# GLIF3: A Representation Format for Sharable Computer-Interpretable Clinical Practice Guidelines

Aziz A. Boxwala[4,1], Mor Peleg[5], Samson Tu[2], Omolola Ogunyemi[1], Qing T. Zeng[1], Dongwen Wang[3], Vimla L. Patel[3], Robert A. Greenes[1], Edward H. Shortliffe[3]

[1]Decision Systems Group, Brigham and Women's Hospital, Harvard Medical School, Boston, MA

[2]Stanford Medical Informatics, Stanford University, Stanford, CA

[3]Department of Biomedical Informatics, Columbia University, New York, NY

[4]Eclipsys Corporation, Boston, MA

[5]Department of Management Information Systems, University of Haifa, Israel

Address Correspondence to:
Aziz A. Boxwala
Eclipsys Corporation
1550 Soldiers Field Road
Boston, MA 02135
Tel: 617-779-4954
Fax: 617-787-5964
aziz.boxwala@eclipsys.com

Running Title: GLIF3 guideline representation

## Abstract

The Guideline Interchange Format (GLIF) is a model for representation of sharable computer-interpretable guidelines. The current version of GLIF (GLIF3) is a substantial update and enhancement of the model since the previous version (GLIF2). GLIF3 enables encoding of a guideline at three levels: a conceptual flowchart, a computable specification that can be verified for logical consistency and completeness, and an implementable specification that is intended to be incorporated into particular institutional information systems. The representation has been tested on a wide variety of guidelines that are typical of the range of guidelines in clinical use. It builds upon GLIF2 by adding several constructs that enable interpretation of encoded guidelines in computer-based decision-support systems. GLIF3 leverages standards being developed in Health Level 7 in order to allow integration of guidelines with clinical information systems. The GLIF3 specification consists of an extensible object-oriented model and a structured syntax based on the Resource Description Framework (RDF). Empirical validation of the ability to generate appropriate recommendations using GLIF3 has been tested by executing encoded guidelines against actual patient data. GLIF3 is accordingly ready for broader experimentation and prototype use by organizations that wish to evaluate its ability to capture the logic of clinical guidelines, to implement them in clinical systems, and thereby to provide integrated decision support to assist clinicians.


Keywords: knowledge representation, guidelines, knowledge sharing

# 1 Introduction

Clinical practice guidelines are intended to improve the quality and cost effectiveness of patient care by fostering best practices [1]. Such guidelines seek to reduce demonstrated unexplained practice variations among providers, and a surprisingly high incidence of sub-optimal care and medical errors [2]. Unfortunately, the promulgation of guidelines has not been optimally effective in changing clinical practice [3, 4]. One likely reason is the inherent limitation of the prevalent mode of dissemination of guidelines as text documents, with their clinical impact mainly dependent on education of providers. Reliance on educational approaches puts primary emphasis on human memory and thus limits the ability of guidelines to effect changes in clinical practice. Publication of text guidelines on the Internet or a practice's intranet can improve access to guidelines at the point of care [5]. But even if they are available when needed, they are typically not custom-tailored for a specific patient and they require reading and interpretation during a busy clinical schedule, which limits their usefulness.

Studies have shown that computer-based decision-support interventions such as automated reminders and alerts to providers are beneficial and can change clinical practice [6]. Computer-based physician order entry (CPOE)is another point of contact where decision support has been shown to be effective [7]. Similarly, decision-support systems have been utilized successfully to deliver patient-specific guideline recommendations at the point of care [8] although such systems have not been used widely and have typically depended on the local development of guidelines with idiosyncratic methods for introducing them into the clinical systems..

As we have noted, for guidelines to be most useful for decision-support, they need to be integrated into the patient care process and to be patient-specific. To achieve this, the clinical recommendations in guidelines must be encoded in a form that enables automated access to stored clinical data and unambiguous execution of their decision logic, i.e., they must support computer-based interpretation. We refer to guideline recommendations that are encoded for interpretation by computer programs as computer-interpretable guidelines (CIGs). Well-developed evidence-based guidelines are expensive to develop. The additional effort required to encode guidelines rigorously in a logically-consistent, unambiguous, and complete computer-interpretable format imposes further demands on guideline developers.

Further, even guidelines that are encoded for computer interpretation often cannot be shared across institutions or even across different types of applications within an institution, due in part to differences in formats or languages that are used to represent the guideline knowledge, in data models and terminologies used within the encoded guidelines, and in clinical information systems with which they are integrated. Without sharing of guidelines, each institution, vendor, or research group seeking to implement approved guidelines via decision-support systems must redo the work involved in making them executable. Moreover, we have shown in other work [9, 10] that this process of encoding and implementing CIGs is not straightforward, and often discloses differences in interpretation. The problem is compounded because guidelines are rarely static, and must be updated as medical knowledge changes. This means that implementations dependent on them must be also updated.

A common representation format for the clinical knowledge in guideline recommendations is accordingly needed to foster sharing and implementation of encoded CIGs. Such sharing of CIGs potentially could reduce the cost and effort of implementing them in dissimilar clinical information systems.

A challenge in the development of a common representation format for CIGs is the variety of applications within clinical information systems that may be used to implement guideline-based decision support. These include consultation-advice dialogue applications [11], protocol-based data entry (e.g., for clinical trials or for other standardized care settings, such as dialysis or transplant care) [12], event-driven alerts and reminders [13, 14], background quality monitoring and assessment [15], interaction and constraint-checking in order-entry systems [16], and applications aimed at facilitating workflow involving coordination of multiple participants[17, 18]. These varying application purposes, as well as others, have differing functional requirements (e.g., timing constraints in work-flow applications) that have led modelers of CIGs to develop application-specific modeling languages.

The GuideLine Interchange Format (GLIF) was created by the InterMed collaboratory, a joint project of biomedical informatics groups at Harvard, Columbia, and Stanford universities, to serve as a common representation format for CIGs [10]. Version 2 of GLIF (GLIF2) was an incomplete specification with respect to computer-interpretation and execution of guideline knowledge. Among the major deficiencies that we identified [19] in GLIF2 were:

1. It did not specify a structured representation for important attributes of guidelines and guideline steps, such as patient data to be acquired, recommended actions,

and decision conditions. Values of most attributes were specified simply as text strings. Such guidelines could not be interpreted reliably by the computer.

2. GLIF2 provided only a limited set of low-level flow-control constructs. Important concepts such as those for describing iteration, patient-state, exception conditions, and events were lacking.

3. Integrating GLIF2 guidelines with heterogeneous clinical systems was difficult, as GLIF2 lacked features for mapping patient data references to entries in the electronic medical record (EMR).

The limitations of the GLIF2 model and the considerations pertinent to creating sharable guidelines are discussed in more detail elsewhere [20].

GLIF3 [19] is a revision of GLIF that attempts to overcome several of GLIF2's limitations described above. The significant modifications in GLIF3 include

1) Those providing more structure to the representation

   a) Specification of classes whose instances form values of the attributes of existing GLIF2 and new GLIF3 classes.

   b) Addition of object-oriented expression and query languages.

2) Those improving the flow-control:

   a) Revision of classes for describing decisions (the *Decision_Step* class) and parallel flows (*Branch_Step* and *Synchronization_Step* classes).

   b) Addition of a new class for specifying patient states (*Patient_State_Step* class).

   c) Addition of classes for specifying iterations, events, and exceptions.

3) Those enabling interfacing GLIF guidelines with clinical information systems:
    a) Addition of a layered model to facilitate interfacing to patient data and external medical knowledge during guideline execution
4) Those adopting open standards:
    a) Use of Resource Description Framework (RDF) [21] as the exchange syntax for guidelines.
    b) Use of Health Level 7 (HL7) standards and the ability to incorporate standard medical terminologies.
5) Incorporation of features to manage complexity of large guidelines, which became an important requirement with a larger more complex representation model.

With the addition of these features, it is now possible to encode guideline recommendations in GLIF3, resulting in a specification that can then be interpreted by a computer program. Indeed, an execution engine has been developed for the latter purpose [22]. The encoded guidelines and the engine were validated by testing them against clinical data. Columbia University informatics researchers are working to enhance the decision-support capability of clinical information system at the New York-Presbyterian Hospital by integrating the GLIF3 execution engine with that system.

In order to encode executable guidelines in GLIF, an individual must have a comprehensive understanding of the structure and semantics of the GLIF3 model. Previous publications have described the motivation for creating GLIF [10], the approach to the development of the model [20], and the specific enhancements made in GLIF3 [19]. This paper provides an overview of the entire GLIF3 structure. This paper

also serves to describe, briefly and informally, the semantics of GLIF3. In doing so, we delineate the representational features and capabilities of GLIF3 using a cough management guideline from the American College of Chest Physicians [23] as an illustrative example. A comprehensive description of the GLIF3 model is not sutiable for journal publication; we accordingly offer a detailed technical specification [24] on our web site at http://www.glif.org.

# 2 GLIF3 Representation

We discuss the GLIF3 representation in terms of (1) the considerations that influenced its design; (2) the components and features of the GLIF3 model and how they are used to encode guidelines; and (3) the RDF file format used for storage and exchange of GLIF3-encoded guidelines.

## *2.1 Considerations that influenced the design of GLIF3*

A number of considerations influenced the design of GLIF, derived from encoding requirements, potential applications, and implementation of CIGs.

### 2.1.1 Representation of guidelines for different applications

GLIF is intended for representation of different types of guidelines such as those for screening, diagnosis, and treatment, during primary or specialty care, and in acute or chronic problems. Hence, the GLIF guideline model consists of a number of low-level primitives that give it the expressivity to specify this range of guideline types.

### 2.1.2 Ability to execute

GLIF guidelines are designed to be executable in decision-support environments. The GLIF3 representation provides formal structure for the attributes and classes defined in GLIF2. For example, GLIF3 includes an explicit data model and an expression language for decision criteria.

### 2.1.3 Integration into clinical workflow

Beyond satisfying the requirement of being computer-interpretable, CIGs must be integrated into clinical workflow and into clinical information systems. GLIF3 includes a structured data model that is based on the Reference Information Model standard from Health Level 7 (HL7) [25] for the purpose of facilitating mapping of data needed by guidelines to entries in the EMR. Similarly, GLIF3's structured hierarchy of action specifications is designed to facilitate mapping of guideline recommendations to implementable clinical actions, for example, via order-entry systems. Further, an event model improves workflow integration by enabling guideline steps to be triggered asynchronously in response to clinical events.

### 2.1.4 Management of complexity

A CIG representation model should make it easy to encode and to understand CIGs. This need may often conflict with making the model expressive by supporting a large number of constructs. In order to overcome this conflict, the GLIF model attempts to manage complexity in different ways such as (1) by separating the model into different layers of computability ranging from visual and textual representation for human comprehension to a fully-encoded executable guideline; and (2) allowing top-down and

bottom-up design of a guideline by specifying details of recommended actions and decision criteria in the form of modular flowcharts known as subguidelines.

### 2.1.5 Extensibility

As is discussed later, the lifecycle philosophy inherent to the GLIF3 development approach imposes a requirement that the GLIF3 model be extensible. Since the GLIF3 model is object-oriented, it can in fact be extended with new concepts. For example, new data types can be added to the patient data model, new types of interventions can be added to the action model, or new decision-making models can be implemented. As an example, GLIF has been extended to support decision-making models that use Utility Theory [24, 26].

## *2.2 Design approach*

In an analysis of guideline modeling languages [27], it was noted that differences in application focus have given rise to a variety of features in the various modeling languages. For example, intentions in Asbru [28] support quality assessment by scoring adherence to guideline intentions, and the workflow model of GUIDE [29] supports guideline integration into organizational workflow. GLIF3 incorporates features from many of these representations so as to have utility as a vehicle for sharing executable guideline knowledge. Nevertheless, it is not possible to support fully all the features of all the different guideline modeling languages, given the continuing development of the various models. A lifecycle philosophy is adopted instead [30], such that as application experience is gained with particular kinds of functionality, such experience will over time identify those features most important to add to the shared model as it evolves.

## *2.3 The GLIF3 model*

### 2.3.1 Overview

The GLIF3 model consists of classes, their attributes, and the relationships among the classes, all of which are necessary to model clinical guidelines. We have used Unified Modeling Language (UML) class diagrams [31] to describe the model. Additional constraints on represented concepts are specified in the Object Constraint Language (OCL), a part of the UML standard. A high-level view of the GLIF[1] class diagram is shown in Figure 1.

INSERT Figure 1 HERE

In GLIF, guidelines are represented as flowcharts of temporally ordered nodes called guideline steps and represented by an abstract class called *Guideline_Step*. This class has the following subclasses:

- The *Decision_Step* class represents decision points in the guideline. A hierarchy of decision classes provides the ability to represent different decision models.
- The *Action_Step* class is used for modeling recommended actions or tasks.
- The *Branch_Step* and *Synchronization_Step*, working together, are used for modeling multiple concurrent paths through the guideline.

[1] Reference to GLIF will hereafter imply that GLIF3 is the topic of discussion. We specify GLIF2 or GLIF3 only when the distinction is important.

- *Patient_State_Step* describes the clinical state that characterizes a patient. A patient state step can function as a label summarizing the clinical state of a patient and as an entry point into the flowchart.

The GLIF specification includes an expression and query language that operates on an object-oriented data model. The query language provides a means to access patient data and to map them to variables used in decision criteria and other expressions. The data model, used for representing patient information, is based on standards being developed in HL7 [20].

The remainder of this section describes the GLIF model in more detail. A guideline for management of chronic cough [23] is used to illustrate different concepts in the model.

### 2.3.2 Encoding the guideline

Encoding a guideline in GLIF3 requires an understanding of the GLIF3 model as well as clinical domain expertise. It has been our experience that a team of two people, an informatician trained in GLIF and a clinician, can work collaboratively to encode a guideline in GLIF3. The two design a flow structure for the guideline. The informatician then encodes the guideline, while the clinician validates the encoding. This encoding and validation process is iterated until the guideline is fully specified and validated.

The first step in encoding a guideline is to create an instance of a *Guideline* class. The *Guideline* class is used to model both top-level guidelines and subguidelines (i.e., detailed specification of the steps contained in the top-level guideline). This class contains an attribute of type *Maintenance_Information* for recording information pertaining to authors, encoders, status, last modification date, and version. In addition,

the intention of the guideline, eligibility criteria, relevant associated references or multimedia material (collectively known as didactics), and the set of exceptions that interrupt the normal flow of execution of the guideline can be added to the guideline object. Each guideline also defines the patient data items that it needs, such as the variables used in decision criteria.

### 2.3.3 Building the flowchart

The guideline recommendations are encoded in the form of a flowchart. The flowchart is an instance of the *Algorithm* class. It contains a collection of steps that are instances of the *Guideline_Step* class. The flow of the steps in the algorithm is specified by linking explicitly a step to its subsequent step(s) using appropriate attributes in each subclass of *Guideline_Step*, e.g., the *next_step* attribute of an *Action_Step* (Figure 2). The *first_step* attribute of the *Algorithm* class indicates the starting point of the algorithm. However, alternative starting points in the algorithm can be specified using patient state steps, as is described later.

INSERT Figure 2 HERE

### 2.3.4 Specifying actions

Action steps specify work that is to be performed by the decision-support system, the provider, or other external agents. The work is specified as a set of tasks, of type *Action_Specification*. An action step has attributes that specify the strength of the recommendations of its tasks, the strength of empirical evidence supporting its tasks, and associated didactics. Additional attributes supporting the execution of the step specify if and how the step must be iterated, the duration within which the step must be

executed, events that trigger execution of the step, and exceptional conditions that abort the execution of the step. An action step has a *next_step* attribute that identifies the step to execute once the action step has finished its execution.

The action specification hierarchy is divided into two major types of actions:

- Actions that are carried out by the guideline execution engine (referred to in the GLIF object model as *programming-oriented* actions), such as invoking a subguideline, performing inferencing (e.g., stage of a tumor from a set of observations), or computing values for data (e.g., age of patient).
- Clinical actions (referred to as *medically-oriented* actions[2]) that are carried out by a care-provider or an external agent, such as changing a medication for a patient. Medically-oriented actions reference GLIF3's medical ontology for representation of clinical concepts such as procedures (as seen in Figure 3), medications, or referrals. The ontology classes provide parameters for structured description of the medical action (e.g., medication name, dose, frequency).

INSERT Figure 3 HERE

### 2.3.5 Specifying decisions

Decision steps are points in the algorithm where a choice has to be made among competing, mutually exclusive alternatives known as decision options. These decisions

[2] The term “medically-oriented actions” is a misnomer. Clinically-oriented actions would be a more appropriate and broadly-applicable term.

are specified in the *Decision_Step* class[3]. A Boolean attribute called *automatic_decision* distinguishes between decisions that are executed automatically and those that have to be approved by an external agent, such as a physician, other health care provider, or another software program. For example, when a decision is followed by a programming-oriented action such as for data-retrieval or invocation of a subguideline, the decision may be executed without burdening the user for approval. In other cases, when the actions following the decision step may involve significant risk, other clinical considerations that may mitigate the decision, or the decision criteria are ambiguous, it is essential to get the decision choice approved by an external agent.

INSERT Figure 4 HERE

Each decision step contains multiple decision options (Figure 4). A decision option is a combination of a reference to a guideline step (known as a *destination*) and a selection criterion. During execution, decision criteria of all the options are evaluated. Flow of execution for automatic decisions is then directed to a *destination* step corresponding to the criterion which represents the optimal decision per the decision model. For non-automatic decisions, flow is directed to the step corresponding to the option selected by an agent. In this case, selection criteria are evaluated and the results presented to the agent to guide his or her (or a computer program's) decision. The selection criterion is specified by one of the subclasses of the abstract *Decision_Condition* class: *Rule_In_Choice* and *Weighted_Choice*. In *Rule_In_Choice*, encoders specify rules for

[3] In a previous draft of GLIF, we represented automatic and agent-approved decisions in separate classes, Choice_Step and Case_Step respectively. However, retaining this structure would have created redundancies in the object model; for each new type of decision model, we would have had to create two subclasses, for example,

and against selecting the *Decision_Option*. There are four categories of rules: rule-in, strict-rule-in, rule-out, and strict-rule-out. The syntax for writing the rules is defined in the Section “Expression and query language”. A *Weighted_Choice* decision condition contains an array of criteria, each associated with a weight. The sum of the weights for each criterion in a choice must equal 1.The total weight of an option is the sum of the weights of all the criteria that evaluate to true. The total weights of the options of a *Decision_Step* are used to rank the options.

Different types of decision models can be incorporated in GLIF by extending the *Decision_Step* class. Typically, this would also involve subclassing the *Decision_Condition* class to specify the additional parameters required to describe each option. We have created a subclass of *Decision_Step*, known as *Utility_Choice_Step*, and a corresponding subclass of *Decision_Condition* known as *Utility_Choice* that models patient’s preferences in decision-making by incorporating the utility theory framework as part of the class [24, 26]. The *Utility_Choice_Step* contains a pointer to a decision analysis tree or an influence diagram used to evaluate the choices. The *Utility_Choice* class represents a node in the decision analysis tree or the influence diagram.

### 2.3.6 Describing patient states

A *Patient_State_Step* is a guideline step that is used for two purposes. One use is to serve as a label that characterizes a patient state that is achieved as a result of the

---

Utility_Case_Step and Utility_Choice_Step. In order to eliminate such redundancies, in the current version of GLIF, Decision_Step is used to specify either of these types of decisions.

successful execution of previous steps. In this way, a guideline may be viewed as a state transition graph, where transitions among these patient state steps are formed by networks of other guideline steps. The other purpose of a patient state step is to serve as an alternative to the first step as an entry point to the guideline. At the time of invocation of a guideline, patient state steps are evaluated for applicability to the patient. If the patient is confirmed to be in the state specified in the step (i.e., the criterion for the state evaluates to true), the execution is moved to that step.

A patient state step has a criterion that specifies the state. If a criterion refers to a generalized patient state (e.g., patient has pneumonia) it also applies to specializations of that state (e.g., when the patient has bacterial pneumonia).

### 2.3.7 Other flow-control mechanisms

Among the mechanisms of flow control described earlier in this paper are the *next_step* attribute of *Guideline_Step* subclasses for sequential flow, and the *destination_step* of *Decision_Option* for conditional flow. Several other flow-control mechanisms exist in GLIF3 that provide the flexibility not accommodated in a strictly flowchart model such as that of GLIF2. If either an action step or a decision step has a nested subguideline, flow is directed to the subguideline (discussed later in section titled “Managing Complexity”). Exception conditions can halt the execution of a guideline. In addition, the *Patient_State_Steps* and *first_step* attribute of the *Algorithm* are used as entry points into the guideline. Other control mechanisms can be used for parallel flows.

#### 2.3.7.1 Defining parallel paths or unspecified sequences in an algorithm

In order to model parallel execution of two or more sequence of steps, a combination of a *Branch_Step* and a *Synchronization_Step* is used. This combination may be used also to describe an algorithm with flexible flow when the steps do not need to be constrained to a sequential flow. Branch steps direct flow to multiple paths with each path containing a sequence of guideline steps. All of these paths are executed in parallel. These paths can converge eventually to a single synchronization step. A *continuation* attribute specifies whether all, some, or one of the preceding steps must be completed before control can move to the next step. The continuation is expressed as a logical expression of guideline steps (e.g., (*Step_A* or *Step_B*) indicates that flow must continue once either *Step_A* or *Step_B* is completed). The bottom of Figure 2 shows a branch step spawning two branches (labeled “Chest X-Ray” and “Treatment of Cough”). The branching paths converge at the synchronization step (labeled “Wait for Treatment”). This step suspends further execution until the tasks specified in the step “Treatment of Cough” are completed.

#### 2.3.7.2 Iterating through an action or decision

*Iteration_Specification* is used in order to specify if and how the execution of an action or decision must be repeated. An *iteration_info* attribute of type *Iteration_Specification* in *Action_Step* and *Decision_Step* classes is used for specifying this flow control mechanism. Only action steps and decision steps may be iterated. These steps are iterated until the abort condition or stopping (normal termination) condition criteria of the iteration specification are satisfied (e.g., repeat step until systolic BP falls below 130 mm Hg, or repeat step until 7 days from now). The iterations are carried out at a certain

frequency (e.g., repeat step every 3 days), which is expressed by an *Iteration_Expression*.

#### 2.3.7.3 Asynchronous triggering

The *Triggering_Event* class describes events that initiate the execution of associated guideline steps. Action steps, decision steps, and patient state steps have an attribute, called *triggering_events*, which specifies the events that trigger the execution of the step. During execution, when the flow reaches a step that has associated triggering events, this step is executed only after one of its triggering events occurs. If more than one triggering event occurs at the same time, then the highest priority event is chosen to trigger the step, as specified by the *priority* attribute of the *Triggering_Event* class. Different types of events are defined in the model: end of execution of a previous guideline step, availability of new patient data (e.g., recording of laboratory test results), and temporal events (e.g., after 2 weeks from now).

As discussed, conditional flow of execution is dependent on the values of patient data. The next section describes how patient data are represented in GLIF.

### 2.3.8 Modeling patient data and medical knowledge

To provide patient-specific automatic decision-support services during clinical encounters, a CIG should be integrated with clinical information systems. Patient data items and medical concepts to which CIGs refer must be mapped to electronic medical record (EMR) entries. We designed a medical ontology that defines the structure of patient data and medical knowledge classes in a way that potentially facilitates such mapping, as explained below.

The first layer of the ontology, Core GLIF, is part of the GLIF specification language. It contains the foundation classes for specifying the datatypes of medical data items (such as patient-specific data), medical concepts (such as concept identifiers from controlled terminologies), and medical knowledge specified as relationships among concepts (e.g., *ACE-inhibitor is-contraindicated-by Diabetes*, where *ACE-inhibitor* and *Diabetes* are concepts). By defining patient data and medical knowledge in terms of concepts from controlled terminologies, encoded guidelines do not contain proprietary terms that are specific to a local institution. This facilitates sharing of an encoded guideline.

The second layer, the Clinical Information Model (CIM), defines the data model for representing patient information. The CIM currently built into GLIF is based on HL7's Reference Information Model (RIM) [25]. The CIM includes classes of patient data, such as medications, procedures and observations. Mapping the CIM layer to an EMR enables several guidelines that use the same patient data items to be mapped to an EMR once, via the CIM-EMR mapping. However, it is possible to use other CIMs specified in terms of Core GLIF classes.

Figure 3 illustrates the use of Core GLIF and CIM. *4-view sinus radiograph* is an instance of a *Literal_Data_Item*, a Core GLIF class. The details of the sinus radiograph are specified as an instance of *Procedure*, a class in the CIM.

The function of the third layer, known as the Medical Knowledge Layer, is to specify methods for interfacing to sources of medical knowledge that are not part of the GLIF3 specification. For example, such knowledge is encoded in: (1) controlled terminologies that define taxonomies of medical concepts, either implicitly in the hierarchies in ICD-9 [32] or explicitly in defined relationships in SNOMED [33], and (2) medical knowledge

bases that define drug classes (commercially available from several vendors), and normal ranges for test results.

We believe that the multi-layered approach to patient data modeling will facilitate sharing of guideline knowledge. However, since the current implementation of the GLIF3 guideline execution engine known as GLEE [22] has not yet been integrated with a clinical information system, the layered model is not supported in the software. We intend to support additional features of data encoding specified in the GLIF model in future versions of GLEE as we obtain more experience integrating the engine with clinical information systems.

### 2.3.9 Expression and query language

Action specifications (tasks) and decision conditions reference patient data items and medical knowledge through logical expressions and queries. *Guideline_Expression* and its subclasses are used to specify queries (e.g., results of last serum potassium), decision criteria (e.g., Age > 32), formulae (e.g., body-mass index), expressions for abstracting from raw data (e.g., presence of hypertension from blood pressure measurements), iteration conditions, events, and temporal intervals and constraints.

Different expression languages can be used with the *Guideline_Expression* class. Previously, we had developed a language called Guideline Expression Language (GEL) [34] that is based on the Arden Syntax [35]. However incompatibilities between this language (which was designed for a time-stamped, list-oriented data structure) and the object-oriented CIM soon became apparent [36]. We thus redesigned the language to an object-oriented form. This new language, dubbed GELLO [37] (loosely for “guideline

expression language, object-oriented"), supports query and expression formulation. In this language, the queries and expressions share a common object-model because the results of queries are used (as variables) in decision criteria and other expressions, and because expressions are used as data selection predicates in queries.

GELLO query statements [38] map patient data (that are subsequently used in expressions) to entries in the medical record. The query syntax has been designed in the context of the decision-support execution model proposed in the HL7 Clinical Decision Support Technical Committee (CDSTC). This model envisions the use of a "virtual medical record" (vMR) compatible with the HL7 RIM that provides a standard data model as an intermediary to heterogeneous medical record systems [39]. In the current GLIF specification, the CIM serves the function of a vMR data model. The specifications for a standard vMR are being developed in the HL7 CDSTC. We will adopt the standard model when it is published. Note that the query syntax for GELLO does not depend on specific classes or tables in the vMR. However, it does depend on the general framework of an object-oriented data model. The query statement below[4] retrieves currently active ACE-inhibitor medication prescriptions for a patient:

```
Medication->select(meds :

        meds.service_cd.equals(
```

[4] The GELLO language had evolved rapidly during the latter part of the InterMed project. The example here uses the syntax that is based on Object Constraint Language (OCL) [38]. Warmer J, Kleppe A. The Object Constraint Language: Getting Your Models Ready for MDA. 2nd ed. Boston, MA: Addison-Wesley Pub Co; 2003.. In an earlier version of GELLO, used in GLIF 3.5, the syntax was based on Object Query Language [40]. Cattell RGG, Barry DK, Berler M, et al., editors. The Object Data Standard: ODMG 3.0. San Francisco, CA: Morgan Kaufmann Publishers; 2000. and Temporal Structured Query Language [41]. Snodgrass RT. The TSQL temporal query language. Boston, MA: Kluwer Academic Publishers; 1995..

Concept.new(“ACE-inhibitor”, “C-80150”, “SNOMED-CT”)) and

meds.critical_time.max_time_stamp.greaterThan(now))

The expression syntax is strongly-typed and object-oriented. In addition to basic data types and operations, it allows the use of classes, class attributes, and methods that can be used to create complex mathematical, logical, and temporal expressions. The expressions often consist of operations over variables initialized by the queries, (e.g., *active_ACE_inhibitor_orders.is_empty(),* where *active_ACE_inhibitor_orders* is a variable assigned the result of the query above).

Work on the GELLO expression and query language is continuing in the HL7 CDSTC and other committees to extend the application of GELLO to different specifications in HL7 that require constraints, expressions, and mapping of variables to data. Among potential application specifications are those for guidelines, Arden Syntax rules [35], and templates. Accordingly, the focus of the effort is on making the language independent of particular data models, making it free of side-effects (i.e., preventing GELLO expressions from altering application variables), and compatible with the basic datatypes specification in HL7’s version 3.0 specification.

### 2.3.10 Managing complexity

The flowcharts of guidelines can often get large and complex. GLIF supports several mechanisms for managing such complexity. One mechanism involves separating the conceptual flowchart level from the computations specification. This mechanism is described in detail elsewhere [20]. Some other complexity management techniques that have been investigated are described in this section.

GLIF defines a mechanism for specifying guideline steps recursively through the nesting of subguidelines in action and decision steps. Thus, a guideline can be specified as a concise flowchart of high-level actions and decisions. The details of the actions and decisions contained in the flowchart can be specified as successively deeper levels of subguidelines of the respective steps. Further, because nesting allows grouping of segments of a guideline flowchart into modular units (subguidelines), it promotes reuse of these segments.

INSERT Figure 5 HERE

Decisions are nested by specifying a subguideline in the *decision_detail* attribute of a decision step (Figure 5). The subguideline specifies the steps of a complex decision (e.g., does a patient have gastro-intestinal reflux disease (GERD), one of the causes of chronic cough). This subguideline is executed before the decision criterion for that step is evaluated. During its execution, the subguideline assigns values to variables (e.g., *gerd := true*). The use of these variables in the decision criteria (e.g., *gerd == true*?) of that step makes the decision nested.

Action steps are nested by including a task of type *Subguideline_Action* in the action step's *tasks* attribute. The *Subguideline_Action* task has an *action_detail* attribute that contains the nested subguideline. The *parameters_passed* attribute of *Guideline* specifies the parameters that need to be passed to it by the guideline that calls it or those that are passed out (e.g., the value of *gerd*) to the calling guideline.

Another feature that is still being developed is called a macro step. A macro step is used for specifying domain-level constructs [42] such as schedule of screening tests (e.g., breast cancer screening). Macro steps include a declarative mapping of the

domain-level construct to a pattern that comprises a flowchart of guideline steps. The macro step can then be instantiated as a set of underlying GLIF steps. Thus, a macro step enables the guideline author and the viewer of guidelines to create and peruse the guideline at a domain level. The execution engine can map the macro to GLIF steps in order to execute the guideline.

In addition to the above, GLIF provides view filters and let expressions as complexity management mechanisms. View filters enable customization of views for a user or a scenario. Let expressions can be used for simplifying expressions by dividing a complex expression into smaller expressions. Details on these features can be found in the GLIF3 technical specification [24]. The next section describes the *Supplemental_Material* class which can, among other applications, be used to annotate and document a complex guideline.

### 2.3.11 Documenting the guideline

The *Supplemental_Material* class is used to associate didactic and other external resources with a guideline and its steps. Supplemental materials are knowledge items that are not required for the execution of GLIF3 guidelines but are an important means for providing additional information to the clinician receiving the guideline recommendations. Supplemental material can be of different types such as (inline) text, URL linking to a resource external to the guideline, or a keyword that can be used for dynamically identifying resources from external searchable repositories. The *Supplemental_Material_List* class is used to package a number of different supplemental material objects that serve the same purpose. These objects can be used

to include explanations (Figure 6), patient education materials, or references, or to index the guideline and its steps. All supplemental materials specify their format as an Internet Mail Extensions (MIME) type such as text/plain, text/html, image/gif, and mov/qt. Preliminary work to add attributes of the Guideline Elements Model (GEM) [43] for narrative guideline markup has been done [44]. These attributes enable encoded guidelines to reference portions of the original narrative guideline document that focus on non-algorithmic guideline content, such as the methods for evidence collection.

INSERT Figure 6 HERE

## *2.4 RDF file-format*

GLIF guidelines are stored and exchanged in the RDF format [21]. RDF is an infrastructure that enables the encoding, exchange and reuse of structured metadata. It has been developed under the auspices of the World Wide Web Consortium (W3C). RDF has an explicit model for expressing object semantics (objects, attributes), and uses XML (eXtensible Markup Language) [45] as a common syntax for exchanging and processing of metadata. The data structure or metadata definitions of GLIF's object model are specified as an RDF Schema. The schema can be downloaded from the GLIF website (http://www.glif.org). Guideline instances are specified in the XML syntax of RDF. One RDF file can contain many GLIF guidelines. A *Guideline_Collection* object specifies the top-level guidelines in the file. For example, a file containing immunization guidelines may include separate top-level guidelines for influenza vaccination and Hepatitis-B vaccination.

# 3 Discussion

GLIF3 is intended to facilitate the sharing of computer-interpretable guidelines. Our previous work on GLIF2 focused on defining guideline step classes and flow of control to make these more consistent, as a basis for sharing of guidelines. GLIF3 extends the work of GLIF2 by addressing logical consistency and syntactic and semantic control to enable execution of guidelines. In addition, GLIF3 provides features that facilitate sharing of CIGs such as a flexible representation that can be used to encode different types of guidelines, an object-oriented data model (CIM) and a query language that allow mapping of decision variables to data in the medical record, and the use of controlled medical terminologies.

We have encoded 12 large, complex, and differing guidelines in GLIF3. One study, drawing on cognitive science approaches, has found that the CIGs developed in GLIF3 contained greater level of representational detail and less ambiguity than those developed in GLIF2 [46]. However, additional studies are needed to assess other aspects of the GLIF representation model such as sufficiency and consistency.

While these CIGs have yet to be implemented in the clinical setting, the execution of GLIF3-encoded guidelines has been evaluated using the GLEE execution engine. We used both simulated clinical data and real clinical data from an electronic medical record in the evaluation. The results demonstrated that GLIF3's semantics could be correctly interpreted by GLEE and that recommendations comparable to a reference system could be produced [47].

GLIF is designed to provide flexibility in modeling sharable guidelines. The ability to reference different domain ontologies, the use of standards being developed in HL7 to limit platform-dependence, independence from a particular application model (see next paragraph), and an extensible object-oriented architecture are all steps in this direction.

Because flexibility in modeling can lead to variation in encoding,, when possible we avoided overlap in the functionality of different GLIF3 constructs, and we sought to assure that a single GLIF3 construct could not be used to model two different guideline situations. For example, the branch step is no longer used to represent decision choices, as as it had been in GLIF2, which inappropriately mixed semantics of concurrency and decision-making [19].

GLIF does not have a particular application orientation. Its focus is on portraying the medical decisions and actions and their proper flow of control. The form of interaction of the guideline with the user and the environment is a decision of individual implementers on particular platforms, in particular settings, and for particular applications. As a result:

1. GLIF cannot encompass all of the features identified in the many guideline modeling research and development projects underway [27], some of which have been designed to facilitate specific kinds of applications. Thus GLIF is not a true bi-directional interchange format. It cannot represent the full set of features of some models for export of guidelines developed in those modeling frameworks to other environments; similarly, importing a GLIF guideline into another modeling environment may require addition of some missing features that are specific to the GLIF model.

2. GLIF is expected to evolve over time as a result of a lifecycle process in which the results of successful applications indicate certain guideline modeling features that should be included in sharable guideline specifications, thereby facilitating future development of similar applications. While the essence of the GLIF model as a flowchart of *Guideline_Step* classes should remain static, much of the evolution is likely to occur by extending or modifying the existing GLIF classes.

Another limitation in GLIF3 is that it does not specify fully the implementation layer. Since this layer requires interfacing with heterogeneous clinical information systems, we are working with the HL7 CDSTC, to define the mappings between the guideline data model and the electronic medical record (i.e., the vMR work noted above [39]), and to enable support for execution of guideline recommendations via order-entry systems.

We have developed a suite of software tools to be used for encoding and execution of computer-interpretable guidelines. These tools continue to guide our understanding of the requirements of GLIF for encoding and implementation of guidelines. Earlier versions of software for GLIF2 [48] have been re-architected and implemented for use with GLIF3. A custom guideline-encoding tool has been developed [49]. In addition, the Protégé knowledge editing tool [50] can be used for encoding and validating [51] GLIF3 guidelines using the RDF schema mentioned previously. We also implemented our guideline execution engine (GLEE) based upon a flexible, client-server design [22].

Future work depends to a large extent on the progress of HL7 as it addresses the above tasks and other decision-support related interface standards (including evolution of GELLO [52]), and on the appearance of tools that implement GLIF3 encoding, validation, navigation, eligibility searching and retrieval services, and execution. It also

depends strongly on experience with implementation, determination of what kinds of applications are effective, and identification of what additional modeling features or tools would be helpful. A number of tool-development and implementation projects are underway currently by us and by others.

# 4 Conclusions

GLIF3 extends the work of GLIF2, aimed at creating a formal means for specifying computer-interpretable guidelines that can be implemented at the point of care. Unlike a number of guideline modeling and implementation projects, the aim of GLIF3 is to facilitate the sharing and implementation of high-quality guidelines that are developed with the goal of widespread dissemination and use in a variety of potential platforms and application settings. Further development of GLIF3 is focused on facilitating the integration of CIGs into a broad spectrum of clinical information systems.

## Acknowledgments

This research was supported by grant LM06594 from the National Library of Medicine, with collaborative support from the Telemedicine and Advanced Technology Research Center of the US Army, and the Agency for Healthcare Research and Quality. Elmer Bernstam, Ronilda Lacson, and Peter Mork have made important contributions to the development of GLIF3.

# References

[1]. Woolf SH, Grol R, Hutchinson A, et al. Potential benefits, limitations, and harms of clinical guidelines. BMJ 1999;318(7182):527-530.
[2]. Kohn LT, Corrigan JM, Donaldson MS. To err is human: building a safer health system. Washington, D.C.: Institute of Medicine. National Academy Press; 1999.
[3]. 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(3):138-44.
[4]. Wolff M, Bower DJ, Marbella AM, et al. US family physicians' experiences with practice guidelines. Fam Med 1998;30(2):117-21.
[5]. Zielstorff RD. Online practice guidelines: issues, obstacles, and future prospects. J Am Med Inform Assoc 1998;5(3):227-36.
[6]. Johnston ME, Langton KB, Haynes RB, et al. Effects of computer-based clinical decision support systems on clinician performance and patient outcome. A critical appraisal of research. Ann Intern Med 1994;120(2):135-42.
[7]. Bates DW, Teich JM, Lee J, et al. The impact of computerized physician order entry on medication error prevention. J Am Med Inform Assoc 1999;6(4):313-21.
[8]. Lobach DF, Hammond WE. Computerized decision support based on a clinical practice guideline improves compliance with care standards. Am J Med 1997;102(1):89-98.
[9]. Peleg M, Patel VL, Snow V, et al. Support for guideline development through error classification and constraint checking. Proc AMIA Fall Symp 2002:607-11.
[10]. 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(4):357-72.
[11]. 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.
[12]. Musen MA, Tu SW, Das AK, et al. EON: a component-based approach to automation of protocol-directed therapy. J Am Med Inform Assoc 1996;3(6):367-88.
[13]. Kuperman GJ, Teich JM, Tanasijevic MJ, 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.
[14]. McDonald CJ, Hui SL, Smith DM, et al. Reminders to physicians from an introspective computer medical record. A two-year randomized trial. Ann Intern Med 1984;100(1):130-8.
[15]. Advani A, Goldstein M, Musen MA. A framework for evidence-adaptive quality assessment that unifies guideline-based and performance-indicator approaches. Proc AMIA Symp 2002:2-6.
[16]. Kuperman GJ, Teich JM, Gandhi TK, et al. Patient safety and computerized medication ordering at Brigham and Women's Hospital. Jt Comm J Qual Improv 2001;27(10):509-21.

[17]. Quaglini S, Stefanelli M, Lanzola G, et al. Flexible guideline-based patient careflow systems. Artif Intell Med 2001;22(1):65-80.
[18]. Stefanelli M. Knowledge management to support performance-based medicine. Methods Inf Med 2002;41(1):36-43.
[19]. Peleg M, Boxwala AA, Ogunyemi O, et al. GLIF3: The evolution of a guideline representation format. Proc AMIA Annu Fall Symp 2000:645-9.
[20]. Peleg M, Boxwala AA, Tu S, et al. The InterMed approach to sharable computer-interpretable guidelines: a review. J Am Med Inform Assoc 2004;11(1):1-10.
[21]. Lassila O, Swick RR. Resource Description Framework (RDF) Model and Syntax Specification. W3C Recommendation. Cambridge, MA: World Wide Web Consortium; 1999 22 Feb 1999. Report No.: REC-rdf-syntax-19990222.
[22]. Wang D, Shortliffe EH. GLEE – A model-driven execution system for computer-based implementation of clinical practice guidelines. Proc AMIA Fall Symp 2002:855-9.
[23]. Irwin RS, Boulet LP, Cloutier MM, et al. Managing cough as a defense mechanism and as a symptom. A consensus panel report of the American College of Chest Physicians. Chest 1998;114(2 Suppl Managing):133S-181S.
[24]. Peleg M, Boxwala AA, Tu S, et al. Guideline Interchange Format 3.0 Technical Specification. Technical Report. Boston, MA: Brigham and Women's Hospital; 2002. Report No.: DSG-TR-2002-014. Available at http://www.glif.org.
[25]. Russler DC, Schadow G, Mead C, et al. Influences of the Unified Service Action Model on the HL7 Reference Information Model. Proc AMIA Symp 1999:930-4.
[26]. Lacson RC. Knowledge representation of situation-specific clinical practice guidelines [MS thesis]. Cambridge, MA: Massachusetts Institute of Technology; 2000.
[27]. Peleg M, Tu S, Bury J, et al. Comparing computer-interpretable guideline models: a case-study approach. J Am Med Inform Assoc 2002:In press.
[28]. Shahar Y, Miksch S, Johnson P. An intention-based language for representing clinical guidelines. Proc AMIA Annu Fall Symp 1996:592-6.
[29]. Quaglini S, Stefanelli M, Lanzola G, et al. Flexible guideline-based patient careflow systems. Artif Intell Med 2001;22(1):65-80.
[30]. Greenes RA, Peleg M, Boxwala AA, et al. Sharable computer-based clinical practice guidelines: rationale, obstacles, approaches, and prospects. Medinfo 2001:201-5.
[31]. Eriksson H-E, Penker M. UML Toolkit. New York, NY: Wiley Computer Publishing; 1998.
[32]. International Classification of Diseases, 9th Revision, Clinical Modification (ICD-9-CM). 6th ed. Baltimore, MD: The Centers for Medicare & Medicaid Services; 2002.
[33]. Stearns MQ, Price C, Spackman KA, et al. SNOMED clinical terms: overview of the development process and project status. Proc AMIA Symp 2001:662-6.
[34]. Ogunyemi O. The Guideline Expression Language (GEL) User's Guide. Technical Report. Boston, MA: Brigham and Women's Hospital; 2000. Report No.: DSG-TR-2000-001.

[35]. E 1460 Standard Specification for Defining And Sharing Modular Health Knowledge Bases (Arden Syntax for Medical Logic Modules). ASTM Standards v 14.01. Philadelphia: American Society for Testing and Materials; 1992.
[36]. Peleg M, Ogunyemi O, Tu SW, et al. Using features of Arden syntax with object-oriented medical data models for guideline modeling. Proc AMIA Symp 2001:523-7.
[37]. Ogunyemi O, Zeng Q, Boxwala AA. BNF and built-in classes for object-oriented guideline expression language (GELLO). Technical Report. Boston, MA: Brigham and Women's Hospital; 2001. Report No.: DSG-TR-2001-018.
[38]. Warmer J, Kleppe A. The Object Constraint Language: Getting Your Models Ready for MDA. 2nd ed. Boston, MA: Addison-Wesley Pub Co; 2003.
[39]. Johnson PD, Tu SW, Musen MA, et al. A virtual medical record for guideline-based decision support. Proc AMIA Annu Fall Symp 2001:294-8.
[40]. Cattell RGG, Barry DK, Berler M, et al., editors. The Object Data Standard: ODMG 3.0. San Francisco, CA: Morgan Kaufmann Publishers; 2000.
[41]. Snodgrass RT. The TSQL temporal query language. Boston, MA: Kluwer Academic Publishers; 1995.
[42]. Boxwala AA, Mehta P, Peleg M, et al. Representing guidelines using domain-level knowledge components. Proc AMIA Annu Fall Symp 2000:Abstr.
[43]. Shiffman RN, Karras BT, Agrawal A, et al. GEM: A proposal for a more comprehensive guideline document model using XML. J Am Med Inform Assoc 2000;7(5):488-98.
[44]. Yin JQ, Peleg M, Boxwala AA, et al. Combining a document model and an execution model for clinical guidelines. Proc AMIA Fall Symp 2001:1064.
[45]. Bray T, Paoli J, Sperberg-McQueen CM, et al. Extensible Markup Language (XML) 1.0 (Second Edition). W3C Recommendation. Cambridge, MA: World Wide Web Consortium; 2000 6 October 2000. Report No.: REC-xml-20001006. Available at http://www.w3.org/TR/2000/REC-xml-20001006.
[46]. Patel VL, Branch T, Wang D, et al. Analysis of the process of encoding guidelines: a comparison of GLIF2 and GLIF3. Methods Inf Med 2002;41(2):105-13.
[47]. Wang D. A generic execution model for sharing of computer-interpretable clinical practice guidelines [Dissertation]. New York, NY: Columbia University; 2003.
[48]. Greenes RA, Tu S, Boxwala AA, et al. Toward a shared representation of clinical trial protocols: Application of the GLIF guideline modeling framework. In: Silva JS BM, Chute C, Langlotz C, , Hiland JC SW, Douglas JV, editors. Cancer Informatics: Creating a Knowledge Management Infrastructure for Cancer. New York, NY: Springer-Verlag; 2001.
[49]. Boxwala AA, Zeng Q, Vuong J, et al. GLIF software components and tools: Programmer's manual and reference documentation. Technical Report. Boston, MA: Decision Systems Group; 2001. Report No.: TR-2001-10.
[50]. Musen MA, Gennari JH, Eriksson H, et al. PROTEGE-II: computer support for development of intelligent systems from libraries of components. Medinfo 1995;8 Pt 1:766-70.
[51]. Peleg M, Patel VL, Snow V, et al. Support for guideline development through error classification and constraint checking. Proc AMIA Symp 2002:607-11.

[52]. Sordo M, Ogunyemi O, Boxwala AA, et al. GELLO: An object-oriented query and expression language for clinical decision support. In: Musen MA, editor. AMIA Annu Fall Symp; 2003; Washington, DC: American Medical Informatics Association; 2003. p. 1012.

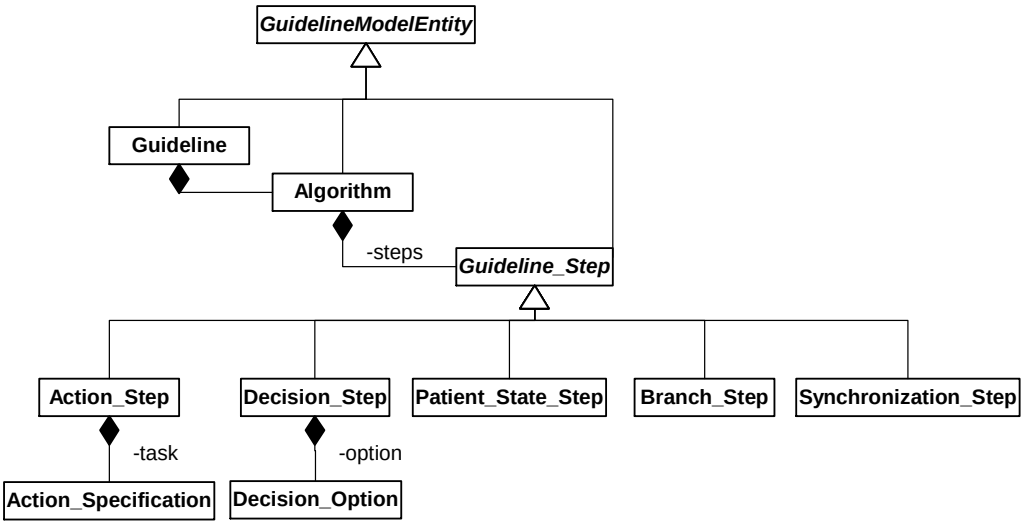


**Figure 1. A high-level view of the major classes in GLIF. The lines between classes denote relationships: a diamond-shape arrowhead indicates an aggregation or containment relationship, and a triangle shape indicates a generalization relationship.**

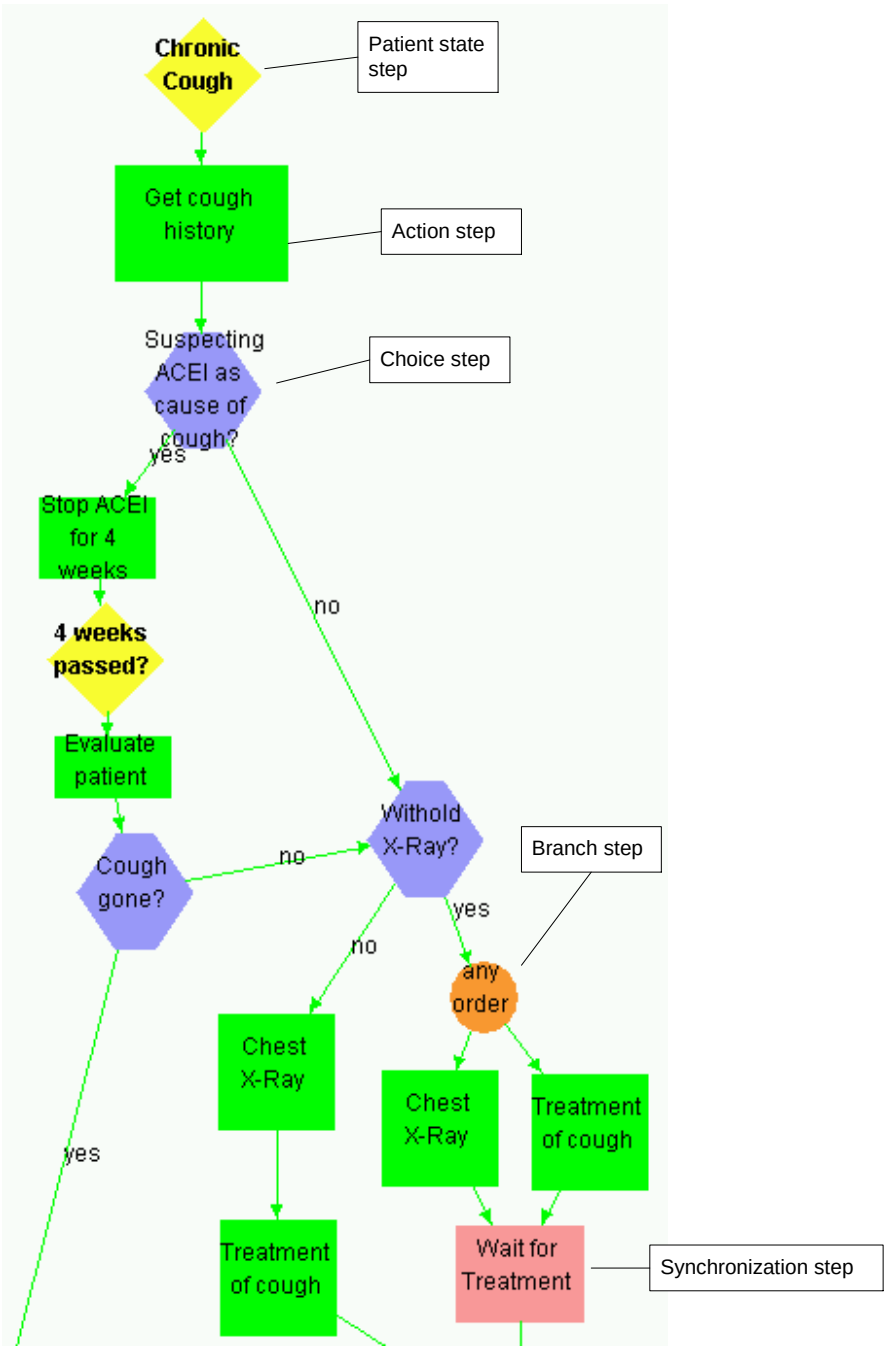


**Figure 2. Algorithm for the treatment of chronic cough showing different types of steps that are connected to each other, forming the flowchart. The step named "Chronic Cough" is the first step of this algorithm.**

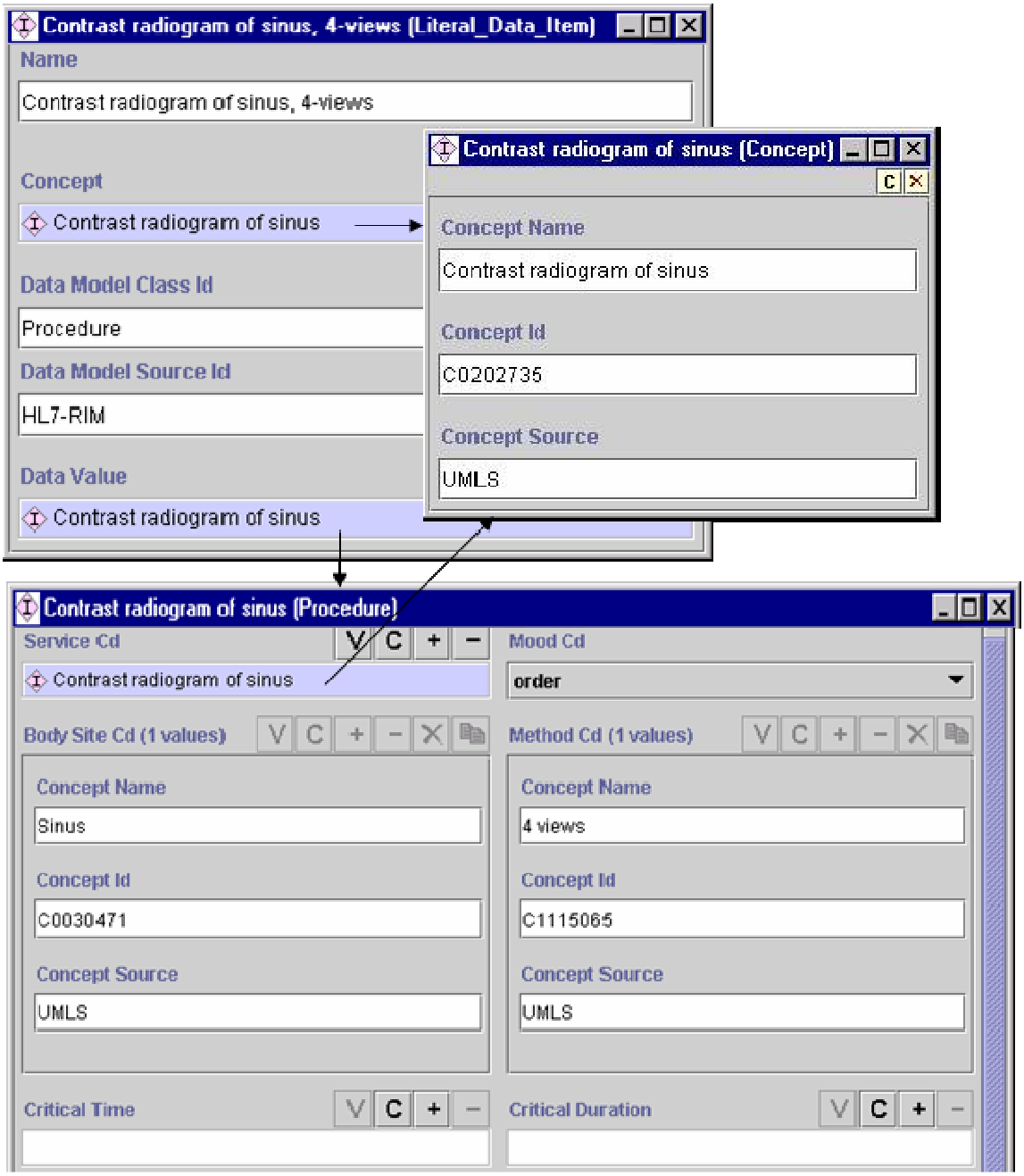


**Figure 3. Details of the four-view sinus radiograph *Literal_Data_Item* that is referenced in an *Action_Specfication*. *Concept* specifies the unique identifier (*Concept_Id*) of a term from a controlled medical terminology (UMLS, in this example). *Data_Model* provides parameters for the task. In this example, the parameters are those of a medical *Procedure*, and include the procedure's code (*Service_Cd*), the body site at which the procedure will be performed, the method by which the procedure will be performed, and the time at which to perform the procedure**

**and its duration. Values for the latter two attributes are not specified in this example since such temporal constraints are not part of the guideline recommendation.**

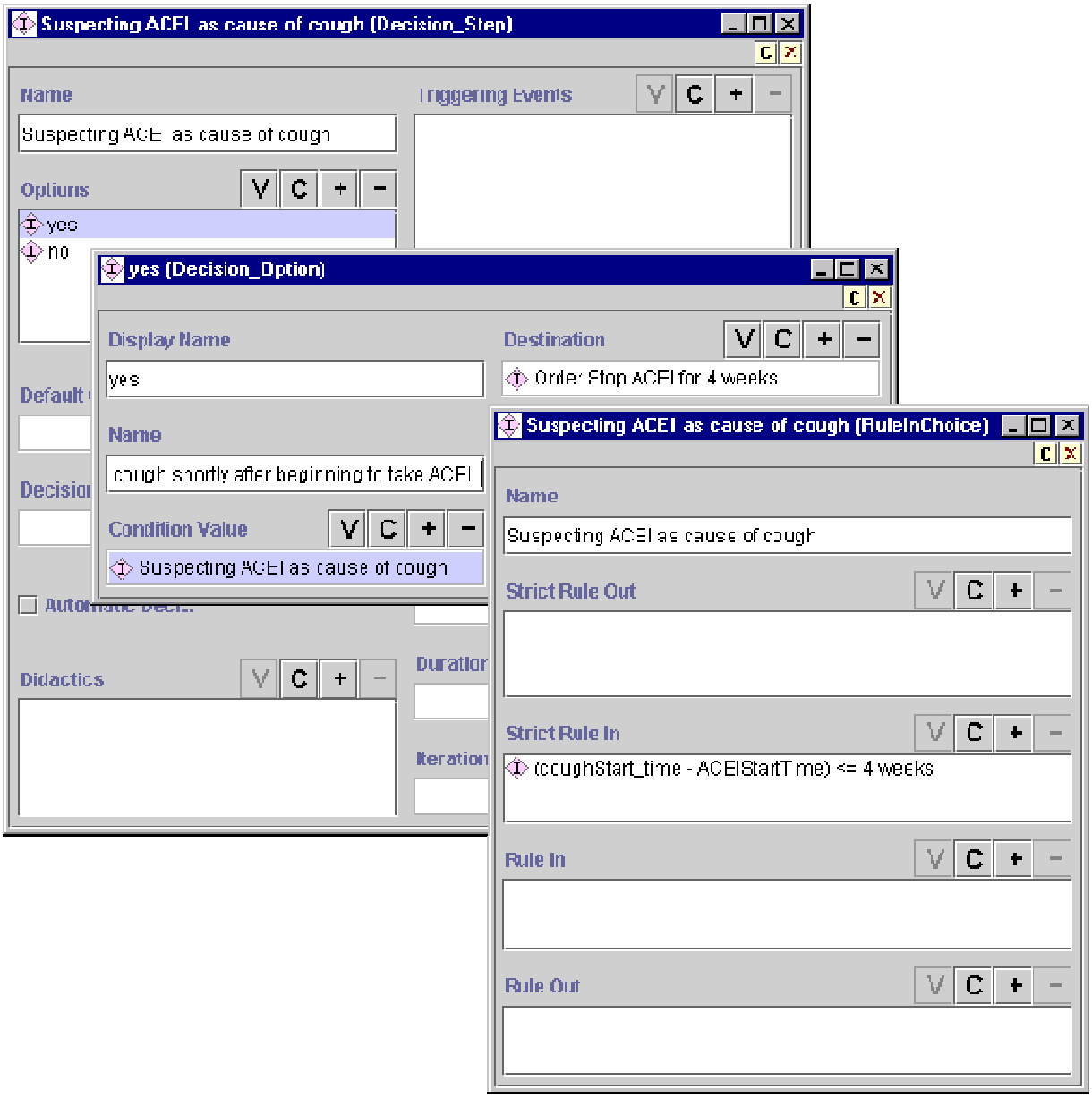


**Figure 4. A decision step from the algorithm in Figure 2. The step has two decision options. The window overlying the decision step window shows the detail of one of the decision options. A decision option has a condition that is used in selecting among various options and a destination that specifies the step to execute if this option is selected. The condition is an instance of a *Rule_In_Choice*, where a strict-rule-in is specified in a formal expression language.**

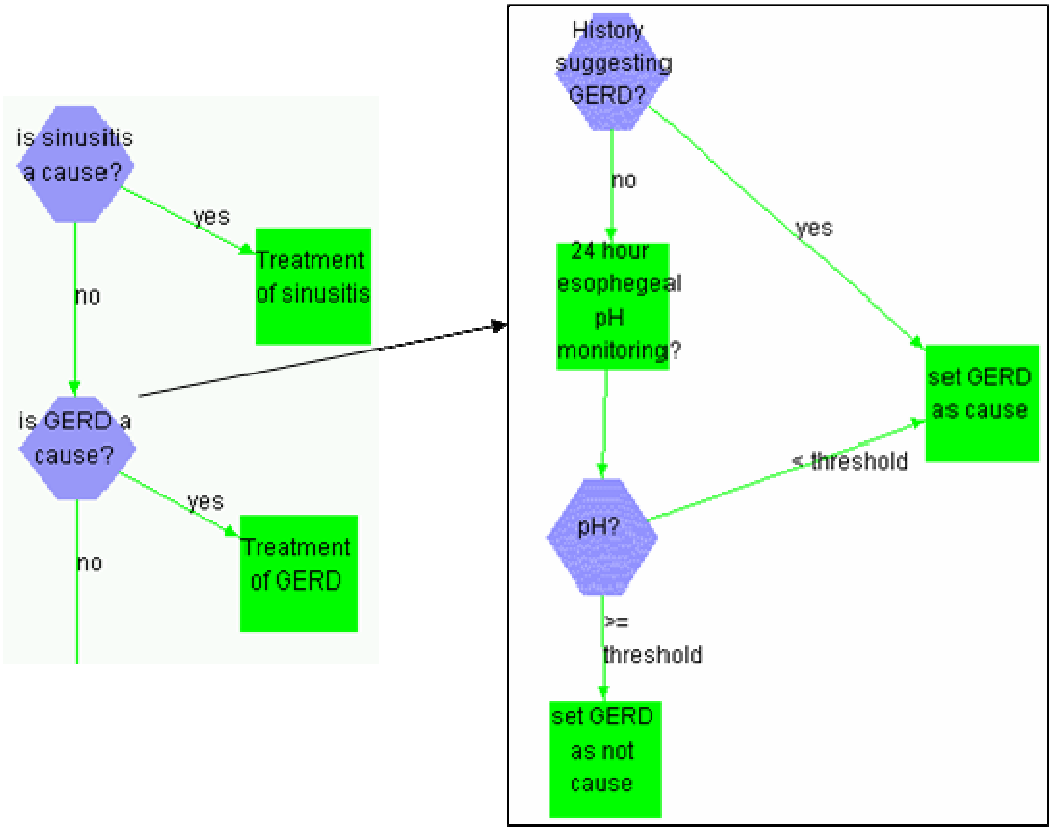


**Figure 5. Nesting a decision in a hypothetical cough treatment algorithm. The left panel shows part of a guideline for treatment of cough. The decision step "Is GERD a cause?" in the guideline shown on the left contains the subguideline for evaluation of gastro-intestinal reflux disease (GERD) shown on the right hand side. The steps "set GERD as cause" and "set GERD as not cause", which are part of the subguideline, set the value of a variable *gerd* to *true* or *false* respectively. This variable is evaluated in the decision criterion in the container decision step "Is GERD a cause?" after the subguideline has executed.**

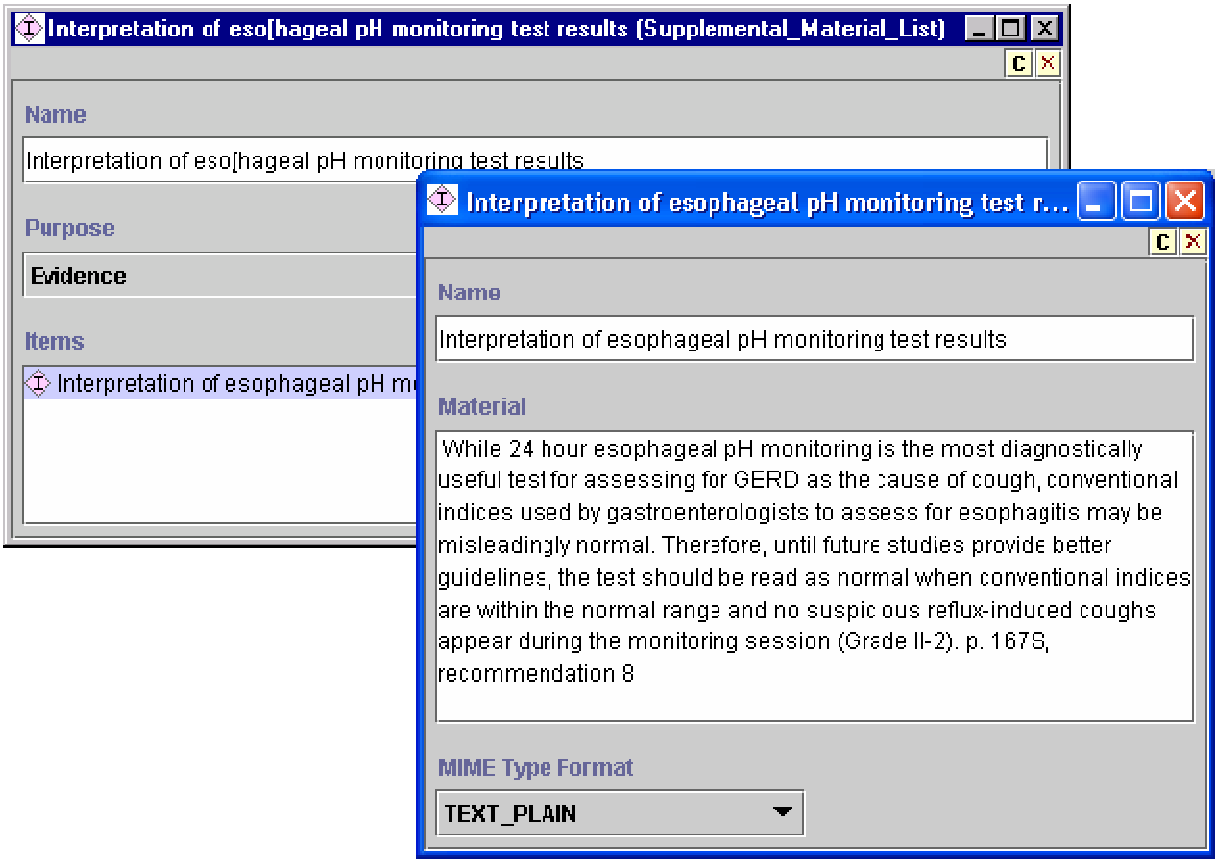


**Figure 6. A didactic object elaborating on how results of esophageal pH monitoring tests should be interpreted for the evaluation of GERD.**