Evaluating a Speech Interface System for an ICU

Roland El-Khoury
B. Eng., (University of Texas at Austin), 1987

Department of Electrical Engineering
McGill University, Montréal
March, 1994

A thesis submitted to the Faculty of Graduate Studies and Research
in partial fulfillment of the requirements for the degree of
Master of Engineering

Abstract

This Thesis presents the development and evaluation of a speech I/O interface system for a bedside data entry application in an intensive care unit. The speech I/O interface consists of a speech recognition input and a speech generation output. The main objective is to permit users to perform "hands-free" and "eyes-free" data entry. This thesis begins with a literature survey of patient data management systems, speech applications, the difficulties and the key factors of speech evaluations. This is followed by an overview of the current system. The design and the implementation of the speech interface system are described. The evaluation scheme and test results are presented and discussed followed by an outline of future work and recommendations for the system.

Résumé

La présente thèse traite du développement et de l'évaluation d'un système d'interface d'entrée/sortie basé sur la voix, système conçu pour permettre l'entrée de donnés prises au chevet du lit dans une unité de soins intensifs. L'interface est constituée d'un module de reconnaissance vocale pour l'entrée, et d'un module de génération vocale pour la sortie. L'objectife visé est de rendre possible l'entrée de donnés sans l'aide ni des mains, ni des yeux. Cette thèse commence par un survol des travaux traitant des systèmes informatiques médicaux, des applications faites des technologies vocales, ainsi que des ouvrages étudiant les difficultés et les points important relatifs à l'évaluation de ces technologies. L'on retrouve ensuite une présentation d'ensemble du système à l'étude. Une description de la conception et de l'implémentation de l'interface vocale est donnée. Ensuite, la méthode d'évaluation ainsi que les résultas de nos tests sont présentés et analysés. Enfin, nous présentons une esquisse des travaux à venir et nous fournissons nos propres recommendations quant au système à l'étude.

Acknowledgements

I would like to express my sincere gratitude to my supervisor Dr. Alfred Malowany for his invaluable advice, support, and above all his encouragement throughout this work, especially in the preparation of this thesis.

I am grateful to Mr. Franco Carnevale and Dr. Ron Gottesman of the Montreal Children's Hospital for their assistance and helpful suggestions.

Thanks are also given to all the members of the PDMS group and especially to my colleague Marco Petroni, who began the Fluid Balance system development and who, in many ways, helped me to continue and finish my work.

Above all, special thanks to my wife Claire and my son Jonathan for their unlimited support and understanding throughout this long and demanding journey.

Table of Contents

List of Figures

List of Tables

Chapter 1 Introduction

With the appearance of personal and minicomputers, medical-computing activity broadened and accelerated in scope. These machines made it possible for individual departments or organizational units, such as the Intensive Care Unit (ICU), to acquire their own dedicated computers and to develop their own application systems. These systems offer a wide range of services which are indispensable to the health-care professionals.

This chapter first discusses the motives and the objectives of a patient data management system in section 1.1.1. Human interaction with the machine is vital to using and accepting a system. Section 1.1.2 presents the key factors to a successful system in the medical domain. The resulting medical capabilities of using computers in the future are discussed in section 1.1.3.

Speech I/O interface is a more natural and preferable way to interact with computers. However, speech interface may become a must when mobility is required or when hands and eyes are busy. The speech technology and its categories and features are discussed in section 1.2.1. Speech applications as applied in and outside the medical domain as well as their success are outlined in section 1.2.2.

Speech evaluation methodologies are difficult to establish because of the immaturity of this domain. Human factors also play a major role in speech I/O systems. Section 1.2.3 elaborates on the factors of evaluation and on the best

solution to human factors. Then, section 1.2.4 presents the speech I/O systems of the future and the current research component for a conversational computer system. Finally, an overview of the thesis is presented on the last section of this chapter.

1.1 Patient Data Management System

1.1.1 Motives and Objectives

Intensive care units can be considered as good prototypes of medical departments since they are characterized by the significant accumulation of medical information, the intensive use of modern technologies [Duerer, 1992], and exceptional demands on physicians and nurses. These factors and other motives accentuate the need for effective use of information technologies and in particular, the use of patient information management systems.

The first, and perhaps one of the most significant factors, is that of technology: both hardware and software. During the past two decades, there has been a rapid development in computer technology, with a remarkable reduction in cost and size of computers, substantial improvements in speed and power [Tong et al.,1982], and tremendous evolution from single tasking and unfriendly mainframes to highly interactive, multiuser mini and personal computers.

A second factor is the motivation of the people involved: both the developers and the users [Hammond, 1987]. The developers' interest is to apply their

knowledge and to create a new market place whereas the users' main interest is to bring the ICU management under better control.

Economic motives also influenced progress. While the cost of computers has been decreasing, the cost of delivering patient care has continued to increase. Computers seem to offer one way to reduce and control these costs [Gardner, 1989].

Another motivation is the tremendous increase in the amount of data generated, either by the monitoring equipments, collected through frequent observations, or ordered from the laboratories, as well as the demand for that data by a variety of individuals [Wiederhold et al., 1990].

Finally, the influence of external factors such as the government and third-party payers has contributed significantly to the development of patient management systems.

Different PDMSs have been created over the years to satisfy the above needs in the ICUs. However, all of them were centered around the following objectives: 1) increasing the availability and accuracy of data, 2) computing derived variables that could not be measured directly, 3) allowing display of the time trend of patient data, and 4) increasing patient-care efficacy.

1.1.2 Keys to Successful Systems

The success of a system depends not only on the system's ability to meet the informational needs of its users, but also on the manner in which human and machine interact [Perrault, 1990]. A high proportion of PDMS attempts have

floundered, due in part to insufficient attention being paid to the human factors issues involved in system introduction and design [Green et al., 1991]. Human factors are very relevant to the design and introduction of computer systems, and to the evaluation of such systems within their operational environment.

Gleaser and Thomas summarized the criteria for a truly effective ICU system as follows: 1) output from the system must provide, at the bedside, information which is relevant, accurate, reliable, accessible and familiar; 2) the output must be appropriate to local procedures, styles of clinical case and personnel allocation; 3) retrieval of information must be simple, even for occasional users; 4) data input must be accurate and valid without disrupting patient care routines; 5) the system will only work if it is introduced without staff opposition and if users see that the system is of value to them [Glaeser et al., 1975]. These points are still valid and provide a useful benchmark for the discussion of the human factors issues involved in computerization in ICUs.

To realize such systems, it is imperative to establish a communication channel between the health-care and computer professionals as early as the start of the design stage, in defining problems and developing successful solutions. Overcoming staff resistance is a key element. It is noted by Blum that the developers of the PROMIS system believe that the essential prerequisite for an effective application is effective leadership [Blum, 1986]. Leadership within the medical staff are crucial. Effective training and assistance are vital. Gradual introduction of computerized systems is essential [Hodge, 1987] [Manzano et al., 1980]. When nurses and doctors see their own ideas and feedback implemented in the system, and attention is given to the advantages rather than the limitations of computers, resistance is overcome.

1.1.3 Future Outlook

The opportunities for medical informatics to grow and meet the challenges of the future derive from several sources. These include 1) the rapid growth of awareness and interest in computers and information-management systems by health professionals of all types, 2) the growing usefulness of medical information systems for helping with biomedical research and health-care problems, and 3) the continuing rapid development of the technological base so that these systems can be made increasingly useful [NLM, 1986] [Fagan et al., 1990].

The capabilities expected within the next 20 years as predicted in the literature include:

- Networking technologies that will allow individual machines to share information and work in unison when necessary, while permitting independence when untethered processing is more appropriate. Telemedicine with remote consultation and collaborative maintenance of on-line manuscripts will be the norms.

- Progress in speech understanding and speech generation suggests that systems will be able to converse with users using natural language in the years ahead [Lange, 1990].

- Mass storage techniques, with methods for distributed data processing are making feasible the notion of large, nationwide health data bases. With ever-improving techniques for maintaining data privacy, integrity, and confidentiality, it has become practical to mount efforts to

standardize data formats so that health assessment, planning, and focused clinical trials can be greatly facilitated.

- Decision-support technologies, including expert systems and large data base tools capable of providing better diagnosis, treatment, and patient management [AAMC, 1986].

Unlike humans who can tolerate ambiguity and make powerful use of metaphors, computers require uniform language standards. Establishing a Unified Medical Language System [Humphreys et al., 1989] [McCray, 1989] that could assure some uniformity of information gathered at multiple institutions across the country is a challenge to overcome [NLM, 1986]. Although considerable progress has been made in the areas of automatic indexing, learning, and drawing knowledge from examples [Michalski et al., 1983], the technology is still embryonic. Most of the knowledge bases in use have been constructed by hand, in an extremely tedious and time consuming process. For example, the INTERNIST-1 knowledge base [Miller et al., 1986] has already consumed approximately 30 person-years of work from expert researchers in the knowledge-base methodology [NLM, 1986]. Few such expert systems are available and their development is time consuming.

1.2 Speech I/O Interfaces

Speech I/O computer interfaces offer the advantages of hands free or eyes busy operation [Schneiderman, 1987]. This can be very important in medicine for allowing doctors and nurses to use computers when their hands or eyes are

busy. In addition, and according to Flanagan, both mobility and remote access can be achieved with speech I/O interfaces [Flanagan et al., 1990]. In medical technology, there are many examples where at least one of these conditions is given [Robbins et al., 1987] [Liang et al., 1989]

Speech I/O systems are spreading over a wide range of applications. They are beginning to be recognized as affordable and efficient alternatives to the mouse and keyboard [Karl, 1991]. The growing interest in such systems is due not only to the technological advances in speech processing but also to the continuous strife of man to make his communication with machines as natural as it can be.

1.2.1 Speech Processing Categories

The technologies for capturing, representing, recognizing, conveying and reconstituting spoken information have traditionally been termed "speech processing". The technical focus in speech processing has shifted and broadened over time. Speech processing can be divided into three major sectors: speech coding which primarily concerns human-to-human voice communications (even with a machine intermediary); speech recognition which relates mainly human-to-machine communications; and speech synthesis which is concerned with machine-to-human communications.

Speech Coding

The objective in speech coding is to analyze and represent (encode) speech as a digital signal that uses the smallest amount of transmission capacity required to

recreate (decode) the speech at the receiver. Ideally, the resulting auditory quality should be comparable to the original. The early work in speech coding can be traced back to the late thirties by Dudley [Dudley, 1939a] [Dudley, 1939b]. The rapid development of VLSI and signal processing technologies, in the last two decades, provided an open route to many applications. An 8 KBPS speech coding algorithm [Gerson et al., 1990] has already been selected for the dual-mode air-interface standard (IS-54 North American Standard) for digital cellular phones. Nikil et al. describe the coding of wideband speech for store-and-forward voice mail offered in many PBXs today [Nikil et al., 1990]. Many coding algorithms such as scalar (Pulse Code Modulation, Adaptive Differential Pulse Code Modulation) and vector (Vector Quantization. Linear Predictive Coding, Waveform Coders), are presented in the literature [Cummiskey et al., 1973] [Jayant et al., 1984] [McCree et al., 1991] [Smith et al., 1986]. The current research objective in speech coding is to achieve high quality at and below 4.8 KBPS.

Speech Recognition

Speech recognition systems are classified according to the following modes and characteristics: recognition mode (speaker dependent or independent), input mode (isolated word versus continuous speech), and vocabulary size (small or large).

In speaker dependent recognition, the system is trained to recognize speech from a particular person. This type of system is suitable for a small set of specific people and the requirement for memory space is linearly related to the number of speakers. Speaker independent recognizers are designed to recognize speech

from a large population of speakers. Clearly, the recognition is more difficult than speaker dependent recognition, particularly if the speaker population represents many dialects and accents. Recognition of connected words (continuous speech) is also substantially more difficult than isolated word recognition. This is mainly because boundaries in a sequence of connected words are often difficult to identify, and co-articulatory effects can dramatically influence the character of a particular word. For example consider the connected digit string "eight eight nine". When this digit string is spoken, the stop consonant at the end of the second digit is often not pronounced. The size and the nature of the vocabulary limits the performance of the recognition system. This limitation is due to the increased complexity and the difficulty in distinguishing between similar sounding phonemes like b, d, g, etc. Spanias in his review added other attributes that affect speech recognition performance namely channel characteristics and background noise. He claims that wider bandwidth of the channel provides a richer set of acoustic features. Thus, practical recognition applications require that recognition algorithms be robust to channel and environmental characteristics as well as human factor related problems [Spanias et al., 1992].

Of the various approaches to automatic speech recognition, two techniques have achieved a very high performance rate, namely the dynamic time wraping (DTW) and the hidden Markov models (HMM). In the DTW approach, the unknown utterance feature set is matched against the stored patterns using a procedure that dynamically alters the time dimension to optimize the accumulated score for each pattern. This makes the recognition process insensitive to variation in talking rate. Myers et al. detail a performance tradeoffs utilizing the DTW approach [Myers et al., 1980]. In the HMM

approach, instead of directly comparing the unknown utterance to the known examples, HMM creates stochastic models from the known utterances and compares the probability that the unknown utterance was generated by each model [Rabiner et al., 1986] [Poritz, 1988]. The pattern matching component of the recognition system uses stored HMMs to determine the phoneme string that is most likely to have given rise to the sequence of observation vectors for the word.

Another method for automatic speech recognition is the connectionist approach. Lee et al. have reported a fully parallel mandarin speech recognition system with very large vocabulary and almost unlimited texts based on the neural-net approach combined with the emergence of parallel processing architectures [Lee et al., 1991].

Speech Synthesis

The objective in speech synthesis is to provide a broad speaking capability to computers. In its simplest form, the machine just plays back one of several stored messages. Messages are recorded and stored digitally, sometimes after compression to save memory space. With added complexity, more versatile announcements can be composed from a library of message parts - sentences, phrases or words. Message content is constrained by the size and flexibility of this library. Josenhans et al. discuss in more detail the speech processing application standards and the methods involved in speech synthesis [Josenhans et al., 1986]. Ideally, a talking machine should be able to convert any arbitrary text message into its spoken form without the restriction of a pre-stored library of encoded phrases. Hirschberg et al. show that the research frontier in text

synthesis is in achieving human-like quality, varied voices, dialects and languages, and ultimately, in understanding the exquisite details needed to duplicate the voice characteristics of an individual [Hirschberg et al., 1990].

Three important dimensions are evident in speech synthesis: speech quality, message versatility and complexity. The desired objective is high quality and versatility with low complexity [Crochiere et al., 1986]. In reality, simple systems that store recorded waveforms have high quality and low complexity, but they have also low versatility; i.e. the messages can be used only in the form recorded. At the other extreme, text-to-speech systems that are based on computational synthesis have great versatility but also have high complexity and, as yet, limited quality [O'Malley, 1990].

1.2.2 Speech Applications

In the medical domain, examples can be found of using spoken commands to get actions or to produce medical documents. A 5000-word speech input system is used for the generation of radiological reports. This system had sufficient capacity, enough to include all radiological examinations and all desired technical words. The system is used over 88% of the time and its reliability in word recognition was 98% [Robbins et al., 1988]. Due to the speech interface, the medical reports are prepared directly on line. There is no need to review post dictation transcription of the radiologist reports for approval. The reports may be easily integrated into a centralized hospital information system [Rowberg, 1987] [Haseman, 1988] [Rosen, 1988], or into expert systems [Wulfman, 1988].

Speech input may not only be used for information collection but also for direct control of machines. Speech commands for microscopes in microsurgery have become rather common in medical applications for some years [Liang et al., 1988]. In this application the main characteristics to be controlled are focus, x-y-coordinates and zoom. Although the vocabulary may be rather small, in the range of 10 to 20 words, it performs the required tasks precisely and efficiently. Liang concludes in his study that the voice control has eliminated the need for the cumbersome foot paddles and proximity switches [Liang et al., 1988].

In another example, a voice recognition system eliminates the need for assistance every time the patient wants to re-position the bed settings. Liang describes a speech recognition system that controls the settings of a robotic bed [Liang et al., 1989]. Mangold's review describes various speech input recognition and voice generation systems for aiding handicapped people [Mangold, 1988]. These range from entertainment systems, wheelchairs, talking calculators, reading machines, speech training and many more that help in the activities of daily living [Rash, 1989].

A speech interface combined with graphics was developed at Stanford University to facilitate the entry of the complex medical terminology into a medical diagnostic system. The patient information is entered on a body diagram associated with locations. The system then submits the findings into the Quick Medical Reference Decision Theoretic, which performs diagnostic reasoning about diseases in internal medicine [Shwe, 1990]. The study shows that due to the user friendly speech interface, doctors are consulting the diagnostic system on a regular basis in the routine clinical settings [Shiffman et al., 1991].

Applications of speech generation in the medical domain are not as numerous as the applications of speech recognitions. An application discussed by Dintruff uses voice synthesis to give flexible guidance through the screening system of preparing medical and patient records in emergency situations [Dintruff et al., 1987].

Studies performed at the University of Alberta hospital showed that the use of non-verbal warnings such as beepers were perceived to be threatening by some patients. McIntyre et al. present a system in which the non-verbal warnings were substituted by human-voice messages that led to better error interpretation by nurses and a more relaxed atmosphere for patients. The studies concluded that taped voices message merit more trial in ICUs [McIntyre et al., 1989].

Speech I/O systems are also intensively used outside the medical domain. Peacocke et al describe a few different environments where the speech I/O systems have proven their usefulness for certain applications. Delco Electronics uses a recognition system to collect circuit board inspection data while the operator repairs and marks each board. Southern Pacific Railway inspectors now routinely use a PC-based recognition system to enter car inspection information from the field by walkie-talkie [Peacocke et al., 1990]. Another system that is worth mentioning, is described by Nakatsu. It is called automatic Answer Network System for Electrical Requests (ANSER). The system employs speaker-independent speech recognition and speech synthesis interfaces over the telephone network. Since its introduction in 1981, the system has provided information services for the banking industry. Most Japanese banks use ANSER, serving customers at a rate of several hundred thousand calls per day. The system has also been introduced in the securities industry [Nakatsu, 1990].

Speaker identification and verification applications have been implemented by a number of companies in an effort to increase credit-card security. US-Sprint communications tested a system for credit card calls that verifies the caller's voice based on matching a requested spoken password with the caller's voice print. On the other hand, Bell Communications Research developed a speaker identification system that stores a voice print (digitized spoken password) in an integrated circuit on a credit card so that the caller can be identified by comparing template to a spoken password [Andreas et al., 1992] [Lennig, 1989]. A good article that describes current applications of speech I/O systems over the telephone network is treated by Karis et al. [Karis et al., 1991].

1.2.3 Evaluation Factors in Speech I/O Systems

Evaluation methodologies of man-machine voice interaction are difficult to establish [Delogu et al., 1991]. The difficulty arises from many intrinsic sources; the technology itself, the application at hand, and most importantly, the human factors involved.

The most reported measure of performance of a speech recognition interface is given in terms of the percentage of utterances which are correctly recognized. According to Chollet et al. and Bergeron et al., recognition performance or accuracy is a function of the size, syntax, and complexity of the vocabulary, the quality of the audio environment including the bandwidth and level of ambient noise, the amount of distortion of user's speech, and interpersonal variation in speech [Chollet et al., 1988] [Bergeron et al., 1990].

To assess a speech recognition interface, Danielsen et al. present an acoustical database gathered from many speakers. This database of acoustically similar words is being used to test for substitution errors which constitute the most common recognition errors [Danielsen et al., 1989].

Waterworth describes the problem of particular speakers used both in training and in the test phase as an important factor in the evaluation process. There is a considerable between-speaker variation in recognition figures, especially due to the sex of the speaker. Females are generally less well recognized than males [Waterworth, 1984].

Another interesting study on the effect of noise on speech recognition performance and speech quality has been carried out at the TNO Institute for Perception in Soesterberg, The Netherlands [Steeneken et al , 1988]. They have developed a noise database with 18 different noises arising in military environments, recorded from various noise sources such as jet-planes, helicopters, tanks and command rooms. This database can be used to simulate the acoustical room environment of the speaker or of the listener.

With regard to voice generation output, the performance evaluation measure normally used is the "quality". This kind of measure is rather abstract and vague for many reasons. Subjective judgments of quality vary from person to person, and the quality of speech depends highly on the application at hand [Delogu et al., 1991]. For example, a synthesized voice that may be judged acceptable for a child's toy may not be considered suitable for giving landing or takeoff instructions to a pilot in an aircraft cockpit [Carlson et al., 1989]. However, based on more precise definitions of synthetic properties, different measures have been proposed in the literature, such as intelligibility (how identifiable is

the linguistic message), naturalness (how much the voice output is like a natural voice), and acceptability (the overall user's satisfaction in a specific task) [Voiers, 1983] [Jekosch, 1989].

Human factors also contribute to the evaluation process of the speech I/O interface. In a study, Delogu et al. have shown that the error rate escalated from about 4% in laboratory tests to about 14% in the field. She states that after an error, the subjects altered their voice, giving rise to more errors, in a loop of errors-alteration-errors known as the "exacerbation process" [Delogu et al., 1991]. Research has shown that people speak very differently in a highly noisy environment than in more quiet conditions [Rollings, 1985].

In general, human factor studies can be very useful in boosting the speech I/O interface performance and determining guidelines for developing voice interface designs. At the Speech Research Laboratory of Indiana University, research into the role of the user in a man-machine interaction with voice devices has demonstrated that experience and learning are the most important factors in improving the speaker's and the listener's performance. Interestingly, this improvement in performance is primarily a function of specific training [Schwab et al., 1985].

1.2.4 Future Outlook

With the recent improvements in speech and other related technologies, many prototype applications begin to identify and shape the speech interface and the computer capabilities of the future. Perhaps, the ultimate goal of automatic

speech recognition and voice generation is to allow natural, effortless communication between humans and machines.

The HuManNet system which is an experimental system built by Berkley et al., demonstrates how the technologies of voice, image, data, computing and teleconferencing can be combined in a sophisticated information system to accomplish office operations that are natural, speech driven, and easy for people to use [Berkley et al., 1990].

A prototype application by Morishima et al., in the context of intelligent human-machine interface, synchronizes the speech synthesis output with an automatic facial motion image animation in a real-time environment [Morishima et al., 1991]. The interface is capable of giving computers a sense of emotions by imposing facial characteristics such as sadness, and happiness depending on the speech synthesis context.

New understanding in acoustics, inexpensive transducers and sophisticated microelectronics combine to support auto-directive speech-seeking beam-forming microphone arrays for hands-free audio conferencing. The system can dynamically locate, follow and lock onto active talkers as they move to obtain a high quality signal in the presence of reverberation and noise, thus providing natural-voice interactive environment systems [Flanagan et al., 1985].

Karis and Dobroth detail the requirements of future systems (conversional computer systems), which will permit more natural communication based on needed advances in speech recognition, language comprehension, conversional dynamics, discourse structure and speech production [Karis et al., 1991]. Although Karis conversional systems are still subjects of future researches, Allen

has shown that under constrained domains, natural language interfaces work well and efficiently [Allen, 1987].

1.3 Thesis Overview

This thesis will present a speech interface evaluation for a computerized Fluid Balance Management System in a pediatric intensive care unit. Chapter 2 describes the patient data management system project, including a presentation of the hardware and software architectures. The software modules and their functionalities are discussed.

Chapter 3 presents a global description of the Fluid Balance software design and a detailed explanation of its sub-systems. The second section focuses on the implementation issues and the integration of its components.

Chapter 4 presents a three-stage evaluation scheme used to test and evaluate the speech interface and the Fluid Balance system. The criteria, purposes, and test procedures are outlined. The evaluation results are discussed followed by the recommendations for future improvement before concluding the thesis in Chapter 5.

Chapter 2 Description of the Patient Data Management System

This chapter presents an overall description of the Patient Data Management System (PDMS). The PDMS is an ongoing project being developed at McGill University for the pediatric Intensive Care Unit (ICU) of the Montreal Children's Hospital. The PDMS is a personal computer-based real-time information system aimed at providing full-scale support for the ICU, ranging from documentation to improved decision making. The PDMS spectrum of functionalities spreads over a wide range of every day needs in the intensive care for data acquisition, surveillance, data analysis, record keeping, information storage and retrieval, and decision support.

This chapter presents the PDMS hardware and software architectures. Then, the system functionalities and modules are described.

2.1 The PDMS as a Custom-Designed Project

Many commercial systems exist in the market requiring only a simple "turn of a key" installation to make them operational, whereas custom-designed systems are expensive and involve a long interactive cycle of development. The problem with commercial systems is that they rarely perfectly match an

institution's information-management needs. The system may not perform all the desired functions, may provide superfluous features, or may require some reorganization and modification of responsibilities and established flows of information within the institution.

Having to choose a system and after analyzing the functional compatibilities of the commercial systems, researchers at the Montreal Children's Hospital have decided to sponsor the PDMS to be a custom-designed project.

The design and development of the PDMS has been in progress since 1988. Many new enhancements and functionalities have been added throughout the design and the implementation span of the project. These enhancements were introduced as well-defined problems by the health-care professionals and their corresponding workable solutions have become the functional features of the system. Their principal objectives have always been to accelerate the collection, processing and presentation of patient data and to reduce the workload of the ICU staff.

2.2 The Hardware Architecture

The PDMS hardware configuration consists mainly of two sub-systems; the Hewlett-Packard CareNet sub-system which provides a local area network acquiring the electronically physiological data and the host sub-system which provides the environment where the PDMS software package is running. Figure 2.1 outlines the complete system configuration.

In order to acquire the data of a patient, each bed in the ICU is equipped with a HP78532A Physiological Monitor/Terminal that is linked to a Network System Communication Controller (NSCC) HP78581A. The NSCC is the central unit that connects all the fourteen bedside monitors in a star configuration network architecture. Furthermore, to interface and establish a link between the two sub-systems, a HP78580A Careport unit is attached to one branch of the NSCC. The Careport is a programmable interface unit acting as a presentation layer that translates the requests/commands of the PDMS host sub-system into a proper CareNet format and transforms the delivered data into the format expected by the PDMS. The communication protocol used between the two sub-systems is implemented via the standard RS-232-C serial interface.

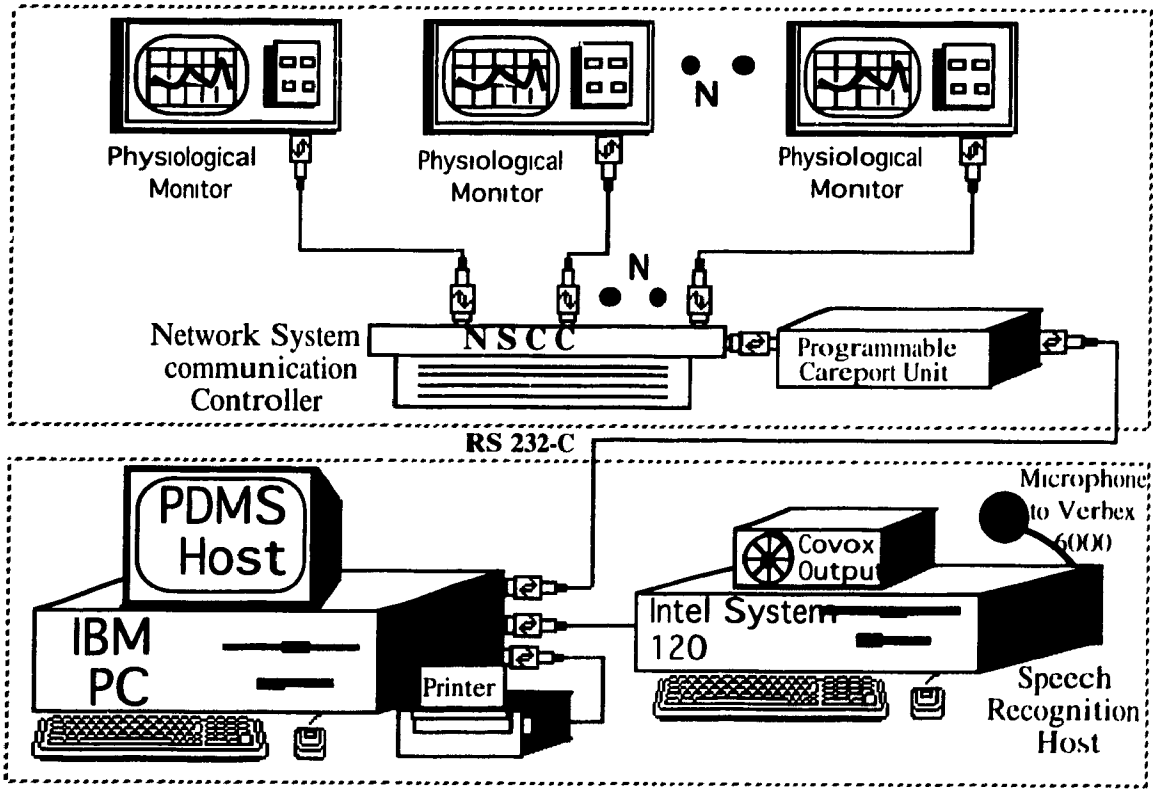

Figure 2.1: The PDMS Hardware Configuration

The current PDMS setup at the hospital consists of an IBM PC with 486 CPU, 16 Mbytes of RAM memory, a 300 Mbyte hard disk, an 8514 color display and an EPSON LQ-2550 color printer. This host runs all the software modules of the PDMS under the OS/2 multi-tasking operating system version 2.0 and uses the Presentation Manager functionalities for the user interfaces.

The Intel system 120 running DOS version 5.0 is used to accommodate the speech recognition hardware; the Verbex Series 6000 board and the Covox Voice Master Key II. This host is used to provide the speech interface to the Fluid Balance module as will be explained later in chapter 3. The configuration is currently being extended for a distributed network of personal computers using IBM LAN manager.

2.3 The Software Architecture

The initial development of the PDMS started on an IBM Personal Computer running the MS-DOS operating system. The lack of real-time capabilities significantly complicated its development and this approach was abandoned.

In 1988, IBM and Microsoft released the OS/2 version 1.0 operating system and, consequently, a new design of the PDMS was proposed [Panisset, 1989] which revised the user interface completely and exploited the power of the new multi-tasking environment. The PDMS became a collection of independent processes each managing a specific part of the overall design. These processes interacted with each other through the operating system primitives such as pipes, semaphore, and shared memory segments. With the active involvement of the

medical staff in defining problems and developing workable solutions, several modules were produced and are currently running in the hospital. Figure 2.2 shows the software modules as well as the architecture of their interaction. All the program modules except for the Data Link Controller, interact with the user by generating screen displays.

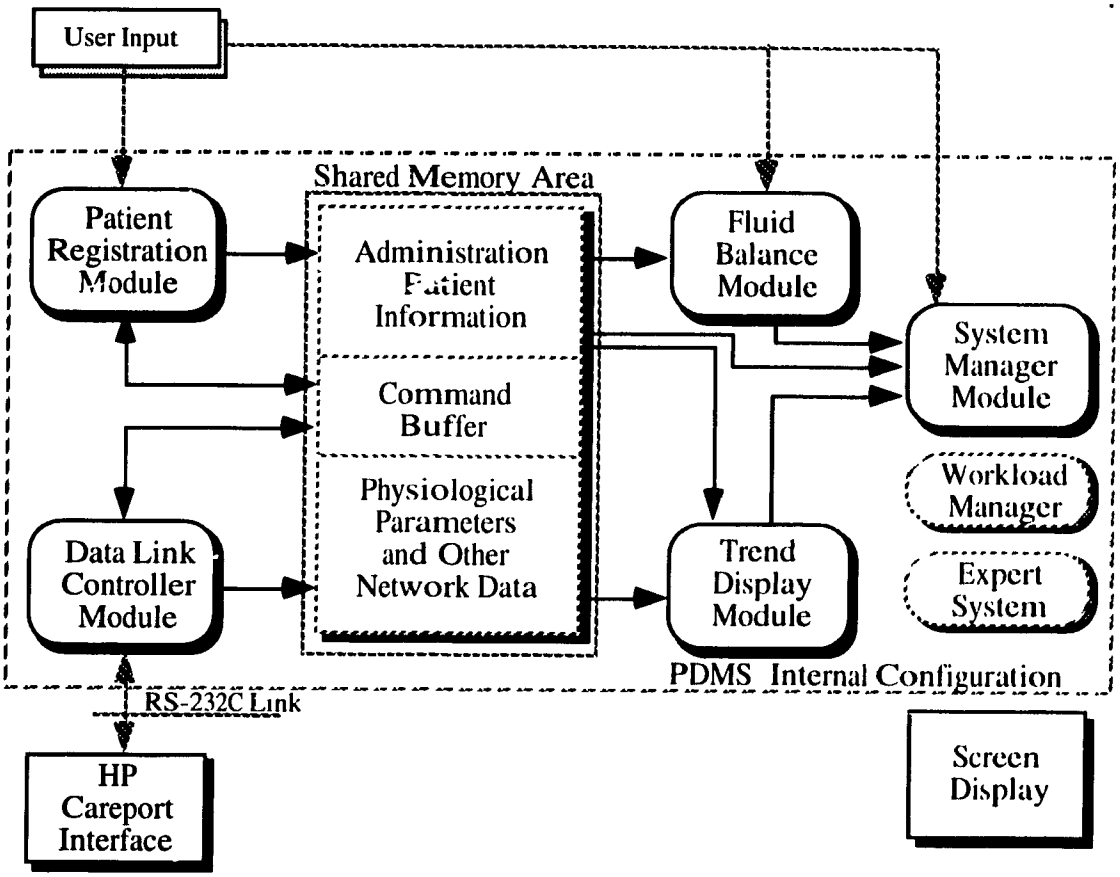

Figure 2.2: The PDMS Software Architecture

2.4 The PDMS Functionalities and Modules

2.4.1 Data Acquisition

The main responsibility of the Data Link Controller (DLC) module is to acquire and process all the physiological parameter values from the bedside monitors given by the Hewlett Packard network. These values are captured around the clock every two seconds and placed into what is called the "seconds" queue. The seconds data is averaged and placed into a "minutes" queue and in turn these minute values are averaged and placed in a "half-hour" queue. This acquired data is then processed by other modules to support the different functionalities and periodically archived to the hard disk for future use.

2.4.2 Data Analysis

Data analysis presents information in a form that is more understandable than the raw data. The PDMS Trend Analysis module processes the DLC data queues and displays the data graphically to facilitate trend analysis. It does so at different time scale resolutions of seconds, minutes and half-hours basis. As requested by the hospital staff, the module was configured to resemble and emulate the current manual chart, thus assuring familiarity and user acceptance.

The Fluid Balance module manages and logs all the intake and output fluids of a patient. The entries of the Fluid Balance module are generated periodically from manual measurements of urine bags, infusion pumps and laboratory analysis. These entry readings are checked against abnormal values and their totals are kept up-to-date. To supplement the manual data entry of the Fluid Balance module, a speech interface has been added. The speech interface uses the

speech recognition technology for data entry and a speech synthesis output for data confirmation.

2.4.3 Surveillance and Decision Support

The PDMS Expert System module, which is currently under development, can help doctors and nurses to cope with important conflicting events and diagnostic conditions. It does so by generating warnings in the case of a predicted critical patient situation and by providing trend analysis interpretations. The Expert System module is a multi-level expert system that acquires its data from the Data Link Controller. The raw data of the patient's vital signs is smoothed, using digital filters categorized as normal, alarming, or critical, in either high or low directions, and then submitted to a rule-based expert system for diagnosis. The diagnosis of the current and predicted patient conditions for the fourteen beds of the ICU are then shown on the user interface to aid the health care professionals in the decision-making process. Formal testing and objective evaluation are now in progress, before the final decision of using the module in the ICU.

2.4.4 Record Keeping and Automation

Because the PDMS is not yet linked to the Hospital Information System (HIS), some of the administrative information of the patients is duplicated in the Patient Registration module. This duplication hopefully will disappear once the two systems are linked together. The Patient Registration module accepts basic static information such as name, date of birth, sex, address, admission and

discharge dates and times as well as bed assignment, allowing for the monitoring and filing of the patient's medical recordings.

Automation is one of the most powerful means to reduce cost and enhance efficiency. Aiming to that goal, a Nursing Workload Manager module is currently being developed. Its purpose is to generate nursing care plans, automate the PRN (Progressive Research in Nursing) workload measurement scoring, schedule nursing activities and log the completion of medical acts and interventions. This module can also be used by the administration of the ICU to determine staffing allocations and monitoring productivity.

This completes the outline of the functionalities of the PDMS current software modules. The next chapter will describe the Fluid Balance system in greater detail.

Chapter 3 The Fluid Balance System and its Implementation

3.1 The Fluid Balance System

3.1.1 Global Description

In the critical environment of hospital ICUs, more emphasis is being placed on a higher degree of automation for the management of patient information. Such systems are also required to receive a large amount of data manually. The main objective of the Fluid Balance system is to enable the entry, calculation, and correction of the volumes of all the fluid intake and output of a given patient. The Fluid Balance is an attempt to integrate and automate in a spreadsheet format the manual data readings taken periodically by the nurses from various fluids, ranging from the intake of intravenous solutions to the output of blood and urine.

The Fluid Balance system operates in the graphical environment of OS/2's Presentation Manager driven by the keyboard, the mouse and the speech interface as inputs modalities. Its speech interface is meant to allow "hands-free" and "eyes-free" data entry of level values and to provide a more natural and familiar means of entering data by the use of the nurses' voices.

3.1.2 System Specification

The Fluid Balance system is an integral part of the PDMS. The structure of the Fluid Balance software consists of three major components shown in figure 3.1: 1) the speech recognition input; 2) the spreadsheet; and 3) the voice generation output.

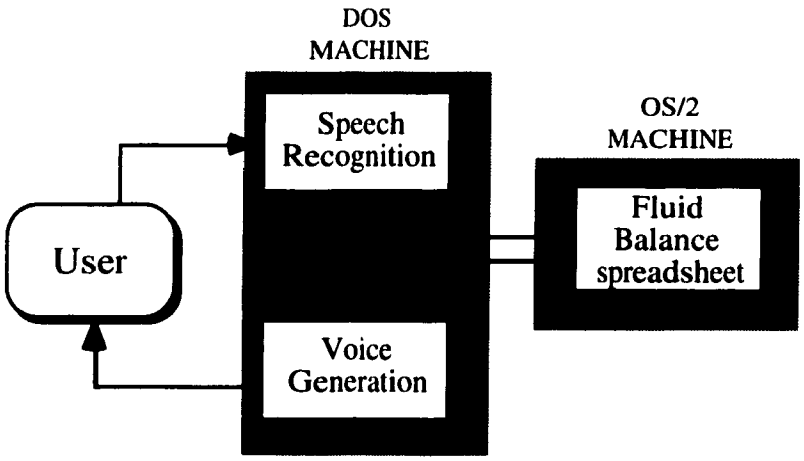

Figure 3.1: The Structure of the Fluid Balance System

The Fluid Balance system handles all the manual data entries made by the nurses. The speech recognition input translates natural language words spoken into a headset microphone, into data that the attached host computer can understand. The translated data is then sent to the spreadsheet application for processing. The speech recognition input software runs under the DOS operating system environment.

The Fluid Balance spreadsheet application consists of three separate windows; the main window, the ingesta window, where all the intake fluid data is registered and the excreta window, where all the output fluid data of a patient is recorded. The spreadsheet application automatically calculates all entries, alerts for any abnormal data and records the time of the entries. The Fluid Balance

spreadsheet emulates the manual spreadsheet format used in the hospital. The spreadsheet application runs under the IBM OS/2 operating system environment.

The voice generation output provides feedback and confirmation of the entries processed by the spreadsheet application. It translates coded numbers into files names. These files have pre-recorded digitized natural voices that are played back into the headset speaker of the user. The voice generation output software also runs under the DOS operating system environment.

Due to the fact that the speech interface (input/output) runs on a DOS machine while the spreadsheet runs on an OS/2 machine, a data link controller component must exist to ensure proper and reliable transfer of messages between both machines. The data link controller packs and unpacks messages between the two systems and alerts users in case of errors.

3.1.3 Sub-Systems Design

At the time of development, there was no speech recognition or speech generation hardware or software for the OS/2 platform. All available systems were compatible with the DOS operating system environment only. This constraint has influenced, to a large extent, the choice of the hardware and consequently imposed the architecture of the speech interface software for the Fluid Balance system.

3.1.3.1 The Speech Interface

Two hardware peripherals are used to handle the speech recognition and the voice generation. They are the Verbex Voice Conversational I/O System and the Covox Voice Master Key II, respectively.

3.1.3.1.1 The Speech Recognition Input

Setting up an application for the Verbex "Recognizer" is a five-step process [VERBEX,1990]. First, a text file containing grammar rules, specified in a special notation called Verbex Standard Notation, must be created. These grammar rules specify what words the Recognizer should listen for, what orders and what patterns of these words it should accept, and the translations which are the character strings that should be sent to the host computer upon word recognitions. Then, the grammar definition file is compiled into a "machine-readable" file that can processed by the Recognizer. Third, the compiled Recognizer file is transferred from the host computer to the Recognizer's memory. Fourth, the words and the phrases defined in the grammar rules must be trained by the user. Finally, after the training, the user's voice templates for the grammar definitions are stored in a voice file. Because the Verbex Recognizer is a speaker dependent device, every user must create a voice template of the grammar definition to be able to use the system.

The grammar definition file is organized in a tree fashion to resemble the menu of the Fluid Balance architecture. This allows the Recognizer to be context sensitive and recognize only the words that are relevant to the context of execution. There are several rules that govern the creation of the grammar

definition file to reduce its complexity and increase its efficiency. Naming a few examples, these rules are:

1) Using multiple grammar definitions

2) Limiting the number of the first words

3) Decreasing word instances

4) Avoiding escape words

5) Avoiding confusing words

More details about these rules and the evaluation of the grammar file will be covered in the next chapter.

Figure 3.2 shows a block diagram of the speech recognition input process.

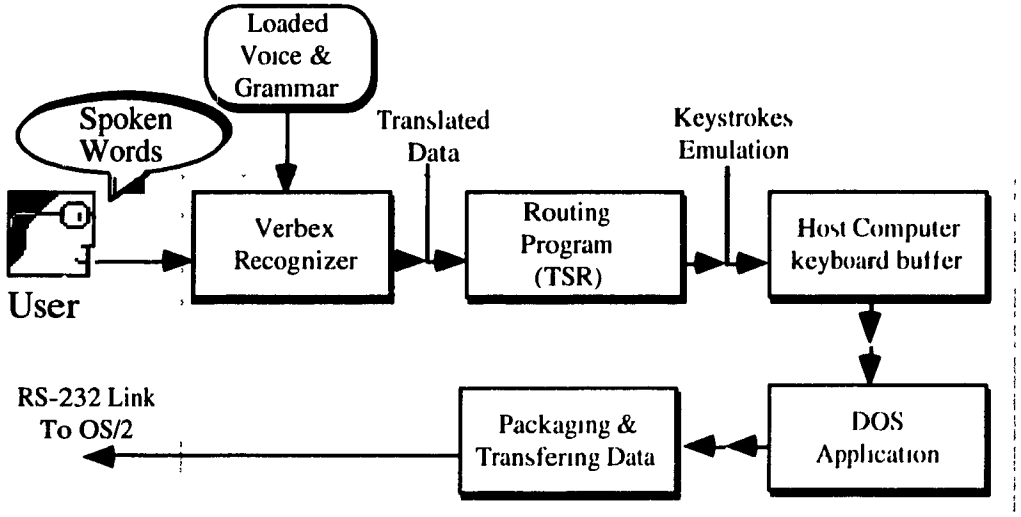

Figure 3.2: Speech Recognition Block Diagram

The execution process starts by loading the voice template of the user and the grammar file into the Recognizer's memory as initialization. Once that is done, the Recognizer becomes active and waits for the user's spoken words. Upon recognition of words, translated data is transferred to the host computer. A terminate-and-stay-resident routine program is provided with the Verbex utilities to redirect the translated data to the DOS computer's keyboard buffer. This redirection emulates a sequence of keystrokes on the keyboard which activates the DOS side application. The main processing loop on the DOS side performs the following:

1) captures the translated data sent by the Recognizer from the keyboard buffer,

2) packages the data and appends the appropriate header and trailer to the packet,

3) transmits the packet across the serial RS-232-C link to the computer running the OS/2 system,

4) waits for the voice response packet from the OS/2 system,

5) receives and decodes output data,

6) acknowledges proper receipt of a packet,

7) plays back the words or phrases as indicated by the Fluid Balance application.

3.1.3.1.2 The Voice Generation Output

The Covox Voice Master Key II is a speech and music input and output computer I/O peripheral which connects to a host computer's parallel port [COVOX, 1990]. Its input capabilities of performing speech recognition were abandoned because of its limited 64-word vocabulary and the non-continuous speech utterance support. However, its output features were exploited for our system. Thus, it was necessary to have two hardware devices in our implementation, the Verbex for input and the Covox for output. The audio output offers feedback for confirmation and for synchronization with the Fluid Balance spreadsheet application. It would be wrong to assume that whatever is recognized by the Verbex is always carried out inside the spreadsheet. Therefore, the voice generation output provides real time feedback of the command execution within the spreadsheet of the Fluid balance module. Figure 3.3 shows the block diagram of the output process.

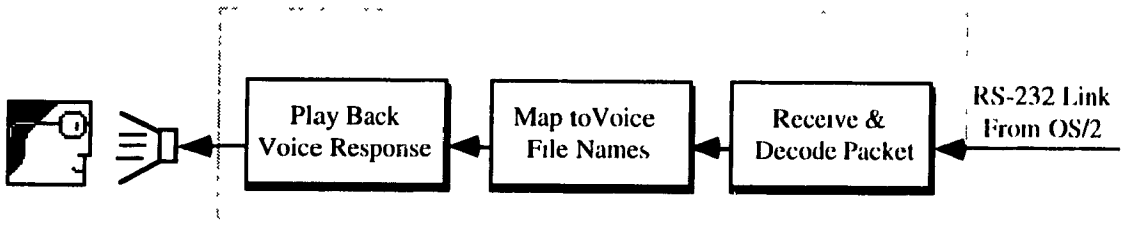

Figure 3.3: Voice Generation Output Block Diagram

The Master Key II software package has several utilities to help set up the system for voice generation. The first step is to record and edit the words and phrases so that a natural voice response file could be used. The voice files must be loaded into the host computer's memory as RAM disk files. Then, when the link controller receives a packet from the OS/2 side, it decodes the data and sends it to the DOS application. The DOS application maps the data to its

corresponding voice files and signals the Covox Master Key II to play back the speech to the user.

3.1.3.2 The Fluid Balance Spreadsheet

On the OS/2 side, the software consists of two separate and independent tasks; namely the Fluid Balance spreadsheet program and the OS/2 serial port handling process. Running in a multitasking environment, the two programs communicate using the Dynamic Data Exchange mechanism provided by the OS/2 operating system. The DDE mechanism is activated when the serial port process posts a message into the Fluid Balance message queue. Noting the event, the Fluid Balance's message handling procedure decodes the message and activates the user interface interaction objects which perform the desired functionalities depending on the context of execution. Then, the Fluid Balance software prepares voice responses and posts a message through the DDE to the serial port process which packages and passes it on to the DOS side. Figure 3.4 shows the OS/2 side block diagram of the Fluid Balance and the Port Process. More details about the Port Process are given in the section 3.1.3.3.

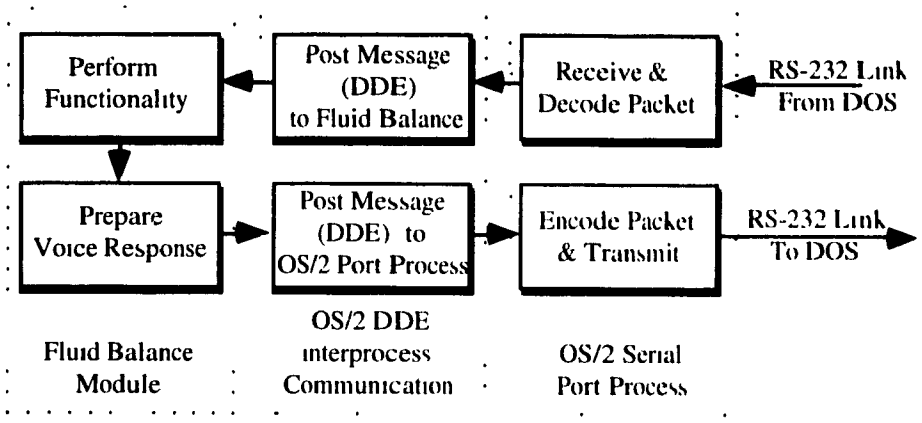

Figure 3.4: Fluid Balance & Port Process Block Diagram

The Fluid Balance spreadsheet software consists of three main windows which are implemented with the Presentation Manager of the OS/2 operating system. The main window shown in Figure 3.5 is responsible of creating windows for instances of the Ingesta and the Excreta sheets where all the fluid intake and the fluid output of a given patient are registered. This is done when the "Ingesta" and the "Excreta" menu commands are selected. The "Novice" option allows the user to adjust the software to accommodate a novice or expert user. This option allows an expanded voice feedback to be activated, leading the user through the tree of the voice commands. "Settings" selects the patient for the session entries giving the bed number, the patient name and the hospital ID.

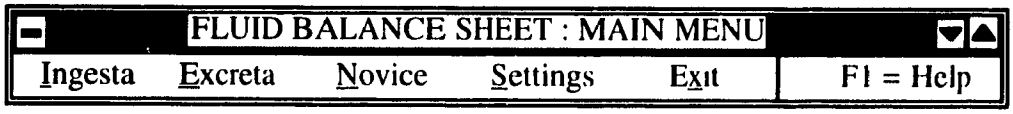

Figure 3.5: The Fluid Balance Main Window

The spreadsheet is made to emulate the "look and feel" of the manual charts of the hospital by using several interaction objects like pull-down menus, dialog boxes, entry filling fields, and push buttons depending on the context of execution.

Figure 3.6 shows the Ingesta window which offers the commands as menu pull-down objects. Up to five different intravenous solution levels may be recorded in IV1 to IV5. All entries are time-stamped automatically at the time of the entry or explicitly by using the "Time" command.

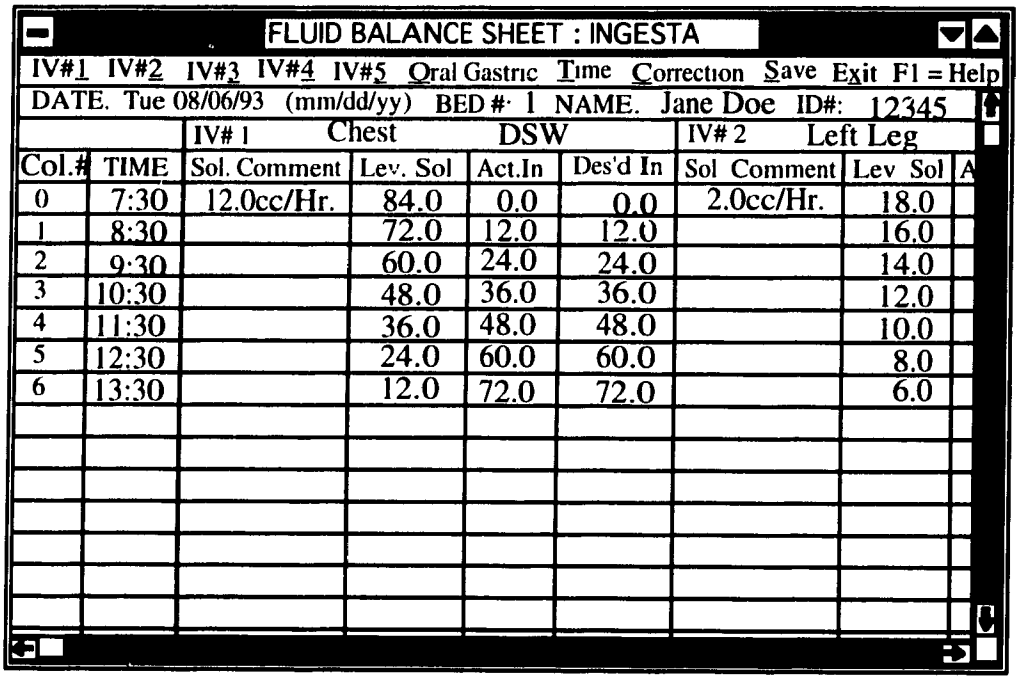

Figure 3.6: The Fluid Balance Ingesta Window

The "Oral Gastric" menu command allows the entry of the type and the level of the oral gastric solution. "Save" allows the user to save the data into a file for a later retrieval and use. The "Correction" menu option allows the user to re-edit an entry according to the line number and the solution name. The "Exit" menu destroys the window, exits the Ingesta instance and goes back to the main window. "Help" creates an interactive help screen.

Figure 3.7 shows the Excreta window. The "Blood", "Urine", "Gastric", "Stool", and "Other" allow the recording of fluid output according to their categories and indicate the time at which these readings were recorded. The "Time",

"Correction", "Save", and "Exit" options perform identical tasks to those in the Ingesta window.

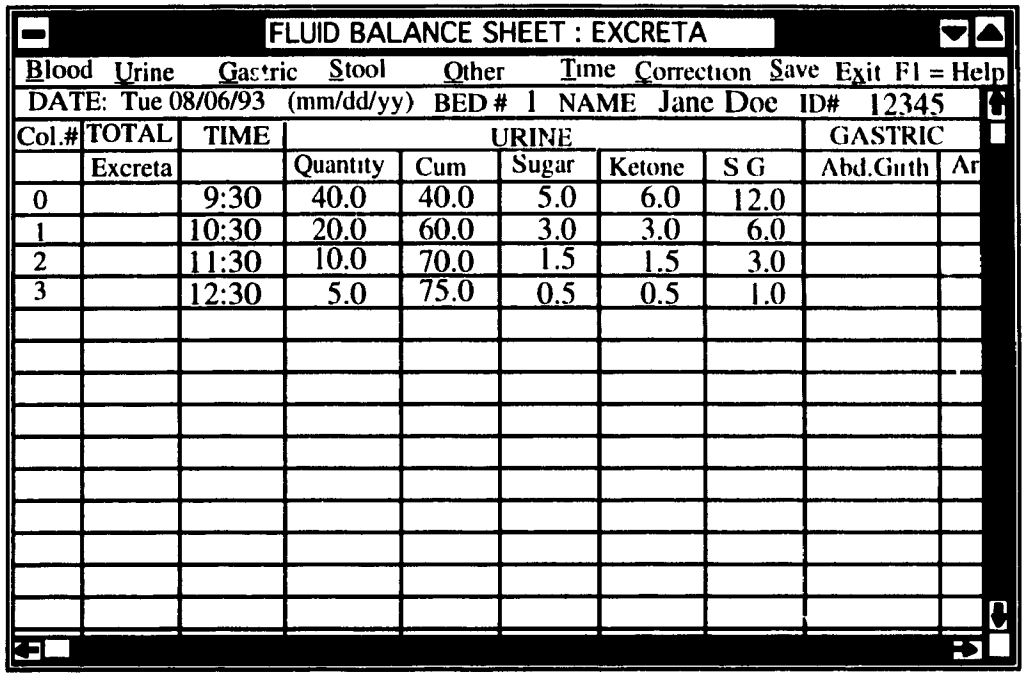

Figure 3.7: The Fluid Balance Excreta Window

When interacting with the Fluid Balance module through the keyboard or the mouse input modalities, the position of the user and the proper message action are posted to the software application by the Presentation Manager automatically. However, with the voice command as an input modality, the developer used state variables to keep track of where the user is in the menu tree. The state variables ensure the synchronization of the voice commands with the syntax of the allowable traversal of the menu structure in the Fluid Balance module. This synchronization is mirrored inside the grammar definition that is loaded into the Recognizer. The benefits of this synchronization is better performance by the speech recognition board, because of the minimum number of the legal first words. Also, the voice generation can be grouped into several sets of responses depending on the position of the user inside the menu tree.

Figures 3.8, 3.9, and 3.10 show the menu tree for the main window, the Ingesta window and the Excreta window, respectively.

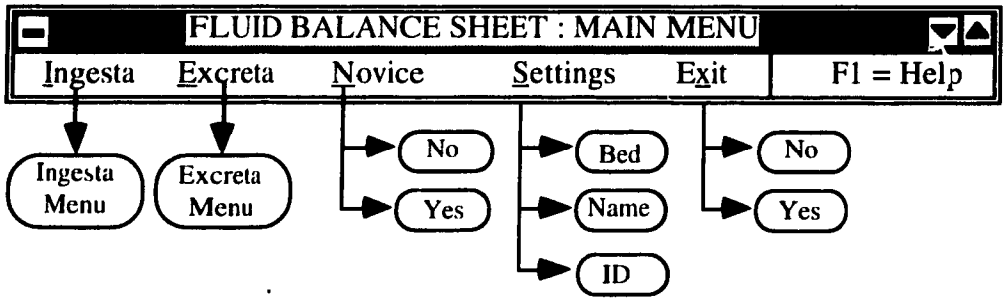

Figure 3.8: The Fluid Balance Main Menu Tree

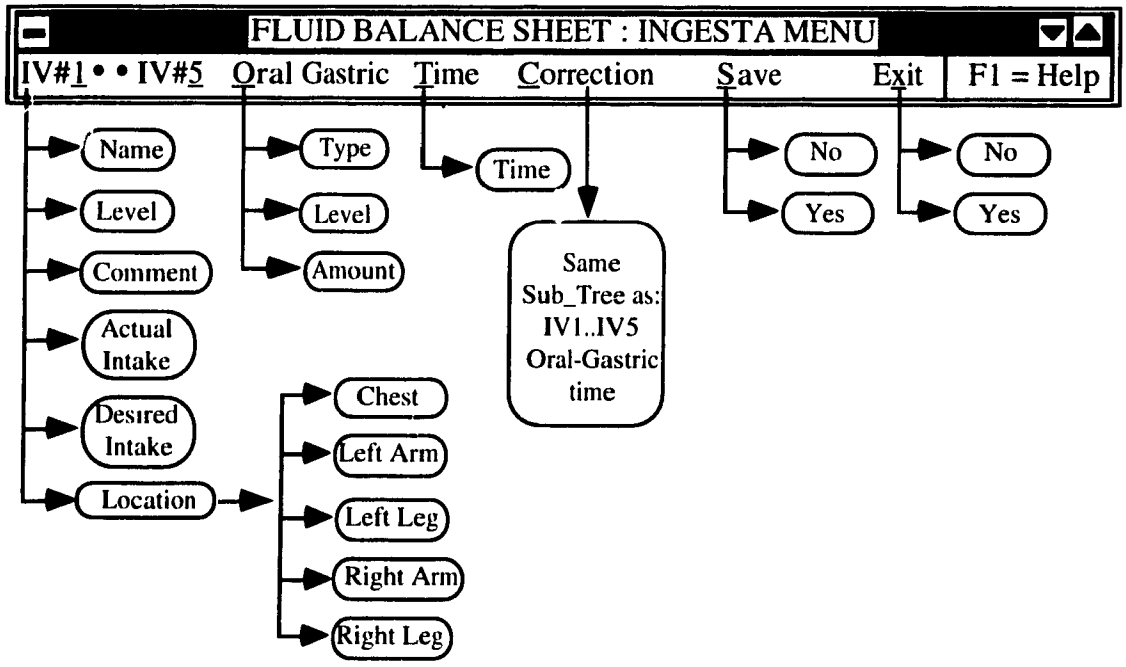

Figure 3.9: The Fluid Balance Ingesta Menu Tree

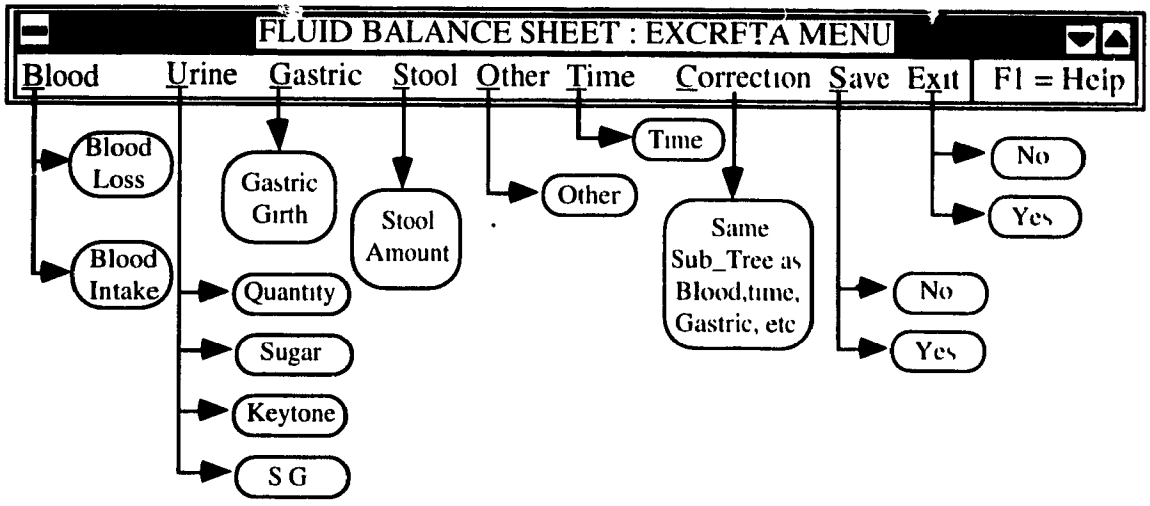

Figure 3.10: The Fluid Balance Excreta Menu Tree

3.1.3.3 The Link Protocol

The physical link between the DOS and OS/2 machines is realized through RS-232-C specifications and communication uses a modified Intel Hex format. The application level on the DOS machine is serviced through a user defined interrupt capable of handling a baud rate of 9600 without any loss of characters. On the OS/2 side, the serial port communication is handled through an independent task as mentioned earlier. The data packet format for communication between the two system is shown in figure 3.11 where the fields are as follows:

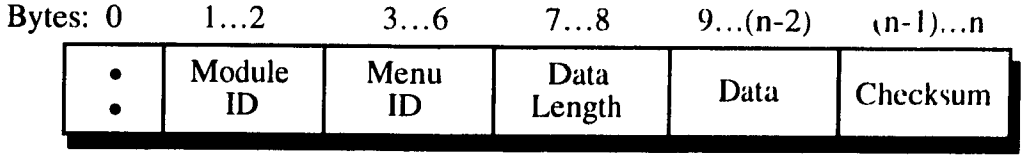

Figure 3.11: Data Packet Structure

- ':' denotes the start of a packet.

- "Module ID" is the destination module ID composed of two hex digits allowing 256 different modules to be serviced. Hex digits are coded using their ASCII representation (0-9, A-F).

- "Menu ID" is the node number of the corresponding menu tree inside the Fluid Balance module. It is used to track the voice command of the user and is composed of four hex digits.

- "Data Length" is the number in bytes of data to follow (two hex digits).

- "Data" is the voice command or the voice response between the two systems. Going toward OS/2, the data contains the valid ASCII entry of the menu shown underlined in figures 3.8, 3.9, and 3.10. Going toward DOS, the data contains the file descriptor ID of the voice file to be played back.

- "Checksum" consists of two hex digits calculated by negating the sum of all character values in the message after the colon character using modulo 256.

The packet transmission is governed by the use of the ACK-NACK protocol. The packet is re-transmitted if the sender receives a NACK representing a failure in matching the received checksum value with the internally calculated one at the receiver site. Otherwise, the sender will receive an ACK representing a good transmission and the process continues normally.

3.2 The Implementation

The Fluid Balance system is implemented in the C programing language using Microsoft C compiler version 6.00. The spreadsheet windows part runs under the Presentation Manager environment of the OS/2 operating system version 1.3, while the other part, the speech recognition and the voice generation, runs under the DOS operating system version 5.00. The two machines are linked using the RS-232-C protocol. The implementation exhibits modular design with high cohesion and minimum data coupling characteristics. Small modules, single function per module, and limited fan-out structure and data complexity are the basic principles used in implementing the Fluid Balance system.

3.2.1 The Grammar File and Playback Files

The grammar file is organized into small sets of grammar definitions to resemble the Fluid Balance spreadsheet tree menu. As a whole, it is made up of approximately 250 words. Each set activates a context dependent interface to either trigger a menu command, select menu item, respond to a dialog box, or enter digits from 0 to 9. Two escape words, "escape" and "main", are implemented to re-synchronize the user in case of a possible loss of synchronization situation. "Escape" is relative, taking the user to the next higher level position in the tree structure and "main" is absolute giving the user an anchor to the top of the tree.

As for the speaker, it requires about 25 Kilobytes of space for the compiled grammar file and about 20 Kilobytes of space for the trained voice file. The total of space required by both the trained voice and the compiled grammar is about

17% of the total amount of memory available inside the Recognizer hardware unit. The training session of the grammar file requires about 23 minutes. This session consists of uttering each of the words of the vocabulary and a few scripts that are the concatenation of the words into phrases as prompted by the compiler system. The Recognizer algorithm does not train all the combinations of the words because of the explosion in the number of scripts. Instead, it requires a minimum set from which the algorithm extrapolates all the remaining set of scripts.

The voice generation output files were recorded and digitized using the utilities of the Covox Master Key II with a sampling rate of 14 KHz and 8-bit representation per sample. The number of words used for the responses total about 250 words. The disk space required to store all the 250 words is 2.5 Megabytes. Every second of digitized speech requires about 14 Kilobytes of storage and the average word length is about 0.7 seconds.

These words were edited to delete any leading and trailing silence so that any concatenation of words into phrases will result in a natural speech output. The voice output is generated when the DOS system receives a packet from the Fluid Balance module and decodes the response number that maps directly to the voice file name.

3.2.2 The Fluid Balance Spreadsheet

The Fluid Balance spreadsheet module consists of a single thread program running under the OS/2 operating system. All the user interactions are done through the proper calls to the Presentation Manager's toolkit functions. The

Fluid Balance module is divided into 10 files of source coding totaling approximately 6400 lines. About 40% of this total is dedicated to a speech interface program which implements the state variables for tracking the user's position in the menu tree, the processing of voice commands and the delivery of response messages. The size of the executable file is approximately 170 Kilobytes. The inter-process communications between the spreadsheet process and the serial port process of the OS/2 side are done through the Dynamic Data Exchange feature using the client-server model. More details are presented in [Petroni, 1991].

The main window initially created when the program starts is the parent of two children windows: the Ingesta and Excreta windows. All the tree windows share the same data structure containing the administrative patient information such as the name, hospital ID and bed number. Also, all three windows share the variables that are required for message posting between the windows and the window handlers. Every window has its own source file program to take care of all user interactions as well as writing into the client area of the window. A dedicated program is used to handle the message queue and to decode the DDE messages. In brief, the 10 source files are divided among the logical functionalities of the Fluid Balance module.

3.2.3 The OS/2 Port Process

The OS/2 Port Process program contains about 950 lines of source code. The size of the executable file is about 32 Kilobytes. The OS/2 Port Process is made up of two threads: the child thread to handle the serial port reads and writes, and the parent thread to handle the messages posted in the OS/2 message queue.

Communication between the two threads is done via message posting and semaphores. The Port Process program is implemented under the Presentation Manager for two reasons; one is to provide the user with an easy means to terminate the process via the Task List, and the other is to enable the creation and use of the message queue which is crucial to the DDE communication with the Fluid Balance process.

Upon receiving an incoming data packet from the remote DOS system, the OS/2 child thread decodes the packet and dynamically allocates enough memory to store the incoming data based on the data length field contained in the packet header. Moreover, it posts a message in the parent thread's message queue and sets a semaphore. The parent thread proceeds to send a DDE data structure containing the menu number, the data length, as well as the data itself to the module specified by the module number field (i.e. the Fluid Balance process) in the packet header. When the parent thread receives a response from the Fluid Balance module, it places any outgoing data in the global storage accessible by the child thread and clears the semaphore to activate the child process to go ahead by transmitting data to the remote system.

3.2.4 The DOS Port Process

The DOS Port Process contains some 650 lines of source code and the size of the executable file is about 25 Kilobytes. Due to the single task characteristic of DOS, a user defined interrupt routine is implemented to service the incoming serial port data after setting the computer's programmable interrupt controller's (PIC). This interrupt services the data received at the port by placing it in a

circular queue of 16 Kilobytes in size. All insertions are placed at the tail pointer of the queue and all retrievals are taken from the head pointer of the queue.

After placing the data from the OS/2 remote system in the circular queue, the Port process proceeds to decode the sequences of data which correspond to group of four hex digits representing the words or a sequence of words to be played back. The RAM disk implementation (see section 3.1.3.1.2) uses a binary search tree which maps the voice file name digits into the RAM locations of the host computer. The voice files are loaded in RAM for faster play back to the user by avoiding time delays associated with the retrieval of the disk files. About 3.0 Megabytes of RAM are required for the voice files.

Chapter 4 The Evaluation Process and its Results

4.1 The Evaluation Overview

The goal of the evaluation process of the Fluid Balance system is to formally instill confidence in the quality, leading to the acceptance by the potential users upon delivery of the final product. This formal evaluation is an important and essential part of the Fluid Balance (FB) software life-cycle. It is designed to prove that the system meets its functional objectives from the user point of view and to locate any potential weaknesses, that would lead to further improvements.

The evaluation process concentrates on a set of attributes which are the main quality criteria of the FB software, namely:

- Correctness of the software functionalities;
- Efficiency and ease of use of the input/output data processing;
- Reliability of the registered data and of the overall system.

Clearly, before the FB is installed and used by the hospital staff in a real environment, all the required functionalities of the fluid intake and output and their respective calculations must be proven correct and trustworthy. Data input should be at least as efficient and easy as the previous manual input method, take no longer to execute, and be presented in a familiar form. The

output of the FB should be accessible rapidly on the video monitor as well as on a hard copy, when required. Reliable software must insure robustness and the ability to operate and survive within the ICU environment. It must also insure the integrity without any loss of data entered.

The FB evaluation was designed to be carried out incrementally in three stages. Each stage has its own goals and methods. Stage one involves "unit testing" in which the FB is treated as separate and independent modules or sub-systems, that are individually validated versus a well defined set of functionalities and characteristics. Stage two involves "system testing" in which the FB modules are combined to form the complete product. Different techniques are applied to validate the overall qualities and the robustness of the software. Finally, stage three, "acceptance evaluation", is carried out by the real users according to planned training sessions so that the users perceive the benefit and the positive value of the product to the everyday operations in the ICU environment.

4.2 Stage 1: Unit Testing and Validation

An effective way of implementing a complex system is to structure it into a collection of smaller simpler sub-modules. These simpler modules become easier to implement and test. The goal of such unit testing is to compare the function of the module against the functional requirements or interface specifications defining the module in order to uncover any discrepancies. A variety of specific techniques are utilized in the unit testing namely grammar file analysis, boundary-value analysis and design-based functional testing.

The four sub-systems of the FBS to be tested are:

- The Speech Recognition Input module (Verbex)
- The Voice Generation Output module (Covox)
- The Link Protocol modules (DOS and OS/2)
- The Spreadsheet module (PM windows)

We shall now describe the Unit testing performed on each of the above modules in the following four sub-sections.

4.2.1 Complexity Analysis of the Grammar File

The Speech Recognition Input module is the software executed by the Verbex hardware system (outlined in section 3.1.3.1.1.) that handles the dialog commands of the FBS. The speed at which the Recognizer processes voice commands during operation is directly related to the complexity of the grammar file used. The larger and more intricate the grammar file, the more time the Recognizer requires to process an utterance. For very complex grammar files, a potential overflow or overrun is highly probable. The complexity rating of a grammar file is the ratio of the rate at which data is being entered into the Recognizer, versus the rate at which the Recognizer can process and translate the data.

$$\text{Complexity} = \frac{\text{Speech data entry rate}}{\text{(Data processing / Translation) rate}} \qquad \text{Eq. (1)}$$

Several approaches can be taken to reduce the complexity of a grammar file. The following criteria are used to evaluate the grammar file quality:

- Structured subdivisions into multiple grammars;
- Reducing the number of the first words in sentences;
- Decreasing the maximum number of word instances in order to reduce the memory space of the required active grammar;
- Avoiding too many escape words.

4.2.2 Boundary-Value Analysis

Boundary-value analysis requires selecting test cases that represent all the situations directly on, above, and beneath the edges or boundaries of both the input equivalent classes and the output equivalent classes. For example, the data length field specified in the structure of the link protocol packet defines the valid length of the data to be from 1 to 255 characters long. The test cases of such input equivalent classes would analyze packets of 0, 1, 255, and 256 character in length. The result space should also be considered and test cases must be written to verify all the boundary-values of the output.

4.2.3 Design-Based Functional Testing

A distinction can be made between requirement functions and design functions. Requirement functions describe the overall functional capabilities expected of a program, and cannot usually be implemented without inventing other "smaller functions" to design the program. If one thinks of this relationship as a tree structure, then a requirement function would be represented as a root node, and

the "smaller functions", all those functional capabilities corresponding to design functions, would be represented by boxes at lower levels of the tree. This successive refinement taking place during the design process can generate multiple levels in the tree structure, where the (n+1)st-level nodes are refinements or sub-functions of the (n)th-level functions.

To apply the design-based functional testing, the functional design trees of the Fluid Balance system must be constructed. Furthermore, all the functions included in the design trees should have an important common criteria namely that the function be accessible for independent testing. Given these multiple functions to analyze, test data generation can proceed as described in the previous section on boundary-value analysis.

4.2.4 Incremental Approach

The Spreadsheet module and the Link Protocol module are best suited to be divided into smaller blocks from which the functional design tree can be built and the boundary-values analysis can be explored. The manner in which the unit testing technique is carried out has an important bearing on the form of the test cases. For example, in incremental testing, rather than testing each one of the smaller blocks in isolation, the next small block to be tested is first combined with the set of the smaller blocks that have already been tested.

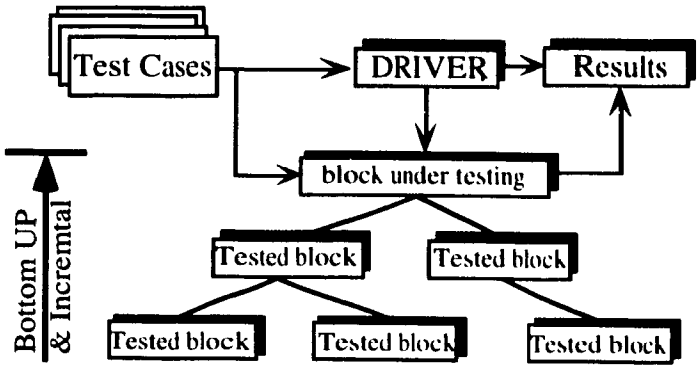

Figure 4.1: Bottom Up and Incremental Testing Approach

The bottom up approach starts with the smaller blocks at the lowest level in the module, i.e. blocks that do not call other smaller blocks. The only requirement imposed before testing the next block is that all of the block's subordinate blocks must have been tested previously. Using this approach, all programing errors related to mismatching interfaces or incorrect assumptions among modules will be detected earlier. Incremental testing can result in a more thorough testing than a non-incremental one. Furthermore, the most important test conditions (applied by driver modules) are easy to create and the reading of the test results is simple.

4.3 Stage 2: System Testing and Validation

System testing and validation is not only the process of testing the functions of the complete system but, most importantly, involves comparing the system to its original objectives. Therefore, the goal is to demonstrate that the system does properly meet its intended specifications. In system testing, all of the bugs

should be logged in error forms so that the cost of the system validation can be calculated. In our case, the FB system will be subjected to two techniques :

- State transition analysis;
- Stress testing and synchronization analysis.

4.3.1 State Transition Analysis

State transition is a technique that aids in selecting, in a systematic way, a high yield set of behavioral test cases. The goal is to point out any incompleteness and ambiguities in the grammar file as well as in the menu commands of the FB Spreadsheet. The state transition is based on three sets of sequences: input sequences, corresponding transition or next-state names and outcome sequences. An error can manifest itself as one or more of the following symptoms:

- wrong number of states;
- wrong transition for a given state-input combination;
- wrong outcome for a given state;
- states or sets of states become dead (hang up);
- states or sets of states become unreachable.

4.3.2 Stress Testing and Synchronization Analysis

Stress testing of the system involves subjecting the FBS to heavy loads or stresses. A heavy stress is a peak volume of data encountered over a short span of time. The objective is to see if the speech recognition input modality will loose synchronization with the Spreadsheet module. Further stress testing would explore the system's reaction to such an unexpected loss and would examine the recovery procedures.

4.4 Stage 3: Acceptance Evaluation

Acceptance evaluation consists of the verification and validation of the system's "fit" into the real environment. Acceptance evaluation tests the interfaces between the software system operations and the users and their procedures. To perform the acceptance evaluation, two important steps were designed. The first evaluation is carried out within the McGill lab by a few users and the second is done in the real environment by the real users. The first evaluation is performed in order to get an assessment or feedback of the human factors involved as well as the user interface interactions. The second evaluation will use the feedback of the first performed step to overcome any resistance toward the system and to get it certified by the sponsors of the project.

4.4.1 Lab Performance Evaluation

Conducting performance testing presumes a robust, working, debugged, and stable system. In the presence of bugs that crash the system, the system's performance is nil and no rational performance testing can be done. The prerequisites for such evaluation are first, a clear statement or comparison objective (speech input should be at least as fast and easy as the keyboard and mouse modalities), second, a controlled experimental process or tested, and third, an analytical way to process and interpret the data.

4.4.2 Real Environment Evaluation

For this part, the system should be installed in the hospital and the real users should evaluate the FB after a few training sessions that will ease the normal human resistances of the nurses and doctors. Without such training sessions, the evaluation cannot be objective. Two different evaluation forms are suggested to rate the software. In the first form, the goal is set to identify the difficulties that the user is experiencing. With proper training sessions such problems should be minimized. The second form should have several sections covering all modules and functionalities, to get an objective evaluation from the users.

4.5 Evaluation Results

4.5.1 Unit Testing Results

4.5.1.1 Complexity Results of the Grammar File

Figure 4.2 shows the results of the Fluid Balance grammar file analysis according to the previewed set of criteria.

Analysis Criteria	Fluid Balance Grammar File	Recommendations
Multiple Grammars	36	Use Multiple
Number of First Words	[1, 15]	Minimize
Active Grammar Memory	23%	Maximum 100%
Escape Words	2	Minimize

Figure 4.2: Results of the Grammar File Analysis

The grammar file was implemented using 36 grammars, one for each of the context switching states that directly correspond to the command menus of the Spreadsheet module. Therefore, as speech data is entered in a continued sequence, only one grammar context state is active at one time. This approach improves not only the accuracy but also the response time, since the number of statements being listened for at each step is limited by the particular grammar active at that time.

The number of first words range from 1 to 15 depending on the context. The Recognizer's toughest task is deciding whether or not the sound that is being picked up is the beginning of one of the 15 valid utterances. For all of the

possible first words, the Recognizer continually compares segments of the sound it hears with sound templates, or stored sound patterns. A maximum of fifteen comparisons is required to decide if further incoming sound should be analyzed.

Figure 4.3 shows a representation of the Recognizer's grammar memory.

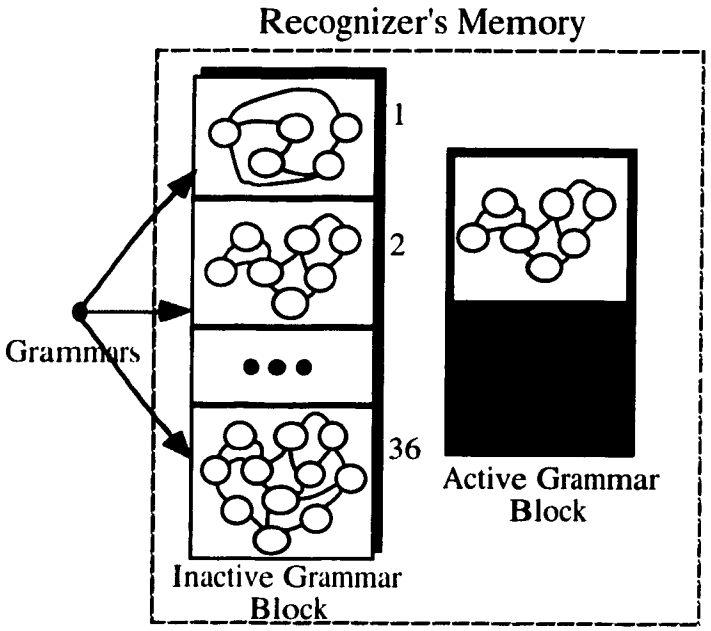

Figure 4.3: Representation of the Grammar Memory

The compiled grammar file is mapped into structures represented by arcs and nodes and assigned values for the arcs. The larger the number of the word instances, the larger will be the size of the required memory. As shown in figure 4.3, block 2 of the grammars occupies only 50% of the memory available in the active grammar block. In our implementation of the Fluid Balance system, the maximum percentage of word instance requirement calculated by the VERBEX CONVERT utility of the Recognizer is only 23%, which easily fits into the Recognizer active memory block.

The use of escape words greatly increases the number of arcs used in the grammar, thereby increasing the grammar's intricacy. Only two escape words are used to navigate properly inside the grammar file as shown in figure 4.2.

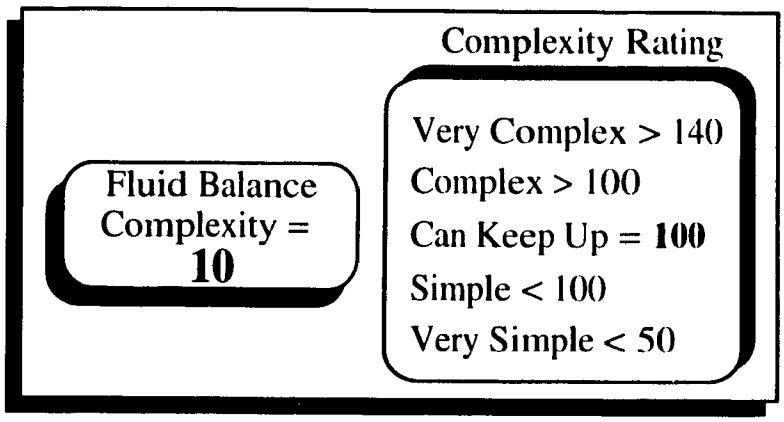

Figure 4.4: Grammar File Complexity Value

The above figure 4.4 shows the complexity rating of the grammar file given by Eq.(1) and computed automatically by the Recognizer's utility. The resulting complexity of 10 validates the requirement of the grammar file for ensuring an excellent response time. However, in addition to the response time requirement, we need a robust system capable of giving an acceptably low error rate. This can be influenced by the environmental variations in the presence of RF and ambient noises as well as by the proper selection of non confusing words. Although the words were carefully chosen for robustness, the final validation will have to wait until the acceptance evaluation is done in the real ICU environment. At this "Unit testing" stage, the Speech Recognition Input was successfully exercised to verify and confirm the correct operation of all of the designed grammars.

The statistics of the grammar file shows that there are about 250 words. The compiled Recognizer's grammar file requires about 25 Kilobytes of disk space,

and a typical user file of trained voice templates requires about 20 Kilobytes per speaker. This total requirement of 45 Kilobytes, fits quite comfortably into the 512 Kilobytes of RAM available on the VERBEX unit.

4.5.1.2 Voice Output Verification

The voice output verification used mainly a path traversal over the voice file branches. The traversal was to verify and validate the completeness and the correctness of all the entries in the voice file. Figure 4.5 shows a typical branch of the traversal list at the main entry state of the Fluid Balance.

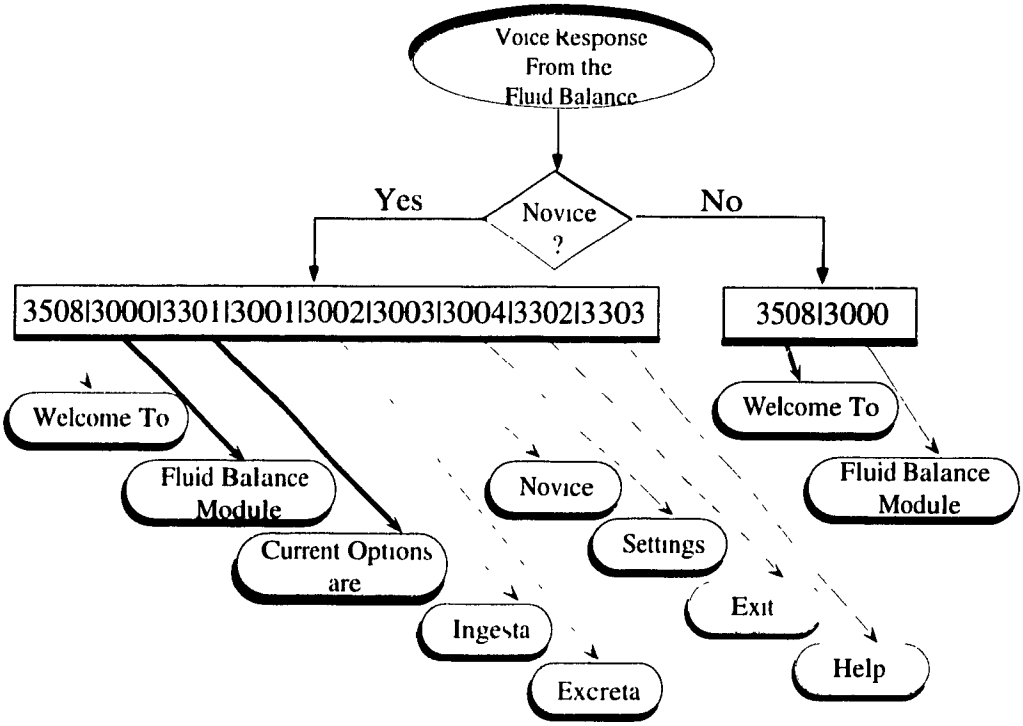

Figure 4.5: A Typical Voice Response Branch

The validation process considered not only the novice setting in which a detailed voice output is echoed back to the user as shown in figure 4.5, but also

the expert setting in which a minimum set of voice files are echoed back. The concatenation of files that make up the phrases was fast enough to resemble a natural voice (the voice files are loaded into the high RAM) and all the extra silences were trimmed off. The voices used were of two female students. Table 4.1 shows the breakdown of the voice files into different categories.

Groups	Code Ranges	# of Files
Digits	9100 - 9109	10
Letters	9200 - 9225	26
Ingesta	1000 - 1999	26
Excreta	2000 - 2999	17
Common	3000 - 3999	37

Table 4.1: Voice Files Categories

As can be seen from the above table, significant future expansion of the voice output vocabulary words can be easily implemented by extending the coding system used.

4.5.1.3 Link Protocol Unit Testing Results

The unit tests were mainly based on the functional design trees that were built for each module. Each test case was coded into a driver routine and was carried out for every block or node of the functional design trees. Figures 4.6 and 4.7 show the functional tree of each module of the link protocol, in which the numbers show the sequence in which the subroutines were tested.

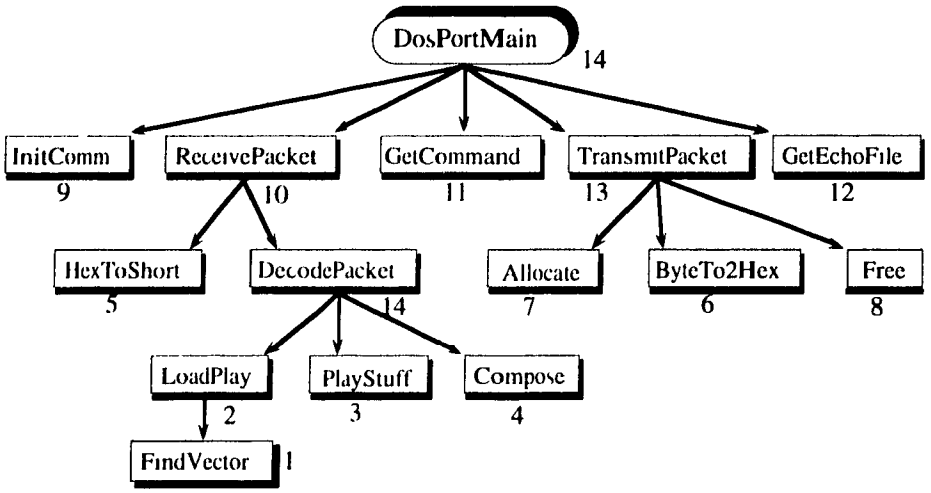

Figure 4.6: Test Sequence of the DOS Subroutines

The DOS Port module of the FB consists of 14 subroutines, whereas the OS/2 Port module consists of 11 subroutines. The sequence of testing shown in the mentioned figures follows the incremental bottom-up strategy, as explained earlier.

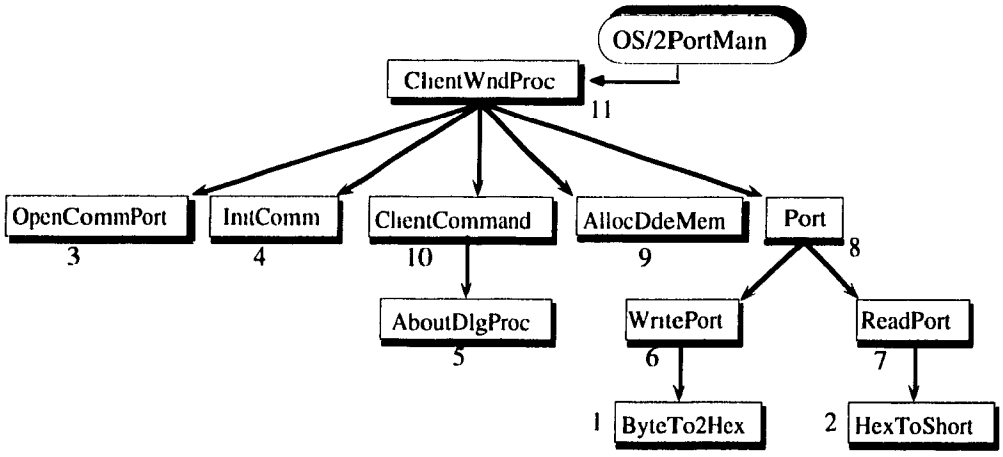

Figure 4.7: Test Sequence of the OS/2 Port Subroutines

Table 4.2 shows some samples of test cases from both DOS Port and OS/2 Port modules after the boundary value analysis of each subroutine. The analysis determined the valid set of test cases which were included in the drivers for every subroutine. Drivers were modified at each level as a new subroutine was

added to the previous set of subroutines. By this process, all errors related to interface mismatching that were found, have been corrected and removed. Table 4.2 also shows the behavior of the subroutine for each test case.

Subroutine Name	Input Equivalent Classes	Output Equivalent Classes	Test Cases	Subroutine Behavior
Decode Packet	Packet is Multiple of 4 bytes	•••	0, 5 bytes	Error Detected Retry transmission
			4 bytes	OK
	•••	Play Utility	Loaded	OK
			Unloaded	Message to load
InitComm	Data Definition	•••	5, 7, 8	7-Recognized 5,8 Error Flagged
	Data Parity	•••	Even/Odd/None	Even OK, Odd & None Rejected
	•••	Speed Control	DTR/Overrun	NO loss of data
	•••	Physical Cable	Connect/noCon.	OK/User Alerted
HexToShort	Hex characters	•••	0, F, G	0 F Ok Else Error

Table 4.2: Subroutines Sample of Test Cases

4.5.1.4 The Spreadsheet Unit Testing Results

Figures 4.8, 4.9. and 4.10 show the sequences of the unit testing and the dependencies of the subroutines in the Main, Ingesta, and Excreta modules, respectively.

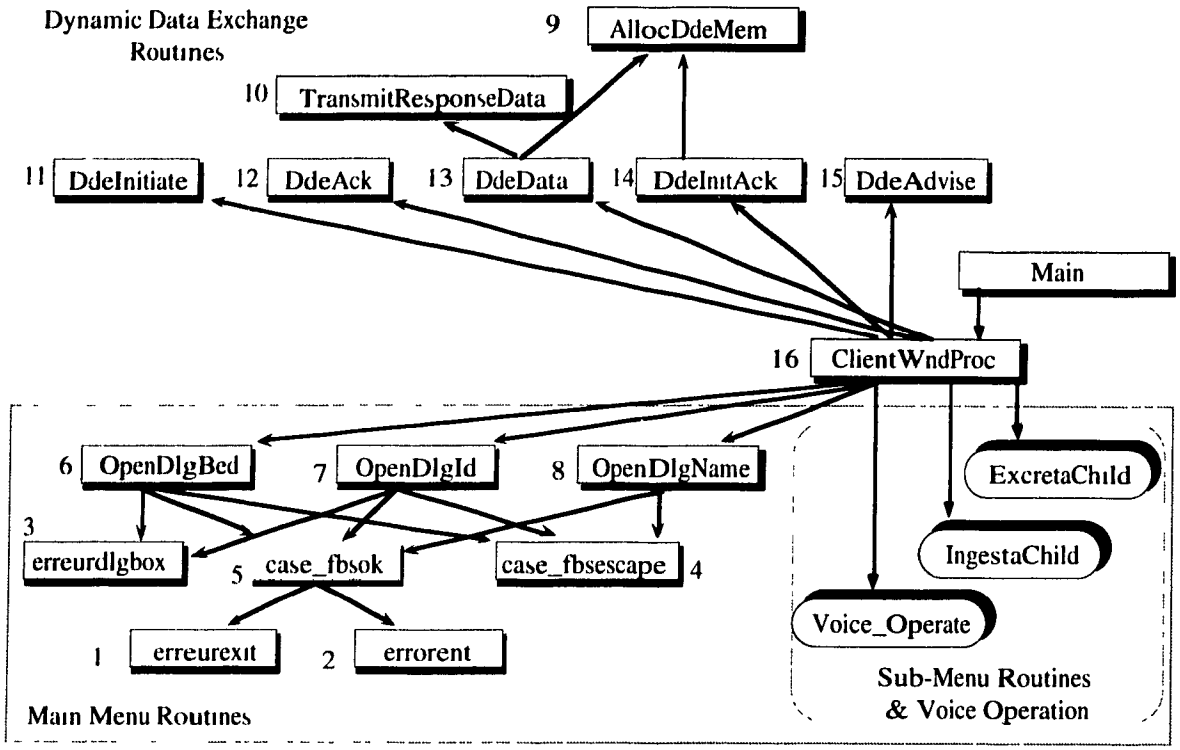

Figure 4.8: Test Sequences of the Subroutines for Main

A total of 75 routine drivers were written to test the Spreadsheet design functionalities. A few subroutines had crashed the system when subjected to non-valid inputs and as a result, stronger and more restrictive error handling measures were implemented to ensure proper behavior of the software.

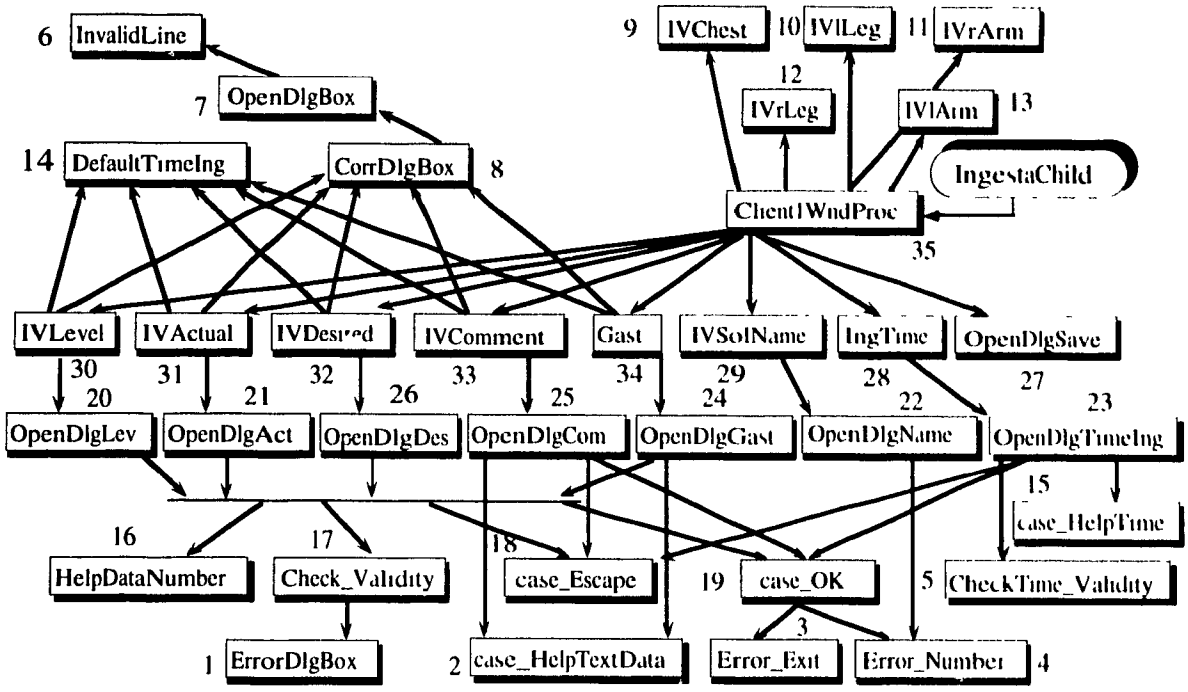

Figure 4.9: Test Sequences of the Subroutines for Ingesta

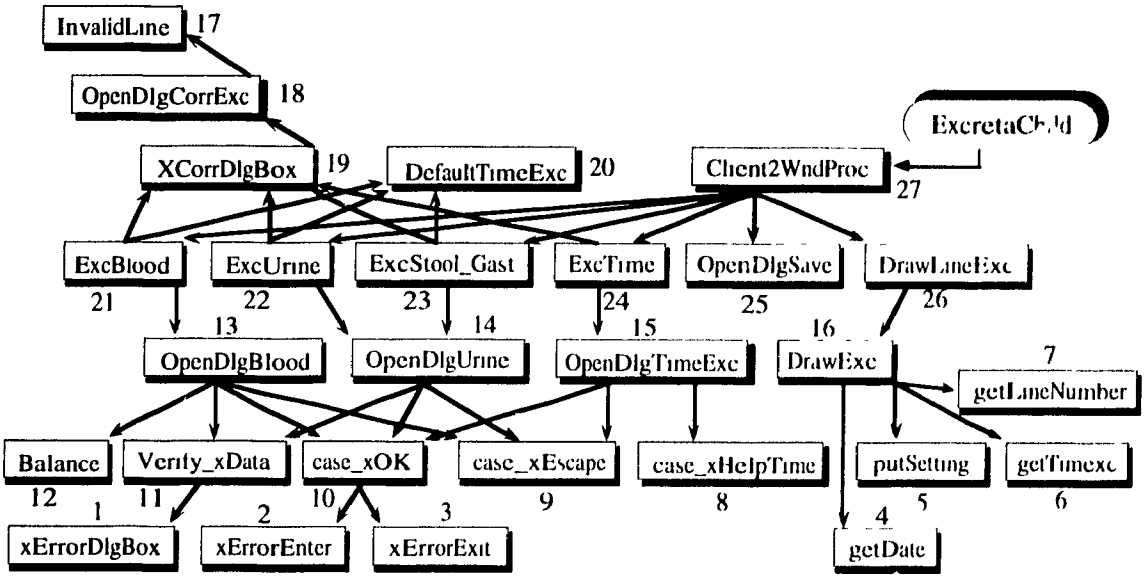

Figure 4.10: Test Sequences of the Subroutines for Excreta

4.5.2 System Testing Results

4.5.2.1 State Transition Validation Results

A finite state transition table has been built to model the Fluid Balance system behavior. This table is mainly used to generate test cases such as the one shown in figure 4.11. Figure 4.11 illustrates a test case starting at the "Main" state, then sequencing through Ingesta, Oral Gastric, Type and Plasma_OK inputs and finally ending at the IngestaMenu state.

Current Grammar State	Voice Input	Spreadsheet Action	Voice Expert	Output Novice	New Grammar State
Main	Start State	Main window created	3508/3000	3508/3000/3301 3001/3002/3003 3004/3302/3303	Main
	Ingesta	Ingesta window created or activated	3001/3300	3001/3300/3301 1000/1001/3100 3200/3302	**IngestaMenu**
	Excreta	Excreta window created or activated	3002/3300	3002/3300/3301 2000/2001/2002 2003/3200/3302	ExcretaMenu
	Novice	Novice menu pulled down	3003/3505	3003/3505	NoviceMenu
	•••				
IngestaMenu	IV(x) (x= 1-5)	IV(x) menu pulled down	1104/910(x)	1104/910(x)/3301 3400/1103/1105 1100/1102/1101	**IVMenu(x)**
	Oral gastric	OG menu pulled down	1001	1001/3301/1110 3400/1111	**OGMenu**
	•••				
OGMenu	**Type**	Type dialog box created	3101/1110	3101/1110	**TypeMenu**
	•••				
TypeMenu	**Plasma_OK**	Plasma is entered	N/A	N/A	**IngestaMenu**
	•••				

Figure 4.11: Partial State Transition Table

At the "Main" state, an utterance of the word 'Ingesta' forces the Spreadsheet module's action to either activate (bring to the foreground) or create (if non existent) the "Ingesta Window", then causes the Grammar file to make a transition to a new state the "IngestaMenu", and then depending on the voice setting, drives the voice output to echo either the detailed or abbreviated command options. Similar steps are repeated for the "Ingesta Menu" and "OGMenu" states. Finally, at the "TypeMenu" state, an utterance of the word 'plasma_OK' causes a transition in the Grammar file to the "IngestaMenu" state, the literal word "plasma" to be entered into the Ingesta Window as required and the voice output to echo the options of the Ingesta window commands.

After eliminating all the invalid combinations of states and input conditions, eighty seven (87) test cases were generated and tested. A few errors were found, most of which were related to a wrong state transition, a missing state action, or got the system to a hung state, in which there is no way out. These errors, as well as the errors described in the next section, have been corrected and logged into a "Sample Error Tracking Form" shown in Figure 4.12. The forms are used to measure the cost of the system validation in terms of human effort, computer time, and resource utilization. Also, the forms serve to keep a history of the system's bugs and their respective corrections.

As an example, the form in figure 4.12 is used for the "OS/2 Crash on Escape Key" bug which crashed the Spreadsheet module every time the reported "sequences to re-generate" was executed. It took about 2 hours to diagnose the problem and with the help of the OS/2 system manual it took about six hours and half to locate and correct the bug.

Sample Error Tracking Form

Roland El-Khoury	*April 3, 1992*	*Spreadsheet Module*	*1.1*	*34*
Completed by	Date	Program ID	Version	#

Means of Initial Problem Detection

- ☐ a Incorrect Output Found by User
- ☒ b. Incorrect Output Found by Prog
- ☐ c. Hand Processing/Review
- ☐ d. Changed Specification
- ☐ e Other, specify _____

Effort in Diagnosting the Problem

- ☐ a Working Time *2* Hrs
- ☐ b Outside Help Required ☒ystem Manual ☐tocol Anlyser I☐ugger

Summary Problem Description: **Bug Title:** ***OS/2 Crash on Escape Key***

After requesting the ingesta menu using the voice input and returning to the main window, if the user tries to set the bed number to #1 and upon negating the confirmation by using the escape key the Spreadsheet will crash

Sequences to Re-Generate Bug:

Go to ingesta window, return to the main window, set the bed number to 1 and then press the Escape Key

Roland El-Khoury	*April 6, 1992*
Completed by	Date

Change Required to Software - Reasons

- ☐ a. New Application Requirement
- ☐ b Specification Incorrect
- ☐ c. Specification Misinterpreted
- ☐ d Specification Incomplete
- ☐ e. System Software Change
- ☒ f. Program Bug Data and I/O
- ☐ g. Program Bug Missing Function
- ☐ h Program Bug Incorrect Function
- ☐ i. Other, specify _____

Change in Program Not Required - Reasons

- ☐ a. Error Was in the Test
- ☐ b. Error Is Not Repeatable
- ☐ c There Was No Error
- ☐ d Other, specify _____

Size and Difficulty of Correction

- ☐ Computer Runs *30 secs.*
- ☐ Working Time *6.5* Hrs
- ☐ Total Elapsed Computer Time *135* Mins.
- ☐ Total Elapsed Time *56* Days
- ☐ Module Name: ***FBSDLG.C*** Source Lines: ☒Added ☐eleted

Summary of Correction Made

The mapping between the bed number entered and the index of the buffer was properly aligned. The substraction was resulting in indexing -1 instead of 0. A source line of code was added to properly handle the indexing of the buffer.

Figure 4.12: Sample Error Tracking Form

The Form shows that the cumulative human effort required to correct all the bug reports to date (including #34) totals 56 days and the computer time of compilations is about 135 minutes.

Overall, the state transition testing was found to be a good technique to validate the functionalities of the system and to discover any incompleteness and incorrect behavior of the menu commands.

4.5.2.2 Stress & Synchronization Testing Results

To insure good synchronization and data reliability such that what is being said into the microphone is what has been registered into the Spreadsheet module, a batch file was used to subject the FB system to the stress testing technique. The batch file is used so that the stress testing will not be limited by the Recognizer's voice processing speed. Instead, the batch file runs on the DOS machine, causing direct translations of the data and commands to the OS/2 Spreadsheet. Then, both the outcome of the Spreadsheet values and the feedback received by the voice output generation were monitored. The characteristics of the test data in the batch file were created to force many switchings of window control between the Main window, Ingesta and Excreta windows, to extensively use the two escape words and their re-synchronization algorithms, to enter and correct data and finally, to monitor the behavior of the FB in the case of incorrect input entries. Figure 4.13 shows a sample sequence of windows switchings, data entry and correction, and describes synchronization tests.

Semantic Meaning	Batch File Data	SpreadSheet Action	Voice Output in Novice Setting
@ Startup	@ Startup	Main Window	Fluid Balance Module current options are Ingesta, Excreta, Novice, Settings, Exit, Help
Ingesta Window	1	Ingesta Win Actived	Ingesta menu current options are IV one to five, Oral Gastric, Correction, Time, Exit
Main Window	999	Main Win Activated	Fluid Balance Module current options are Ingesta, Excreta, Novice, Settings, Exit, Help
Ingesta Window	1	Ingesta Win Actived	Ingesta menu current options are IV one to five, Oral Gastric, Correction, Time, Exit
••• **Repeated N Times (Window Control Switching)**			
IV#1	110	IV#1 menu pull down	IV number one current options are Level, Member, Solution, Actual Intake, Desired Intake
Level	111	Level menu pull down	Enter Level
level is 23	23	Actual data Entry	You entered two three Please say OK or Correction
Confirmation	OK	Data logged Ingesta Win	Ingesta menu current options are IV one to five, Oral Gastric, Correction, Time, Exit
Correction	del	Data Errased Reprompt	Enter Level
••• **Repeated N Times (Confirmed & Corrected Data Entries)**			
IV#1	110	IV#1 menu pull down	IV number one current options are Level, Member, Solution, Actual Intake, Desired Intake
Level	111	Level menu pull down	Enter Level
Escape word	esc	Back to Ingesta Menu	Ingesta menu current options are IV one to five, Oral Gastric, Correction, Time, Exit
••• **Repeated N Times (Escape Words & Re-Synchronization)**			

Figure 4.13: Sample of Test Cases Included in the Batch File

The synchronization analysis was done by comparing the expected results with the Spreadsheet logged entries and the captured voice file numbers sent out by the OS/2 module.

The re-synchronization escape algorithms were very effective to re-establish a common state whenever the host and the voice input module lost contact with each other. There are two escape words that can be used at any time; one repositions the data entry focus to one menu higher relative to the current menu state, and the other is an absolute anchor to the main menu in the Main window. The data entered is discarded if any of the escape words is used at any time

before committing the data by confirming the entry. The confirmation technique of saying "OK" or "Correction" after the entry of data was used instead of implementing an undo function while the voice modality is in use.

Two problems were found that need to be re-worked. One is to eliminate all the dialog boxes that are activated upon entry of incorrect data while the voice modality is being used. These dialog boxes are suitable when the keyboard and the mouse are used because the user can see the screen and dismiss such dialog boxes by "clicking" on OK. However, the dialog boxes have no corresponding states in the voice input tree, consequently a loss of synchronization occurs. The second problem is that the help facilities are not designed for audio operations when the user is away from the screen.

4.5.3 Lab Performance Evaluation Results

The lab performance evaluation was done using six users with different levels of computer experience. The users were asked to perform some typical fluid balance data entry tasks after a tutorial review for those who were not familiar with the fluid balance spreadsheet. The users were divided into three groups; those who are computer and Fluid Balance experts, those who are computer experts but have no experience with the Presentation Manager software system environment, and those who have no computer or Fluid Balance experience.

The assessment of the evaluation was based on the following information:

- **Average task completion time:** indicates the average time required to complete the entry of one fluid balance value into the Fluid Balance Spreadsheet.

- **Number of errors in performing task:** indicates the total number of errors the users committed when performing the sample entry of data in the Fluid Balance Spreadsheet.

- **Number of tasks performed in 10 minutes:** indicates the total number of correct fluid levels entered into the Fluid Balance Spreadsheet during a 10 minute interval.

- **Training time:** is the time required by the user to train the Fluid Balance module vocabulary.

- **Recognition rate:** is the rate at which the system responds correctly to user utterances (1 - (false rejection rate + substitution rate)). The false rejection rate is the rate at which valid utterances are ignored or rejected by the system. The substitution rate is the rate at which a word uttered by the user is incorrectly recognized to be another word in the vocabulary.

The above statistics are useful in evaluating the speech input modality to see if it is as effective and easy as the traditional input modalities of the keyboard and the mouse.

The tables below show each group and the corresponding results. The following observations can be made:

For the Fluid Balance and computer experts, there was no significant advantage of using voice over mouse or keyboard input as shown in the results of table 4.3. This can be attributed to the fact that experienced users of a given application can easily manoeuvre the mouse and the keyboard to achieve the task. During the evaluation process, both users found the voice feedback

annoying, since they already knew from memory what options were available to them at a given level in the menu tree. Overall, both users felt that a speech interface would be quite useful in the ICU, in the case where users are not expected to be seasoned computer users. Also, they felt that the mobility for "hands-free" and "eyes-free" operation is desired.

User A	Speech	Keyboard & Mouse
Average Task Completion Time	12 secs.	10 secs.
Number of Errors in Performing Task	0	0
Number of Tasks Completed in 10 Min.	33	35

Training Time	Recognition Rate
20 min. 15secs	97.8%

User B	Speech	Keyboard & Mouse
Average Task Completion Time	20 secs.	15 secs.
Number of Errors in Performing Task	0	0
Number of Tasks Completed in 10 Min.	23	29

Training Time	Recognition Rate
22 min. 38secs	95.3%

Table 4.3: Test Set 1: The Computer and Fluid Balance Experts

In the case of the computer experts without knowledge of the Presentation Manager user interface, the speech input lead to shorter task completion times, compared to performing the same tasks using the keyboard and the mouse, as shown in table 4.4. The users felt that the use of detailed voice feedback was

extremely helpful in the data entry process. They also mentioned that operating a module in an unfamiliar operating system environment was less intimidating when the speech interface and the voice feedback were used.

User C	Speech	Keyboard & Mouse
● Average Task Completion Time	30 secs.	45 secs.
● Number of Errors in Performing Task	2	5
● Number of Tasks Completed in 10 Min.	20	13

Training Time	Recognition Rate
23 min. 15secs	95.2%

User D	Speech	Keyboard & Mouse
● Average Task Completion Time	32 secs.	40 secs.
● Number of Errors in Performing Task	4	5
● Number of Tasks Completed in 10 Min.	19	15

Training Time	Recognition Rate
21 min. 47secs	96.8%

Table 4.4: Test Set 2: The Computer Experts without any PM Experience

As expected, the non expert users took the longest average time to complete a task. However, the number of tasks completed using the speech interface was greater than the number of tasks done when using the keyboard and the mouse (table 4.5). These users echoed the same comments of the previous set of users about the speech interface and the feedback of the voice output.

User E	Speech	Keyboard & Mouse
Average Task Completion Time	35 secs.	45 secs.
Number of Errors in Performing Task	6	12
Number of Tasks Completed in 10 Min.	17	13

Training Time	Recognition Rate
20 min. 45secs	97.0%

User F	Speech	Keyboard & Mouse
Average Task Completion Time	37 secs.	48 secs.
Number of Errors in Performing Task	0	0
Number of Tasks Completed in 10 Min.	13	7

Training Time	Recognition Rate
24 min. 05secs	96.3%

Table 4.5: Test Set 3: The Non-Expert Users

Based on the values of the recognition rates, response times as well as the robustness or overall recognition rates obtained, it is felt that the speech input system is adequate for the application at hand. The effectiveness and the ease of use were obvious for novice computers users or for users who do not know the application system environment. The voice feedback proved especially useful to users that do not know the options available at a given level.

4.6 Future Work and Recommendations

The evaluation process has reached the final phase of stage three of the evaluation scheme. The Fluid Balance system is now ready for field testing at the ICU of the Montreal Children's hospital. The assessment of the user's interaction with the software remains to be completed. Ideally, from a user's point of view, a system's interface is the system. Therefore, special attention should be given to the users' human factors and to any resistances encountered. The final assessment should be accompanied with careful training sessions and customer support, in order to make sure that the software will be accepted and viewed as a positive contribution to the users' everyday routines. Even if the two previous stages were successful and the Fluid Balance system performed as expected by the developers, the final word rests with the end users that have to accept the product.

A number of future enhancements are evidently required. The Fluid Balance system should be adapted to a multi-user environment, in which more than one nurse could use the system simultaneously. Improved help facilities should incorporate the use of multimedia technology such as video and voice. Another suggested enhancement of the FBS would allow users to operate it in a "mixed mode", using combinations of the keyboard, the mouse and the voice interface. This would require a major re-design of the system, since it is currently designed to be operated using the keyboard and mouse or completely "hands free - eyes free" with the voice interface.

Chapter 5 Conclusion

The development and the evaluation of a Fluid Balance system is discussed in this thesis. A literature survey on successful patient data management systems in the Intensive Care unit is presented. Issues on Speech I/O systems and their applications, including the factors of evaluations, are illustrated with an outlook to the future of such systems. The hardware and software architectures of the PDMS are outlined. The implementation of the Fluid Balance modules and of the speech interface is explained in detail. A strategy for evaluating the implementation of the Fluid Balance system is presented based on the unit testing, system testing and user acceptance. The results are presented and discussed. A number of future enhancements and recommendations are proposed.

References

[AAMC, 1986] Association of American Medical Colleges, "Report of the Steering Committee on the Evaluation of Medical Information Science in Medical Education," Washington, 1986.

[Allen, 1987] J. Allen, *"Natural Language Understanding,"* Menlo Park, CA: Benjamin/Cummings, 1987.

[Andreas et al., 1992] A.S. Spanias, and F.H. Wu, "Speech Coding and Recognition: a Review," *The Institute of Electronics, Information and communication Engineers,* on Fundamentals of Electronics, Communications and Computer Sciences, vol. E75-A, no. 2, pp. 132-148, Feb. 1992.

[Beizer, 1983] Boris Beizer, *"Software Testing Techniques,"* Van Nostrand Reinhold Company Inc, NY, 1983.

[Beizer, 1984] Boris Beizer, *"Software System Testing and Quality Assurance,"* Van Nostrand Reinhold Company Inc., NY 1984.

[Bergeron et al., 1990] B. Bergeron, and S. Locke, "Speech Recognition as a User Interface," *MD Computing,* vol. 7, no. 5, pp. 329-334, 1990.

[Berkley et al., 1990] D.A. Berkley, and J.L. Flanagan, "HuManNet: An Experimental Human-Machine Communications Network Based on ISDN Wideband Audio," *AT&T Technical Journal,* vol. 69, no. 5, pp. 87-97, Sept/Oct 1990.

[Blum, 1986] B.I. Blum, "Clinical Information Systems: a Review," in Medical Informatics, special issue, *Western Journal of Medicine,* Dec. 145, pp. 791-797, 1986.

[Carlson et al., 1989] R. Carlson, and B. Granstrom, "Evaluation and Development of the KTH Test-to-Speech System on the Segmental Level," *Proc. ESCA Workshop,* Noordwijkerhout, The Netherlands, 1989.

[Chollet et al., 1988] G. Chollet, and C. Montacie, "Evaluating Speech Recognizers and Data Bases," Eds. H. Niemann, M. Lang, and G. Sagerer, *Recent Advances in Speech Understanding and Dialog Systems,* London, Springer-Verlag, pp. 345-348, 1988.

[COVOX, 1990] Covox Inc., Eugene, OR, Covox Voice Master Key II, 2.04 ed., April 1990.

[Crochiere et al., 1986] R.E. Crochiere and J.L. Flanagan, "Speech Processing: An Evolving Technology," *AT&T Technical Journal*, vol. 65, no. 5, pp. 2-11, Sept./Oct. 1986.

[Cummiskey et al., 1973] P. Cummiskey et al., "Adaptive Quantization in Differential PCM Coding of Speech", *Bell System Technical Journal*, vol. 52, no. 7, pp. 1105, Sept. 1973.

[Danielsen et al., 1989] S. Danielsen, P. Dalsgaard, B. Lindberg, and C. Henriksen, "Speech Databases and Mass Data Storage," Eds. A. Fourcin, G. Harland, W.J. Barry, and V. Hazan, *Speech Input and Output Assessment: Multilingual Methods and Standards*, Chichester: Ellis Horwood, 1989.

[Davenport, 1987] D. O. Davenport, "Computerized Monitoring Systems," *Nursing Clinics in North America*, vol. 22, no. 2, pp 495-501, July 1987.

[Delogu et al., 1991] C. Delogu, A. Paoloni, and P. Pocci, "New Directions in the Evaluation of Voice Input/Output Systems," *IEEE Journal on Selected Areas in Communications*, vol. 9, no. 4, pp. 566-573. May 1991.

[Dintruff et al., 1987] D.L. Dintruff, and H.J. Luttinger, "Evaluation of Speech Technology in Mental Health Assessment," *Speech Technology*, pp. 32-38, March/April, 1987.

[Dudley, 1939a] H. Dudley, "Remaking Speech," *Journal of the Acoustical Society of America*, vol. 11, pp. 169, 1939.

[Dudley, 1939b] H. Dudley, "The Vocoder," *Bell Laboratories Record*, vol. 17, pp. 122, 1939.

[Duerer et al., 1992] H. Duerer, K. Wang, M.B. Wischnewsky, and J. Zhao, "Intensive Help, A Knowledge-Based System for Intensive Care Units," *Proceedings of the Fifth Annual IEEE Symposium on Computer-Based Medical Systems*, pp. 336-344, May 1992.

[Fagan et al., 1990] L.M. Fagan and L.E. Perrault, "The Future of Computer Applications in Health Care," Eds. E. H. Shortliffe and L. E. Perrault. *Medical Informatics*, chapter 20, pp. 620-644, Reading, Massachusetts: Addison-Wesley Publishing Company, Inc., 1990.

[Flanagan et al., 1985] J.L. Flanagan, J.D. Johnson, R. Zahn, and G. Elko, "Computer Steered Microphone Arrays for Sound Transduction in Large Rooms," *Journal of Acoustical Society of America*, vol. 78, no. 5, pp. 1508-1518, Nov. 1985.

[Flanagan et al., 1990] J.L. Flanagan, and C.J. Del Riesgo, "Speech Processing: A Perspective on the Science and Its Applications," *AT&T Technical Journal*, vol. 69, no. 5, pp. 2-13, Sept./Oct. 1990.

[Gardner 1990] R. M. Gardner, "Patient-Monitoring Systems." Eds. E. H. Shortliffe and L. E. Perrault. *Mecical Informatics*, chapter 12, pp. 366-399, Reading, Massachusetts: Addison-Wesley Publishing Company, Inc., 1990.

[Gardner et al., 1989] R. M. Gardner, D. F. Sittig, and M. C. Budd, "Computers in the Intensive Care Unit: Match or Mismatch?" Eds. W.C. Shoemaker and S. Ayres. *Textbook of Critical Care*, chapter 26, pp248-259, Philadelphia: W.B. Saunders Company, Inc., 1989.

[Gardner, 1986] R. M. Gardner, "Computerized Management of Intensive Care Patients." *M.D. Computing*, vol. 3, no. 36, 1986.

[Gerson et al., 1990] Gerson and Jasiuk, "Vector Sum Excited Linear Prediction (VSELP) Speech Coding at 8 KBPS," *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing*, pp. 461-464, New Mexico, April 1990.

[Glaeser et al., 1975] D.H. Glaeser and L.J. Thomas, "Computer Monitoring in Patient Care," *The Annual Review of Biophysics and Bioengineering*, vol. 4, pp. 449-476, 1975.

[Green et al., 1991] C.A. Green, K.J. Gilhooly, R. Logie, and D.G. Ross, "Human Factors and Computerization in Intensive Care Units: a Review," *International Journal of Clinical Monitoring and Computing*, vol. 8, pp. 167-178, 1991.

[Hammond, 1987] W. E. Hammond, "Patient Management Systems: The Early Years," Eds. B.I. Blum and K. Duncan, *A History of Medical Informatics*, pp. 370-381, ACM Press History Series, Maryland, Nov. 1987.

[Haseman, 1988] D.B. Haseman, "Voice Recognition Used in Radiology Interpretation," *Proceedings of Speech Technology*, New York, NY: Media Dimensions, 1988.

[Hirschberg et al., 1990] J.B. Hirschberg, S.A. Riederer, J.E. Rowley, and A.K. Syrdal, "Voice Response Systems: Technologies and Applications," *AT&T Technical Journal*, vol. 69, no. 5, pp. 42-51, Sept./Oct. 1990.

[Hodge, 1987] M.H. Hodge, "Direct Use by Physicians of the TDS Medical Information System," Eds. B.I. Blum and K. Duncan, *A History of Medical Informatics*, pp. 345-356, ACM Press History Series, Maryland, Nov. 1987.

[Humphreys et al., 1989] B.L. Humphreys and D.A.B. Lindberg, "Building the Unified Medical Language System," *Proceedings of the 13th Conference on Computer Applications in Medical Care,* IEEE Computer Society Press, Los Alamitos, pp. 475-480, California, 1989.

[Jayant et al., 1984] N.S. Jayant, and P. Noll, "Digital Coding of Waveforms," *Prentice-Hall Inc.,* Englewood Cliffs, NJ, 1984.

[Jekosch, 1989] U. Jekosch, "The Cluster-Based Rhyme Test: A Segmental Synthesis Test for Open Vocabulary," *Proc. ESCA Workshop,* Noordwijkerhout, The Netherlands, 1989.

[Josenhans et al., 1986] J.G. Josenhans, J.F. Lynch, Jr., M.R. Rogers, R.R. Rosinski, and W.P. Vandame, "Speech Processing Applications Standards," *AT&T Technical Journal,* vol. 65, no. 5, pp. 23-33, Sept./Oct. 1986.

[Karis et al., 1991] D. Karis, and K. Dobroth, "Automating Services with Speech Recognition over the Public Switched Telephone Network: Human Factors Considerations," *IEEE Journal on Selected Areas in Communications,* vol. 9, no. 4, pp. 574-585, May 1991.

[Karl, 1991] L.R. Karl, "Speech Versus Mouse Commands for Word Processing: An Empirical Evaluation," *The International Journal of Man-Machine Studies,* vol. 39, pp. 667-687, 1993.

[Lange, 1990] H.R. Lange, "Voice Recognition and Voice Response: a Report on Tomorrow's Technologies," *Proceedings of the National Online Meeting,* pp. 233-240, 1990.

[Lee et al., 1991] L. Lee et al., "A Fully Parallel Mandarin Speech Recognition System with very Large Vocabulary and Almost Unlimited Texts," *Proc. ISCAS,* -91, Singapore, June 1991.

[Lennig, 1989] M. Lennig, "Using Speech Recognition in the Telephone Network to Automate Collect and Third-Number-Billed Calls," *Proceeding of Speech Technology,* pp. 124-125, 1989.

[Liang et al., 1988] M.D. Liang, and K. Narayanan, "Voice Activated Robotic Retrovit System Surgical Microscopes," *Proceedings of Speech Technology,* New York, NY: Media Dimensions, 1988.

[Liang et al., 1989] M.D. Liang, and K. Narayanan, "The Application of Voice Recognition to Robotic Positioning of a Hospital Bed," *Speech Technology,* vol. 5, pp. 30-33, 1989.

[Mangold, 1988] H. Mangold, "Principles of Automatic Processing of Speech Signals and Their Application in Medical Technology and for Aids for

Handicapped," *Medical Progress Through Technology*, vol. 14, pp 39-56, 1988.

[Manzano et al., 1980] J.L. Manzano, J. Villalobos, R.N. Church, and J.J. Manzano, "Computerized Information System for ICU Patient Management," *Critical Care Medicine,* vol. 8, no. 12, pp. 745-747, 1980.

[McCray, 1989] A.T. McCray, "The ULMS Semantic Network," *Proceedings of the 13th Conference on Computer Applications in Medical Care,* IEEE Computer Society Press, Los Alamitos, pp. 503-507, California, 1989.

[McCree et al., 1991] A. McCree, and T. Barnwell, "A New Mixed Excitation LPC Vocoder," *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing*, pp. 593-596, Toronto, May 1991.

[McIntyre et al., 1989] J.W.R. McIntyre, and T.M. Nelson, "Application of Automated Human Voice Delivery to Warning Devices in an Intensive Care Unit: A Laboratory Study," *International Journal of Clinical Monitoring and Computing,* vol. 6, pp. 255-262, Dec. 1989.

[Michalski et al., 1983] R.S. Michalski, J.G. Carbonell, and T.M. Mitchell, "Machine Learning: an Artificial Intelligence Approach," *Palo Alto,* Tioga Publishing Company, 1983.

[Miller et al., 1986] R.A. Miller, M.A. McNeil, S.M. Challinor, F.E. Masorie, and J.D. Myers, "The INTERNIST-1/Quick Medical Reference," *Western Journal of Medicine,* project-status report, no. 145, pp. 816-822, 1986.

[Morishima et al., 1991] S. Morishima, and H. Harashima, "A Media Conversion from Speech to Facial Image for Intelligent Man-Machine Interface," *IEEE Journal of Selected Areas in Communications,* vol. 9, no. 4, pp. 594-600, May 1991.

[Myers et al., 1980] C. Myers, L.R. Rabiner, and A.E. Rosenberg, "Performance Tradeoffs in Dynamic Time Wraping Algorithms for Isolated Word Recognition," *IEEE Transactions on Acoustics, Speech, and Signal Processing,* vol. 28, no. 6, pp. 623-635, Dec. 1980.

[Myers, 1979] Glenford Myers, *"The Art of Software Testing,"* John Wiley & Sons, NY, 1979.

[Nakatsu, 1990] R. Nakatsu, "Anser an Application of Speech Technology to the Japanese Banking Industry," *IEEE Computer,* vol. 23, no. 8, pp. 43-48, August 1990.

[Nikil et al., 1990] J.S. Nikil, V.B. Lawrence, and D.P. Prezas, "Coding of Speech and Wideband Audio," *AT&T Technical Journal,* vol. 69, no. 5, pp. 25-41, Sept./Oct. 1990.

[NLM, 1986] National Library of Medicine, "A Vision of the Future," *Medical Informatics,* Report of Panel 4, pp. 16-25, December 1986.

[Nolan-Avila et al.] L. Nolan-Avila and M. Shabot, "Life Without Computers in the ICU," *Critical Care Nurse*, vol. 7, no. 3, pp 80-83, 1987.

[O'Malley, 1990] M.H. O'Malley, "Text-To-Speech Conversion Technology," *IEEE Computer,* vol. 23, no. 8, pp. 17-23, August 1990.

[Panisset et al., 1989] J.F. Panisset, S. Malowany, N. Khoury, D. Lambidonis, A.S. Malowany, F.A. Carnevale, R. Gottesman, and A. Rousseau, "An Intensive Care Unit Patient Data Management System," *Proceedings Graphics Interface'89,* London, Ontario, pp. 275-282, June 1989.

[Peacocke et al., 1990] R.D. Peacocke, and D.H. Graf, "An Introduction to Speech and Speaker Recognition," *IEEE Computer,* vol. 23, no. 8, pp. 26-33, August 1990.

[Perrault et al.,1990] L. E. Perreault, and G. Wiederhold, "System Design and Evaluation," Eds. E. H. Shortliffe and L. E. Perrault. *Mecical Informatics*, chapter 5, pp. 151-178, Reading, Massachusetts: Addison-Wesley Publishing Company, Inc., 1990.

[Perry 1986] William E. Perry, *"How to Test Software Packages,"* New York: Wiley, John & Sons, 1986.

[Petroni, 1991] M. Petroni, " A Speech Interface for Bedside Data Entry in an Intensive Care Unit." M. Eng. thesis, Department of Electrical Engineering, McGill University, July 1991.

[Poritz, 1988] A. Poritz, "Hidden Markov Models: A Guided Tour," *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing,* vol. 1, pp. 7-13, 1988.

[Rabiner et al., 1986] L.R. Rabiner, and B.H. Juang, "An Introduction to Hidden Markov Models," *IEEE ASSP Magazine,* vol. 3, no. 1, pp. 4-16, Jan 1986.

[Rash, 1989] W. Rash, "A Helping Hand," *Byte,* no. 14, pp. 129-130, 1989.

[Robbins et al., 1987] A.H. Robbins, D.D. Horowitz, and M. Srinivasan, "Speech-Controlled Generation of Radiologic Reports," *Radiology,* vol. 164, no. 2, pp. 569-573, 1987.

[Robbins et al., 1988] A.H. Robbins, M.E. Vincent, K. Shaffer, R. Maietta, and M.K. Srinivasan, "Radiology Reports: Assessment of a 5000-Word Speech Recognizer," *Radiology,* vol. 167, no. 3, pp. 853-855, June 1988.

[Rollings, 1985] A.M. Rollings, "Speech Recognition and Manner of Speaking in Noise and in Quiet," Eds. L. Borman, and B. Curtis, *Human Factors in Computing Systems,* Amsterdam, The Netherlands, North Holland, 1985.

[Rosen, 1988] R.A. Rosen, "A Voice-Activated System in Diagnostic Radiology," *Proceedings of Speech Technology,* New York, NY: Media Dimensions, 1988.

[Rowberg, 1987] A.H. Rowberg, "Voice Control and Input to a Radiology Information System," *Proceedings of Speech Technology,* New York, Media Dimensions, 1987.

[Schneiderman, 1987] B. Schneiderman, "Designing the User Interface: Strategies for Effective Human-Computer Interaction," *Reading,* Massachusetts: Addison-Wesley Publishing Company, 1987.

[Schwab et al., 1985] E.C. Schwab, H.C. Nusbaum, and D.B. Pisoni, "Some Effects of Training of the Perception of Synthetic Speech," *Human Factors,* vol. 27, no. 4, pp. 395-408, 1985.

[Shiffman et al., 1991] S. Shiffman, A.W. Wu, A.D. Poon, C.D. Lane, B. Middleton, R.A. Miller, F.E. Masarie Jr., G.F. Cooper, E.H. Shortliffe, and L.M. Fagan, "Building a Speech Interface to a Medical Diagnostic System," *IEEE Expert,* vol. , no. , pp. 41-49, Feb. 1991.

[Shwe, 1990] M. Shwe, "A Probabilistic Reformulation of the Quick Medical Reference System," *IEEE Proceedings of the 14th Conference on Computer Applications in Medical Care,* IEEE Computer Society Press, Los Alamitos, pp. 816-822, California, 1990.

[Smith et al., 1986] M.J.T. Smith, and T.P. Barnwell, "Exact Reconstruction Techniques for Tree-Structured Sub-Band Coders," *IEEE Transactions on Acoustics, Speech, and Signal Processing,* vol. 34, pp. 434, June 1986.

[Steeneken et al., 1988] H.J.M. Steeneken and F.W.M. Geurtsen, "Description of the RSG-10 Noise Data-Base," *TNO Institute for Perception,* Soesterberg, The Netherlands, rep. IZF, vol. 3, 1988.

[Strathmeyer, 1990] C. Strathmeyer, "Voice in Computing: An Overview of Available Technologies," *IEEE Computer,* vol.23, no. 8, pp. 10-15, August 1990.

[Tong et al. 1982] H.D. Tong, and A. Gupta, "Personal Computers," *Scientific America,* vol. 247, no. 87, 1982.

[VERBEX, 1990] VERBEX Voice Systems, Edison, NJ, *"VERBEX Conversational Voice Input/Output System Grammar Development Manual,"* 2.00 ed., May 1990.

[Voiers, 1983] W.D. Voiers, "Evaluating Processed Speech Using the Diagnostic Rhyme Test," *Speech Technology*, vol. 1, no. 4, pp. 30-39, 1983.

[Waterworth, 1984] J.A. Waterworth, "Interaction with Machines by Voice. A Telecommunications Perspective," *Behav. Inform. Technol.*, vol. 3, no. 2, pp. 163-177, 1984.

[Wiederhold et al.,1990] G. Wiederhold, and L. E. Perreault, "Hospital Information Systems." Eds. E. H. Shortliffe and L. E. Perrault. *Medical Informatics*, chapter 7, pp. 219-243, Reading, Massachusetts: Addison-Wesley Publishing Company, Inc., 1990.

[Wulfman, 1988] C.E. Wulfman, "Speech Interface with a Medical Expert System," *Proceedings of Speech Technology*, New York, NY: Media Dimensions, 1988.