Architecture for a Multipurpose Guideline Execution Engine

Aziz A. Boxwala, MBBS, PhD, Robert A. Greenes, MD, PhD, Stephan R. A. Deibel
Decision Systems Group, Brigham and Women's Hospital, Harvard Medical School
75 Francis Street, Boston, MA 02115

Integration of guideline knowledge into the clinical workflow is essential, for improving adherence to guidelines. Guidelines in structured formats can be utilized by computer programs to provide decision support in clinical information systems, as well as to facilitate workflow. We have designed an architecture for a flexible guideline execution engine that can be utilized in clinical decision support applications. The engine may be utilized for other applications such as referral management, medical education, and conducting clinical trials. The engine executes guidelines that are defined in an extension of the Guideline Interchange Format (GLIF). GLIF was extended to support representation of constructs that are essential to the execution of the guideline. A prototype of the engine was implemented based on this architecture. The engine is being utilized in two clinical applications that draw on guidelines for decision support. The engine was also used for developing an educational application aimed at testing knowledge of guideline recommendations.

INTRODUCTION

In the current healthcare environment, guidelines are being promulgated as a primary means to standardize care, improve quality of care, and increase the cost-effectiveness of services that are provided [1]. However, studies have found that compliance with guidelines in practice has not been satisfactory [2, 3]. Better integration of guideline knowledge into the clinical workflow has been demonstrated to improve compliance [4-6]. The integration has often taken the form of computer-generated patient-specific reminders to clinicians during the encounter with the patient [7].

A major obstacle in the implementation of guidelines on a large scale in computer-based decision support systems is the effort required for creating guidelines in a structured computer interpretable format. The guidelines developed for use in such systems usually tend to be in a proprietary format. This limits the sharing of guidelines across institutions and even across different types of applications.

Lobach et al have developed a relational model for representation of guidelines [8]. The Proforma model represents the pros and cons of competing recommendations in a guideline so as to enable reasoning from such guidelines [9]. Dazzi et al have developed the Patient Workflow Management System. The system models guidelines and organizational characteristics of the institution in order to enable workflow management based on care guidelines [10]. Arden Syntax, a published standard [11] for the representation of clinical rules, does not yet support representation of guidelines (such as those for disease management). Shahar et al have proposed the Asbru representation of guidelines that emphasizes intentions of guideline decisions and recommendations [12].

Guidelines so structured have been used in a variety of applications. The most common use for guidelines is to provide decision support during the care of patients. As mentioned earlier, guidelines can also be used for providing workflow management support. In this case, recommendation information contained in the guideline can be used to anticipate clinical actions. Structured guidelines may also be used for quality assurance evaluations [13]. These evaluations can be conducted by measuring compliance to the guideline. Compliance is measured by comparing clinical orders noted in the patient record to guideline recommendations. Protocols in clinical trials may also be encoded as structured guidelines [14]. These guidelines are used to improve compliance to protocols through better data collection and through reminders for tasks to be carried out during encounters with study subjects. Structured guidelines may also be used in simulation programs for educational purposes.

The Guideline Interchange Format (GLIF) is a structured representation format for guidelines created by the Intermed Collaboratory [15], which included this laboratory. An important goal in creating GLIF was to enable sharing of guidelines among institutions and across computer applications, including their associated documentation.

GLIF specifies an object-oriented model for guideline representation and a syntax for guideline transport. A GLIF encoded guideline is essentially a flowchart representing a temporally ordered sequence of steps. Different types of steps in the flowchart represent clinical actions or decisions.

1091—8280/99/$5.00

Our hypothesis was that a shareable representation for guidelines, such as GLIF, could be utilized in different types of applications. We have developed an architecture for a guideline execution engine that utilizes guidelines encoded in GLIF. This execution engine is intended for use in a variety of applications. The engine traverses the guideline by evaluating logic conditions specified in the guideline against patient data values. The results of the evaluation are used to generate patient-specific recommendations from the guideline. The published GLIF representation [15] (GLIF Version 2.3 or GLIF-2.3) was inadequate for execution by the engine. Several enhancements were made to GLIF in order to make it executable, and which we expect to propose as formal extensions to the GLIF. The architecture for the engine was implemented in a prototype system. We are using the prototype implementation of the engine in two pilot clinical applications and in a simulation program that is aimed at testing knowledge of guideline recommendations.

METHODS AND MATERIALS

Enhancements to GLIF

GLIF-2.3 did not provide a complete definition for the contents of a guideline, especially as required for automated execution of the guideline in a computer program. The representation provided a structure consisting of slots for describing attributes of a guideline and its constituent parts such as the steps in the guideline. However, GLIF-2.3 did not specify a format for the contents of the slot, an essential requirement for the execution of the guideline.

We augmented GLIF by specifying the format for the contents of these slots. The enhancements to GLIF are divided into the following categories:

Enhanced patient data model. A richer patient data model was created which supports a number of data types, and permits specification of cardinality and temporal and logical constraints on the values of the data (Figure 2). The new data types were created by sub-classing from the Patient_Data class of GLIF-2.3 (Figure 1). The guidelines can also contain references to meta-data in external sources such as data dictionaries.

Enhanced action model. An object-oriented model for actions (guideline recommendations) was added that could support different types of actions (e.g., prescription, notification, or referral). An abstract class called Action_Parameter (Figure 1) was created to define attributes required for executing an action. Different action types are described by adding an

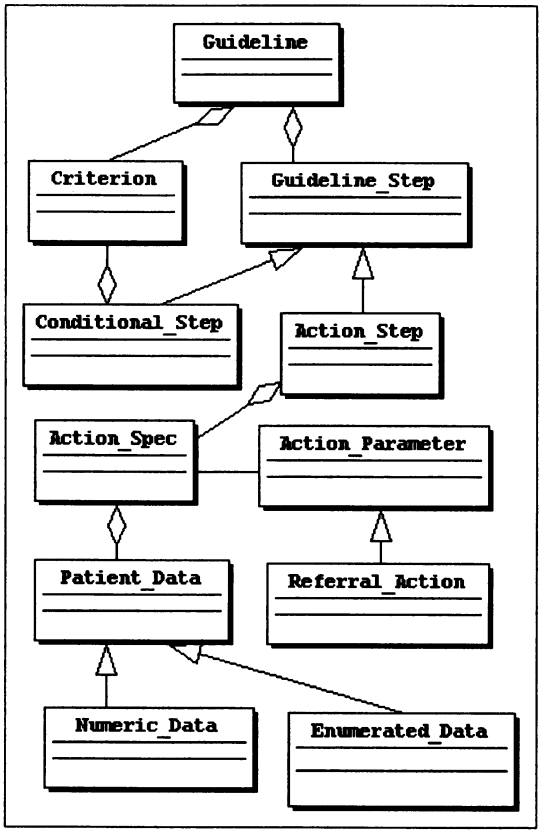

Figure 1. A simplified object model in UML notation showing enhancements to GLIF. (A link with triangular arrow head indicates a generalization relationship. A link with the diamond arrowhead indicates an aggregation relationship. A link with no arrowhead indicates an association).

object of a sub-class of the Action_Parameter class to the Action_Spec class. The sub-classes add action specific parameters for execution of the action (e.g., drug name, dose, frequency are parameters for the prescription action).

Syntax for logical constraints. Logical constraints in GLIF are used to specify decision logic in

```
Numeric_Data 2 {
  name = "Serum Cholesterol";
  scale = "mg/dl";
  required = TRUE;
  default_value = 0;
  minimum_value = 100;
  maximum_value = 200;
}
```

Figure 2. Example of meta-descriptions of patient data in GLIF. The description is for serum cholesterol data. Serum cholesterol is stored as a numeric data type, is measured in mg/dl, is a required item for this guideline, and has minimum, maximum, and default values.

conditional steps, eligibility criteria for the guideline (Figure 1), and constraints on values of patient data. For specifying such logic, we adopted a modification of the logic statement grammar in Arden Syntax [16]. The modifications to Arden Syntax logic grammar were made in order to support reasoning from a richer patient data model and from knowledge bases containing more complex data structures (Figure 3). The modified syntax for the logic statement is described in more detail elsewhere [17].

Other enhancements. Other features were added to GLIF that enable use of GLIF encoded guidelines in large knowledge bases. Attributes were added for version control and for unique identification of guidelines. In order to aid retrieval and management of guidelines, facilities for assigning guidelines to categories were also added.

An XML-based packaging was created for GLIF as an alternative to the original Object Data Interchange Format packaging. XML is fast becoming the basis of data interchange and we expect that an XML-based format for GLIF will improve the ability to share guidelines. The execution engine utilizes the XML format exclusively.

Architecture for the guideline execution engine

We have developed a system-independent and application-independent architecture for a guideline execution engine that utilizes GLIF-encoded guidelines. The architecture can be used to implement a decision support engine for clinical applications.

```
medication1 is-a beta_blocker
bp.systolic > 120
```

Figure 3. Examples of logic statement syntax in GLIF. The first line illustrates the "is-a" operator required for the hierarchical data structure. The second line illustrates the "." operator, exemplifying a compound data type with more than one field (similar to *struct* in the C language).

The application independence and system independence are provided by utilizing a component-based paradigm for the architecture. In order to adapt the engine to different applications and systems, some components may be replaced with functionally different components.

The components comprising the engine are the Guideline Selector, the Guideline Accessor, the Guideline Traverser, the Logic Evaluator, the Data Dictionary, the Action Realizer, and the Patient Data Accessor (Figure 4). Each of these components is described next.

Guideline Selector. This component selects guidelines from a potentially large database of guidelines. The guideline may be selected based on automatic matching of patient data with eligibility criteria for the guidelines. Alternatively, the guideline may be manually selected, as may be the case in an interactive application.

Guideline Accessor. This component loads the selected guideline from the database into the engine. The Guideline Accessor also provides other

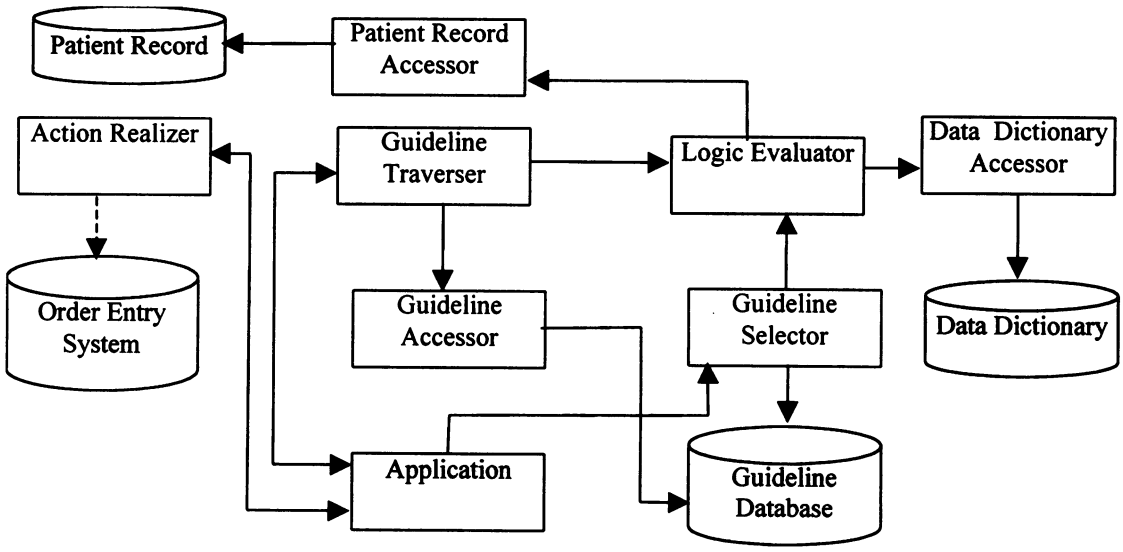

Figure 4. Components of the guideline execution engine and examples of interfaces with external systems. The external systems are drawn shaded and the engine components are not shaded. The arrows indicate the direction of the service request among components.

components of the engine with access to the constituent objects of the guideline.

Guideline Traverser. The Guideline Traverser provides the engine with capability to navigate a guideline. This component keeps track of the current position in the guideline and the path taken through a guideline, and determines the next step in the guideline based on patient data.

Logic Evaluator. This component is called by the Guideline Traverser to evaluate logic statements against actual values for patient data. The Logic Evaluator, in turn, calls the Patient Data Accessor to get values of patient record items needed to evaluate a logic statement. The Logic Evaluator accesses the Data Dictionary to validate the patient data values.

Patient Record Accessor. The Patient Record Accessor component provides the Logic Evaluator with values for patient data items. This component would be implemented differently in different systems. For example, different implementations of the component would provide access to data from an electronic medical record than from direct user input to a form.

Data Dictionary Accessor. This component provides the engine with access to items from a data dictionary. The data dictionary contains meta-information about the patient data. The meta-data are used to validate values for patient data. This component may also be used by the Action Realizer component to implement data collection forms dynamically.

Action Realizer. The Action Realizer component actualizes clinical action recommended by the guideline. The implementation of this component would vary by system and by application. As part of a clinical system, the Action Realizer may connect with the order entry system to assist the care provider in implementing the recommendations. In an interactive training application, the Action Realizer may present a list of guideline recommendations.

A typical sequence of interactions among the components of the engine and the external systems starts by an application searching for a guideline from the guideline database using the Guideline Selection component. The application now interacts with the Guideline Traverser. The guideline is loaded in the engine using the Guideline Accessor. The Guideline Traverser gets guideline step information from the Guideline Accessor. The Logic Evaluator is requested to evaluate any logic conditions by obtaining patient information via the Patient Record Accessor. The actions or recommendations are passed on to the Action Realizer from the application, which implements the action. The Action Realizer, in turn, notifies the application of change in status of the action, as when an action is completed.

RESULTS AND DISCUSSION

This architecture was partially implemented in a prototype engine. The components were implemented as ActiveX server components on the Microsoft Windows NT™ platform.

The engine is being used currently to develop two pilot clinical applications, which provide decision support from knowledge contained in guidelines. The clinical applications are in the domains of neurology and dermatology. An educational application was also developed with this engine.

The first application is aimed at assisting neurologists at an academic medical center in managing referrals for acute stroke from remote hospitals. The neurologist must assess the most appropriate therapy for the patient and decide whether the patient should be transferred to the center. The guidelines for this application will help select the therapy option (intravenous or intra-arterial thrombolytics, neuro-protective therapy, etc), in some cases to decide whether the therapy should be delivered at the medical center or at the referring institution, and to guide in the delivery of the selected therapy.

The second application of the guideline execution engine is in a dermatology decision support system for primary care physicians (PCPs). The system aims at guiding the PCP in the assessment and management of the dermatological problem. When a referral to a dermatologist is necessary, the system will recommend the referral and assist in setting it up via a telemedicine system. Since the telemedicine system in use is an asynchronous "store-and-forward" system, the decision support system will guide the PCP in gathering and forwarding relevant patient information to the dermatologist.

The guideline execution engine has also been used to develop a simulation program. This program generates patient management options from a guideline based on a patient profile. The user-selected option is compared to the correct recommendation of the guideline for that patient. The latter recommendation is determined by executing the guideline execution engine in a batch-mode for this patient profile.

The three applications described above utilize the same architecture and the same core component implementations for the execution engine. However,

due to the varying requirements, different Action Realizor and Patient Data Accessor components are being implemented for all these applications.

CONCLUSIONS

We have developed an open architecture for a guideline execution engine that could be used in different types of applications including those that require clinical decision support. The architecture is based on the GLIF representation, which is intended as a shared and open format for guidelines.

We intend to further enhance and test the architecture in order to extend its applicability in other settings. The GLIF specification needs further refinement. We will define new Action types and new Patient data types. The logic specification syntax will be enhanced to support operations on new data types. We are enhancing the architecture to provide improved facilities for mapping from Patient Data items to electronic medical records. We propose to use the architecture in other applications such as for supporting conduct of clinical trials.

Acknowledgments

This research was funded by Contract MDA972-94-3-0047 from DARPA and in part by Order 467-MZ-802302 from the National Library of Medicine. We thank Dr. Ohno-Machado for reviewing a draft of this paper.

References

1. Woolf SH, Grol R, Hutchinson A, Eccles M, Grimshaw J. Potential benefits, limitations, and harms of clinical guidelines. *BMJ.* 1999;318:527-530.
2. Weingarten S, Stone E, Hayward R, et al. The adoption of preventive care practice guidelines by primary care physicians: do actions match intentions? *J Gen Intern Med.* 1995;10:138-44.
3. Wolff M, Bower DJ, Marbella AM, Casanova JE. US family physicians' experiences with practice guidelines. *Fam Med.* 1998;30:117-21.
4. Headrick LA, Speroff T, Pelecanos HI, Cebul RD. Efforts to improve compliance with the National Cholesterol Education Program guidelines. Results of a randomized controlled trial. *Arch Intern Med.* 1992;152:2490-6.
5. Nilasena DS, Lincoln MJ. A computer-generated reminder system improves physician compliance with diabetes preventive care guidelines. *Proc Annu Symp Comput Appl Med Care.* 1995:640-5.
6. Lobach DF, Hammond WE. Computerized decision support based on a clinical practice guideline improves compliance with care standards. *Am J Med.* 1997;102:89-98.
7. Zielstorff RD, Teich JM, Paterno MD, et al. P-CAPE: a high-level tool for entering and processing clinical practice guidelines. Partners Computerized Algorithm and Editor. *Proc Amia Symp.* 1998:478-82.
8. Lobach DF, Gadd CS, Hales JW. Structuring clinical practice guidelines in a relational database model for decision support on the Internet. *Proc AMIA Annu Fall Symp.* 1997:158-62.
9. Fox J, Johns N, Rahmanzadeh A. Disseminating medical knowledge: the PROforma approach. *Artif Intell Med.* 1998;14:157-81.
10. Dazzi L, Fassino C, Saracco R, Quaglini S, Stefanelli M. A patient workflow management system built on guidelines. *Proc AMIA Annu Fall Symp.* 1997:146-50.
11. E 1460 Standard Specification for Defining And Sharing Modular Health Knowledge Bases (Arden Syntax for Medical Logic Modules). ASTM Standards v 14.01. American Society for Testing and Materials, Philadelphia; 1992.
12. Shahar Y, Miksch S, Johnson P. The Asgaard project: a task-specific framework for the application and critiquing of time-oriented clinical guidelines. *Artif Intell Med.* 1998;14:29-51.
13. Advani A, Lo K, Shahar Y. Intention-based critiquing of guideline-oriented medical care. *Proc Amia Symp.* 1998:483-7.
14. Musen MA, Tu SW, Das AK, Shahar Y. EON: a component-based approach to automation of protocol-directed therapy. *J Am Med Inform Assoc.* 1996;3:367-88.
15. Ohno-Machado L, Gennari JH, Murphy SN, et al. The guideline interchange format: a model for representing guidelines. *J Am Med Inform Assoc.* 1998;5:357-72.
16. Hripcsak G. Writing Arden Syntax Medical Logic Modules. *Comput Biol Med.* 1994;24:331-63.
17. Wang SJ, Ohno-Machado L, Boxwala A, Mar P. Representing Criteria in Guidelines and Clinical Trial Protocols: Common Needs and Solutions. Technical Report, TR-1999-04. Decision Systems Group, Boston, MA; 1999. http://dsg.harvard.edu/tr/.