Sharable Representation of Clinical Guidelines in GLIF: Relationship with the Arden Syntax

Mor Peleg, Ph.D.[1], Elmer Bernstam, M.D., M.S.E. [1], Aziz A. Boxwala, M.B.B.S., Ph.D.[2], Samson Tu, M.S. [1], Lucila Ohno-Machado, MD, Ph.D.[2], Robert A. Greenes, M.D., Ph.D.[2], Edward H. Shortliffe, M.D., Ph.D.[1, 3]

[1]Stanford Medical Informatics, Stanford University School of Medicine, Stanford, CA
[2]Systems Group, Harvard Medical School, Brigham & Women's Hospital, Boston, MA
[3]Department of Medical Informatics, Columbia University, New York, NY Decision

Abstract

The Guideline Interchange Format (GLIF) and the Arden Syntax are methodologies that were created for the purpose of sharing certain kinds of medical knowledge. While the Arden Syntax is already a standard of the American Society for Testing and Materials (ASTM) and has been used to implement medical decision rules, GLIF is an evolving methodology for representing the logic and flow of clinical guidelines. In this paper we seek to define the relationship between GLIF and the Arden Syntax, highlighting the complementary role that they play in sharing medical knowledge. While the Arden Syntax was designed to represent single decision rules in self-contained units called Medical Logic Modules (MLMs), GLIF specifies entire guidelines that are generally intended to unfold over time. An MLM has a single specification that is computable, while guideline development in GLIF can be done at three different levels of abstraction: an *conceptual flowchart* of medical decisions and actions, a *computable specification* that includes well-defined delineation of decision criteria and patient data, and an *implementable specification*, that includes local adaptations and mapping of guideline variables to institutional databases. The current version of GLIF uses a modification of the Arden Syntax logic grammar to specify logical and temporal decision criteria, but includes additional constructs that support other elements required by complex clinical guidelines. GLIF also includes an MLM-macro class that enables mapping of guideline recommendations into MLMs, in cases where the implementing institution wishes to use MLMs.

1. Introduction

Computer-based best-practice guidelines and decision-support systems for use at the point of care have been proposed as a way to standardize care in order to improve both quality and efficiency. Several groups are working on developing formats for specifying medical guidelines and decision rules that would facilitate their sharing across different institutions. For example, Lobach et al. describe a relational database schema for automating the delivery of clinical practice guidelines using a hybrid of structured and procedural knowledge representation schemes [1]. Another model, PRO*forma*, is based on an object-oriented model as well as a logic-programming approach. Guidelines are constraint satisfaction graphs, where nodes represent "tasks"of clinical actions, decisions, enquiries, or complex plans. Arcs connect tasks within plans. The order of tasks reflect logical, temporal, or other constraints [2]. In a third project, the Asbru language [3] can be used to create a guideline representation that includes the explicit intentions of the guideline's authors. Asbru can be used to represent complex, time-oriented actions and world states, as well as multiple intentions. Yet another guideline model, EON, enables the specification of a guideline through a combination of modeling primitives, such as different types of decision-making mechanisms, control-flow constructs, actions, activities, and exceptions [4].

The Arden Syntax is a standard of the American Society for Testing and Materials (ASTM) that was originally published in April 1992 as ASTM E1460-92. A second version of Arden, Arden 2.0, was developed and published under Health Level Seven, Inc (HL7) [5, 6]. The Arden Syntax is a language for creating and sharing medical knowledge in the form of independent Medical Logic Modules (MLMs). Each MLM represents a single medical decision. Arden Syntax was not originally intended to be used for encoding large, complex clinical practice guidelines. However, encoding of at least some kinds of complex guidelines using Arden Syntax is possible. This was demonstrated by the implementation of a care plan for the post-operative management of patients following coronary artery bypass graft surgery [7]. Nevertheless, there is no support in the Arden specification for aiding in the understanding of the way different MLMs interact with each other. Therefore, the authoring of such guidelines, though possible, is difficult and the resulting interactions among MLMs may be difficult for an observer to understand. Furthermore, some of the recommendations that are given by a guideline concern decisions that should be made by a physician or other health worker and cannot be automated. Consequently, not every guideline recommendation could be mapped into an MLM using the existing Arden Syntax model.

The GuideLine Interchange Format (GLIF) [8] is designed to allow exchange of clinical practice guidelines among institutions and computer-based applications. The GLIF specification consists of an object-oriented model and a text-based syntax for sharing and interchange. The object-oriented model consists of a set of classes for guideline entities, attributes for those classes, and data types for the attribute values. Like Arden, GLIF-encoded guidelines are stored as text files. Version 2.0 of GLIF (GLIF 2.0) was published in 1998 [8]. GLIF 2.0 defined classes for guideline steps, patient data, logical and temporal decision criteria, and action specifications.

Recent work, done by members of the InterMed Collaboratory, has concentrated on the development of the next version of GLIF, called GLIF 3.0 [9]. This version of GLIF (1) incorporates a modification of the Arden Syntax as the language for expression of logical and temporal decision criteria; (2) expands GLIF classes to support representation of several new concepts, such as iteration specification, and patient state; (3) further structures GLIF 2.0 classes, by creating hierarchies of action and decision steps to represent better the different forms of actions and decisions found in medical guidelines [10]; and (4) supports the three different levels of abstraction, to enable specification at conceptual, computable, and implementable levels. GLIF 3.0 supports top-down specification of guidelines. Starting from the textual informal description, it enables the creation of high-level flow charts and their step-wise refinement into a precise and formal specification, which includes logical and temporal decision criteria, actions to be taken (reminders to clinicians, calculations to be preformed) and iteration information [11].

When the GLIF 2.0 specification was published (1998), some observers expressed concern as to whether GLIF was simply a competitor to the Arden Syntax. This paper is aimed at clarifying that Arden and GLIF are related, complementary methodologies for the expression and sharing of medical knowledge. Because Arden Syntax is already an ASTM standard and MLMs have already been implemented by medical institutions we want to leverage the years of effort that have gone into its development. Background material on Arden and GLIF 3.0 is given in section 2. In section 3, we demonstrate how one would model a guideline using GLIF 3.0. We start from a conceptual, top-level flowchart of decisions and actions, and gradually refine the guideline specification to obtain an implementable product. Precise decision criteria are included, and then institution-dependent implementation decisions are made in order to map guideline recommendations into locally usable procedures. Finally, we discuss the relationship between the Arden Syntax and GLIF, showing that the two methodologies have different goals and are not competing methodologies.

2. Background

Arden Syntax for Medical Logic Modules

Named for a New York State retreat at which the initial planning meeting was held, the Arden Syntax [5, 6] was developed in 1989. Developed in response to the inability to share medical decision logic among different institutions, the initial version of the Arden Syntax was based largely upon the encoding scheme used for generalized medical decision support in the HELP system [12]. The medical knowledge that is shared is in the form of individual MLMs, each of which represents a single medical decision. Most MLMs are triggered by clinical events (e.g., admission of a patient, storage of medical data). They then evaluate logical decision criteria, and, if appropriate, perform an action, such as sending a message to a health-care provider. An MLM contains structured slots, grouped into three categories: *maintenance*, *library*, and *knowledge*. The knowledge slot contains the functional components of the MLM. MLMs separate the institution-specific entities, such as mapping of patient data references to fields in an electronic medical record, from the logic of the MLM. The mappings between the institution-specific terms and the MLM's variables are defined within the *data* component of the knowledge slot. The other three main components of the knowledge slot are *evoke*, *logic* and *action* that specify the events that trigger the MLM, the logical criterion that is evaluated, and the action that is performed if the logical criterion holds. These components define the logical rule that the MLM specifies.

Various authoring tools have been developed for writing MLMs. Jenders and Dasgupta created a tool [13] that guides the authoring process in a stepwise manner. MÉDAILLE [14] is an application generated by the PROTÉGÉ-II [15] knowledge-acquisition tool that provides support for entering and editing MLMs. MEDAILLE has a syntax-checker and an integrated terminology. Translation of MLMs into an executable form can be done by a number of compilers [16, 17].

GLIF 3.0

GLIF 3.0 (hereafter simply GLIF) is an object-oriented modeling method that enables exchange of clinical-practice guidelines among institutions and computer-based applications. Like GLIF 2.0, GLIF 3.0 is not an intermediate language that allows interchange from one guideline formalism to another[1]. Instead, it is a single-formalism sharing language [18] (i.e., a language that can be imported and exported by at least two computer programs that execute logic that is encoded in the single formalism). So, after a guideline specification is encoded in GLIF, a variety of tools will be able to import the GLIF code and to view it. Currently, there are two GLIF authoring tools: Protége [19], and the GEODE Authoring Tool [20]. Both of these tools help to visualize the entire guideline as a flow chart. GLIF 2.0-encoded guidelines can be executed by the execution engine described in [21] utilized in clinical decision support applications.

We are exploring the use of a modification of the Arden Syntax as the language for expressing logical and temporal decision criteria. A modification of the Arden Syntax is needed, and not Arden Syntax itself for several reasons:

1. The data model of Arden syntax is much simpler than that used by GLIF. It allows representing each data item as a lst of simple values, each associated with a time stamp. Lists of complex structures are not supported by Arden Syntax, but are required by GLIF. For example, in order to represent a medical concept, GLIF needs both a vocabulary code and the ID of the vocabulary from which the code was taken. Data items refer to medical concepts. A list of data items that represent medical conditions, medications taken, etc. has to be represented as a list of complex objects in GLIF. In Arden, list and the list operators are defined for lists of Arden's basic types, which are null, Boolean, event, destination, message, term, number, time, duration, or string. Lists of complex types and operations on them are not defined in Arden Syntax.
2. Arden syntax assumes that each data item is associated with a time stamp, which represents the medically relevant time of occurrence (**primary time**). The data model that we are using in GLIF allows expressing time intervals in addition to time stamps. In addition, data items may have different associated times with them. The structure of data items is defined by a Reference Information Model (RIM). Different RIMS may be used, for example, HL7's Unified Service Action Model (USAM) [22]. In USAM several attributes for time may be associated with a data item: the activity time interval that represents the time during which the medical action took place, the critical time interval that represents the biological-relevant time period for the action, and the recording time of the action. Therefore, Arden's *latest* operator is meaningless if more than one timestamp is associated with each data item. Instead we need to allow temporal operations based on every timestamp that can be associated with a data item.
3. Because of the difference in the data models of GLIF and Arden, as summarized in the two points above, other operators should be dedined in addition to the existing Arden operators. These operators include "overlaps" that is needed to support the interval-based time semanitcs, "is unknown" that is needed because

[1] In this sense, the word "interchange" in the exp ansion of the GLIF acronym is a misnomer. The goal is actually to have guidelines authored and shared in GLIF format and to support adaptation of GLIF-encoded guidelines for a variety of local uses and representations.

GLIF distinguishes between null and the truth-value *unknown* and "at least *k* of …", which is a type of criterion that was introduced by GLIF2.

We propose two possible expression languages for GLIF, which are both based on Arden Syntax.

1. An expression language that supports lists (and list operations) of GLIF's basic types that are not supported by Arden (Extended_Boolean, integer, float, data_item, set of GLIF's basic type, and list of GLIF's basic types, concept). We call this expression language Guideline Expression Language (GEL)
2. An object-oriented expression language that associates functions with each class in GLIF's data model and RIM. This expression language has several predefined classes, such as List and Math that contain Arden's current operators. For example, *merge* is an operator of the List class, whereas *sin* is an operator of the Math class. We call this expression language Object-Oriented Gueiline Expression Language (OO-GEL).

GLIF enables the creation of a guideline specification in a top-down manner, starting from an imprecise, potentially incomplete conceptual view of the guideline, progressing to a fully detailed, precise and computable specification, and subsequently to an implementable specification where data and actions are mapped to institution-specific data and procedures. When medical experts author a guideline using GLIF, they can start from a description of the guideline written in prose that they can transform into a flowchart showing a temporally ordered sequence of steps. Different types of steps represent clinical actions or decisions, the specification of control flow enabling the expression of iteration, concurrency and synchronization. An alternative or complementary conceptualization and authoring approach is to start with a high level specification of major steps and their flow; particular steps may then be expanded and more fully detailed in an iterative manner. The information contained in the guideline steps is mostly textual and unstructured. The guideline is refined from its text-based level by adding a structured specification that corresponds to the text to obtain a computable level of the guideline specification. A team of an informatician aided by a medical domain expert usually performs this process. Finally, at the implementation level, the guideline is mapped into data and procedures used by the implementing institution.

GLIF 3.0 is designed to support specification of large and complex guidelines. A nesting mechanism is used for managing the complexity of large guidelines. Nesting enables specifying a guideline at a high level. The high level guideline can then be elaborated through the use of sub-guidelines. In addition to aiding in management of complex guidelines, nesting may also be useful in showing the relationship to other guidelines by creating a top-level view that makes explicit such relationships. Since nesting allows grouping of parts of a guideline into a single unit, this is a mechanism that can allow model extensibility and reuse of part of a guideline as well as the adaptation of a guideline to a specific institution by replacement or elaboration of specifications for well-defined sections of a guideline (i.e., replacing a goal with a procedure).

A new feature in GLIF is the *macro step*. Like Visual Basic, Object linking and embedding Custom Control (OCX), and Java Beans, a Macro step is a special class that has attributes that define the information that is needed to instantiate a set of underlying GLIF steps that represent a pattern that appears in clinical guidelines. For example, a risk-assessment guideline follows a pattern of data-collection steps, followed by risk calculation, followed by recommendations that are based on the risk profile. Similarly, an MLM could be described using a pattern of GLIF components: an event (trigger), followed by a decision step (logic slot), followed by action steps (action slot). Using macro steps to represent commonly occurring patterns has benefits for authoring, visual understanding, and execution of guidelines. Macros allow a way to specify declaratively a procedural pattern that is realized by a set of action, branch, or decision steps.

GLIF is intended to support specification of guidelines that differ in their (1) clinical area (i.e., the disease/condition and the clinical field addressed); (2) guideline category (e.g., screening, disease management); (3) intended users (e.g., physician, patient); and (4) setting (i.e., ICU, out of hospital) [23]. GLIF addresses several important issues such as adaptations of encoded guidelines for institutional or local use. In order for guidelines to be adapted for such use, two types of mappings need to be defined. The first is the mapping between the domain ontology used by GLIF and institutional term dictionaries (or to procedures used to obtain the corresponding data elements). The second is a mapping between actions and recommendations in the guideline and the institutional procedures used to carry out these actions.

3. The Stable Angina Guideline Case Study

We use the American College of Cardiology/American Hospital Association/American College of Physicians-American Society of Internal Medicine (ACC/AHA/ACP-AISM) guidelines for the management of patients with chronic stable angina as a case study [24] to demonstrate how GLIF can be used to model a complex guideline, and how it integrates the Arden Syntax. The integration of Arden Syntax is done in two forms: (1) logical and temporal decision criteria are specified using a superset of the Arden Syntax logic grammar; and (2) MLM-macros are used to map guideline declarations into a procedural specification that defines evoke, logic and action slots, that correspond to respective MLM slots.

Figure 1 shows the top-level flow-chart view of the guideline that was easily created (in this case using PROTÉGÉ) from the textual description and accompanying flowchart supplied by the guideline authors. When a guideline specification is created in GLIF, this is usually the first approximation of the guideline specification. It is still incomplete and imprecise. The information contained in the guideline steps is, for the time being, text only.

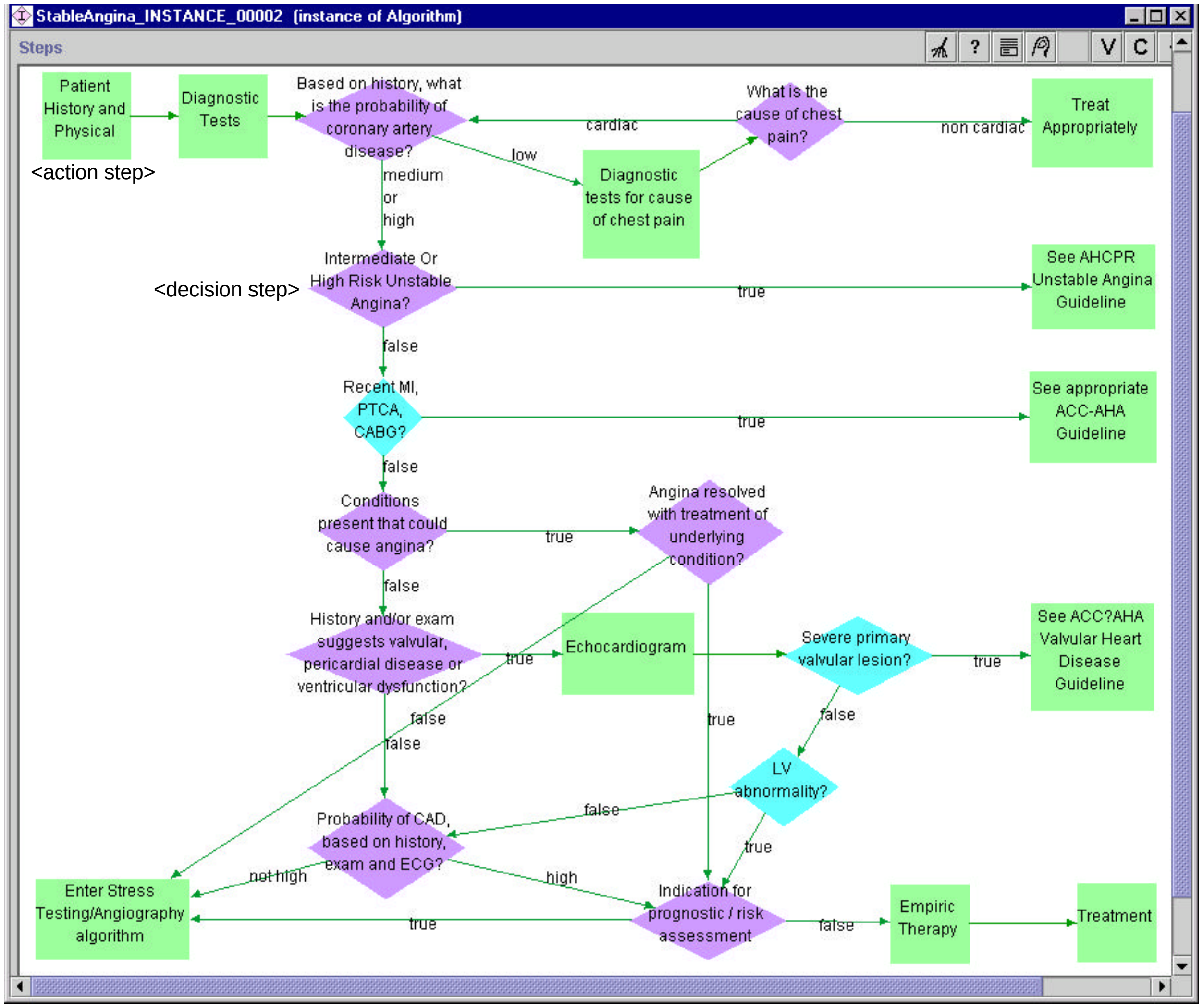

StableAngina_INSTANCE_00002 (instance of Algorithm)
Steps
Patient History and Physical
<action step>
Diagnostic Tests
Based on history, what is the probability of coronary artery disease?
cardiac
low
medium or high
Diagnostic tests for cause of chest pain
What is the cause of chest pain?
non cardiac
Treat Appropriately
<decision step>
Intermediate Or High Risk Unstable Angina?
true
See AHCPR Unstable Angina Guideline
false
Recent MI, PTCA, CABG?
true
See appropriate ACC-AHA Guideline
false
Conditions present that could cause angina?
true
Angina resolved with treatment of underlying condition?
false
History and/or exam suggests valvular, pericardial disease or ventricular dysfunction?
true
Echocardiogram
Severe primary valvular lesion?
true
See ACC?AHA Valvular Heart Disease Guideline
false
false
true
false
LV abnormality?
false
true
Probability of CAD, based on history, exam and ECG?
not high
high
Enter Stress Testing/Angiography algorithm
true
Indication for prognostic / risk assessment
false
Empiric Therapy
Treatment

Figure 1. A top-level abstract view of the stable angina guideline

The guideline is refined from its text-based level by adding a structured specification that corresponds to the text. When decision criteria are used, they are specified in a modified Arden Syntax. This refinement turns the guideline into a computable specification. An example of such refinements can be seen in Figure 2, where a specification is given for the abstract decision “Conditions present that can cause angina?” that was stated in text at the top-level view of the guideline specification (Figure 1).

Decision criterion specifications for:
"Conditions present that can cause angina?"

a) GEL expression:

```
is in(severe_anemia, Problem_List) OR is in(hyperthyroidism, Problem_List) …
```

b) OO-GEL expression:

```
Problem_list.contains(severe_anemia) OR Problem_list.contains(hypertension)…
```

Figure 2. Refining the top-level specification. Replacing the textual abstract description of "Conditions present that can cause angina?" with a decision criterion specification. Problem_List is a list of variable data items that correspond to the current problems that a patient has. It is created during the patient history collection step. severe_anemia and hyperthyroidism are literal data items. "is in" is a list operator in GEL that checks for membership of the left argument in the right argument, which is a list. "contains" is a List function in OO-GEL that checks for membership of the parameter in the list.

At the implementation level, the guideline is mapped into data and procedures used by the implementing institution. An example of mapping into procedures is shown in Figure 3. In this example, the implementing institution has chosen to implement the decision step "What is the cause of chest pain", of Figure 1, as a reminder to the physician to determine the cause of chest pain when the results of diagnostic tests are available. GLIF uses MLM-macros to implement reminders. The guideline modeler needs to fill in *evoke*, *logic*, and *action* slots of the MLM-macro. The macro can be replaced by a sequence of two GLIF guideline steps that do not include macros. The first of these two steps is a decision step that references instances of two GLIF classes – *event* class and *criterion* – that correspond to evoke and logic slots of the MLM-macro. The decision step is followed by an action step that references an instance of the GLIF class *message action-specification* that corresponds to the action slot of the MLM-macro.

Note that the guideline does not specify which diagnostic tests for determining the cause of chest pain should be performed. Therefore we would like to define the evoking event of the MLM-macro as "storage to the medical record of diagnostic results of a test for non cardiac cause of chest pain", meaning that no matter what tests were chosen to be ordered by the physician, when the results of one of these tests returns, the MLM-macro is evoked. The issue of mapping this kind of event in a standard and portable manner into an institutional database is difficult, and has not yet been resolved by either GLIF or Arden. Therefore, the example in Figure 3 assumes that the diagnostic test consists of the levels of hemoglobin, fasting glucose, and fasting lipid panel (total cholesterol, HDL cholesterol, triglycerids, and calculated LDL cholesterol).

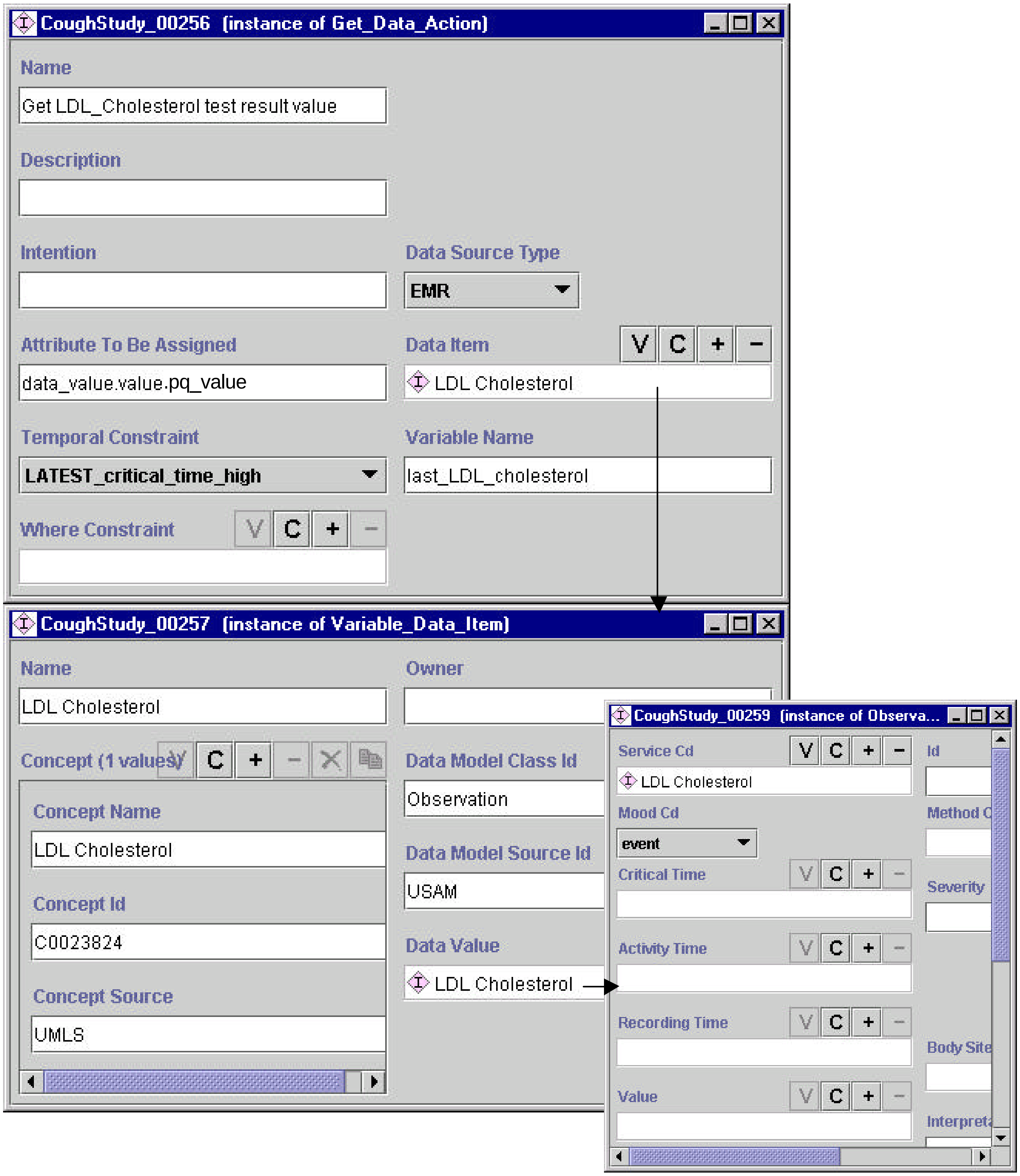
```
                    Reminder: decide the cause of chest pain (MLM-macro)
evoke:  Event:
              event_name: hemoglobin_storage        event_type: patient_data_availability
        Event:
              event_name: fasting_glucose_storage event_type: patient_data_availability
        Event:
              event_name: fasting_lipid_storage     event_type: patient_data_availability
logic:  Criterion:
        is number(last_hemoglobin ) and (last_hemoglobin_end_time > (now – 12 hour)) and
        is number(last_fasting_glucose) and (last_fasting_glucose_end_time > (now – 12 hour))
        and is number(last_total_cholesterol) and (last_total_cholesterol_end_time > (now – 12
        hour)) and
        is number(last_HDL_cholesterol) and (last_HDL_cholesterol_end_time > (now – 12
        hour)) and
        is number(last_triglycerid) and (last_triglycerid_end_time > (now – 12 hour)) and
        is number(last_calculated_LDL_Cholesterol) and
        (last_calculated_LDL_Cholesterol_end_time > (now – 12 hour))
action: Message_Action
         message: Diagnostic tests results for cause of chest pain are back. Please decide
                  cause of patient's chest pain
         destination:  mail_dest;
```

Figure 3. Mapping to procedures used by the implementing institution. The institution creates a reminder for the decision on the cause of chest pain. The ***is number*** *operator returns* ***true*** *if the argument's data type is a number. Otherwise it returns* ***false****. The logic criterion is written in GEL.*

last_hemoglobin, last_fasting_glucose, last_total_cholesterol, last_HDL_cholesterol, last_triglycerid, and last_calculated_LDL_cholesterol are variables that represent test results that are extracted from data items that represent the test results. last_hemoglobin_end_time, last_fasting_glucose_end_time, last_total_cholesterol_end_time, last_HDL_cholesterol_end_time, last_triglycerid_end_time, and last_calculated_LDL_cholesterol_end_time are variables that represent test results that are extracted from data items that represent the end of the time interval that represents the biologically-relevant time of the test.

GLIF has a Get_Data_action that is used to extract from data items, retrieved from the EMR, variables that can be used by the expression language. The Get_Data_Action assigns an object to a variable whose name is specified by *variable_name*. The object that is assigned is selected from a list of data items that correspond to the data item declaration given by the *data_item* attribute of Get_Data_Action. The list is "trimmed" by two constraints. One is a temporal constraint operator that selects one element from the list, that corresponds to the latest or current data_item, where latest and current relate to one of the time stamps that can be associated with the data item. The other constraint is a *where constraint*, that specifies an Arden criterion that the data_item must satisfy in order to be selected (and not trimmed away). An example for a where constraint is: "where concept is in (hypertension, severe_anemia)". Figure 4 shows how the last_HDL_cholesterol variable is extracted from the LDL_Cholesterol data item.

Note that we could not use the Arden expression "**is number(last_HDL_cholesterol where it occurred within the past 12 hours)",** because Arden's *occurred* operator assumes that a single time stamp is associated with each data item. GLIF's Get_Data_Action has a "where_constraint" attribute, where Arden expressions can be written. However, we cannot write expressions that assume multiple time stamps or complex data structure for a data item, for example, we cannot write "data_value.critical_time.high > (now – 12 hours)". As a consequence the use of a *where* clause is limited and might not accommodate all criteria easily.



CoughStudy_00256 (instance of Get_Data_Action)

Name: Get LDL_Cholesterol test result value

Description:

Intention:

Data Source Type: EMR

Attribute To Be Assigned: data_value.value.pq_value

Data Item: LDL Cholesterol

Temporal Constraint: LATEST_critical_time_high

Variable Name: last_LDL_cholesterol

Where Constraint:

CoughStudy_00257 (instance of Variable_Data_Item)

Name: LDL Cholesterol

Owner:

Concept (1 values)

Concept Name: LDL Cholesterol

Concept Id: C0023824

Concept Source: UMLS

Data Model Class Id: Observation

Data Model Source Id: USAM

Data Value: LDL Cholesterol

CoughStudy_00259 (instance of Observa...

Service Cd: LDL Cholesterol

Id:

Mood Cd: event

Method C

Critical Time:

Severity

Activity Time:

Recording Time:

Body Site

Value:

Interpret

Figure 4. Extraction of the last_HDL_cholesterol variable from the LDL_Cholesterol data item. Note that the Data_Value attribute of the LDL_Cholesterol variable data item contains a declaration of a data item. It does not represent an actual data item instance that is retrieved from

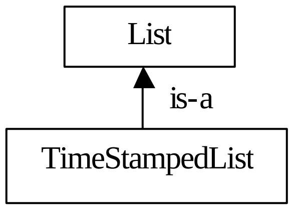
the electronic medical record. Therefore, there are no values for the critical time, activity time, recording time, and the observation value.

The same logic criterion of Figure 3 can be written in OO-GEL in the following way:

```
sorted_HDL_cholesterol := HDL_cholesterol.sortDescending
(Observation.getComparator(CRITICAL_TIME_HIGH));
latest_hdl_cholesterol := sorted_HDL_cholesterol.first();
conclude (latest_hdl_cholesterol.critical_time.high < (now-12 hours))
```

Figure 5. Object-oriented syntax for the criterion "is number(latest HDL_cholesterol) where it occurred within past 12 hours". WE assume that Get_Data assigns a list of Observations to the variable name HDL_Cholesterol. The first line produces a sorted list of HDL_cholesterol from HDL_cholesterol. The sort is performed with the aid of a comparator function that uses the critical_time.high attribute of the Observation class. The sortDescending function is also defined in the Observation class. The latest value of HDL cholesterol is extracted from this sorted list. The time stamp on the latest value is then checked to see if it occurred within the last 12 hours.

Temporal sorting can be performed using Arden operators such as *latest* if a class in the RIM can implement an interface, in Java like fashion. In this example, the Observation class defines a function called isLaterThan(Observation anotherObsInstance) that compares "this" object to another instance of an Observation. The TimeStampedList class can contain a list of observations. The *latest* function of the TimeStampedList class calls the isLaterThan function of each of its elements to perform a sort.



4. Discussion

We have described the relationship between the Arden Syntax and GLIF. Arden Syntax was developed in 1989 for the purpose of sharing single decision rules, each specified by an MLM, among research centers. Arden Syntax is very useful for sharing medical knowledge that can be expressed by a single MLM, for example, for generating warnings about drug interactions or reminders and alerts regarding abnormal lab results. Arden Syntax was not originally intended to be used for encoding large, complex medical guidelines that unfold prospectively over time. Therefore, although the encoding of complex guidelines as a set of MLMs is not an impossible task, it is difficult and not particularly natural, either to conceptualize or to implement. There is no support in the Arden specification to aid in the understanding of the way different MLMs interact with each other. Furthermore, some of the recommendations that are given by a guideline concern decisions that could not be mapped into MLMs.

GLIF is an evolving methodology that is being developed by the InterMed Collaboratory. Work on GLIF began in 1996 with the explicit goal of sharing guidelines. GLIF 3.0 unites the modeling approaches taken by the Arden syntax for MLMs and GLIF 2.0. This enables the creation of a model that captures the algorithmic guideline logic in a way that is natural and computable. GLIF 3.0 is very much different from MLMs that represent single decision rules and from GLIF 2.0 models that represented guidelines in a way that is not computable, since text slots represented a great part of the guideline logic in that earlier version.

Both the Arden Syntax and GLIF try to solve the problem of creating a machine-interpretable guideline specification that could be locally adapted by an institution and integrated into institutional clinical systems. The Arden syntax defines very specific ways in which MLMs are triggered and recommendations are delivered to users. These implementation-related choices are specified as part of the MLM's knowledge slot. An institution that uses an MLM has to commit to these guideline-delivery choices. GLIF, on the other hand, supports three levels of abstraction, so a guideline may be modeled on the conceptual and computable levels without making implementation commitments. This way, a GLIF-encoded guideline may be more easily shared among different institutions, since a large part of the conceptual and computable levels may not change across different implementations by different institutions.In order for the GLIF specification to be computable and implementable, it must be refined from its top-level, conceptual flow-chart representation. In order to do so, three things must be done: (1) imprecise textual descriptions have to be given precise and complete specifications; (2) literals used by the top level GLIF specification need to be mapped to specific data elements used by the implementing institutions; and (3) decisions need to be made as to how to turn a declarative specification into a procedure used by the implementing institution. The first of these refinement tasks involves further specification of action and decision steps. Actions that were vaguely described are refined to include well-defined tasks, such as scheduling, or referrals. Patient data that need to be accessed by action and decision steps are specified, including specification of units and ranges, allowed values, etc., as well as temporal constraints on the data elements. We have two alternative expression languages for specifying decision criteria, which are both based on Arden Syntax. Mapping literals is not supported yet, but we intend it to be done by augmenting the domain ontology used by GLIF. Turning a declaration into a procedure is done in two stages. First, the nesting/macro mechanisms of GLIF enable specifying specific procedures that should be done locally in the implementing institution. Then, these procedures have to be mapped into the local institution's clinical information system's functions. It is important to note that local adaptation may not simply involve a refinement of declarations into procedures but may also change the overall guideline logic [25].

GLIF-based guidelines may be implemented in a variety of possible ways, among which could be compilation or interpretation from GLIF into executable actions, or translating them into sequences of MLMs. For the latter, it is important to stress that the process of generating a set of MLMs from a GLIF-encoded guideline depends on many decisions that have to be made along the way, which may not be technical in nature. One reason is that not all of the GLIF 3.0 guideline steps may be desirable to be converted into MLMs-macros; relevant guideline steps should thus be chosen carefully. But even once it is decided that a guideline step should be mapped into an MLM-macro, more decisions remain to be made. For example, a guideline may specify that "the patient temperature has to be measured every 2 hours", but will not tell you how this information should reach the user. Should care-providers be reminded every two hours right before (and how long before) this action needs to take place? Should care-providers be reminded only if they do not perform the action on time? The local institutions must make these decisions in order to implement the guideline. Once these decisions are made, MLM-macros can be added to the GLIF-encoded guideline, and only then might it be possible to generate automatically a set of MLMs that correspond to the GLIF-encoded guideline.

Although it has been suggested by some that GLIF is intended to replace or compete with Arden, we see the Arden Syntax and GLIF as complementary standards with different strengths, limitations, and goals. Arden is designed to represent efficiently multiple simple guidelines and events, but representing complex, algorithmic guidelines in Arden is difficult. An explicit goal of the design of GLIF, however, is the expression of large, complex guidelines. GLIF may be cumbersome, however, for the expression of single medical decisions leading to an action. In fact, GLIF now relies on Arden Syntax for expression of criteria and other critical functions. Recognizing that Arden MLMs are very well suited to expressing single medical decisions, GLIF provides a mechanism to invoke MLMs from GLIF encoded guidelines. In summary, we feel that both Arden and GLIF have important roles in the representation of shareable clinical-practice guidelines.

References

1.Lobach DF, Gadd CS, Hales JW. Structuring clinical practice guidelines in a relational database model for decision support on the Internet. In: Proc AMIA Annu Fall Symp; 1997; 1997. p. 158-62.

2.Fox J, Rahmanzadeh A. Disseminating medical knowledge: the PROforma approach. *Artificial Intelligence in Medicine* 1998;14:157-181.

3.Shahar Y, Miksch S, Johnson P. The Asgaard Project: A Task-Specific Framework for the Application and Critiquing of Time-Oriented Clinical Guidelines. *Artificial Intelligence in Medicine* 1998;14:29-51.

4.Tu SW, Musen MA. A Flexible Approach to Guideline Modeling. In: AMIA Symp; 1999; 1999. p. 420-424.

5.Hripcsak G, Ludemann P, Pryor TA, Wigertz OB, Clayton PD. Rationale for the Arden Syntax. *Comput Biomed Res* 1994;27(4):291-324.

6.Clinical Decision Support & Arden Syntax Technical Committee of HL7, inventor Arden Syntax for Medical Logic Systems, version 2.0. Draft revision. USA. 1999 July 7, 1999.

7.Starren J, Hripcsak G, Jordan D, Allen B, Weissman C, Clayton PD. Encoding a post-operative coronary artery bypass surgery care plan in the Arden Syntax. *Comput. Biol. Med.* 1994;24(5):411 - 417.

8.Ohno-Machado L, Gennari JH, Murphy S, Jain NL, Tu SW, Oliver DE, et al. The GuideLine Interchange Format: A Model for Representing Guidelines. *Journal of the American Medical Informatics Association* 1998;5(4):357-372.

9.Peleg M, Boxwala A, Ogunyemi O, Zeng Q, Tu S, Lacson R, et al. GLIF3: The Evolution of a Guideline Representation Format. In: Proc. AMIA Annual Symposium; 2000; 2000. p. 645-649.

10.Boxwala AA, Peleg M, Greenes RA, Shortliffe EH, Patel VL. Functional requirements for a shared guideline representation; 1999. Report No.: IM-2000-1.

11.Greenes RA, Shortliffe EH. Representation of Clinical Practice Guidelines to Facilitate Sharing; 1999 November 12, 1999. Report No.: IM-2000-4.

12.Kuperman GJ, Gardner RM, Pryor TA. HELP: A Dynamic Hospital Information System. NY: Springer-Verlag Inc.; 1991.

13.Jenders RA, Dasgupta B. Assessment of a knowledge-acquisition tool for writing Medical Logic Modules in the Arden Syntax. In: Proc AMIA Annu Fall Symp 1996; 1996; 1996. p. 567-71.

14.Bang M, Eriksson H. Generation of development environments for the Arden Syntax. In: Proc AMIA Annu Fall Symp; 1997; 1997. p. 313-317.

15.Musen MA, Tu SW, Eriksson H, Gennari JH, Puerta AR. PROTEGE-II: An Environment for Reusable Problem-Solving Methods and Domain Ontologies. In: International Joint Conference on Artificial Intelligence; 1993; Chambery, Savoie, France; 1993.

16.Hripcsak G, Cimino JJ, Johnson SB, Clayton PD. The Columbia-Presbyterian Medical Center decision-support system as a model for implementing the Arden Syntax. In: Proc Annu Symp Comput Appl Med Care; 1991; 1991. p. 248-52.

17.Kuhn RA, Reider RS. A C++ framework for developing Medical Logic Modules and an Arden Syntax compiler. *Comput Biol Med* 1994;24(5):365-70.

18.Shwe M, Sujansky W, Middleton B. Reuse of Knowledge Represented in the ARden Syntax. In: Proceedings of the Sixteenth Annual Symposium on Computer Applications in Medical Care; 1992 November 1992; 1992. p. 47-51.

19.Grosso WE, Eriksson H, Fergerson R, Gennari JH, Tu SW, Musen MA. Knowledge Modeling at the Millennium (The Design and Evolution of Protege-2000). In: Gains BR, Kremer R, Musen M, editors. The 12th Banff Knowledge Acquisition for Knowledge-Based Systems Workshop.; 1999; Banff, Canada; 1999. p. 7-4-1 to 7-4-36.

20.Greenes RA, Boxwala A, Sloan WN, Ohno-Machado L, Deibel SR. A framework and tools for authoring, editing, documenting, sharing, searching, navigating, and executing computer-based clinical guidelines. In: Proc AMIA Symp 1999; 1999; 1999. p. 261-265.

21.Boxwala AA, Greenes RA, Deibel SR. Architecture for a multipurpose guideline execution engine. In: Proc AMIA Symp; 1999; 1999. p. 701-705.

22.Schadow G, Russler DC, Mead CN, McDonald CJ. Integrating Medical Information and Knowledge in the HL7 RIM. In: Proc. AMIA Annual Symposium 2000; 2000; 2000. p. 764-768.

23.Bernstam E, Ash N, Peleg M, Tu S, Boxwala AA, Mork P, et al. Guideline classification to assist modeling, authoring, implementation and retrieval. In: Proceedings of the American Medical Informatics Association (AMIA) Annual Symposium,; 2000 November 2000; Los Angeles, CA,: American Medical Informatics Association; 2000. p. 66-70.

24.American College of Cardiology/American Heart Association/American College of Physicians-American Society of Internal Medicine. Guidelines for the Management of Patients with chronic Stable Angina. *J Am Col Cardiol* 1999;33:2092-2197.

25.Fridsma DB, Gennari JH, Musen MA. Making generic guidelines site-specific. In: Proc AMIA Annu Fall Symp 1996; 1996; 1996. p. 597-601.