1981) in the office information systems literature). Before discussing the simulation, analytic procedures for computing the quality of the outputs and the time required from human resources will be presented. These procedures will be used on simple cases to verify that the simulation is computing these measures correctly. Discussion of these procedures provides some insight into how the information in the model is used to compute performance measures.

Analytic Computations for Quality - Since quality was modeled as a Markov chain, the quality at the terminal state can be computed using matrix multiplication. For this method of computing quality, a state is a vector of probability values. (The representation of a state is different when using simulation to compute quality.) The values in the state vector represent the probability that an object has each possible quality value. For the quality values in the example, [0.40, 0.60] represents a 0.40 probability of having acceptable quality and a 0.60 probability of having unacceptable quality. On a path through the activity network that all objects follow, the sum of the probabilities in the state vector is 1. After an OR-split, the sum of the probabilities in a state vector along any one of the arcs is less than or equal to 1. The sum of the probabilities in a state vector represents the probability that an object follows that path.

For each activity structure, the matrix computations are different. For transformations, the transition probability matrix describes how input quality values are transformed into output quality values. Given the input state and the probability matrix of transformations, the output state can be computed, as follows: $s^T_1 P = s^T_0$ where s^T_1 is a row vector representing the input state, s^T_0 is a row vector representing the output state, and P is the transition matrix.

For OR-split activities, a matrix multiplication is done for each output arc resulting in an output state vector for each arc. A different transformation matrix is used for each arc. This matrix is a diagonal matrix whose diagonal elements are the elements from the column of the probability matrix for the OR-split corresponding to that arc. For the example presented earlier, there are two diagonal matrices:

From Quality Value	To Quality Value Acceptable	Unacceptable	
Acceptable	q_{11}=0.90	0.00	
Unacceptable	0.00	q_{21}=0.05	*for the detected acceptable arc*

From Quality Value	To Quality Value Acceptable	Unacceptable	
Acceptable	q_{12}=0.10	0.00	
Unacceptable	0.00	q_{22}=0.95	*for the detected unacceptable arc*

The input state vector is multiplied by each of these diagonal matrices to obtain an output state vector for each arc from the exception detection activity.

For OR-join activities, the corresponding elements in all of the input state vectors are summed. For example, element 1 in the output state vector is the sum of element 1 in all of the input state vectors.

Moving through a network of activities, such as the one shown in Figure 1 using the calculations specified above, the quality at the output state of each activity can be computed until the terminal state is reached. This produces a state vector of probabilities for each terminal node in the network. These final state vectors represent the quality of the objects produced from the network. For the example presented earlier, the result would be the probability of an acceptable output from the network and the probability of an unacceptable output from the network.

There is a load analysis calculation that can easily be performed given the state vectors from the Markov chain analysis for computing the quality of the outputs. If the average number of information objects to be processed for some time period can be estimated, the probabilities in the state vectors can be multiplied by this estimate to determine the load on the information handling process. For example, suppose an exception detection activity sends 40% of the information objects to an exception handling activity for correction of problems. If the input to the activity network is 1,000 objects per month, then a person performing the exception handling activity must be able to correct 400 objects per month. This simple load analysis helps to understand the impacts of the quality probabilities.

Analytic Computations for Time - The human resource time for the network is relatively easy to compute analytically since it does not depend on idle time. The average human resource time required to process an information object is computed by summing the human resource time for each activity weighted by the probability that the object is processed through that activity. That is, $h=\Sigma_i (h_i \times p_i)$ where h is the average human resource time, h_i is the human resource time for activity i, and p_i is the probability that an information object requires processing by activity i. p_i is the sum of the probabilities in the state vector representing the input to activity i. This sum of probabilities is one for any arc in the network that all objects follow and is less than one for paths taken by only selected objects.

This computation produces the average human resource time required to handle a single information object which can be multiplied by the average number of objects to be processed in a time period to estimate the human resource time required for the time period. Alternatively, the focus of the analysis can be placed on particular activities within the information handling process. For this analysis, the human resource time for an activity is multiplied by the average number of objects that must be processed through that activity. This produces an estimate of the total human resource time required to perform a particular activity.

Using the same calculation procedure given for human resource time, the average processing time for the information handling process can be computed. That is, the processing time for each activity weighted by the probability that an object is processed by the activity can be summed over all activities in the network. This calculation represents the total time an object is worked on while it is moving through the information handling network, and does not include idle time. In this research, the focus is on flow time rather than processing time. Processing time is only of interest because it provides a bound on the feasible reduction in flow time. An analysis of idle time is important because idle time for an object provides an opportunity to perform quality control activities to increase quality without affecting flow time.

Simulation of the Activity Network - To simulate the processing of information objects by the activity network, information about the arrival of information objects is needed. A real stream of arrivals may be used or this stream may be represented by an arrival distribution. These are the inputs that drive the simulation model. As initially set up, the simulation will model the processing of information objects that are the responsibility of one human processor. For the particular field site, this means the orders for one product line.

Each information object enters the activity network with a particular quality value. For example, each object may start with unacceptable quality because later processing will add information needed to make the object of acceptable quality. The state vector in the simulation model indicates the quality value for an information object, not the probabilities for each quality value. That is, each object has a particular quality value. For example, the object either has acceptable or unacceptable quality, not a 40% probability of being acceptable.

For each different activity structure, transformation, OR-split, and OR-join, the simulation performs different functions. At transformations, the simulation stochastically selects the output quality value for an object given its input quality value and the transition matrix of probabilities for the transformation activity. It accumulates the processing time and the human resource time associated with the activity, and also records the idle time since the last activity finished for this object. Finally, it sends the object to the next activity in the network.

For OR-split activity structures, the simulation stochastically selects an output arc for the object based on its input quality value and the matrix of probabilities associated with the activity. This output arc determines the next activity in the network to which the object will be sent. The simulation accumulates the processing time and human resource time associated with the activity and also records the idle time since the last activity finished for this object.

For OR-join activity structures, no processing is required from the simulation. These structures are included in the activity network to indicate the end of separate branches, but no quality changes or processing time are associated with them. These structures are needed in the Markov chain analysis to indicate the consolidation of probability state values from the branches being joined.

The information objects are processed at each activity when the activity is scheduled to be performed on a first-come, first-served basis except for a small set of objects designated as priority objects. Activities performed by people are scheduled more frequently to handle any waiting priority objects. That is, priority objects move through the activity network as fast as possible given limitations from the scheduling of computerized activities.

BUILDING AND USING THE MODEL

In this section, two issues about the practical use of the modeling concepts in a field setting will be discussed. These two issues are: (1) formulation of a design problem in a field setting in terms of the modeling concepts, and (2) collection of the data to specify the model parameters.

An Example Design Problem

The field site is a manufacturing firm which assembles complex, high value-added electronics products to customer specifications. The office studied serves as an interface between sales and manufacturing groups. Its primary function is order administration. The inputs to the information handling process studied are customer orders collected by the sales organization. The outputs are orders with additional information needed by the manufacturing organization to assemble the products.

Two of the primary responsibilities of the office being studied are to ensure that correct engineering and sourcing information are attached to the order. The office has other order administration responsibilities, but this example will focus on the process of supplying and verifying the correctness of these two pieces of information. Engineering and sourcing information are not part of an order when it arrives from sales, but must be included with an order before it is sent to manufacturing. The firm is in the process of designing, developing, and implementing expert systems to supply these two pieces of information. The design problem examined here is deciding on the capabilities of these expert systems and the design of quality control activities to ensure that correct engineering and sourcing information are supplied to manufacturing.

Engineering information is a specification of how the components ordered by a customer should be assembled to form a functioning product. There are generally multiple solutions to this problem, some of which are better than others, but there is no agreed upon optimal solution. A key issue is to avoid supplying infeasible or unbuildable engineering specifications, rather than searching for a "best" solution. The process of supplying engineering information is a complex decision process requiring extensive product knowledge about how components can be assembled together. Each decision rule is relatively simple (e.g. component A can only be part of the product if component B is included), but the set of decision rules is large and each decision rule is applicable in only some situations. This problem has been called the "configuration problem" and at least one firm routinely uses an expert system for configuring products.[6] However, solution of the configuration problem in each firm is unique because the solution depends on the particular products.

Sourcing information is a specification of which of the firm's plants should supply each subassembly of the order. In the firm, subassemblies of a product are assembled at different plants and multiple plants are capable of supplying a particular subassembly. An acceptable solution to the problem depends on the engineering information (how the product is to be assembled), the capabilities of the plants, and the status of the plants. The engineering information is used to determine which components must be assembled in the same plant and information about the plants is used to determine which plants can assemble those components.

The firm operates in a dynamic environment. Each year many new products are introduced. This means that the decision rules for supplying acceptable engineering and sourcing information must be continually updated to include new product knowledge. The extensive knowledge base required to make acceptable engineering and sourcing decisions and the rate of change of the knowledge base makes these decisions candidates for expert systems.

Formulating the Model for the Example

The modeling framework is used to evaluate alternative options for using expert systems to

[6] See Scown(1985) for a case study of Digital Equipment Corporation's expert system for configuring computer systems.

supply engineering and sourcing information within the order processing system. First, order processing for these two decisions is represented as it is before using expert systems. This provides a baseline scenario with which to compare design options for using expert systems. Next, the design options for using expert systems are represented in the model to form descriptions of alternative scenarios for supplying engineering and sourcing information. Then, the baseline and alternative scenarios are compared in terms of the three performance measures, quality of the resulting orders, flow time, and human resource time.

A key part of representing an information handling process in terms of the model is selecting the quality values to be modeled. For this example, four values are used. They are formed from combinations of acceptable and unacceptable information from the engineering and sourcing decisions.

1. both engineering and sourcing information acceptable
2. engineering unacceptable, sourcing acceptable
3. engineering acceptable, sourcing unacceptable
4. both unacceptable

These quality values define the structure of the quality transition matrices. Each transition matrix is a 4 x 4 matrix representing the probabilities of transitioning from each of the four quality values in the input state to each of the four quality values in the output state.

The other key part of representing an information handling process in terms of the model is specifying the activity network. For this example, the structure of the activity network is the same for the baseline and alternative scenarios. What is varied between scenarios is the probability matrices for quality changes for each activity in the activity network, the processing time for these activities, and the human resource time for these activities. This equivalent structure is not always the case and the modeling framework can be used to compare processes whose activity networks differ. In some situations the information handling process is significantly restructured when installing computer systems. This is done to avoid automating poor processes. For this field site, process restructuring has already been done in preparation for a more automated process. This restructured process as it functions before the installation of expert system is the process represented in the baseline scenario.

The activity network for this design example is shown in Figure 2. The activity network shows two transformations at the beginning, one to supply engineering information and one to supply sourcing information. These two transformations are followed by an exception detection activity to find problems in the information produced by the first two transformations, and an exception handling activity to fix detected problems.

The first transformation, supplying engineering information, is performed manually in the baseline scenario and by an expert system in the alternative scenarios. The second transformation, supplying sourcing information, is performed by a conventional computer system in the baseline scenario and by an expert system in alternative scenarios. The conventional computer system for supplying sourcing information produces a large portion of unacceptable sourcing decisions because the system does not use the engineering information as an input. This means that the sourcing decision is made on a component-by-component basis independent of knowledge about which components must be assembled together.

For both the baseline and alternative scenarios, exception detection and handling after the two transformations is performed manually using information supplied by the transformations. In the baseline scenario, there is implicitly an exception detection and handling process built into the first transformation, since the technicians manually performing this transformation check their own work and fix any problems found. Production of quality outputs is incorporated into the transformation (quality assurance) rather than explicitly including a separate quality control activity in the process.

The alternative scenarios represent different management decisions about the use of expert systems in ordering processing. Two management decisions about using expert systems will be represented. The first is the amount of expertise to build into the expert system, i.e. how

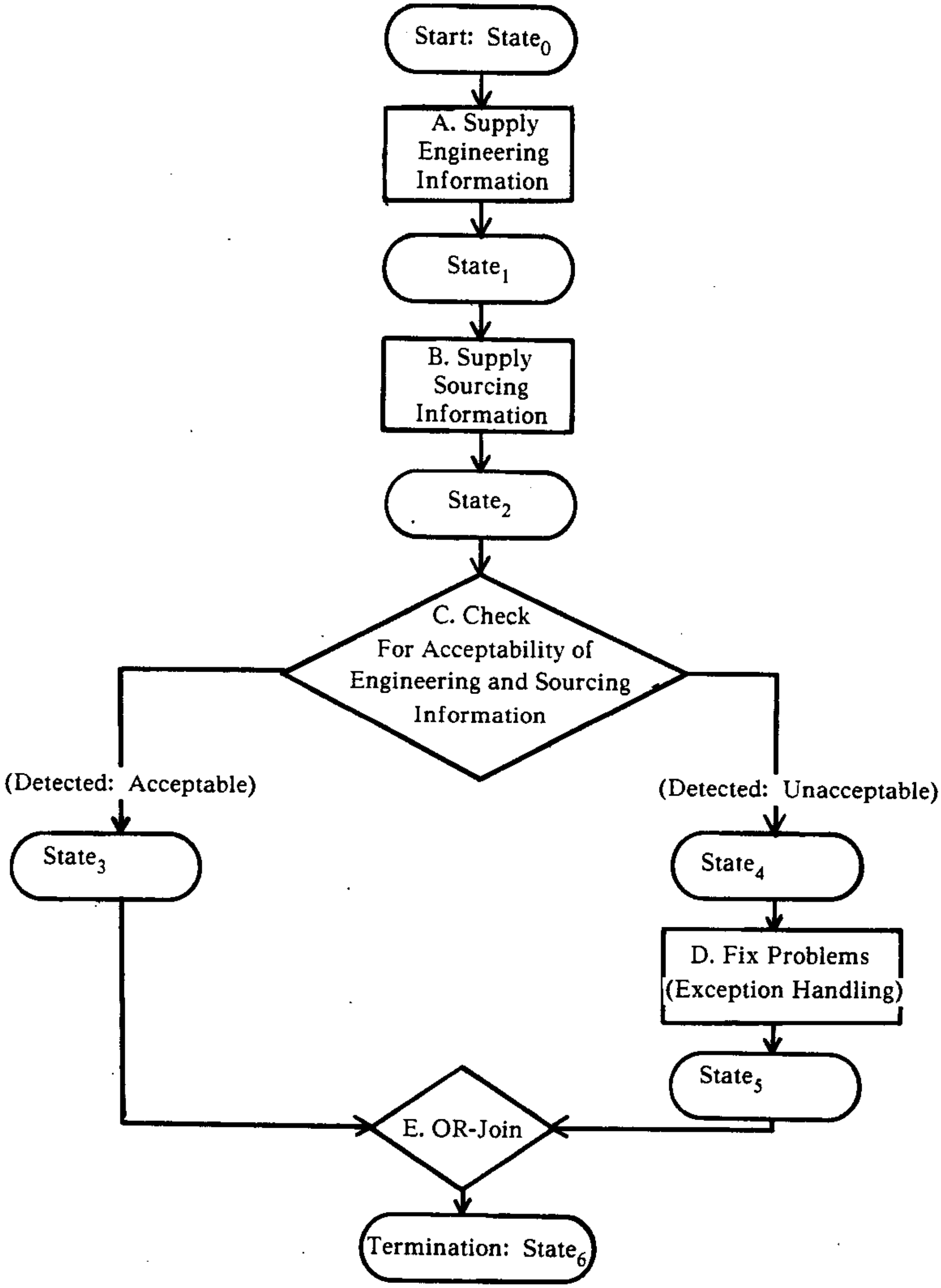

Figure 2. Partial Activity Network for Order Processing

comprehensive the decision making rules will be. The second is the assistance provided by the expert system with exception detection and handling. Both of these are choices about the functionality to be designed into the expert systems that are expected to affect the performance measures for the order handling process.

The comprehensiveness of the decision rules represents a tradeoff between supplying the information manually and developing and maintaining rules in the expert system. Some types of orders are rare and may be complex. It may be easier to process these orders manually than to develop and maintain rules to handle these cases. The comprehensiveness of the decision rules in an expert system is measured as the percentage of orders a system can handle correctly.[7] The modeling framework evaluates the performance impacts of different choices for the percentage of orders handled by an expert system.

The percentage of orders handled correctly is represented in the model in terms of the quality transition matrix for an expert system. Orders that cannot be handled by the expert system become unacceptable outputs of the transformation. Also, including more rules in the system to handle a higher percentage of the orders may affect the time to process an order through the expert system. This situation is represented in the model by increasing the processing time for the transformation.

The assistance provided by an expert system for exception detection and handling is functionality included in the expert system to help people find exceptions and correct them. For this design problem, exceptions are orders with unacceptable engineering or sourcing information. This functionality could take several forms including exception reports listing orders not correctly processed, information about what caused orders not to be processed, a trace facility to demonstrate the decision rules used in processing an order, and an interactive facility in which the expert system functions as a decision support tool to help a person supply acceptable information. For example, if the expert system for supplying sourcing information encountered a subassembly in an order for which it could not determine a plant to supply the subassembly, the expert system could produce a report describing the order, the particular subassembly it could not process, and its reason for not processing the subassembly (e.g. the subassembly is not in its database).

This assistance from an expert system is represented in the model in terms of parameters for the exception detection and handling activities. Additional assistance from an expert system should make exception detection and handling activities function with higher quality and take less time. However, the expert system may take longer to process orders with this increased functionality. This would be represented in the model by increasing the processing time at the activity performed by the expert system.

Analysis of the Baseline and Alternative Scenarios

The activity network and the quality values define the overall structure of the model. To compute the three performance measures, the parameters for each activity must be specified. These parameters are the quality transition matrix, the processing time, and the human resource time for each activity. Specific model parameters, based on investigations at the field site, are used below to demonstrate how baseline and alternative scenarios can be analyzed. However, since data collection for the parameter values is in initial stages, the parameter values used below are only rough estimates of actual and planned operations.

First, consider the transformation to add engineering information to an order. In the baseline scenario, this process takes an average of 45 minutes for each order and 45 minutes of human resource time. For cases in which the incoming quality of the engineering information is unacceptable (the usual case), the people producing the information supply acceptable information 85% of the time and unacceptable information 15% of the time. If, for some reason, the incoming order already has acceptable engineering information (a rare case), this information is reviewed by the person and 5% of these orders are changed to have unacceptable

[7] The full version of the model will include facilities to model multiple types of information objects (e.g., types of orders). The percentage of orders handled correctly by an expert system can be specified for each type of order. Then, when the mix of orders changes, the percentage of orders handled does not need to be adjusted. Multiple types of information objects will not be discussed in this paper).

engineering information. These quality changes are represented in the following matrix:

Input Quality Value	Output Quality Value 1	2	3	4
1 Both Acceptable	.95	.05	.00	.00
2 Only Eng. Unacc.	.85	.15	.00	.00
3 Only Sourcing Unacc.	.00	.00	.95	.05
4 Both Unacceptable	.00	.00	.85	.15

The usual case is that orders enter the activity network with both unacceptable engineering and sourcing information, since this information has not yet been supplied. This is represented in the last row of the matrix.

An alternative scenario for this transformation uses an expert system which takes 2 minutes to process each order and requires no human resource time. The expert system does not consider the input quality value of the order; it produces 95% acceptable engineering information independent of the input quality value. These quality changes are represented in the following matrix:

Input Quality Value	Output Quality Value 1	2	3	4
1 Both Acceptable	.95	.05	.00	.00
2 Only Eng. Unacc.	.95	.05	.00	.00
3 Only Sourcing Unacc.	.00	.00	.95	.05
4 Both Unacceptable	.00	.00	.95	.05

The 95% acceptable outputs represent the percentage of orders handled by the expert system. This is a goal of the expert system development, but since this goal may not be achieved, other values should be tested in the analysis. Varying this percentage and the time to process an order through the expert system are the alternative scenarios modeled for this transformation.

By examining the parameters for the two scenarios for this transformation, it is easy to see that the expert system increases the quality (from 85% acceptable to 95% acceptable) while reducing the processing time (from 45 minutes to 2 minutes) and the human resource time (from 45 minutes to zero). However, suppose the expert system can only handle 70% of the orders. (This is a likely case during initial use of the system.) Then, the effect of the expert system on the performance measures depends on the parameters for the exception detection and handling activities. The processing time and human resource time for these activities could be large with only minor improvements in quality.

Similar modeling considerations apply to the transformation for supplying sourcing information. The conventional computer system in the baseline scenario produces acceptable sourcing information for only 20% of the orders. It does not consider engineering information as an input nor does it examine the order to determine whether acceptable sourcing information has already been supplied for the order. Therefore, the quality of the sourcing information produced is independent of the quality value of the input. This means that each row of the matrix will have 20% and 80% values in the appropriate columns as follows:

Input Quality Value	Output Quality Value 1	2	3	4
1. Both Acceptable	.20	.00	.80	.00
2. Only Eng. Unacc.	.00	.20	.00	.80
3. Only Sourcing Unacc.	.20	.00	.80	.00
4. Both Unacceptable	.00	.20	.00	.80

The expert system modeled in the alternative scenarios for the sourcing transformation takes as an input the output of the expert system for supplying engineering information. Thus, the quality of the sourcing information produced by the sourcing expert system is influenced by the capabilities of the expert system for supplying engineering information. If the input engineering information is unacceptable rather than acceptable, it is more likely that the sourcing information produced will be unacceptable. This condition is shown in the following quality transition matrix:

Input Quality Value	Output Quality Value 1	2	3	4
1. Both Acceptable	.95	.00	.05	.00
2. Only Eng. Unacc.	.00	.30	.00	.70
3. Only Sourcing Unacc.	.95	.00	.05	.00
4. Both Unacceptable	.00	.30	.00	.70

The numbers in this matrix indicate that the expert system produces acceptable sourcing information 95% of the time if the engineering information is acceptable, but only 30% of the time if the engineering information is not acceptable. The 30% figure indicates that some of the problems with the engineering information do not affect the sourcing decision.

Since both the baseline and alternative scenarios for this transformation are performed by computer systems, the human resource time for both is zero. The processing time is expected to be larger for the expert system than for the conventional computer system, since more functionality is included in the system. For sourcing information, the use of an expert system improves quality with some increase in processing time. As in the first transformation, the percentage of orders the expert system can handle correctly will be varied in alternative scenarios.

Exception detection and handling activities follow the two transformations in the activity network. An analysis of the quality results from the first two transformations provides information about the need for these quality control activities. These results assume that all orders enter with unacceptable information for both engineering and sourcing. In the baseline case, the exception detection and handling activities are clearly needed because only 17% of the orders processed through the first two transformations have acceptable information for both engineering and sourcing. This is primarily caused by the poor performance of the conventional computer system for sourcing. For the alternative scenario with expert systems, 90% of the orders have acceptable information for both. If the engineering expert system only produces 70% acceptable orders, then only 67% of the orders processed through both transformations will have acceptable information.

These three scenarios, 17%, 90%, and 67% acceptable orders, provide different situations for quality control. In general, when most of the orders are exceptions (e.g. the case of only 17% acceptable orders), exception detection is not very important, because all of the orders will be reprocessed manually. Efficient exception handling to reprocess these orders is more critical than exception detection. When few of the orders are exceptions (e.g. the 90% case), exception detection is important because orders will not be reprocessed unless exceptions are indicated. Reliable exception detection is more important than efficient exception handling. For a case between these two extremes (e.g. the 67% case), both reliable detection and efficient handling are important.

Several different options for exception detection and handling can be represented in the model to handle these cases. One modeling option is no exception detection and handling. Then the output from the network is the same as the output after the two transformations. This can be modeled without changing the activity network by an exception detection activity that takes no time and sends all orders to the acceptable branch.

A second option is an exception handling activity with no exception detection activity. This is represented in the activity network by an exception detection activity that sends all orders to the unacceptable branch. This would be used to represent a case in which all orders are manually re-processed. A third option is one exception detection activity followed by an exception handling activity. This is the case assumed in the activity network shown in Figure 2.

A final option is one exception detection activity followed by multiple exception handling activities. This represents different exception handling activities for different types of problems. For example, the detection activity could be designed to determine whether there is a problem with the engineering information, the sourcing information, or with both, and then send the order to an exception handling activity for the problem detected. This situation is not represented in the activity network for the field design problem but it can be handled in the modeling framework.

These exception detection and handling options demonstrate that many alternative order processing scenarios could be represented in the model and analyzed. In addition to the options described above, the quality, processing time, and human resource time for these quality control activities can be varied in the alternative scenarios. The computerized simulation model should be able to analyze the many interesting options.

Data Collection and Availability

Building a model using the concepts presented in this paper requires careful attention to the problem of collecting data for the model parameters. There are two general problems: keeping the data requirements to a manageable size and collecting accurate data.

Data requirements

As noted above, the activity network and the quality values define the structure of the model.

If there are a large number of activities in the network and more than just a few quality values, the data required to specify the model parameters may exceed practical limits on the amount of data that can be collected. This is an important consideration when formulating the structure of the model.

The four recommendations given below will help to keep the data requirements to a manageable size. The suggestions are:

- Focus on a portion of the entire information handling process.
- Focus the level of detail in the activity network at the decision making level.
- Focus on critical quality problems.
- Focus on key alternative scenarios to model.

These recommendations focus on the problem-solving function of the modeling framework, which is to provide information for better decision making about the use of computer systems in an information handling process, and not simply to build a detailed description of the process. The model structure will be focused on the key problem areas to be investigated.

The first recommendation is needed because information handling processes in organizations involve many activities. For example, the information handling system for order processing starts with information from customers and ends when the product has been shipped and invoiced. Focusing on a portion of an information handling process is usually necessary to keep data requirements manageable.

When formulating the activity network, activities will be aggregated to a level that corresponds to the decision making level. Defining the work represented by a single activity is part of formulating the activity network. Activities will not be modeled in more detail than is needed to support the decisions to be made. To make these concepts more concrete, consider a series of computer systems (or a series of computer programs) that process orders with no manual intervention between systems. They could be modeled as a series of separate activities or as one activity. If parameters for individual systems within the series will not be changed in the analysis (i.e. decisions are not being made about individual systems within the series), then modeling the series as one activity will be considered. This second recommendation reduces the number of activities in the activity network as much as possible which reduces the number of parameters to specify.

The focus on critical quality problems limits the quality values to those that are necessary to study the problem. Critical quality problems are problems that will cause costly disruptions or customer dissatisfaction if the problem is not caught before the information object exits the activity network. A problem that can be easily caught and fixed after the object has been processed is not a critical quality problem. Using a minimum number of quality values makes the data collection of the probability matrix values possible. In the example from the field, acceptable and unacceptable values were used for two pieces of information. This focuses on the critical pieces of information and aggregates all critical problems with each piece of information into one value called "unacceptable."

In addition to the first three recommendations about setting up the model structure, the number of alternatives to be investigated will be kept small. Each alternative requires additional data collection of parameter values. It is easy to think of many possible alternatives to test, but the alternatives will focus on the options that are important for management decision making.

Data collection

Collection of accurate data for the model is a time consuming process. Accuracy in this context means valid and reliable data, not necessarily data that are precise to many digits. Accurate data are a valid representation of the processing in an information handling system.

Initial data can be collected by interviews and from any existing documentation of the work process. However, cognitive psychology literature has documented the unreliability of retrospective data (Ericsson and Simon, 1984). This means that observation of the work process must be done to ensure accurate data. Also multiple people must be observed to determine whether there are important differences between people. Examples of important differences are 1) processing times or quality percentages that differ by orders of magnitude between people performing the same activities and 2) activity networks that differ substantially between people performing the same tasks.

In the field investigation for this research, data collection for the baseline scenario started with existing documentation of the work process and interviews to clarify the documented information and to determine the structure for the model (the activity network and the quality values). These data are being verified by work observation. The parameter values for the model are also being collected by work observation with some verification from archival records. The details of these data collection procedures and what was learned from using them will be described in a forthcoming paper.

Collection of data for the alternative scenarios represents a different problem since these activities cannot be observed. In the firm studied, data for the design options for the expert systems is being collected from existing documentation and from system designers. The information about the percentage of orders to be handled, the processing time, and the functions to be provided for assistance with quality control are documented in functional specifications for the systems. A difficult part not yet addressed in this research is translating the functions for assistance with quality control into time and quality parameters for the quality control activities.

An important part of testing the alternative scenarios is to investigate the effect of not achieving the planned design values. For example, suppose an expert system can only handle 70% of the orders rather than the design value of 95%. The modeling framework can be used to test the sensitivity of the performance measures to the design values.

SUMMARY AND CONCLUSIONS

This paper has presented a model for representing an information handling process. The model simulates an information handling process and produces performance measures on the quality of the outputs from the process, the time to produce an output, and the human resources required to produce an output. These performance measures are used to evaluate alternative designs for an information handling process. These alternative designs model different capabilities of computerized information systems used in the process and different characteristics of quality control mechanisms with the process.

The purpose of developing the model is to provide a tool for studying the relationship between designs choices in an information handling process and the performance of the process. The research efforts have included substantial field work to ensure that model is addressing "real-world" problems and to demonstrate the feasibility and usefulness of this approach. The previous section of this paper provided an example of the use of the model for a design problem in a firm and discussed the issues involved in collecting data for the model.

As noted in the introduction, this paper describes research in progress. The next steps planned for this research effort are:

- data collection,
- design and development of a baseline simulation model,
- validation of the baseline simulation model
- modeling of alternative scenarios, and
- use of the simulation model to compare and evaluate alternative scenarios.

Collection of data for the model parameters is in process. When data collection is completed, a simulation model of the information handling process in the field will be developed. This model will be validated against the current operations of the process. This validated simulation model is the baseline against which alternative designs for the process will be compared.

Then several alternative scenarios for the information handling process will be represented in the simulation model. The alternatives to be modeled have been specified in an experimental design, but the data for the model parameters have not yet been collected. The simulation experiments specified in the experimental design will be run and the results from these runs of the simulation model will be used to compare and evaluate the alternative scenarios.

Future avenues of research that may be pursued involve both continued field work and extensions to the simulation model. The field work would involve following the changes to the information handling process due to using expert systems and following the use of the simulation model to assist with planning for these changes. Simulation model extensions would focus on adding more dynamic features to the model. Of particular interest is adjusting parameters over time to study learning in the information handling process. Also, dynamics to study end-of-period phenomena may be pursued. In addition to these extensions, the model may be extended to include data and methods for evaluating tradeoffs between alternative designs of an information handling process.

REFERENCES

Bodnar, G. "Reliability Modeling of Internal Control System," *The Accounting Review,* Volume L, Number 4, October 1975, pp. 747-757.

Bracchi, G. and Pernici, B. "The Design Requirements of Office Systems," *ACM Transactions on Office Information Systems,* Volume 2, Number 2, April 1984, pp. 151-170.

Carroll, D. C. "On the Structure of Operational Control Systems," In *Operations Research and the Design of Management Information Systems,* J. F. Pierce, Jr. (ed.) Technical Association of the Pulp and Paper Industry, 1967

Cohen, M. D. "Artificial Intelligence and the Dynamic Performance of Organizational Designs," January 1984.

Cushing, B. E. "A Mathematical Approach to the Analysis and Design of Internal Control Systems," *The Accounting Review,* Volume XLIX, Number 1, January 1974, pp. 24-41.

Cyert, R. M. and March, J. G. *A Behavioral Theory of the Firm,* Prentice-Hall, New York, New York, 1963.

Ellis, C. A. "Formal and Informal Models of Office Activity," 1963.

Ellis, C. A. and Nutt, G. J. "Office Information Systems and Computer Science," *Computing Surveys,* Volume 12, Number 1, March 1980, pp. 27-60.

Ericsson, K. A. and Simon, H. A. *Protocol Analysis: Verbal Reports as Data,* The MIT Press, Cambridge, Massachusetts, 1984.

Galbraith, J. R. *The Design of Complex Organizations,* Addison-Wesley, Reading Massachusetts, 1973.

Galbraith, J. R. *Organization Design,* Addison-Wesley, Reading, Massachusetts, 1977.

Johnson, J. R., Leitch, R. A., and Neter, J. "Characteristics of Errors in Accounts Receivable and Inventory Audits," *The Accounting Review,* Volume LVI, Number 2, April 1981, pp. 270-291.

Kickert, W. J. M. *Organisation of Decision-Making: A Systems-Theoretical Approach,* North-Holland, Amsterdam, Holland, 1980.

Knechel, W. R. "The Use of Quantitative Models in the Review and Evaluation of Internal Control: A Survey and Review," *Journal of Accounting Literature,* Volume 2, 1983, pp. 205-219.

Knechel, W. R. "A Simulation Model for Evaluating Accounting System Reliability," *Auditing: A Journal of Theory and Practice,* Volume 4, Number 2, Spring 1985, pp. 38-62.

Mair, W. C., Wood, D. R. and Keagle, W. D. *Computer Control and Audit,* The Institute of Internal Auditors, Inc., New York, New York, 1978.

March, J. G. and Simon, H. A. *Organizations,* John Wiley & Sons, New York, New York, 1958.

Markus, M. L. *Systems in Organizations: Bugs and Features,* Pitman Publishing Inc., New York, New York, 1984.

Nutt, G. J. and Ricci, P. A. "Quinault: An Office Modeling System," *Computer,* Volume 14, Number 5, May 1981, pp. 41-57.

Scown, S. J. *The Artificial Intelligence Experience: An Introduction,* Digital Equipment Corporation, 1985.

Suchman, L. A. "Office Procedure as Practical Action: Models of Work and System Design." *ACM Transactions on Office Information Systems,* Volume 1, Number 4, October 1983, pp. 320-328.

Yu, S. and Neter, J. "A Stochastic Model of the Internal Control System," *Journal of Accounting Research,* Volume 11, Number 2, Autumn 1973, pp. 273-295.

Zisman, M. D. *Representation, Specification and Automation of Office Procedures,* Doctoral Dissertation, University of Pennsylvania, 1977.