

MEDICAL DOCUMENTATION AUTOMATION FOR PSYCHOTHERAPISTS

PRACTICING IN GERIATRIC LONG-TERM CARE

RESIDENTIAL FACILITIES

THESIS

Presented to the Graduate Council of
Southwest Texas State University
In Partial Fulfillment of
The requirements

For the Degree

Master of SCIENCE

By

Jeffrey Thompson, B.L.S

San Marcos, Texas
December 2001

COPYRIGHT 2001 by Jeffrey Thompson

Acknowledgements

I would like to acknowledge and extend my appreciation to the members of my committee. My sincere thanks go to Charles Johnson and Wilbon Davis.

This thesis would not have been possible without the cooperation of H. John Durrett, Ph.D., Esq. His experience, insight, analyses, and suggestions were invaluable in addressing the issues brought out by this topic. Thank you John. I am a better software professional because of this experience.

This thesis was submitted for committee consideration on 9 Nov 2001.

TABLE OF CONTENTS

LIST OF FIGURES.....	viii
LIST OF TABLES.....	x
I. Problem Domain	11
Current Company Process.....	13
Delivery Stage Process Details	16
Automating Document Generation.....	19
II. Requirements Analysis	22
Software Methodology Selection	22
User Requirements.....	26
Corporate Requirements.....	28
Environmental Requirements	31
III. Existing Solution Alternatives.....	36
Therascribe 4.0.....	36
Quicdoc 3.7.....	48
Analysis Conclusions	63
IV. Data Design.....	65
Data Identification	65
Other Data.....	70
Problem Domain Objects	72
V. Sentence Engine Design	79
First Design Iteration - Frames.....	81
Sentence Frames	83
Second Iteration – Linked Elements.....	87
Sentence Design Data Requirements	100
VI. Hardware design.....	102

VII.	User Interface Design	116
VIII.	Prototype Testing	160
IX.	Conclusions	167
	BIBLIOGRAPHY	170

LIST OF FIGURES

Figure 1: Therascribe 4.0 Main Page.....	38
Figure 2: Demographics Sub Category, Personal Data Entry.....	39
Figure 3: Progress Notes Section Interface	40
Figure 4: “Presentations for Depression” Sentence Selection List Example	41
Figure 5: Objective Rating Assignments.....	42
Figure 6: Narrative Progress Note.....	42
Figure 7: Progress Notes Section of Clinical Report	45
Figure 8: Manual / Therascribe Time Test Results	48
Figure 9: Quicdoc Main Screen	49
Figure 10: Quicdoc Patient Data Screen.....	50
Figure 11: Quicdoc Run-Time Error Message.....	51
Figure 12: Quicdoc Progress Note - Session Information Screen.....	52
Figure 13: Quicdoc Context-Process Screen.....	54
Figure 14. Quicdoc Progress Note - Interventions Screen	55
Figure 15. Quicdoc Progress Note – Plan Screen.....	57
Figure 16. Quicdoc Progress Note – Risk Management.....	57
Figure 17. Quicdoc Progress Note Report Screen	59
Figure 18. Quicdoc Progress Note – Quicword Report Preview	61
Figure 20. Sentence Engine ERD.....	75
Figure 21. Phase 1 – Main Screen.....	117
Figure 22. Phase 1 Patient Data Screen Model.....	119
Figure 23. UI Sentence Input Screen.....	120
Figure 24. UI Notes Screen	120
Figure 25. UI Phase 2 – Main Screen	124
Figure 26. Phase 2 UI Sessions Tab (Default).....	124
Figure 27. Phase 2 UI Sessions Tab (Nominal)	125
Figure 28. Phase 2 Sentence Frame Dropdown List Options	127
Figure 29. Phase 2 Sentence Frame UI	127
Figure 30. Select Target	128
Figure 31. Select Amount.....	128
Figure 32. Select Duration.....	129
Figure 33. Final Sentence.....	129
Figure 34. Phase 3 Main Screen.....	132
Figure 35. Phase 3 Facility Data UI Form	134
Figure 36. Phase 3 UI Session Interface.....	135
Figure 37. Phase 3 UI Session Interface (Session Tab)	135
Figure 38. Phase 3 Sentence UI Form	137
Figure 39. Phase 3 Sentence UI Form – Less Elements	138

Figure 40. Phase 3 Sentence UI Form – More Elements	138
Figure 41. Phase 4 Main Screen.....	141
Figure 42. Phase 4 Navigation Nodes	142
Figure 43. Phase Four Sentence Tree.....	144
Figure 44. Phase Four Sentence Negation Menu.....	146
Figure 45 TOAD UI – Status Bar Error.....	155
Figure 46. TOAD UI Error Message Box	156
Figure 47. Prototype Timing Test Results	165

LIST OF TABLES

Table 1. Session Notes Form Data Element List	66
Table 2. RPS Data Elements	68
Table 3. Weekly Roster Data Elements	69
Table 5. Current Status Sentence Architecture	90
Table 6. Therapeutic Interventions Sentence Architecture	94
Table 7. Progress Sentence Architecture	95
Table 8. Everex A-20 Battery Endurance	108
Table 9. Prototype GUI Design Checklist Results	150
Table 10. Microsoft Good Interface Design Checklist.....	158

I. PROBLEM DOMAIN

America is an aging nation. The population is living longer better lives than any previous generation in history. Some research estimates place the percentage of Americans 65 years of age or older by 2020 at 17%. Along with this aging populace is the increasing chance that at some point after age 65, it is 43% likely that a person will spend some period of time in a nursing home. An unfortunate consequence of this rising population of nursing home residents is the high percentage of residents with mental disorders. Most estimates place the percentage of nursing home residents with mental disorders at 50%-90%, with the prevalent maladies being dementia, delirium, and depression. These estimates point to a very strong demand for geriatric long-term residential care mental health services (Zarit, 4, 11, 321).

Remediation of these mental health problems are complicated by the fact that most all resident patients are under Medicare insurance coverage rather than private insurance. Medicare, through the auspices of the Health Care Finance Administration [HCFA] (recently renamed to the Center for Medicare & Medicaid Services [CMS]), has set certain documentation requirements in order for mental health claims to be paid. Failure to meet or comply with the

requirements results in denial of payment or recoupment, a term referring to the demand for repayment of funds paid for previous claims. In this environment lay the problem of mental health documentation.

SPS [Senior Psychology Services] is a professional company of licensed psychotherapists that practice geriatric long-term residential mental health care. Very few of the thousands of patients seen by company therapists each month are covered by funding other than Medicare. Medicare requirements, therefore, have a profound impact on the financial status of each practicing therapist. Those therapists who are able to generate notes that meet Medicare requirements have the highest percentage of net gain – those whose notes are not in compliance suffer not only loss of revenue through denial and/or recoupment, but also increased workload when attempting to recover the lost funds by re-documenting, challenging denials in administrative court, and increasing their workload to offset lost funds.

The company considers the issue a serious problem. A Quality Management group was created just to instruct and audit therapist session documentation to ensure compliance. But, even with internal auditing, a practicing therapist at the company put the number of denied claims at over 10% (Durrett, 28 May 1999). The company suffers from a documentation process that is susceptible to certain Medicare documentation requirement breakdowns. The

company uses a manual system of paper forms, with scant human audits, to process claims for psychotherapy services rendered. An examination of the system reveals the possibility for an information technology automation solution.

Current Company Process

The life cycle of the company's geriatric psychotherapy service delivery consists of three sequential stages: referral, delivery, and cessation. The referral stage is composed of patient acquisition activities. Only those patients with medical orders from their primary care physician are eligible for Medicare reimbursement. To ensure all patients have Medicare funding available, this is the only entryway for new patients. The referral stage is characterized by diagnostic and administrative processes serving facilitation of the delivery stage. Referrals begin through requests for psychological assessments. Facility staff and/or patient primary care physicians make the request to the company via the Request for Psychological Services [RPS]. Upon receipt of the signed RPS form to the company, the referred resident is assigned a therapist. The therapist receives permission from the company to conduct the initial interview and diagnostic procedures. Time is scheduled to access the resident and conduct assessment tests. After the therapist has administered the assessment, a written report is produced detailing the results. This report is reviewed by the resident's physician, who holds the authority to authorize psychotherapy treatment. If so

ordered by the physician, written doctor's orders are delivered to the company authorizing some number of psychotherapy sessions. Upon receipt of the doctor's orders, the resident is added as a patient to a therapist's patient list (known as the "weekly roster"). Once the patient appears on the weekly roster, service delivery may commence. Should the resident physician decline to authorize psychotherapy, the company is paid a diagnostic fee to cover the assessment, and the company-resident relationship is terminated (cessation stage).

The delivery stage begins once a resident appears on a therapist's weekly roster form. The delivery stage covers the psychotherapy services rendered in accordance with the doctor's orders of the patient's primary physician. Delivery is characterized by psychotherapy sessions, occurring at regular intervals, between therapist and patient. Each authorized psychotherapy session must be documented by the therapist in writing to comply with HCFA regulations for reimbursement under Medicare (Draft). It is this reiterative session documentation process that is addressed by this paper's solution.

Doctor's orders specify the number of psychotherapy sessions authorized. Once the allotment of sessions has been consumed, the cessation stage is entered. There is little documentation specific to the stage itself. The final psychotherapy session should make an entry that the psychotherapy regimen is concluded. Thus,

each delivery stage requires a cessation stage in order to complete the entire therapy process. Cessation is documented by act of removing a patient's name from the therapist's weekly roster. It is possible that the resident may have psychotherapy ordered again for the same or different symptoms at some point in the future. New doctor's orders for psychotherapy begin the service delivery cycle again.

Of the three stages, the delivery stage is unique in that the documentation it produces is expected to be in a uniform format and adhere to a set of guidelines. Referral stage documents may be of a varied nature – not all diagnostic tools are used for all patients. HCFA does not regulate which particular diagnostic procedure is used to determine if a resident requires psychotherapy services. There is an established battery of diagnostic tools and procedures that are well understood and precedent in assessing mental state. HCFA regulatory concerns arise in consideration of the methodology used to treat the diagnosed condition(s) (Draft, 10). Treatment methods are varied and center on verbal interaction between therapist and patient. This activity does create document artifacts as a by-product of the activity, such as completed test forms, or tabulations of responses to questions. The documentation of psychotherapy sessions is an attempt to ensure the patient receives professional treatment of a type that addresses the diagnosed malady, and to assess the

patient's response to the therapeutic interventions. Documentation for referral and cessation stages serve the business interests of the service provider and are not designed to satisfy any particular regulatory requirement(s).

Delivery Stage Process Details

The company has devised a paradigm for documenting psychotherapeutic activities based on the general SOAP model of symptoms, observations, actions, and progress. The current company session document implements this paradigm through the physical make-up of the session document form. The company uses a symptom-intervention-progress analogy where there are three distinct sections to the body of a session note form. A session note form is a 3-part carbon copy document consisting of corporate letterhead, patient and session detail elements, and three ruled sections providing space for manual script entry ("Current Status" [Symptoms], "Provider Interventions" [Interventions], and "Assess progress towards goal" [Progress]). This document is pre-printed and provided to therapists through the company. The company provides a number of preprinted forms for each patient – enough to satisfy the number of sessions authorized in doctor's orders, and to allow for errors or mistakes requiring a rewrite of the note form.

The session note form contains baseline and biographical data such as patient identification, diagnosis codes, session date and type, facility, and other

associated data. This form of data is fairly static in nature, with few elements likely to change over the prescribed duration of psychotherapy. The session note form currently provides only the patient name and a company specific patient ID code preprinted. All other data are left for the therapist to enter manually by the servicing therapist.

In the current status section, the expectation is that patient behavior will be documented – what they did, what they said, what behaviors have been demonstrated. This section is intended to explain why the patient requires psychotherapy. It is intended to be a specific listing of behaviors that are derivative of the initial diagnostic assessment for which the psychotherapy was ordered.

The symptom section is followed by the intervention section. Interventions are those actions taken by the psychotherapist during the session to modify the patient's behavior or mental state. Intervention works at multiple levels. Interventions may be done to address the overall affliction, or targeted to eliminate of certain behaviors or mental functioning. Interventions represent the reason a Ph.D. level psychotherapist is required to address the mental problems ascribed to the patient. Regardless of the level at which the intervention is targeted, the intervention act itself must be professionally competent, therapeutically efficacious, and appropriate for the patient's symptoms. Specific

techniques and methods used in the session must be documented.

The progress section completes the session note. The intent of progress is to provide an assessment of the effectiveness of intervention actions. Like intervention, progress can occur at general (overall diagnosis) and specific (behaviors) levels. Progress is intended to assess patient behavior patterns over time, in a hierarchical way. In general, symptoms improve, remain steady, or worsen. Progress section comments should communicate this information, and (ideally) provide an empirical scale for the magnitude, rate, and direction of progression.¹

The patient's official medical chart should be updated to include the original copy of the session note document upon delivery of service. The company position is to have the official, signed, completed session document in the patient's chart within hours of session completion. The triplicate carbon form's top copy (white) is placed into the patient's medical chart, a copy of the session note is submitted to company offices for payment processing (pink), and a copy is available for the therapist to keep in private records, if desired (yellow) (Durrett, 28 May 1999).

This level of documentation requires a large commitment of time from the practicing therapist. A company therapist stated he spent "about 25 minutes" per

¹ "Effective Documentation for Long Term Care: Handout for Psychologists", pg 4-5

session just in filling out the progress notes for the session (Durrett, 10 Oct 1999). For a full-time therapist seeing 30 patients a week, about 15 extra hours per week are required to complete the session process. This time is in addition to time spent traveling to and from care facilities, time spent with patients, and other time required getting from one task to the next. With this type of additional burden placed on the therapist, the high rate of denial and recoupment due to errant or incomplete documentation is explicable, but unacceptable.

Automating Document Generation

To automate the session documentation process, two distinct data groups must be dealt with. Based on reuse and data value persistence, there are two data groups represented on the session note form: dynamic, which represents the freeform sentences entered into the symptom-intervention-progress sections; and static, representing the data whose values seldom change once set. Static data requires little to no structure to support it. Static data are analogous to name-value pairs, where the data type supplies all necessary data access information. The dynamic data sections are different in that a sentence contains data elements within a context, constrained by general and specific syntactic rules (English grammar) affording multiple manifestations for the same intended message. In addition, the dynamic data of a session note is subjective in that the data are indirectly processed by users. Readers of session notes could possibly have

varying interpretations of any particular sentence. Sentences transmit data to readers via a narrative context, and the narrative context is by and large incomplete. A sentence attempts to send readers a complex, sophisticated message. In order to derive the meaning of the sentence, readers use their own experience and perspective to complete the message the sentence is sending. This interpretive aspect of sentences as a means of data conveyance makes for an error-prone and faulty data system. This problem is highlighted by the documentation guidelines enacted by the company. These guidelines attempt to curtail ambiguity by providing for legible script containing as many concrete data elements as possible. Sentences will always have a degree of uncertainty regarding their interpretation – a successful sentence automation engine reduces this uncertainty to a level acceptable to all involved consumers of the sentence information.

As a consequence of automating document generation, the company should expect decreased rates of denial and recoupment, increased employee satisfaction through reduced therapist administrative workload, and intangible benefits such as enhanced reputation for professionalism and a leadership role in the geriatric mental health field. Given that therapists are already under formidable workloads when manually processing session notes, documentation automation could be argued to offer no greater encumbrance to therapists than

the current documentation system imposes.

II. REQUIREMENTS ANALYSIS

Software Methodology Selection

A software methodology provides the framework from which the software product is created. Selecting the appropriate methodology for this project involved identification of the project's characteristics and matching those characteristics to a methodology's strengths. Both current and traditional methods were considered in order to locate the best option.

Current software development methodologies center around two aspects: the human developers and the business need to make schedule commitments. The foundation for these current methodologies is broadly described as the concept of rapid application development [RAD]. McConnell describes RAD as not any specific tool or method, but as software projects that need to emphasize development speed (2). RAD promotes four general strategies and four dimensional aspects that carry out the strategy guidelines. The general strategies are: avoid classic mistakes, apply development fundamentals, manage risks to avoid catastrophic setbacks, and apply schedule-oriented practices. The four dimensions are people, process, product, and technology. The key to successful rapid development is to determine which of the four dimensions are limited and

which can be leveraged for maximum advantage, then stretch each dimension to the utmost (McConnell, 22).

Examples of current methodologies built on the RAD concept include agile software development [ASD] and extreme programming [XP]. Both are specific methodologies with similarities.

There are five basic characteristics of ASD. Team size is fairly small, with two to eight programmers total involved. Usage experts (customer representatives) are part of the development team and integrally involved. Product releases occur within a one to three month cycle. Fully automated regression testing is employed. Finally, programmers on the team are mostly experienced developers, not novices (Cockburn, 178-80).

ASD holds the following values: Individuals and interaction over process and tools; Working software over comprehensive documentation; Customer collaboration over contract negotiation; and Responding to change over following a plan. ASD is goal driven, and the primary goal of ASD is to deliver software; documentation comes second (Cockburn 213-218).

XP builds on the concept of ASD with additional processes. The following is an example of XP used in a development environment. A team is composed of a small group of programmers and customer representatives. Requirements are elicited through “story content” provided by the customer representatives. A

fundamental tenet of XP states that the simplest design possible is sought.

Coding is done with two programmers working on one computer as a pair. All code written must strictly adhere to agreed upon standards. As code is written, it is tested immediately. Code is often integrated and the entire system rebuilt many times per day for integration testing. Programmers write unit tests, customers write integration (functional) tests. XP provides that anyone can change any code at any time, thus implementing collective ownership in the system. Code is worked until it passes testing 100%. The release cycle of XP is characterized by a simple initial production release, followed by multiple releases on very short (weeks versus months) cycles. Finally, communication is extremely important and valued (Beck 54).

While these modern practices play a role in this author's professional activities, they are inappropriate for application to this paper. This paper is an individual effort, and most of the RAD concepts described above do not scale down well to individual-effort projects. Those that do, such as selecting a RAD development language, are utilized in support of the methodology selected for this paper. The choice of methodology depends on the nature of the customer. In this project, the customer uncertainty regarding the system is high. The customer is not aware of what is possible in a computer automation of the session documentation process, thus the need for guided directions and options is high.

Given the state of the customer, the novelty of the problem domain being entered, the small development team size, and the end goal of this paper (a working model of a documentation system), prototyping is the most appropriate methodology to use. There are two types of prototyping, evolutionary and throwaway, where evolutionary means one basic design is tweaked and molded through iterations of customer review, and throwaway implies that a new prototype could be created after each customer review iteration until an acceptable design is found. For this paper, the more flexible, but more time costly, throwaway prototype methodology will be used.

Requirements gathering is the first step towards system design and problem resolution. As Pressman notes, the prototype paradigm of software engineering begins with requirements gathering (22). Requirements develop the scope of the software project, define what tasks the software must do, and establish means for evaluating success in software development. The iterative approach and use of interactive software models is best suited to ensuring that all necessary requirements are discovered and addressed. Pressman has noted that prototyping is useful if detailed requirements are not available from users (32). Requirements have been categorized for this project in terms of user, corporate, and environmental.

User Requirements

“User” in the context of this paper, refers to a company therapist who will be using the documentation system on a regular basis. To collect requirements for this stakeholder group, a practicing company therapist was interviewed regarding documentation systems and computer software systems.² Open-ended questions were asked in order to discover those aspects of high importance to the user, rather than pre-define the scope of questions and force a user to answer within the bounds of the questions.

From the interview responses, certain common concepts were uncovered. The most commonly mentioned aspect was that a documentation system had to be easy to use. The interviewee stated the system must not be complex or busy – it should be very easy to get to a desired action, and the screen should not be filled with data from top to bottom. “One click” was a phrase often mentioned, and when asked about it, the idea meant was that the system shouldn’t require a lot of effort on the part of the user to make something happen. The user desired to point and click and type as little as possible. The user suggested that a voice recognition system where the therapist would dictate the note text and the system format it, add in necessary static data, and print out the hardcopy would be the ideal solution.

² 4 interviews were conducted with Dr Durrett, a practicing SPS psychotherapist, between May 99 and Sep 00 See bibliography for detail

The display characteristics were important to the user. The user claimed to represent the “average” therapist at the company, and the issues of font size and readability were real concerns. The system must be clearly readable for persons whose sight is not strong. The user requested very large font so as to clearly read the text without eyestrain.

The interviewee complained of fatigue when writing session notes in the current paper form method. The system must not be physically or mentally taxing – it should know the data the form requires, format output to match the current formats, and not require the therapist to think too much about writing the note. Ideally, the therapist would be thinking only of the content of the session when completing a session note form.

The user demanded that any note system deployed must be easy on the body as well. Heavy notebook computers over seven pounds were too heavy to lug around a residential facility all day. The system needed to “fit into a coat pocket”, and ideally would be as light as a cell phone. The user explained that in residential geriatric treatment, the patients are seen wherever they are located, and not often in a private place. The system needed to be completely mobile so as to go where the therapist goes, with no more physical impact on the therapist than a notepad and pen.

From these user stated requirements, there are certain common themes that arise. Ease of use is strongly desired in the system. This concept runs the gamut of operation use to mental tasking to mechanical factors. Where feasible, the choice that represents the least impact/requirement to the user will be the best choice to select first. Also important is the mobility of the system. The system design must consider the hardware characteristics of the necessary components – computers and peripherals can not be too heavy, bulky, or require a multitude of cables, connections, and setup.

Corporate Requirements

Corporate requirements are those procedural/operational requirements that are mandated by the business. This typically includes corporate rules and regulations. In this software system, the established corporation documentation guidelines and procedures must be supported. The documentation system must support the methods of business that the company uses, and comply with the approved corporate documentation guidelines currently in place for the manual note generation system. The system must support the company's proprietary symptom-intervention-progress documentation model (as described in the problem definition section of this paper). The system must also support the company's current guidelines on psychotherapy documentation.

Medicare documentation requirements are of critical importance to the company, in that a great majority of patients seen are funded through Medicare for their psychotherapy (Durrett, 28 May 1999). There are three criteria the company has identified that Medicare uses to establish that psychotherapy is “reasonable and necessary”: 1) Patient has a condition that requires assessment and intervention; 2) A diagnosis ICD-9 code is assigned to the identified condition; 3) Nursing/Social Services notes corroborate the diagnosis, and other intervention efforts were unsuccessful.³ It is required that session documentation meet these criteria in order to assure reimbursement for psychotherapy services rendered. Most important to the session notes documentation system are items 1 and 2.

The company has programs and references in place to ensure session documentation produced by member therapists is professional, defensible, and complete with Medicare’s essential criteria. Company guidelines for documentation are communicated through memos from the company Quality Management director. The following items are specific guidelines that must be present in each session document (Rogers, 1):

- 1) Patient name must appear on every page
- 2) Month/Day/Year must appear on every page
- 3) Every session notes document must be signed with therapist’s professional title

³ “Effective Documentation for Long Term Care”, pg 1

- 4) No scratch-outs, white-outs, erasures, or felt-tip markers
- 5) Error corrections may only be single line through error, labeled "Error", initialed, correction placed nearby
- 6) No blank entries – "N/A", or reason not done, must be entered
- 7) Note content should be accurate, timely, objective, specific, concise, consistent, comprehensive, logical, **LEGIBLE**⁴, clear, descriptive, reflective of treatment
- 8) Be specific – document specific observed behaviors
- 9) Be specific – avoid general characterizations
- 10) Session notes may have to stand alone for reimbursement
- 11) Be objective – document facts

These guidelines are considered minimum standards for session notes created by company therapists. Any documentation software must meet these guidelines to a degree acceptable by the company. Subjective guidelines, specifically 7-11, are more difficult to define and categorize than the objective guidelines 1-6. The company uses samples to illustrate sentences that meet the company guidelines. The company document, "Effective Documentation for Long Term Care", discusses how a session note is structured to meet the Medicare requirements, and each of the three criteria has example sentences that the company cites as meeting Medicare criteria. In addition, the example sentences conform to the company's documentation guidelines listed above. Following are samples of the example sentences provided in the company memo (Rogers, 2-5):

Criteria: Medical Necessity

- "Patient was crying and moaning at beginning of session."
- "Resident unwilling to leave room to participate in activities today."

⁴ The word "legible" is all-caps and bolded in the company document for emphasis

- “Nursing and social services report that Mrs. Conte⁵ was sarcastic and verbally abusive to staff on 5 occasions during last week.”

Criteria: Therapeutic Intervention/Level and Complexity

- “Therapist taught Relaxation Techniques to resident and practiced deep breathing with resident.”
- “Reviewed with patient how he can use Cognitive Reframing to quell his anxiety.”
- “Applied Relational Techniques in discussion with patient about her difficulty controlling her impulses and behaving inappropriately toward male residents.”

Criteria: Progress Assessment

- “Resident attended three out-of-room recreational activities this week.”
- “Resident was able to make a positive statement about the care she is receiving during today’s session.”
- “Resident’s anxiety level was rated a 3 on the 5-point scale by PM Shift Supervisor.”

Environmental Requirements

This set of requirements deals with the physical and operational aspects of the locations/structures in which the software system will be used. These requirements are sourced from the user group, as they are most familiar with the conditions in which they render services. Three main environmental factors having impact on the proposed system are the nursing facility itself, the facility staff, and the patients.

Long-term care facilities are institutions housing numerous elderly

⁵ Mrs. Conte is a fictitious name used only as an example

individuals. These residents are housed in semi-private rooms, and typically have few personal possessions. The facility provides meals in a communal dining room, and residents are afforded access to planned activities on a regular basis. For the practicing psychotherapist, this has constraining implications. Psychotherapy is best conducted in comfortable settings and explicit privacy. Long-term care facilities offer very little in the way of privacy. Meetings with resident patients often occur wherever the patient is located at session time. Many patients are not ambulatory, thus they may be placed in a common room, in a hallway with a view, or in some other public place when the psychotherapist arrives to conduct therapy. If available, it is possible that the therapist could move the patient to a private room temporarily, but the existence of such space is rare. It is common practice to conduct therapy in semi-public surroundings (other patients within 20 feet). Semi-public venues mean that ambient noise will be greater than normal. A table or other work surface is not always within reach – even a chair for the therapist is not guaranteed. Residents can be easily disturbed and therapy disrupted by adjacent activities. It is important that a documentation system be portable and flexible, such that it can be used in a variety of settings and easily moved from one place to another (Durrett, 29 Mar 2000).

Facility staff is an important factor to consider. Staff members can make issues of access and progress easy or hard. The company notes in the therapist's handbook that cultivating a positive relationship with facility staff and integrating one's self into facility routines is important in maintaining a professional presence and increasing the chance for additional patient referrals (Senior 1998b, 4-2). It can be postulated that the documentation system must not place a burden on the staff or the facility resources. The system should be self sufficient, even to the point that self-powered is a preferable, but not mandatory option. Ideally, the only impact the documentation system would have on staff is improved efficiency in their jobs due to the presence of the improved session note document left in the patient's chart, and the absence of problems that illegible chart documents cause.

Patients in long-term care facilities seen by the company therapists are elderly patients. As psychotherapy by definition requires a verbal interaction between patient and therapist, patient capabilities may already be taxed by the session experience. Therapist interviews tell that patients are easily distracted if the therapist does not give the patient his/her full attention. Therapists may be the only persons a patient interacts with for a number of days at a time. Patients have been seen to exhibit an expectation that they have the therapist's full attention. For most patients, technology is not native to their experience, and the

use of a device such as a cell phone, palm pc, notebook pc, or other fairly substantial object may serve only to distract the patient, or give the impression that the therapist is ignoring the patient in favor of the device. The company therapist interviewed continued to state that the ideal device would not require a user to look at it, or punch a lot of keys. It would be "...best if the therapist could generate the entire session notes from 4-5 clicks on a pad..." (Durrett, 29 Mar 2000).

Patient sensitivity is a complicating aspect in that in satisfying the user requirements, i.e. paying complete attention to the patient, the system becomes less able to realistically perform its job. In the case of patient requirements, the degree to which the patient can be accommodated is variable, yet slanted towards the negative side in that there will be some minimum of interaction the system requires. The closer the system is to the patient, the fewer the interactions the therapist can make with the system. This will affect the end design by either limiting the capability of the documentation system, thereby reducing the inputs required, or moving the system out from the patient, meaning that the documentation system is not real-time, but accessed soon after session completion. In moving the system out from the patient, the system then begins to intrude on the therapist's requirements of portability and lightweight and the facilities limitations in accommodating the therapist's needs for power, a place to

sit and work with the system, print out notes, and the like.

These base requirements provide enough information to begin evaluating existing software solutions. An existing piece of software is like having a set of answers before knowing the questions. Software has a certain set of capabilities, and in the evaluation process, the set of existing capabilities will be compared to the set of required capabilities. Selecting an existing software solution is a balance between the degree to which the existing software fails to meet the existing requirements, and the cost difference between buying existing software or paying for custom code. In the next chapter, two of the best-suited candidates as solutions are evaluated in light of the requirements analysis.

III. EXISTING SOLUTION ALTERNATIVES

Existing software solutions must be considered prior to committing resources to designing new software for time and fiduciary considerations. Integration of existing software is very likely to require less time to complete than creating new code bases of a stable and reliable nature. Custom software development is an expensive proposition, not only in time, but also in money. Often existing software solutions are less expensive per unit than custom development (in this case, per unit is per therapist). The field of medical software is a growing one, and two leading programs were obtained for evaluation. They represent software that is within a financial range likely to be acceptable to the company, and that does not require hardware beyond what is nominally provided in a personal computer.⁶

Therascribe 4.0

Therascribe is a commercially produced psychotherapy software system designed around the treatment planning methodologies developed by the software's designer, psychologist Arthur Jongsma, Ph.D. It is a Windows-based software system that provides patient management, assessment, diagnosis,

⁶ \$500/yr was used as a base acceptable cost. The company would not offer any guidance on acceptable prices.

treatment planning, reporting, and discharge record keeping for psychologists. A free demonstration version of the software was available via Internet download as of October 2001.⁷

Therascribe 4.0 is the software implementation of Dr. Jongsma's ongoing written efforts regarding psychotherapy treatment planning. He is the author of a series of treatment planning manuals that offer practicing psychotherapists well-organized and detailed treatment plan options for the majority of psychotherapy cases seen. Psychologists often use manuals of this kind, as they provide structured treatment options and detailed therapy options with little work required on the part of the psychologist (Durrett, 28 May 1999). The software version of the treatment planning methodology adds the elements of electronic data management and of automated document generation to the detailed treatment planning offered in the texts.

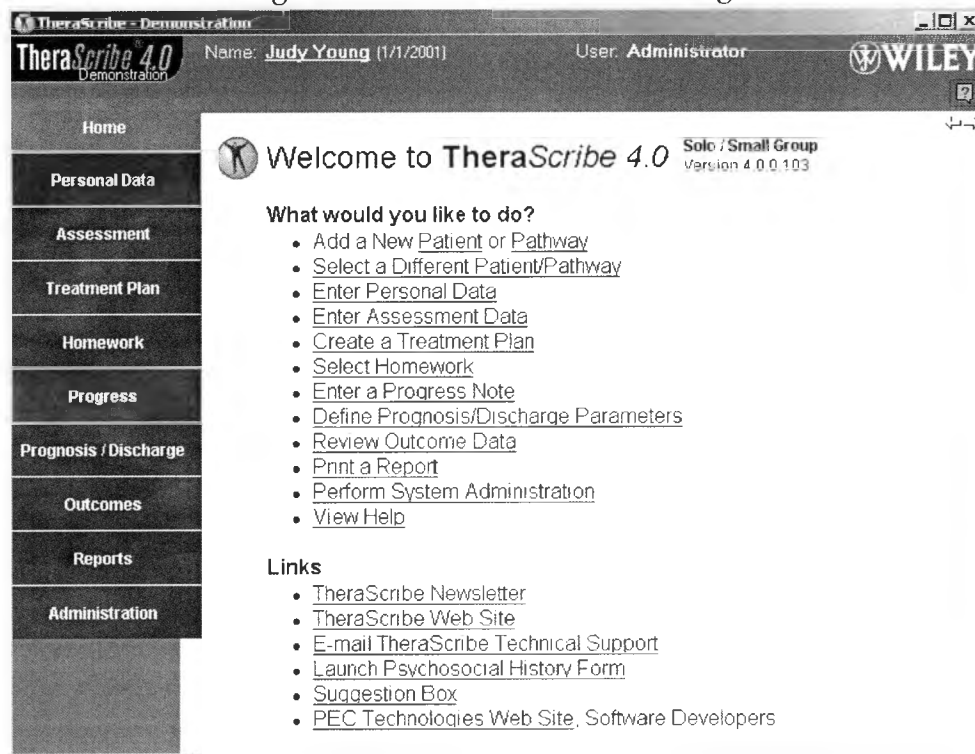
Functional Overview

Therascribe provides a user interface-based on internet-browser navigation and traditional file tab presentation of data. The main menu, or home page, of the application appears much like a standard framed web site, where there are point and click navigation links in the left most column, reference data displayed along the top, and specific action items in the primary interface pane

⁷ <http://therascribe.wiley.com/>

that dominates the application.

Figure 1: Therascribe 4.0 Main Page



The links in the white pane (seen under “What would you like to do?”) provide access points to specific data elements contained in the sections identified in the left-side navigation column. Entries beneath the “Links” label are internet-based support resources for the software application.

Data are entered in a sequential, determined manner. That is, information entered in initial stages of the application (Personal data, Assessment, Treatment Plan) has direct effect on the scope and nature of data entry occurring in the latter stages (Progress, Prognosis/Discharge, Outcomes). For each of the

application sequence steps, there are numerous sub categories of data, displayed as folder tabs, available to the user.

Figure 2: Demographics Sub Category, Personal Data Entry

TheraScribe - Demonstration
TheraScribe 4.0
 Name: **Judy Young** (1/1/2001) User: **Administrator**
Personal Data
 Demographics | Provider | Insurance | Clinical Notes | Attachments | Custom Fields

Home | Personal Data | Assessment | Treatment Plan | Homework | Progress | Prognosis / Discharge | Outcomes | Reports | Administration

Last Name: Young First Name: Judy MI:
 ID Number: 1 SSN: 444-44-4444 Birth Date: 4/9/1949 Age: 52
 Address: 1 Main Street
 City: Capitol City State: TX Zip: 78727
 Home Phone: 911-555-1212
 Military Rank:
 Employer:
 Work Phone:
 Referred by: HJD
 Marital Status: Widowed
 Race: Other
 Gender: Female
 Setting: Inpatient
 Department:
 Treatment Start: 1/1/2001
 Last Review: 10/20/2001
 Treatment End: 12/20/2001
 Primary Care Physician: Joe Smith
 Psychiatrist: Jeff Thompson
 This is an Active Client ☒
 This Client was Previously Treated ☐
 Other Treatment Episodes

Start	End	Primary Problem

 New Episode

The Assessment and Treatment Plan sections of the application implement Dr. Jongsma's planning methodology. The data elements and selection options come from the planning texts he has written (Jongsma, 1999). The Progress section contains the interface to create psychotherapy session notes. This functionality is most important in determining the suitability of Therascribe 4.0 as a solution to the documentation problem identified in this paper. Figure 3 shows how the note interface appears to the user. The method of work on this

screen is:

Figure 3: Progress Notes Section Interface

TheraScribe - Demonstration Name: **Judy Young** (1/1/2001) User: **Administrator** **WILEY**

Progress

Session Details

Date	Start	End	Length	Modality	Progress Rating	Provider
10/20/2001	10:00 AM	11:00 AM	60 min	Individual Psychothe	Some Progress	

Session # Authorized Sessions Remaining

Progress Notes Planner Objective Rating Narrative Progress Note

Problems focused on

Depression

Presentations for Depression

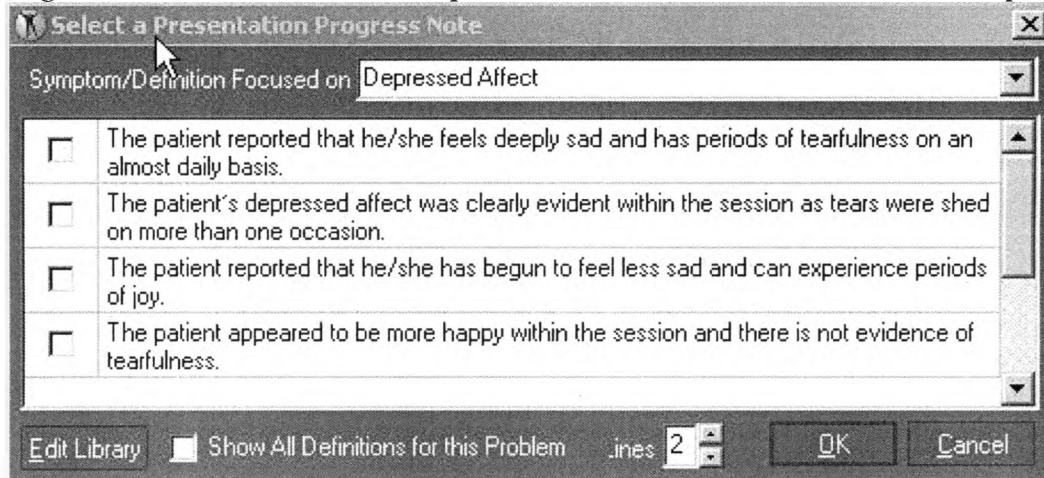
The patient appeared to be more happy within the session and there is not evidence of tearfulness.

Interventions for Depression

The patient was asked to describe his experience of depression for the signs and symptoms that are present in his daily living.

1. In the Session Details box, select a session to work on, or click “Add” to create a new session.
2. On the Progress Notes Planner tab, click the triangle in next to the Problem field to view a list of all patient problems previously entered in the assessment section. Select a problem.
3. With a problem selected, the “Presentations for Depression” list is populated with a list of static, generic sentences to describe how the patient demonstrates the problem. Access the list by clicking the triangle in the upper left box corner. Selected sentences will appear in text to the right of the triangle box.
4. “Interventions for Depression” list is also populated upon selection of a problem. It works in an identical fashion to the presentations section.
5. Multiple sentence selections are possible – in this example, there are a total of four possible presentation options for the depression problem, as shown in figure 4 below.

Figure 4: "Presentations for Depression" Sentence Selection List Example



Objective Ratings are generic overall descriptors of patient progress in one of the pre-determined patient objectives (these objectives are determined in the Treatment Planning section). There are five generic degrees of progress, and a blank for no information. Objectives are set at two levels, overall, or problem specific. If a specific problem is selected from the problem list box, all pre-selected objectives for that problem are displayed in the box below. Each objective can then be assigned a rating. In figure 5, all recorded problems are displayed, and the rating menu for the first problem is shown.

Figure 5: Objective Rating Assignments

Progress Notes Planner | Objective Rating | Narrative Progress Note

Problem: (All) Lines: 2

Objective	Rating
Report to appropriate professional the effectiveness of medications and any side effects.	Significant Regression Completed Regression Some Progress No Change
Take prescribed medications responsibly at times ordered by physician.	
Verbally identify, if possible, the source of depressed mood.	
Describe the signs and symptoms of depression that are experienced.	

The last section is for narrative sentences. This section covers all subjects/issues/topics/events not addressed in the progress or objective tabs. This interface is simply a textbox to accept user keyboard input.

Figure 6: Narrative Progress Note

Progress Notes Planner | Objective Rating | Narrative Progress Note

Text area with cursor (I) and Edit Text button.

The software captures all entry as live data, and changes to the data are permanent – any errors must be manually undone – no rollback or cancellation of data entry is possible.

Suitability Analysis

Therascribe was evaluated against the identified goals for a software documentation system: 1) Ease of use; 2) Implement company session documentation scheme; 3) Produce current company forms on hard-copy; 4) Comply with company documentation guidelines; 5) Avoid repetition in computer generated sentences; 6) Reduce therapist documentation task workload. Question one involved completely subjective analysis, where the other five goals could make use of some quantifiable elements.

In addition to checking the software's performance against goals, the level of effort required to integrate the software into the company's operational practices must be considered. The issue of budget and cost has no direct bearing on how a software system meets requirements goals, but it does act as an immutable boundary that any selection must reside within. Budgetary matters are highly confidential matters for any company. Thus, for the purpose of evaluating existing software solutions in this paper, the cost question has been mitigated by accepting both systems as "within limits."⁸ Beside fiduciary concerns, the amount of work required to successfully place the software system into operation must be considered as well. The degree to which companies must modify or change their business practices is finite. Many costs are associated

⁸ Note: In a real evaluation process, the budget question would likely be the final determiner from among a screened group of acceptable options.

with retooling and retraining a workforce, especially one where business procedures have been in place for a long time. Changing business practice to compliment the business software is not a sound practice – thus the degree of flexibility the software has in conforming to existing business models is an important factor in the adoption decision. For purposes of this prototype paper, the degree of expected change will be considered. The evaluation of that degree of change will not be done, as only the adopting company itself can make that determination.

Goal One: To test Therascribe’s performance in ease of use, a professional software engineer and a practicing psychologist familiar with the documentation project conducted usability testing. Testing was patterned to resemble 10 weeks of session notes. Testers were provided a list of symptoms, interventions, and progressions for 10 sessions. Therascribe’s data for the patient at the outset of testing was limited to only minimal demographic data. Problem and treatment information required by Therascribe would have to be entered as part of the session note generation process. Analysis of qualitative responses from the testers provided metrics for evaluating usability test results.⁹

Overall, the tester’s results indicated Therascribe 4.0 was an inappropriate choice for a software documentation system. Comments regarding ease of use

⁹ Timing results are discussed in requirement six summary in this section

included “too complicated”, “overkill”, “too many fields, too many screens”, and “frustrating when you miss putting a data value in somewhere”. When asked about what training would be necessary for using the software confidently, testers replied that classroom, hands-on instruction was a good idea, and the estimated duration of training ranged in days from two to five (Durrett, 18 Sep 2000).

Goal Two: Therascribe does not follow the symptom-intervention-progress documentation scheme in use by the company. Each of the three areas is represented in the Therascribe session, but in capacities that do not match the company emphasis in each area. The user is required to manually add the patient’s facility name.

Goal Three: Therascribe’s reporting facilities are setup to report in a format that is not compatible with the corporate documentation model. Below, figure 7 is an example of Therascribe reporting:

Figure 7: Progress Notes Section of Clinical Report

Progress Notes			
Session 1	Date: 10/20/2001	Time: 10:00 AM to 11:00 AM (60 min)	
	Modality: Individual Psychotherapy	CPT Code:	Progress Rating: Some Progress
Problem Addressed: Depression			
Patient Presentation (Signs and Symptoms)		Interventions Implemented	
<ul style="list-style-type: none"> • The patient appeared to be more happy within the session and there is not evidence of tearfulness. • The patient reported that he has begun to feel less sad and can experience periods of joy. 		<ul style="list-style-type: none"> • The patient was asked to describe his experience of depression for the signs and symptoms that are present in his daily living. 	
Provider Signature: _____		Date _____	

There are many discrepancies between the existing corporate session note form and this form: The session note form's fields for Facility and Modality are not similarly represented here. There is also no provision for the currently established, static treatment goal reference – it is replaced by the much more detailed and involved treatment planning process that Dr. Jongsma advocates in his books. Progress is communicated through only the phrase displayed next to the "Progress:" label. The company documentation scheme requires that progress be subjectively and quantitatively stated, along with a measure of progress towards immediate and long-term goals. The corporate session documenting model is not accommodated in this product, meaning that the model would require modification, or abandonment for adoption of a model fitting the software's capabilities.

Goal Four: Therascribe does not comply with company documentation guidelines in the area of progress. The progress notations in Therascribe are insufficient. Progress notes would have to be added to the text section of the note, resulting in Progress values displayed in either the "Interventions" or "Patient Presentations" section.

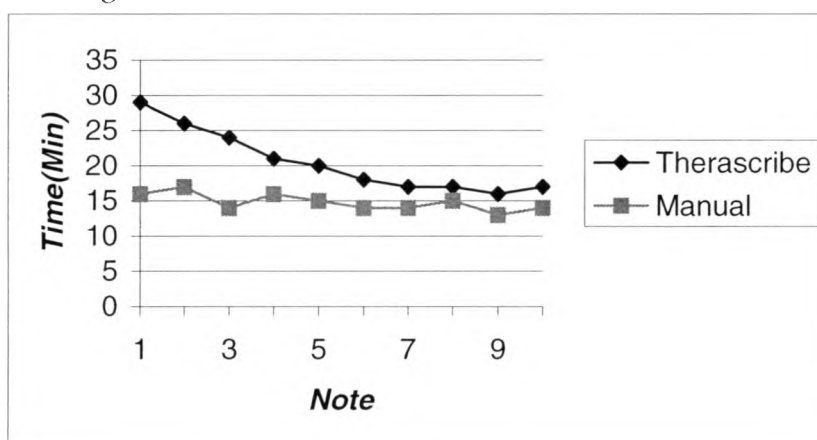
Goal Five: The software constrains users to create documents in a set fashion that does not correlate to the company's proprietary documentation model. As seen in figure 4, all patients with depression identified as one of their

problems would have a minimal 25% chance of having the exact, verbatim sentence describing their condition as any other patient under the therapist's care. If more than one sentence is selected, the chances for duplication increase. The psychologist tester reported that the use of totally static, pre-written sentences would be noticeable to anyone reviewing a patient's file – it would be likely that all of the corporation's depressed patients may have the same comment regarding the depression in their notes. This would not comply with the corporation documentation guideline that states that sentences should be variable in nature so as to not sound re-used or artificial. The company position on this kind of automated duplication exists to curtail the impression that patient notes, when generated by software, may not be based on any actual interaction between the patient and therapist. The company concern is that file auditors might deem it improper that notes for one patient copy the notes of another patient's too closely. Company guidelines require that a therapist provide the professional services – a "copycat" note creates suspicion that services were fully rendered in each documented session (Durrett, 10 Oct 1999).

Goal Six: The testers did not note any improvement in documenting duration – test timings to produce a session note at the end of the testing cycle were still an average of about 12% higher than the average for manual documenting. Figure 8 charts the raw timing values. Once the software system

was more familiar, testers were able to produce Therascribe session notes more than 30% faster than at the start of the test. The average difference of the timings over the last three notes generated was about 26%. It is possible that additional experience with Therascribe will result in additional small reductions in note generation timings. However, this software does not appear to address the need to reduce the documentation activity workload for the practicing therapist.

Figure 8: Manual / Therascribe Time Test Results



Quicdoc 3.7

Quicdoc is a commercially developed software system from DocuTrac.¹⁰

An evaluation version of the software was available at the company website as of Fall, 2001. Quicdoc is notably different from Therascribe in organization and presentation of data. From a program architecture perspective, Quicdoc is a traditional Windows client application. Browser technology, as seen in

¹⁰ DocuTrac, Inc. 20140 Scholar Dr, Ste 218, Hagerstown, MD 21742 301-766-9397

www.quicdoc.com

Therascribe, is not present in Quicdoc. Menus and buttons drive all user actions.

Quicdoc also contains a security piece that is much more integrated than Therascribe's. Quicdoc also makes use of proprietary scheduling and word processing functionality.

Functional Analysis

Quicdoc begins with user authentication. Quicdoc security was much more developed than what was seen in Therascribe. Once a username and password recognized by the system are entered, the software application starts.

Figure 9: Quicdoc Main Screen

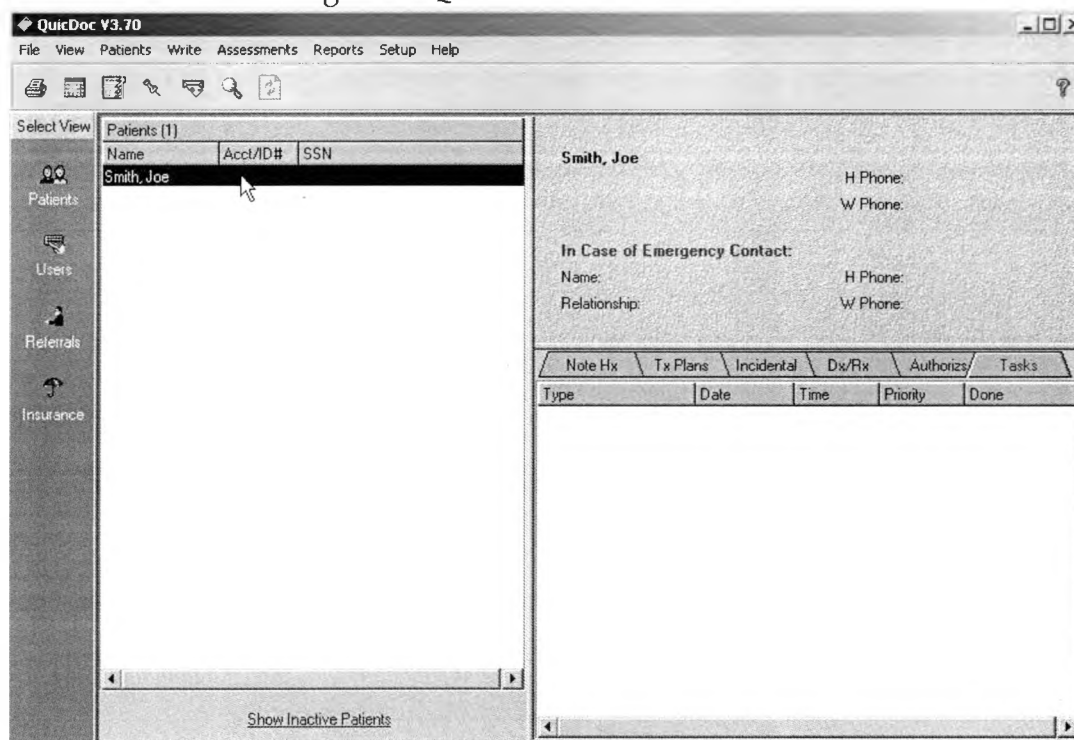


Figure 9 shows the patient view – other available views are accessible in the left

most column. Patient data are far less complex than in Therascribe. Patient data are edited via a four-tab dialog box. Only basic demographic data are maintained, though an optional screen permits user-specified additional text fields. A patient can be entered and ready for use with only three fields populated: Last Name, First Name, and Provider.

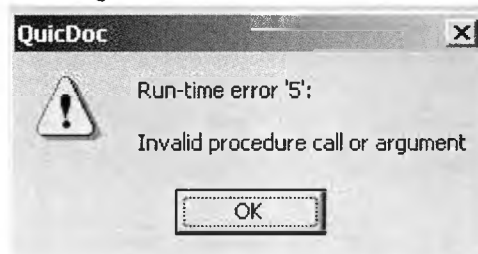
Figure 10: Quicdoc Patient Data Screen

The screenshot shows a window titled "Patient Information" with four tabs: "Personal", "Address", "Insurance/Referral", and "Emergency". The "Insurance/Referral" tab is selected, indicated by a mouse cursor. Below the tabs, there is a section with an umbrella icon and the text "Enter Insurance and Referral Information here". This section contains several input fields: "Primary Insurance" (a dropdown menu with "Medicare" selected), "Secondary Insurance" (an empty dropdown menu), "Referral Source" (an empty dropdown menu), and "Referral Type" (an empty dropdown menu). Below these are two rows of text input fields: "Primary Care M.D." and "Other Therapist", each followed by a "Phone" field with a format of "() - ". At the bottom left, there is a link labeled "Additional Information". At the bottom right, there are "Ok" and "Cancel" buttons.

It is noteworthy that in this patient data screen a run-time error was encountered that terminated the application. In the patient data screen, there are drop down list boxes that allow users to input their own data, so as to allow the list to grow with custom entries. In the insurance box, "Medicare" was entered,

as the list was blank. The cursor was then moved to the emergency tab and immediately the error message shown in figure 11 appeared.

Figure 11: Quicdoc Run-Time Error Message



After clicking OK to clear the error message, the system displayed a dialog box that asked if the user wanted to add the entry to the list. “Yes” or “No” were the two options available. After clicking yes, the program crashed. In a subsequent test, clicking either yes or no resulted in no data saved and abrupt program termination. In this author’s professional experience, this type of error is the result of programmer error in coding. This error is noteworthy in that it indicates inadequate testing and possible defective coding practices. This type of error calls into question the reliability of the software.

Once a patient is entered, a progress note (which is the Quicdoc equivalent of a session note) may be entered by going to menu Documentation -> New -> Progress Note. This brings up the main progress note interface screen. On this screen, users go through four steps to complete a progress note: session information, content-process, interventions, plan. Session information relates to

descriptive aspects of the session itself. Users are provided drop down lists for most session information fields. Figure 12 illustrates the session information screen and the Program data options list.

Figure 12: Quicdoc Progress Note - Session Information Screen

The time controls work in an unexpected manner. To enter a time, users click in the box and type out a number. The box auto-formats the data to an “hh:mm AMPM” type. If “12:00” is typed in, it defaults to “12:00 PM”, or noon. When “1:00” is entered in the End time box, it auto-defaults to “1:00 AM”, which then causes an error message box to popup saying the end time is prior to the start time. On dismissing the message box, the end time is deleted and left blank.

It required several minutes to come up with a way to trick the system into accepting the time. It was interesting to find that the length field, which has up and down arrows, does not affect the end time. Entering “60” in the length box had no affect in setting the end time box. This behavior was another indicator of problematic coding and a potentially unreliable product.

After the session information is set, the user moves on the content-process section. This section deals with general aspects of the patient’s mental health. The content tab deals with data surrounding the reason for having the session, akin to a primary symptom. Users pick a session theme from a group of checkbox topics. Checkboxes allow for multiple themes to be selected. Except for session theme, all data in this section is selected from drop down lists. Session theme is a very generalized categorization of the session. It has options that are not appropriate for elderly, long-term residential facility care patients, showing that Quicdoc was developed as a non-specific, general use documentation system. There are text boxes available for users to type in text to describe themes not present in the checkbox collection, note changes in stressors, and any general session content related information. The process tab is a group of drop down lists that permit single selection of entries that describe the events, techniques, and outcomes of the session. The process pick lists are user configurable for customization. Figure 13 illustrates the screen and a sample drop down list.

“Other” is a text entry field where users may enter specific custom text regarding any issue addressed on the screen. “Interventions” is the next step in progress note generation.

Figure 13: Quicdoc Context-Process Screen

The interventions screen is used to describe specific actions taken by the providing therapist, what response/reaction was observed from the patient, and any interim behavioral related tasks assigned to the patient (Quicdoc euphemistically refers to these assignments as “homework”). An intervention is the first selection made on the screen. Quicdoc provides 11 pre-scripted sentences to describe the entire range of therapeutic interventions a therapist may have taken. Therapists may add new custom sentences to the list as desired.

Multiple interventions may be selected, and all selected interventions are displayed in a list box directly below the pick list. This pick list selection is the only method of entering intervention statements. Below the list of selected interventions is a text box labeled “Response to Interventions”. In this small box, the user may type sentences to describe the patient’s response to the interventions. The screen is completed by two additional text boxes for use in documenting “homework” – homework results, and new homework assigned. Figure 14 illustrates the interventions screen with 2 selected interventions and sample intervention response text.

Figure 14. Quicdoc Progress Note - Interventions Screen

Progress Note - [Smith, Joe]

File Help

10/28/2001

Progress Note

Session Information

Content-Process

Interventions

Plan

SOAP Note

Optional Sections

Interventions Used

Enter an intervention or select from drop-down list and Click Ok to insert.

Ok

1 assist patient in identifying and labeling emotions

2 encourage formulation of plan of action

Delete

Response to Interventions

Response to 1. Response to 2

Homework

Results of previous homework, if any

New Homework Assignment

The final screen in the progress note tool is the Plan screen. The screen is set to display treatment plan and risk management data. While Quicdoc uses a file tab format to display subsets of patient data on one screen, in this case the application uses buttons to control the screen contents, much like channel tuning.

Treatment plan covers treatment modalities and patient medication regimen. There are checkboxes to indicate a yes/no answer to a printed question that acts as a textbox label. The first textbox label asks if the treatment modality should be continued. The second textbox label asks if current medications should be continued. Consultancy requests are entered in a text box, and a fourth text box is labeled as discharge planning.

Risk management offers users radio buttons and checkboxes to indicate degree and state for suicide and physical violence. Both topics share the same radio button options for degree, and the same checkbox options for state. A large text box covering the bottom half of the screen is labeled as other specified risks. Figures 15 and 16 illustrate the plan and risk management screens. Quicdoc does not require the user follow a specific sequence in generating a progress note. At any time, the progress note can be generated based on current data selections and entries. Fields with no data are left blank in the preformatted report document generated by Quicdoc.

Figure 15. Quicdoc Progress Note – Plan Screen

Treatment Plan

1. Continue current treatment modalities? If no, state rationale and address changes in Treatment Plan under Optional Sections. ☒ Yes ☐ No

Treatment progress could go here?

2. Continue current medications? If no, state rationale and address changes in Medications under Optional Sections. ☒ Yes ☐ No ☐ NA

Medication information here?

3. Referrals or Consultations needed? Specify below and state reason.

4. Discharge Planning

Figure 16. Quicdoc Progress Note – Risk Management

Risk Factors

Suicide

☒ None ☐ Ideation

☐ Mild ☐ Plan

☐ Moderate ☐ Means

☐ High ☐ Attempt

Physical Violence

☐ None ☐ Ideation

☐ Mild ☐ Plan

☐ Moderate ☐ Means

☐ High ☐ Attempt

Specify Risk Management Plan and other risks (if any)

Quicdoc progress notes are generated via a proprietary word processing component called Quicword. Quicword creates document files in the Microsoft

Rich Text File (*.RTF) format. Note documents are generated via two methods: main screen menu item File->Print->Notes and progress note screen menu item File->Report. If using the main menu option, users are presented a dialog box that permits a range of notes to be selected. On clicking print, one generic note format is applied to each selected note, which is then displayed in an instance of Quicword.¹¹ Users then must click print in Quicword to generate the paper document. If using the menu option on the progress note screen, the user is shown a full-featured dialog box offering report format customization. Button “Generate” starts an instance of Quicword with the formatted report ready to print. Formatting of the report consists of selecting various subsections of data for inclusion. Users select which data subsections will appear in the report and in what vertical sequence the sections are oriented. Report and page headers may be customized in a limited manner. Report subsection customizations can be saved for re-use as “sequences”. Figure 17 illustrates the progress note report screen.

¹¹ Quicword is a separate application. It opens in a separate window outside Quicdoc and is a separate process in MS Windows Task Manager.

Figure 17. Quicdoc Progress Note Report Screen

Note: The Generate button is initially disabled. At least 1 note section topic must be placed in the Sequence list box via the “Add>” button in order to create a report.

Suitability Analysis

As with Therascribe, Quicdoc was analyzed as to performance against the six general requirements (goals) for a documentation system. Quicdoc was tested and assessed by the same testers. Testing was patterned to resemble 10 weeks of session notes. Testers were provided a list of symptoms, interventions, and progressions for 10 sessions. Quicdoc’s data for the patient at the outset of testing was limited to only minimal demographic data. Problem and treatment information required by Quicdoc would have to be entered as part of the session note generation process. Analysis of qualitative responses from the testers

provided metrics for evaluating usability test results¹²

Goal One: Quicdoc rated high in ease of use, compared to Therascribe.

Quicdoc required a logon authentication each time the application was started, but the reduced number of tabs, buttons, and options led to comments of “fast”, “quick”, “easy to figure out” (Durrett, 18 Sep 2000). The application had far less layers of data than Therascribe, and this reduced number of layers led to much faster acquisition of system processes. The input process in Quicdoc is far less structured than in Therascribe. Quicdoc was noted as much more flexible than Therascribe.

Goal Two: Quicdoc is capable of providing a facsimile of the symptom-intervention-progress (SIP) psychotherapy service model the company operates under. The system does not facilitate this type of reporting, but it does not prohibit it. Therascribe’s internal processes prevent the creation of SIP documents. Quicdoc could produce a document that meets the requirements of SIP, but it would basically be a progress report with symptoms and possibly interventions manually entered into the report inside the Quicword tool. The possibility of technical modifications to Quicdoc to accommodate SIP is not a likely prospect, as that level of exchange with DocuTrac could place the company at risk of losing the proprietary advantages they currently enjoy. Figure 18

¹² The content of the Quicdoc test sessions was different from the content of the Therascribe test sessions

illustrates a Quicdoc progress note generated via the “File->Report” menu option.

Figure 18. Quicdoc Progress Note – Quicword Report Preview

Progress Note

Date of Session 10/28/2001
 Session # 2
 Service (CPT) 90808
 Session Length 61 minutes
 Session Time 12:00 pm - 1:01 pm
 Type of Visit Scheduled
 Program residential treatment

Patient Smith, Joe

Session Content

Other themes Roommate hatred
 Session Content Resident still hates their roommate

Session Characteristics

Motivation fair
 Resistance moderate
 Cognitive Focus shifts from one topic to another
 Cognitive Flexibility difficulty seeing alternative perspectives
 Affect Expression guarded in display of affect
 Activity Level waits for therapist to initiate discussion
 Use of Session avoids dealing with conflictual issues
 Treatment Compliance minimal degree of compliance with treatment

Interventions

1) encourage formulation of plan of action

Response to Interventions None

Homework

Homework Assigned Do the plan of action

Provider Name Test User

 Provider Signature Date

Figure 18 also illustrates that Quicdoc reports must be manually altered to include information about the facility where the patient resides.¹³ Facility information is required data in the current paper session document form, making manual entry of facility name required minimum.

¹³ Therascribe also fails to include default Facility information in reports

Goal Three: As with Therascribe, this requirement is met functionally. The system can produce hardcopy report documents to any printer to which the host computer can communicate. The output, however, is limited to the capability of Quicword. Computer-generated forms (resembling the current company forms) would not be exact matches, though the manual edit feature of Quicword allows for all required data elements to appear on a session note form.

Goal Four: As Quicdoc does not have the rich context capability of Therascribe, it meets goal four only in the sense that users are free to type anything onto a report from within Quicword, and the report file can be saved to disk. Data that is manually entered into a report does not become part of the database record for the report – therefore, the potential exists that progress notes and session data in the application would not match the printed session notes report left behind in the patient’s medical chart.

Goal Five: Quicdoc uses the same technique for writing sentences in a session note – users choose from a list of complete sentences. In following this strategy, the candidate sentences can be complex, elaborate, and detail-laden. However, their static nature means the sentence cannot adapt to the context in which it is used. Using sentences in a variety of contexts leads to misunderstanding and ambiguity in interpretation of meaning. It could lead to false assumptions regarding a patient’s status, which could have disastrous

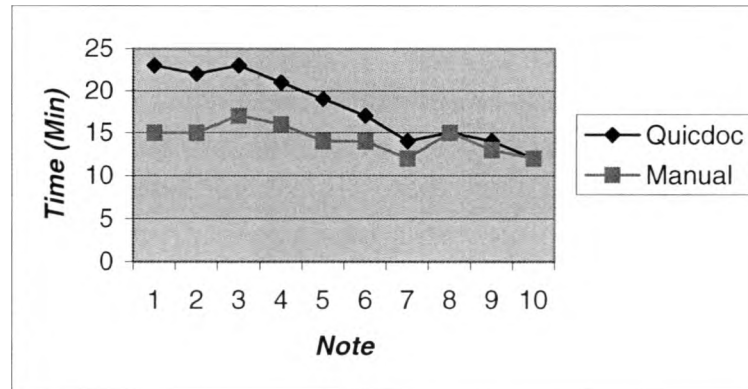
results. Quicdoc provides more freedom to include user-written text in progress notes, but as with Therascribe, the system's automation sentences are static and immutable.

Goal Six: Note creation tests corroborated tester's positive usability comments in that the degree of separation between handwritten and system generated notes is smaller for Quicdoc. Two of the last three Quicdoc timings were short enough to record the same duration as manually written notes (after rounding). After the first five notes were generated, the average excess time for Quicdoc generated notes ranged from a high of 21% to 0% (recorded twice). The average excess time for the final five notes was 9%, and it is expected that this amount could be reduced as the therapist gains expertise in the system and builds libraries of pick list sentence entries that the therapist can reuse often. Should the therapist use pick list sentences frequently and curtail manual text entries, note generation via Quicdoc could likely become faster than the manual method currently used by the company. Figure 4.19 illustrates the timing results.

Analysis Conclusions

Therascribe is unsuited for use as a documentation system for the company. Its complexity and low degree of "user-friendliness" disqualify it from consideration, even before other deficiencies are considered (such as lack of support for the company SIP model, lack of required data elements in reports –

Figure 19. Manual / Quicdoc Time Test Results



facility, and static sentences re-used without consideration of context). Quicdoc holds far more promise as a documentation system, but it also is deficient in areas of report data elements and in static sentence reuse in varied contexts. Neither system truly addresses the problem of reducing therapist workload, though there is a potential for a light reduction in workload requirements for Quicdoc, after a period of use by a therapist. Either option would also require substantial work in order to replicate the current format of the session document form. In consideration of these issues, a custom software system built to the company's requirements is viable, preferable resolution.

IV. DATA DESIGN

The first step in designing a database to support a software application is to identify the data elements that must be stored. The set of data elements that result from the identification process, along with the data elements required to support the database structure, constitute the scope of the database. Once the scope of the database is established, relationships between the data elements can be modeled in the database design. This relationship modeling will help identify any support entities or structures the database requires in order to meet its specification requirements. Modeling in a relational database is commonly done via entity relation diagrams (ERD) once pertinent attributes/elements/fields have been determined. The database to support this design will be created to third normal form, a common benchmark for reliable databases.

Data Identification

Data required for the application can partially be determined from reviewing all pertinent corporate forms used in the current process, and reviewing the process documentation to find data elements that the current process either omits or handles improperly. User requirements and specifications will identify data elements that must also be included in the database design. Of

the SPS corporate forms, the most critical form to analyze is the current session notes form. A catalog of the current data elements on the form is shown in the table below:

Table 1. Session Notes Form Data Element List

Form Data Element	Form Data Format	Description
Corporate header	Preprinted Text	This is the corporate name, full address, and contact phone numbers
Form Title	Preprinted Text	The name of the form
Resident name	Manual text entry	Patient's first and last name
Acc't	Manual text entry	The SPS account number assigned to the patient.
Facility	Manual text entry	Intended to be Facility name and SPS ID number
Diagnosis	Manual text entry	Intended to be ICD-9 code and textual description of patient diagnosis
Provider	Manual text entry	Full name and title of therapist conducting session
License	Manual text entry	Therapist's state licensure ID
Treatment Goal	Preprinted text	Ten numbered goals are printed out – the therapist circles a number to indicate a goal selection
Current behavioral objective	Manual text entry	Intended to be the shorter term, specific objective in respect to patient behavior
Date of service	Manual text entry	Spaces for month/day/year entry. Form is meant to hold two sessions – two dates fields exist.
Treatment modality	Preprinted text	Four modality options are allowed. A box next to the modality is checked to indicate choice. Two modality lists exist – one for each session.
Current status	Manual text entry	The form provides three lines for sentences regarding patient status and symptoms.
Provider Interventions	Manual text entry	Five lines are provided for recording intervention actions. Two sections exist, one per session.
Assess progress towards goal	Manual text entry	Four lines are provided to show patient progress. Two sections exist, one per session.
Signature	Manual text entry	A signature line is provided after each progress session to indicate session note is official and complete.

These are the data elements currently being collected at every session. Besides the session form, two other forms are prominent in moving a patient through a cycle of psychotherapy sessions: the Referral For Psychological Services, and the Weekly Roster.

The Referral For Psychological Services [RPS] is a form created by SPS that is filled out by a patient's MD to request psychotherapy. This form is the initial step in the psychotherapy program. The RPS form causes a patient to be included on a therapist's Weekly Roster. Once on the roster, a patient may be seen by the therapist and receive psychotherapy. The RPS acts as the physician's medical orders for psychotherapy treatment. This medical order is important to meeting HCFA requirements that psychotherapy be of a degree that a licensed psychologist is required to administer the therapy. The physical nature of the form is identical to the session notes document. It is a multipart, preprinted, carbon copy form requiring the user to fill in the data manually. Table 2 describes each data field on the RPS:

Table 2. RPS Data Elements

Form Data Element	Form Data Format	Description
Letterhead	Preprinted text	SPS corporate letterhead
Fax instructions	Preprinted text	Large type, bolded text stating "Fax with face sheet to ..."
Date	Manual text entry	Date of referral
Facility Information	Manual text entry	A box is provided to write the name of the facility. The box is large enough to hold up to three lines.
Referred by	Manual text entry	Name of referring MD
Physician	Manual text entry	Name of patient's MD
Resident	Manual text entry	First and last name of patient
Room	Manual text entry	Patient's room number at facility
Date of Physician's order	Manual text entry	Date of MD's orders for psychotherapy
Is patient capable of giving consent?	Manual selection	This question is printed in bold on the form with yes and no checkboxes provided.
Has patient received prior psychological services?	Manual selection	This question is printed in bold on the form with yes and no checkboxes provided.
Referral to	Manual selection	There are three track options: Standard SPS, Behavioral Management, and other. Only one selection is allowed.
Reason for referral	Manual selection	37 general behaviors are listed with checkboxes. Any number of behaviors may be selected. The behaviors are generic and negative in nature.
Comments	Manual text entry	Two lines are provided for free-form sentences
Instructions	Preprinted text	There are bold face, large font instructions at the bottom of the form instructing the form writer to fax the original copy to SPS and give the "canary" color copy to nursing to obtain physician's order before sending referral to SPS.

The Weekly Roster [WR] is a form that is sent to therapists on a weekly basis. The WR lists each patient to be seen for psychotherapy in the calendar week recorded on the form. The therapist enters very little data onto this form. Its primary function is to organize the therapist and ensure that all patients seen are legitimate patients, within the guidelines of HCFA, and currently under physician orders to undergo psychotherapy. The table below identifies the data elements existent on the WR:

Table 3. Weekly Roster Data Elements

Form Data Element	Form Data Format	Description
Provider	All data formats are pre-printed text, except provider signature.	The name, title, and SPS ID number of the therapist. The date printed is also on this line
Facility		The SPSID and name of the facility are present. Also included is the number of beds, a contact name at the facility, and a phone number.
Date		This prints the month and days of the week. The day number appears over the preprinted day name column.
Patient name		Patient's first and last name
Acct #		The patient's SPS acct number
Diag		The ICD-9 numeric diagnosis code
INS		The SPS specific code signifying the type of insurance to be billed
SA, SU, MON, TU, WE, TH, FR		Columns to indicate days of the week. The intent is that a mark is made in the day column when the session is conducted.
D/C		No longer used
NoSvc		No longer used
Provider Signature	Manual text entry	The therapist does sign this form once all the patients are seen. This form is returned to SPS and is required for billing payments to the therapist for services rendered.

Other Data

The database will hold additional data to support increased capability for psychotherapist activities. In user meetings, certain data was described as desirable or helpful in increasing therapeutic efficiency. While the patient's name is critical to conducting therapy, biographical and medical history information would help the therapist evaluate the patient's mental state more accurately and more quickly. More information on a patient would allow for the development of rapport with the patient more quickly, leading to quicker demonstrations of therapeutic success. Having access to patient information like date of birth and current medications would have positive impacts on the therapist-patient interaction and the effectiveness of the therapy routine. The psychological state of many elderly patients is affected by medication regimens that may go unnoticed if not readily available to psychotherapists. Should a patient's medications be causing aberrant or undesired behavior, that fact plays a major role in the therapeutic action taken – a medication change may be the most prudent and responsible course of action, rather than behavior modification (Zarit, 147).

HCFA and Medicare are careful to observe the results of psychotherapy, and as such, authorize a limited number of sessions to be paid for. It is critical from a corporate standpoint to not provide services were payment is un-

collectable. Past history at SPS has seen repeated occurrences of psychotherapists conducting sessions with patients whom are beyond their allotment of sessions. This sort of activity reflects poorly on the professionalism of the psychotherapists, and draws into question the validity of any bill submitted by the corporation for payment. Basically, going over the session allotment increases the potential for increased scrutiny of corporate submissions to HCFA. There is no value gained by the company through closer HCFA scrutiny of company submissions.

The patient's physician is a vital component in the psychotherapy treatment regimen. In order to treat a patient, the patient's physician must formally order psychotherapy treatment as medically necessary for the patient's health and well-being. It is important to the company that physicians feel that psychotherapy delivered by company therapists is valuable, beneficial to patients, and administered by professionals. Knowledge of the patient's physician offers a chance to increase the patient pool through increased referrals, and elevate the professional image of company psychotherapists above other psychotherapists. In a similar vein, knowledge of the patient's facility will help in creating a cohesive, complementary treatment record. The facility staff is the primary source of information regarding the patient's behavior. A rapport and understanding with the facility staff will provide benefits in diagnosis and

treatment activities. Knowledge of the staff contact name and facility phone numbers serves to show that the company psychotherapist considers the facility staff participants in the treatment plan. It is important to note that facility staff therapist rapport is a goal set forth in company documentation (Senior 1998b, 4-3).

Problem Domain Objects

In modeling the process and systems currently in use, entity objects may be identified. Objects are things that have properties, and methods to manipulate those properties. Objects directly interact with data, thus identification of system objects and evaluation of their data requirements is necessary. A review of the psychotherapy session process highlights the following objects:

Patient

A patient is a facility resident who has been placed in psychotherapy under their doctor's orders. Patients have many data elements that are static over time, and certain elements that may change over time. Patients come and go, and patient status is variable over time. Patients have demographic, medication, and session data elements. Demographic data (name, account number, contacts, room, etc.) is part of the Patient construct. A patient element also provides a hierarchical framework for organization of distinct medication and session data objects.

Medication

Medications are important data elements for assessing root cause of behaviors. Certain medications used in the long-term geriatric care scenario have side effects that include undesired behavioral modification. Certain medications used in unison may have deleterious effects upon the patient's well being (Zarit, 180). Medication elements hold data such as medication name, dosage, start, and end dates.

Session

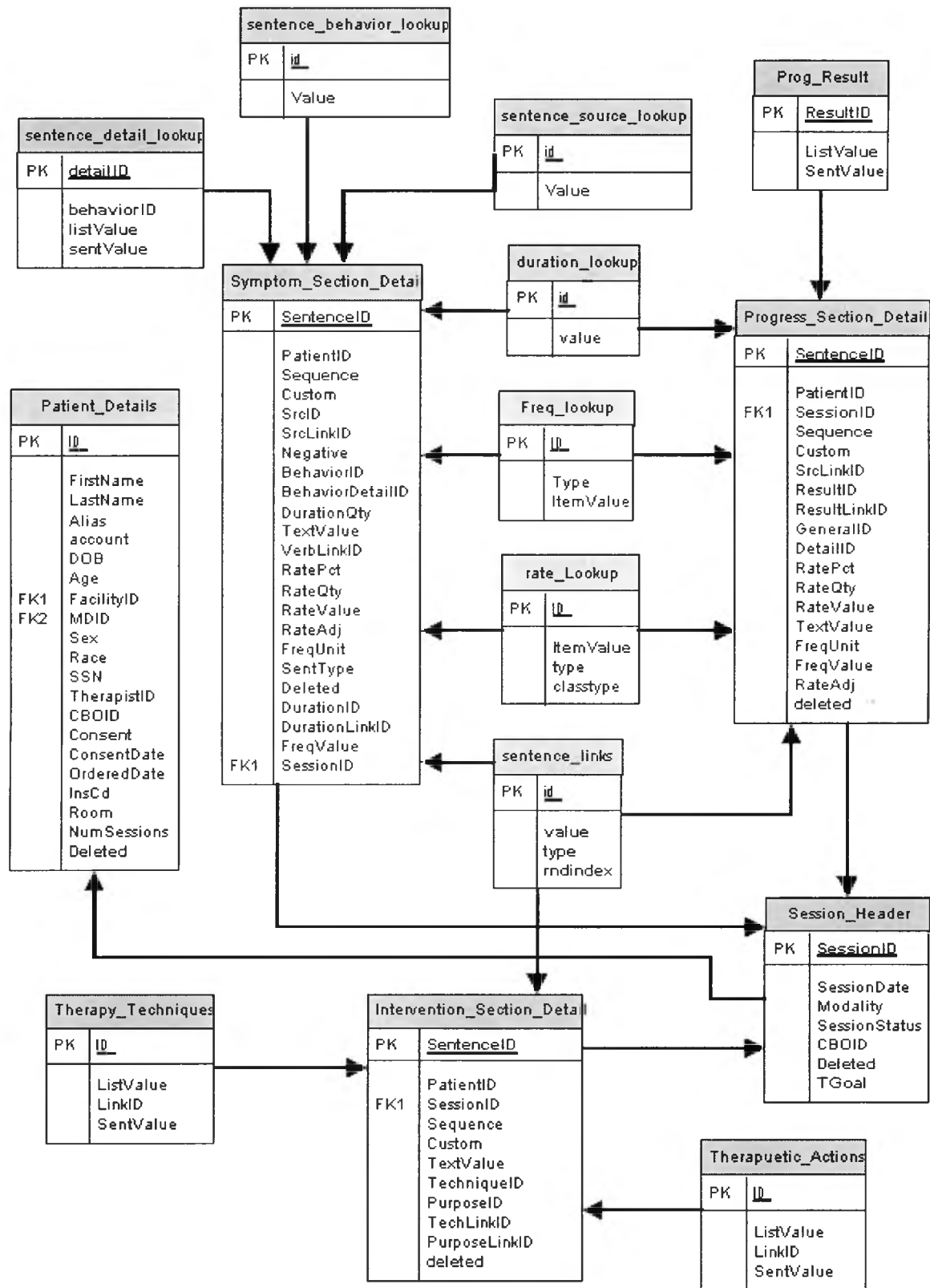
A session is the delivery of services to a patient that is documented in the session note record. Sessions are limited in quantity. Session context data are variable, given that each session is unique. Very few session elements are static from session to session. The session element contains all the data on the session note form except for the patient demographics (name, room, diagnosis, behavior objective, account, facility).

With these primary data objects identified and their scope determined, prototype development can commence. During prototype construction, the relationships between the main data objects will be clearer and more exact, permitting refinement of the data objects' design.

The data objects will access their data from a relational database. The design of the relational data tables is based on dependencies between the data

elements. The data objects that consume data stored in the database are designed based on the tasks they must accomplish. These two perspectives overlap each other at times, leading to the fact that data in one table may appear in several objects, and a single object may contain data from several tables. The database relational design derives itself from the identified data elements used in the current manual system, plus data elements that support the infrastructure of the database itself. Supporting data elements may be characterized as attributes that facilitate key relationships that in turn implement the cardinal nature of data dependency. The design of the database supporting the prototype tested in this paper is illustrated by an Entity-Relation Diagram (see figure 20). This diagram is used to communicate the name and type of each data element (attribute) and the nature of all relationships between tables (relations) in the database.

Figure 20. Sentence Engine ERD



The ERD shown supports sentence creation and storage. Cardinality is indicated by the arrows connecting the relations. The arrowhead marks the “many” end of the relationship. The origin end of the arrow indicates the “1” end. The ERD illustrates certain key points in the database design that require explanation.

Lookup tables: Sentence design makes the use of lookup tables efficient. Different sentence types use similar phrases and words in describing how often and when acts occur. Administration of the data are eased in that one table serves many tables consuming the data.

SentenceID: This is a place where the use of TNF (third normal form) does not meet the table requirements efficiently. On first examination, a sentence could be uniquely identified by the patient id, session id, and sequence number it. These three fields could form a composite key to any of the three session detail tables. However, using this composite key implies that only the most current data for the sentence is stored. If the company were to request that the database hold all edits/corrections/deletions to a session note, there would be no opportunity to review the history of the sentence. It could be argued that by adding the “deleted” attribute to the composite key, thereby accommodating the history aspect. That is only partially true. At most, a history of two records could be kept – the current valid record and the most recently deleted record. As seen

in the analysis of sample session notes, multiple edits to a document are a nominal occurrence. Preservation of all input going into creating a session note is possible by using an integer value as the unique key for the table.

Why would “deleted” data be value? Should a therapist delete the second sentence in the interventions section five times, the database will be able to produce the five deleted sentences when queried. This data could prove valuable to ongoing quality control efforts, and efforts to make the database data as useful and relevant as possible. In the end, the company would decide the value of this degree of data collection. It is implemented for the prototype, though, as an example of the range of possible data collection opportunities. Using an integer value as a table key does come with responsibilities. Programmers must ensure that for any given patient-session-sequence combination, there is only one record that is marked as not deleted. This is not too difficult, however, as the database table itself can have automated safeguards to prevent this from occurring.¹⁴

TextValue: In addition to storing all numeric data needed to reproduce the sentence through the sentence engine, a tuple in the detail relations stores the engine text result as well. This attribute was originally intended to accommodate custom sentences that are entered by the user. By storing the automated sentence text, unnecessary workload on the sentence engine is avoided, and the

¹⁴ In Oracle, for example, triggers, checks and stored procedures can handle tasks of this nature easily

generation of the note document for print out is speeded up by the elimination of sentence engine latency.

The data design is best summarized as the data dictionary supporting the ERD seen in figure 20. This subset of the complete data dictionary represents the most dynamic elements of the prototype, thus the elements most likely to change during the remainder of the project's life cycle.

V. SENTENCE ENGINE DESIGN

Automating creation of session note sentences is the core benefit provided by the documentation system. Most of the time spent in writing a session note is in the status/intervention/progress sections. These sections are most likely to be the source of deviations from the HCFA document guidelines. By automating the creation of a sentence, the system will relieve the user of the composition and organization tasks. Therapists will be concentrating only on the content of the sentence, not how it is worded, spelled, or if it's legible.

The question of how much language processing power the system would require was an important initial question concerning the sentence engine design. The literature on natural language processing is broad and detailed, offering sophisticated models for rendering natural language from word elements. The degree of capability implemented in the sentence engine follows the recommendation of George Smith: "... the governing assumption is that the best model is the least complex" (156). Only enough capability to meet the needs of the session document was designed into the sentence engine. As the paragraphs following show, the sentence engine, in essence, writes three sentences. It can write a symptom, an intervention, and a progress sentence. Each sentence has

some variation, and there is an element of randomness introduced in the phrases used to link the elements. For the most part, however, each of the three sentence types shares a great deal of commonality. This commonality works to simplify the engine design. This simplification is beneficial in ways discussed in chapter two.

Automating sentence construction requires that a common structure, or grammar, be developed such that any sentence can be described in terms of the grammar. Sentence construction begins with analysis of the current user-group's documentation guidelines already in place. The company has already categorized sentences into three groups: current status, provider interventions, and patient progress. Each sentence within a group shares the same purpose. The status section is composed of sentences all dealing with recent patient behaviors and statements that describe the patient's current mental and, to a much lesser degree, physical state (physical state is documented in context of its affect on the patient's mental state), and also meet the HCFA "chief complaint" requirement. The interventions section is concerned with HCFA guidelines on complexity of treatment and medical necessity. The interventions section needs to describe the therapeutic actions taken by the provider to address those issues noted in the current status section, as well as long-term goals and the patient's overall behavioral objective. The company has identified several corporate guidelines for

documenting interventions that provide therapists examples of sentences considered “correct” (Senior 1998c). The third section deals with the patient’s progress in the treatment plan. Here the important aspect is to quantify, or distinctly qualify, the direction and degree of patient performance. Patient improvement or decline is noted here. It is important to recall that medical necessity is most important in this section, as meeting goals will indicate a secession of treatment, and failure to meet goals will indicate psychotherapy’s lack of beneficial effect on the patient. Lack of progress is interpreted by HCFA to mean therapy is ineffective and thus not required. With these general points considered, each section can begin to construct a common framework of structure.

First Design Iteration - Frames

Sentences are the most important element of psychotherapy session notes. Without sentences, no encounter exists from the perspective of HCFA. Other data are important, such as service date, but this type of data is factual and requires very little infrastructure to convey its meaning. Sentences communicate both factual information and contextual information. Sentence information requires a well developed host structure in order to convey its meaning clearly and unambiguously. Broad interpretation is possible of sentence data, thus the

creation of sentences must strike a balance between the widest range of meanings possible and the smallest degree of ambiguity present.

Sentences can be viewed as a dynamic system, where the value of a sentence may change upon modification to any of its component elements. To design a dynamic sentence creation system, the characteristics of the system members must be identified, and the relationships between system members must be defined. Modeling the system in question is one way to determine the aspects of the system and permit the creation of an automated replica of the system. Defining the sentence model is the first step in creating sentences.

Samplings of actual production psychotherapy session notes were reviewed for their sentence content. Subject and verb were analyzed, as well as prepositional phrases and adverb phrases. Trends in structure and recurrent elements were sought. Example sentences include:

“He says he is feeling somewhat better.”

“He expressed appreciation for the opportunity to let his feelings out.”

“Mr. ----- was up in a wheelchair attractively dressed and groomed.”

“Nursing reports that resident is seen taking naps during the day.”

(Durrett, 10 Oct 1999)

Many sentences examined were of a compound nature where two to four distinct concepts are communicated within one sentence structure. Examples of compound sentences include:

“He feels he is unable to do more for himself with self care but has decided to use the stationary bicycle each day to build strength in his legs.”

“This will give him something to focus on versus trying to sleep and not being able to and may in fact add to his sleeping.”

“He cried when speaking of the falls he has had and when speaking of the changes in his life that he believes resulted from the first fall when he was at work.” (Durrett, 10 Oct 1999)

From a syntactical perspective, the sentences seem to be a form of either a single simple sentence or a collection of simple sentences strung together with conjunctions. Two distinct, subtle patterns appeared in the session notes analyzed: The patient was most often the subject of a sentence; and sentences appeared to be composed of the subject, some action or descriptive phrase, then a duration phrase. This relationship formed the initial basis of the sentence frame architecture.

Sentence Frames

The first design attempt at creating dynamic sentences was to allow a user to specify subject/verb/adjective/durations values and then display these values in a sensible text construct. The sentence frame would eliminate the need to coordinate the elements of a sentence into a single statement form. A sentence frame would dictate the sequence of all sentence elements, thus controlling grammar issues. Upon selecting a frame, users would be offered a series of choices for critical sentence elements. A choice would be selecting a phrase from a list. Once the choices are made, the sentence is created. The decision to go with

a framed architecture was facilitated by the realization that there is a finite scope of data elements when discussing geriatric psychotherapy and long-term care geriatric behaviors. Analysis of session notes and treatment planning books indicate a recurring body of concepts and events exist within the psychotherapy session domain. Different patients often exhibit similar behaviors and responses. This narrowing of element value possibilities made the idea of list based selection tenable. This is an example of a sentence frame:

Patient will discuss [EVENT] a with [TARGET] b [AMOUNT] c time(s) within [DURATION] d

Examples of other element types include: [STRESSOR] – a listing of stimuli that cause anxiety or induce stress in the patient; [RATE] – a listing of percentage values to communicate a degree of something. The variety of sentence possibilities may be increased by creation of a new frame using existing elements, creation of new frame using newly created elements, or modifying an existing frame to include more new or existing elements. For example, if a user wanted to make an entry regarding the patient's medications, and it was necessary to indicate medications by name, an element list of medications would need to be provided. A marker such as [MEDICATIONS] would be created, and a list of suitable medications composed.

Table 4. Sentence Frame Architecture Components

Key	Element	Description
(na)	Italicized words	These are the static infrastructure elements of the sentence. The user has no influence on these words – they remain part of the sentence always. Besides creating a sensible sentence structure for the dynamic elements, these words give the user an idea of what kind of information the sentence can communicate. This acts as a sort of “preview” so the user can select the appropriate frame for the idea he or she needs to document.
a	[EVENT]	This is a marker to indicate an event phrase goes in this location of the sentence frame. [EVENT] is but one type of element. The user would select the desired event from a pre-determined list. Once selected, the event is inserted in place of the marker in an appropriate, grammatically sensible manner.
b	[TARGET]	This marker is similar to [EVENT] except that it represents a person or persons. The user would select a target from a list of candidate targets.
c	[AMOUNT]	This marker is similar to the first two markers, except that it represents some scalar number value from 0 to X.
d	[DURATION]	This marker represents some amount of time measured in units like hours, days, weeks, months... This element combines a scalar qty with a grammatically appropriate noun (1 weeks, 2 weeks..)

Once the list once complete, the element could be either added to an existing frame, or a new frame constructed to host the new element.

Advantages of the frame architecture:

- 1) The data elements are well defined. Each entry is specific.
- 2) Frame structure can help users compose ideas into sentences. Users need not worry about how to structure a sentence starting from only a few ideas.

- 3) Granularity of element data are fine. Element members are specific to the element theme. Ambiguity over the meaning of a element is reduced.
- 4) The size of a frame sentence is dynamic, based on the number of elements included. Compound sentences are possible.

Disadvantages of the frame architecture:

- 1) The sentence structure is tightly defined. No variation is possible from the defined structure. If a frame or set of frames is inappropriate for a given idea, a new frame must be created.
- 2) Because of grammatical considerations, the static infrastructure text cannot be changed without analysis into how the change affects the sentence elements. A change in plurality could lead to numerous grammar syntax errors.
- 3) Since each sentence is specific, a large list of base frames must be provided to cover most aspects of psychotherapy. Users would have to go through 40+ sentence frames to find one appropriate to the patient condition to be reported. This adds time and workload to the user that the system is intended to reduce.
- 4) Each element in a frame sentence must have a value. The sentence selected requires that all elements be assigned a value. Blank or non-selected sections result in invalid sentences. It is possible that a user may find a

sentence frame that fits his or her documentary need, but also has extraneous elements that are not needed.

- 5) Frame creation is non-trivial and requires planning, analysis, and time.

The Decision to Abandon Frames

When the user environment is analyzed, it becomes clear that the workload of selecting a frame, then selecting elements, then ensuring the final product meets expectations, is too great to demand of a provider. A provider might have to memorize, or become very familiar with, possibly hundreds of frames and elements. In order to determine if a frame is appropriate, the user must know what values are in each element listing. As the number of element lists increase, so does the memory retention required of the user to make productive use of the sentence frame methodology. It is contradictory to the stated general goal of simplification for the user if the system imposes excessive mental workloads to support the automation methodology. The automated system would require extensive study and memorization or at least an extensive help system to maximize efficiency. This increased user workload overcomes any efficiency introduced by the system. The sentence frame concept was abandoned as the primary sentence generation technique.

Second Iteration – Linked Elements

The ease of use design requirement had not been met in the frame design

scheme. The sentence design must not require significant memorization yet be flexible and tolerant of individual user practices. The rigid frame architecture did not compensate for users failing to make or missing an element selection.

Documentation sources were reexamined: company document guidelines, HCFA document guidelines, and actual, non-denied, field session notes. Upon further review of all the data regarding sentences, it became clear that a recurring theme was each sentence had some important piece or pieces of data that met one or more of the guidelines and requirements. The base structure for a sentence could be viewed as a group of necessary data elements linked together to form a sentence. A basic representation of the model is:

$$\textit{Element1} + \textit{Link} + \textit{Element2} + \textit{Link} + \dots\dots$$

where “ElementX” is some necessary data, and the “Link” is a word or words necessary to sensibly connect the preceding element to the following element.

From an English grammar standpoint, this basic model is consistent with standard sentence formation where one may have a noun or noun phrase followed by a prepositional phrase.

Each data element will require some form of UI representation such that users may see the data and manipulate it. The number of elements in a single sentence then became a limiting factor – in a design where the UI size is fixed and not scrollable, a large number of elements will not be manageable. The

sentence needed a static set of data elements. Having a static number of data elements provided many benefits to outweigh the potential downside of a rigid sentence structure:

- 1) The sentence UI was fixed. The UI would not be changing from sentence to sentence within a section. UI widgets would not be added or subtracted from the interface. The number of elements would remain constant for a given sentence.
- 2) Usability is increased. A fixed UI requires less mental effort to operate than a UI that presents a dynamic, changing appearance to a user.
Every sentence in a section has the same structure.
- 3) Training time is minimized. Users would require more training to deal with the greater number of variable outcomes in the UI, if the number and type of UI widgets was not static.
- 4) Ease of use. Static UI widgets require the least amount of user effort to understand and operate.

Analysis of existing sentences along with mapping guideline requirements to data elements revealed that, for a status sentence, elements listed in table five were necessary to produce acceptable sentences.

Table 5. Current Status Sentence Architecture

Element	Description	Requirement Mapping
Source	The person, persons, entity, or document that was the source for the data presented in the sentence.	English language syntax
Behavior	The act, action, physical activity, or verbal comment that the patient reported or was reported to have committed.	Chief complaint
Detail	This is to provide contextual detail for the behavior described. This is to enhance the reported behavior and it's role in the patient's mental state.	Medical necessity & Ph.D. level complexity
Rate	This is to denote the amount at which the behavior occurs	SPS guidelines
Frequency	This is to denote the duration of the reported behavior.	SPS guidelines

In addition to the data elements, there are four links required to join the five data elements. The status sentence model has developed into:

Source + Link + Behavior + Link + Detail + Link + Rate + Link + Frequency

The order of the five data elements was based on a few assumptions and observations of existing session notes. First, detail logically follows behavior. Second, as the sentences reviewed commonly described events of the recent past, it is logical to move the rate and frequency data values to follow behavior and detail. Third, rate precedes frequency as a matter of sensibility. Common sense tells one that “how often” comes before “how long”. The remaining data element, source, is in actuality the subject of the sentence, and English grammar syntax holds that the subject precedes the predicate.

One other consideration that bore out of the first design iteration is that while automated sentences can represent most of the conditions, events, and details of a patient's mental condition, there are occasions where the sentence engine will be unable to successfully document the condition. By not allowing free form user text entry, it is certain that there will be encounter sessions that are improperly documented. This eventuality is completely unacceptable, and must be addressed in a successful sentence engine design. Thus, the concept of sentence types was introduced in the second iteration. Custom sentences are those sentences that the user enters manually via the keyboard. It is 100% user driven as to sentence content. By virtue of its flexibility, it allows any documentation need to be met.

Though free-form sentences were part of the sentence engine design, the free-form sentence requires a great deal of effort on the part of the user. To meet the requirement that the documentation system reduce user workload and increase efficiency, automated sentences must be utilized to the greatest extent possible. The inclusion of custom sentences does not allow for any restraints on the scope of the automated sentence design. Automated sentences must encompass the largest amount of data possible in order to reduce the amount of effort required to document the session.

With the sentence positional structure established, the question of format

and style becomes important. To be a sensible sounding sentence, the data elements must agree with each other in plurality and tense. Tense was determined to be 100% past tense. This decision was facilitated by several factors: over 98% of status sentences reviewed made use of past tense; present and past tense combinations required multiple versions of words and of link phrases that added complexity to the design without a tangible payoff in sentence creation efficiency; and lastly, past tense was the sensible tense to use for the situation. The decision to use past tense had an important effect on the implementation of behavior.

Behavior was modeled as a one action-word phrase. This facilitated a design feature not yet discussed, optional data elements, as well as offered the user a one-word selection to describe a patient's basic activity. Being past tense, there were two options for one-word behavior phrases – a list of “-ing” words or a list of “-ed” words. Both versions of a word are in past tense (e.g. “was crying”, “has cried”). However, the “-ing” version of verbs offers additional capabilities. While the current model depicts sentences as all past tense, “-ing” offers the flexibility to include present tense at a future date. The number of possible links for “-ing” words is greater than that of “-ed” words. In some examples, “-ed” words had no suitable linking word – the appropriate structure was *element* + “-ed”. This is a limiting factor. “-ing” words have multiple link

possibilities that apply to any word that ends in “-ing”: “was”, “has been”, “had been”. This pool of links would allow sentences that did not resemble each other too closely (another improvement over the frame design).

The definition of source and behavior to this level allowed for a list of suitable source to behavior links to be created. At this point, the concept of negation was introduced. Many status sentences described patient conditions in the negative (e.g., “not cooperating”, “not dressing”). The source to behavior link was the logical place to contain the negation so both positive and negative links were created. The UI would have an interface for the user to specify either a positive or negative sentence.

Detail data was dependent on the behavior selected. The nature of detail is to augment behavior by providing a contextual setting for the behavior. Each behavior data element had a custom list of detail phrases created for it. These detail phrases were determined through session notes review and treatment planner guides. As each detail is specifically created for its master behavior, the need for a link between behavior and detail is abated by making the link inherent in the detail phrase.

A powerful feature of the linked element sentence model is the fact that most elements are optional. No sentence is required to have more than one element. For current status sentences, behavior is the required selection. A

sentence consisting of just a behavior selection would be written in a manner like “Patient was crying”. A behavior-only sentence has limited applicability, but is valid. Each sentence element selected in addition to the required element adds a cumulative degree of detail and specificity to the sentence. Both “Patient was crying” and “Nursing staff reported that the patient was crying over the death of their spouse four times in the last two days” communicate the same basic idea, but the second version adds much more contextual information. This added contextual information aids in establishing medical necessity, and thus reduces the chance for payment denial. Yet while beneficial to add as much as possible, the user has the option to provide whatever level of detail desired.

Table 6. Therapeutic Interventions Sentence Architecture

Element	Description	Requirement Mapping
Technique	The professional level action or method used by the therapist	HCFA guidelines
Purpose	What the technique was supposed to accomplish	SPS guidelines

For intervention sentences, the therapeutic technique is required.

Therapeutic techniques are those practices or activities that are recognized by Medicare/HCFA as being effective in the treatment of mental disorders.

Examples of therapeutic techniques would include: supportive listening, rational-emotive therapy, and cognitive reframing (Senior 1998b). While it is recommended that a purpose be selected to augment the technique selection, this

is not required. The primary goal of the intervention section is to meet the requirement that the services provided by the therapist were of a nature and complexity that Ph.D. therapist-level skills and expertise were required to administer the treatment.

Table 7. Progress Sentence Architecture

Element	Description	Requirement Mapping
Result	General nature of the inclination of the patient's behavior and mental state	SPS guidelines
Behavior	The behavior or patient activity to which the progress assessment is referring	SPS guidelines
Detail	Extra contextual data regarding the behavior.	SPS guidelines
Rate	The degree to which progress has changed	SPS guidelines
Duration	A statement of time setting a temporal context for the rate	SPS guidelines

Progress sentences begin with the concept that the sentence must give an indication of what the patient's progress is towards the treatment goal. After review of session notes and the Company literature on session documentation, two levels of progress reporting became apparent. This document shall refer to the two progress levels as high and low, in reference to the associated level of abstraction. The high level reporting consists of answering the question: "Has the patient improved, stayed the same, or worsened?" This type of report is usually linked to the treatment goal or one of the patient's identified symptoms.

However, in notes for sessions where payment was denied, this high level assessment was stated generally, such as “Patient seemed better”, or “Patient improved.” Low level reporting includes specific event reporting. Low level reporting documents those actions and behaviors done by the patient that are indicative of the patient’s mental health. An example of the low level report would be a statement like “Patient had three social interactions in the last week.” This sentence is a quantified statement of behavior that when taken in account of a stated treatment goal of increasing a patient’s socialization shows direct impact of psychotherapy in the patient’s behavior.

In creating an algorithm to write a progress sentence, a range of sentence types had to be specified. This range of sentence types had to account for ways to create both high and low level progress sentences. Corporate guidelines, developed with respect to HCFA documentation requirements, were used to determine what issues progress sentences would document:

- a) Increase/decrease/maintenance of behaviors identified in patient symptoms.
- b) Patient efforts to record or write down facts regarding events or issues that affect their behavior.
- c) Increase/decrease in general behaviors common to residents in long term care facilities

- d) Statements made by patient regarding therapy, their behavior, or any topic related to the psychotherapy.

Progress sentences keep the element and link structure described earlier. Thus for each sentence class (A-D) identified above, the elements and necessary links must be specified.

“Patient” is the subject of all progress sentences. Issues a-d center around some aspect of patient behavior. Thus, all sentences will have “patient” as the subject. It can be written that “Patient did..” any of the four issues above, so the first structure of a progress sentence is:

“Patient” + link + elements [] (set of all sentence elements).

For sentences of type A, it should be possible to specify a direction (increase/decrease/steady), behavior, rate, and duration. Specifying direction is straightforward, and is a required element of a type A progress sentence. The direction value is not valid unless it has a context – behavior is also a required element of a type a progress sentence. In keeping with the requirement that progress sentences have reference to patient behavior, the behavior element can reasonably be equivalent to those behavior elements in current status sentences. The behavior element shall have two parts – the general behavior and an optional detail component. The progress sentence structure now appears as:

Patient + link + direction + link + behavior + [detail]

Note that there is no link between behavior and detail; this is in keeping with the behavior-detail relationship in current status sentences. The brackets around detail indicate it is an optional piece and not absolutely required to create a sentence.

The rate element of the progress sentence provides a scale for measurement of the direction element's strength. Rate shall be similar to rate expressed in current status sentences – percentage, scalar, or subjective amounts may be indicated. The duration element indicates the length of time required to achieve the behavior status. Duration will be used in a manner consistent with current status sentence duration. The sentence will order rate before duration, in keeping with current sentence status format. With all the progress elements identified for type a sentences, the sentence structure is:

Patient + link + direction + link + behavior + [[detail] + [link] + [rate] + [link] + [duration]].

Type B progress sentences are of a prescriptive nature. The sentence shall identify an action the patient will perform. It is a sort of verbal contract between patient and therapist, where the patient commits to some therapeutic action, and the therapist will review the results with the patient. A typical example of a sentence like this is: "The patient stated they would write down where they were and what they were doing when they feel lonesome or sad." This type of written

record is a proven technique for use in patients with reduced memory capability (Zarit, 190). Type B progress sentences are less complex in nature when compared to type a sentences. There are two elements to account for: the agreement and the behavioral condition. Both elements are required for a type B sentence. This produces a structure of:

Patient + link + agreement + link + condition.

Type C sentences are moderately more complex than type B. The purpose of type c sentences is to indicate trends in general behavior that may have a role to play in the specific behaviors identified in the current status section. An example of this type of expository sentence would be "Patient was more talkative at end of session", or "Patient was less remunerative." These general descriptions of patient behavior may be both: valid (though less detailed) indicators of a patient's mental state, and a qualifier for some customized, anecdotal statement about the patient that a therapist may wish to add out of personal/professional practice. The type C sentence structure would be:

Patient + link + Behavior + Rate + Duration

where patient is a constant, behavior is a required selection from a list of specially chosen general behaviors (different than the action behaviors of type a sentences), rate is an optional selection, and duration is optional as well.

Type D sentences help meet HCFA requirements that a patient be of a

physical condition that they can take an interactive role in psychotherapy. This role requires that a patient be able to hear and speak with some degree of lucidity. A sentence that describes the patient's verbal responses to therapy confirm that the patient understands what is being attempted, and is able to participate in a way that allows the patient to gain benefit from the therapy.

Example sentences of this type would be "Patient stated they wished to continue therapy", or "Patient said they felt therapy was helpful in dealing with their problems", or even "Patient said they didn't want to get therapy any more."

Review of session notes and treatment planning guides suggest that there are recurring, common statements that patients in therapy make. These statements shall be pre-scripted and available for inclusion in the progress section. A therapist would select an appropriate topic to gain the sentence. The sentence structure for type D sentences is very simple:

Pre-scripted phrase

where the entire sentence is chosen from a descriptive clause. The available clauses would appear as first element choices.

Sentence Design Data Requirements

In the linked element sentence model, the data requirements are clear in that all components of a sentence are known entities from a particular set of data. To design the database tables supporting the sentence engine, the needs of the

most complex sentence configuration possible identifies the necessary table attributes. The table design is completed when relational integrity and other infrastructure supporting data are assigned attributes in the table. The table design is closely coupled with the sentence architecture it supports: any change to the sentence configuration has repercussions to table design (attributes may need to be added/deleted/modified depending on the change). Given this close coupling to the database, it is advisable to consider changes to existing sentence configurations carefully. Changes to sentence configuration should be few and far between with coordination between data and application personnel. The database table designs are discussed in chapter 4. The sentence tables' ERD may be seen in figure 20.

VI. HARDWARE DESIGN

Personal Digital Assistants have been in the marketplace for some 5 years now, using the Palm Pilot as the epoch of a family of hand-held computing devices. (It is noteworthy to mention that micro-computer hardware dates back to the early 1980's, most notably with the Timex Sinclair computer.) PDAs held the promise of desktop-like computing in a calculator size package, embodying the notion of truly mobile computing. The PDA device would seem ideal for geriatric psychotherapy given that it is moderate in price, lightweight, portable, small size so as to be un-intrusive, interoperable with desktop computers, and capable of presenting an already accepted GUI experience (MS Windows).

With these possibilities in mind, a Windows CE (now known as "Pocket PC") PDA device was procured for suitability testing as a platform to host a documentation system. An Everex A-20 model (circa 1999) CE device was purchased, along with modem and secondary battery carrier. Microsoft software development kits for Visual C++ and Visual Basic were purchased as prototype platforms. A Canon BJ-50 portable printer was purchased to support a fully mobile documentation system that could fit into a small case or bag. The prototype test results showed quite clearly that the promise of PDA was not

achievable in practical use. A number of usability and operability issues made the Everex PDA device little more than a handheld solitaire game. Problems with the PDA were found in these areas: interoperability/interface to host PC, SDK for developing applications to run on the CE platform, battery life, CPU performance, and manufacturer support. Additional usability issues arose when considering the human-computer interface implications of the PDA platform.

Interoperability/Host Interface: The Everex A-20 PDA documentation clearly states that the PDA is to be used as an extension of a host PC. The PDA must have a relationship with a host PC in order to load programs and data, and transfer data from the PDA. The relationship between PDA and host was to be conducted through a Microsoft software application called CE services (there were many versions before Microsoft abandoned the software – for this exercise, we used version 2.1, which was shipped with the product on CDROM). Once installed on the host PC, the PDA is connected to a serial port on the host PC bus. This initiates a task where the RAM memory of the PDA becomes an available resource to the host, similar to a mounted hard drive. With this relationship, data and programs may be moved from PDA to host, and vice-versa.

The PDA solution failed here in the bug prone code of the CE services. Installation of the CE services on a desktop PC required many sequential steps. The VBCE toolkit was an add-on software component to VB 5 that allowed

development of CE applications on a Windows NT desktop, and software emulation of the PalmPC (PDA) on the NT desktop for testing. Installation of the toolkit was very involved, as was the process to configure the NT desktop to connect to the PDA. Configuration of the NT desktop to permit connection to the PDA required a minimum of 17 steps. In these 17 steps, there are a minimum of four system restarts and two installations of the NT service pack three (MSDN Q192985). If an error was encountered along the way, re-installation of the NT service pack and system reboot were most often specified. Installation of the Visual Basic Toolkit required additional re-installations of the NT service pack three. Over the course of three months, the development system for this project was operational less than 10% of the time. The software simply did not function as advertised in a stable manner. Reliable, stable development platforms were not possible to maintain.

This experience was echoed by PDA software developer Lee Church, who stated that CE development systems had operating system, service pack, and toolkit re-installs done on a weekly basis when developing CE projects. He estimated the cost differential between Palm OS development and CE development to be a minimum of 3 to 1. His practice was to strongly advise PDA software clients to opt for a Palm OS platform instead of CE, based on his experience with the connectivity software and development platforms (Church,

1999).

CE SDKs: Microsoft's marketing plan for WinCE was based on three tiers of development software. The primary tier consisted of the development language IDE of choice, and in the case of WinCE, namely Visual J++, Visual C++, and Visual Basic. The secondary tier consisted of WinCE toolkits, or software applications designed to run within the primary tier IDE and provide compiler support for the special microprocessors present in CE devices. The tertiary tier consisted of development support software specific to the CE hardware platform. The main feature this tier supplied was PDA emulation capability on the IDE platform system. The first two tiers were purchase only, with the development tool costing about \$500 (1999 dollars) as a standalone application, and the second tier toolkit costing \$200 (1999 dollars). The last tier, classified as an SDK by Microsoft, was freely available as a download. It was intended to be versioned on a regular basis to provide support for increases in CE device hardware capabilities.

The SDK suffered the same degree of instability as the toolkits. For example, the PalmPC SDK downloaded and used in this project required an additional two re-installations of the NT service pack version three. In addition, no other SDK for WinCE could co-exist on the development server. The SDK came in three types: AutoPC, Handheld PC, and PalmPC. AutoPC was a rare,

highly specialized environment. Handhelds and PalmPCs, however, were quite close in capability and market niche. Both systems ran the same versions of Windows CE. Where Handhelds and PalmPCs differed was in their form factors. The Handheld screen was 640X240 while the PalmPC was 160X240. Once installed, the PalmPC SDK emulation was erratic in that it did not respond to window events as expected (click events primarily). The GUI representation of the PalmPC on the desktop was not always correct – intermittently, areas would be shaded or blacked out and rendered unresponsive.

It is important to note the past two years of market contraction history of Windows CE toolkits, SDKs, hardware platforms, and the operating system itself. Windows CE as an operating system trademark was abandoned in 2000 for the moniker “PocketPC”. In the development software arena, Microsoft dropped the Visual J++ toolkit from the market altogether. For the PocketPC, the once separate toolkits, costing \$200 dollars each, were combined into a downloadable package, at no charge, from Microsoft’s website. As for the three SDKs, all three have been pulled from the Microsoft website and are no longer available. The abandonment of the “Windows CE” marketing plan, the move from for charge to for free software toolkits and the consolidation of different toolkit IDE platforms into one package, and the removal of the hardware specific SDK lend credence to the conclusion that Microsoft’s Windows CE products in 1999 were not ready for

production distribution and use.

Battery Life: Battery life for a PDA is eminently more important for a PDA than for a notebook computer. The PDA has no persistent dynamic memory. The operating system is embedded into a ROM chip along with certain application software. All user data are contained in a RAM chip inside the PDA. Thus, constant voltage must be present to maintain data. The Everex A-20 PDA was built with a 3-tier battery system. Primary power came from two onboard AAA rechargeable batteries. Backup power came from a watch style battery. The backup battery's purpose was to maintain data should the primary battery be depleted for a short time. The auxiliary battery system consisted of a base for the PDA that had an RS-232 serial port and an RJ-11 modem jack built into it, as well as dual AA rechargeable batteries. When the PDA was docked in the base and then used, the battery power was theoretically more than doubled. The manual stated a broad range of battery life duration, but the average life was 2-4 hours on the primary system and 4-8 hours when using the base auxiliary power. The battery endurance table outlines the average battery duration seen in the CE device. Battery life was recorded for four distinct types of activity: light use, no backlight – playing solitaire with no other tasks running; light use, backlight – playing solitaire with the backlight; heavy use, no backlight – recording and playback of voice messages, creating notes, starting and stopping multiple tasks;

heavy use, no backlight – recording and playback of voice messages, creating notes, starting and stopping multiple tasks with the backlight. Batteries were charged to maximum on the CE device power indicator and then timed until 50% capacity was reached. The device became essentially inoperable below about 30% power. A 50% power rating represents a compromise of risk and device availability.

Table 8. Everex A-20 Battery Endurance

Task	AAA	AAA/AA
Light use, no backlight	74	160
Light use, backlight	49	93
Heavy use, no backlight	63	148
Heavy use, backlight	42	76

Notes:

- a) All endurance values shown in minute units.
- b) AAA refers to primary battery only
- c) AAA/AA refers to primary and cradle secondary battery

The battery life results shown clearly indicate that the PDA device is not likely to last a full day of psychotherapy sessions without frequent stops to recharge batteries. Battery recharge time from 50% to 100% runs just under 1 hour in time which places stringent limitations on the therapist's mobility. On this aspect alone, the Windows CE device option for a documentation system platform is untenable.

CPU Performance: The Everex A-20 microprocessor is an NEC VR4211 series chip running at a clock speed of 66MHZ. Yet, the CPU performance in

terms of response time was less than desirable. In using the device, one experiences noticeable stutter like response from the unit. When opening a file or creating a note, the device would at times pause to complete processing even though the user had continued to touch the pad and enter data. Playback of audio files at times would not immediately follow the button action. Sounds associated with actions at times would not be temporally in sync with the user action. For example, when using the soft keyboard (a GUI representation of the QWERTY keyboard on the device screen), the device played an audio sound of a manual typewriter keystroke for each soft keyboard click. At a certain input rate, the audio playback was noticeably behind the input actions. While this was more an annoyance than a hindrance, it illustrates the CPU utilization is high. An additional indicator of inadequate CPU performance is task shutdown. In Windows CE, there is not the concept of closing an application. Once started, an application continues to run as a background task until the unit is turned off or until the user goes into the task manager utility and ends the process. The task manager utility lists out the active tasks and permits the user to close a task by selecting it, then clicking a close button. As an example, when closing the Windows NT version of solitaire, the application is removed almost immediately from the NT task manager list. In Everex's Windows CE, the same operation requires three to five seconds. During this time period, an animation of a rotating

hourglass dominates the CE display. This large discrepancy in application shutdown time between CE and NT demonstrates that CPU utilization in CE is higher than in NT and thus a large contributor to the CE battery drain.

Manufacturer Support: For an operating system to be successful, the services it offers must meet the needs of a large population. The operating system is involved in all aspects of the device functionality, but it is the applications hosted on the device and the support of the hardware that make a device useful to users. Already, the software development kits supplied by Microsoft for CE were shown to be bug-prone and unstable. This has the effect of limiting the number of CE applications that are available to the public. Limited software means a reduction in the size of the user population. When manufacturers begin to stop supporting the hardware, the operating system is not going to be viable in the marketplace. Windows CE experienced this problem with about half its hardware vendors.

In October 1999, CNET reported that Philips was exiting the handheld PC market effectively killing all CE devices Phillips made to that date. Philips had model lines in the handheld space (Velo) and in the PalmPC space (Nino). The article noted that “customers have complained that Windows CE is difficult to use and synchronize with devices, issues Microsoft is trying to fix.” Also mentioned in the article is Microsoft’s refutation that problems with the

operating system contributed to the Nino's demise (Miles 1999a).

Everex, the maker of the PDA used in this project, announced it was abandoning its PalmPC product line a month after Philip's announcement. Everex stated the reasons for discontinuation were two-fold: lack of interest in monochrome CE devices and scarcity of color CE display hardware. Yet in the article, an unnamed Everex spokesman is quoted as saying, "Battery life is the major drawback", referring to battery life that is far shorter than PalmOS devices. The article also states: "Microsoft denied the assertion that Everex's bad fortune in the market was due to any problems with the company's [Microsoft] software" (Miles 1999b).

IBM also followed Philips and Everex in dropping Windows CE devices from their product line. It was reported in March 2000 that IBM had discontinued the Workpad Z50, a handheld Windows CE device. It continued to produce a line of Workpad devices based on the PalmOS operating system. Microsoft was reported in the same article to have stated that they would improve the software for PalmPC devices, but announced no plans to update Windows CE versions for Handheld PCs like the Workpad Z50. The article, as in the previous articles, noted consumer complaints that the Windows CE operating system was "glitch-prone, difficult to use, and hard to synchronize with other software, including Microsoft Windows." Microsoft was also noted to have said

the devices were not going away, but that they were fading from view as PocketPC prepared for an April 19, 2000, launch (Miles 2000).

On Microsoft's corporate site, the SDKs are no longer available for download. The Visual Basic CE toolkit, while displayed, is not available for purchase through shop.microsoft.com. The Visual C++ toolkit is available with the caveats that the development system be Windows NT with service pack three. It is important to note that as of October 1, 2001, the Windows NT family of operating systems will officially retired by Microsoft. The C++ toolkit is necessary to support the industrial deployments of Windows CE, known as embedded CE, rather than to support any PDA type device development.

PDAs, Navigational Threshold, and Text Management: In light of the issues above, the Windows CE platform is not an option for use in mobile documentation systems for any industry. The hardware and software available for development is not of a reliable and stable nature so as to ensure consistent operation. The Everex A-20 is now nothing more than a solitaire game given it's inability to connect to other systems, lack of third party applications, battery life of less than two hours, and no persistent storage built in. There is, however, another PDA option to consider that overcomes the Windows CE shortfalls listed above. The PalmOS operating system has had great success at interconnectivity and reliability. PalmOS devices run on regular alkaline batteries with lifespan

measured in weeks rather than hours. The PalmOS device is comparable to the Everex Windows CE device in screen size, unit dimensions, and unit weight.

Why would this device also not be a candidate for use in a mobile documentation system? The answer comes out of user testing with the Everex.

User requirements showed there are several levels of hierarchical data that must be displayed: patients, patient details, patient sessions, session details, session sections, session section details, and completed note forms. The relationships of the layers are such that the user may have to navigate through multiple levels in a sequential order to work with desired data. Each layer can be thought of as a screen interface to the user. To enter information for a session's progress section, a user must, at a minimum, select the patient, the session, then the symptom section. The PalmOS screen size is 160x160, so the amount of data displayed is not great. What results is that several screens must be navigated in order to reach specific data. This navigation path is not accessible to the user from any given screen – there is not enough space to display the user's location in the application, and the application itself, on one screen. Users must keep mental note of where they are in the application. They must also know how to go from section to section in the application. Should there be multiple paths to select from, the demands on the user's attention increase noticeably. In working with the Everex unit's built-in software applications, this author found himself

hesitating when moving through applications of more than four screen levels.

This author found that he was concentrating on remembering how he got to application functions that he needed to use more than the data he was attempting to process.

The Everex unit was given to a psychotherapist representative of the company's therapist population. The therapist was asked to open applications, play the solitaire game, create appointments, enter note messages, and make sound recordings. While using the applications, those applications that required more than three screens of interface produced a greater number of requests for help than did the single screen applications. Users were able to work the solitaire game with few questions, but when working the note system, the number of questions rose by over 100%. When asked how they liked the unit after about 30 minutes of use, the response was negative. Research into user satisfaction and UI navigational effort may reveal a navigational threshold that defines how much application architecture a user can retain and use. Such usability research has potential benefit to any resource constrained user interface driven system.

In conclusion, the PDA platform for medical documentation automation was not a valid option. The remaining alternatives were laptop/notebook computers, or traditional desktops. Desktop systems have logistical and storage requirements that could not expect accommodation by all resident facilities as

discussed in the environmental requirements section. Thus, the viable choice for the prototype's interface development is the laptop/notebook computer.

Microsoft Windows was the only operating system considered due to its overwhelming operating system market share.

VII. USER INTERFACE DESIGN

The user interface [UI] design for this software system went through three distinct phases culminating with the final UI design described in phase three.

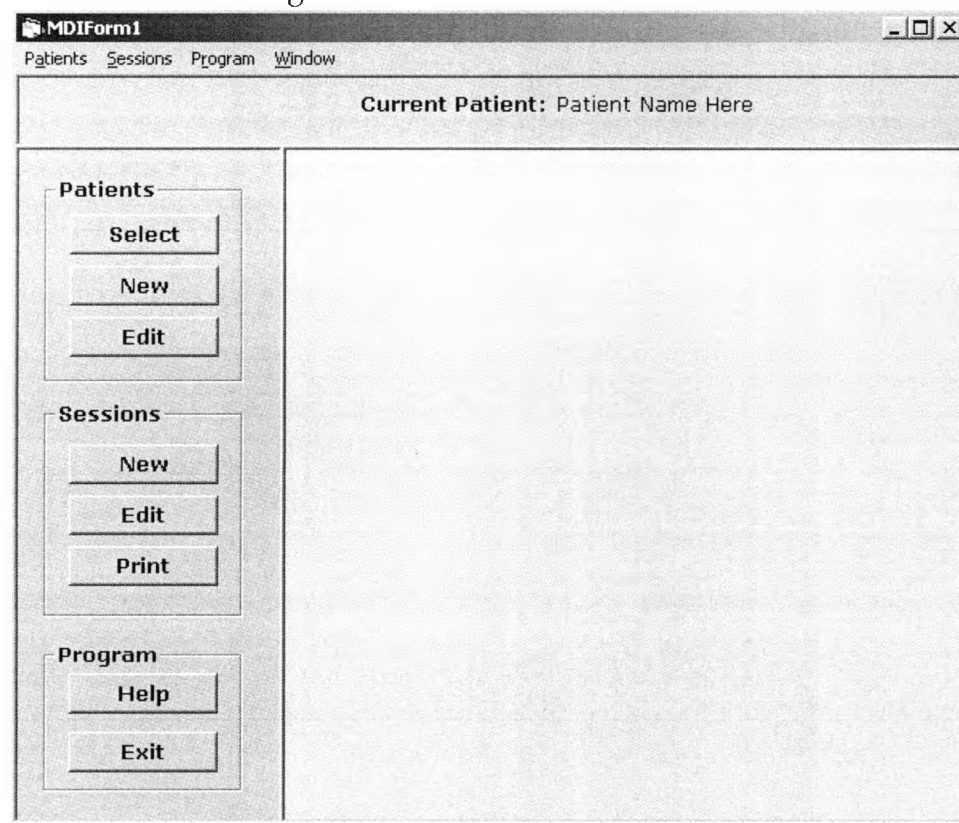
Phase one and two can be generally characterized as mutations – the general scheme in which data are displayed and accessed shares fundamental techniques with existing software systems. Phase three is evolutionary – the scheme for interacting with data is not yet found in software systems used for the same purpose, but is used in other software packages with success. The application of the successful technique in a new problem domain is innovative. The three UI design phases will be reviewed in the order in which they were developed. The design methodology was to create a prototype mock-up of the base UI for the notes system. A practicing company therapist then reviewed the mock-up. From the discussion regarding the design and its characteristics, subsequent design phases ensued, culminating with the phase four designs.

Phase One (Traditional Based): Phase one began with a screen where widgets had unary event models – that is, a button or menu item fired one task, and only that one task. Widgets did not have different functionality for different situations. Inside the “Patients” frame, the “New” button created a new patient.

It did not create a new session or new note. Buttons on the screen were specific and closely coupled to their designated task. Program actions were initiated by button clicks, menu clicks, or keyboard input (shortcuts/hotkeys). Figure 21 illustrates the phase one main screen.

Traditional menu access appears along the top from the left side going right. The data bar directly beneath the menus spanning the UI is for displaying parent reference data for whatever data are in the main panel (in fig. 21, the main panel is empty).

Figure 21. Phase 1 – Main Screen



Most common task actions are fired via the buttons stacked in the left column.

“Patients” and “Sessions” each have “New” and “Edit” buttons while “Patients” has a “Select” button, and “Sessions” has a “Delete” button. Tasks that were implemented on a button are those tasks identified by the user community as being most important. The menu system provided redundant access to these tasks as well as all other necessary tasks of secondary importance.

The patient data screen demonstrated the use of tabbed property pages to render subsets of data on independent, dedicated screen space. Patient data at the highest level of organization appeared in appropriate UI widgets¹⁵ at the top of the patient UI screen. Specialized data are grouped by section (three total) and displayed in a tabbed property page. Figure 22 shows the phase one patient screen with the Behavioral Objective section selected. Data entry is straightforward and pick lists are supplied for those items where the values are known prior. “Save”, “Cancel”, “Close” buttons offer edit capabilities for the patient data.

The note generation interface was a rudimentary list box selection design. Again, a tabbed property page interface allowed separate screens for Current Status, Interventions, Progress, and the culminated Notes. The three data input screens shared the same UI design; the Notes screen echoed the contents of the

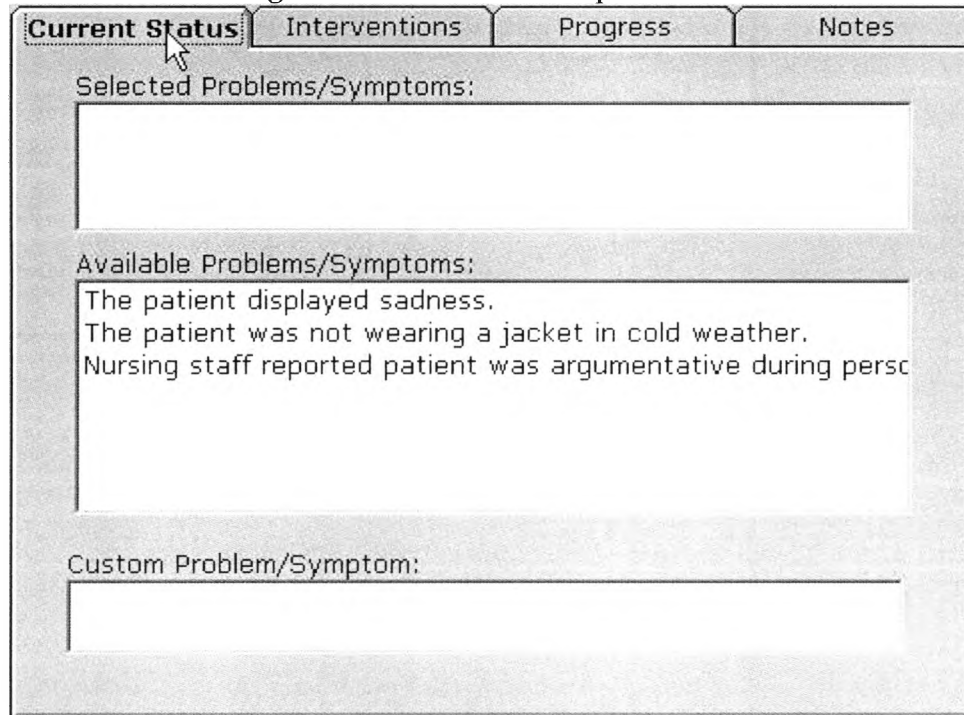
¹⁵ Examples of widgets include the textbox, combo box, list box, radio button, and checkbox

Figure 22. Phase 1 Patient Data Screen Model

three input screens back to one screen (in three distinct text boxes – one for each area). Figure 23 shows a sample of a sentence input screen, figure 24 shows the Notes screen.

The UI's *modus operandi* was planned to conform to UI guidelines published on the SEPA website and made available in Microsoft's UI guideline text. The guidelines advocated are largely sound and reasonable. They are generalized in nature such that they are applicable to a wide range of problem domains.

Figure 23. UI Sentence Input Screen



The UI Sentence Input Screen features a tabbed interface with four tabs: 'Current Status', 'Interventions', 'Progress', and 'Notes'. The 'Current Status' tab is selected, indicated by a mouse cursor. Below the tabs, there are three input areas: 'Selected Problems/Symptoms:' with a large empty text box, 'Available Problems/Symptoms:' with a list of three sample sentences, and 'Custom Problem/Symptom:' with a single-line text input field.

Current Status | Interventions | Progress | Notes

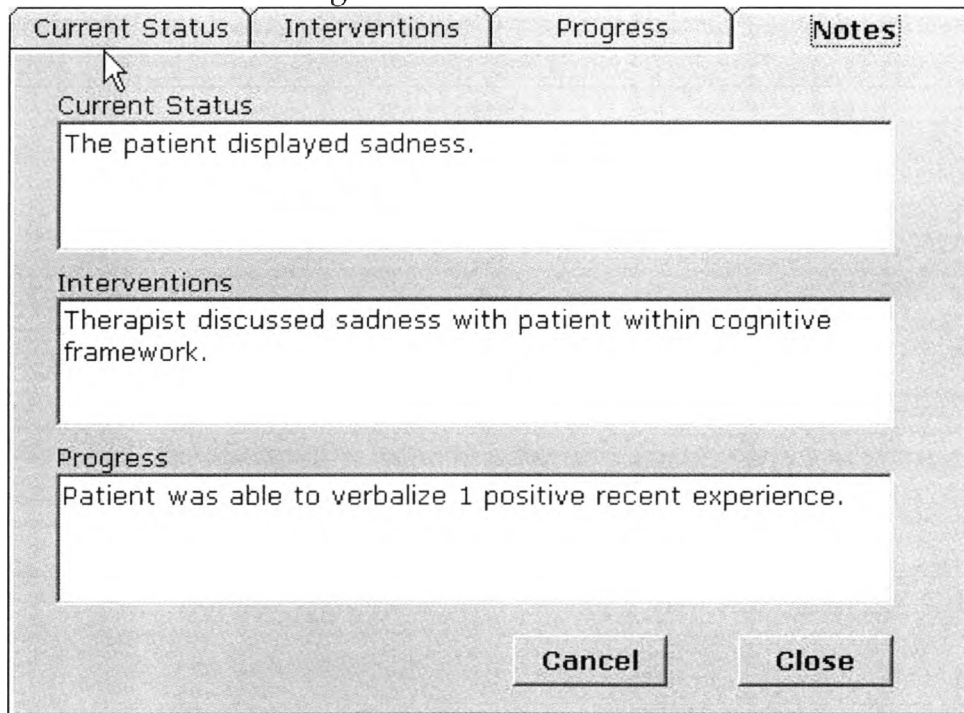
Selected Problems/Symptoms:

Available Problems/Symptoms:

- The patient displayed sadness.
- The patient was not wearing a jacket in cold weather.
- Nursing staff reported patient was argumentative during persc

Custom Problem/Symptom:

Figure 24. UI Notes Screen



The UI Notes Screen features a tabbed interface with four tabs: 'Current Status', 'Interventions', 'Progress', and 'Notes'. The 'Notes' tab is selected, indicated by a mouse cursor. Below the tabs, there are three text input areas: 'Current Status' containing 'The patient displayed sadness.', 'Interventions' containing 'Therapist discussed sadness with patient within cognitive framework.', and 'Progress' containing 'Patient was able to verbalize 1 positive recent experience.'. At the bottom right, there are 'Cancel' and 'Close' buttons.

Current Status | Interventions | Progress | **Notes**

Current Status

The patient displayed sadness.

Interventions

Therapist discussed sadness with patient within cognitive framework.

Progress

Patient was able to verbalize 1 positive recent experience.

Cancel Close

They are not used as directives for specific implementations. In the case of the sentence interface, it is important to note that the screen caches nothing. All changes are immediate and permanent. This has the advantage of being able to reduce the number of command widgets (less buttons and menus to edit, or save, or create), but the disadvantage that errors committed by the user may not be recalled. There is no ability to reset data to a previous state unless the data are intentionally manipulated by the user to make it so. To make a sentence in the status section, the user either selected a sentence from the “available” list or typed some text in the “custom” box. The sentence would then appear in the topmost “Selected” box. To edit sentences in the “Selected” box, functionality would be provided to place the sentence in the textbox for editing if it were custom text or deleted. Changes to any of the three session sections were immediately reflected in the “Notes” tab.

Phase One Evaluation: This interface was designed prior to the sentence engine frame design. It was quickly found to be unusable for reasons that also hampered Therascribe and Quicdoc. Complete pre-written sentences have usefulness in only specific situations. Lists long enough to contain enough sentences to affect a majority of situations encountered by therapists would be impractical to navigate. There is no sound basis for sorting a list of pre-written sentences. Alphabetically is pointless, given that “Patient” is the subject of over

half of all sentences. Depending on how the sentence is written, a right scroll might be needed just to see the content of the sentence. If the user finds a sentence that is partially applicable, how much time is spent considering if the sentence is close enough or if another sentence should be selected? The design was inherently maintenance intensive and sentences too specific to use in multiple instances. Phase one was not shown to the user representative for evaluation, given that its weaknesses were discovered early and were not remediable through implementation.

Phase Two: Phase two design was greatly influenced by the “frame sentence engine design”. Phase two design goals were set as: a) implement the sentence frame engine design; b) keep the form count to a minimum; c) make the UI as compact as possible. Goals a) and b) were straightforward in origin. Goal c) was established to satisfy a user request that the application be viewable by a screen set to VGA (640x480) resolution.

The Phase two design was heavily influenced by experience in using the tabbed property page control in Phase one. This GUI control offered a simple way to provide multiple pages of content to users without opening any additional operating system windows. In a subtle manner, use of the tabbed interface also provided a means of displaying navigational hierarchical location. A user would know where in the program they were by looking at the selected

tab names. Tab names would also show other content options in the program, in a way acting as an outline of the program's functionality. Figure 25 renders the main screen for the Phase two prototype.

The Patient tab is the current selected tab. Data for the entire tabbed property page control is set by the name selected from the "Patient Name:" dropdown list. The New button on the right sets all the tabs to default values for a new patient. Patient data are demographic in nature and organized in three sub groups contained within frames. Note that there are buttons next to the three date fields – these buttons activate a calendar for users to select dates from. In the "Facility" and "Physician" sections, an Add button is present. This button is used to add a new entity to the section's dropdown list. All data stored about an entity appears in the text boxes inside the frames. Entity data are dynamic and changes are immediately reflected in the database.

To write a session note, the user clicks the Sessions tab. The tab default display is a dropdown list of sessions entered for the patient, and a New button to begin a new session (see Figure 26). Once a session is selected, the Sessions tab is populated with its own tabbed property page and a grouping of buttons.

Figure 27 shows the tab contents after a session has been selected.

Figure 25. UI Phase 2 – Main Screen

SDI PSA Notes

Patient Name: Thompson, Jeffrey **New**

Patient | Diagnosis | Medications | Sessions | Admin

Personal

Last Name: Thompson Acct: testacct
 First Name: Jeffrey Room: 55 DOB: 3/3/1966
 SSN: 467-55-6344 Sex: M MD Ord Date: 3/3/2000
 Ins Code: 666 Consent ☒ Consent Date: 3/11/2000

Facility

Name: Mariner Old Person's Home **Add**
 Number: 67 Fax: 911-555-1212
 Voice: 911-555-1212

Physician

Name: Nick Riviera **Add**
 Voice: Fax:

Help **Consent Form**

Figure 26. Phase 2 UI Sessions Tab (Default)

Diagnosis | Medications | **Sessions** | A

Sessions: **New**

Figure 27. Phase 2 UI Sessions Tab (Nominal)

The session tabbed property page has 4 tabs: mode/date, and one for each of three session document notes sections (Symptoms/Interventions/Progress). At the bottom of the Session's tabbed page are six buttons whose actions are specific to the session UI. The button functionality is summarized below:

Help	Opens the help system
Preview	Displays the ready to print session note
Print	Displays and prints the session note
Freeze	Locks out changes to the session note ¹⁶
Cancel	Cancel all changes since opening the tab page
Save	Write the current sentence values to the database.

¹⁶ Added as a result of general concern to therapist that session records may be changed in the database after a hardcopy has been left in the patient chart, thereby creating a discrepancy.

Within the section tab is the access to the sentence frame interface. The top two controls add either a new frame (or pre-existing custom sentence) or a new manual (custom) sentence to the section. The list box below shows the current sentences for the given section. In the example figure, there are two current sentences in the Symptoms section. Finally, the Delete button removes an existing sentence completely. The Edit button places the selected sentence in the appropriate form for editing (automated frame or manual).

Add a sentence: Adding a custom sentence is simple. Clicking the Custom button produces a form where the sentence can be typed in. On accepting the sentence, it is written to the current section, and to the dropdown list for possible use in later session notes. Adding an automated sentence involves a few more steps. First, an appropriate frame is selected from the drop down list. General frame descriptions appear in the list, not the actual frames themselves. A sample of the first eight entries in the Symptoms dropdown list shows a few custom sentences followed by frame candidates (figure 28). By clicking the highlighted frame relating negative outbursts, the system displays the engine interface where the frame specific values can be set.

Figure 28. Phase 2 Sentence Frame Dropdown List Options

The screenshot shows a software window with four tabs: **Symptoms**, **Interventions**, **Progress**, and **Mode/Date**. The **Symptoms** tab is selected. A dropdown menu is open, displaying a list of sentence options. The options are:

- Testing adding a period to a custom sentence.
- What if a period exists.
- This is the first pass changed twice.
- This is the second go round.
- Continued Therapy ed.
- This is yet another custom sentence that is going to go on f
- Demonstrated anger
- Negative outbursts reported

To the right of the dropdown menu is a button labeled **Custom**.

Figure 29. Phase 2 Sentence Frame UI

The screenshot shows a window titled **Complete the sentence....**. Inside, there is a section labeled **Current Sentence Text:** with a text area containing the sentence: "[TARGET] reports patient reported to have had [AMOUNT] negative outbursts within past [DURATION]".

Below this is a section labeled **Token Values** with three rows:

Token	Value	Action
Target	[Dropdown Menu]	Custom
Amount	[Dropdown Menu]	Custom
Duration	[Dropdown Menu]	Custom

At the bottom of the window are two buttons: **Cancel** and **Save**.

Bracketed terms in the sentence frame match the dropdown lists provided in the “Token Values” section. Each token must be replaced with a value. Users may select a value from the list, or add a custom value to the list. User selections replace the token markers in the sentence, thus producing the end result. The following sequence of figures illustrates the process step by step.

Figure 30-34: Phase 2 Sentence UI Creation Sequence

Figure 30. Select Target

Complete the sentence....

Current Sentence Text:

[TARGET] reports patient reported to have had [AMOUNT] negative outbursts within past [DURATION]

Token Values

Target	<input type="text"/>	<input type="button" value="Custom"/>
Amount	roommate	<input type="button" value="Custom"/>
	staff	
Duration	family	<input type="button" value="Custom"/>
	spouse	
	son	
	daughter	
	children	
	grandchildren	

Figure 31. Select Amount

Complete the sentence....

Current Sentence Text:

Staff reports patient reported to have had [AMOUNT] negative outbursts within past [DURATION].

Token Values

Target	staff	<input type="button" value="Custom"/>
Amount	<input type="text"/>	<input type="button" value="Custom"/>
Duration	1	<input type="button" value="Custom"/>
	2	
	3	
	4	
	5	
	6	
	12	
	TestAmount	

Figure 32. Select Duration

Complete the sentence....

Current Sentence Text:

Staff reports patient reported to have had 2 negative outbursts within past [DURATION].

Token Values

Target	staff	Custom
Amount	2	Custom
Duration	<div> 1 week 2 weeks 3 weeks 4 weeks 1 session 2 sessions 3 sessions 4 sessions </div>	Custom

Cancel Save

Figure 33. Final Sentence

Complete the sentence....

Current Sentence Text:

Staff reports patient reported to have had 2 negative outbursts within past 2 weeks.

Token Values

Target	staff	Custom
Amount	2	Custom
Duration	2 weeks	Custom

Cancel Save

With only 3 selections, a quantifiably scaled sentence pertaining to a patient's negative emotional outbursts has been created – clear, perfectly legible, and free of correction markings. This process would be repeated for the Interventions and Progress sections in order to create the entire session note document.

Phase Two Evaluation: The design was shown to the therapist representative. The system was fairly well received, but the representative's overall assessment of the system followed the assessment of the sentence frame architecture. Frames required that all tokens be utilized. Frames were static structures that did not often match the needs of the situation and, rather than spend time looking for a frame that was adequate, a user would turn to writing all sentences manually negating most benefits of documentation automation. The user also commented that the screen was "too busy", that the "information was too crowded" and that there were "too many words to read". The user said that having multiple windows pop up in the application was distracting – it would be "nice" if there was one screen that the user had to look at and no pop up windows or modal screens that overlay and block viewing of data beneath the current screen (Durrett, 29 Mar 2000). These comments were taken and considered when designing phase three.

Phase Three: Element based sentences finally offered the promise of dynamic sentences that could be reused in varying contexts. For version three,

changes were made to the basic UI interface based on therapist critique of version 2, in addition to sweeping redesign of the sentence interface system. The observation was made that the therapist wanted to see certain data regarding a patient at all times during note creation. In addition, a new company memo had been distributed to therapists stating that the frequency of sessions being conducted after the prescribed number of sessions had been exhausted was too high. The therapist made a specific request that session and consent status be visible at all times in addition to the generally negative statements regarding automated sentences (“I hardly got to use any of the sentences. After 3-4 notes, I quit trying and typed all the information in the boxes myself – faster that way” (Durrett, 18 Sep 2000).

Phase three expanded on the common website navigational layout of version one in that the action buttons on the left side of the interface were replaced with richer controls. Data and action were now in the navigational column. This would act to satisfy the desire that patient data be visible at all times. Version three also borrowed from website design the idea of a horizontal panel at the top of the interface that would display data in manner similar to a bulletin board. The base UI design implemented the request that data manipulation would primarily occur in the main panel bordered by the navigation and information panels. Figure 34 illustrates the version three UI

main screen with a patient selected and the patient data screen in the main data panel.

Figure 34. Phase 3 Main Screen

NoteMaker

Patient
 Joe Blow
 New Edit Delete

Status
 Consent Days Remaining: 6 Sessions Entered/Total: 1 / 20

Sessions

Date:	Status:
07/30/2000	Open

Note New Edit Delete

Software
 Help Admin Exit

Patient Data

Personal Diagnosis Consent Medications

First Name: Joe
 Last Name: Blow
 SPS Account: 1
 Insurance Code: 1
 Social Security: 1
 Date Of Birth: 04/04/1920
 Facility: San Antonio Elderly Rest Home
 Room: 1
 MD: Nick Riviera
 MD Orders Date: 07/20/2000
 Sex: ☒ Male ☐ Female

Action buttons were specific to certain data sets as indicated by the new, edit, delete button groups in both “Patient” and “Sessions” frames. A “Software” section was added to cover all program related issues. Depending on whether an existing item or a “new” button was activated, the appropriate data interface would appear in the main panel area. In figure 34, a patient has been selected from the list, and the patient data tabbed property page interface occupies the main data panel. Note that the “Sessions” frame has also been

populated with existing sessions for the selected user. Above the patient data tabbed interface is the special data display frame. This UI feature is the implementation of the previous request to display session status and consent information to the user at all times. In consultation with the user, session status was determined to be presentable by showing: a) number of sessions used and remaining; b) number of consent days remaining for the patient.¹⁷

The tabbed patient data interface was not greatly different than from version two. Consent information was added, but the methods of entry and editing did not change. On the Patient data tab shown in figure 34, the labels for data fields “Facility” and “Physician” are buttons rather than static text strings. This feature was added to permit users to have a rich UI for entering and updating information on these two areas. As noted in Zarit, integration with the facility staff and presiding physician is highly desirable for psychotherapeutic success (Zarit, 320). Figure 35 provides a sample of the UI. Shown is the facility data form. The physician data form is very similar in design and capacity.

The sentence UI for phase three implemented the sentence engine element design. The sentence UI in phase three was organized like phase two in that access to the sentence UI form was through an intermediary form [Iform]. This

¹⁷ At the time of version 3 design, patient consent to treatment was required every 30 days. A consent form signed by the patient for a specific span of days was required before psychotherapy could be administered. This company requirement had been placed in effect as a “precautionary safeguard to ensure compliance with HCFA medical necessity requirements”, according to the company therapist interviewed. (Durrett 1999b).

Iform appeared in the main data panel on new session creation or session edits.

Four session aspects were displayed in the Iform, one dynamic and three static.

Figure 35. Phase 3 Facility Data UI Form

Facility Details

Edit facility data or click "New" to add a facility...

Facility Name: San Antonio Elderly Rest Home

Number: 12345

Contact Name:

Voice: - - x 231231234

Fax: - - x

Pager: - - x

New Save Delete Close

The dynamic aspect (sessions) on the Iform was the session header information. Here, the user could directly manipulate session date and modality values. The remaining aspects (symptoms, interventions, progress) were static read-only compilations of all sentences for the aspect viewed. An “edit” button on the aspects fired the sentence UI form where sentences could be generated, edited, deleted, or manual (custom) sentences entered. In figure 36, the session has been selected in the sessions list box, and the symptoms section tab is currently active. Figure 37 shows the content of the sessions aspect.

Figure 36. Phase 3 UI Session Interface

NoteMaker

Patient
 Joe Blow
 New Edit Delete

Status
 Consent Days Remaining: **6** Sessions Entered/Total: **1 / 20**

Sessions
 Date: Status:
 07/30/2000 Open
 Note New Edit Delete

Software
 Help Admin Exit

Session Data
 Session Symptoms Intervention Progress

Symptom Comments:
 Daughter noted that Patient was crying excessively about being placed in the facility. Patient was last visited by family by 3 months ago.
 Edit

Figure 37. Phase 3 UI Session Interface (Session Tab)

Session Symptoms Intervention Progress

Modality
☐ Individual 20-30 min
☒ Individual 45-50 min
☐ Group
☐ Family

Session Date:
 07/17/2000

Close Session

The “Session” aspect includes a “Close Session” button that removes the session UI form from the data panel.

Phase one session data was closely coupled to the database – changes to data in the UI were immediately reflected in the corresponding database records. In phase three, this was modified for the sentence UI form. Data manipulation in the sentence UI form is batched. The user has the opportunity to save (commit) all edits done or cancel all changes and revert to the values that were in place when the sentence UI form was last loaded up.

The sentence UI form was the most sweeping change in phase three. The UI introduced the new sentence construction concept of elements. Figure 38 illustrates the sentence UI form for the symptoms section illustrated in figure 36. The first sentence has been selected, and the appropriate values for the required elements have been populated. The sentence UI form appears as a separate modal form above the main application form.¹⁸

Figure 38 illustrates the important features of the UI implementation of the element sentence methodology. Starting at the window-top, the UI form displays each individual sentence for the given aspect. Selecting a sentence from the list will set the automated sentence controls to the appropriate values should the sentence be of automated type. Custom sentences produce a new window containing a textbox pre-filled with the sentence text.

¹⁸ In Microsoft terms, the application is an SDI (Single Document Interface) type, where a form is represented in its own window. This form is modal, meaning the user cannot activate any other form until this window is closed (via Save or Cancel buttons)

Figure 38. Phase 3 Sentence UI Form

Symptoms Section

Current Symptom Notes:

Daughter saw Patient was observed crying excessively of being placed in the facility.
 Patient was last visited by family by 3 months ago.

Source: Daughter

Action: crying excessively

Target: being placed in the facility

Rate:

Duration:

Sentences

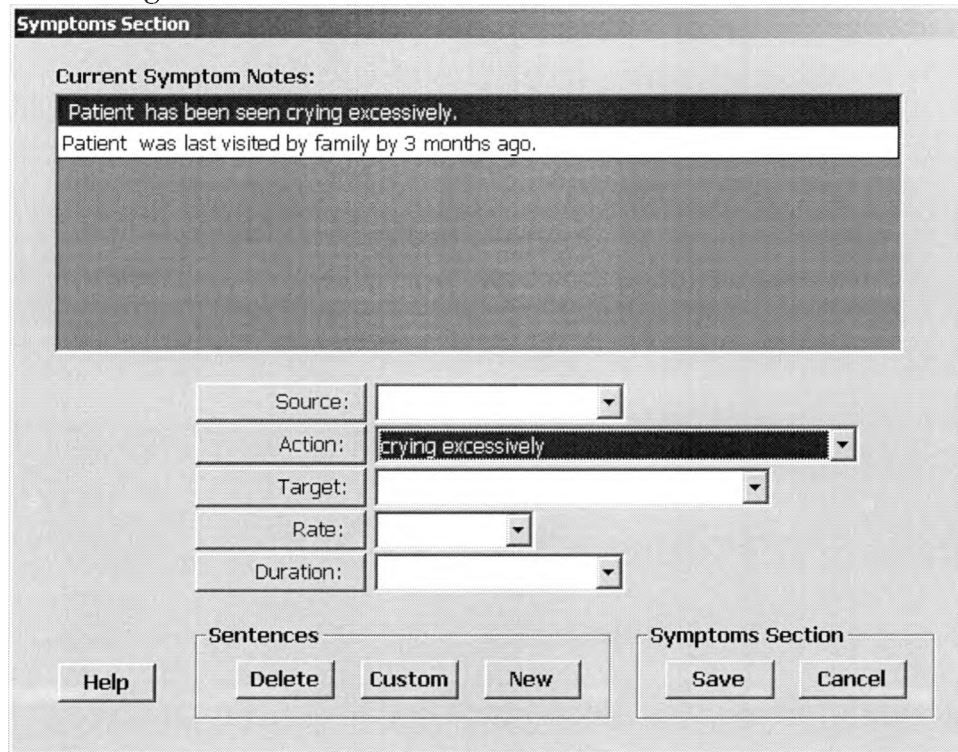
Help Delete Custom New

Symptoms Section

Save Cancel

The five elements in the middle of figure 38 are at the core of the sentence element model's strength. An automated sentence may consist of an action plus any other mixture of the four remaining elements. The sentence highlighted in figure 38 is composed of three elements (source, action, target), but note how the sentence changes when the number of elements set decreases (see figure 39) or increases (see figure 40). Sentences now had a variable, cumulative nature that could be used effectively by a therapist to create sentences for multiple situations and contexts.

Figure 39. Phase 3 Sentence UI Form – Less Elements



Symptoms Section

Current Symptom Notes:

Patient has been seen crying excessively.

Patient was last visited by family by 3 months ago.

Source:

Action:

Target:

Rate:

Duration:

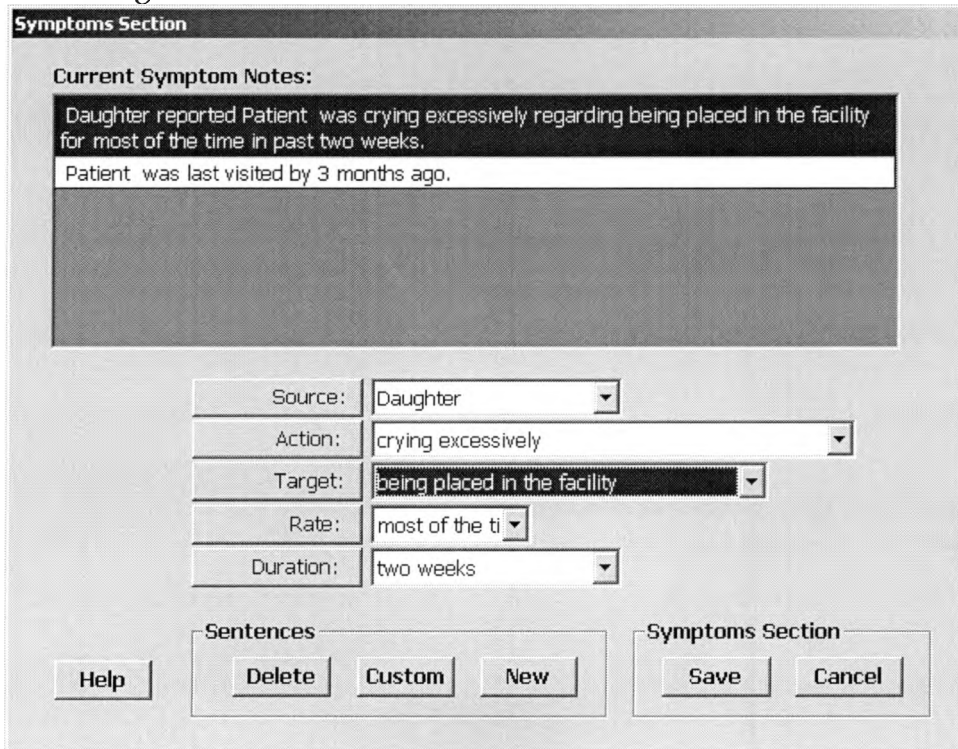
Sentences

Help Delete Custom New

Symptoms Section

Save Cancel

Figure 40. Phase 3 Sentence UI Form – More Elements



Symptoms Section

Current Symptom Notes:

Daughter reported Patient was crying excessively regarding being placed in the facility for most of the time in past two weeks.

Patient was last visited by 3 months ago.

Source:

Action:

Target:

Rate:

Duration:

Sentences

Help Delete Custom New

Symptoms Section

Save Cancel

Phase Three Evaluation: Phase three was a vast improvement over phase two in terms of sentences. However, certain issues remained open. In contrast to the user's request from phase two reviews, the sentence screen was still a separate window overlaying the main application. The therapist representative found the element usage "cumbersome" in that options for the elements were too specific themselves in certain areas. For example, the rate element is specific to the quantity and units of rate. If the user doesn't find their rate in the list, the rate has to be typed in and added. If there are so many rates as to force the user to scroll more than three screens [24 options] worth of data, then the user stated they became frustrated, assumed the value was not in the list and added it to the list, even though the desired rate might already be in the list, but just too far down to expect users to find it before giving up. The user representative concluded by stating the system was "OK, but just not right", and when pressed for what was "not right", the user could offer only the same critique as the previous two iterations: "too cluttered", "frustrating at times", and "too busy" (Durrett, 29 Mar 2000).

Phase Four: Iteration four of the GUI design was not quickly engaged. Three iterations had produced some success but overall failure in meeting the user's requirements. Of the options to either refine and rework the existing design, or trash it and come up with a new approach, the latter was chosen. It

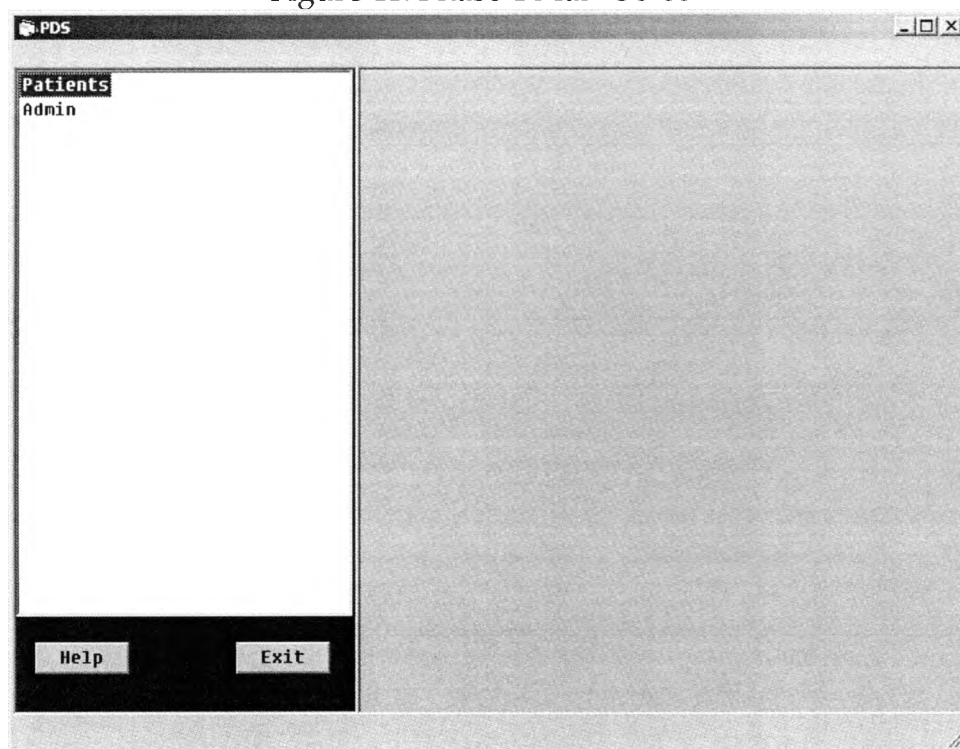
was apparent that the ability of the user to accurately define the exact characteristics he desired in a system was not well developed. Further efforts in a design a user already had a unfavorable opinion of would most likely result in enhancing that negative bias, rather than cause a complete change of heart towards a design that is only slightly altered from a previous manifestation. Besides the argument of preconceived notions, the designs submitted so far were not reaching the “easy to use” goal. It was time to consider alternative possibilities.

The genesis of the phase four design began after the realization surfaced that all the data involved in the session documentation process is hierarchically linked to each other. There is an inherent tree like logical association between patient and sessions. The Windows operating system displays such parent-child relationships via treeviews. This GUI control is seen prominently in Windows Explorer when a user navigates disk folders. By viewing a single folder node, a user is able to also collect a good deal of information as to the location of the folder in the disk, its depth from the root directory level, and what other similar folders are available (assuming folders are grouped for content, or by name). Single clicks within the Explorer treeview open windows and display file data. A treeview scheme applied to patient and session data objects might provide the single screen experience the user requested as well as a simple interface that

provides the most direct access to related data objects possible. Phase four design began with the treeview control as the cornerstone of the design architecture.

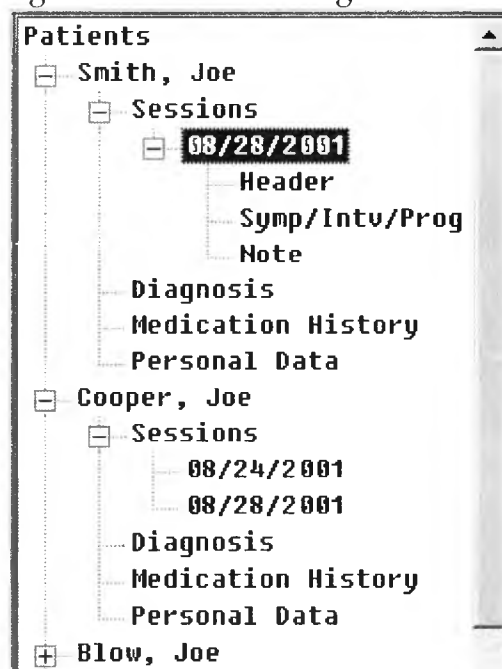
Figure 41 illustrates the main screen of the phase four design. It is divided into two sections, navigation on the left, data interface on the right. Root nodes of the left-side tree mark data interface screens. All higher-level nodes represent logical sub groups of data and do not have data interface screens. They exist to sort, compartmentalize, and organize data. At the bottom of the navigation treeview control are two buttons: help, which launches the help files, and exit, a redundant but unmistakable means of shutting down the program.

Figure 41. Phase 4 Main Screen



To view the list of patients in the system, the “Patients” node is clicked. All patients in the system appear as second level root nodes. Expanding a patient node reveals the first leaf node data screens (Diagnosis, Medication History, and Personal Data), and the “Sessions” third level root node. Expansion of the Sessions root node displays all existing session nodes by date. As shown in figure 42, from one control, without changing screens or perspectives, the user can garner information about whether the patient is new or established, what patients have been seen in the last few days, and what patients haven’t been seen in too long a time. Used in this fashion, the treeview control is able to generate practice metadata without extra effort or resources to do so.

Figure 42. Phase 4 Navigation Nodes

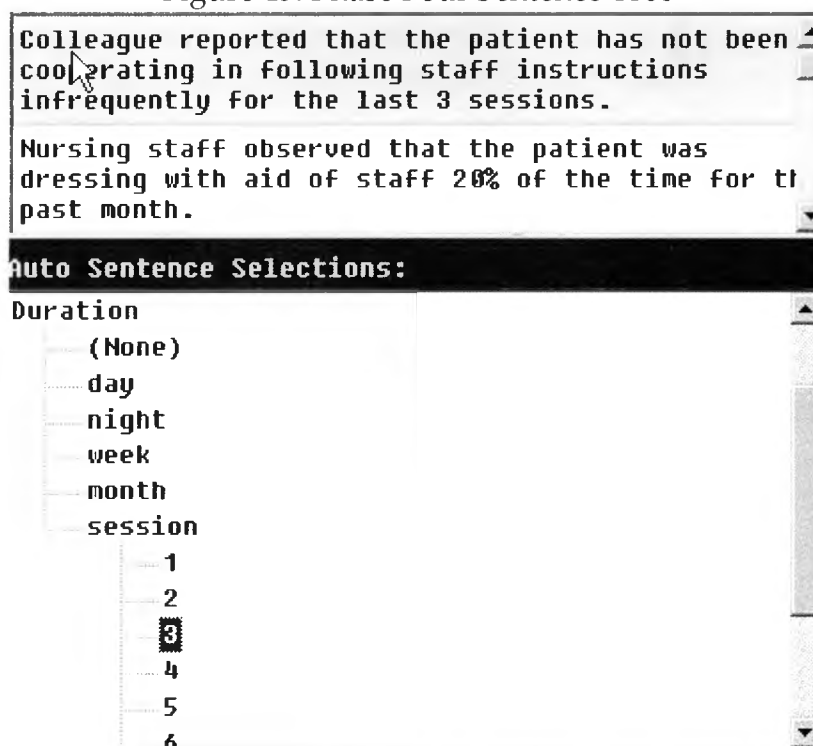


Session date is itself a fourth level root node. Expansion of the date node selected provides access to session leaf nodes (Header, Symp/Intv/Prog, Note). Header data are the same as phase two and three session header data. Symp/Intv/Prog is a single screen access to all the sentences in the session note. Each of the three sections share a common interface such that sentence creation is the same process for all three sections. Note is a node that prepares a mockup of the session note form for viewing and/or printing.

Phase four sentence generation: Phase four continues to use the sentence element engine design. The UI has been changed to resemble the same UI used to navigate the application. Rather than use multiple text controls to represent those data elements that are user determined, and require a large amount of screen space, phase four uses a treeview to represent each of the possible selections for a sentence data element as a leaf node. This has certain advantages in that the data can now be organized logically without any impact to the data itself. For example, in phase three, the rate options were very specific (1/day, 2/day, 1 time, 2 times, etc.). These rates are really communicating a scalar amount (1-x) and a unit (day, week, night, hour, etc.). Each unique combination of amount and unit would require one entry in the options list. The length of a thorough list becomes prohibitive. If the units and amounts can be grouped, and a parent-child relationship is present to make the logical connection, the list becomes far more

manageable. Figure 43 illustrates how the treeviews root node grouping helps to offer a rich set of possible selections in a short amount of space. In figure 43 the yellow background sentence is the current sentence. In the space of 15 lines the user is able to create 50 different duration phrases, each of which is syntactically correct for whatever sentence the user creates (by means of the sentence engine phrasing constructor). The selection made in this example is three, under session node. The sentence is already updated stating "...for the last 3 sessions." Frequency is another node that benefits from this use of sub grouping.

Figure 43. Phase Four Sentence Tree



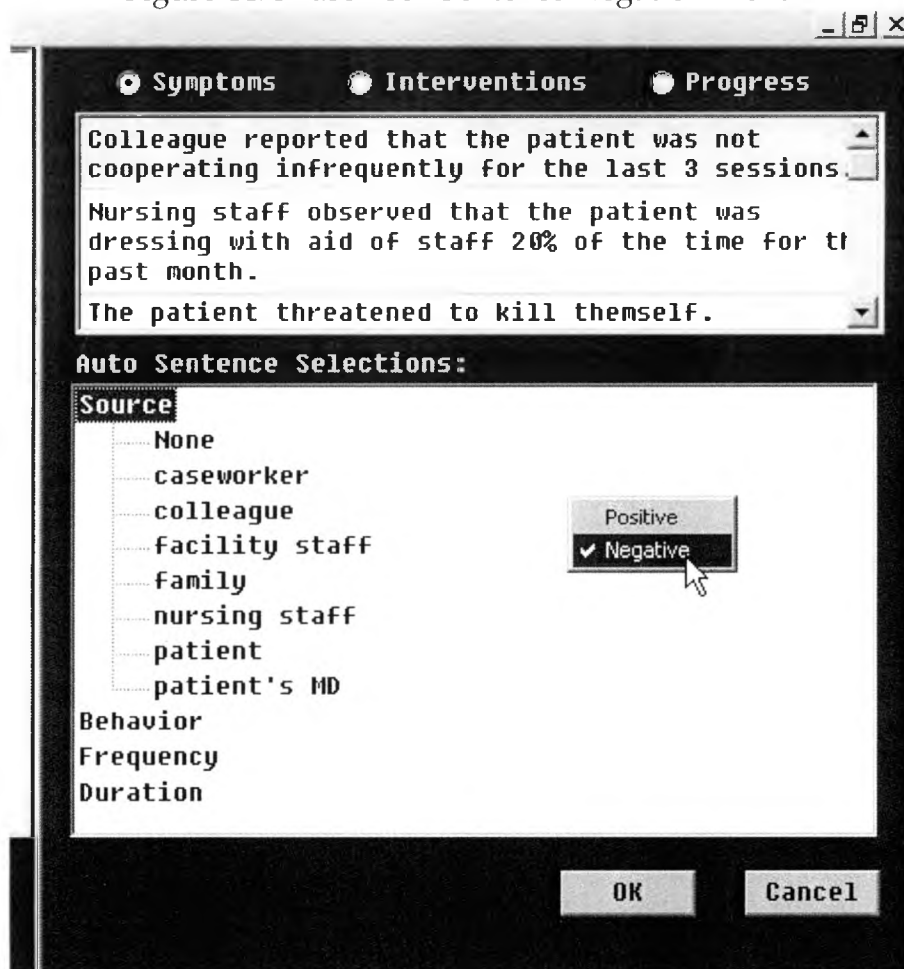
To display a list of 24 distinct options, three full screens were required. By sub-

grouping frequency into three groups, the longest list is only 10 entries. Users need not go over 24 entries to find the one they want. Knowing the sub-groups, they can narrow the list down substantially at the beginning of the search. The remaining node structure also takes advantage of the data arrangement possible via root nodes. For example, the symptoms tree has four root nodes: Source, Behavior, Frequency, and Duration. Source contains all leaf nodes beneath it. Behavior has a number of sub root nodes. These sub root nodes specify a general behavior. It is possible to write a sentence specifying only a general behavior. If more detailed context is desired, each behavior node can be clicked to show all the possible detail options available. The list never gets longer than the longest list of detail possibilities. Should a detail list get too long, it is likely that another sub-root node level could be introduced to subgroup the detail options to make the list more accessible. Sub-root nodes make it possible to organize the display of information solely on the basis of user requirements.

Negation: In reviewing the existing software systems, it appeared that there was a distinct slant towards sentences that affirmed a patient behavior or action. Most sentences said what, or how often, or how much the patient did of something. In reviewing historical session notes, sentences might say a patient was cooperating with staff part of the time, while another patient's notes would say the patient wasn't cooperating with staff. Essentially the same sentence is

used in both contexts; one is the negation of the other. In this case, the negation of a sentence provides a new context for what is essentially the same grammar construct. Sentences produced by the sentence engine might be able to double their effectiveness if a negation method could be applied. Application of negation was relatively simple – in the Symptom section we are using as an example, a simple one or two word negation phrase is attached to the behavior verb of the sentence.

Figure 44. Phase Four Sentence Negation Menu



Now, by adding negation where appropriate, sentences communicate a sense of trend and increased context in a sentence. Since negation exists, sentences that do not have the negative phrase still benefit from the increased contextual value in that all non-negative sentences are positive.¹⁹ Negation is implemented via a popup menu option accessible by right mouse clicking in the sentence treeview (manual sentences do not fire the menu action). An example of the popup negation menu is shown in figure 44.

The remaining functional features of the sentence interface include the sentence list, the OK and Cancel buttons, and the section radio selectors. Along the top of the data screen are three radio option buttons. Through these radio buttons the content of the sentence treeview box is changed to suit the notes section selected. Figure 44 indicates the symptoms section is selected, so the treeview contains the source, behavior, frequency, and duration root nodes. The sentence list box is a multi-function component. Besides displaying sentence text values, custom sentences have their text entered directly into the list cell. This eliminated the need for a separate window to collect and edit custom, user-entered text. The current sentence has a yellow background to easily designate its status. Finally, the OK button closes the Symp/Intv/Prog form and writes the current values to the database. The Cancel button closes the form and abandons

¹⁹ Or affirmative, to be more accurate. Positive implies a desired, wanted outcome that the context may not support

all changes since the form was opened. Like version three, this version was able to rollback data states to a particular point in time.

Hardcopy output: This problem has been addressed in different ways in the other software packages. Quicdoc wrote their own full-fledged word processor (Quicword) just to generate their reports. Therascibe appears to take advantage of Microsoft Access reporting capabilities to generate hard copy reports. For this prototype, it was imperative to have a document format that was portable, supported by all operating systems, and able to recreate the company session note document somewhat closely to the original. Rather than lock into an application technology that may be unsupported in a few years (like the proprietary Quicword or Microsoft specific Access reports), the session note documents were developed as HTML documents. Using HTML for the session note means that any browser can display and print a session note. Portability is ensured to the extent that any internet browser can render the note form. HTML also permits enough formatting techniques to generate a faithful recreation of the company session note document. In addition, the lifespan of HTML appears to be set for the foreseeable future. Any computer with an internet browser and access to a printer can print a session note.

Phase Four Evaluation: Testers were given roughly the same amount of time to familiarize themselves with the program as had been given for the

Therascribe and Quicdoc evaluations. Testers completed 10 session notes, manually and programmatically, and the timing results (rounded to the nearest minute) were evaluated. General functional impressions were also documented for comparative analysis.

Evaluation of the phase four design was very positive. The test users were able to pick up on the tree structure interface quickly and began to successfully use it almost immediately. Consensus amongst the two testers was reached that the prototype was fit for development and ready to be tested in more exhaustive, real-world situations. Comments from the user representative tester included “easy to find stuff”, “comfortable”, “intuitive”, and, as in the previous testing sessions, “still too many mouse clicks”. In consideration of the mouse click count, there is no immediately apparent resolution for this issue. In the next step of broader testing, this issue would be presented to the larger testing body for consideration and comment. It may be that the concern for mouse clicks is a concern localized to the one tester only.

With the tacit approval of the users in hand, the UI was ready for technical evaluation. To this point in the design, functionality above all else was the key aspect to model in the prototype. A refined UI design takes the base, functional correct design accepted by the user and applies accepted UI guidelines to it in order to complete the design and ensure a level of performance that exceeds the

base requirements. In evaluating the design UI for completeness and any problematic issues, two analyses were done. First, a generalized checklist of sound UI design practices was applied to the design to spot any subtle deficiencies. Secondly, in consideration that Microsoft Windows would be the most likely operating system on which the design would run, the design was reviewed against Microsoft's user experience guidelines text in order to spot any consistency deficiencies with standard Microsoft Windows application expectations.

UI Checklist: Pressman provides a list of "common GUI errors" found in windows applications (Moseley). The table below is the results of applying the checklist to the prototype system. The checklist issue is on the right side; the left side is the response of the prototype to the issue. Also in the left column are observations regarding the issue – these issues are guidelines, not unbreakable rules. There may be good cause to omit or alter the expected response to an issue.

Table 9. Prototype GUI Design Checklist Results

#	Issue	Assessment
1	Assure startup icon is unique	To be addressed in development phase
2	Assure control menu in window/dialog box	No menus in prototype. To be added.
3	Assure MDI of each window – Child windows must appear within parent	Confirmed
4	Assure all windows have consistent look and feel	Confirmed by way of orientations, reuse of screen design, and reuse of button names
5	Assure all dialog boxes have consistent look and feel	See 4
6	Assure that child windows may be cascaded or tiled in parent window	Child windows can be minimized, maximized, or any size in between. No

		specific cascade/tile function in place yet
7	Assure that icons representing minimized child windows are arrangable within the parent window	Done
8	Assure File menu exists	To be added
9	Assure Help menu exists	To be added
10	Assure Window menu exists	To be added
11	Assure other menus logically required by the application exist	To be determined and added if needed. This application concentrates on mouse driven operation.
12	Assure proper commands and options are in each menu	To be done
13	Assure tool bar buttons have corresponding menu commands	No tool bar buttons in application
14	Assure menu commands have hot key sequences for keyboard invocation	To be added
15	Assure tab dialog tab names are not abbreviations	Done
16	Assure tab dialog tabs are accessible via hot key sequence	To be done
17	Assure tab dialogs do not have duplicate hot keys	See 16
18	Assure tabs are placed horizontal along top	Done
19	Assure usage of ESC key to roll back changes made	Not implemented. Certain areas of program are immediate updates to DB. Plan to query users for necessity of this.
20	Assure CANCEL button functions same as ESC key	See 19
21	Assure CANCEL button becomes CLOSE when changes can't be rolled back	To be done
22	Assure command buttons used by current window are only buttons present	Questionable. Parent window help and exit buttons are available at all times, but are not explicitly used by any child window.
23	Grey out command buttons when they should not be used	Done
24	Assure that OK and CANCEL are grouped separate from other command buttons	Done in the sense that OK and CANCEL are always next to each other, on the right most edge
25	Assure that command button names are not abbreviations	Done
26	Assure command button names are meaningful to users	Done
27	Assure command buttons are of similar shape and size	Done
28	Assure command buttons can be accessed via hot key sequence (except for OK and CANCEL – not required)	None of the current buttons have this capability. To be added.

29	Assure command buttons in the same window do not duplicate hot keys	See 28
30	Assure each window has clearly defined default value invoked when ENTER key is pressed	Questionable. The control having focus in the window determines what occurs when ENTER is pressed. When child windows are activated, focus is set to a particular control.
31	Assure program object focus makes sense	Done
32	Assure option labels are not abbreviations	Questionable. Is it better to write out "Individual, 50 minutes" or "Individual, 50 min". Current company session forms use the abbreviated minutes notation. The abbreviation is used in the modality section of the session header.
33	Assure option names are not technical labels	Done
34	If present, assure option hot key sequences are not duplicated	To be added
35	Assure option box names are not abbreviations	Not applicable – not used in GUI
36	Assure option box names are meaningful to users	See 35
37	If present, assure option box hot key sequences are not duplicated	See 35
38	Assure option and radio buttons are logically grouped in clearly demarcated areas	Questionable. Modality option buttons are grouped logically and spatially, but without any specific borderlines.
39	Assure demarcated areas have non-abbreviated user-meaningful names	Done
40	Assure TAB key sequence is logical	Done
41	Assure parent window has status bar	Questionable. The degree of status message that is appropriate to display to the user is debatable. Error messages require more notice than a small left corner note. Error handling in a system used by technical novices should be explicit.
42	Assure all user-related system messages are presented via the status bar	See 41. This is not sound in my professional experience. The status bar is often neglected by users. The expectation is (and has been) that when an error occurs, messages regarding it are clear, concise, and most importantly noticeable.
43	Assure consistency of mouse actions across windows	Done
44	Assure color RED is not used to highlight active objects	Done

45	Assure user has control of desktop color, Highlighting	Questionable. As written, this seems to refer to an operating system parameter. Very few applications offer the ability to change color schemes at will, unless the application is visually oriented in context. The application will not interfere with the operating system color settings in any way.
46	Assure GUI does not have cluttered appearance	Questionable. Only users can say if a GUI is cluttered or not. According to test results, the GUI is not cluttered.

There were four areas (more than four items) where the prototype system did not meet the checklist expectations. These four areas can be distinguished as menus, keyboard equivalents, abbreviations, and status bars. For each area, there are compelling reasons to deviate from the checklist assumptions.

Menus are required for any software design subjected to this checklist.

The question can be asked “why are menus mandatory?” Detweiler and Omanson published a set of interface design guidelines for Ameritech where the issue of menus arose only in one section (9.2) and was described as an alternative to radio buttons for selecting from a list (Detweiler 1996). As it happens, the document refers to the design of websites, but in our internet-enabled world, the line between web site and system application is rapidly blurring. Indeed, Microsoft intends to merge the two application types into one (called web services) via the technology of .NET. Rather than a requirement, menus should take the posture of other GUI options available to designers – used when appropriate. As mobile computing becomes more prevalent, the concept of

menus will not be a viable option, as screen space will be at a premium.

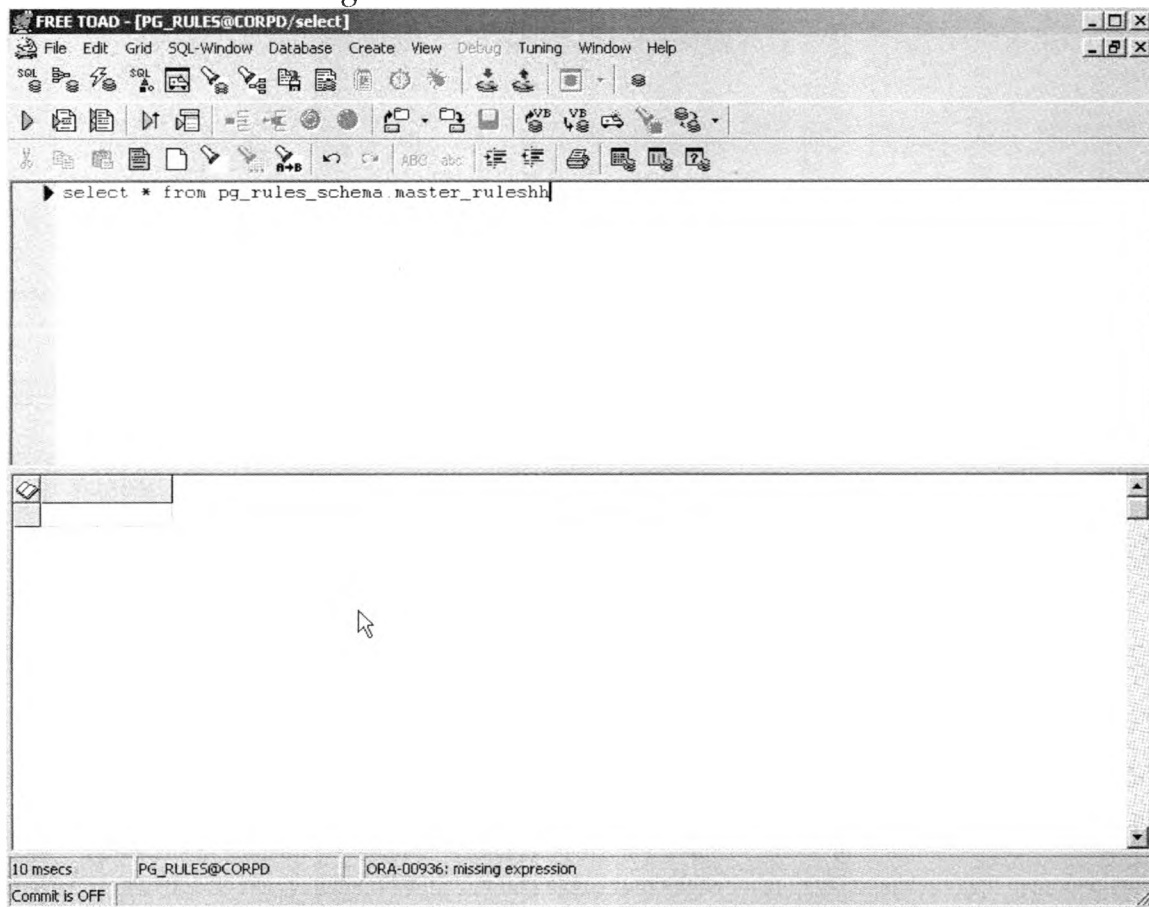
Consideration of the deployment architecture and the user requirements should dictate the use of appropriate control methodologies. No single control methodology should be mandated for universal application.

The argument against keyboard equivalents is similar. Keyboard input enjoys the longest lifespan of any user input device, but it is not appropriate for use in all applications. PDA devices are an excellent example where the idea of an ESC key is not beneficial or efficient in solving the issue of how to rollback changes. In voice driven systems, for example, how would a user voice the usage of an ESC key? Again, the methods of control in a program must fit the technology it is developed to work in and the user expectations it must fulfill.

Abbreviations are often necessary to conduct business in user environments. In the health community, when was "ICD-9" last used in a fully spelled out manner? To attempt to put the fully written ICD-9 name on a form would be ludicrous. The interface conventions for taxonomy must reflect the user's taxonomy. If abbreviations are used in the real world process, then they must be used in the modeled process as well. The software system is a model of a real world process, and as such must not rework aspects of the environment that are not reworked in the real world process. The GUI checklist is wrong to include this item as a mandatory component of "good" GUI design.

The status bar references in the checklist seem not to take into consideration the error handling capabilities of the software system. Any system message important enough to display to the user most likely requires that the user take some action, even if only precautionary in nature. The status bar as implemented in Windows applications is far from noticeable. It resides in the low left corner of the screen in 8pt font that is small and easy to skip over.

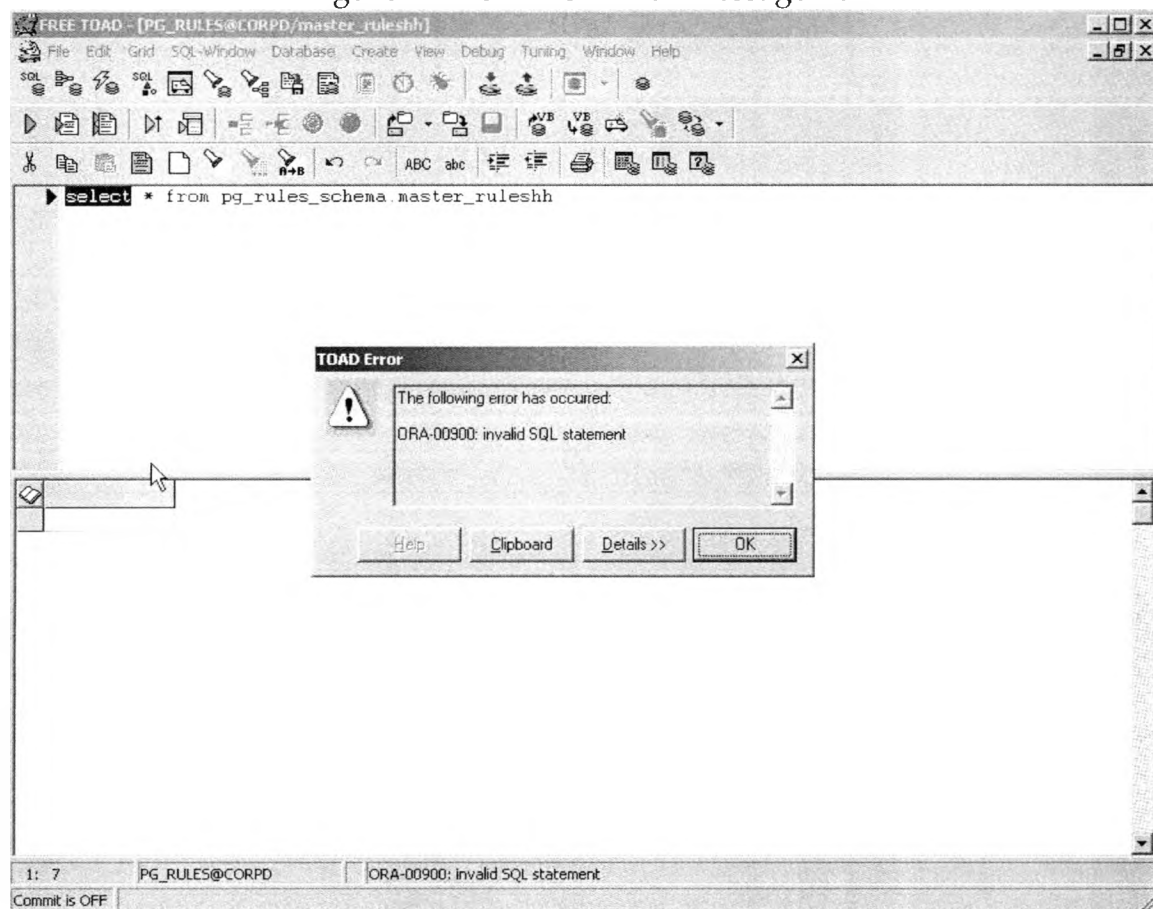
Figure 45 TOAD UI – Status Bar Error



As an example, figure 46 is an illustration of a status bar error message and a programmer defined error message dialog window. Comparison of the messages

shows how the status bar is not the place to put messages that need to be seen by the user. The application in the example is TOAD (Tool for Oracle Application Development), a popular tool used by Oracle application developers.²⁰ Note the ORA-00936 error message. For certain errors, the application detects the error and displays the message in a more appropriate manner:

Figure 46. TOAD UI Error Message Box



The status bar echoes the message in the box, but now the effect is much more helpful to the user. Status bars are fine for supplementary messaging, but to

²⁰ Quest Software, 8001 Irvine Center Dr, Irvine CA, 92618 www.quest.com

make status bars the primary message medium is not a workable option.

Microsoft GUI Guidelines: Microsoft Windows User Experience is a book published by Microsoft that, as the subtitle states, is the “Official Guidelines for User Interface Developers and Designers” (Microsoft 1999). The foreword, written by the “User Experience Team” at Microsoft, states that the intent of the book is provide guidelines to user interface design culled from the development experiences with Microsoft products. The text is actually a manual for how Microsoft Windows operating system controls and menus work, down to what specific menu commands do at a general level. There are very few pages devoted to guidelines applicable to software other than Microsoft Windows or Microsoft products. Over 500 pages are spent in a virtual display of “see, this is how we did it, so you must do it this way too, or you’re wrong.” Microsoft accomplishments are remarkable in the field of software development to be sure. However, in the case of user interface design, running out the Microsoft Windows operating system as an example of the “ideal” user interface is not warranted. In the section called “How To Apply These Guidelines”, the narrow focus of the book is evident when the authors wrote:

...following all of these guidelines is encouraged...adhering to these design guidelines does not guarantee usability... You can extend these guidelines, provided that you do so in the spirit of the principles on which they are based. (Microsoft 1999)

The “spirit” is not defined in the text. It appears that the intent of the Microsoft text is not to produce better user interfaces, but to produce more interfaces that appear and respond just like the Windows operating system. With this as a primary goal, user requirements can easily become secondary in importance rather than the primary status they should have.

The text is weak on assessment techniques for user interface designs when the design is anything but Windows-based. There is a GUI design checklist titled “Checklist for a Good Interface” (Microsoft, 22). This is as universal a set of criteria as the text had and the prototype was compared to the checklist.

Table 10. Microsoft Good Interface Design Checklist

#	Issue	Response
1	Application [App] installs easily with minimal steps	More of a deployment issue – not an application GUI issue
2	App installation doesn’t require system restart	More non GUI related issues
3	Reading a README file before using application is not required	See 2
4	User generated data files are stored in My Documents folder by default	See 2
5	App avoids cryptic file names for files visible to users	See 2
6	App doesn’t create folders outside of Program Files folder	A behavioral issue with specific reference to Windows OS
7	App doesn’t write files to root of hard disk	See 6
8	If using disk cache, app registers with Disk Cleanup utility	Another software configuration issue, not a GUI issue
9	App has no entries to Help, Readme, or Uninstall files on the Start menu	See 6
10	Don’t install icons without user permission	See 6
11	If app is run at startup, no splash screen or dialog box while loading	See 6

12	App uses notification area (status label in taskbar) only to alert users of important changes	This is the closest issue yet to a pure GUI design issue.
13	App applies color choices made by user in Display properties of control panel	First GUI related item
14	App is keyboard accessible	Real GUI issue
15	App works correctly if user increases font size	Related to GUI issue, but truly a program behavior issue
17	App supports standard keyboard shortcut set where applicable	The “where applicable” piece permits flexibility in system integration with keyboard entry
18	App uninstall process leaves no files behind, nor registry entries	Behavioral issue of installation methodology used
19	App has no Jargon in UI text.	Industry specific and technical terms only with user acceptance
20	App adjust appropriately when user changes display resolution	GUI issue. Prototype system designed to work in narrow band of resolutions, but absolute minimum of 640x480 is set as a requirement

Many of the so-called GUI issues in the Microsoft checklist were system behavior issues, under the pretense of being a GUI issue. Far more valuable usability information is gained through the Pressman checklist. The results of the Microsoft checklist are at best inconclusive as to how “good” the prototype GUI is. Still, in considering Pressman’s list, the older, dated paradigm it used to draft questions will diminish the Pressman list’s applicability to a growing segment of application development (internet and TCP enabled applications).

VIII. PROTOTYPE TESTING

Testing of the prototype was broken into four components: Goals, Plan, Configuration, and Results. The overall testing process was modeled after the evaluative testing performed on Therascribe and Quicdoc. In this manner, the performance of the prototype would be more directly comparable to the previous performance of Therascribe and Quicdoc.

The initial component, Goals, was the determination of what system aspects the testing should specifically measure. Test goals should align themselves closely with system requirements, as test goals are the questions to be answered regarding the system's capability. The existing software qualitative and quantitative testing provides an evaluative tool whose scope is centered on the identified system requirements.

The Plan component consists of how to conduct the testing to produce results. This phase was also easy to plan in that the evaluation testing could be repeated for the prototype software. Usage of the evaluation testing methods had the additional benefit of providing a baseline from which to make comparative analyses between the three software pieces.

Configuration refers to a problematic circumstance in regard to the

prototype's setup. Existing software was evaluated out of the box, meaning that there was no preprocessing of the data sets the systems provided to create note content. In the case of the prototype, it is custom software designed specifically for a particular problem domain. In deployment to actual use, it would be able to "preconfigured" with data drawn directly from the company's own data stores. It would be primed for the types of situations that company therapists regularly see. It was therefore decided to preconfigure the prototype's data tables with data that would match the test situations. This decision was made for the following reasons: a) The existing systems automated sentences were completely pre-scripted. Their basic premise is inconsistent with the dynamic sentence engine used by the prototype; b) The testing should indicate a best case scenario in order to show its maximum potential; c) The preconfigured data state would represent the data set a therapist would interact with in the field; and d) The test cases to be documented were drawn from generalizations of issues found in historical session notes – the data would be included in the prototype's preconfiguration whether in the test cases or not. There is an obvious argument against slanting any test for success, but the final arbiter of the validity of the test results is the company. The nature of the preconfiguration would, of course, be fully disclosed.

Results are the raw data and conclusions drawn from the raw data. This

component of testing is actually quite subjective in that interpretation of data is a speculative function. Many factors play into how accurate testing answers a particular question. Again, the validity of the results is left to the discretion of the company after full disclosure of process.

Goals: From the requirements gathered and discussions with company user representatives, the core requirements for the system were agreed upon to be: 1) ease of use; 2) reduce time making notes; and 3) comply with company and regulatory documentation guidelines. Measuring the performance of these three aspects became the testing goals for the prototype.

Plan: With the testing goals established, the test plan was developed. The issue in developing the test plan was to answer the questions “what do we measure?”, “how do we measure?”, and “how do we evaluate collected measurements?”. Each test goal had to answer these three questions. For goal one, user perceptions were the primary source of usability data. User perceptions would be collected via written comments submitted by testers after using the prototype system. To evaluate the comments, the feedback would be qualitatively analyzed for the ratio of positive to negative comments. Goal two measurements were straightforward. Time reduction testing involved measuring the duration of each note created. The timings could be empirically evaluated to the results of Therascribe, Quicdoc, and the manual process average timing

estimates with the anticipated result being shorter durations for the prototype system than the rest of the compared options. Goal three testing would be qualitative in measure. The company would have to determine if the prototype system met their expectations for guideline compliance. The company would need to review the prototype system's functionality and its output artifacts (session note forms) in order to reach a conclusion. Because of the degree of involvement required on the part of the company in this sort of evaluation, and the fact that only the company can answer this test question definitively, test goal three has been indefinitely deferred until such time as the company officially reviews the findings.

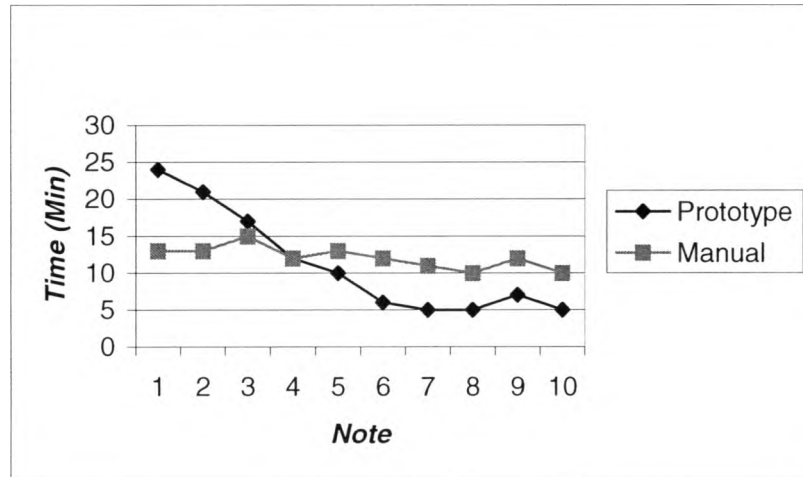
Configuration: Configuration of the prototype system was a debatable issue. Both Therascribe and Quicdoc had been tested essentially right out of the box with no sentence data content enhancements due to design aspects both systems shared. Sentences in both systems were of two possible types: completely pre-written or wholly manual text entered by the user. There was no seed data to provide that affected the outcome of the sentence capabilities of either system. In the prototype, the sentence engine design is dramatically different to that of the software previously evaluated. Since users create dynamic sentences by defining certain elements of the sentence, a pool of element options is a necessary component to the sentence engine. The scope and magnitude of

these option pools could range from very limited to all inclusive. The question to answer was “how much element data shall the test prototype be seeded with?” After consideration of the seeding issues described in the beginning of the chapter, an all-inclusive seeding was done.

In addition, to get an estimate of the amount of therapist workload/time reduction the prototype’s sentence engine can provide, it is necessary that the engine-produced sentences be measured. The assumption was made, based on test experience with Therascribe and Quicdoc, that manually entered sentence timing would be near identical for any of the software systems evaluated. All three entered manual text in the same fashion – keyboard entry into a text box control. Measuring the prototype’s performance in manual sentence mode provides no data of value in assessing the dynamic sentence engine’s design performance characteristics.

Results: Testing timing results raw data are displayed in figure 47. The degree of improvement in generating session notes is promising. Testers were not pre-exposed to the design interface, and the steep curve of the timings indicates that the systems operation is more quickly learned than that of Therascribe and Quicdoc. The sentence timings average better than 50% faster than the manual sentence entries. This timesaving, carried over a one hour session paid at an estimated \$40, results in a per hourly rate of savings of \$10.

Figure 47. Prototype Timing Test Results



Each session note generated with the prototype has the potential to permit additional income in time that is reclaimed from administrative paperwork tasks and applied to conducting additional sessions. Using a hypothetical set of values, for a therapist that sees 30 patients per week, using the prototype documentation system would save $(30 \text{ sessions} * 10 \text{ min/session})$ 300 minutes a week. At 60 minutes per session, five additional patients could be seen before eclipsing the amount of work done to see 30 patients using the manual documentation process (documenting these five additional sessions would take 30 minutes additional time at the average of six minutes per session note). For an increase in therapist workload of 30 minutes per week, at \$40 per session, the prototype system could offer an additional \$200 in income. If a therapist chose to remain at his/her current patient load, the therapist in the hypothetical example could save five hours of time per week – an additional hour of free time added to each day.

These temporal benefits are noteworthy and potentially valuable on many ways.

Tester qualitative commentary regarding usability included “easy to pick up”, “easy to follow”, “very fast” on the positive side, and “too many clicks” on the negative side. From comments such as “neat” and “cool”, the system appears to have a novelty effect on the user. Overall the commentary was mostly positive, with negative comments totaling less than 10% (Durrett, 18 Sep 2000).

The overwhelming positive slant of the commentary suggests the prototype design has merit for use, and the impressive quantitative results in documentation duration and time savings suggest the prototype design offers real value to prospective users.

IX. CONCLUSIONS

The dynamic sentence engine concept appears to hold promise as a mechanism for automating sentence creation through a simple point and click interface. Rather than see the same sentence time and time again, possibly with varying subjects and adjectives, but in the same hosting “frame”, random sentences regarding the same topic are possible as is the degree of context provided by the sentence. In continuing to explore the dynamic engine concept, the following questions must be answered:

- a) How large must the sub-trees get in the sentence interface in order to adequately support 90% of the situations seen by therapists? The initial body of session notes reviewed was too small a collection of the entire set to give a truly accurate picture of what a company therapist documents in a session. 90% was selected as a target for no particular reason – the company would set the required percentage.
- b) How well does the general user (unfamiliar with computers) adapt to thinking about their psychotherapy practice is sentence engine terminology and syntax? As the system would be the primary documentation tool, its usage would lead to a behavior change in how

therapists perceive their work. Therapists able to adapt quickly gain more benefits faster. It would be important to have a means of instructing therapists in acquiring this skill.

- c) Does the dynamic engine have any dire limitations? For one, the engine cannot make compound sentences in its current configuration. It will be necessary to determine if compound sentences are needed – if so, modifications to the engine would be required.

In speculating on the future of the prototype, there are deployment issues to consider. Representative issues include training, help, and security. In training, the makeup, content, duration, and delivery means of training therapists to use the tool is needed. The prototype system does not have an adequate help file system. Help is an absolute requirement prior to any production use of the system. Help contents, layout, and organization must be determined. Finally, security is of paramount importance when one considers that the system is dealing with medical record information. All data in the system is confidential, and like Quicdoc demonstrated, a security system must be put in place prior to production use.

In conclusion, the problem presented in this paper is exactly the kind of human process that information technology was created to address. A critical process is hampered by the human factor in the process workflow, and the

process steps are such that automation is applicable to the problem and satisfactorily resolves the problems. Further development of this prototype is likely to be a successful venture in making information technology an indispensable tool in the practicing geriatric psychotherapist's repertoire.

BIBLIOGRAPHY

- Adamski, Joseph J. and Pratt, Philip P., Database Systems Management and Design, (Danvers: Boyd & Fraser, 1994)
- Beck, Kent, Extreme Programming Explained, (Boston: Addison-Wesley, 2000)
- Church, Lee, interview with Author, 15 May 1999, Austin, TX, telephone Interview
- Cockburn, Alistair, Agile Software Development, (Boston: Addison-Wesley, 2002)
- Davis, Jim and Miles, Stephanie, "Evidence mounts of exodus from Windows CE", 16 Nov 1999, <http://news.cnet.com/news/0-1006-200-1449994.html?tag=bplst>; Internet; accessed on 10 Oct 2001
- Detweiler, Mark C. and Omanson, Richard C., Ameritech Web Page User Interface Standards and Design Guidelines, Ameritech Corp, 1996
- "Draft Evaluation & Management Documentation Guidelines (June 2000 with December Revisions)", (Health Care Finance Administration, 2000; accessed 10 Oct 2001) available from <http://www.hcfa.gov/medlearn/2000emd2.pdf>
- Durrett, H. John, interview by Author, 28 May 1999, San Marcos, telephone interview
- Durrett, H. John, interview by Author, 10 Oct 1999, San Marcos, telephone interview
- Durrett, H. John, interview by Author, 29 Mar 2000, San Marcos, telephone interview
- Durrett, H. John, interview by Author, 18 Sep 2000, San Marcos, telephone interview
- Jongsma, Arthur E. Jr. and Fraser, Deborah, The Older Adult Psychotherapy Treatment Planner (New York: Wiley & Sons, 1999)

McConnell, Steve, Rapid Development, (Redmond: Microsoft Press, 1996)

Microsoft Windows User Experience, (Redmond: Microsoft Press, 1999)

Miles, Stephanie, "Philips pulls the plug on Nino handheld", CNET, 6 Oct 1999, <http://news.cnet.com/news/0-1006-200-1591475.html?tag=bplst>; Internet; accessed on 10 Oct 2001

Miles, Stephanie, "Everex abandons palm-sized PC line", CNET, 15 November 1999, <http://news.cnet.com/news/0-1006-200-1441967.html?tag=mainstry>; Internet; accessed on 10 Oct 2001

Miles, Stephanie, "IBM pulls plug on Windows CE clamshell device", CNET, 27 Mar 2000, <http://news.cnet.com/news/0-1006-200-1591475.html?tag=bplst>; Internet; accessed on 10 Oct 2001

Moseley, Daniel J., "A Checklist of Common GUI Errors Found in Windows, Child Windows, and Dialog Boxes", (CSST Technologies, 1996; accessed 10 Oct 2001) available from <http://www.csst-technologies.com/guichk.htm>; Internet

Pressman, Roger S., Software Engineering: A Practitioner's Approach, 4th Ed. (New York: McGraw-Hill, 1997)

"Q192985: Connecting Windows NT-Based Computer to Windows CE-Based Device", Microsoft Developers Network; Microsoft Press, July 2001, CDROM

Rogers, Christi, R.N., "Documentation And Medical Necessity", Senior Psychology Services, San Diego, CA, 1998; corporate document

Senior Psychology Services, "Effective Documentation for Long Term Care: Handout for Psychologists", Senior Psychology Services, San Diego, CA, 1998; corporate document

Senior Psychology Services, Psychologist's Orientation Manual, Senior Psychology Services, San Diego, CA, 1998; corporate document

Senior Psychology Services, SPS Documentation Training, Senior Psychology Services, San Diego, CA, 1998; corporate document

Smith, George W., *Computers and Human Language*, (New York: Oxford University Press, 1991)

Zarit, Steven H. and Zarit, Judy M., *Mental Disorders in Older Adults: Fundamentals of Assessment and Treatment*, (New York: Guilford Press, 1998)

VITA

Personal: Born 3 March 1966 in San Antonio, Texas, son to Hugh Allan and Verna Thompson.

As of December 2001, Jeff may be reached at:

2014 Mimosa Trail
Round Rock, Texas 78664

Email: jt20816@onr.com

Academics:

Bachelor of Liberal Studies, St. Edward's University, Austin TX, 1993
Member, Alpha Sigma Lambda National Honor Society, 1993

Graduate student, Southwest Texas State University, San Marcos, TX,
1995-2001

Professional Credentials:

Jeff has held the following positions within the high-tech sector: process engineer in rapid thermal anneal for semiconductor manufacturer, production manager for semiconductor burn-in test laboratory, production control scheduler for a computer manufacturer, software developer for various entities, and project manager for a technology-based manufacturing company. Professional software interests include process automation design and systems integration.

