Clinical Decision Support for Physician Order-Entry: Design Challenges

Carol A. Broverman, Ph.D., Jeffrey I. Clyman, M.D., Joseph M. Schlesinger
ALLTEL Information Services, Health Care Division
San Jose, California

Elliotte Want, M.B.A.
Clinical Information Systems, IBM Worldwide Healthcare Industry
Bethesda, Maryland

We report on a joint development effort between ALLTEL Information Services Health Care Division and IBM Worldwide Healthcare Industry to demonstrate concurrent clinical decision support using Arden Syntax at order-entry time. The goal of the partnership is to build a high performance CDS toolkit that may be easily customized for multiple health care enterprises. Our work uses and promotes open technologies and health care standards while building a generalizable interface to a legacy patient-care system and clinical database. This paper identifies four areas of design challenges and solutions unique to a concurrent order-entry environment: the clinical information model, the currency of the patient virtual chart, the granularity of event triggers and rule evaluation context, and performance.

CLINICAL DECISION SUPPORT

Today in health care, the driving forces are cost containment and quality of care. Since most patient care actions and a large proportion of variable costs are generated by physician orders, enhanced feedback at the physician/computer interface can improve health care delivery [1,2]. This paper describes a project to provide real-time value-added assessments of captured clinical data to assist physicians during order-entry.

We assume that the goal of *clinical decision support (CDS)* is to present the "right information" to the "right person" at the "right time," in the "right format." We make the following distinctions regarding CDS design choices:

- *passive* versus *active* systems: passive systems are invoked by users while active systems perform functions continuously as a byproduct of data management;
- *aggregate* versus *individual patient-based* systems: some CDS systems analyze patterns across a patient population; other systems analyze data for one patient at a time;
- *concurrent* versus *retrospective* systems: some systems provide feedback concurrent with the care-giving while others provide feedback after events have occurred;
- systems *integrated with patient data base* versus *standalone* systems: some systems are integrated with a full electronic patient record; standalone systems make decisions in a more isolated context or rely on user-supplied information.
- *proprietary* versus *standards-based encoding:* some systems use proprietary coding schemes for patient data, logic/rules and clinical concepts, while other systems adopt industry standards where available and appropriate.

Our position is that the goal of CDS is best achieved by an active, patient-based system that is concurrent and integrated with the patient database and order-entry process. Furthermore, a standards-based solution utilizing open-systems technology will maximize reuse and broaden market acceptance. For these reasons, our joint design focuses on *event-based rules processing triggered by clinical orders.* The approach described also has advantages of being modular, extensible and customizable, and allows for enterprise "ownership" of clinical content.

The Need for Standards

Event-based processing is not new. Several institutions have developed their own implementations of event-based CDS, demonstrating cost savings and improved quality of care [3,4,5]. Lessons learned from these implementations led to the development of the Arden Syntax as a standard

0195—4210/96/$5.00

representation for medical logic to facilitate knowledge reuse [7].

The anticipated widespread use and sharing of Arden rules has been not been achieved. The academic community has been responsible for most CDS implementations using the Arden standard; vendor-supported implementations are not widely available. Since Arden specifies neither a data model nor an architecture, demonstrations of the usefulness of the standard are limited to home-grown implementations with complex infrastructures. All have used proprietary databases, locally built decision-support servers, and different forms of controlled vocabularies and data dictionaries. Furthermore, requirements for "synchronous" rule feedback to support order-entry functions in a timely manner (preferably before the order is completed and processed) have not been adequately addressed.

PROJECT SCOPE

A joint development project between ALLTEL Information Services Health Care Division (formerly TDS) and IBM Worldwide Healthcare Industry is investigating the feasibility of a general CDS solution for the order-entry environment. Our approach has the following features: (a) the use of emerging standards (Arden and SNOMED (College of American Pathologists)) , (b) the encapsulation of the clinical patient data model to facilitate maintenance, (c) real-time support for physician order-entry, and (d) the integration of legacy clinical patient information with CDS through retrieval and data transformation mechanisms. The IBM Clinical Information System (CIS) toolkit represents "middleware" that will operate with different existing interfaces both at the front and back ends. It is designed with open systems standards to allow portability across multiple platforms.

The components of the IBM CIS toolkit are (a) a transient clinical information repository server, (b) an Arden rule-authoring environment, (c) a clinical lexicon server for controlled vocabulary, and (d) a decision-support server to evaluate rules against patient data to produce CDS feedback. All components interact through published interfaces.

ALLTEL is enhancing its proven physician order-entry capabilities by integrating the IBM CIS toolkit to provide CDS. To achieve this, ALLTEL is adding components to the design solution including an event monitor, a high-performance data server, and a action manager. The event monitor and action manager coexist on a Windows 95® or Windows NT® desktop client with the *TDS 7000* order-entry interface. The *event monitor* detects and filters clinical events at the user interface. The ALLTEL data server imports patient data from the ALLTEL Permanent Patient Record into the transient repository in response to decision-support server requests. The *action manager* presents decision-support server feedback to the clinical user in a timely fashion. The architecture of the system we are implementing is shown in Figure 1.

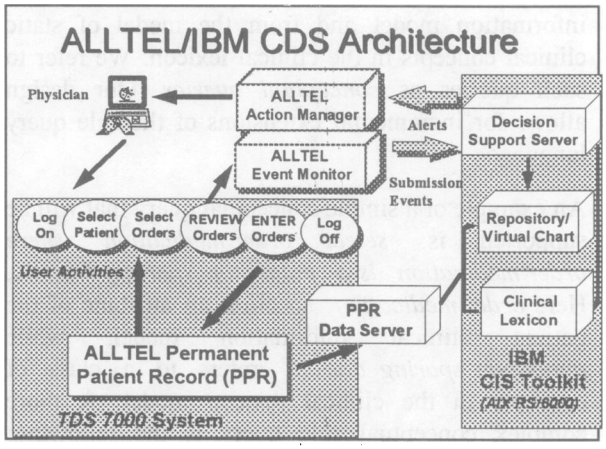

Figure 1: ALLTEL/IBM CDS Architecture

A brief summary of the system flow is as follows: The physician selects a patient from her patient list. She follows ordering pathways to compose orders for that patient. Then she invokes the *REVIEW* function to check the orders. The *REVIEW* function triggers the capture and submission of clinical events to the decision-support server. If the orders are acceptable to the physician and do not generate any CDS alerts, the physician invokes *ENTER* to electronically sign her orders. Our goal is to provide feedback to the physician before the *ENTER* action is invoked.

The rest of this paper discusses issues encountered during design, emphasizing requirements of the order-entry environment. This is work in progress, with a proof-of-concept phase scheduled to begin at three ALLTEL test sites in January 1997.

REQUIREMENTS AND CHALLENGES

Four areas have presented challenges for our design: (a) the clinical information model, (b) currency of the patient virtual chart, (c) event granularity and rule evaluation context, and (d) performance.

Clinical Information Model

Rules-based processing requires a consistent clinical information model to bind patient data with rule logic at run-time. In Arden, this is done through the *rule query language*. While a single standard *clinical information model* is still a "holy grail," some form of data model is required.

In order to author rules effectively, a rich and intuitive view of the clinical information model is needed. Ideally, the rule query language should allow the author to seamlessly specify queries that combine elements from both the patient clinical information model and from the model of static clinical concepts in the clinical lexicon. We refer to such queries as *conceptual queries.* Our design allows for incremental extensions of the rule query language.

An example of a simple conceptual query that will be supported is: *select order.medication where order.medication is-a potassium_sparing_diuretic.* Here *order.medication* specifies an attribute of the patient clinical information model, while *potassium_sparing_diuretic* refers to a class of concepts in the clinical lexicon model. A more complex conceptual rule query is the following: *select order.medication where order.medication.dosage > medication.maximum_dosage.* In this case, *medication* is both an attribute of the *order* object in the clinical information model and the name of a clinical lexicon model concept. The implementation of the rule query language must support traversal of both models to support this query.

While the rule author requires awareness of the clinical information model, a well-designed system will encapsulate it to facilitate modification as databases and data needs change. Our design represents the clinical information model declaratively within the *CIS schema knowledge base.* This encapsulation (a) insulates the event monitor and the data server components of our architecture from the details of the clinical information model, and (b) allows those components to exploit reuse of data transformation code.

The event monitor is responsible for detecting, packaging, and transmitting user-interface events, known as *submission events,* to the decision-support server. After submission, the decision-support server invokes a set of predefined transformations on these objects to convert the original submission data format into the clinical information model format. Likewise, a similar set of transformations is applied to the data imported by the data server.

Currency of the Patient Virtual Chart

Other design decisions deal with the persistence and currency of the model of patient data. Functional requirements should be defined via analysis of the ideal manual process. When a physician is about to generate orders for a patient, she should be familiar with all current orders, lab results, and other pertinent information such as allergies and concurrent medical problems. We refer to the electronic analog of the manual chart as the *virtual patient chart.* It contains the current snapshot of patient data according to the published clinical information model.

To support both functional and performance requirements, we define the concept of a *session* as the appropriate time frame to keep a virtual patient chart "in focus" for CDS functions. In *TDS 7000,* a session begins when a patient is selected by the physician, and concludes when the physician selects another patient or logs off.

At the start of a session, the patient data required to populate the clinical information model is imported to form the baseline for the virtual patient chart. To maintain the currency of the patient chart, new orders are added to the chart as the session proceeds. This process is described in the following section when describing the *context for rule evaluation.*

When a session ends, the virtual patient chart is flushed. With this concept of a session, the patient data required for decision-support server execution is optimally accessible during the time window that this patient is being considered by the physician. This design has positive implications for performance.

Trigger Granularity and Rule Evaluation Context

The definition of what constitutes a rule-triggering *clinical event* is unclear [6]. The optimal granularity of such events is also unclear. Should a single order for an IVP within a set of orders for a patient with possible kidney stones be considered a trigger for CDS, or should triggering be done at the level of all related orders (e.g., serum creatinine lab test order, plus an IVP order)? The issue here is analogous to determining the appropriate use of a spelling/grammar checker. Should spell/grammar-checking be done at the letter, word, phrase, or sentence level? Are order sets (i.e., sentences) clearly delineated? More research is needed in this area.

Event bundling. Our design takes advantage of the behavior of the *TDS 7000* user interface to establish a proposed granularity for events. We identify the notion of a *submission event bundle,* delineated by a *REVIEW* action, to transmit orders to the decision-support server. While a single submission event bundle may contain 1 to N order events, the event transmission to the decision-support server and CDS processing is invoked at the submission event bundle level. The coarser grained event bundle establishes a "sentence" context (refer to the spell/grammar checker analogy). This viewpoint assumes that a batch of orders may represent some higher-level construct, such as a protocol or a *conceptual order set.* For example, a conceptual order set might be "*orders for patients with possible kidney stones = (IVP, a serum creatinine lab order, analgesics, ultrasound).*"

Submission data context. When the physician selects *REVIEW,* the orders generated since the last *REVIEW* in *TDS7000* are bundled together and appended to a *submission data accumulator* that persists on the client until the end of a session. The accumulated submission data is sent to the decision-support server with each new submission event bundle and is added to the virtual patient chart. This behavior assures that rules will evaluate each new bundle of orders within an *ORDER/REVIEW/ENTER* cycle within the context of all orders generated during the session. The patient virtual chart is thus kept as current as possible.

The handling of *submission data* also has other desired implications for our design. First, since submission data is maintained on the client before *ENTER*ed, the physician may remove a submission from the order screen in response to an alert without requiring access or modifications to the patient virtual chart on the server side. The necessary bookkeeping may also be done at the client desktop to modify the status of session orders without affecting the patient database until final confirmation by the physician. The maintenance of the status of events has implications for allowable behaviors of the physician in response to an alert (e.g.; a *submitted_still_proposed* event may be removed by the physician before it has been sent to the database; a *submitted_and_signed* order must be discontinued rather than simply removed from the order screen).

In our system, individual Arden rules are invoked by a single order, but the rule logic is evaluated within the context of all the orders contained in the accumulated submission data. This design approach alleviates part of the problem identified as the *future events* problem in a similar event-based system [8]. For example, if the IVP order in the previous discussed conceptual order set was evaluated as an event without the context of the serum creatinine lab order (the next order in the event stream before *REVIEW*), an unnecessary reminder might be generated to order the creatinine. Bundling them together results in correct system behavior.

Event bundling may not always be appropriate, however. In cases where we intend to alert for *errors of commission* such as disregarding a medication allergy, the rule logic should be invoked at the level of the selection of a single order for a potentially offending medication rather than waiting for the *REVIEW* action associated with multiple orders. On the other hand, *errors of omission* (e.g., reminders to order a test that may have been overlooked) are often best detected when evaluating events within a larger context (event bundles). Future work will investigate mechanisms for setting the appropriate granularity for event submission and rule triggering; this may require developing algorithms for preprocessing rules and require extending the Arden language.

Performance

The key feature of ALLTEL's electronic patient record system is its acceptance by physicians for direct order-entry; a critical success factor is its *speed.* Physicians are unlikely to tolerate anything less than a lightning-fast response time if they are to incorporate electronic order entry into their busy schedules. Thus, some of the most important challenges in our joint design are to discover performance bottlenecks and reduce them.

One bottleneck is the volume of events being transmitted to the decision-support server. In an event-processing system, every order is a potential trigger event. Our design goal is to reduce, filter, and/or collapse the number of events that are deemed "interesting" as soon as possible, either before they are sent to the decision-support server or at least before the rule has started to execute. In addition to event bundling, we filter at a gross level on the desktop, and rely on a conceptual event filter in the decision-support server that makes use of the hierarchical relationships between concepts in the clinical lexicon (e.g.; *spironolactone is-a potassium_sparing_diuretic).*

Another potential bottleneck is the number of workstations that are sending events. To reduce this traffic, we have chosen to activate CDS only for specific user classes. Events are submitted only for users who are authorized to act on an alert, such as ordering physicians. A processing thread is dedicated to a workstation at the user log-on level to maximize performance for our usage patterns. The memory cache used during rule evaluation is tied to a session for the same reasons.

Experience has shown that accessing patient data imposes the biggest load on CDS systems [9]. Therefore one design objective is to reduce the amount of data imported from the database to evaluate rules. Initially, we have reduced the data model to a minimal set of key data elements, and will add data elements incrementally. Future work will investigate algorithms to preprocess rules as they are authored. This should result in intelligent schemes to limit the scope of imported data.

We also employ caching and anticipatory "preloading" to reduce I/O. A periodically computed list of currently admitted patients and expected appointments guides a daily preload of patient data to approximate currency of the patient charts likely to be accessed. This strategy minimizes the extent of the incremental updates that are performed at the start of a patient session. This optimization is especially important in our case, since we are also transforming transactional data from an existing format into the CDS data model.

SUMMARY

Our goal is to enable clinician decision support in a production environment at physician order-entry time, where it can have the greatest impact on health care costs and quality. We have discussed design challenges and solutions to promote reusability and maintainability in a concurrent order-entry environment. Key issues identified are: maintaining currency of the patient virtual chart, encapsulation of the clinical information model, setting granularity of event, and performance. By also basing our design on open standards (Arden Syntax and SNOMED) and open technology, we hope to offer a solution that is repeatable across multiple enterprises and that realizes the goal of Arden Syntax to share rules across institutions.

Acknowledgments

This work is being done in conjunction with other members of the IBM CIS team, in particular, Laifong Leung, Roger Reider, Kevin Brown, Jean Loving, Mark McCourt, Randall Ho, and Shelia Moore.

References

1. Tierney WM, Miller ME, Overhage JM, McDonald CJ. Physician inpatient order writing on microcomputer workstations: Effects on resource utilization. *JAMA*, 1993; 269(3):379-383.

2. Evans RS, Cassen DC, Stevens LE et. al. Using a hospital information system to assess the effects of adverse drug events. *Proceedings of the Annual Symposium of Computer Applications in Medical Care*, 1993; 161-65.

3. McDonald, DJ, Blevins L, Tierney WM, Martin DK. The Regenstrief medical record, *MD Computing*, 1988; 5(5): 34-47.

4. Pryor TA, Gardner RM, Clayton PD, Warner HR. The HELP system. Springer-Verlag, New York, 1984.

5. Hripcsak G, Cimino J, Johnson S, Clayton P. "The Columbia-Presbyterian Medical Center decision-support system as a model for implementing the Arden syntax," *Proceedings of the Annual Symposium of Computer Applications in Medical Care*, 1992; 248-252.

6. Broverman CA. A clinical event is more than a database update. *Proceedings of AMIA Spring Congress*, 1994; 66.

7. Hripcsak G, Ludemann P, Pryor TA, Wigertz OB, Clayton PD. Rationale for the Arden syntax. *Computers and Biomedical Research*, 1994; 27:291-324.

8. Ludemann P. Mid-term report on the Arden syntax in a clinical event monitor. Computers and Biological Medicine. 1994; 24(5):377-383.

9. Hripcsak G, Johnson SB, Clayton PD. Desperately seeking data: knowledge base-database links. *Proceedings of the Seventeenth Annual Symposium on Computer Applications in Medical Care*, 1994; 639-643.