1976

New Perspectives for Information Systems Education

Thomas I. M. Ho

Report Number:
77-212

NEW PERSPECTIVES FOR INFORMATION SYSTEMS EDUCATION

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

CSD-TR 212

November 1976

Reference: Proc. 1977 National Computer Conference
pp. 569-573.

New perspectives for information systems education

by THOMAS J. M. HO
Purdue University
West Lafayette, Indiana

ABSTRACT

Systems analysis is the examination of a problem situation in order to define the requirements of a solution, often computerized, to that problem. The diversity of problems and the constraints of computing technology require that a problem be thoroughly analyzed in order to insure that the problem is clearly understood. Then, and only then it can be determined how computing technology can be applied to solve the identified problem. Therefore, systems analysis education teaches the discipline of clear and complete problem definition.

Several recent developments offer new perspectives for systems analysis education. These developments provide a conceptual framework for understanding information system and data base characteristics. This framework supports an improved methodology for systems analysis and thereby contributes to higher quality systems analysis education.

INTRODUCTION

A pair of technological developments in computer science are currently making a significant impact on the development of computerized information systems. Data Base Management Systems (DBMS) support sophisticated information systems with flexible facilities for the maintenance and manipulation of large complex data bases. In particular, a DBMS enables the sharing of data resources in order to co-ordinate the diverse activities of an environment supported by an information system. Furthermore, a DBMS enables the representation of complex logical structures in a data base that models the environment supported by that data base.

Computer aids to information systems development manage the activities of personnel engaged in the various phases spanning a development project from the conception of a need to the installation of the solution fulfilling that need. For example, a computer supported data dictionary manages the definition of data resources in order to provide a complete and consistent view of data for use in a DBMS application.

The impact of these technological developments stems from their contribution to relieving the difficulty of applying computing technology to information systems. The size and complexity of information systems are the major obstacles to their successful development. An information system for a large organization consists of several functional subsystems that represent the various functions performed by the organization. Co-ordination of the subsystems is necessary in order to achieve organizational objectives. Therefore, a DBMS enables the sharing of data resources among subsystems in order to co-ordinate these subsystems. Furthermore, each subsystem is a large problem by itself and so therefore, responsibility for a complete information system must be shared among many individuals. Therefore, computer aids to information systems development enable communication among systems development personnel to aid co-ordination of their activities.

The impact of these technological developments is further magnified by the wide spectrum of applications spanned by computing technology. The breadth of the span of applications justifies both the creation and the further improvement of educational programs that provide instruction in the application of computing technology to information systems.

The computer science curriculum at Purdue University includes an information systems program that provides instruction in the application of computing technology to information systems. This program includes instruction in both systems analysis and systems design techniques. Systems analysis is the problem definition activity that provides a complete and clear perception of the problem for which a computing solution must be determined during systems design. The dissimilarity between the varied problems in the problem domain and the varied technologies in the computing domain presents a unique situation that challenges the capabilities of information systems education.

The dissimilarity between the problem domain and the computing domain suggests the need for a common interface at which the system analysis and systems design activities can meet. Such an interface is a model of an information system which provides a conceptual framework for the statement of requirements of an information system. The statement of requirements represents the problem

requirements determined during systems analysis that must be satisfied by the technological solution selected during systems design. The conceptual framework must be consistent with a wide class of problems in the problem domain and with a wide class of technological tools in the computing domain.

The current approach to information systems education suffers from the absence of a conceptual framework for information systems. A narrow view of systems analysis education restricts instruction to a survey of traditional implementations of common application problems. As a result, the student encounters a motley assortment of computerized solutions that are applicable to only specific problems and dependent on specific implementation solutions of those problems. Therefore, *no* general problem-solving approach is apparent to a student whose education is limited by this narrow perspective.

Furthermore, a narrow view of systems design education restricts instruction to a survey of hardware and software technologies. In the absence of a common conceptual framework for information systems, such a technological survey leaves the student with no apparent approach to the selection of technological tools that are isolated from the problem domain. The virtually unlimited variety of the problem domain makes it practically impossible to indicate the technological solution to each problem. In addition, the absence of a conceptual framework makes it difficult to motivate the need for the capabilities of the technological solutions in the computing domain. For example, the need for representation of complex data base structure that is fulfilled by DBMS is not necessarily apparent to anyone who is unaware of the size and complexity of contemporary information systems problems. However, characterization of the role of the data base in the context of both the information system model and the problem domain motivates the need for complex logical structure.

CONCEPTUAL FRAMEWORKS FOR INFORMATION SYSTEMS

Information system model

The study of information systems is complicated by the dissimilarity of the organization system and the computer system. The organization system performs the activities that must be supported by the information system. The computer system performs computational and data management functions. There is *no* correspondence between the organization's activities and the computer's functions. The organization system contains the persons, objects, and events that are the subjects of organizational activities. The computer system contains the hardware and software facilities that perform computerized functions. There is *no* correspondence between the organization's subjects and the computer's facilities.

The gap between the organization and computer systems suggests the need for a conceptual bridge between these two systems. Such a bridge would guide and structure a systems analyst's activities while he formulates his approach to an organizational requirement. Such a bridge would *not* remove the necessity for the analyst to be familiar with the application domain with which he is dealing. Instead, the conceptual link identifies the concepts common to all applications of management information systems in order to supplement specific knowledge of organization and computer systems.

The conceptual link is an information system model[1] that provides a standard that enables organization system concepts to be expressed in a conceptual framework that is also compatible with computer system concepts. 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 role of the information system model is illustrated in Figure 1.

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 functional 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.

Data base 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

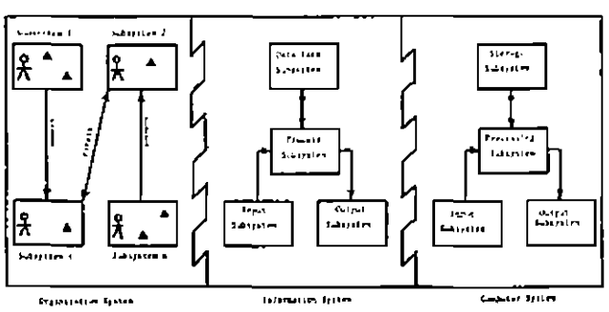

Figure 1—Management information system

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.

Entity-relationship model of data

In its role as a decoupling mechanism, the data base subsystem should therefore contain representations of the persons, objects, and events of interest to organizational activities. The elements of the data base subsystem that represent these persons, objects, and events are called *entities*. Furthermore, the data base subsystem should also contain representations of the relevant associations among the organizational persons, objects, and events. The elements of the data base subsystem that represent these associations are called *relationships* among the corresponding entities. The concepts of entity and relationship for data base definition have been proposed by both Teichroew,[2] Chen,[3] and ANSI/X3/SPARC Study Group on DBMS.[4]

An *entity* type is a model of a person, object, or event of interest to the organization system. An entity *occurrence* is the representation of an instance of the person, object, or event represented by the corresponding entity type. Therefore, EMPLOYEE may be an entity type while JOHN DOE is an occurrence of EMPLOYEE. An entity type consists of *attributes* that describe the entity. An entity occurrence consists of *facts* that describe the instance being represented. Therefore, if EMPLOYEE consists of the attributes NAME and ADDRESS, JOHN DOE might consist of the facts NAME is JOHN DOE and ADDRESS is 123 MAIN STREET. One or more of the attributes must serve as an *identifier* whose value distinguishes one occurrence of an entity from another occurrence of the same entity.

The scope of an entity is arbitrary. Part of one entity can be separately defined as another entity. For example, an object entity called Product can also be defined in terms of another object entity called Subassembly. Conversely, a *collection* of entities can be separately defined as another entity. For example, the collection of object entities Part and Product can be defined instead as the single object entity Material. Hence, in any organization system, any number of entity types is possible. Some guidelines for the selection of attributes of an entity have been presented by Brown.[5]

An entity type may be associated with some other entity type, *not* necessarily different from the first, by a *relationship* type. For each occurrence of one entity type, a relationship type defined between that first entity type and some other entity type defines a set of occurrences of the second entity type that have a common property (implied by the relationship) with respect to the occurrence of the first entity type. For example, a relationship type defined between the entity types CUSTOMER and ORDER defines the set of ORDER occurrences that were placed by each CUSTOMER occurrence. An important property of a relationship is its *connectivity*. For each occurrence of one entity type, connectivity indicates the maximum size of the set of occurrences of the second entity type that have the common property with respect to the occurrence of the first entity type. For example, the relationship CUSTOMER and ORDER has connectivity 1 to N (>1) because each CUSTOMER may place *more* than one ORDER, but each ORDER is placed by *only* one CUSTOMER.

The relationship concept is essential to the fulfillment of the decoupling role of the data base subsystem. The integration of the various functional subsystems of the organization system is promoted by relationships among the entities that represent the various persons, objects, and events of interest. For example, integration of the activities of the Order Entry, Inventory, and Shipping subsystems motivates the relationships indicated in Figure 2. A rectangle is used to represent an entity and a diamond is used to represent a relationship. The relationships indicate the creation of either SHIPMENT or BACKORDER occurrences to fulfill each ORDER occurrence. In addition, the relationship between BACKORDER and SHIPMENT indicates creation of a SHIPMENT occurrence when each BACKORDER occurrence is fulfilled. With respect to a customer's inquiry concerning any ORDER, the contribution to integration is apparent in the ability to respond with relevant information of either SHIPMENTs or BACKORDERs that fulfill that ORDER. Feedback is also apparent in the ability to inform a customer of the imminent receipt of his unfulfilled ORDER by virtue of the fulfillment of the responsible BACKORDER.

Effective organizational control is promoted by the relationship concept. Control is possible only if there exists a sensor mechanism to detect a system state that is at variance with some designated system standard. The sensor mechanism is enabled by the data base representation of a relationship that enables ready detection of the variance condition. As illustrated in Figure 2, a relationship type between BACKORDER and PRODUCT enables easy detection of the variance condition exhibited by excessive backorders for any particular product.

The relationship concept also restores the loss of structure that is apparent when the scope of an entity is

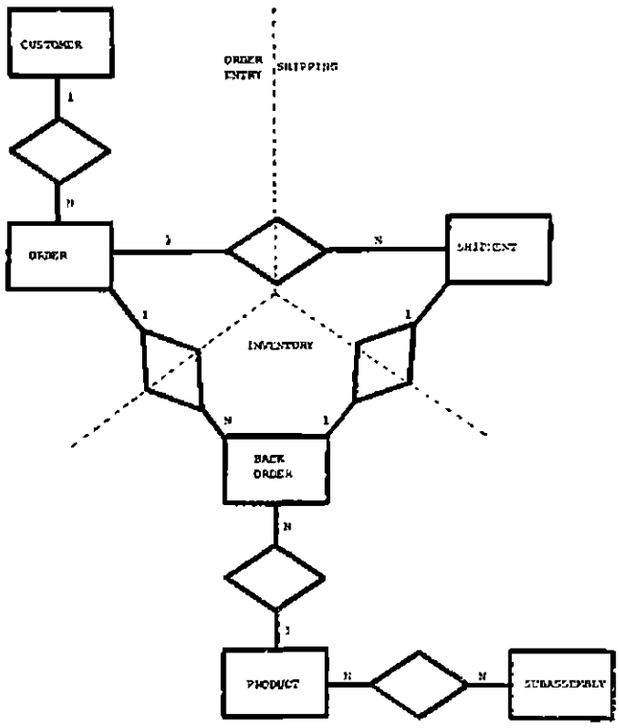

Figure 2

narrowed. When part of one entity is separately defined as another entity, the original data structure can be preserved by defining a relationship between the two entities. As illustrated in Figure 2, when a PRODUCT entity is defined in terms of a SUBASSEMBLY entity, structure can be preserved by a relationship type that defines the set of SUBASSEMBLY occurrences that compose each PRODUCT occurrence.

Finally, the relationship concept promotes data nonredundancy and its recognized contribution to data consistency and storage savings. Figure 2 includes a relationship type between CUSTOMER and ORDER that avoids redundant representation of CUSTOMER data in multiple ORDER occurrences placed by the same CUSTOMER.

CONCEPTS FOR INFORMATION SYSTEMS EDUCATION

Requirements statement language

An information system model suitable as a conceptual framework for information systems education is provided by a Requirements Statement Language (RSL). An RSL[1] is a high-level language for describing information system requirements that are determined during systems analysis and fulfilled during systems design. An RSL is *not* a programming language since an RSL statement expresses what requirements must be fulfilled rather than how those requirements are implemented in a hardware and software solution.

The most advanced RSL is the Problem Statement Language (PSL) developed by the Information Systems Design and Optimization System (ISDOS) Project.[6] PSL facilities for statement of data requirements conform to the entity-relationship model of data.[3] As described by Ho,[7] the entity-relationship model instills a data management perspective that exemplifies the role of the data base in the information system. To complement this systems analysis perspective on data management, a systems design perspective on data management is provided by techniques for the inference of a data base schema from a PSL statement of data requirements.[8]

In the case of statement of data manipulation requirements, PSL does not provide the desired facilities. PSL facilities do not provide the capability for stating detailed requirements for data manipulation and processing of data elements. The Accurately Defined Systems (ADS) technique[9] provides a practical method for describing system flow at the data element level. ADS describes the composition of the output, input, process, and data base subsystems of the information system model. Then, ADS describes system flow by specifying the source of each data element occurrence in the output, process, and data base subsystems. The source of a data element is an input, process, or data base occurrence of the same data element. However, ADS does not provide any facility for the statement of data manipulation requirements that might be fulfilled by a DBMS.

Requirements statement analyzer

The effective use of an RSL is supported by a software package known as a Requirements Statement Analyzer (RSA). An RSA performs logical checks on an RSL statement for compliance with completeness and consistency conditions defined in terms of the information system model.[10] The relevance to information systems education is evident in the RSA contribution to improving the quality of systems analysis and design. System design is entirely dependent on the completeness and consistency of the requirements determined during systems analysis. In addition, an RSA also displays the system requirements in various tabular and graphical formats that provide a system perspective of inestimable value to systems analysis and design.

PSL is supported by the Problem Statement Analyzer (PSA), an RSA that provides extensive capabilities for maintaining and displaying system requirements expressed in a PSL statement. The use of ADS as an RSL is vitally supported by PSA/ADS, an RSA for ADS reported by Nunamaker, Ho, Konsynski, and Singer.[11]

Requirements statement tools in information systems education

Currently at Purdue, undergraduate instructional activity in systems analysis is supported by the use of ADS as an RSL. Students learn the systems analysis task by partici-

pating in a term project that produces a requirements statement for an information system described in a case study. The case study problem includes a diverse collection of interacting organizational activities in order to create a problem situation of sufficient size and complexity. These characteristics of the problem situation create the need for data sharing and project management that motivates the use of DBMS and computer aids for information systems development.

However, ADS has proven to be inadequate for statement of the data definition and manipulation requirements for data management that can be fulfilled by a DBMS. Furthermore, use of PSA/ADS for requirements statement analysis is less than satisfactory due to the limitations of its primitive implementation. All these shortcomings do not exist in PSA/PSL with one major exception. PSL does not provide facilities for stating detailed requirements for data manipulation as might be performed by a DBMS. In addition, PSL does not allow specification of the source of each data element occurrence as provided by ADS. Our experience in systems analysis education has demonstrated that the concept of data sources is pedagogically essential to understanding the flow of data in an information system. In all other respects, PSA/PSL has proven to be an excellent tool for systems analysis education. Our experience with use of PSA/PSL in graduate systems analysis education here at Purdue has been extremely favorable.

In addition to the more obvious contribution to a conceptual framework for information systems education, computer aids also contribute to the productivity gain that results from the detection of systems analysis errors and the automated maintenance of requirements statements. Productivity is especially vital in the educational environment which is expected to provide relevant student experiences in courses of relatively short duration. We therefore confront students with problems of realistic size and complexity that require high productivity for their successful completion in the short span of a course.

CONTRIBUTION TO INFORMATION SYSTEMS EDUCATION

The greatest impact of the products described herein will be higher quality systems analysis education. The capability for completeness and consistency checking afforded by the RSA provides feedback on student performance and better systems analysis technique as a result. The structure inherent in RSL statement re-inforces the concept of structured programming and other progressive software development disciplines. The productivity gain accomplished by the use of automated techniques encourages creativity by relieving the user of the burden of manual systems analysis techniques. The exposure to state-of-the-art techniques expands student perspectives beyond the traditionally confined horizons of traditional data processing. In summary, the impact is most evident in the improved ability to apply computing technology to the wide and varied spectrum of problems that could benefit from data management relevant to their decision-making and other activities.

REFERENCES

1. Ho, T. I. M., "Systems Analysis Perspectives," *Proc. 14th Annual Conference on Computer Personnel Research*, July 1976.
2. Teichroew, D., *et al.*, *An Introduction to PSL/PSA*, University of Michigan, Department of Industrial and Operations Engineering. ISDOS Working Paper No. 86. March 1974.
3. Chen, P. P. S., "The Entity-Relationship Model: Toward a Unified View of Data," *Trans. Database Systems*, Vol. 1, No. 1, March 1976, pp. 9-36.
4. ANSI/X3/SPARC Study Group on DBMS, *Interim Report*. Doc. No. 7514TS01, CBEMA, Washington, DC, February 1975.
5. Brown, A. P. G., "Modelling a Real World System and Designing a Schema to Represent It," *Data Base Description*, Douque, B. C. M. and G. M. Nijssen (eds.), North-Holland Publishing, 1975, pp. 339-347.
6. Teichroew, D. and H. Sayani, "Automation of System Building," *Datamation* Vol. 17, No. 16, August 15, 1971, pp. 25-30.
7. Ho, T. I. M., *Data Base Concepts for Systems Analysis*, Purdue University, Computer Sciences Department Technical Report, November 1976.
8. Blosser, P. A., *An Automatic System for Application Software Generation and Portability*, Ph.D. dissertation, Purdue University, May 1976.
9. Lynch, H. J., "ADS: A Technique in System Documentation," *Data Base* Vol. 1, No. 1, Spring 1969, pp. 6-18.
10. Ho, T. I. M. and J. F. Nunamaker, Jr., "Requirements Statement Language Principles for Automatic Programming," *Proc. 1974 ACM National Conference*, November 1974, pp. 279-288.
11. Nunamaker, J. F., Jr., T. I. M. Ho, B. Konsynski, and C. Singer, "Computer-Aided Analysis of Information Systems," *Comm. ACM*, to appear during Winter 1977.