# The Evolution of a Computerized Medical Information System

W. Ed Hammond and W. W. Stead

Duke University Medical Center
Durham, North Carolina

## Abstract

This paper presents the eighteen year history leading to the development of a computerized medical information system and discusses the factors which influenced its philosophy, design and implementation. This system, now called TMR, began as a single-user, tape-oriented minicomputer package and now exists as a multi-user, multi-database, multi-computer system capable of supporting a full range of users in both the inpatient and outpatient settings. The paper discusses why we did what we did, what worked, and what didn't work. Current projects are emphasized including networking and the integration of inpatient and outpatient functions into a single system. A theme of the paper is how hardware and software technological advancements, increasing sophistication of our users, our increasing experience, and just plain luck contributed to the success of TMR.

## Introduction.

Since 1968, we, at the Duke Medical Center, have been involved in the development and implementation of a computer system for use in patient care applications. Over that 18 year period, our progress has been influenced by hardware, software, and people. This paper presents the historical development of a clinical information system, now referred to as The Medical Record or TMR*, and discusses those factors which influenced philosophy, design and implementation. TMR grew out of a single-user, tape-oriented minicomputer package and now exists as a multi-user, multi-database, networked multi-computer system capable of supporting a wide variety of users in both inpatient and/or outpatient settings. This paper discusses the interaction and evolution of hardware technologies, system software technologies, and applications software capabilities and how those parameters meshed with the increasing sophistication of users, an expanding level of need and utilization along with the growing experience of the system designers. We include approaches which work along with those which did not. The current system is described, and future directions are identified.

* TMR is a registered trademark of Database, Inc.

## The Beginning.

We were introduced to the field of medical computing by becoming involved with an IBM cooperative project known as the Clinical Decision Support System (CDSS). Our part of this project was the development of an initial medical history which was designed as a mark-sense document with some narrative data. The history was scanned using an Optical Scanning Corporation DIGITEK 100 Optical Scanner with output available on nine-track magnetic tape. The narrative data was entered onto punched cards, and the tape, data cards, and program were delivered to the Triangle Universities Computation Center some 15 miles away. Problems encountered related to the number of steps involved in producing a processed history and included the inability to deliver tape, program, and cards simultaneously to TUCC and in alignment of a tape written on one vendor's product and read by another vendor's product. Frequently these problems resulted in the processed history not being available by the time the patient presented to the doctor. The timely gift of a Digital Equipment Corporation PDP-12 minicomputer solved these problems and influenced the initial approach to the computerized information system.

The PDP-12 minicomputer had a 4K 12-bit word memory, two DECtape units each with a 255 Kword storage capacity, a 12 by 24 character CRT screen, and a 10 cps teletype printer. LAP6/DIAL was the single user operating system. Since no adequate higher level language was available, the programs were written in assembly language. The definition of the database structure was influenced by CDSS. The history questions could be answered by



0195-4210/86/0000/0147$01.00 © 1986 IEEE

selecting one or more items from a list, by a numeric value, or by a text string. Having experienced annoyance at seeing names truncated with fixed-length fields, we elected to support variable length free text fields. The historical data also suggested a hierarchial database structure. The database design, which has remained basically unchanged over the past 18 years, is one which supports 3 bit data nodes plus text pointers and a variable length text buffer. The file structure designed supported variable length records blocked to the DECtape record of 128 words. Records were stored sequentially on tape. Individual records could be retrieved by scanning until the desired record was identified.

Programs were written specifically to perform the required tasks to process the medical history. The PDP12 was interfaced directly to the optical scanner, and a program (SCAN) was written to acquire this data, convert it to the desired data base format, and store the results. Another program (INPUT) was designed to accept text entry from the teletype keyboard in which the operator entered a node number followed by the text. Because of operator errors in typing the node number, a video, menu-driven program (FRAMES) was written to acquire this narrative data. A rather sophisticated output program (PRINT) was written to convert the coded data into a narrative report. This print program supported the hierarchial concept, provided concatenation of phrases, provided automatic punctuation, and supported Boolean logic in the selection of what phrases to print. This output was directed to the teletype and was only available in upper case. Record maintenance required certain housekeeping programs to be written to initialize, modify, delete and retrieve records. A single DEC tape could hold approximately 125 medical histories. With this system, a medical history could be processed within 4 minutes after receiving the history form. An updated version of these automated medical histories are still in use today.

## Interactive questionnaires.

With the success of the initial medical history, we decided to develop a series of speciality histories to capture information about current illnesses. The high degree of branching required by such a questionnaire made a paper-based mark sense approach undesirable. Consequently, a self-administered, computer real-time based, interactive questionnaire was developed. The initial questionnaire developed was for patients with functional headaches. The questionnaire included clinical symptoms, neurological manifestations, prior treatment, emotional factors, and personality problems. Questions were displayed on the CRT screen, and patients responded on the teletype keyboard. The system selected pertinent questions depending on the patient's individual responses. User-friendliness was provided by painting important keys, for example, RETURN was blue, DELETE was red, etc. The program FRAMES was expanded to support the variety of displays required to support the interactive questionnaires. Five basic frame types were introduced:

1) Information frame: conveys infromation to patient. No response is requested from the patient.

2) Free text frame: obtains free text or numeric data.

3) Exclusive response frame: obtains a single response from a displayed list (1-9 choices). Only one answer may be selected, but multiple database nodes may be assigned a value as a result of that selection. Branching to subsequent frames may also be dependent on the specific selection.

4) Multiple response frame: obtains none, one or several responses from a displayed list (1-9 choices). Both selected and not selected values can be stored in designated data base nodes.

5) Branch frame: looks at designated data base nodes, and using threshold logic, selects frames to be displayed.

An interactive questionnaire could be easily designed by defining the frame number and type, the text to be displayed, and the data base interactions and branching logic. These frame descriptions as well as the print rules were defined in a text source file called a "table". Programs were written, called "build" programs, which converted these text files to fixed-formatted, interpreted files in which parsing was done, offsets calculated, numbers converted to binary, etc. In other words, the build programs performed as much source processing as possible short of the interaction with the actual patient data.

A logical extension of the headache program was the introduction of some clinical decision making tools to assist in the diagnosis of the type of functional headache. Another program (MASSAGE) was introduced to provided decision-making logic, and the system was taught successfully to differentiate between common, classical and cluster migraine, muscle contraction, and other types of headaches. In spite of the fact that the interactive questionnaires had a demonstrated utility, they were not very popular at Duke. The neurologists felt that they did not need the support of a

computer to obtain a history and diagnose the headache. There was considerable interest outside of Duke, but the cost of a system (about $30,000) was prohibitive. We did support one community physician in Charlotte, NC with a telephone modem linkage, but the 30 minute telephone charge was still too expensive. We still do not use the computer to acquire interactive data from the patient.

At this point in time, the system provided few user aids. Programs were run independently, and the system flow often required going back and forth between programs. Errors encountered were displayed on the console lights in binary. Documentation for both system programs and users was nonexistent. As more people began to use the programs, it was necessary to develop some sort of system to link the programs together. An executive program was created to provide that linkage, and the system was named GEMISCH, an acronym for GEneralized Medical Information System for Community Health. Error messages were written to the screen as numbers, and the first user manual was written.

In order to extract data from the stored medical history database, a generalized, selective retrieval program (ISAR) was written to provide a query across the entire data base for patients with selected characteristics. This query program supported numerical comparisons, text comparisons, IF ... THEN clauses, talleys, and list of contents of data base nodes.

## Obstetrics Medical Record.

In 1971, we were asked develop a prenatal care record for the Division of Obstetrics at the Duke Medical Center. Three decisions influenced the design and implementation of this application. The first decision was to permit the physicians to continue to use check sheets for data collection and to use data processing technicians to enter the data into the computer. The output of the system would also have the appearance of the traditional medical note. The second decision was to use the existing data base system and modify the system as necessary to satisfy new requirements - i.e., an approach of evolution and modification rather than generation. The third decision was to commit immediately to the computerized forms and reports and not run in parallel with a manual system. Such a committment required strong departmental backing to suffer through early design mistakes.

The initial data collection was to be through a mark-sense questionnaire. The 19 page initial medical questionnaire along with additional sheets to acquire OB specific data was used to collect the data. A form was mailed to the patient to be returned and processed at the initial visit. For those patients capable of completing the form, about 80 %, the results were a rather detailed and complete medical history. Although the record was now legible, the resulting report was between 5 and 15 pages in length, and the doctors could not easily identify pertinent information. This deficiency led to the development of printouts in which the most inportant information was moved into a high-lighted area near the top of the printout. The automated history was redesigned and shortened to 5 pages.

The computerized medical record had little impact on the doctors until a teletype was added to the delivery suite, and the medical data on a patient was instantly available 24 hours per day, 7 days per week. The available printout was reformatted into an admission note, printing data available and leaving blanks to be filled in for data not yet available. The admitting physician completed these blanks, the data was entered into the computer, and the admission note was printed which the admitting physician then signed. For the first time, the computer reduced the doctor's work load and provided data that was frequently not otherwise available. The doctors also found that they could leave messages in the patient's record to be recalled when the patient came in for delivery or for special tasks to be performed.

Since the teletype was available to more people, we became concerned about security. Our first security measure was to print four numbers, three of which were the coefficients of a quadratic equation and the fourth number was a root which the physician was to plug into the equation and type in the result as the password. The system was so secure nobody could use it. Even random phrase passwords created problems. We finally settled for a single letter password which was prominently displayed above the terminal.

The impact of the OB system was felt through a tenfold increase in the daily volume. The active record file was now approximately 1000 records and required twelve tapes. Information had to be both collected and printed at several locations. We wrote an interrupt routine which would suspend normal data entry when a request for a record printout was made from the delivery suite - our first approach to multiple users. The computer was expected to be available at all times since it contained critical medical data.

## Migration to a disk-based system.

The next major event in the evolution of the system came with the purchase of a DEC PDP-11/20 with 28K of 16 bit word memory, a 2.4 megabyte movable head disk, a line printer, and a video terminal.

New programs were written in assembly language to facilitate record management and backup of data. Problems with data validation, missing data, changing ID's, and missing patients generated additional programs. For example, programs were written to highlight changes in a list of yesterday's patients compared with today's patients. A program was written to identify missing lab data. Procedure manuals and check lists for data entry technicians became mandatory. Monthly statistics using ISAR for the OB system replaced manual data keeping at a savings of people time. The system had become a part of the daily routine of the Division of Obstetrics. A variation of this system, along the lines of the traditional medical record, is still in operation at Duke.

In order to deal with the increased number of programs and the frequent switching from one program to another, we developed an executive program which permitted the various programs to be selected from a menu and permitted the automatic linking of programs to perform a defined set of tasks. Demands on the system increased with use and each of the programs continued to expand in functionality.

## Time-shared Operation System.

The demands on the single-user computer as both a developmental device as well as an operational device prompted us to write a multiple user operational system. Using DEC's DOS as a base, we wrote an multiple-user operating system which would support up to 7 simultaneous users in 28Kwords of memory. This new system, called Timed-shared Operating System (TOS), used a fixed-head disk for swapping users in and out of memory in a round-robin fashion. The TOS was turned on with an average of 10 crashes per day, which, amazingly, the operational users were willing to tolerate. The number of errors reduced to about one per week after about six months.

## Primary Care Record.

Throughout our 18 years' experiences, we seemed to reach plateaus upon which major new systems are built; 1974 was such a year. We began to develop a series of applications including a primary care medical record for use in the University Health Services Clinic (UHS), a primary care clinic serving students, employees, faculty and community residents. The data base structure included a base record for the patient demographics, a permanent problem list, current medications and treatments, previous hospitalizations, critical medical data, and patient encounter data. The data was entered from a check sheet by data entry clerks employed by the computer group, not the clinic. Although this system had at least some capacity for recording medical data, it was in fact used only to record administrative data necessary for management and financial decisions. The initial use of the system was primarily for the generation of daily, weekly, and monthly management reports. This computer system was more of an academic exercise and had little impact on the daily operation of the clinic and even less on the actual delivery of care. It was obvious that the system needed to do more if it were going to become an important component of the health care delivery system, and, of more importance to us, someone would be willing to pay for its use.

In order to increase the capabilities of the system, both new system programming features and applications had to be developed. The FRAMES program was combined with the MASSAGE program and additional features added. ISAR was expanded to include double precision arithmetic operations and enhanced logic statements, and included some of the functions provided by the PRINT program. ISAR began to look like a higher level language and a translator was written to preprocess both the ISAR and FRAME source programs. The file structure was built into ISAR and expanded to permit dynamic allocation of record storage. Space within the file was managed by assigning a bit to each block with a value of 0 indicating that block was being used. A single patient's record was stored sequentially, using a first-fit algorithm to find adequate space for that record. An index structure to permit direct retrieval of a single patient record was designed using a nine-digit identification code and the beginning block number of that record. Assumed random characteristics of the ID were used to provide additional selectivity for both the bit mapping and the index block selection. File capacity was expanded to 65,000 records.

As operational inefficiences were identified in the programs, new functions were added. CASE and SUBROUTINE statements and data arrays were added to ISAR. The index structure was expanded to include 12 bits which could be set under program control and could be used from the index to identify a subset of records with a defined characteristic. For the UHS system, the Social Security Number was

selected for the primary ID. A number of patients did not have a SSN, and a dummy number was assigned. These patients would forget that number, and on subsequent visits, that patient's record could not be retrieved. A hash-code program was written to convert a part of the patient's name into an index into a bucket from which the patient's ID and subsequently the patient's record could be retrieved.

The UHS Clinic system was expanded into a real-time system accommodating approximately 1000 encounters per week and, by the end of 1974, contained records of over 20,000 patients. A record contained detailed personal data, insurance data, primary care medical data, billing data, a clinic visit summary, and the details of the patient's last visit to the clinic. Whenever a new visit occurred, the clinic summary data would be updated, and the previous clinic visit data would be "plucked" to a seconday on-line file where it would be used for summary data retrieval. Complete billing capabilities were introduced, and both a video and a hard-copy itemized bill could be generated at the end of a patient's visit. An insurance claim was automatically printed at the end of the day. Patient's bills were generated monthly, and payments were entered back into the records as received. At any time, a copy of the patient's administrative summary or last clinic encounter could be printed. Since the computer data base was now the only copy of the financial record, it was necessary to insure that data could not be lost. A unique "back-up" procedure was developed in which a record was doubly stored when updated to both the primary file and a daily file, located on a different device. At the end of the day, all updated records were stored to a backup primary file thus creating a duplicate copy of all data. The daily file also permitted more efficient report generation of daily statistics and summaries.

A patient appointment system was introduced into both the UHS and the OB clinics. The system was designed to handle multiple clinics and required the ability to handle a patient record, the appointment record, and the clinic and MD schedules at the same time. Each of the clinics was handled as a separate clinic and controlled by the executive as though a single system. The appointment system provided various reports and schedules and generated patient reminders and medical chart requests. The no show rate was reduced by 30% in the OB clinic. Other applications introduced included a health-risk analysis, identification of high risk pregnancy factors, patient care protocols for various acute illnesses such as urinary tract infections and management of incomplete abortions, a tumor registry, and various financial, management and medical summary reports.

During 1974 and into 1975, we formalized the separate programs such as FRAMES, PRINT, ISAR, STORE etc. into a higher level language which we continued to call GEMISCH.

In 1975, supported by an NIH grant, the primary care record system was transfered to another promary care facility, the Family Medicine Center. Although the FMC system began as an identical system to that installed in the UHS clinic, within one month the programs had begun to diverge, and within 6 months the two programs were so different they had to be supported by different programming staffs. At the same time, we began to develop and implement a data base management system for the Renal Outpatient Clinic at Duke. Where the UHS and FMC systems were largely administrative with little medical data, the ROC system would contain predominately medical data.

Experiences from these three applications lead us to rewrite the automated medical record system in such a way that a single program would handle all three clinics. This new program had several features which we felt were significant: (1) modular construction which simplified programming, documentation, development, expansion, and maintenance; (2) data and application independent programming through data definition tables or dictionaries which permits the same program to be used in any clinic; (3) two input modes - a direct input for source entry of data and an indirect input using data entry personnel; (4) a variety of output formats, independent of data entry and data storage; and (5) the direct coupling of various protocols to data entry and data display. This applications program, which we called The Medical Record (TMR), has proved to be flexible enough to support all our applications since 1975.

Our hardware was expanded in 1975 with the purphase of a DEC PDP 11/45 with 120Kwords of memory, a 16 channel multiplexer, a high speed matrix printer, two 300 megabyte, dual-ported disk drives, and a 2.4 megabyte removable platter disk drive. We converted GEMISCH to RSX-11D and supported 16 simultaneous users. GEMISCH continued to evolve into a single powerful data base management language with full decimal and integer arithmetic capabilities and extensive text handling capabilities.

UHS and FMC continued to use the primary care record over the next two years as TMR was developed. By 1977, UHS had expanded to some 50,000 patient records with over 125,000 encounters per year. That clinic was supported by 6

video terminals and one printer. Approximately 3 minutes were required to register a new patient while a returning patient could be checked-in within 20 seconds. Timing data was used in both UHS and in the OB clinics to evaluate patient flow and efficiencies. The concept of an encounter summary was introduced into UHS and FMC clinics which included data from the last 12 encounters. A full financial package was introduced and accepted by the Duke auditors.

The FMC was supported remotely (5 miles) using dedicated telephone lines and short haul modems. Features were added to document and report clinical experiences of the family practice residents. The data base included 5800 records for 4500 families with approximately 20,000 yearly encounters. These activities were supported by three video terminals and one printer.

The introduction of the real-time medical information system immediately involved a number of people who had not previously been exposed to computers. Most responded with a negative attitude, distrust, and a resistance to the addition of the computer. Accuracy of data was a major concern and had to be insured before the clinical staff was willing to rely on the computer data base. We also pressured the clinics to assume operational responsibility for daily maintenance and backup.

During 1975, this project was formally evaluated by a team headed by Dr. Gio Wiederhold under contract to the National Center for Health Services Research and the evaluation was quite valuable in helping us to formalize where we were and where we were going.

TMR and the data dictionary concept continued to develop over the next two years. Recognizing inefficencies in the hierarchial data structure, we adopted a modular data storage in which data was stored into a problem module, a studies module, a therapy module or a subjective and physical examination module. Experiences had taught us that data entered as narrative was likely to contain misspellings or variations in expression which made it difficult for the computer to recognize that two entries were identical. Furthermore, the time required to enter text was longer and storage requirements were greater. At the same time, we recognized that total freedom of expression was required before the system would be accepted as a replacement for the manual record. The data dictionary along with an automatic text to code interpreter permitted us to code over 90% of the data. The data dictionary also allowed us to provide adequate flexibility to permit any clinic to define its own environment of providers, patient types, problems and coding schemes, studies, therapies, subjective and physical exam data, supplies. Cost data is included with each billable item.

In late 1977 TMR became a real operational system as UHS and later FMC were converted to this system. The development of the renal data base was shifted from Duke to the renal dialysis clinic at the Veterans Hospital in Durham. We were now supporting four clinical systems. A variety of individuals were involved in entering data into the computer including the doctor in a few limited situations. Log-on procedures had to be uncomplicated yet provide adequate security. Data entry must be simple, fast, unambigious and both flexible and at the same time structured to insure correctness. Mistakes must be easily corrected while maintaining an adequate audit trail. Systems must be supportable by clinic, not computer, personnel.

In 1978, the clinics added real time ordering of laboratory data. In the UHS clinic, no test was performed by the laboratory without a computer-printed requisition thereby eliminating lost charges and a significant increase in laboratory revenue. Xray ordering was later added with similar results. The appointment system was redesigned to increase flexibility, ease in use, and response time. The modular design approach permitted this redesign to be accomplished, tested, and then integrated into the clinical systems.

The next few years were ones of continued growth and refinement. Demands from the user community continued to drive development. By now, some of the patients had been in the system for several years, and particularly in the renal system had generated substantial amounts of data. We had developed a program to transfer patients to a purged file which was kept off-line for the UHS system to deal with students graduating and leaving the Durham area. However, in the case of the renal patients, we wanted the data to be available for instant retrieval so we could look at trend data. As a solution, we again relied on experience, and developed a "pluck" program which would extract data from the primary record and store that data in a forever expandable overflow record. Since we were likely to retrieve that data for a specific data type or even parameter, data was plucked into studies, subjective and physical examination, and accounting archive files. In each case, the data structure was identical to that in the main record, and the same programs could be used with both the main record and the archive records. Over time, we have learned that almost any function which needs to be performed with the current record needs to be performed with the archived records including data

entry and data correction. One patient in the renal system has had a Chem 18 collected weekly for the past 10 years. This data is stored in the current record plus three archived records. All data for a single parameter can be retrieved and displayed in about 30 seconds.

## Transfer to a non-academic setting.

By the end of 1980, TMR was operational in 6 different settings in the Duke Medical Center using 2 DEC PDP 11/45's and 2 PDP 11/40's and using RSX 11D and IAS operating systems. In January 1981 we installed TMR in a private internal medicine clinic, California Primary Physicians (CPP), located in downtown Los Angeles. This clinic was staffed by 33 doctors and 3 physician assistants and treated a patient population of approximately 8200. For the next 6 months, our programming activities were dominated by making this installation work. During this period, the content of the data dictionary expanded considerably to include a definition of the hardware configuration and data flow. For example, at the Duke installations, all documents were printed at a common point. At CPP, the patient flow was different, and documents were printed at different locations. We also became painfully aware of how often our programming skills were necessary to solve a local problem and how frequently we corrected the consequences of a problem rather than to correct the cause. We also discovered that we did not have a "system" but a series of programs that required user sophistication to select and run the proper programs at the proper time. One lesson that we learned and continue to learn is that each new implementation uncovers programming errors that have been present since the code was written but not yet encountered. Since we had to support this clinic across the country using a dial-up modem, it was necessary to tighten the code and make the system less people-dependent and more people-proof.

CPP has continued to grow along with TMR. The clinic has migrated from the PDP 11/45 on which the system was initially installed to a VAX 750, and in 1985, to a VAX 785/VAX 750 cluster. CPP now has supports over 100 providers and has a patient population in excess of 50,000. TMR, as a result of this experience, became more self-contained, more error-free, more complete and better documented.

## Adaptation to an inpatient environment.

As the number of Duke users increased, we upgraded to a VAX 780 and the VMS operating system. Other users at Duke converted to the PDP 11/44 and RSX 11M operating system. In both cases, GEMISCH was converted to the new operating system, and all application programs were immediately operational. The conversion of GEMISCH to native mode assembly language on the VAX took about one man month.

By 1983, TMR had been implemented in 10 sites, all of which were ambulatory care based. In April 1983, TMR was implemented at the Kenneth Norris Cancer Research Hospital to satisfy the informational needs of both an ambulatory care clinic and a sixty bed inpatient setting. Fortunately, the hospital had just been constructed, and the inpatient service grew very slowly. TMR was adapted to the obvious differences between the inpatient and outpatient settings. As a real-time outpatient system, TMR had required the resolution of all activities on the day on which they occurred. For the inpatient setting, the patient's "encounter" is for the length of the hospitalization. New functions were added to support an open encounter which automatically added the next day at midnight and moved certain charges forward. The biggest impact on TMR was the tremendous increase in the volume of data that accumulated for each patient. Data entry screens had to be redesigned to page for overflow. GEMISCH was modified to double the size of the current record. Once the system could handle the sheer volume and accommodate the basic requirements of an inpatient system, we began to focus on specific subcomponent needs. Six major groups use the patient record: administration, business office, laboratory, pharmacy, nurses, and doctors. Each group has a concept of what it wants independent of any other user. The system design must be capable of meeting those needs without comprimising other requirements. TMR has continued to evolve in this inpatient setting. We are now in the second or third round of addressing the needs of any specific group. An independent but integrated laboratory system was introduced in 1986. We estimate that TMR is about 50% complete in satisfying the desires and needs of an inpatient system.

## Data collection and report generation.

Since the late 1960's, the Division of Cardiology at Duke has been involved in the development of a natural history data base for patients with coronary disease. This research data base is used to provide diagnostic and prognostic profiles of new patients based on previous experiences. In 1984, this data base was converted to TMR in order to adapt the system into a real-time clinical setting and to expand easily into more clinical areas. The data collection capabilities of TMR at that time were limited to the selection of a

parameter and the entry of a result. Data entry could be grouped by category such as lab, and the user could be stepped through all entries for a given encounter. However, the cardiology entry requirements for such tests as a cath report required much more prompting, selection and branching features. TMR was modified to support dictionary-driven data entry based on test, patient problem, patient category, or protocol. The data was distributed by the system to the appropriate data storage module. Each data item in TMR was stored with a code identifying the item and with a date (Julian) and a value for each occurrence. Each parameter was stored in a different data base node along with all occurrences. For a cath report, often with several hundred parameters, this storage scheme was inefficient. Tests for which the components were rarely viewed independently and have the same date were stored in a new record linked to the primary record. This concept had already been introduced in TMR for the storage of past medications. The data collection screens permitted menu-driven data entry in which item selection could "stack" data entry frames for subsequent data entry.

Since GEMISCH had itself evolved as a report generator, no general purpose report generator was supported within TMR. All reports, including special reports, were programmed in GEMISCH. Cardiology required flexible report generation and did not want to depend on a programmer. A report generator was written in which a non-programmer can define the characteristics of the output using structured English. The report generator has gone through 4 generations and is now used for a number of applications including surgical operative notes. The report generator automatically generates a GEMISCH program.

## Microcomputer-based systems.

In 1984, with the introduction of DEC's Micro/PDP 11/23 microcomputer system, we felt that a medical information system was affordable for a small medical practice. TMR was installed in a three-person internal medicine clinic and in a four-person Ob/Gyn clinic. Both systems were installed with a 30 megabyte Winchester disk, four to six video terminals, two 100 cps printers, and the Micro/RSX operating system. No one at either clinic had any previous computer experience, and again we learned how to make the computer more self-reliant, self-protective, and self-performing. Both clinics continue to grow; both have expanded to a 70 megabyte disk in addition to the 30 megabyte disk, and both have increased the number of video terminals and printers.

## Networking and geographically distributed system.

In 1985, TMR was installed in a "loosely-connected" group practice which included five different clinics in two states, Pennsylvania and New York, and a hospital in which patients were admitted from all clinics. The maximum separation of sites is approximately 80 miles. The computer resides in one clinic with each additional clinic and the hospital connected by a pair of statistical multiplexers supporting up to four lines each and operating at 2400 baud over dial-up telephone lines. Since each clinic functioned independently for the patient encounter, the data dictionary had to be expanded to permit every possible output to be directed as a function of some other parameter such as place or revenue center.

## Looking to the future.

At the present time, TMR has been implemented in 25 settings including 10 locations outside of Duke. TMR is currently being implemented in a retirement community setting which required modifications to permit billing for non-medical events, such as guest meals; to accommodate monthly billing charges and to automatically create the next months encounter; to support waiting lists; and to handle billing for encounters at a number of "outside locations". Within this setting, clients may simply be residents, may use the medical clinic, may be in an extended care setting, or may be hospitalized.

At Duke, as new applications are added and existing applications mature, the need for communications and data exchange between systems has arisen. Ethernet provides the networking between the DEC computers and is used for both data exchange and system backup. One current application will network 8 computers and incorporate at least 5 distributed data bases. Linkages to the IBM-based Duke Hospital Information System (IBM's PCS) are provided through IBM PC's using an IRMA connection.

As more and more functions and applications are added to TMR, menus have become lengthy, often inappropriate and confusing. Users are interesting in doing the minimum amount of work to interact with the system. We are currently adding the capability to TMR to permit user-specific menus to be defined in the data dictionary and linked to a user identification code. Each primitive function of TMR is being converted into an independent subroutine or "tool". User specific pathways are then defined by specifying a sequence of these tools in the data dictionary. In addition, common

responses or limited selections, for example a set of lab tests that can be ordered, may also be defined.

Response time is perhaps the most critical factor in user acceptance of any system. Even with adequate computer power, a number of tasks must be performed in any user interaction which do not require user input. For example, the print out of a cath report after data entry does not require user input. We now extract these types of tasks and execute them through a computer "gnome" immediately freeing the user for the next user transaction.

We have introduced the concept of a workstation in which data is extracted from any available source and grouped for presentation and review. One example is the referring MD registry in which all patients referred by a specified MD and clinic are displayed along with date of referral, primary diagnosis, and critical tests performed. Additional data about any patient may be selected by a single button push. We anticipate this workstation concept to be expanded to link with national databases including bibliographic retrieval systems, to support on-line modelling and simulation, and to permit electronic-mail conferences with colleagues at other locations. Another example of a workstation function is the ability to plot over any time interval one or more data elements. These graphs can be expanded, critical events can be displayed, and mathematical functions can be performed such as inversions or ratios of two parameters.

TMR now supports a generalized query language which permits the selection of patient records satisfying certain defined conditions. As the volume of data continues to increase, the time required for retrieval increases to where some systems now require several hours for a pass of the total data base. These queries can be automatically performed overnight; however, there are some predictable queries in which an answer is required immediately. One example is to determine the most likely outcome of a given test for patients similar to this patient for a defined set of parameters. We are now implementing additional data storage in alternate formats including inverted files and position-defined, fixed-formated files to meet these needs.

A number of systems at Duke now contain detailed data on patients accumulated over a number of years. New approaches for extracting knowledge from these data are being explored. We are also exploring new ways for looking at data from a quality assurance perspective.

A number of the functional needs of an informational system can be met by existing programs. Examples are SAS for statistics or Knowledgeman for a spreadsheet. Rather than to develop our own similar packages we have elected to develop interfaces between TMR and these other programs. Communication can be automatic and is dictionary controlled.

## Summary.

Over the past 18 years we have evolved a clinical information system which supports a wide variety of applications in many very different settings. Our progress has been a function of the hardware and system software available to us; to what our users perceived as needs, what they wanted, or what they were willing to do; and how well we understood those needs and what we were capable of satisfying. We have always felt that luck was an important part of our progress as technological advances seemed often to get us out of trouble. This remark is perhaps most aptly illustrated by the advances in disk storage technology in volume, speed and cost. Early design decisions often proved to be fortunate to meet later demands. The evolution of TMR has also been strongly influenced by what we observed others to be doing. If someone else had solved a problem, we were quite willing to incorporate that idea into TMR.

In the remainder of this paper, we would like to summarize the key issues which we feel have been critical to the growth and success of TMR. Although we have been criticized for using a "non-standard" programming language, we feel that a large part of our success has been in the control of our language. Over the years, we have used over 10 operating systems. With each new operating system, all we have to do is to modify GEMISCH and all applications run in the new system. Changing computers has been equally easy, and TMR is operational on the full DEC line of mini and microcomputers. Controlling our own language has also permitted us to adapt quickly and easily to new requirements. Examples include terminal and printer independence where GEMISCH is able to compensate for non-standard control characters, and network interfacing. We also were able to solve problems of simultaneous record updating in a cluster configuration through GEMISCH. As TMR needs new functions, GEMISCH is modified to provide them. At least half of the current GEMISCH commands have evolved since the initial design. A typical example is the introduction of a function which identifies the number of items in a data base node separated by a defined

delimiter. This function was defined when we realized that it would replace 4 old instructions.

The simple file structure of GEMISCH has also proved to be an excellent design choice. Current data for a single patient is kept together in a single record and is brought into memory for review or update. Other patient data records are accessed automatically through a single linkage or common ID in related files. Other GEMISCH data structures such as fixed-length, directly addressable files, hash files, and sequential access files support additional TMR requirements.

Until the design of TMR beginning in 1975, we approached each new information system application as a new design and a new concept. Since we committed to TMR, all development has been carried out within its basic framework. The policy of allowing only one copy of the TMR source code has permitted a very small staff to develop applications for a number of different users. The modular design of TMR has permitted a controlled approach to growth and modification. Modules are reprogramed rather than patched for the most part. Each module has been reworked at an interval of about three years. The initial approach for some new requirement is to develop a specific program for that need; test and refine that program; and then generalize the function with control and definition provided through the data dictionary. Sometimes these needs are met through independent programs which are ultimately integrated into TMR proper; in other cases, the programs are integrated immediately into TMR but are conditionally-coded for that application.

Another critical design feature of TMR is the independence of data entry, data storage, and data presentation. TMR supports various presentations of data including problem-orientation, time-orientation, encounter-orientated, and a user-selected mode we call demand-orientation. Data may be viewed alone or with other data. It may be viewed numerically or graphically; it may be incorporated into a narrative report through the report generator. A group of data may be viewed for one date or over all dates. Data for a single patient may be presented, or data across a group of patients may be displayed.

There is an old saying, "What goes round, comes round." We are finding this saying to be increasingly true for TMR. For example, the dictionary definition of the data collection routines of TMR are very similar to the table frame definitions of the FRAMES program. Dictionary-defined, user-specific displays for function selection and data entry are similar to the specialized menus of early systems. The query language is similar in function and form to ISAR.

People make mistakes and any data entry must be correctable at any point in time. TMR has incorporated routines to permit users to correct any entry and automatically take steps to alter data as necessary in related modules and also to maintain audit trails as necessary. For example, if a patient's name is corrected, TMR will automatically correct the name in the backup data base and in the appointment system, other files or e, TMR automatically adjusts accounting entries, makes adjustments as data, TMR automatically adjusts accounting entries, makes adjustments as necessary and refiles insurance if appropriate. Nightly, TMR passes the entire data base and evaluates the integrity of the data base by comparing details of today's accounting transactions to previous transactions and insures that both totals and details are correct. If an error is detected, TMR saves the before and after records and performs a series of tests designed to identify the specific error. We are currently designing similar data integrity checks for medical data.

TMR has developed in real clinical environments. Each group of users thus far has driven development at one time or another. Each new user not only builds on the current capabilities but also helps to refine those capabilities. We have been frequently surprised with the similarity of needs and use of the system across what might be assumed to be very different users.

We have no doubts that TMR will continue to evolve if for no other reason than technological advancements. The motivations will be the same: ease and clarity of use, reliability and dependability, availability and completeness, and appropriate functionality.