Extending the VA CPRS Electronic Patient Record Order Entry System Using Natural Language Processing Techniques

Christian Lovis, MD, Thomas H Payne, MD

VA Puget Sound Health Care System
Seattle, Washington

ABSTRACT

An automated practitioner order entry system was recently implemented at the VA Puget Sound Health Care System. Since the introduction of this system, we have experienced various problems, among them an increase in time required for practitioners to enter orders. In order to improve usability and acceptance of the order entry, an alternate pathway was built within CPRS that allows direct natural language based order entry. Implementation of the extension in CPRS has been made possible because of the three layers CPRS architecture and its strong object oriented models. This paper discusses the advantages and needs for a natural language based order entry system and its implementation within an existing order entry system.

INTRODUCTION

In 1998 the Computerized Patient Record System (CPRS) was first released at a national level. CPRS organizes and presents all relevant data on a patient to support clinical decision-making. It allows clinicians to interact with the patient's data, add problems, notes and enter orders. It supports alerts, notifications and guidelines. CPRS has been made possible because of the extensive set of clinical and administrative application within V*ist*A. CPRS must be seen as a line of tightly integrated products, using open and distributed architectures, which is able to support evolution and local adaptations.

In a recent review of the implementation of the CPRS system at the VA's Puget Sound Health Care System, it was emphasized that one of the most common problems described by physicians is the time required to enter orders [1]. One of the major challenges in designing the electronic patient record (EPR) is to meet the needs of detailed documentation while keeping the burden for direct care providers within an acceptable range. Tightly controlled and structured data entry can be a major burden for health care providers with high costs in time [2] [3]. However, handwritten orders are a common source of medical errors. Twenty to 70% of handwritten medication orders are incomplete [4] [5]. Most often, the missing information is date (15-20%), time (50-80%), or signature (5-15%). Dosage, formulation or special instructions account for 2-8% [6]. As much as twenty percent of the medication orders and 75% of the signatures are illegible or legible with effort [4] [7]. In addition, an error rate of 0.5-4.5% has been reported at the moment of preparation and distribution of drugs [8]. Beside largely avoiding errors due to writing and transcription, electronic order entry is an important step needed to provide comprehensive medical information and decision support systems [9] [10] [11] [12]. Electronic order entry is, in addition, cost-effective [13].

The evolution of computer interfaces toward the use of mouse, menus, forms and dialog boxes has largely simplified the use of computers, especially for novices. Command-line oriented systems, while difficult for novices, are very efficient for experienced users. The use of fast and robust natural language processing techniques allows novices to use the command-line approach efficiently. Such techniques are especially important in academic settings where there is a high turnover among interns and other healthcare providers. While building the data entry system of the EPR, it should be kept in mind that keyboard entry and command-line systems may be replaced by voice recognition in a near future. Natural language techniques are necessary steps towards such trends.

THE JIL[1] PROJECT

To improve and simplify the use of the existing electronic order entry in CPRS, we implemented an alternate ordering pathway based on natural language techniques. This development uses existing resources for maintaining available orders. It is not intrusive in the actual CPRS program nor does it introduce security holes. It minimizes modification of the actual code of the CPRS program and is easily maintainable. It is completely implemented as an additional unit to CPRS, which is tightly connected to

[1] Gill also Jil, Jill or Gill (jîl) noun: A girl, often one's sweetheart (Middle English gille, from Gille, a woman's name). JIL stands for *Just-Inside-Language*.

actual functionalities of CPRS. The source code and the naming convention are similar to those used in CPRS in order to facilitate integration.

CPRS can be considered as a semi-thick client at the top of a three-layer architecture. The deepest level consists of the MUMPS databases running on the **V***ist***A** servers with business rules that govern the authorized users of an available client/server application. Finally, it insures that the remote procedure calls have been registered and are valid for the application being run. Considering these elements, it appears that the best integration for JIL is to be part of the CPRS system rather than an independent order entry system.

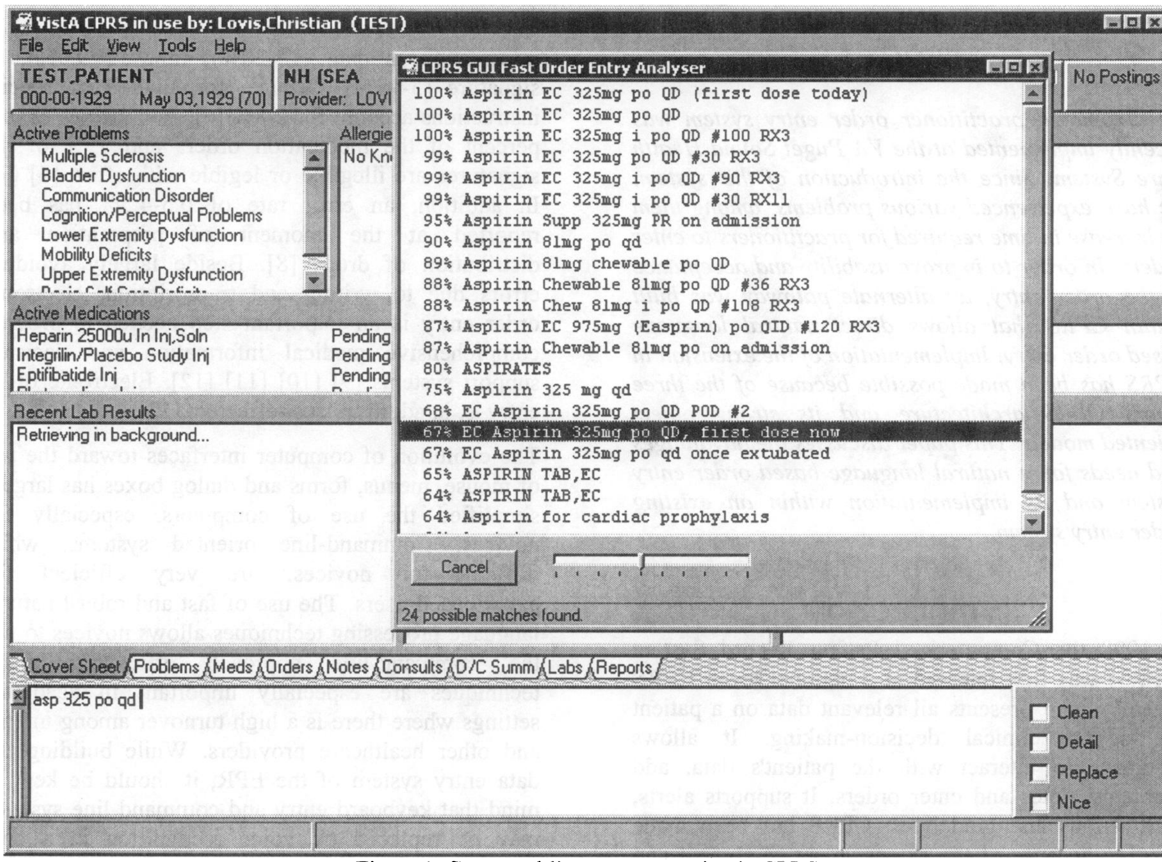

Figure 1: Command-line entry extension in CPRS

interactions with databases at the server level and provides management support for queries. The second level consists of the remote procedure call (RPC) broker that runs on both the server side and the client side and allows communication between servers and clients. The RPC broker is a key element playing the role of a bridge between servers and clients. It insures identification of clients and offers standardized interfaces and protocols for communication, data exchange and functions. It permits both the client and the server to be on independent and different hardware platforms. The uppermost and last layer is CPRS.

Besides offering a common interface for both the client and the server side, the RPC broker supports a three-part security-process. First, it insures that users have a valid access and verify codes and that they are

JIL architecture

JIL was entirely written using an object-oriented approach similar to the one used for CPRS. JIL is functionally divided in three parts. The first is devoted to conditional initialization. The second part deals with user's input analysis and conditional launch of commands. The third part manages the dynamic user interface of JIL and user interactions. These three groups of functions are tightly connected. The main objectives that lead the architectural development of JIL are:

1) Integration
2) Maintenance
3) Usability

Structurally, JIL is completely embedded in an object structure with two exceptions: The initialization function is external to the object structure so that it can be referenced easily by any calling procedure. It alleviates also the burden of JIL objects creation that is completely done within the initialization function. The second exception is the repository for the data dictionary. The data of the dictionary is kept in a sequentially packed array of structured records external to the JIL object but allocated dynamically. This feature makes the data easily available for other parts of CPRS. In addition, it permits initialization of the data independently from the JIL object, which can be more convenient to spare space complexity of the JIL add-ons.

JIL interacts directly and only with CPRS. The important point about this feature of JIL is that privacy, security, and user's identification are managed by CPRS. In addition, as JIL will only have interaction with the network through CPRS, it insures that JIL will remain largely compatible with further releases of CPRS. Furthermore, JIL does not need any additional or modification of existing RPCs or business rules in the servers. Finally, any modification of rights accesses done for CPRS will automatically be valid also for JIL.

All the visible elements that are added by JIL to the actual CPRS are built dynamically, which means that only minimal parts of CPRS source-code have to be modified in order to be compliant with JIL.

Users can interact with JIL within CPRS using three distinct elements. The first element is a menu item that is added dynamically and conditionally to the CPRS menu. The second element is the main editor that will be added to CPRS's main window and the third element is the dialog box that JIL will display if the user's command-line entry analysis leads to more than one result. The dialog box allows the user to select the most appropriate order in a list of orders matched with the command-line.

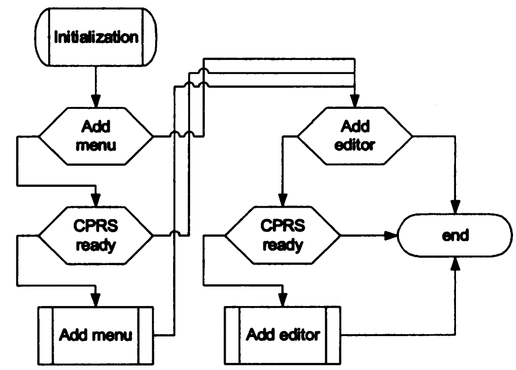

Figure 2: Dynamic interactions with CPRS

Users can change dynamically the specificity of the analyzer. All three elements are object-oriented and their initializations and allocations are done at runtime.

Menu caption	System meta order (TXT)	Help (HLP)
ABG	LRT ABG PLUS WC Q4H X24H	ABG, routine, Collect on Ward, Q4H X 24H
ASCORBATE. 1000 QD	PSJQ ASCORBIC ACID 1000MG PO QD	Ascorbic Acid 1000mg, oral, QD
CA QAM X 3	LRT CALCIUM SERUM LC QAM X 3	Calcium, Routine, Collect on Ward

Table 1: Example of menu description

Command-line entry	Order that will by executed by CPRS
Dx GIB	Admission diagnosis: Gastrointestinal hemorrhage
Cond critical	Condition is critical
Vitals per routine	Vital signs per ICU routine,
Orthostat on adm	Check orthostatic vitals on admission
NKDA	No known drug allergies
Activity bed rest	Patient must rest at bed

Table 2: Example items recognized by JIL

Main data dictionary

The quality of knowledge used for analyzing the user's entries is one of the most important factors for the overall quality of analysis [14]. The initial choice of knowledge source must take into account the quality of that source, its availability and also its maintainability [15]. The CPRS GUI offers a complex and deep structured menu to access specific and common orders. These menus are known as "quick orders". Quick orders consist of a large dataset of frequent orders. They cover all specialties and domains, including admission/discharge/transfer, conditions, severity and problems, diagnoses and procedures, drugs orders, diet, surveillances and laboratory orders among others. The menus are built dynamically during CPRS startup from a central database and updated as often as needed according to user's needs, usually daily. These menus allow users to quickly order items. However, most users complain about this system being difficult to use because of the number of menu items available. Unfortunately, while the addition of more quick orders accelerates direct entry, it also increases user's confusion in finding the desired order in a complex menu structure. This menu structure is used as main dictionary for JIL. An interesting feature of this solution is that it is usable in any menu-driven systems. It offers consistency between what is available through standard menus and command-line alternate pathway. In addition, the CPRS support team only updates one source, both for menus and for natural language based entry. Resources needed to maintain this source are already available and familiar to CPRS support team. Finally, menu-driven entry is the main pathway used in many VA facilities and other EPR systems for entering orders. The VA Puget Sound main menu dictionary contains more than 6,000 entries that have been refined during the last two years to answer the needs of a broad spectrum of clinicians.

The concept of generating automatically a data source from the dynamic menu of existing software can be used for many other applications and is not restricted to CPRS. One of the drawbacks of this approach is the lack of linguistic knowledge in the data dictionary. It contains only basic structure along a mono-axial construction and very little conceptual knowledge can be inferred from this hierarchy. However, the hierarchical structure of the menu and the accumulated contextual knowledge along the menu axes can be used to increase the power of representation of the menus caption. This is a similar situation as the structure and the expressions of the International Classification of Disease (ICD) [16]. In order to improve the quality of the analyzer, extended information is currently added to the existing fields. Actually, the knowledge available can be grouped in four categories: i) the contextual knowledge that is in the hierarchical structure of the menus; ii) the captions of the menus; iii) the metastructure of each menu used to access the RPC broker and iv) the extra knowledge currently added in order to refine the results of the JIL analyzer.

Analysis of text entered by users

The analysis is divided into three distinct steps. The first step is a morphological partial string pattern matching that uses the Boyer-Moore-Horspool algorithm because it allows fast processing on streamed texts [17]. The second step is devoted to computing a proximity score index for each entry in the dictionary. To do this, a pragmatic approach has been taken. The heuristic formula used to compute the proximity score takes into account various factors like the position of the match, the length of the match, the number of mismatches, the presence and frequency of collocations, the category of the match, etc. This formula minimizes silence. It has been refined experimentally in order to achieve the best overall results. This approach has the advantage of being fast and relatively robust even when there are abbreviations and typographical errors. The major drawback is that it generates noise. Therefore, users can vary dynamically the sensitivity of the formula. The results are presented in a list box and sorted by their proximity scores. One order usually generates between 10 and 20 responses. All processing is done in real-time. If an element of that list is selected, the order is sent to CPRS emulating actions that would have been taken using the usual menu pathway. The usual processing of the order is then performed.

The recognition of elements is strictly morphologic and based on an extract string pattern-matching algorithm. The user's entry is first tokenized into distinct chunks using space as separator. Each chunk is then compared to the data dictionary. The comparison is done in order to allow a match of the chunk at any position of the dictionary, therefore permitting abbreviations. The comparison is not case sensitive. Each entry in the data dictionary is divided in two categories, named HLP and TXT, as explained below. HLP is used to design the textual information extracted from menu captions, that is the text in the menus that is shown to users. HLP is the combined information that is used internally to describe actions taken by each menu item and the comments added by the informatics team to help maintenance of the menus.

According to the matches, a proximity score is given to each entry in the dictionary. The overall best score is kept and represents the maximal score that any entry in the data dictionary might have when compared to the user's entry. The score for every entry in the data dictionary is reported as a fraction of this maximal score. At the end of the processing, every entry in the data dictionary has a score between 0 and 1, the latter being considered as a 100% match. In fact, an entry that has a score of 1 will be the best match to the user's entry when compared to all other data-dictionary entries.

The score is computed using a heuristic formula that has been refined during preliminary tests. Because no conceptual knowledge is available, a semistochastic approach has been taken. The formula takes into account the category in which the match occurs: TXT or HLP. A match in HLP is weighted twice as much as a match in TXT. It takes into account the length of the pattern being matched, i.e., the longer the length the stronger the match. It also takes into account the position of the chunk currently tested in the user entry; the first position receives more weight. And, it takes into account the position within the data-dictionary where the match occurs, the first position receiving more weight. Finally, it gives additional weight to situations where the chunk is the first one and the match occurs at the first position in the data-dictionary. All weights are additive.

In essence, this formula exploits the fact that the data dictionary is divided in two categories. The TXT category is the more specific and the HLP is the more sensitive. In addition, it weights more matches at the beginning of words or sentences.

CONCLUSIONS

We have successfully implemented an alternate pathway for order entry for the CPRS system without introducing modification of the CPRS source code thanks to its three layers architecture and its strong object oriented implementation. The source dictionary allowing the analysis is directly and dynamically extracted from the menu structure of CPRS. Despite this apparent simplicity, the system is robust and efficient and allows users to enter most common orders using a natural language based interface. A formal evaluation of the system with a user's satisfaction survey is ongoing.

Acknowledgement

We are grateful to Margaret Bender for editing and helpful criticisms.

This work was supported by a grant of the University Hospital of Geneva, Switzerland.

REFERENCES

1. Payne TH. The transition to automated practitioner order entry in a teaching hospital: the VA Puget Sound experience [In Process Citation]. Proc AMIA Symp 1999(1-2):589-93.
2. Bates DW, Boyle DL, Teich JM. Impact of computerized physician order entry on physician time. Proc Annu Symp Comput Appl Med Care 1994:996.
3. Tierney WM, Miller ME, Overhage JM, McDonald CJ. Physician inpatient order writing on microcomputer workstations. Effects on resource utilization. Jama 1993;269(3):379-83.
4. Winslow EH, Nestor VA, Davidoff SK, Thompson PG, Borum JC. Legibility and completeness of physicians' handwritten medication orders. Heart Lung 1997;26(2):158-64.
5. Francois P, Chirpaz E, Bontemps H, Labarere J, Bosson JL, Calop J. Evaluation of prescription-writing quality in a French university hospital. Clin Perform Qual Health Care 1997;5(3):111-5.
6. Tissot E, Henon T, Cornette C, Jacquet M. [Incomplete prescription: a potential medication error]. Presse Med 1999;28(12):625-8.
7. Brodell RT, Helms SE, KrishnaRao I, Bredle DL. Prescription errors. Legibility and drug name confusion. Arch Fam Med 1997;6(3):296-8.
8. Taylor J, Gaucher M. Medication selection errors made by pharmacy technicians in filling unit dose orders. Can J Hosp Pharm 1986;39(1):9-12.
9. Bates DW. Frequency, consequences and prevention of adverse drug events. J Qual Clin Pract 1999;19(1):13-7.
10. Kuperman GJ, Teich JM, Bates DW, Hiltz FL, Hurley JM, Lee RY, et al. Detecting alerts, notifying the physician, and offering action items: a comprehensive alerting system. Proc AMIA Annu Fall Symp 1996:704-8.
11. Jha AK, Kuperman GJ, Teich JM, Leape L, Shea B, Rittenberg E, et al. Identifying adverse drug events: development of a computer-based monitor and comparison with chart review and stimulated voluntary report. J Am Med Inform Assoc 1998;5(3):305-14.
12. Kuperman GJ, Teich JM, Tanasijevic MJ, Ma'Luf N, Rittenberg E, Jha A, et al. Improving response to critical laboratory results with automation: results of a randomized controlled trial. J Am Med Inform Assoc 1999;6(6):512-22.
13. Simborg DW, Derewicz HJ. A highly automated hospital medication system. Five years' experience and evaluation. Ann Intern Med 1975;83(3):342-6.
14. Lovis C, Baud R, Michel PA, Rassinoux AM, Rodrigues JM, Scherrer JR. Full text multilingual automatic morphosemantems for stand-alone or Internet based applications. Medinfo 1998;9(Pt 1):155.
15. Lovis C, Baud R, Rassinoux AM, Michel PA, Scherrer JR. Medical dictionaries for patient encoding systems: a methodology. Artif Intell Med 1998;14(1-2):201-14.
16. Baud R, Lovis C, Rassinoux AM, Michel PA, Scherrer JR. Automatic extraction of linguistic knowledge from an international classification. Medinfo 1998;9(Pt 1):581-5.
17. Horspool R. Practical fast searching in strings. Software-Practice and Experience 1980;10(6):501-506.