1976

System Analysis Perspectives

Thomas I. M. Ho

Report Number:
76-182

SYSTEMS ANALYSIS PERSPECTIVES

Thomas I. M. Ho
Purdue University
Computer Sciences Department
and
School of Management
West Lafayette, Indiana 47907

March 1976

CSD TR 182

SYSTEMS ANALYSIS PERSPECTIVES

Thomas I. M. Ho
Purdue University
Computer Sciences Department
and
School of Management
West Lafayette, Indiana 47907

ABSTRACT

Systems analysis is sorely in need of a conceptual framework that establishes principles and guidelines for the task of trarslating the qualitative and unstructured perceotion of organizational information requirements into the technical and rigid solutions available with computerized hardware and software. This paper outlines the major characteristics of a model for erecting a bridge between the diverse organization and computer systems that must be joined by an information system. Structured programming has provided the programmer with such a model. Surely, the systems analyst can benefit from new and old perspectives on his even more comolex task.

INTRODUCTION

It is widely recognized that the systems analyst's task is a formidable one. To perform his duties, the systems analyst must rely upon a broad array of disciplines. In particular, he must be familiar with the intricacies of two diverse worlds: that of the organization and that of the computer. Aside from the obvious size and complexity of these concerns, the contrast between these two ends of the spectrum is most striking. On one hand, the organization is qualitative and unstructured. On the other hand, the computer is technical and rigid. Therefore, it is no surprise to anyone that we have had difficulty with the application of computing technology to the information systems of organizations.

Therefore, it is clear that the diversity of these disciplines is the primary obstacle to the development of systems analysis. Furthermore, we have not orovided the systems analyst with the conceptual devices to enable him to translate an unstructured organizational information problem into the rigid hardware and software solutions available with comouting technology. Not even the technological software solutions offered by high-level programming languages and data base management systems have completely bridged the gap between the computerized implementation and an organization's perception of its information system. Technological solutiors only

aid the implementation of solutions that have already been formulated. The systems analyst needs a framework to help him to infer the elements of the technological solution from the problem that requires the solution.

Such a framework would provide the interface needed to translate the organization perspective into the computerized perspective. Achieving this interface first dictates a characterization of the features of these opposing perspectives that must be linked together.

THE SYSTEMS PERSPECTIVE

The unifying perspective on organizations and computers is the systems perspective. The perception of these entities as systems is an indispensable aid to understanding their respective essential features.

The organization system is composed of functional subsystems that represent the various functions performed by the organization. Each of these subsystems performs actions or decisions in order to achieve designated goals. These actions or decisions can only be made if the subsystem receives the data that establishes the constraints and objectives of the decision. For example, the production subsystem has responsibility for decision-making relevant to the manufacturing function, e.g. what to make, how many to make, and when to make them. These decisions can be made only if data is available on the demand for products and on the availability of resources for production. This data is only available from the environment and from other subsystems. For example, the marketing subsystem provides data on product demand and the inventory subsystem provides data on raw material availability. The production floor environment provides data on the availability of manpower and machinery. This interaction among subsystems requires the establishment of interfaces between the subsystems to enable them to function in an integrated fashion to accomplish their respective goals.

The computer system is composed of functional subsystems that represent the various functions performed by the computer. Each of these subsystems fulfills its designated role in the performance of computer workloads. A subsystem can fulfill its role only if it interacts with the other subsystems that support its function. In this way, the input subsystem receives the data to be processed, the storage subsystem stores the data before and after it is processed, the processing subsystem processes the data, and the output subsystem generates the data after processing. This interaction among subsystems also requires the establishment of interfaces between the subsystems to enable them to function in an integrated fashion to process the workload.

Finally, the systems perspective offers the concept of a decoupling mechanism to enable interfacing subsystems to interact in orderly fashion. Therefore, organizations maintain inventories to compensate for unequal rates of raw material procurement and consumption. An inventory is a decoupling mechanism between the purchasing and production subsystems. Computers use standard data representations to enable their functional subsystems to communicate with each other.

The organization system and the computer system are themselves interacting subsystems of an overall system. Therefore, these subsystems possess an interface that requires a decoupling mechanism to simplify their interaction. This decoupling mechanism is an information system model. This model provides a standard that enables organization system concepts to be expressed in a conceptual framework that is also compatible with computer system concepts.

PROPERTIES OF AN INFORMATION SYSTEM MODEL

The information system model is itself a system composed of interacting subsystems:

1. Input subsystem
2. Output subsystem
3. Data base subsystem
4. Process subsystem.

The correspondence to the various subsystems of the computer system is clear and this is no surprise. Correspondence to the elements of the organizational subsystem can be established. The elements of the output subsystem correspond to the actions and decisions performed by each organization subsystem. The elements of the process subsystem correspond to the procedures and models used to perform each action or decision. The elements of the input subsystem correspond to the data received from the environment by the elements of the process subsystem to generate the elements of the output subsystem. Finally, the elements of the data base subsystem correspond to the data received from another process by an element of the process subsystem to generate the elements of the output subsystem. For example, in the case of the production subsystem, the elements of its output subsystem compose the production schedule that assigns manpower, machinery, and material resources to the manufacturing function. The elements of the process subsystem compose the production scheduling model. The elements of the input subsystem compose the transactions that report the status of manpower and machinery resources. The elements of the data base subsystem compose the data that report product demand and raw material availability from the sales forecast and inventory status elements of the process subsystem.

The data base subsystem serves as a decoupling mechanism between the input and output subsystems. The input subsystem gathers the data from the environment to be used to generate information to the environment through the output subsystem. However, the output subsystem does not necessarily generate information at the same time nor at the same rate as the input subsystem receives data. Therefore, the data base subsystem is an inventory of data resources. Furthermore, the output subsystem does not necessarily request information in a format that is identical with that of the data used to generate the desired information. Hence, the data base subsystem maintains a standard specification for data resources in order to decouple the incompatibilities between the input and output subsystems. The decoupling role of the data base subsystem in these respects motivates the residence of the data base subsystem in the storage subsystem of a computer system.

With respect to the organization system, the data base subsystem also functions as a decoupling mechanism. The various functional subsystems of an organization system are interacting subsystems that must communicate with one another to achieve the desired synergistic effect. Again, the data base subsystem serves as both an inventory and as a standard for the data resources that are generated by any functional subsystem and can be used by any other functional subsystem in pursuit of that subsystem's objectives. Similarly, the data base subsystem also decouples separate procedures and models within a single subsystem. However, it is the data base subsystem's role as a decoupling mechanism between functional subsystems that elevates it to its central role in an integrated information system.

In order to function effectively as a decoupling mechanism, the date base subsystem must maintain the data resources so that they accurately reflect the state of the organizational environment. Only in this way can the process subsystem use the data resources to effect the appropriate organizational decisions and actions. Therefore, the process subsystem functions as a data base control mechanism that insures that the data base subsystem maintains the standard established by the environment. Then, the update function of the process subsystem senses the state of the data base for comparison with the state of the environment. If the data base is not in conformance, the update function invokes a feedback mechanism to modify the data base to conform to the state of the environment.

The application of this systems conceptual framework to systems analysis requires a means to describe the elements of the various subsystems in order to perceive their relevant characteristics. The resulting description is a blueprint for the design of the information system whose requirements are determined by the systems analysis. The blueprint is an implementation of the concepts motivated by the information system model. The model identifies the essential characteristics of the elements of the model's subsystems in order to

describe those properties that are relevant to the organization and computer systems that must be united by the desired information system.

An element of a subsystem is called a relational structure, a set of data names. A relational structure is simply a first normal form relation as defined by Codd [1]. The collection of relational structures that compose a subsystem represent a template of permissible forms that may be assumed by data residing in that subsystem. Data moves from one subsystem to another at prescribed rates and according to prescribed rules. The relevant characteristics of a relational structure include:

1. Composition
2. Identification
3. Timing
4. Volume.

Composition describes the data names that compose a relational structure. Composition defines various types of relational structures. The distinction between a relational structure type and a relational structure occurrence is crucial. The type represents the composition of the structure. The occurrence represents an instance of the structure. An occurrence corresponds to some identity in the real world that is being represented by the occurrence. For example, the relational structure type TIME-CARD = {EMPL-NO, PAY-PERIOD, HRS-WORKED} represents an input transaction for reporting employee time. The occurrence set OC(TIME-CARD) = {oc(TIME-CARD,i): $1 \leq i \leq$!OC(TIME-CARD)!} represents all possible transactions for all employees and all pay periods.

To distinguish one occurrence of a relational structure from other occurrences of the same relational structure, one or more data names of the structure are designated as identifiers. The identifiers form an identifier set, a subset of the corresponding relational structure, so that the identifier set occurrence of an occurrence of the structure is not identical to the identifier set occurrences of all other occurrences. The concept of identification is essential to the operation of matching corresponding occurrences of different relational structures to enable the flow of a data item from one relational structure to another. The specification of this data flow from one information system model subsystem to another is the purpose of the information system blueprint that we seek to construct.

This data flow occurs at prescribed rates that are determined by the timing and volume of the various relational structures. The timing of a relational structure indicates the frequency of data flow into the subsystem in which the structure resides. Together with the frequency, the volume of a relational structure dictates the rate of data flow into the subsystem in which the structure resides.

ADVANTAGES OF AN INFORMATION SYSTEM MODEL

The advantages of this mathematical formulation of an information system are manifold. Using mathematics as a vehicle for description of the model affords the clarity of expression for which mathematics is recognized. This approach is consistent with the earlier efforts of Young and Kent [2] and the CODASYL Development Committee [3]. A comprehensive formal definition of the model described herein has been presented by Ho and Nunamaker [4]. That formulation also motivates another advantage of the mathematical approach. The model is used as the foundation for determination of the quality of a systems analysis. Precise definitions of requirements completeness and consistency lay the foundation for establishing principles and guidelines for systems analysis. The absence of such principles is the major cause of the current incomplete and inconsistent system studies upon which many system design efforts have relied. This situation has resulted in a multitude of information system failures for which Morgan and Soden [5] provide a glimpse at only a very small sample.

Aside from the much-needed introduction of rigor into the practice of systems analysis, the establishment of a formal approach to systems analysis creates the overdue opportunity for development of tools for the management of systems analysis. Armed with only a motley assortment of graphical and narrative devices for describing numerous complex system requirements, today's systems analyst is faced with an unenviable task. However, the situation is no longer desperate since the advent of computer-aided techniques to assist the systems analyst. Most notable among these techniques has been the development of Requirements Statement Languages. A Requirements Statement Language (RSL) is a high-level language for use by systems analysts in describing the requirements of information systems. An RSL is *not* a programming language since a requirements statement expresses what organizational requirements an information system fulfills rather than how those requirements are implemented in a hardware and software solution. In other words, a requirements statement is an application of the formal information system model that interfaces the organization and computer systems. The effective use of Requirements Statement Languages is supported by software packages known as Requirements Statement Analyzers (RSA) that maintain requirements statements for subsequent use by all systems development personnel. The maintenance of a requirements statement uses RSA algorithms that perform logical checks on RSL statements for compliance with the completeness and consistency properties defined in terms of the information system model. Finally, an RSA also displays the system requirements in various tabular and graphical formats that enable the communication of system requirements to all personnel involved in the systems development effort.

The most advanced implementation of the RSL/RSA approach is the Problem Statement Language/Problem Statement Analyzer (PSL/PSA)

developed by the Information Systems Design and Optimization System (ISDOS) Project [6]. PSL is an English-like language possessing flexible facilities for describing data definition, timing, and volume. However, PSL facilities for describing system flow only provide the capability to document high-level flow without indicating data manipulation and processing requirements at the data element level. PSA provides extensive capabilities for maintaining and displaying system requirements expressed in a PSL statement. Therefore, PSL and PSA provide superior facilities for managing the documentation of high-level system requirements.

The Accurately Defined Systems (ADS) technique [7] provides a practical method for documenting system flow at the data element level. ADS describes the members of the output, input, process, and data base subsystems by indicating the data elements that compose the relational structures in each subsystem. Then, ADS documents system flow by indicating the source of each data element occurrence in each relational structure of the output, process, and data base subsystems. A source is another data element occurrence in a relational structure of either the input, process, or data base subsystem. The use of ADS as an RSL is vitally aided by an RSA for ADS reported by Nunamaker, Ho, Konsynski, and Singer [8]. Most important of all, both PSL and ADS and their accompanying software are currently being used in actual systems development and in systems analysis training.

The most important advance in support of the RSL/RSA approach has been the development of software for aiding program module and data base design in fulfillment of the requirements expressed in an RSL statement. Software for the generation of a data base schema and of data management application programs from a PSL statement has been developed by Blosser [9]. Software for the generation of programming specifications from an ADS statement has been developed by Ho [10].

CONCLUSION

The future of systems analysis is brightened by the prospect of the general availability of advanced tools and techniques for the performance and management of this difficult task. Although these tools are not yet generally available, today's systems analyst can already benefit from the insights into the analysis function that have been gained from the formulation of models upon which the tools are based. The systems analyst can take advantage of a conceptual framework to guide his analysis in much the same way that the programmer is guided by the maxims of structured programming.

REFERENCES

1. Codd, E. F. 1970. A relational model of data for large shared data banks. Comm. ACM 13, 6 (June 1970), 377-387.

2. Young, J. W. Jr. and Kent, H. K. 1958. Abstract formulation of data processing problems. Journal of Industrial Engineering (Nov.-Dec. 1958), 471-479.

3. CODASYL Development Committee. 1962. An information algebra phase 1 report. Comm. ACM 5, 4 (April 1962), 190-204.

4. Ho, Thomas I. M. and Nunamaker, J. F. Jr. 1974. Requirements statement language principles for automatic programming. Proc. 1974 ACM National Conference (Nov. 1974), 279-288.

5. Morgan, H. L. and Soden, J. V. 1973. Understanding MIS failures. Data Base 5, 2-4 (Winter 1973), 157-167.

6. Teichroew, D. and Sayani, H. 1971. Automation of system building. Datamation 17, 16 (August 15, 1971), 25-30.

7. Lynch, H. J. 1969. ADS: a technique in system documentation. Data Base 1, 1 (Spring 1969), 6-18.

8. Nunamaker, J. F. Jr.; Ho, T.; Konsynski, B.; and Singer, C. 1976. Computer-aided analysis of information systems. Comm. ACM, to appear during Spring, 1976.

9. Blosser, P. A. 1976. An automatic system for application software generation and portability. Ph.D. dissertation, Purdue University (in progress).

10. Ho, T. 1974. Toward a formal theory for the requirements statement, analysis, and design of information systems. Ph.D. dissertation, Purdue University (December 1974).