SOME USER-ORIENTED CONSIDERATIONS IN THE DESIGN OF MEDICAL INFORMATION SYSTEMS

ABSTRACT

There are a broad range of factors which must be considered in order to create interactive medical information systems which will be acceptable to their users, in ease of use, in realiability, and in cost. This paper proposes a methodology for designing and developing such systems, based upon the principles of user engineering and software engineering. This methodology, called user software engineering, requires a clear understanding and precise definition of the problem, as well as attention to program construction techniques, management of software development, testing, and documentation.

Among the issues which are discussed are hardware and software reliability, the role of modularity in achieving reliability, terminal selection, the human/computer dialogue, and psychological factors in the design of medical information systems. Illustrations of recommended approaches to handling these areas are given. An attempt is made to show that virtually every decision concerning the design, development, selection, evaluation, or installation of a medical information system must be considered from the standpoint of the users of that system.

INTRODUCTION

Over the past few years, there has been a noticeable trend toward increased usage of on-line medical information systems, involving a growing number of health care professionals. Among the various interactive systems developed for medical use are clinical laboratory systems [1], automated interviewing programs [2], medical record systems [3,4], and medication systems [5], to name just a few. In some hospitals and clinics, the use of conversational programs is very widespread and virtually every employee must interact with a computerized medical information system [6].

One of the primary objectives in the development of all medical information systems is that they be acceptable to the people who will be using them and to those who are paying for them. Among the factors which determine the success of medical information systems are the cost and cost-effectiveness of the system, its reliability, its correctness both from the computing and a medical standpoint, and the interface between the system and its users. In addition, another important element in the production of a successful medical information system is the working relationship that must be developed between those who are designing the system, those who are building the system, those who are paying for the system, and those who will eventually be using the system [7].

Despite the recognition of the importance of these factors, the history of the development of medical information systems shows that a significant percentage of such systems fail to meet the needs of their users and that many systems go unused after a large developmental expenditure. There are a number of primary reasons for these problems, including:

1) poor decisions on equipment selection;
2) unavailability of low cost, reliable, non-keyboard terminals;

3) poor software development practices;
4) lack of good tools for the design of interactive computer programs, with particular weakness in program testing and error prevention facilities;
5) design and implementation of medical information systems by persons unfamiliar with the medical environment or the health care delivery systems;
6) inaccurate medical content;
7) user resistance to the introduction of information technology; and
8) failure to include psychological as well as technical considerations in an overall systems design.

Because of the poor documentation of previous unsuccessful developmental efforts, and the slow progress being made in overcoming some of the above problems, these problems and others continue to make the design and development of successful medical information systems quite difficult. In particular, economical techniques for developing quality software have failed to advance to the level required by the nature of medical data processing. Urgent medical data processing needs have been forced to rely upon existing hardware and software technology, with the result often being ad hoc systems which are uneconomic and unreliable.

However, the goal of this paper is not to examine these reasons for failure in more detail, but rather to suggest techniques which can be used in order to design and develop medical information systems which are truly acceptable to their users. With the increasing number of on-line systems under development and the increasing cost of software development, it is desirable to identify as many as possible of these potential trouble spots in advance, suggest workable approaches to some of them, and thereby minimize the chances of their occurrence. The methodology proposed relies both on the principles of user engineering and the concepts of the emerging discipline of software engineering. An essential step in this process is to focus on medical information systems from the standpoint of their users, thereby making it apparent that the needs of the users must take precedence over other considerations if such systems are to be successful.

User software engineering is defined as the combination of software engineering principles with human factors considerations in order to produce a satisfactory human/computer environment. Software engineering may be defined as the application of principles, skills, and art to obtain economically software that is reliable and works efficiently on real machines [8,9]. It therefore includes the specification, design, development, management, testing, and maintenance of software systems. User engineering may be defined similarly as the application of principles, skills, and art to the creation of software which is acceptable to its users in terms of ease of use, cost, and reliability.

Together, software engineering and user engineering require a broad range of considerations by systems designers throughout the definition and development process. Above all, user needs and desires must be carefully treated throughout the software development process. The need to understand sufficiently and define clearly the users' problems is established as a fundamental objective of user software engineering. At the same time, attention must be given to hardware selection, program construction techniques, program verification, management of software development, testing and evaluation, and documentation. Sound program development practices can favorably influence the operational characteristics of any information system.

A high degree of user software engineering is desirable if information systems are to be truly valuable to a wide range of potential users. This is especially true in the design of systems for non-programmers, since they tend to be less tolerant of computer problems than are programmers. This paper

attempts to cover many of the various factors which must be considered when designing an interactive medical information system.* These factors are quite broad in scope, because of the wide diversity of persons who may use such a system. The order of presentation of factors goes from the most quantifiable to the less tangible, with the initial ones having a solid technical base and the latter ones being largely subjective. The approach taken is to present a general set of guidelines to be followed by systems designers and developers with regard to each topic.

Emphasis is given to issues surrounding the computing environment for a medical information system, rather than the medical content of a system. It should be recognized, however, that both the computing environment and the medical content must be acceptable in an operational system. Furthermore, of the factors listed, different factors should be given greater or lesser emphasis in specific systems, depending on the projected class of users for the system. However, none should be ignored.

RELIABILITY AND INTEGRITY

The primary consideration in any computer system is that it be available and working properly when the user wants to use it. Reliability is such an overriding concern that it should never be compromised against some other factor. The ease of conversation, the elegance of software design, and the convenience of the terminals are entirely irrelevant if the system will not work dependably when the user needs it. This notion of reliability and integrity applies to the hardware, the software, and the data involved in the complete system; attention must be given to each of these factors.

Hardware

The notion of hardware reliability implies a high mean time between failures for each hardware component. In some cases, when the need for availability on a twenty-four hour per day basis is essential, it may be necessary to replicate part or all of the hardware in order to minimize the time that any system is unavailable, and to have extra pieces of equipment available for replacement of faulty components such as terminals. The hardware needs often extend beyond the computer system itself and include electrical power and possibly lighting or air conditioning.

From the standpoint of reliability, hardware modularity has been shown to be extremely valuable in avoiding the cost of replicating an entire system. If the hardware is a single, indivisible central processing unit, the failure of any component within that unit will cause the entire system to fail, making it impossible to do any productive work until the failure has been repaired or a back-up system has been activated. In a modular configuration, however, it is possible for part of the system to fail and still allow processing to be done by those parts of the system which remain in working condition. In most cases, the absolutely critical processes can continue to work.

Consider, for example, the processing needs of hospital information system for a medium to large scale hospital. Such a system must be available, at least to a limited degree, at all times. If the system is configured around a single processor, the desired level of system availability can only be achieved through the existence of a back-up processor which can be quickly brought into action if the primary processor fails. Alternatively, however, consider the use of a number of smaller processors to replace the single central processor. Each of

*Many of these factors are also applicable when selecting a medical information system from a vendor.

these smaller processors could be placed in a central area where each would serve one hospital department, with one of these computers connected to all of the others to allow communication among all of the departments [10]. If one of the small computers serving one of the nodes of this "star" network should fail, only that particular department is affected by the failure. Other departments would continue to have operating computers and are only affected by their inability to exchange data with the department whose system has failed. If all of the computers in the system are compatible, then one extra computer can serve as a back-up for all of the other systems, including the central communications processor, and system "down-time" would be minimal. In this way, a high degree of reliability can be achieved for a relatively low cost.

Software

Software reliability is, of course a key issue in software design, as efforts are made to develop techniques which lead to the creation of correct programs. Much work has been done recently on program testing, program debugging, proofs of program correctness, and other related notions [11]. Attempts to create correct programs have resulted in better understanding of program control structures, which has led to the notions of structured programming [12], as one means to develop programs which can be verified easily. There are some rather significant software requirements in order to develop absolute reliability in an interactive system. Since a user at a terminal may enter incorrect data which may lead to an execution error and unexpected program termination, it is necessary to create "fault tolerant" software. In order for computer programs to achieve this, it is necessary for the conversational program to be able to treat every possible user response as a meaningful response, rather than assuming the proper input will always be made. In addition, the program must be prepared for other kinds of error conditions which may arise, including the following:

1) input-output errors;
2) no input at all, possibly due to a broken connection between the terminal and the computer; and
3) a hardware failure.

Virtually no presently available programming language contains all of the features needed to provide this degree of reliability. In many languages, for example, it is very difficult to trap arithmetic overflows or invalid array subscripts, thereby making it possible for the user to cause a program to fail unexpectedly, for the user to lose all or part of his work, or for the user to receive a meaningless message. A lack of adequate features for the development of fault tolerant software discourages programmers from building in these important user-oriented features, since control of certain of the errors requires a considerable amount of programming effort.

One technique which has been shown to be extremely useful in software development is modularity. Software modularity refers to the systematic programming techniques which should be used to build reliable software [13]. Practical history and the recent development of computer systems has shown program modularity and organization to be vital for project success. Various components in a system must be carefully separated and designed and then combined with a sensible sequence of module construction [14]. An overall stepwise refinement of system design can be used to assist in integrating the modules to perform the various tasks [15].

The advantages of this approach are numerous, including the ease of modification of individual units, the ability to identify and replace those modules which are performing incorrectly or inefficiently during the development stages, and the improved logical organization which results in breaking a large

task into a number of smaller ones by proceeding from an abstract problem statement through a series of stepwise refinements to a complete program. This modular approach to program development also leads to increased reliability of the finished software, since it simplifies the testing and debugging procedures.

Data and Security

In addition to reliable hardware and software, there must be reliable data storage. Every effort must be made to preserve the integrity of data to make certain that no data are lost. Therefore, there should be replication for off-line storage, or an audit trail, so that any lost data can be easily recreated. Most users will not tolerate a situation which forces them to reenter data because of program or system failure. Also, the ability to alter the data base should be severely restricted and a verification of data base changes should be required as a means to preserve accuracy.

Closely related to reliability of data storage are the issues of security and confidentiality. Adequate security is necessary for overall reliability of the information system, and for protection of the data from unauthorized access. Users of the system should only have access to those pieces of the data which they "need to know," with adequate checking of persons using the system and its files. There can be an additional level of passwords for access to particular programs or data, and there should be protection for the programs running on the system so that they do not interfere with each other and so that they cannot be modified in such a way as to permit improper use of information [16].

Adequate security, then, is necessary for overall reliability of the system and its data. Unprotected or unreliable hardware, software, or data, within a medical information system, can easily lead to failures in system performance, errors in medical treatment, and unauthorized access to and dissemination of confidential information.

EASE OF USE

It can be argued that the primary consideration of any information system is the manner in which the users work with the system [17]. This consideration involves terminal devices, man-machine dialogue, and a large number of psychological considerations, all of which must be satisfactory for the users. While this argument ignores the importance and difficulty of achieving reliability, security and modularity, one can easily make the case that those factors are rightfully not the user's concern and should be taken for granted by the user. Indeed, the system designer should often taken this view when working with nonprogramming users, so that careful attention can be given to definition of the problem, structure of the software interface, terminal selection, and other user-oriented issues. Then, those features can be incorporated in a software system which exhibits all of the features discussed above.

The Hardware Interface

In an interactive medical information system, choice of terminals and input mechanisms is extremely important to the users of the program, since the terminal will be the sole contact with the system for most users. If the user is not comfortable with the terminal, then the user will be uncomfortable with the entire system. Therefore, system designers must take into consideration the types of users of the system when selecting terminal devices. There are also a broad range of terminal features to be compared.

For example, a system to be used by persons whose job requires typing might be based upon teletypewriter terminals with keyboard input, since those users

can type; a system which requires typing by persons who do not normally type is far less likely to be successful, since those users are not trained typists and generally dislike having to type. For this latter class of users, alternate input methods and terminals must be explored, and weighed against training people to use a keyboard.

Similarly, the location of terminals, the noise produced by terminals, and speed of their output are also important factors. The noise produced by a typewriter-like terminal is generally too great to make such a terminal acceptable on the wards of a hospital, but the same terminal might be acceptable in an office area. Terminal output rates can vary from ten to approximately 1,000 characters per second on a conventional terminal, with even higher rates available on some frame-oriented systems. In some applications, low speed terminals will suffice; for many applications, however, a far greater speed is necessary for user satisfaction or for the volume of output.

Thus, there is no type of input device or output device that is to be universally favored. It should be recognized, though, that more expensive devices, including high speed, low noise printers, and a variety of sophisticated input devices, such as light pens and touch screens, are often necessary expenditures to achieve user acceptability of a medical information system.

There is often an interesting tradeoff here; for example, if a medical professional is unwilling to type more than a single character in response to a program-generated query, a frequently used alternative is to provide the user with a variety of choices from which to select one. This decision, however, means that the list of alternative choices must be presented very rapidly, thereby requiring a higher speed output terminal, since a lengthy delay in presenting the alternatives may be equally unsatisfactory.

These types of considerations play a major role in the broader issues of hardware selection for medical information systems. The key problem in these systems is usually not one of computational speed, but rather one of information transfer. It must be possible to retrieve a given piece of data and display it to the user very rapidly, if the system is to be widely accepted. For this reason, many medical information systems can best be thought of as communication systems, with a primary decision factor being the transmission rate between a secondary storage device and a terminal. Traditional factors of computer selection, such as core memory and cycle time, must now be balanced against such factors as the availability of multi-ported memory and the availability of secondary storage devices with low latency times. Thus, a diversity of factors, including noise, expected volume of output, required speed of output, and input mechanisms, along with response time (see below), must all be considered when selecting computer hardware for use in medical information systems.

All of these user-oriented criteria must be applied to the selection of the central processor, the secondary storage devices, the input and output devices, and the communications subsystem, bearing in mind that those features which are most attractive to programmers are <u>not necessarily</u> most attractive to a non-programmer. A serious deficiency in this area almost certainly leads to wide user dissatisfaction with the medical information system and can thereby render it ineffective.

The Software Interface

The software interface may be defined as the dialogue between the computer system and the user and is in many ways the most important aspect in the design of user-oriented systems. The format of the output(s) presented to the user and the required format(s) of user input(s) are hard to define, but are crucial factors in overall system development. For this reason, it is especially

important that the system designers develop a thorough understanding of each user task and the precise terminology used in the performance of that task, so as not to present the user with a new set of tasks to be performed or with a large number of unfamiliar terms. This problem is especially noticeable when one institution "imports" a medical information system being used by another institution. Unfortunately, procedures and terminology vary considerably from one institution to another, so that a system which is effective in one site may fail in another, even though it may have been entirely satisfactory in its original site.

Interaction computer programs have a "personality" which may be rigid and inflexible on one hand, or helpful and forgiving on the other. It is desirable to develop programs which are "soft," tailored to ease of use, rather than "hard," designed for ease of processing. It should be remembered that two of the key objectives of user software engineering are minimizing the amount that the user has to learn about the computer, and maximizing the degree to which an information system can assist that user in performance of a job.

One of the most essential achievements in developing a "soft" personality is computer system invisibility. The typical nonprogramming user knows little or nothing of computer sciences and regards the computer only as a tool which may be able to assist him in his work. Error messages should be expressed in terms of the user's task, interaction with a system job control language should be minimized, and there should be no necessity to learn about technical details such as data base structures and disc files.

Among the considerations which must go into the construction of the human/computer dialogue are a number of "user-oriented" features, including ease of learning and remembering how to use the system, flexibility of usage for skilled vs. unskilled users, error correction and prevention, and on-line assistance, allowing the user to interact in a "natural" way with the program, without having to learn cryptic codes or type lengthy amounts of information in a precise format.

Consider the issue of flexibility. Since different users will have different degrees of skill in working with the system, there should be facilities to accommodate these varying levels. Lengthy output messages and explanations can often be reduced or suppressed for experienced users. Another problem with flexibility is that different people have different mental organizations and work habits, and that different people will wish to make different types of information requests from a data base. For example, a physician or nurse ordering laboratory tests may want to specify all of the tests to be conducted on a given patient, while the clinical laboratory may alternatively wish to list all of the patients on which a given test is to be run. The ability for a medical information system to handle these orthogonal requests requires considerable flexibility in system design. Some of the research and development currently under way in the field of relational data base management [18] should be extremely helpful in this regard. From the standpoint of the system designer, he must be able to strike a reasonable balance between a rigid, inflexible system and a totally general system which would serve everyone's needs, but at great development and operational expense.

A related need is the ability to correct errors quickly. The physician, for example, should be able to eliminate orders as quickly as make them, without disturbing valid orders. A system which forces its users to delete and re-enter a sizeable segment of information in order to correct a relatively minor error places an unnecessary demand upon its users. This need for error correction facilities is especially important in medical applications when data are used for patient management. Incorrectly entered laboratory results, for example, can easily lead to improper diagnosis or treatment of a patient. Difficulty in

making corrections can cause errors to be propagated through the system, thereby making it undependable for patient care.

Efforts must also be made to prevent errors before they occur. A set of orders for a patient should be displayed as a group before they are accepted by the system. In addition, the program should perform some kind of "reasonability check" in order to guard against errors, and should require verification by the user for seemingly unreasonable input. A program action should be provided for every possible response that the user might make, so that no user input can cause the system to fail.

Although there are other factors which could be included in the design of the software interface, these considerations serve to illustrate the complexity of a problem and the need to give it careful attention throughout design and development. The factors cited here must be supplemented by thorough planning and system design, combined with those programming techniques suggested above, so that development and modification of the software interface can be made a straightforward matter. This last point is especially important since no medical information system is entirely acceptable to its users in its initial version.

EASE OF CHANGE

Medical information systems must be designed for a dynamic, rather than for a static, environment. The designers must assume that the computing equipment will change, the programming and system development staff will change, and that the nature of the task performed by the system will evolve in time, as institutional procedures are modified, and as information requirements change. Failure to provide for these types of changes will result in a rigid system which cannot easily respond to advances in hardware technology, the loss of a key staff person, changes in the human/computer interface, or new reporting requirements. Such a system may satisfy users initially, but can quickly become unacceptable. As a result, portability and adaptability of programs becomes an important issue to users of medical information systems. Poole and Waite [19] have defined portability as a measure of the ease with which a program can be transferred from one execution environment to another and adaptability as a measure of the ease with which a program can be altered to fit differing user images and system constraints.

Considerations of portability and adaptability suggest the need for a strong attempt to preserve machine independence. They suggest the use of modular programming techniques as discussed above in order to simplify program modification. They also favor very strongly the use of higher level languages, particularly those languages for which there is a recognized machine-independent standard. Experience has shown that fully half of the cost of a major software system is associated with its maintenance and with conversions to new equipment. Thus, readability of programs, clear documentation, and an avoidance of programming "tricks" must be balanced against the traditional measures of efficiency in order to develop information systems which can have a long lifetime, without depending upon particular persons or equipment for their continued useful existence.

PSYCHOLOGICAL FACTORS

The eventual acceptance of an information system into routine usage is often dependent upon a number of psychological and related nontechnical issues, rather than upon the mechanical aspects of the system. Failure to consider these essential psychological issues will often produce an information system which will be rejected by those for whom it was supposedly designed and

developed. These factors which affect user attitudes include the general previous disposition of the user toward computerized information systems, and the process by which system development and introduction is undertaken, along with the operational characteristics of the system as noted above.

Predisposition of users toward information systems is something which must be considered before one even determines whether to go ahead with building a system. In some organizations, it is extremely difficult to get the degree of support needed at all levels to make a system work effectively. There is often a wide difference between the users of a system and those who pay for its development and operation. Similarly, the structure of some kinds of organizations is better suited to development of such systems than are others.

By far the most important predisposition attitude is that of personal threat. Computer systems can represent an ego threat by appearing to do a job better than a human and an economic threat by appearing to be able to replace them in their jobs. It is certain that an individual whose ego or income is threatened by a computer system will be opposed to it and will make a considered effort to make the system fail. The concept of personal threat is taken most lightly by system designers, perhaps because information systems are not a threat to them. The developers of the Community EKG Interpretation Service took these fears into account, going to considerable effort not to threaten or displace human EKG readers while still tyring to provide a rapid, computerized EKG analysis to a remote location [20]. By simply labeling the computer's interpretation "UNCONFIRMED" and by continuing to use the human readers for verification and/or correction of the computer analysis, intellectual superiority of the humans was affirmed and the readers continued to collect their fee for reading the EKG trace. At the same time, though, the computer analysis was completed in a few minutes, providing a much more rapid response than was previously possible, with a fairly high level of accuracy, thereby assisting in the care of patients with cardiac problems. It seems certain that this EKG system would have been unsuccessful had its developers simply tried to replace the human EKG readers.

The second important psychological area is that of system development and introduction. First, users must be participants at all levels and stages of the development or selection of the medical information system, rather than having the system imposed upon them. The system designers thereby obtain a first-hand view of the way that the user performs his job, and of that user's real information needs. Closely related to this area is that of user training. The users must be provided with adequate introductory and reference documentation, personal instruction and, ideally, on-line assistance. The purpose of this training is to make the user community largely independent of the system development team, making it possible for users to solve their own problems within a short period of time after the system has been implemented.

One of the most effective techniques for system design, testing, and training is to have several users participate in the design and development process. This small group of users, with its strong knowledge of the functional details of the system, can then teach use of the system to everyone else.

Several operational characteristics are also important in acceptance of a medical information system. Of those not previously discussed, one of the most important is that of response time. Both speed and variability of response time are important issues in user psychology. Estimates of maximum acceptable periods for response time range from a fraction of a second up to about two seconds for the system to respond to "normal" requests. Similarly, two inputs of similar complexity should have approximately the same response time. When response time is too long or too erratic, there is a tendency for the user to lose a train of thought and this leads to a higher degree of errors, which, while not directly attributable to the computer system, can be reduced by better

response characteristics.

By far the most important psychological consideration, however, is interpersonal communication. The successful design and development of a medical information system requires the close cooperation of health care professionals, computer professionals, industrial engineers, and management, since extremely few people possess knowledge in all of these areas. In addition to extremely diverse backgrounds, education, and terminology, various individuals will have widely differing perceptions of the functions of a medical information system. For this reason, it is extremely important to give lengthy attention to the total design of a system, making certain that everyone agrees on the proposed capabilities and limitations of the system. When implementation is begun prior to the completion of an agreed-upon design, it is almost certain that there will be proposed changes in the system specification, once everyone understands the ramifications of existing design decisions. The result of such changes is that the design becomes a "moving target" and the complexity of implementation is greatly increased, thereby disrupting the planned software development procedure.

These psychological factors vary widely among individuals and organizations, thereby making it difficult to provide definitive guidelines for handling these problems. However, it is clear that there is a great need to obtain better understanding of these psychological considerations in the design of medical information systems, and to give them attention throughout system development and design.

CONCLUSION

Although this paper attempts to list a number of important user-oriented considerations in the design of medical information systems, the list is by no means complete. For example, every institution and organization has its own internal political problems which must be overcome. In many institutions, the history of previous successes and failures must be taken into account. Even more important are the economic considerations in the system design. Many economic decisions impinge directly on the degree of user orientation that will be available on a given system. For example, a requirement to reduce costs may result in a less acceptable terminal being selected. Furthermore, measurement of cost benefit, cost effectiveness, or cost justification is often very difficult if not impossible [21]. In summary, following these suggestions is not a guaranteed formula for success; however, failure to follow a significant number of these guidelines will almost certainly guarantee disaster.

Unfortunately, the lack of adequate software tools, and of inexpensive non-keyboard terminals cited above, continue to be problems [22], which will continue to result in unsuccessful systems. Much more research and development needs to be done in this area. The need to develop low cost computer hardware with sufficient software for use in on-line medical informaton systems has become apparent. There are now educational programs in medical information science whose goals are to train persons capable of designing, developing and evaluating information systems for health care [23,24]. Also, the nature of the human/computer interface is becoming better understood and is being taken into greater consideration as designers recognize the shortcomings of other systems, although more rigorous experiments need to be undertaken. Eventually, the present technological reasons which have hampered widespread use of medical information systems will disappear.

In summary, it should be clear that virtually every decision that is made concerning the design, development, selection, evaluation, or installation of a medical information system must be considered from the standpoint of its effect

upon the users of the system. Development of user-oriented medical information systems will almost certainly lead to more efficient utilization of the time of medical professionals, to improved flow of accurate information within an institution, and eventually to improved health care delivery.

ACKNOWLEDGMENT

This work has been partially supported by grants from the National Library of Medicine (LM 00153) and the Commonwealth Fund.

REFERENCES

[1] Williams, George Z. and Williams, Robert L. "Clinical Laboratory Subsystem," in Hospital Computer Systems, ed. M. Collen. New York: John Wiley and Sons, 1974, 148-193.

[2] Budd, M., Bleich, H., et al. "The Acquisition of Medical Histories by Questionnaires," National Center for Health Services Research and Development, U.S. Department of Health, Education, and Welfare, 1970. Publication HSRD-70-13.

[3] Anderson, J. and Forsythe, J. M. (eds.) Information Processing of Medical Records. Amsterdam: North Holland Publishing Company, 1970.

[4] Justice, N., et al. "Design and Development of a Medical/Management Information System at the Harvard Community Health Plan," Proceedings of the 1974 National Computer Conference. Montvale, New Jersey: AFIPS Press, 1974, 159-166.

[5] Goldberg, M., et al. "A Drug Data Bank: Specific Problems in Connection with the Nature of Informations and Operating Methodology," MEDINFO 1974, 869-874 (preprints). [Similar paper in Journees d'Informatique Medicale Toulouse, Institut de Recherche d'Informatique et d'Automatique, 1974, v. 2, 287-300.]

[6] Norwood, D. "Introduction of a User-Oriented THIS into a Community Hospital Setting--Introduction and System Description," MEDINFO 1974. Amsterdam: North Holland Publishing Company, 1974, 295-298 (preprints).

[7] Collen, M.F. "General Requirements," in Hospital Computer Systems, ed. M. Collen. New York: John Wiley and Sons, 1974, 3-23.

[8] Naur, P. and Randell, B. Software Engineering. Brussels: NATO Scientific Affairs Division, 1968.

[9] Beckmann, M., et al. (ed.) Advanced Course on Software Engineering. Berlin: Springer-Verlag, 1973.

[10] Blois, M.S., and Henley, R.R. "Strategies in the Planning of Medical Information Systems," Journees d'Informatique Medicale, St. Lary, Institut de Recherche d'Informatique et d'Automatique, 1971.

[11] Hetzel, W.C. (ed.) Program Test Methods. Englewood Cliffs, New Jersey: Prentice-Hall, Inc., 1972.

[12] Dijkstra, E.W. "Notes on Structured Programming," in Structured Programming, ed. O.J. Dahl, et al. London: Academic Press, 1972, 1-81.

[13] Wirth, N. Systematic Programming: An Introduction. Englewood Cliffs, New Jersey: Prentice-Hall, Inc., 1973.

[14] Liskov, B.H. "A Design Methodology for Reliable Software Systems," Proceedings of the 1972 Fall Joint Computer Conference. Montvale, New Jersey: AFIPS Press, 1972, vol. 1, 191-200.

[15] Wirth, N. "Program Development by Stepwise Refinement," Communications of the Association for Computing Machinery, vol. 14, no. 4 (April, 1971), 221-227.

[16] Hoffman, L.J. (ed.) Security and Privacy in Computer Systems. Los Angeles: Melville Publishing, 1973.

[17] Martin, J. Design of Man-Computer Dialogues. Englewood Cliffs, New Jersey: Prentice-Hall, 1973.

[18] Codd, E.F. "Recent Investigations in Relational Data Base Systems," Information Processing 74. Amsterdam: North Holland Publishing Company, 1974, 1017-1021 (preprints).

[19] Poole, P.C. and Waite, W.M. "Portability and Adaptability," in Advanced Course on Software Engineering, ed. M. Beckmann, et al. Berlin: Springer-Verlag, 1973, 183-277.

[20] Elliott, R.V., et al. "Computer Assisted Electrocardiography in Community Hospitals," in Computers and Biomedical Research, v. 4, ed. R.W. Stacy and B.D. Waxman. New York: Academic Press, 1974, 151-170.

[21] Klarman, H.E. "Application of Cost-Benefit Analysis to Health Care Systems Technology," in Technology and Health Care Systems in the 1980's, National Center for Health Services Research and Development, U.S. Dept. of HEW, 1973, Publication HSM 73-3016.

[22] Davis, L.S. "Problems Facing Large Health Information Systems," Proceedings of the 1973 ACM National Conference. New York: Association for Computing Machinery, 1973, 1-2.

[23] Blois, M.S. and Wasserman, A.I. "A Graduate Program in Medical Information Science," MEDINFO 1974. Amsterdam: North Holland Publishing Company, 1974, 217-222 (preprints).

[24] Anderson, J., et al. "Educational Requirements for Medical Informatics--Results of the First International Study," MEDINFO 1974. Amsterdam: North Holland Publishing Company, 1974, 207-211 (preprints).