MODELING DATA AND PROCESS QUALITY IN MULTI-INPUT, MULTI-OUTPUT INFORMATION SYSTEMS*

DONALD P. BALLOU AND HAROLD L. PAZER

Management Science Department, School of Business, State University of New York, Albany, New York 12222

This paper presents a general model to assess the impact of data and process quality upon the outputs of multi-user information-decision systems. The data flow/data processing quality control model is designed to address several dimensions of data quality at the collection, input, processing and output stages.

Starting from a data flow diagram of the type used in structured analysis, the model yields a representation of possible errors in multiple intermediate and final outputs in terms of input and process error functions. The model generates expressions for the possible magnitudes of errors in selected outputs. This is accomplished using a recursive-type algorithm which traces systematically the propagation and alteration of various errors. These error expressions can be used to analyze the impact that alternative quality control procedures would have on the selected outputs.

The paper concludes with a discussion of the tractability of the model for various types of information systems as well as an application to a representative scenario.
(INFORMATION SYSTEMS—MANAGEMENT; RELIABILITY—QUALITY CONTROL; COMPUTERS—SYSTEMS DESIGN)

1. Introduction

Information systems designed to respond to a broad spectrum of needs continue to proliferate rapidly. However, one of the salient characteristics of such systems is that often a gap develops between user expectations and the performance of the information technology actually implemented. Significant among the factors accounting for this situation is the data quality component (Bailey 1983).

Systems analysts have long recognized the importance of the accuracy dimension of data quality, and consequently extensive and often elaborate edit checks and controls have been developed and successfully implemented (Martin 1973). Nevertheless, management appears to perceive a need for further improvements in data quality. In a study of the attitudes of top and middle managers toward a choice between increased data quantity vs. enhanced data quality, a clear preference (nine to one) was expressed for enhanced data quality (Adams 1973). The continuing importance of this issue together with some of its various facets is discussed by Brodie (1980) and Morey (1982).

A number of important contributions have been made toward the development of a general model to assess the impact of data quality in information systems. These have appeared primarily in the accounting literature and, not surprisingly, have focused on the impact of various errors and controls on ending financial balances. A reliability model was proposed by Cushing (1974) and extended by Bodner (1975), Stratton (1981) and Ishikawa (1975). Yu and Neter (1973) suggest a markovian model, Burns and Loebbecke (1975) use simulation while Hamlen (1980) presents a chance-constrained mixed integer program. These models focused on transaction processing and/or financial reporting systems. Motivating examples included payroll systems,

raw material purchasing and accounting systems, and cash receipts posting and processing systems. The emphasis of these papers was on single, self-contained systems.

The model described in this paper is in some respects more general than the accounting reliability models and in other respects more limited. In any case it has a different emphasis in that it focuses on the impact that various processing activities, especially in a model-based environment, would have on errors in the data. Errors can be amplified or diminished by processing, or remain unchanged (Alonso 1968). The model accommodates multi-system environments characterized by inputs from multiple sources and by various applications of the data. A more detailed comparison of our model and the accounting reliability models can be found in §3.

Increasingly the data residing in or generated by a unit are viewed as an organizational resource to be managed and to be used by anyone with a legitimate need. This sharing of the data resource introduces problems and issues not encountered in traditional transaction processing operations. Furthermore, it has become apparent that data quality is a relative rather than an absolute term and can most usefully be defined in the context of end use (cf. Davis 1974, p. 456). Nevertheless various users of the same data can have markedly different data quality requirements, a fact that substantially complicates any analysis of possible data quality enhancements. This problem is especially significant if different managerial levels are involved. In distributed environments the same types of data may be gathered by different units within the organization but be represented in different forms. Data replication in these environments creates other problems (Bernstein and Goodman 1981, Hansen 1983, Martin 1981).

Purpose of Paper

This paper presents a model which under certain conditions can trace the propagation and alteration of errors in data items within information systems. It also handles the impact of faulty processing on data items. It produces expressions for the magnitudes of errors in selected terminal outputs. The model requires that the data items be in numeric form. (It does not handle alphanumeric data.) Although the concepts are presented in the context of computer-based information systems, the ideas are relevant for manual and mixed systems as well.

Many of the processes and quantitative procedures required to generate desired information use inputs that are themselves outputs from other processes. (Bailey 1983 estimates that 70% of all inputs are of this type.) Thus, errors in the "original" data items may undergo a series of alterations, and the processes themselves can introduce errors. Hence the magnitudes of errors in final or terminal outputs depend in a complicated way on various errors and their interactions. Furthermore, it is not obvious what efforts should be made to improve the quality of the various processing procedures and data sources so as to reduce these terminal errors. For example, in a multi-input, multi-output environment, it is quite possible that errors in a data set could impact only one output, but in a significant way, whereas errors in another data set could influence several outputs but to a lesser degree. If resource limitations preclude enhancing the data quality of both sets, which one should receive priority? To make such a decision, a mechanism is required which can identify the impact that errors in original data sources and error introduced through processing would have on selected outputs. The error magnitude expressions that result from our model enable the analyst to study the impact of alternative quality control strategies. Examination of these expressions will identify likely candidates for quality enhancement. (For example, data values which are processed in a manner that dampens errors are of less concern than those whose processing magnifies errors.)

Characteristics of Model

The model presented in this paper assumes that certain outputs have been identified for data quality analysis, i.e., a determination needs to be made as to the possible magnitudes of errors in certain output values. A data flow diagram of the type used in structured analysis (De Marco 1978) would then be developed. This diagram would include all data sources and all processing nodes or activities required to generate the selected set of terminal outputs. The model produces an expression for possible errors in the selected outputs in terms of possible errors in the original data items together with possible errors introduced by intermediate processing.

In many cases the data flows required for each of the selected terminal outputs intermingle, i.e., they are interdependent. Thus, errors in one data source affect several outputs to varying degrees. The model makes it possible to evaluate the overall impact that errors at one location have on the entire set of outputs. The impact on multiple outputs of improving data quality at various points of the data flow diagram can be analyzed by varying the expressions that represent the error magnitudes.

The focus of our model is process rather than data oriented; that is, the concern is with the transformations or manipulations that errors in data experience as the data values proceed through the various systems. This emphasis is one of the factors that differentiates our model from the accounting reliability models. The model can accommodate transaction processing systems (provided the data are in numeric form), but it is probably of greater value in the context of assessing the quality of outputs used in managerial decision making. Much data used in this context are inherently in error (e.g. forecasts). Furthermore the data inputs come from a variety of sources, both internal and external. Decisions are based in part on information produced from these data. It is important for the decision maker to be aware of the possible magnitude of errors in numerical information. The data flow/data processing model provides such estimates. Also the quantitative models used to support managerial decision making are more complex than the quantitative procedures encountered in transaction processing systems, and hence have the potential to transform errors in ways that are not obvious. In addition, although errors in transaction processing systems can have serious consequences, if faulty information used in decision making results in incorrect decisions, the impact on the well-being of the organization can be substantial.

The next section contains a description of the model. §3 includes an analysis of the tractability of the model for various types of information systems. The concluding section is a representative scenario.

2. Data Flow/Data Processing Model

The network model presented in this section can be used to analyze the flow and processing of data within an organization so as to evaluate the impact that deficiencies in data quality and errors associated with the processing or manipulation of data would have on the various outputs. The links in the network represent the flow of data, either source or processed, between units or components of the organization. The nodes symbolize in a general sense the manipulation or processing of data. Before presenting the general model, some preliminary concepts are examined and notation introduced.

Basic Concepts and Notation

The fundamental building block of the network that represents data flow and data processing activities in the organization is depicted in Figure 1. In general, the model assumes multiple inputs with values represented by $x_1, x_2, \ldots, x_n$ which are processed in some manner to yield an output y. The processing function $F = F(x_1, x_2, \ldots, x_n)$ can be quite general. For example, F could be the operation that stores data item x in
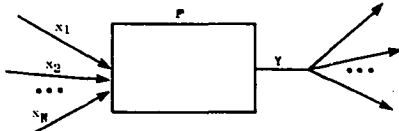
FIGURE 1. The Input-Processing-Output Module.

a data base, y in this case representing the value as stored. The branch on the output y in Figure 1 indicates that the output value can itself serve as input to several other processes.

A processing block should be used wherever data values can experience change. Thus a processing block would be used to represent the manipulation of data by analytic models. Another important use is to model control functions whose purpose is to correct data items, both input and output. (See Cushing 1974 for an analysis of the role of control functions in transaction processing systems.) In fact, one of the primary purposes of the model is to allow the user to explore the data quality implications of alternative placement of the control blocks and to analyze alternative levels of processing control within these blocks. An additional aspect of the processing blocks is that they are not restricted to modeling computer-based processing. Any information system is a composite of computer-based and human activities. The processing block should be used to model both types.

In a network many modules such as displayed in Figure 1 will be present. If F_i represents one such block, then the inputs are denoted by $x_{i,1}, x_{i,2}, \ldots, x_{i,n_i}$, and the output by y_i.

Each of the input data items x_i can be in error. These errors can be modelled using differential notation, that is, if x_i is the value of a data item and dx_i represents the error in this value, then $x_i + dx_i$ is the correct value. This interpretation assumes that the data values are either numeric or in coded form. The model can handle ordinal data (dx_i is simply the difference between the recorded and correct ordinal values) and scale data that are dichotomous. If a data item is subject to multiple types of errors, then dx_i represents the aggregate error, i.e., dx_i equals the algebraic sum of all the errors in the data item.

Although the differential is used to represent or model any deficiency in data quality, it is subject to differing interpretations depending on the particular dimension of data quality involved. Four dimensions that the model can address are: accuracy (the recorded value is in conformity with the actual value), timeliness (the recorded value is not out of date), completeness (all values for a certain variable are recorded), and consistency (the representation of the data value is the same in all cases). The accuracy dimension is the most straightforward and is merely the difference between the correct value and that actually used. The timeliness interpretation is similar. A stored value, or any data item, that has become outdated is in error in that it differs from the current (correct) value. The differential equals the difference. Completeness can also be handled in a satisfactory manner. For example, if a data item is missing, it can be assigned the value "0" in which case dx_i represents the correct (missing) value. Finally, lack of consistency implies that two or more representation schemes are present. One of these should be chosen to be the standard, which implies that dx_i is the difference between the standard value and the value used.

The above appears to assume that dx_i can be assigned a known specific value. This of course is not the case in practice. If the dx_i were known, the correct values would be used. For our purposes dx_i represents the (unknown) value of a variable; the model presented below can be used to study the implications of various values for the dx_i to determine the impact of potential errors.
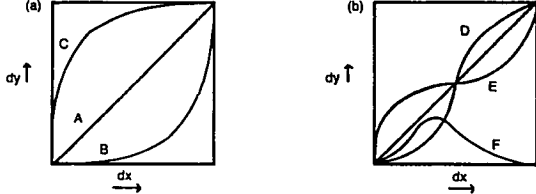
FIGURE 2. Representative Response Functions.

The error in the output y, represented by dy, depends not only on the data errors $dx_1, dx_2, \ldots, dx_n$ but also on the data values used, namely $x_1, x_2, \ldots, x_n$. (The function $y = 1/(x_1 - x_2)$ clearly illustrates this fact.) Thus the error in output resulting from errors in inputs can be represented symbolically by:

$$G(x_1, x_2, \ldots, x_n; dx_1, dx_2, \ldots, dx_n) = G(\mathbf{x}; \mathbf{dx}). \tag{1}$$

The processing blocks can impact data quality in a variety of ways. The purpose of control blocks is, of course, to enhance data quality. Certain activities, such as the averaging of data values, tend to minimize the impact of errors in input data. Other activities exacerbate input errors.

Figure 2 shows a set of frequently encountered error response functions relating dy to dx for constant x.

Case A is a linear relationship. If the slope is less than one, the process will dampen input errors while for a slope greater than one input errors will be amplified.

Case B represents a process which is insensitive to small input errors but highly sensitive to large errors.

Case C corresponds to a process which is intolerant of even small input errors.

Case D is representative of a process which exhibits its primary sensitivity to input errors in the vicinity of some threshold value.

Case E corresponds to a process which approximates a step function. In this case, the first "step" is encountered for small input errors and the second for large errors.

Case F represents a control process designed to detect and eliminate large input errors.

Even if the data items contain no errors ($dx_i = 0$ for all i), the value y can still be incorrect because of a mistake made in processing. (For example, suppose the correct process should have been to compute $(x_1 + x_2) * x_3$; instead, $x_1 + x_2 * x_3$ was computed.) To account for this kind of error, a term dP (for error in processing) can be added to (1) yielding:

$$dy = G(\mathbf{x}; \mathbf{dx}) + dP(\mathbf{x}). \tag{2}$$

In some cases the processing error will be independent of the values being processed; in others it will be value dependent.

In general the determination of G is a highly difficult problem. Also G and/or its derivatives can be discontinuous. For example, suppose that the F in question represents application of the simplex algorithm to a linear programming problem. In this case the values $x_1, \ldots, x_n$ would be the coefficients of both the objective function and the constraints and also the constant values. The output y in this case would be a vector consisting of the optimizing solution values, and dy would be in terms of a suitably chosen metric applied to the solution vectors. Clearly G, a vector-valued function, is highly discontinuous (although it will be piecewise constant). For further analysis in the information-flow network it is desirable to know the values for G over a continuum. For a specific choice of $x_1, \ldots, x_n, dx_1, \ldots, dx_n$, G can be determined easily. (Simply evaluate the process at $\mathbf{x}$ and at $\mathbf{x} + \mathbf{dx}$ and compute the difference.)

However, it may be possible to form a reasonable approximation to G over a continuum using interpolation techniques applied to values generated from various combinations of $\mathbf{x}$ and $\mathbf{dx}$. This possibility is currently being investigated by the authors. A tractable expression for G will be considered in greater detail below.

Data Flow / Data Processing Model

This subsection presents the model designed to permit the analysis of the impact of varying levels of the data quality and processing control on the selected outputs. Such information would be used, for example, to determine where greater data quality control should be exercised.

The basic relationship for the ith processing block is given by: $dy_i = G(\mathbf{x}(i); \mathbf{dx}(i)) + dP_i$. As mentioned above the determination of G in the general case is difficult at best. Furthermore, the process or algorithm described below for constructing the network-based model in the general case would require notation of a highly cumbersome sort. However, if certain continuity and differentiability assumptions hold for the F_i, then the function G takes a form that permits a more complete and tractable development of the model. We shall now pursue the model in this case, not only because it constitutes an important subset of the more general case, but also because it illustrates the procedures that should be followed once all the error functions G have been determined.

If F is continuous with continuous first partials, then to a first approximation the component in the error dy arising from errors in the input data is given by the differential

$$dy = \sum_{i=1}^{n} \frac{\partial F(\mathbf{x})}{\partial x_i} \, dx_i = G(\mathbf{x}; \mathbf{dx}). \tag{3}$$

As an example of the use of (3), suppose that the process involves averaging n values $x_1, \ldots, x_n$, i.e., $F(x_1, \ldots, x_n) = \sum_{i=1}^{n} x_i / n$. Then

$$dy = \frac{1}{n} \sum_{i=1}^{n} dx_i.$$

Thus under these smoothness assumptions on F, (2) takes the form

$$dy = \sum_{i=1}^{n} \frac{\partial F(\mathbf{x})}{\partial x_i} \, dx_i + dP. \tag{4}$$

It should be kept in mind that the vector $\mathbf{x}$ above represents the data values actually processed, and hence for our purposes these values can be viewed as fixed, specific quantities. The unknown and hence variable quantities of interest to us are the error terms $dx_1, dx_2, \ldots, dx_n, dP$.

Often the output y obtained by processing a set of inputs is itself an input to one or more processes. Thus, there can be a complicated intertwining of data flows, and hence errors in the original entry level data, which will be referred to as "primitive" data, may undergo several mutations and influence terminal or final outputs in unforeseen ways. We now describe a procedure or algorithm for tracing the ultimate impact of data errors on final outputs.

This algorithm assumes that the network of data flows and processing activities has been specified using techniques such as structured analysis (cf. DeMarco 1978). Thus, for the system under consideration, all primitive data elements have been identified, as have all inputs to each of the processing activities. The algorithmic-type procedure now described enables one to analyze error propagation and to determine the impact that errors in primitive data items and errors in processing would have on a given final or terminal output value.

Step 1. The initial step is to specify those terminal values, call them T_1, $T_2, \ldots, T_m$, that are to be analyzed. For each T_i the processing function F_i and the inputs $x_{i1}, x_{i2}, \ldots$ required by F_i should be identified. Then the error in T_i can be expressed by

$$dT_i = \sum_{j=1}^{n_i} \frac{\partial F_i(\mathbf{x}_i)}{\partial x_{ij}} dx_{ij} + dP_i. \tag{5}$$

As before the term dP_i represents the possibility of introducing an error through faulty processing. In the next section issues related to the determination of the dP_i will be discussed.

Equation (5), $i = 1, 2, \ldots, m$, can be expressed in matrix notation as follows: $dT = A_1(dX)_1 + (dP)_1$. Here dT is a column vector of the dT_i, $(dX)_1$ is a column vector consisting of the dx_{ij} $\forall i, j$, and $(dP)_1$ is a column vector of the dP_i. The components of the matrix A_i will be 0 or else terms of the form $\partial F(\mathbf{x}_i)/\partial x_{ij}$.

Step 2. The next step is to identify each x_{ij} with differentials appearing as a component of $(dX)_1$ which is itself an output of some process F and is such that no other term with differential appearing in $(dX)_1$ is functionally dependent upon it. Form a new column vector $(dX)_2$ by replacing in $(dX)_1$ certain of the dx_{ij}. More specifically, if x_{ij} meets the above conditions, then replace dx_{ij} with the differentials of those variables which when processed yield the x_{ij} in question, provided the replacement differentials do not already appear in $(dX)_1$.

For example, suppose that $(dX)_1 = (\ldots, dx_{i1}, dx_{i2}, \ldots)^T$, and further suppose that (i) x_{i1} is dependent upon x_{j1} and x_{j2} and (ii) for no term x_{km} with differential appearing in $(dX)_1$ is it true that x_{km} is functionally dependent upon x_{i1}. Then dx_{i1} should be replaced in $(dX)_2$ with dx_{j1} and dx_{j2}, provided neither term already appears in $(dX)_1$.

Once $(dX)_2$ has been determined, construct the matrix A_2 and column vector $(dP)_2$ such that $(dX)_1 = A_2(dX)_2 + (dP)_2$.

Step 3. Apply the process described in Step 2 to $(dX)_i$, $i = 2, 3, \ldots$, to form $(dX)_{i+1}$, A_{i+1} and $(dP)_{i+1}$. Continue until $(dX)_{i+1} = (dX)_f$ consists entirely of the differentials of primitive data elements.

Step 4. To express the terminal output vector dT in terms of the primitive data items, recursively replace $(dX)_i$ with $A_{i+1}(dX)_{i+1} + (dP)_{i+1}, i = 1, 2, \ldots, f - 1$.

Illustrative Application

The concepts and algorithm presented above are now illustrated by means of an example. The information network postulated is displayed in Figure 3. This system has five different categories of "original" or source data (a single value of each type is represented generically by $x_{11}, x_{22}, x_{32}, x_{41}$ and x_{42}). Four different types of outputs are generated ($y_1, y_2, y_3,$ and y_4). Except for y_2, each of these outputs is in turn input
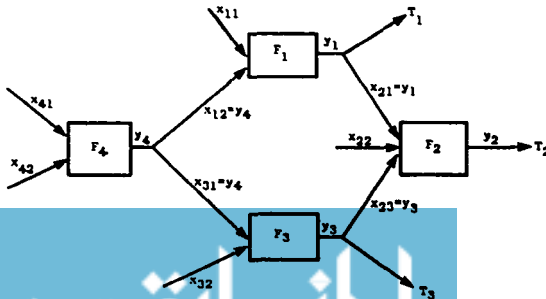

FIGURE 3. Typical Information Flow Network for Illustrative Purposes.

to some process. Three of the outputs have been identified for error magnitude analysis. This example involves both convergent and divergent flows.

$$d\mathbf{T} = (dT_1, dT_2, dT_3)^T,$$

$$(dX)_1 = (dx_{11}, dx_{12}, dx_{21}, dx_{22}, dx_{23}, dx_{31}, dx_{32})^T, \qquad (dP)_1 = (dP_1, dP_2, dP_3),$$

$$A_1 = \begin{bmatrix} \dfrac{\partial F_1}{\partial x_{11}} & \dfrac{\partial F_1}{\partial x_{12}} & 0 & 0 & 0 & 0 & 0 \\[2mm] 0 & 0 & \dfrac{\partial F_2}{\partial x_{21}} & \dfrac{\partial F_2}{\partial x_{22}} & \dfrac{\partial F_2}{\partial x_{23}} & 0 & 0 \\[2mm] 0 & 0 & 0 & 0 & 0 & \dfrac{\partial F_3}{\partial x_{31}} & \dfrac{\partial F_3}{\partial x_{32}} \end{bmatrix},$$

$$dT = A_1(dX)_1 + (dP)_1.$$

The quantities x_{21} and x_{23} are the terms which satisfy the conditions of Step 2. Since the replacement differentials dx_{11}, dx_{12} and dx_{31}, dx_{32} already appear in $(dX)_1$,

$$(dX)_2 = (dx_{11}, dx_{12}, dx_{22}, dx_{31}, dx_{32})^T; \qquad (dP)_2 = (0, 0, dP_1, 0, dP_3, 0, 0)^T;$$

$$A_2 = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ \dfrac{\partial F_1}{\partial x_{11}} & \dfrac{\partial F_1}{\partial x_{12}} & 0 & 0 & 0 \\[2mm] 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & \dfrac{\partial F_3}{\partial x_{31}} & \dfrac{\partial F_3}{\partial x_{32}} \\[2mm] 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}.$$

For Step 3 the terms in $(dX)_2$ which should be replaced are dx_{12} and dx_{31}. Thus

$$(dX)_3 = (dx_{11}, dx_{41}, dx_{42}, dx_{22}, dx_{32})^T; \qquad (dP)_3 = (0, dP_4, 0, dP_4, 0)^T;$$

$$A_3 = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & \dfrac{\partial F_4}{\partial x_{41}} & \dfrac{\partial F_4}{\partial x_{42}} & 0 & 0 \\[2mm] 0 & 0 & 0 & 1 & 0 \\ 0 & \dfrac{\partial F_4}{\partial x_{41}} & \dfrac{\partial F_4}{\partial x_{42}} & 0 & 0 \\[2mm] 0 & 0 & 0 & 0 & 1 \end{bmatrix}.$$

Then $(dT) = A_1 A_2 A_3 (dX)_3 + A_1 A_2 (dP)_3 + A_1 (dP)_2 + (dP)_1$.

This illustrative example highlights several features of the data flow/data processing model. First, the dependence of the terminal outputs (dT) on the primitive data items can be highly convoluted. Second, the outputs are especially sensitive to the nature of the processing activities, as the impact of errors in both primitive data values and processing is governed by appropriate rates of change (partial derivatives) of the various processing activities. Thus the processing activities can either diminish or accentuate any errors that may be introduced into the system. This has obvious implications for systems design activities.

3. Implementation Considerations

In this section we examine several facets of implementation. For the data flow/data processing model to be the value to system designers and users of information systems, it must be possible to implement the model with reasonable effort. Some general observations on practicality are followed by an examination of implementation issues first for transaction processing systems and then for model-based, decision-oriented systems.

One of the major arguments for the tractability of the model is that the identification of the required data flows and data processing activities often is accomplished independently of the needs of this model via commonly used analysis and design procedures. In fact, the structure of the data flows and data processing is essentially the data flow diagram of structured analysis. Thus in many cases once the outputs to be analyzed have been determined, the first step in developing the data flow-data processing model can be straightforward.

It remains to consider how to identify the F's and how the various error terms and rates of change should be handled. In the material that follows, it should be kept in mind that the purpose of the model is to determine the impact that errors could have on various outputs, i.e., the concern is with the possible magnitudes for the dT_i. In particular interest often involves how large the dT_i could be with the idea that if the maximal values are unacceptable, then an analysis should be conducted to determine how best to reduce the magnitudes. The expressions for the dT_i can be used to analyze the impact of reducing the magnitudes of the dx_i and the dP_i, or if necessary the system can be redesigned to avoid F's that are especially prone to error magnification. Although this is similar to traditional sensitivity analysis, the difference lies in having analytic expressions for the dT_i which would guide identification of alternative quality control options.

Implementation Issues for Transaction Processing Systems

As indicated above, most research activity in this field has concentrated on transaction processing systems. Application of the data flow/data processing model is relatively straightforward for systems that process numerical data (e.g., financial systems). In general, these systems can be characterized by large quantities of data and relatively simple processing (summing data items, adjusting balances, etc.). The F's for such operations can be determined and expressed easily, and they are continuous. (The F's consist of various combinations of arithmetic operations.) Thus the required partial derivatives can be obtained.

Most of the data items found in transaction processing files are correct with a small percentage in error (Johnson et al. 1981). The error expression for a specified terminal output involves the dx_i from a variety of data sources. For each file the dx_i for the data items for that file can be handled in two ways. The first approach involves simulation and is in the spirit of the work by Burns and Loebbecke (1975). If the error distribution (percentage in error and pattern of errors) for the file is known, then the dx_i can be randomly assigned values to reflect this distribution (most dx_i will be zero). If the error characteristics are not known, then the implications of various possible error distributions can be explored.

An alternative approach is to make a judgement as to the most likely maximum possible error for data items of a certain type. This value would then be used for all dx_i for that file. (This is reasonable in that edit checks usually limit the size of errors.) This approach is simple and requires less information about the data than the other, but it is clearly less precise.

Usually little information is available regarding the dP_i. However, this error could be one of two types. For stable systems, any sizable processing errors would have been

observed and corrected. However, smaller, more subtle errors could remain. To examine the implications of this case, small, judgementally determined values for dP_i should be used. The other situation involves infrequent processing mistakes (such as mounting the wrong tape) but ones that could have a serious impact. In such situations various large values for the dP_i, judgementally selected, should be used to explore the potential impact.

Implementation Issues for Model-Based, Decision-Oriented Systems

The data and processing characteristics of many model-based information systems are opposite to those of transaction-oriented systems. The number of data items involved is considerably smaller. Furthermore, these items often include forecast and judgementally determined values, both of which are inherently imprecise to a degree that is difficult to determine. In addition the processing functions F can be considerably more complex and possibly discontinuous. Nevertheless the data flow/data processing model has the potential to be of greatest value in these situations, for the information generated by model-based systems is often used in decision processes that can have a significant impact on the organization. This point was amplified at the end of §1.

Any data items used in a model-based environment that originated in transaction processing systems can be handled as described in the above subsection. The other data items, including those from external sources and those determined judgementally, can be treated in a manner analogous to PERT. The "best" estimate is used as the x value and the "optimistic" and "pessimistic" values specify the dx. In this way a bound on the possible errors in the outputs can be obtained.

The processing errors can be addressed as in the above subsection. However, for model-based systems there is an additional complication; the model being used may be inappropriate. (It could be that for the sake of simplicity, a linear model is used whereas a nonlinear one is more appropriate.) Various values for dP should be used to explore the impact that this type of processing error has on terminal outputs.

In a model-based environment the processing functions fall into one of two categories: those specified by a symbolic expression and those incorporating an algorithm. If the process can be described by an expression of some type—formula, series, etc.,—then determination and evaluation of the partial derivatives in (4) presents no difficulty. The situation is more complicated if the process requires an algorithm. However, in this case the values of the function at the data point and at selected points (determined by the dx_i values) along each of the coordinate axes can be used to compute approximations to the partials.

As is the case with most models, implementation issues can be addressed more readily if the system being modeled is relatively simple and if the information requirements are not extensive. This desirable framework can be achieved by limiting the number of terminal outputs that are to be analyzed and by not pushing the identification of the data flows to their original sources in all cases. Modeling will always require judicious judgements and appropriate simplifications. In the final analysis the tractability of this model will be dependent on the skill of the implementor.

Comparison of Accounting Reliability Models and Data Flow/Data Processing Model

The accounting reliability models and our model tend to complement each other. The former can handle any combination of data (numeric, alphanumeric, alphabetic), whereas ours is limited to numerical systems. A major focus of our model is on how processing may amplify or dampen various errors; the accounting reliability models

concentrate on whether or not data items are in error. The emphasis of our model makes it appropriate for managerial decision-making processes that are model based. However, a major strength of the accounting reliability models is that they do not require analytical expressions for the processing activities. (From this it follows that they cannot evaluate the impact of processing on errors.) The accounting reliability models tend to be linear systems (Cushing 1974 focuses primarily on a single processing stage). The model presented above is designed to accommodate multiple inputs which can interact through various processing stages to produce multiple outputs.

4. Representative Scenario

The models proposed in the preceding sections are sufficiently flexible in scope to allow either detailed or macro modeling of a broad range of multi-input multi-output information decision systems. Figure 4 is presented as an illustration of this capability for a centralized computer operation. It could represent, for example, the collection of data concerning dealer inventories in three regions which are then inputted to the system and processed. The three final outputs could represent information to be used for operational control, managerial control, and strategic planning.

At the most detailed level of analysis, the collecting operation can be modeled separately for each of the three regions. The error magnitudes or distributions can be separately specified for data inputs from each region, and collection functions may differ in response to these errors (e.g., input data may be double checked in one region but not in the others). The block used to consolidate these three flows can be used to incorporate completeness and consistency dimensions.

The differing characteristics of multiple input channels in terms of accuracy and timeliness can be explicitly modeled with the consolidation block once again used to include errors relating the completeness and consistency.

Similar but parallel processing of the data by a number of individuals can be modeled to represent differing response functions to input errors (e.g., an experienced employee may be able to recognize certain types of inconsistencies in the data) and differing probabilities of introducing additional errors during processing. In the example, two consolidation blocks are employed, each with potentially separate completeness and consistency error functions to generate two distinct but parallel flows from the processing operation.

One of these flows is then divided for presentation through two output functions (e.g., video and hard copy) while the other output is in a single mode. Once again differing error response and error introduction functions can be employed in each case.
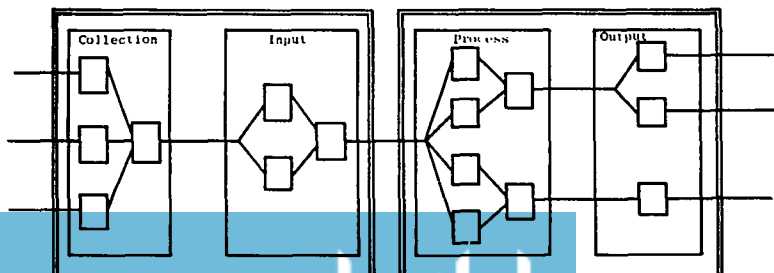

FIGURE 4. Use at Both the Macro and Detailed Levels of the Generalized Data Processing Blocks to Model Multi-Input, Multi-Output Information Decision Systems.

At the second level of detail four blocks could be employed to model the same system and aggregate error response and error introduction functions developed for the collection, input, process and output functions.

In the least detailed representation of this system, the collection and input blocks could be consolidated to determine the error characteristics of the flow entering the processing stage while the process and output blocks could be represented by a function which aggregates the error impact of these stages.

It is also a simple matter to construct a mixed system where one stage is represented in detail while the others are represented at a macro level. In this way a selective, focused analysis could be performed.

If the three outputs in Figure 4 represent information directed to operational, managerial and strategic decision levels, additional process blocks could be incorporated for each of these flows to represent, for example, error amplification through an overly responsive production function, error pass-through when used for evaluation of regional performance and error dampening when data are aggregated across regions for strategic planning.

Concluding Remarks

While the above example applies to the information flow for a centralized data management, similar scenarios albeit more complex can be developed for distributed systems. However, any realistic modeling of the information flows within an organization for the purpose of assessing the impact of potential deficiencies in data quality of selected terminal outputs would require the use of a computer-based realization of the model. Also, although the model is tractable provided the processing functions F are sufficiently smooth, for many applications the general function $G(x, dx)$ would be discontinuous. It is necessary to construct an approximation to such functions using values obtained at selected points $(x; dx)$.

The problem of approximating unknown functions using function values at selected points is a well-established mathematical discipline. (Schumaker 1966 presents a survey of approximation and interpolation techniques together with an extensive bibliography.) These procedures have been widely and effectively applied to certain problems (e.g., oil exploration). However, they are consuming of computer resources, and this would limit their use in the data flow/data processing model to crucial processing blocks.

A potential extension of this work would be a model designed to assess the cost/quality implications of taking corrective actions at various points of the data flow/data processing network. This issue and related ones are currently under investigation by the authors.[1]

References

ADAMS, C. R., "How Management Users View Information Systems," Working Paper 14, Management Information Systems Research Center, University of Minnesota, Minneapolis, 1973.

ALONSO, W., "Predicting Best with Imperfect Data," *J. Amer. Inst. of Planners*, 35 (1968), 248–255.

BAILEY, R. W., *Human Error in Computer Systems*, Prentice-Hall, Englewood Cliffs, N.J., 1983.

BERNSTEIN, P. A. AND N. GOODMAN, "Concurrency Control in Distributed Database Systems," *ACM Computing Surveys*, 13, 2 (June 1981), 185–222.

BODNER, G., "Reliability Modeling of Internal Control Systems," *Accounting Rev.*, 50, 4 (October 1975), 747–757.

BRODIE, M. L., "Data Quality in Information Systems," *Information and Management*, 3 (1980), 245–258.

BURNS, D. AND J. LOEBBECKE, "Internal Control Evaluation: How the Computer Can Help," *J. Accountancy*, (August 1975), 60–70.

CUSHING, B. E., "A Mathematical Approach to the Analysis and Design of Internal Control Systems," *Accounting Rev.*, 49, 1 (January 1974), 24–41.

DAVIS, G. D., *Management Information Systems: Conceptual Foundations, Structure and Development*, McGraw Hill, New York, 1974.

DE MARCO, T., *Structured Analysis and System Specification*, Yourdon Press, New York, 1978.

HAMLEN, S. S., "A Chance Constrained Mixed Integer Programming Model for Internal Control Design," *Accounting Rev.*, 55, 4 (October 1980), 578–593.

HANSEN, J. V., "Audit Considerations in Distributed Processing Systems," *Comm. ACM*, 26, 8 (August 1983), 562–569.

ISHIKAWA, A., "A Mathematical Approach to the Analysis and Design of Internal Control Systems: A Brief Comment," *Accounting Rev.*, 50, 1 (January 1975), 148–154.

JOHNSON, J. R., R. A. LEITCH AND J. NETER, "Characteristics of Errors in Accounts Receivable and Inventory Audits," *Accounting Rev.*, 56, 2 (April 1981), 270–293.

MARTIN, J., *Design and Strategy for Distributed Data Processing*, Prentice-Hall, Englewood Cliffs, N.J., 1981.

———, *Security, Accuracy and Privacy in Computer Systems*, Prentice Hall, Englewood Cliffs, N.J., 1973.

MOREY, R. C., "Estimating and Improving the Quality of Information in the MIS," *Comm. ACM*, 25, 3 (May 1982), 337–342.

SCHUMAKER, L., "Fitting Surfaces to Scattered Data," *Approximation Theory II*, G. Lorentz (Ed.), Academic Press, New York, 1966, 203–268.

STRATTON, W. O., "The Reliability Approach to Internal Control Evaluation," *Decision Sci.*, 12 (1981), 51–67.

YU, S. AND J. NETER, "A Stochastic Model of the Internal Control System," *J. Accounting Res.*, (1973), 273–295.