Mapping Computerized Clinical Guidelines to Electronic Medical Records: Knowledge-Data Ontological Mapper (KDOM)

Mor Peleg[1], Sagi Keren[2], and Yaron Denekamp[3]

[1]Department of Management Information Systems, University of Haifa, Israel, [2]Department of Computer Science, University of Haifa, Israel, [3]Carmel Medical Center and the Faculty of Medicine, Technion, Haifa, Israel

Proofs should be sent to:

Mor Peleg

Department of Management Information Systems

University of Haifa, 31905, Israel

Email: morpeleg@mis.hevra.haifa.ac.il

Phone: 972-4-828-8504

Fax: 972-4-828-8522

Abstract

Clinical guidelines recommend quality standards for patient care Encoding guidelines in a computer-interpretable format and integrating them with an Electronic Medical Record (EMR) can enable delivery of patient-specific recommendations when and where needed. GLIF3 is a language for representing computer-interpretable guidelines (CIGs) and sharing them among healthcare institutions. Sharing a CIG necessitates mapping its data items to the institutional EMRs. We developed a framework called Knowledge-Data Ontological Mapper (KDOM) that enables bridging the gap from abstractions used in CIGs to specific EMRs. Briding the gap involves: (1) using an ontology of mappings, and an optional reference information model, to map an abstraction gradually into EMR codes, and (2) automatically creating SQL queries to retrieve the EMR data. We evaluated the KDOM framework by mapping a GLIF3-encoded guideline into two different EMR schemas and by using the mapping ontology to define mappings from 15 GLIF3 CIGs and one SAGE CIG into our reference information model.

Keywords: clinical guidelines, GLIF, mapping, ontology, EMR

1 Introduction

Clinical practice guidelines intend to provide knowledge needed for decision making about a specific health problem. They include information about modalities and evidence for diagnosis, prognosis and treatment, and other knowledge needed for decision making regarding patient-specific treatment. The main benefit of such guidelines is improvement of the quality of care received by patients [1]. Computerizing guidelines in a way that they can deliver decision support during clinical encounters has been shown to increase guidelines' impact on clinicians' behavior [2].

The Guideline Interchange Format – a shareable computer-interpretable representation

The Guideline Interchange Format, version 3 (GLIF3), is a computer-interpretable language for representing and conveying clinical practice guidelines in a form appropriate for automatic decision support on treatment that can be applied to specific patients during clinical encounters. The ultimate goal of GLIF3 is to develop a standard for representing guidelines that facilitates their sharing across software tools at different medical institutions that manipulate, analyze, or otherwise compute with an electronic representation of clinical guidelines [3].

When guidelines are used for decision support concerning a patient's treatment, it is very useful to have them draw data from electronic medical records (EMRs). In particular, this linkage will save clinicians the need for entering patient data that already exists in the EMR, thus reducing data-entry errors. Hence, patient data items and medical concepts to which computer interpretable guidelines (CIGs) refer must be mapped to EMR entries [4]. Furthermore, it is extremely useful to allow CIG encoders to represent a guideline using generic clinical concepts and patient data items instead of using institution-specific data models and codes. In this way, CIGs that are developed at one institution may be shared by other institutions. Reusing an

encoded guideline's recommendation among different institutions is an important goal, because the encoding task is laborious and time consuming [3].

GLIF3 has a layered approach for referencing clinical terms. This approach can potentially aid in mapping CIG clinical concepts and patient data items to EMR fields. One layer of GLIF3 defines the clinical meaning of the concepts using codes from medical vocabularies. Another layer defines the structure, or *data model*, of the patient data items [5]. The developers of GLIF3 proposed another layer, called the Medical Knowledge Layer, that will define an interface for linking the guideline encoding with external medical information and knowledge sources, such as EMRs, clinical vocabularies, order-entry systems, and drug hierarchies, yet this layer has never been developed. The work proposed here is an attempt to define a systematic approach for mappings between medical concepts and data items of a CIG to an EMR. In addition, the work presented in this paper defines, in part, the Medical Knowledge Layer of GLIF3.

Runtime application of clinical guidelines

GLEE [6] is an execution engine that executes GLIF3-encoded guidelines. It interacts with a user through a user-interface, interacts with the guideline library (that contains the CIG) and trace records, and provides methods for interfacing with clinical applications, such as an EMR and a clinical event monitor. The core components of GLEE define an execution model to realize the generic functions that are required by the GLIF3 representation model, including an interpreter for the clinical expression language (GEL). This generic architecture, shown in bold contour, is also used by other CIG execution engines, such as GASTON [7], GLARE [8], and Spock, and can be extended to support other functionalities. For example, in the Spock [9] Asbru CIG-engine, the mediation to EMR data is done via the IDAN [10] mediator, discussed below.

Mediation of queries to clinical databases

As GLEE traverses a CIG, it exectes GLIF action steps involving retrieval of patient data from the EMR. GLEE's server interfaces with the EMR to retrieve patient data items. When decision steps are traversed, GLEE evaluates decision criteria, written in the GEL expression language, via the GEL interpreter, which receives binding for the patient data items from EMR records. While the GLIF language allows guideline encoders to specify patient data items using codes taken from standard controlled vocabularies, GLEE does not perform mapping of the standard codes to EMR fields. This functionality is outside the scope of GLEE, which only provides the interface to EMRs, which is defined by the standard terminology code and data model code. Mapping these codes to the EMR is left to the specific implementation. We have previously defined an implementation [11] in which the Get_Data_Action behavior of GLEE. was enhanced to support data retrieval from a relational database; the CIG's data items contain information about the EMR's table name and field name, and the method automatically creates SQL queries to retrieve the data from that table's field. That implementation did not support schema matching and concept abstraction, but instead, relied on a 1:1 mapping between CIG data items and EMR codes. This requires either writing CIG decision criteria that do not use abstractions but instead utilize the raw data stored in the EMR, or defining abstractions that refer to the proprietary EMR terms via GLIF's assignment actions. Both of these options impede sharing the guideline encoding by heterogeneous EMRs. The purpose of this work was to develop a framework that would allow specifying CIGs using suitable clinical abstractions and using the framework to map them to the level of abstraction used by various EMRs, thus allowing CIG sharing. Section 6 discusses how this framework fits as a mediator to EMR data for the GLEE engine.

Among the systems designed to provide data retrieval and abstraction services for clinical applications, IDAN [10] is most similar to the work described in this paper. IDAN [10] is a temporal abstraction mediator used by the Spock CIG-execution-engine to retrieve patient data. IDAN integrates the following set of components: (1) the *knowledge-based temporal abstraction* (*KBTA*) ontology, (2) a patient database, storing primitive facts, and (3) an *abstraction service* that is used to derive abstract facts from the database and knowledge-base. The mediator retrieves primitive data from a local database. This involves the use of predefined tables to map the local terminology, database schema, and units of measure, to the format of the standardized data request (temporal query language). The mediator also retrieves the relevant KBTA knowledge from the knowledge base. This knowledge and the original query are translated into a set of rules in the temporal abstraction rules language. The mediator passes the data and rules to the temporal-abstraction service, which processes the rules and returns to the mediator an abstract-concepts answer set, which the mediator returns to the querying application.

Challenges in mapping CIG concepts to EMRs' data

Defining mappings between a CIG's patient data items and EMR fields involves several challenges. One challenge is that the units of measurements and time granularities of the EMR data may be different than that used by the guideline encoding. Shachar [12] dealt with this challenge by mapping EMR's local measurement units to predefined basic measurement units (e.g., gram) of a primitive domain (e.g., weight) or to primitives between two domains (e.g., weight/volume). She built transformation functions from one basic measurement unit to another so it will enable converting from one basic measurement unit to another.

A second challenge is that different institutions may use different data models and terminologies for expressing the same data entry. For example, 'popliteal pulse of the left leg

diminished' may be represented in one institution with two fields - the item and its value (item: 'popliteal pulse of the left leg', value: 'diminished') whereas in another institution it may be represented using three fields - item, position and value (item: 'popliteal pulse', position: 'left leg', value: 'diminished'). The mismatch in data model and terminology combinations is a problem that has not been completely solved in the context of CIGs. Parker and colleagues [13] tried to tackle this challenge by defining detailed clinical models as restrictions on a virtual medical record in order to specify a broad set of classes of information that are of interest to modelers of clinical guidelines. Their intention was to provide a constrained and standard way of expressing a clinical concept, where all the assumptions about the data representation are made explicit.

A very important challenge in mapping CIGs to EMRs involves the ability of a CIG encoding to use high levels of abstractions, which are concepts or ideas not associated with any specific EMR data, (e.g., "malalignments of the foot" that abstracts away from raw data about particular malalignment types found in the right or left foot). High levels of abstractions provide independence from lower-level implementation details that may change with time or may vary across database systems [14]. We note that creating CIG encodings that contain concepts that are very abstract (e.g., "high blood sugar") may result in undesired ambiguity. So, an appropriate level of abstraction is desired so that a concept is well understood but is not tied to particular EMR implementation. In the literature, we can find different kinds of abstractions: temporal abstractions and concept abstractions.

Temporal abstraction provides a useful and flexible method for obtaining an abstract description of multi-dimensional time series [15]. For example, checking whether the patient has "high sugar level at least 5 hours after treatment" means that the time-stamped data should be analyzed to find an episode of 'high' sugar level whose timestamp is greater or equal to 5 hours

after the timestamp of the treatment episode. This episode, whether found or not, must be reported to the guideline execution unit which will commence response. Therefore, if developers of computer applications in the medical domain are to model successfully the ways in which clinical data are recorded and are used, they cannot ignore the temporal dimension of such data [17].

Concept abstractions fall into two categories: terminology abstractions via classification hierarchies and definitions of abstract terms. In a **classification hierarchy**, a concept that is a parent of other concepts creates an abstraction for the lower-level concepts (e.g., an antibiotic is a parent of penicillin). In **definitions of abstract terms**, abstract terms can be defined using expressions that relate to basic concepts. For example, *isolated systolic hypertension* can be defined as a situation in which patients not taking anti-hypertensive agents have systolic blood pressures of at least 140 mmHg and diastolic blood pressures less than 90 mmHg.

There are many benefits to using abstractions. First, by using abstractions, the encoder of the guideline may use terms that are closer to the professional medical language instead of using some programming or querying language to refer to EMR codes directly, a task that medical doctors are usually not familiar with. Second, abstractions allow encoders to use generalizations that save time and space because they do not need to specify a criterion for each specialized case, but can specify a criterion in general terms (e.g., if antibiotics is given then anaphylactic shock is a possible complication. This does not need to be stated for each particular kind of antibiotics). Third, abstractions are useful to infer clinical situations (e.g., anemia - a condition of low hemoglobin values that lasts over an interval of time) from raw data (e.g., time-stamped hemoglobin values).

There are several researches that tried to fulfill these challenges: Crubézy et al. [18] developed a generic mapping ontology that provides the basis for expressing the adaptation knowledge needed to transform data expressed in one data model and terminology to that expressed in a second data model and terminology. Their work did not focus on patterns in any specific domain but in mappings between two different ontologies. Sujansky and Altman [19] defined a global reference schema of clinical information in the form of (1) a declarative query language that allows to perform queries on the schema and (2) a mapping language to specify the correspondence between the global reference schema and the local relational database. Their work uses relational models (augmented with functions) that allowed them to specify the mappings using a formal algebra. Das and Musen [20] defined a foundational model of time for temporal integration of a knowledge-based application and a clinical database. They used the temporal representations and temporal functions in their timeline model to create a database method that maps time stamps with temporal mismatches into a uniform temporal scheme. Boaz and Shahar [10] developed the IDAN mediator, discussed above in the subsection *Runtime application of clinical guidelines*. IDAN supports schema and vocabulary mapping, unit conversion [12], and temporal abstraction, using the ALMA [10] service. ALMA's temporal reasoning includes: (1) temporal-context restriction; (2) hierarchical concept classification; (3) inference from similar-type propositions that hold over different time intervals; 4) temporal interpolation (bridging disjoin but close intervals); and (5) temporal-pattern matching: creation of intervals by matching patterns in a set of interval-based propositions of various types. Correndo and Terenziani [21] used the HL7 Common Terminology standard services to establish a link between a domain ontology and a database ontology in order to cope with heterogeneous term descriptions. This enabled them to use the GLARE modeling language in a way that is not

committed to any specific ontology and database. Table 1 summarizes the challenges that were addressed and the types of solutions provided by related works.

Types of solutions to linking medical decision-support systems to heterogeneous EMRs

Establishing mappings between a CIG's patient data items and several EMRs in order to allow sharing a CIG-based decision-support system by various implementing institutions is one of the varieties of heterogeneous database integration problems discussed by Sujasky [22], termed *integration for application portability*. The author explains that "the task of heterogeneous database integration is to create a single virtual query model that encapsulates the query models of constituent databases and allows users and programs to access data from the databases using this virtual model". As explained in details above, there are many challenges to that program that stem from representational heterogeneity. The solutions for providing uniform query models for heterogeneous databases may involve data translation into a common shared format or query translation into a set of equivalent local queries [22]. Query translation may be based on procedural or on declarative mapping. Sujansky [22] suggests that database interoperability requires the use of a common data model that is sufficiently simple and abstract to represent the contents of various heterogeneous data models and a corresponding query language that can be used to formulate queries at an equally abstract level. Boxwala and colleagues [23] called such a common data model a *medical concept model*, a framework for describing entities or concepts in the medical domain and the relationships among these concepts. The medical concept model provides a means for expressing the abstractions contained in guideline criteria, in terms of the patient data available in the EMR. Similarly, the medical concept model is called a *virtual medical record* in EON, PRODIGY, and SAGE.

In the remainder of this article, we will present our effort to accomplish two of the heterogeneous database integration challenges discussed above – mismatch in data model and terminology combinations, and enabling the use of abstractions in encoding decision criteria instead of specifying criteria in terms of particular EMR schemas. Our solution is based on declarative query mapping and on a common data model. We claim that by evaluating our work we succeeded to (1) implement the third layer of GLIF3, known as the Medical Knowledge Layer, and to specify methods for interfacing to sources of medical knowledge that are not part of GLIF3 specifications, (2) create a mapping model between GLIF3 encoded guidelines and EMR independent of specific EMRs, (3) create a mapping model independent of CIGs so the encoder of a guideline may encode a guideline without considering the new mapping model, (4) create a mapping model that can be applied to already-encoded guidelines, (5) support the creation of high levels of abstractions by using different kinds of mapping functions along with the ability to compose them together, and (6) create a framework that can be easily extended to future needs. Section 2 presents the methods that we used in order to develop our mapping framework. Sections 3 and 4 describe our framework in detail: the mapping ontology and the building blocks of the SQL queries. Section 5 presents guidelines that we developed for implementing the entire process of encoding and implementing CIGs. Section 6 discusses the runtime application of GLIF CIGs via the KDOM framework. Section 7 describes the evaluation that we performed. Discussion and Conclusion sections follow.

2 Methods

2.1 New Mapping Ontology

We developed an ontology of mapping patterns, called Knowledge-Data Ontological Mapper (KDOM), that can be used to link knowledge (e.g., GLIF3 concepts and patient data items) to data (e.g., EMR fields) and to express abstractions of concepts that can be used to match abstractions used in the CIG model with those used by the EMR. While concept abstractions that are not institution-specific may be defined in a guideline model, institution-specific abstractions are expressed as instances of the KDOM ontology. The ontology was defined using the Protégé-2000 knowledge-modeling tool and supports mappings concepts (CIG data items) to relational databases. We developed a tool that retrieves any information from an EMR by automatically generating Structured Query Language (SQL) queries and getting the desired information as a result set that contains the desired data to be processed by the guideline execution engine.

We decided to develop an ontology for three main reasons [24]: (1) we wanted to separate the domain knowledge (mapping knowledge) from the operational knowledge (CIG and EMR knowledge) so it will be possible to configure the mapping knowledge separately independent from the CIG and the EMR, (2) we wanted to make all the domain assumptions explicit so that it could be changed easily if our knowledge about the domain changes, and (3) we wanted to be able to reuse the ontology knowledge in different CIG formalisms. We developed the ontology bottom-up, so that all of the information would be available in the onotological classes to generate the SQL queries. The ontology represents a model of mapping classes and not real-world entities. A detailed description of the ontology will be presented in Section 3 and the complementary tool will be discussed in Section 2.4.

2.2 Mapping a CIG to Different EMRs

Mapping from a CIG to an EMR is a long and complicated process. Although a model for defining mapping abstractions eases the process, it still requires significant efforts. If we were to create direct mappings from a CIG to an EMR table name and table field, we would have to define each time a new mapping, when linking to different EMRs. The largest barrier to heterogeneous database integration is the variety with which similar data are represented in different databases [22]. Hence, we have investigated methods for reducing the number of new mapping procedures needed when mapping from a single guideline to different EMRs. We looked for ways for representing the mapping from a CIG to a canonical data representation (i.e., common data model, as explained in the subsection "Types of solutions to linking medical decision-support systems to heterogeneous EMRs" of the Introduction) and not to particular EMRs. Mapping a CIG to a canonical data representation will enable us to define only one mapping instead of defining multiple mappings to multiple EMR systems. We call this canonical data representation a **connecting model**, which is used for bridging between a CIG and different EMRs (Figure 1).

In order to select a relevant canonical data representation, we had to check two major aspects: (a) the completeness of the data representation (or, in other words, the possibility of representing the medical information stored in the EMR in the canonical data representation), and (b) the ease of converting existing data representation to the canonical representation. In addition, we had to make sure that there will be no replication of data from an EMR to a new connecting model, but instead, a data *view* would be created. The elimination of the need to copy data from an EMR to a new data representation will alleviate the need for maintenance of another database; the patient's data will always be up-to-date and there will be no need to set a replication policy.

Based on all of these considerations, we selected a subset of the HL7 Reference Information Model (RIM) [25], which is the primary interchange standard for clinical data both in the U.S. and internationally, to be our connecting model. The RIM, previously known as Unified Service Action Model (USAM) [26], suggests a new approach to linking clinical variables used in decision support modules to the EMR, which leaves little manual work to be done when guidelines are updated. The subset of the RIM that we used contains the Act class and three of its subclasses: Observation, Medication, and Procedure, as used as the default RIM in the GLIF3 formalism.

We used the RIM as the basis for connecting and integrating the guideline and the EMR; we mapped the medical terms of a CIG to RIM instances instead of defining concrete instances of a specific EMR. To do so, we defined database views [27] of the EMR using a schema suggested by the HL7 RIM so that we can perform queries based on RIM schema instead of on specific EMR schemas. This method enables the creation of only one mapping from the guideline to the data representation using the RIM standard. In order to link a guideline to a new EMR, we will only need to create new views in a structure defined by the RIM standard. A direct linkage will be created between the guideline data items to the newly defined views and not to the real tables (figure 1). Therefore, changing of EMR data structure will not affect the original linkage of the CIG to the EMR view.

2.3 Guidelines and EMRs Used as Case Studies and Test Cases

To fulfill the research goal, we encoded a guideline for the management of diabetic-related foot problems [28] in GLIF3 using the Protégé-2000 knowledge-modeling tool [29]. At the end of the encoding process and upon the approval of the derived CIG by clinicians who are experts in the

treatment of Diabetes, we had a list of data items that the CIG needs in order to generate recommendations based on patient's data.

Now, that we had an already validated CIG, we embarked on linking it to an EMR. The EMR that we selected as a case study is a web-based system [11] that is used by family practitioners who need to treat their patients who have diabetic-related foot problems. They use this database to record patient data. Based on the data entered, they can then seek the advice of Diabetes experts about needed treatment or referral to hospitals. In order to examine the characteristics of the needed mappings from the diabetic foot CIG to the EMR, we manually analyzed the data items that were needed by the CIG and tried to manually match them to EMR fields [30]. This analysis was performed in full collaboration between the encoders of the guideline and the EMR's experts. The analysis result was a complete list of needed mapping instances between the CIG and the EMR.

Based on the analysis results, we defined mapping instances between the CIG and the EMR in two different phases. In the first phase, we defined direct mappings between the two, so that the mappings were tightly coupled to the EMR schema. In the second phase, we defined the mappings using the mapping ontology that we had developed.

To provide a proof of concept that shows that the mapping process can be applied to a test case, we used our mapping ontology to map the CIG to a second EMR. We chose an EMR called CliniCare, which was about to be deployed at Carmel Medical Center in Israel for storing information about patients from the first moment that a patient who is admitted requests a service till his release from the hospital. This EMR is already used in nine other medical centers in Israel. Since the CliniCare EMR was not used by clinicians from Carmel Medical Center for following diabetic foot problems, we had to identify the relevant EMR fields of that EMR that

could have been used for that purpose. Some of these fields were already defined (e.g., type of diabetes), and some were not (e.g., Doppler test results). To define the set of relevant fields, we used the Screen Form for Diabetic foot [28] that was suggested by the developers of the Diabetic Foot guideline, and we consulted with a clinician working at Carmel Medical Center (YD) and with a database administrator familiar with the EMR schema who provided us with all the needed information. Now that we had a second EMR that contained relevant fields, we wanted to link its fields to the original mapping ontology instances that we created when we linked the CIG to the first EMR. To do so, we created new database views of this EMR, created ontology instances that refer to the view and represent the relevant EMR fields, and we linked them to the existing mapping ontology instances.

In addition, we tested 16 different previously-encoded guidelines [31-46] from different medical domains. These guidelines were encoded by different medical information specialists. 15 guidelines were encoded using the GLIF3 computer-interpretable language and one guideline [37] was encoded using the SAGE [47] guideline model. We decided to use an encoded guideline from a different CIG formalism to evaluate the capabilities of the mapping ontology for mapping data items represented in a different guideline representation.

2.4 A Tool for Generating SQL Queries

Defining ontology for mapping patterns is an important step in achieving new mappings abilities and easing the mapping process from a CIG to an EMR. However, to complete the process, we need to generate automatic queries to an EMR, based on the mapping ontology instances. In order to do that, we developed a tool that retrieves ontology instances as an input and produces SQL queries as an output. According to the classification of Sujansky [22], the mechanism that

we implement performs declarative query translation and may optionally use the HL7-RIM as a common data model (connecting model).

We developed the tool using the *JAVA* programming language, which supports the ontology-design ideas that originated from object-oriented design [24]. Multiple inheritance is not supported in *JAVA*, so when we developed the mapping ontology we avoided using this capability so that the classes' representations in the ontology and in the tool would be as close as possible.

3. The Mapping Ontology

The mapping ontology contains 30 classes that represent the different mapping patterns and data representation. The properties that the classes hold can be divided into two groups: one group of properties conceptualize the abstract knowledge (e.g., combines knowledge in temporal relations, in hierarchies, or in logical combinations) and another group helps in mapping and retrieving fields to and from a target EMR table(s).

In the rest of this section we will present the main class of the mapping ontology: *mapping* classes, which provide information for mapping between data items used within a CIG and the corresponding EMR fields. The other classes, which are not elaborated for the sake of brevity, are *operator* classes, which combine information from several EMR fields (e.g., the temporal operators latest, first, and exists, and logical operators: and, or, not), and *result* classes, which define what type of information is retrieved from an EMR (e.g., Boolean, actual value of a field). Figure 2 summarizes the main classes.

When one data item in the guideline is mapped to a single field in the EMR we call it a *one-to-one mapping*. In other words, the mapping result from the EMR will be based only on one

field. Therefore, the one-to-one mapping class has a slot that refers to the EMR's destination field. We defined three types of one-to-one mappings: direct one-to-one, temporal abstraction, and classification hierarchy mapping.

The one-to-many mappings allow combining many fields using logical operators and allow the combination of prior (i.e., previously defined) mapping instances with other mapping instances to create more complex mappings.

3.1 Direct One-to-One Field Mapping Class

This class is used to map a single source data item to a single destination field. It uses a comparison slot (holding an instance of the Comparison class) to specify how the retrieved value should be compared to a constant value. The comparison types include: equal, greater than and less than.

3.2 Temporal Abstraction Mapping Class

It is common that a mapping to an EMR may return more than one relevant result whereas the guideline needs only one result from all the valid results. In order to reduce the returned values, it is useful to allow returning the most relevant result, from a temporal perspective. The temporal operators that we currently support include, restricting the query to a time range, returning the first or latest value, checking whether a Boolean term holds for the patient (i.e., existence checking, such as, does the patient have an ulcer), and the *Current* operator, which returns the latest value during the recent period of time, where the recent interval can be defined for each concept.

Two considerations influenced our choice of temporal operators that are currently supported. First, we examined the decision criteria that refer to temporal aspects of patient data items (e.g.,

cough that started within four weeks after starting to take the drug ACE-Inhibitor) in the 15 GLIF-encoded CIGs and the one CIG encoded in the SAGE formalism. The set of temporal operators that we support, together with the ability to compose mappings, was enough to represent all of the temporal concepts that we encountered. The combination of mapping instances can utilize different types of mappings: Boolean and Extended Boolean, Temporal, Hierarchical mappings, and Prior mappings, that allow comparing values retrieved by two mappings, or comparing a value to a literal. The second consideration was that the GLEE execution engine uses the GEL parser to interpret decision criteria encoded in GLIF. At the moment, the GEL parser does not support temporal operators, so guideline encoders cannot write decision criteria that make use of GEL's temporal operators. Instead, we allow the GLIF encoders to write decision criteria that refer to abstract temporal terms (e.g., ACEI-related-Cough). KDOM is then used to define the abstract temporal terms, via the ability to compose mapping instances to create complex mappings. In the previous example, a KDOM temporal mapping instance can be defined to represent the start-time-of-the-latest-ACEI and a second temporal mapping instance can represent cough that started within the interval of at least 1 week after the first instance and at most 4 weeks after the first instance.

The Temporal Abstraction Mapping class is not intended to be a formal method for the specification of elaborate temporal-abstraction knowledge (e.g., trends such as increasing temperature, or abstracting concepts that hold over an interval of time from episodes of time-stamped data) as is done in the RESUME [48] and IDAN [10] projects but to support only basic operators, mentioned above. While the specification of temporal abstractions via mapping instances is possible, it does not include convenient templates and therefore the specification of complex abstractions may span many mapping instances.

The temporal abstraction mapping class defines the type of temporal abstraction operator and a time range for bounding the mapping for a selected time range from the entire patient's history. Figure 3 illustrates an instance of Temporal Abstraction Mapping class for returning the first visit of a patient during the year of 2004.

The SQL query that is automatically generated for this example is:

SELECT Min(Visit.Time) As TemporalResult //see Section 4.1, 4.2.7; Fig 3(2-3)

FROM Visit, Patient //see Section 4.1; Fig 3(4b)

WHERE Patient.PatientID = 304553341 AND (//see Section 4.2.1; Fig 3(4a,b)

(Visit.PatientIden = Patient.PatientKey) AND //see Sec 4.2.2; Fig 3(4b)

(Visit.Time >= '1/1/2004' AND Visit.Time <= '31/12/2004')) // 4.2.5; Fig3(1,5)

Section 4 explains how the information in the mapping instances is used to automatically generate SQL queries.

3.3 Classification Hierarchy Mapping Class

We often encounter a guideline term that is too general to appear as a field in an EMR. For example, the guideline may describe a term such as antibiotics, without mentioning its type, whereas the EMR stores fields representing the types of antibiotics without referring to the general term *antibiotics*. In these kinds of situations, we cannot link the data from the CIG to the EMR directly because the data is represented in two different levels of generalization.

In order to solve the problem of different generalization levels, we used a class for representing medical hierarchies, called *Hierarchy class*. The information that this class holds is represented by a tree-like data structure with different levels of generalization, where the

information stored in the upper levels are more general than those stored in the lower levels and in the leaves.

Using data structures for representing hierarchies enables the mapping of a CIG data item to a node representing it in the medical hierarchy. The node represents the complete information from the most general node to the most specific leaf. A selected subtree represents information relevant to the selected data item, so we can generate a query to the EMR that includes all the possibilities of the general term that are required by the CIG. To illustrate this, let's assume that the hierarchy contains a simple data structure where the root contains the *pharmacological substance* data item with one child node of *antibiotics* and beneath it two leafs: *Amoxycillin* and *Penicillin-V* data items (Figure 4). Now, let's assume that the guideline needs an EMR data item representing the antibiotics given to a patient. By mapping to the hierarchy node representing the *antibiotics* data item, we can take the whole subtree from the antibiotics node underneath, which contains the antibiotics *Amoxycillin* and *Penicillin-V* data items, in order to look for them in the EMR field that holds this information (e.g., a field that holds the name of a given drug that a patient has taken).

Because this type of mapping is from one field in the EMR to one data item in the CIG, this class inherits all the properties of *one-to-one* mapping class. In addition, it defines the following other properties to conceptualize the abstract medical knowledge: (1) Medical hierarchy property, to hold the medical hierarchy and (2) a medical term, representing the information we would like to locate in the hierarchy.

Figure 4 shows a diagram of a hierarchy of pharmacological substance. Figure 5 shows a use of the Classification Hierarchy Mapping class to find out when a patient has taken antibiotics

3.4 Binary Logical Mapping Classes

The Binary Logical Mapping links two EMR fields using logical operators OR and AND in order to create logical results among the fields. The result is retrieved as one data item to be used by the CIG. In addition, we support extended logical mapping operators (used in the Extended Logical Mapping Class), including **at least** or **at most**. These operators mean that at least (at most) n data fields are valid. In this case, the returned value would be a positive value.

Although implementing such defined condition using *logical operator* is possible, it makes the mapping process and the definition of the conditions very complicated. It becomes even more complicated when the condition is defined using many EMR fields. To illustrate the problem lets consider the following example of selecting at least 2 fields among 3 fields a, b, c. This condition can be implemented using logical operators in this way: *(a AND b) OR (a AND c) OR (b AND c)*. On the other hand we would like to define a dedicated operator for defining the selection of n from m field so it will be possible to define the same condition as: *at least 2 of (a, b, c)*, which has the same meaning as the previous condition but is more intuitive and is in shorten notation.

3.5 Prior Mapping Class

Using prior mapping for a new mapping enables the composition of mapping functions that may strengthen the ability to map from the basic elements that we have defined. In such a way, we can combine different mapping abilities and retrieve data based on previously mapped functions.

The prior mapping can be any instance of any mapping class. In this way, any type of mapping can be used, because every mapping function that we defined inherits from the mapping class. In addition, we can treat the prior mapping as a regular table with a regular table name

exactly like we treat a regular EMR table with the ability of selecting one field among all the fields that we have retrieved previously.

Figure 6 illustrates how the *Charcot* data item that was mapped using an instance the Binary Logical Mapping class.

4 Building Blocks of the SQL Queries

We developed a tool that gets ontology instances as an input and produces SQL queries as an output.

The queries that the tool produces are built from 3 main parts [27]:

(1) SELECT **table-columns-list**

(2) FROM **table-list**

(3) WHERE **qualification** //selection conditions on tables rows

4.1 Select and From Clauses

The one-to-many Mapping classes are composed of prior mappings that refer to other one-to-one or one-to-many mappings, eventually ending at one-to-one mappings. The select table-column-list is determined by the *MappingResultType* slot (see part 2 of Figure 3), of the mapping class, that specifies the type of result to be returned from the selected tables and columns. For example, should the field's value be retrieved or just a true value if the field exists or just the number of rows that satisfy the qualification. Other result type options enable returning all the fields of a record or all records in a table that satisfy the qualification. It is also possible to select fields for retrieval by setting them using the resultField slot that enables multiple selections.

The table-list is determined, in part, by the destinationField slot of the one-to-one-mapping classes. As shown in part (1) of Figure 3, this slot gives information on the EMR's (or

the connecting model's) table and field from which the data item should be located and retrieved. When a mapping class refers to Prior Mapping classes a sub query is added to the tables list with an aliasing table name. In this way, the whole sub query will be treated like a regular table name.

The second part of the mapping class that determines the table-list is the *PatientEMRSpecificField* slot, which specifies the field in a certain table needed to identify the data belonging to a particular patient. In some cases, the destinationField and PatientEMRSpecificField are found in different tables. In this case, the PatientEMRSpecificField contains a Join slot that links these two fields (see Figure 3, parts (1) and (4b).

When a mapping class refers to Prior Mapping classes a sub query is added to the tables list with an aliasing table name. In this way, the whole sub query will be treated like a regular table name.

4.2 Qualification Clause

The qualification query part is used for five different purposes:

1. Identifying data belonging to a particular patient. This was implemented using the condition *<Patient Table>.<Patient Field> = <Selected Patient ID>*, where *patient table* and *patient field* are taken from the patient_EMR_specific_field slot of the mapping class and patient_id is taken from the patient_id slot of the mapping class, as shown in Figure 3(parts 4a-b).
2. Matching rows from two or more tables based on values in common columns. This was implemented using the condition *<table 1>.<field 1> = <table 2>.<field 2>* where these four parameters are specified by the destinationField slot of the one-to-one mapping classes, which holds an instance of the Destination Field Information class (an example of an instance of this class is shown in part 4b of Figure 3). *table 1* is specified by the tableName slot of the Destination Field Information class; the other three fields are specified by the *join* slot of the

DestinationFieldInformation class, which holds an instance on the JoinTable class:. *field 1* is specified by the joinSourceFieldName slot of the Join Table class, and *table2* and *field2* are specified by the joinDestinationTableName and the joinDestinationFieldName slots respectively.

3. Listing of all medical terms in a hierarchy of medical terms. The query is written in the form of <*selected table*>.<*selected field*> *IN (medical term 1, 2, …, n)* where *selected table* and *selected field* are specified in the DestinationField slot of the mapping class, the given medical term is specified by the medical_term slot of the Classification hierarchy Mapping class and *medical term 1…n* represent one general medical term and all its specific terms represented by the medical hierarchy that is accessed via the medical_hierarchy slot of the mapping class., as shown in Figure 5
4. Comparing selected EMR fields to defined values. This is done by using <*selected table*>.<*selected field*> *op* <*comparison value*> notation, where *selected table* and *selected field* represent one EMR field are specified by the mappingAliasTableName and referenceFieldName slots respectively of the Prior Mapping class (see the right part of Figure 6), *op* represents one of the comparison operators {<, <=, =, <>, >=, >} and *comparison value* is a value that the selected EMR field should be compared to. They are expressed by the sign and the value slots respectively in the Comparison class, which is referenced by the compareReferenceField slot of the Prior Mapping class, as shown in Figure 6. This comparison can be used to compare one field or to compare multiple fields using a logical combination operator (e.g., AND or OR).
5. Bounding the mapping for a selected time range from the entire patient's history. This is done by applying <*selected minimum table*>.<*selected minimum field*> >= <*minimum value*>

AND <selected maximum table>.<selected maximum field> <= <maximum value> where the selected minimum and maximum tables and the selected minimum and maximum fields are taken from the DestinationField slot of the Temporal Abstraction Mapping class (Figure 3(1)) and the minimum and maximum values are taken from the *within* slot of the Temporal Abstraction Mapping class, as shown in Figure 3(5).

In addition to the three main query parts, the tool uses aggregation functions in order to produce queries in the following situations:

6. Using the Extended Logical Mapping class to determine validity of n among m fields. Each field is specified by an instance of Prior Mapping class using the priorMappingFunctions slot that is referenced by the Extended Logical Mapping class. The tool generates one query to check validity for each field among the m fields. The query contains *condition* for checking the validity of that field and a *grade* equal to 1 to be returned *if and only if* the condition is met. The condition is expressed using the Comparison class, which is referenced by the Prior Mapping class (an instance of the Comparison class is shown on the bottom right of Figure 6). All the m queries are combined using the UNION ALL operator so the results will contain the grades of all the queries representing the m conditions. To check the number of conditions that are valid, the SUM aggregation operator is used to sum of all values. This sum result is tested against the original n parameter, so if the *at least* operator is used then the sum result must have at least the value of the n parameter. The *at most* operator is checked in a similar way.
7. Using Temporal Abstraction Mapping class to retrieve temporal values. This was implemented using the MIN and MAX aggregation operator for selecting the *first* and the

current value, which are specified in the temporalAbstraction slot of the TemporalAbstractionMapping class (part 3 of Figure 3).

Note that the presented aggregation functions represent only a small part of all the possible aggregation functions of SQL. However, the mapping ontology and the tool could easily be extended to support more types of aggregation operators when needed.

5 From a Guideline to an EMR

The process of encoding and implementing CIGs is not straightforward and is done by medical information specialists and clinical experts. We think that the product that we developed will provide such groups with the ability to directly and more easily encode decision criteria using clinical abstractions that are used in the original narrative guideline without taking into consideration the proprietary EMR schema that the guideline will be linked to. This provides encoding abilities that are not available nowadays and that should shorten the process of existing encoding methods. Moreover, the whole process from encoding a guideline to mapping it to an EMR might be performed by different people with different knowledge and expertise, because the whole process can now be divided into different tasks, as outlined below. Peleg and colleagues described the steps needed in order to encode one guideline and to link it to one EMR [30]. We will present here our recommendations for encoding a guideline and linking it to an EMR that can be easily extended later to mappings to more than one EMR. The first two steps and the last step of this process are similar to the steps of the process described in [30]. Figure 7 illustrates the stages needed in order to encode one guideline and to link it to one EMR. The stages may be done in different order, the following being just one of the possible sequences.

a. **Encoding a guideline –** we propose that a medical information specialist should encode a narrative guideline as a CIG. She may consult a physician for any medical clarification. In order to achieve sharability and reusability of an encoded CIG across institutions, the GLIF model allows the encoder to define both raw and abstract concepts used in the CIG to a set of international terminology standards. Searching for such codes [12], which is not supported by KDOM tools at the moment, can facilitate mapping the CIG's codes to the EMR, as described below. An alternative to encoding the CIG initially by a medical information specialist, suggested by Shalom et al [49], suggests that the guideline may be encoded by developing a consensus, high-level representation of the guideline, performed by a team of informatician and medical expert, then, a semi-formal encoding can be done by a clinician, followed by formal encoding by an informatician (knowledge engineer). In any case, at this stage, no mappings to an EMR should be declared

b. **Defining Mapping instances that define CIG complex abstractions –** a mapping creator creates mapping instances that define complex CIG abstractions (used in CIG data items) by creating and combining other mapping instances representing simpler abstractions

The following three steps are used to create ontology instances that obtain the relevant data from the EMR

c. **Identifying EMR table fields needed by the CIG –** the database administrator should get a list of data items needed by the CIG from the medical information specialist. He should identify the relevant fields in the EMR, based on the given list. In simple cases, locating the terms in the EMR that need to be mapped to the CIG concepts could be done manually, as we did when we mapped the Diabetic Foot guideline into two EMRs.

However, manual methods might not suffice. Since clinical databases often include thousands of data types and vocabularies often includes tens or hundreds of thousands of terms, this task might in fact require the use of sophisticated vocabulary servers [10] and various mapping strategies [50], that may take advantage of the GLIF-encoding of CIG concepts using terms from standard terminologies. At the moment, the KDOM framework does not support aid in locating terms

d. **Defining database RIM-views of the needed EMR fields –** a database developer should get the list of fields from the database administrator and a design of the RIM structure from the medical information specialist as an input, and he should define database views on top of the EMR in the form of relational tables that correspond to a *flattened* model of the HL7 RIM. In the flattened model, a single table schema is generated for an HL7 Act subclass. Act attributes that are simple data types are translated into single fields in the flattened table. Attributes of complex types (e.g., instances of other classes) are recursively broken down until simple data types are reached. For each simple data type, a field in the flattened model is created. From here on, we can work with the views instead of with the EMR

e. **Defining mapping ontology instances for accessing each needed EMR field from its RIM view**– The purpose of this step is to create ontology instances for each needed EMR field, but, instead of referring directly to the EMR fields, referring to the views (defined in the previous step). The instances created are of the class Destination Field Information. This step allows us to access the needed EMR fields from the views. Any person who learned how to use the mapping ontology can define these instances

The following two steps are used to create ontology instances that define how to retrieve the CIG's data items from the ontology's representation of EMR fields

f. **Creating mapping instances that access Destination Field Information instances –** the encoder of the guideline should create instances of the Mapping classes for each Destination Field Information instance defined in Step 4. These mapping instances represent abstractions used by the EMR schema

g. **Defining Mapping instances that bridge the gap between the simple abstractions used in (b) and the abstractions used by the EMR schema –** a mapping creator creates (1) classification hierarchies of concepts that relate the guideline's concepts with the concepts used in the EMR schema (e.g., the example of figures 4 and 5 that allow the guideline to refer to Antibiotic, while the EMR refers to drug names of Amoxicillin and Penicillin-V) and (2) logical combinations of more basic EMR fields to define abstractions used by the CIG

h. **Validation –** At the completion of each step, a validation needed to be performed. In addition, at the completion of all of the previous steps, a medical information specialist may execute several test cases in order to confirm the entire flow.

Notice that the suggested steps can be completed in different order, for example after (a) encoding the guideline and creating mapping instances for defining CIG abstractions (b) one can (e) define ontology instances for accessing each needed EMR field from its RIM view that will be created later, (f) create the mapping instances of the CIG's data item and (g) define mapping instances that bridge the gap between the abstraction of concepts used in the CIG's data items and the abstractions used by the EMR schema. Now it is possible to (c) identify EMR fields needed by the CIG and to (d) define database RIM-views of the needed EMR fields. (h) As

mentioned previously, a validation should be performed at the completion of each step and at the completion of all the previous steps.

When mapping to a new EMR only three steps must be done: (c) identifying the new EMR fields needed by the CIG, (d) defining database RIM-views of the needed EMR fields and (h) validation as explained earlier. These steps alone will be sufficient to map to a new EMR that is representing the same needed information (probably in different data representation).

6 Runtime-application of GLIF CIGs via the KDOM framework

Although we have not yet integrated the KDOM framework into the GLEE execution engine, we outline this simple integration process in this section. As mentioned in section 1, our implementation of GLEE's get data action supports binding of CIG patient data-items to EMR fields by specifying table and field information [11]. Using this information, GLEE generates SQL queries for retrieving the desired data from an EMR. The support of SQL by GLEE serves as the major element of integrating KDOM with GLEE. The idea is simple; instead of using the implementation's ability to generate SQL queries, KDOM will generate the SQL queries, and as before, GLEE would execute the SQL queries to retrieve the data. The approach is illustrated in Figure 8. Data-item specifications in GLIF CIGs will contain references to mapping ontology instances. These mapping instances hold all the information needed for mapping to an EMR. Whenever GLEE would identify an instance of the mapping ontology in a CIG's data item, it would send it to an external component, provided by the KDOM framework (the SQL generator), in order to get an SQL query as a result. GLEE will execute the SQL query to get the result from the EMR, as before. Note that the GLIF CIG will not have to be changed when mapping to different EMRs. This is because the mapping instances to which the CIG will refer

will be those that define the high-level CIG abstractions rather than the mapping instances that are EMR-dependent (see Figure 7).

7 Evaluation Studies

The main goal of this research was to create a mapping ontology that could map a single guideline encoding into various EMR schemas, thus supporting guideline sharing. The first evaluation study, described below, was aimed at evaluating this goal. A second evaluation study addressed the suitability of the mapping ontology for mapping abstractions used in different kinds of guideline encodings. Recall that since the GEL parser does not support temporal operators, GLIF encoders can write decision criteria relating to abstract temporal concepts and use the mapping ontology to define these abstractions. Therefore, the second evaluation study addressed the suitability of the mapping ontology to support mappings of abstract concepts found in guidelines from different medical domains that are encoded in different formalisms.

The first step of our first evaluation study was to use our mapping ontology as a replacement for direct mappings that we created between the CIG and the case study EMR mentioned in section 2.3. We used a set of criteria that were defined by Sujansky [14] by which we tested the usefulness of the ontology and the tool.

The first evaluation criterion is the ***latitude*** criterion, which assesses the ability to *completely* map a guideline to an EMR, and thereby support automation query translation. To evaluate this criterion, we followed the following steps.

(1) We rewrote the original GLIF3 (GEL) decision criteria that had referred directly to the EMR terms as criteria that refer to abstract concepts (e.g., Charcot = true instead of a criterion that referred to the basic terms found in the EMR from which Charcot could be determined).

(2) We created mapping instances using the mapping ontology in order to define the abstract terms used in the new GLIF3 criteria.

We replaced all the 30 direct mapping functions that we previously defined with mapping instances that bridge the gap from the abstract criteria to the EMR fields. We found that all of the 30 direct mapping functions could be replaced using instances of the new mapping ontology. Following the replacement process, not even a single mapping function references the EMR directly, but accesses it through mapping instances.

In addition, we used the new complementary tool that we developed in order to test the automatic query translation based on the mapping instances. This test required executing each query that the tool produced in order to evaluate the correctness of the result sets returned from the EMR. All of the result sets exactly matched the ones returned when we used the direct mapping functions, confirming the automatic query translation.

A second criterion was ***declarativeness***, which assess the ability of the mapping functions to map to an EMR without requiring extensive use of specific EMRs functions. To evaluate this criterion, we mapped the instances of the mapping ontology directly to the EMR fields and not by using database views in a RIM structure. This step was necessary because creating the database views may involve the usage of database functions. The result of this assessment was that all of the 30 mapping functions could be mapped directly to the EMR schema using the mapping ontology with no changes required to the EMR schema, even without creating the database views.

A third criterion that we assessed was ***expressiveness***, which assesses the ability to represent the semantics of clinical queries (decision criteria) that are typically required by CIGs. It is important to mention that, in theory, even if a mapping function cannot be represented fully

using the mapping ontology, it can still be partially represented using the mapping ontology and the rest could be represented by changing the GLIF3 encoding or by defining new constructs in GLIF3's mapping layer. However, our vision is that all of the mapping functions will be described fully using the mapping ontology, which will enable the complete separation of the encoding process from the mapping process. In the CIG that we encoded, we succeeded in creating all the 30 mapping functions using the mapping ontology without using of the functions of the GLIF3 expression languages. This assesses the third criterion, while noting that complex temporal abstractions, complex aggregation functions, and unit conversions are not supported, which limits the potential expressiveness of the current KDOM framework.

Another aspect of the evaluation is showing that only ***minimal changes***, both to the EMR and to the GLIF3-encoded guideline, are made following the mapping process. To show that, we traced all the changes we made while we replaced the direct mapping with the mapping instances that bridge the gap between CIG abstractions and EMR fields. We found that no changes had to be made to the EMR schema or to the EMR data following the mapping process. The only changes that we made were to define new database views representing the EMR data as RIM instances to be used by the guideline. Note that new database views can be treated as a new layer required by the mapping ontology and not as a major change to the EMR structure. We can say that the task of creating database views should not be difficult to learn because it requires only technical steps that could be easily learned.

To assess the changes made to the GLIF3-encoded guideline during steps 1 and 2 described at the beginning of this section, we distinguished between two categories: *algorithmic specifications* of guideline logic, and *declarative medical knowledge* that logical criteria may refer to (e.g., definitions of abstract terms). Because the mapping ontology should not change the

algorithm of the original guideline but should only reflect a different approach for representing mapping criteria, we expected not to need to change any of the algorithmic specifications but only declarative medical knowledge. Indeed, the changes that we had to make included defining (as ontology instances) concepts used in GLIF3 criteria in terms of the fields used in the EMR, which were sometimes less abstract (i.e., more detailed) than the concepts used in the revised GLIF3 encoding (step 1 above). This did not involve any change to the guideline logic. Therefore we can conclude that only minimal changes were required both to the EMR and to the GLIF3-encoded guideline. We believe however, that in the general case, where the starting point is a general guideline encoded in GLIF3 without consideration of the local EMR fields (step 1), then during the process of establishing the mapping from the guideline to the EMR, it may be necessary to add or change some of the information both in the EMR and the CIG in order to complete missing information and to support better mapping functions to achieve improved decision support. For example, we may decide to add EMR fields that are necessary for the guideline to provide decision-support, or we may find EMR fields that are very important to the clinicians who use the EMR and, in their opinion, should be included in decision criteria of the GLIF3 encoding of the guideline.

Showing ***transferability*** to the architecture of another EMR that does not share the same data model is a major aspect of the evaluation process. The transferability should not affect the original mapping to the first EMR and should involve minimal changes both to the new EMR and to the original CIG. To fulfill this requirement, we did not make any changes to the original CIG and treated it as a 'black box'. In addition, we did not change the schema of the new EMR mentioned in section 2.3 but only built new database views to be used by the mapping ontology. In 24 out of 30 cases, we could reuse our original definitions of abstract CIG concepts as

mapping instances that refer to more basic concepts (e.g., the definition of Charcot, as "malalignment and erythema and swelling"). We could even keep the Destination Field Information instances that relate to the RIM view (not to the specific EMRs). However, the new EMR contained relevant data that the original EMR lacked (e.g., the new EMR contained information about dry skin of a patient where the first EMR lacked this information) and vise versa. Therefore, in the original mapping knowledge base, we created additional instances of mapping instances to the new data. The additional data does not affect the original mapping to the first EMR because data that cannot be reached by the mapping ontology instances is treated like it was not defined in the first place. This is done by combining expression that use data that is found only in one of the EMRs with expressions referring to data found in both EMRs using OR logical operator. So although additional mapping functions were defined to retrieve information from the second EMR, the original mapping to the original EMR 'ignores' them because the information they are representing cannot be found in the first EMR. The same rule also applies when data that can be reached from the first EMR cannot be found in the second EMR. By making changes only to the mapping ontology to support additional information and by building new database views and not changing the new EMR schema and the original CIG, we accomplished the transferability requirement.

Our second evaluation study targeted the claim that the mapping ontology can be used to map guideline data items to EMRs in ***many kinds of encoded guidelines***, we tested 16 different previously-encoded guidelines [31-46] from different medical domains, as explained in the Methods section. We evaluated each encoded guideline using the following steps: (1) identifying every single data item that needs to be mapped to an EMR, (2) analyzing the mapping expressions needed to map the data items to our connecting RIM model, (3) replacing the

original GLIF3 (GEL) expressions, which referred to elementary data items and did not use abstractions, with GEL expressions that use abstractions together with mapping ontology instances that map the abstractions into the elementary data items. Furthermore, in every one of the original CIGs, we (4) replaced all the vocabulary codes, such as codes of the Systematized Nomenclature of Medicine - Clinical Terms (SNOMED-CT) [51] with the mapping ontology's Hierarchy instances, in a way that the Hierarchy instances represent the same information as the SNOMED-CT codes. Our success in performing all these steps is a major indicator for the completeness of the mapping ontology, which can thus serve as a mapping layer between CIG data items and EMRs.

Fifteen of these guidelines were encoded in GLIF3 and one was encoded in the SAGE formalism. We decided to use an encoded guideline from a different CIG model to evaluate the capabilities of the mapping ontology for mapping data items represented in a different guideline representation.

Tables 2a-c show the categories of mappings that we created in order to map all the data items that should be linked to an EMR, for each encoded guideline. Due to the large size of the data, we split the results into three tables (arbitrary partition). In each table, each row represents a different type of mapping instance, while each column represents a different encoded guideline. For example, in the Cough guideline, we needed to create six instances of BinaryLogicalMapping class combined with TemporalAbstractionMapping class, seven instances of TemporalAbstractionMapping class, and two instances of BinaryLogicalMapping class combined with ClassificationHierarchyMapping class. Each data item was mapped using one of the different types of mapping types. The row shown in bold, in each table, represents data items that could not be mapped to EMR fields using mapping instances. As the tables show,

we succeeded in creating mapping instances for all the data items in all of the encoded guidelines, regardless of their CIG language: GLIF3 or SAGE, which strengthens the evaluation of the ***expressiveness*** (third) criterion and shows that the framework is applicable to ***other CIG formalisms***.

The guidelines that we used for evaluation were encoded in different levels of generalizations: in 11 guidelines [31-33, 36, 37, 39, 41, 43-46] all the data items that should be linked to an EMR have been explicitly defined, while in five guidelines, shown in Table 2c, the decision criteria did not explicitly characterize the data items that should be mapped to an EMR. We used these differences in the details of the CIG data items in order to test the characteristics of the mappings done in these two sets of guidelines.

We categorized the 513 mapping expressions that we generated for the 513 data items contained or derived from the 16 guidelines into the different mapping classes needed to realize them and identified new design patterns, consisting of basic patterns that always go together. This may help to identify mapping characteristics in CIGs to be later used in order to develop new mapping patterns, as discussed in the Discussion section.

It is important to mention that the PriorMappingClass, which is used to combine more than one mapping function (i.e., represented by rows that include combinations of mapping functions), was used in 217 of the 513 patterns (42.3%) of the mapping functions. Thus, we can conclude that the ability to combine more than one mapping function is a very important feature.

Another interesting aspect that we can learn from the results is that the ClassificationHierarchyMapping class was used in 270 of the 513 data items (52.6%). This shows that abstractions were used often to reuse mappings and also to simplify decision criteria in the modeling language. It should be mentioned that the percentage of data items that were

mapped using ClassificationHierarchyMapping out of the total number of data items was much higher in the guidelines for which decision criteria were defined but the involved data items had not been defined (102/140 = 73% compared to 168/373 = 45%). In these guidelines we were not restricted to any design limitation, so the frequent use of ClassificationHierarchyMapping reflects the best design of mapping functions using the mapping ontology.

Discussion

The main contribution of this work can be concluded in five main points:

- **Reuse** – Mapping instances that have already been defined in one place can be used without any change in different mapping functions, when needed. Mapping instances can be used when several abstract concepts all refer to the same simple concept. For example, both *infection* and *Charcot* patient states are determined based on erythema diagnosis. When defining one mapping instance for erythema referenced by the *infection* mapping instance, we can refer to the same instance when we define the *Charcot* instance. The reuse may span several guidelines, as a data item may appear in several guidelines.
- **Physical separation between the encoding and the mapping processes** – the encoder of the guideline may focus only on the encoding process and leave the mapping process to other people. In this way, the complicated task of encoding and mapping to an EMR can be divided into different subtasks, which may be done by different people. For example, in the current implementation of GLIF3 and its execution engine, GLEE [6], defining *malalignment* involves explicitly defining in the CIG all the proprietary EMR fields. This may involve complicated condition such as: *(mal_forefoot_L = "Yes" or mal_forefoot_R= "Yes" or mal_mid_L = "Yes" or mal_mid_R = "Yes" or mal_hind_L =*

"Yes" or mal_hind_R = "Yes"). When using the mapping ontology, the CIG encoding may state a simple decision criteria (*malallignment* = "yes") while mapping instances may be created to map the *malallignment* concept to a simple field in a RIM view representing one EMR, or to a logical combination of several fields, in a second EMR. This enables us to write the CIG's decision criteria in the most abstract terms: (e.g., Charcot = "yes") and reuse a mapping instance that defines Charcot as an abstract concept (e.g., Charcot := (malallignment = "yes" and erythema = "yes" and swelling = "yes")) when mapping to several EMRs.

- **Mapping to EMR by different people with different skills –** The mapping process task may be divided into subtasks of mappings, each one at a different level of abstraction, which can be done by different people with different prior knowledge and level of expertise. For example, the explicit definition of the EMR's fields may be defined by the DBA of the EMR, while the conditions based on the fields may be defined by a medical doctor who is familiar with the medical knowledge but is not familiar with the EMR data structure.
- **Easing the readability of the GLIF3 decision criteria and mapping functions –** higher levels of abstractions should be more readable than lower levels of abstractions because higher levels terms are familiar to a wider group of people and are closer to the language that they use every day. For example *malalignments=true* is more readable than seeing all the types of malalignemnts of the different feet (left and right) in one mapping function. The mapping functions are built at different levels of abstraction and we can read the written mapping functions from top (most abstract) down (most concrete), which potentially eases the reading process.

- **New mapping functions can be easily added –** mapping functions can be added in two ways. One way is to add a new Mapping class into the hierarchy of mapping classes. In this way, the new mapping class will inherit all the properties of its ancestor mapping classes. This mapping class could be used by instantiating it. An alternative method for creating new mapping functions is not to create a new mapping class, but to use the Prior Mapping class, which composes mapping functions in a way that enables a mapping function to refer to other mapping functions. One can argue, however, that the side-effect resulting from combining several mapping functions in order to create a single new mapping function is that the mapping creator will need to define many instances of Prior Mapping, instead of defining only one instance of a *new* mapping class. To overcome this side-effect, it is possible to use the EasyEdit tool [52]. The Easy Edit tool supports automated creation of instances of classes in a given ontology by gathering information from more than one class into a single data-entry form. In this way, it is possible to define data-entry forms for new mapping functions that use Prior mapping instances. Figure 9 illustrates how the Easy Edit tool was used to generate an entry form for combining Classification Hierarchy mapping with Temporal Abstraction mapping.

Other related researches in the field of schema matching may help in the process of identifying EMR fields to be used by the mapping ontology. For example, Mork and Bernstein [50] developed a match algorithm to align two ontologies, from the domain of human anatomy, using a structural, hierarchical and lexical matching algorithm that solves also the synonymy problem using the UMLS Metathesaurus [53]. We believe that integrating such a tool that would aid in matching CIG terms to EMR terms with KDOM that enables specifying mapping

relationships, should simplify the whole process of matching CIG data items to EMR fields. Another tool that could facilitate searching for local terms that would match the CIG concepts (specified using standard vocabulary codes) is a *vocabulary service* [12], used by the IDAN framework [10], that serves as a search engine for a set of distributed web-based standard medical vocabulary servers, including: ICD-9-CM and SNOMED-CT for diagnoses codes; CPT for procedure codes; LOINC for laboratory tests and physical signs,, and National Drug File (NDF) for drug codes. The vocabulary search uses keywords and the vocabulary's internal structure (e.g., searching in LOINC vocabulary for the keyword "hemoglobin" in the "whole blood" system) [10].

This work is not intended to provide a *full package* of solutions and tools for all the challenges presented in order to become operational in the real world. We wanted to present a broad *infrastructure* and to evaluate it with real EMRs, CIGs, and CIG formalisms. Therefore, we carried out some tasks manually. For example, we did not develop a tool that aids in mapping to terminologies by taking a concept and its children from a controlled vocabulary and automatically instantiating a hierarchy of concepts to be used by our mapping instances. Such a tool can be developed for the different well-known vocabulary systems. In addition, we did not present a solution for the unit conversion problem because a solution for this problem already exists (presented in Section 1) [12]. Furthermore, the ontology is intended to provide the basic mapping patterns and the ability to compose them into more complex patterns when such needed. For example, after the evaluation that we performed, we identified that the rather simple combination pattern that is composed from Classification Hierarchy Mapping class and Temporal Abstraction Mapping class was found in more than 5% of all the needed mapping functions. Based on this datum, we decided to define an Easy Edit data-entry form that can automate the building of all the needed instances for such pattern (shown in Figure 9). Future work should focus on defining other medical patterns that could be added to the ones developed

in this work, and linking existing solutions (a) to aid in locating terms that match a given CIG concept [50]; and (b) to support functionality that is clearly needed for the ***expressiveness*** of mappings and is not currently supported by KDOM, including unit conversion [12] and transformation of values, such as converting a string to an integer or mapping only part of the EMR field to a CIG concept [18], as well as support of complex temporal abstractions [10].

Conclusion

We have shown that the KDOM framework can be successfully used to map guidelines encoded in two types of guideline formalisms to different EMRs. The mapping enables defining abstract terms and bridging the gap between abstractions used in a guideline encoding and those used in an EMR. After the mappings are defined as instances of KDOM's mapping ontology, SQL queries are automatically generated and enable data retrieval from the target EMR to the source guideline encoding. While we experimented with mapping between one kind of data source (EMRs) and two kinds of knowledge sources (GLIF3 and SAGE CIGs), the KDOM framework was designed in order to enable mapping any data and knowledge sources.

As stated in the Discussion section, our work presents an infrastructure that can be extended by (1) integrating functionalities that would aid in data-knowledge conversion, such as terminology and unit-conversion tools and (2) augmenting the mapping ontology with additional mapping classes.

Acknowledgements

The authors are grateful to Samson Tu and Dongwen Wang for their very helpful remarks. The SAGE project has generously made available their immunization guideline knowledge-base for our study.

References

1. Woolf SH, Grol R, Hutchinson A, Eccles M, Grimshaw J. Potential benefits, limitations, and harms of clinical guidelines. BMJ 1999;318(7182):527-530.
2. Shea S, DuMouchel W, Bahamonde L. A Meta-analysis of 16 Randomized Controlled Trials to Evaluate Computer-based Clinical Reminder Systems for Preventative Care in the Ambulatory Setting. J Am Med Inform Assoc 1996;3(6):399-409.
3. 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.
4. Boxwala AA, Peleg M, Tu S, Ogunyemi O, Zeng Q, Wang D, et al. GLIF3: a representation format for sharable computer-interpretable clinical practice guidelines. Journal of Biomedical Informatics 2004;37(3):147-61.
5. 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.
6. Wang D, Peleg M, Tu SW, Boxwala AA, Ogunyemi O, Zeng Q, et al. Design and implementation of the GLIF3 guideline execution engine. Journal of Biomedical Informatics 2004;37(5):305-18.
7. Clercq PAd, Hasman A, Blom JA, Korsten HH. Design and implementation of a framework to support the development of clinical guidelines. Int J Med Inform 2001;64(2-3):285-318.
8. Bottrighi A. Extending computer guideline system with advanced AI and DB facilities. Advisor: P. Terenziani.: UNIVERSITA' DEGLI STUDI DI TORINO; 2006.
9. Young O, Shahar Y. The Spock System: Developing a Runtime Application Engine for Hybrid-Asbru Guidelines. In: AIME; 2005; 2005. p. 166-170.
10. Boaz D, Shahar Y. A framework for distributed mediation of temporal-abstraction queries to clinical databases. Artificial Intelligence in Medicine 2005;34(1):3-24.
11. Karnieli E, Fodor A, Kulter A, Openhimer Y, Snir D, Fogelman Y, et al. Telemedicine in Human Diabetic Foot as a Potential Model for Improving Medical Care and Reducing Costs: Pros and Cons. In: The Fifth Galil Symposium; 2005; 2005.
12. Shachar EG. Linking Medical Decision Support Systems to Clinical Databases: Ben-Gurion university of the negev. Advised by Yuval Shahar; 2006.
13. Parker C, Rocha R, Campbell JR, Tu SW, Huff SM. Detailed Clinical Models for Sharable, Executable Guidelines. In: Stanford Medical Informatics report # SMI-2003-0972; 2003
14. Sujanski W. A Formal Model for Bridging Heterogeneous Relational Databses in Clinical Medicine: PhD Thesis. Stanford University; 1996.
15. Shahar Y. A framework for knowledge-based temporal abstraction. Artificial Intelligence in Medicine 1997;90(1-2):79-133.
16. Shahar Y, Musen MA. Knowledge-based temporal abstraction in clinical domains. Artificial Intelligence in Medicine 1996;8(3):267-298.
17. Das AK. Temporal Mediation of Relational Databases for Clinical Decision Support: PhD Thesis. Stanford University; 2002.

18. Crubezy M, Pincus Z, Musen MA. Mediating Knowledge between Application Components. In: Proceedings of the Semantic Integration Workshop of the Second International Semantic Web Conference (ISWC-03); 2003; Sanibel Island, Florida; 2003.
19. Sujanski W, Altman R. An Evaluation of the TransFER Model for Sharing Clinical Decision-Support Applications. In: AMIA Annual Symposium; 1996; 1996. p. 468-472.
20. Das AK, Musen MA. A Formal Method to Resolve Temporal Mismatches in Clinical Databases. In: AMIA Annual Symposium; 2001; 2001. p. 130-134.
21. Correndo G, Terenziani P. Towards a flexible integration of clinical guideline systems with medical ontologies and medical information systems. Stud Health Technol Inform 2004;101:108-12.
22. Sujansky W. Heterogeneous Database Integration in Biomedicine. Journal of Biomedical Informatics 2001;34(4):285-298.
23. Boxwala AA, Tu S, Zeng Q, Peleg M, Ogunyemi O, Greenes RA, et al. Towards a representation format for sharable clinical guidelines. Journal of Biomedical Informatics 2001;34:157-169.
24. Noy NF, McGuinness DL. Ontology Development 101: A Guide to Creating Your First Ontology: tanford Knowledge Systems Laboratory Technical Report KSL-01-05 and Stanford Medical Informatics Technical Report SMI-2001-0880; 2001.
25. Health Level 7. HL7 Reference Information Model, version 1.0. In; http://www.HL7.org/library/data-model/RIM/modelpage_non.htm2001.
26. Schadow G, Russler DC, Mead CN, McDonald CJ. Integrating Medical Information and Knowledge in the HL7 RIM. In: Proc AMIA Symposium; 2000; 2000. p. 764-768.
27. Ramakrishnan R, Gehrke J. Database management systems, Second Edition edition: McGraw-Hill; 2000.
28. Fryberg RG, Armstrong D, Giurini J, Edwards A, Kravette M, Kravitz S, et al. American College of Foot and Ankle Surgeons: Diabetic foot disorders: a clinical practice guideline. The Journal of foot and ankle surgery 2000;39(5 Suppl):S1-60.
29. Gennari J, Musen MA, Fergerson RW, Grosso WE, Crubezy M, Eriksson H, et al. The Evolution of Protege: An Environment for Knowledge-Based Systems Development. International Journal of Human-Computer Interaction 2002;58(1):89-123.
30. Peleg M, Wang D, Fodor A, Keren S, Karnieli E. Adaptation of Practice Guidelines for Clinical Decision Support: A Case Study of Diabetic Foot Care. In: Workshop on AI techniques in healthcare: evidence-based guidelines and protocols In conjunction with ECAI; 2006; Riva del Garda, Italy; 2006.
31. National Institute of Health. The Sixth Report of the Joint National Committee on Prevention, Detection, Evaluation, and Treatment of High Blood Pressure: National Institute of Health; 1997 November. Report No.: 98-4080.
32. Helfand M, Redfern C, Sox H. Screening for Thyroid Disease. Annals of Internal Medicine 1998;129(2):141-3.
33. Centers for Disease Control and Prevention. Prevention and control of influenza: Recommendations of the Advisory Committee on Immunization Practices. Morbidity and Mortality Weekly Report 2000;49(RR03):1-38.
34. Bartlett JG, Dowell SF, Mandell LA, File TM, Musher DM, Fine MJ. Practice Guidelines for the Management of Community-Acquired Pneumonia in Adults. Clinical Infectious Diseases 2000;31:347-382.

35. Snow V, Weiss K, Wall EM, Mottur-Pilson C. Pharmacologic management of acute attacks of migraine and prevention of migraine headache. American Academy of Family Physicians - Medical Specialty Society American College of Physicians - Medical Specialty Society. Ann Intern Med 2001;137(10):840-852.

36. Irwin RS, Boulet LS, Cloutier MM, Gold PM, Ing AJ, O'byrne P, 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):133S-181S.

37. Institute for Clinical Systems Improvement. Immunization guideline. http://www.icsi.org/knowledge/detail.asp?catID=29&itemIIn.

38. Borten M. Breast Mass. In: Gynecological Decision Making, 2nd Edition edition. St Louis: Mosby-Year Book, Inc; 1998.

39. Gibbons RJ, Abrams J, Chatterjee K, Daley J, Deedwania PC, Douglas JS, et al. Guidelines for the management of patients with chronic stable angina. The American Journal of Cardiology 1999;33(7):2092-2197.

40. Los Angeles Alzheimer's Association. Guidelines for Alzheimer's disease management; 1999 Jan 8.

41. 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.

42. Yeni PG, Hammer SM, Carpenter CCJ, Cooper DA, Fischl MA, Gatell JM, et al. Antiretroviral treatment for adult HIV infection in 2002: updated recommendations of the International AIDS Society-USA Panel. Journal of the American Medical Association 2002;288(2):222-235.

43. Bigos SJ, Bowyer ROR, Braen GR, Brown K, Deyo R, Haldeman S, et al. Acute low back problems in adults. Clinical Practice Guideline no. 14. Department of Health and Human Services; AHCPR publication no. 95-0642. Rockville, MD; 1994.

44. Snow V, Lascher S, Mottur-Pilson C. Pharmacologic treatment of acute major depression and dysthymia. Annals of Internal Medicine 2000;139(9):738-742.

45. Borkan J, Werner S, Porat A, Ribak Y, Reis S. Lower Back Pain: Clinical Practice Guideline Clalit Health Services; 1995.

46. Spencer T, Shalom J, Frenkel M, Meiron D, Presiko M, Zohar S, et al. Acute Otitis Media - Guidelines for Diagnosis and Treatment. 2nd Edition edn. Clalit Health Services; 2003.

47. Tu S, Campbell J, Musen MA. The SAGE guideline modeling: motivation and methodology. Stud Health Technol Inform 2004;101:167-71.

48. Shahar Y, Musen MA. ReSUMe: A temporal-abstraction system for patient monitoring. Computers and Biomedical Research 1993;26(3): 255-273.

49. Shalom E, Shahar Y, Lunenfeld E, Taieb-Maimon M, Young O, Goren-Bar D, et al. The Importance of Creating an Ontology-Specific Consensus Before a Markup-Based Specification of Clinical Guidelines. In: Proceeding of the biennial European Conference on Artificial Intelligence (ECAI); 2006; Riva del Garda, Italy; 2006.

50. Mork P, Bernstein PA. Adapting a Generic Match Algorithm to Align Ontologies of Human Anatomy. In: International Conference on Data Engineering; 2004; Boston; 2004. p. 787-790.

51. SNOMED International. SNOMED Clinical Terms. In; http://www.snomed.org/snomedct/index.html2006.

52. Peleg M, Shmerlin B, Izvecov P, Ivanenko V. Easy Edit Tab for rapid data entry. In: http://mis.hevra.haifa.ac.il/~morpeleg/EasyEdit/University of Haifa; 2005.

53. Lindberg C. The Unified Medical Language System (UMLS) of the National Library of Medicine. J Am Med Rec Assoc 1990;61(5):40-42.

Table and Figure Legends

Table 1. Challenges and the types of solutions that were addressed by related works

Table 2a. The types of mapping instances used in the first five evaluated guidelines.

Table 2b. The types of mapping instances used in the second six evaluated guidelines.

Table 2c. The types of mapping instances used in the third five evaluated guidelines.

Figure 1. Linking a CIG to different EMRs: (a) using database views in a RIM standard and (b) using a direct link to each of the different EMRs.

Figure 2. The main mapping classes.

Figure 3. An example of using an instance of Temporal Abstraction Mapping class to retrieve the first visit of a patient during the year of 2004. We set six properties in order to define it: (1) The destination field property was set with an instance of Destination Field Information that holds the field name of the EMR that records the patients' visit time, (2) the Mapping Result's result field property was set with the same instance like the destination field property so it is used both for locating the tuple containing the first visit (as specified in (1)) and for returning the recorded time (as specified in (2)), (3) the temporal abstraction operator property was set with an instance of the Temporal Abstraction Operator class indicated that the first (the *Min* operator in the SQL query below) value that holds the condition should be returned, (4a) the patient ID and the patient EMR specific field (4b) were filled with a patient identification number and with EMR field that holds the patient key respectively to select the desired patient to which the data

should be retrieved, and (5) the within property was set with an instance of the Time Range class for bounding the selection to year of 2004.

Figure 4. A hierarchy of pharmacological substance used in Hierarchical Mapping. The hierarchy contains 4 nodes with different levels of generality; the pharmacological substance is the most general and the Amoxycillin and Penicillin-V are the most specific. By selecting the antibiotics term from the hierarchy, all the terms beneath it (i.e. Amoxycillin and Penicillin-V) will be considered as part of the original selection).

Figure 5. Example of using an instance of the Classification Hierarchy Mapping class to find out when a patient has taken antibiotics. (1)The antibiotics medical term that we would like to look for (in the medical hierarchy) was set as the medical term property and the EMR (2) destination field that holds this information was set as Drug.Drug.name (instance of Destination Field Information class). This example also illustrates how one data item can be retrieved from the EMR based on other data items (e.g., the *time* field is returned but the EMR destination field (as specified by the MappingResultType slot) that is compared to the medical hierarchy is the *name* field (as specified by the DestinationFiledInfo slot). To implement this, (3) the "mapping result" property was set to an instance of Field Result class called "Drug.Time". This instance indicates that the desired result of the mapping will be an EMR field value with the desired returned field (Drug Time), which was set as the "result field" property using an instance of Destination Field Information class also "drug time field". This instance contains the information of the EMR field that holds the time value at which the event of taking a drug has been recorded.

Figure 6. An example of using an instance of the Binary Logical Mapping class to define an AND logical mapping for a data item of the CIG called *Charcot*. The mapping is composed of three different prior mappings: malaligment, erythema and swelling. The logical condition among the three prior mapping functions is an AND logical condition so only if all the prior mapping functions hold the condition then the returning result will be a true value (as specified by the MappingResultType slot). The right part of the figure shows the details of the Prior Mapping used to define Swelling

Figure 7. The steps needed in order to encode one guideline and to link it to one EMR. The arrows that link different steps indicate dependencies on information found in a step. The ordering of steps may vary. a..g indicate one possible ordering of the steps

Figure 8. Integrating the KDOM framework into the GLEE execution process. Data-item specifications in GLIF CIGs contain references to mapping ontology instances. The mapping ontology instances are sent to KDOM's SQL generator, which generates SQL queries to be later executed by GLEE in order to retrieve patient information from an EMR.

Figure 9. Using the Easy Edit tool to generate an entry form for combining Classification Hierarchy mapping with Temporal Abstraction mapping – to support queries such as "is the current given drug an antibiotic?". In this way, information is gathered from more than one class via a single data-entry form. The data entry form shown in the figure was constructed for the outer mapping class of the combined classification-Temporal mapping (the Classification Hierarchy Mapping class). The first four values of the Easy Edit line (i.e., 'Prior Example',

'Drugs', '23487' and 'Value') define the instance of the Prior Mapping class (shown on the top left) that the Classification Hierarchy instance refers to. The next two values (i.e., 'DrugInfo' and 'FieldValue') define the instance of the Result Type class (shown on the bottom), which is referenced by the Prior Mapping class. The last three values (i.e., 'MedicationName', 'Medication' and 'Name') define an instance of Destination Field Information class (shown on the top right) which is referenced by the instance of the Result Type class. For simplicity, the definition of the Mapping Function field and Patient EMR Specific field in the Prior Mapping instance are not presented, but they are also part of the Easy Edit line and are used to define the Temporal Abstraction Mapping as a Prior mapping function and the table containing the patient ID, respectively.

		Sujansky and Altman [14, 19]	Boaz and Shahar [10], Shachar and Shahar [12]	Das and Musen [20]	Crubézy et al [18]	Correndo and Terenziani [21]	KDOM
Mapping model (properties and components)	Query language	+	+	+	-	+	+
	Formal mapping model	Algebra	XML File	Temporal version of SQL	Frames ontology	XML file	Frames ontology
	reference model	+ (global)	HL7-RIM	-	-	HL7-RIM	HL7-RIM
	Type of connecting model	High-level data-modeling constructs	Database views	Temporal mediator	-	-	Database views
Mapping realization in tool	Defining mappings	Extended relational algebra mapping language	Schema mapping tool	Query Interface	Instances of the ontology	Creating an XML file	Instances of the sontology
	Automated SQL queries generator	+	+	+	Mapping interpreter	+	+
Mapping capabilities	Unit conversion	-	+	time units	-[1]	-	-
	Classification Hierarchy	-	+	-	-	-	+
	Definitions of Abstract terms	-	-	-	-	-	+
	Temporal abstractions	-	+	+ (broad)	-	-	+ (basic)
	Enable composition of mapping functions	-	-	-	+	-	+
	Extensibility: ability to define new mapping types and integrate with existing framework	-	-	-	-	-	+
	The ability to define Mapping relations	-	-	-	+	+	+

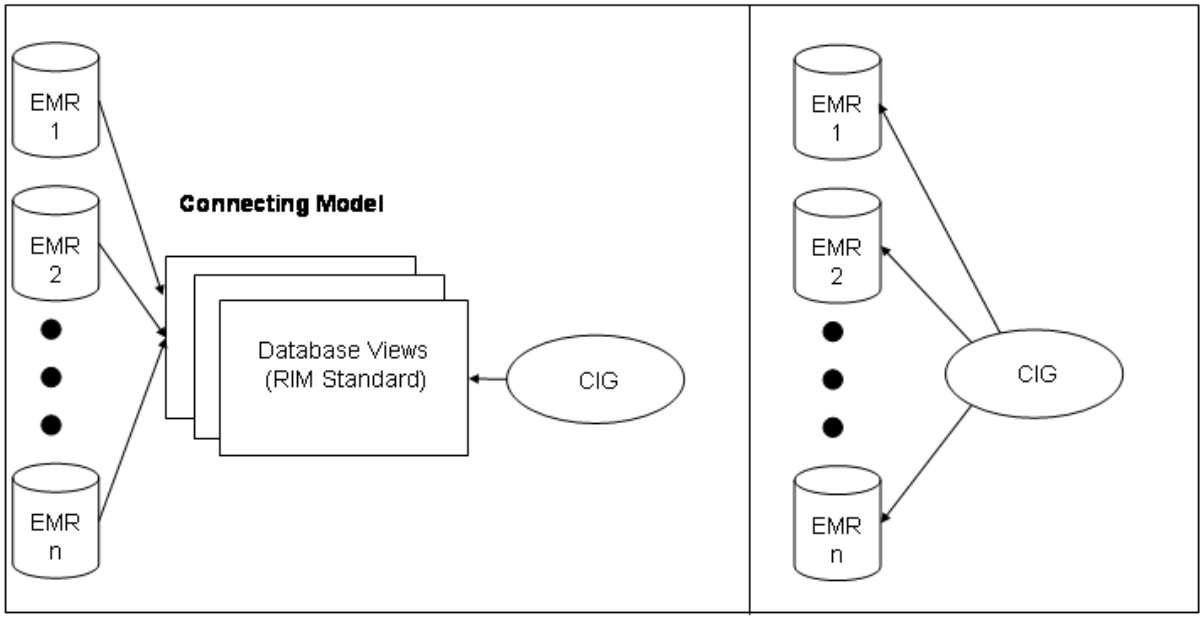
[1] Enabled via the language and operators that you can apply when you map an instance, but no service that has knowledge of units and their equivalents is included

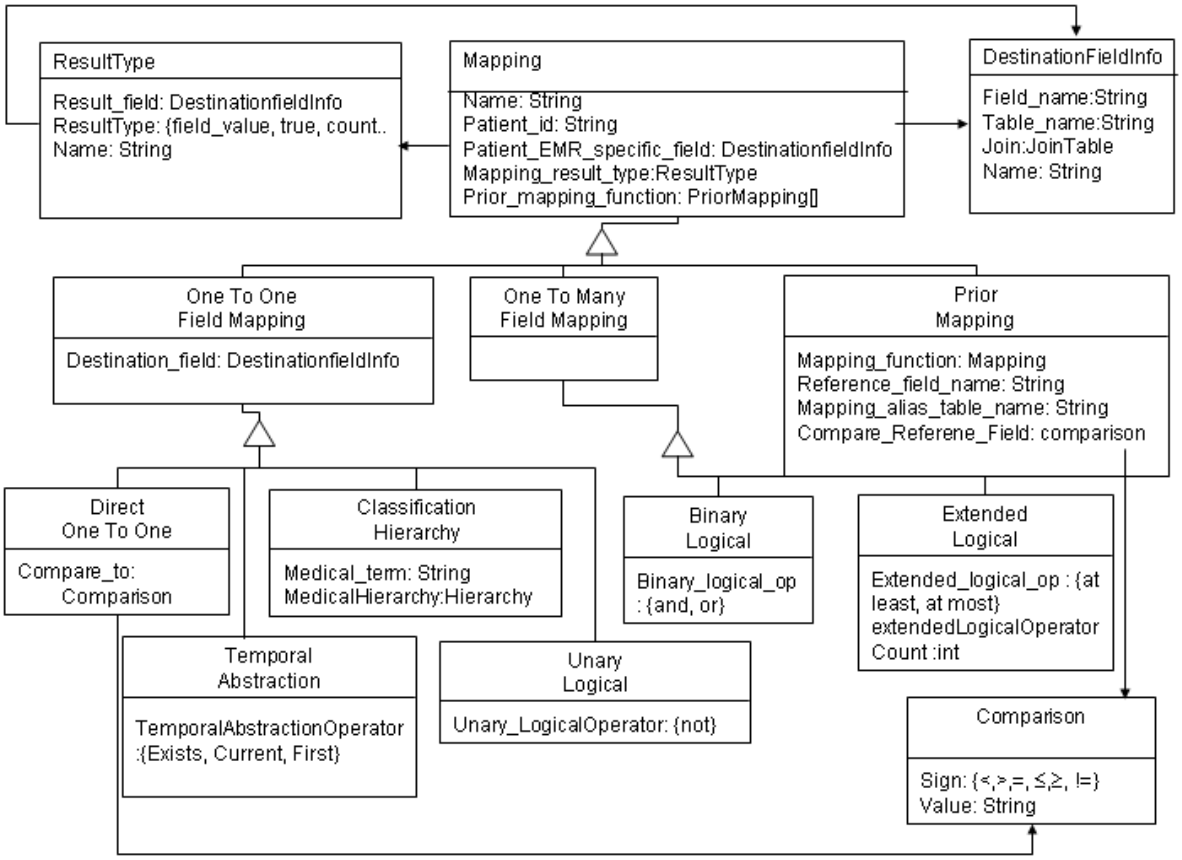
Guidelines / Mapping Instances	Hypertension [31]	Thyroid [32]	Cough [34]	Otitis Media [46]	Stable Angina [39]
BinaryLogicalMapping	10	5	0	2	12
BinaryLogicalMapping + TemporalAbstractionMapping	0	3	6	4	0
TemporalAbstractionMapping	0	1	7	1	3
ClassificationHierarchyMapping	2	1	0	4	0
BinaryLogicalMapping + ClassificationHierarchyMapping	2	0	2	0	1
DirectOneToOneMapping	8	0	0	5	63
ExtendedLogicalMapping	0	0	0	0	2
ExtendedLogicalMapping + ClassificationHierarchyMapping	0	0	0	0	0
ExtendedLogicalMapping + TemporalAbstractionMapping	0	0	0	0	0
TemporalAbstractionMapping + ClassificationHierarchyMapping	21	0	0	0	0
TemporalAbstractionMapping + ClassificationHierarchyMapping + BinaryLogicalMapping	0	0	0	0	0
Unsuccessful mapping	**0**	**0**	**0**	**0**	**0**
Total mappings (=total #data items)	43	10	15	16	81
Sum (composed mappings)	23	3	8	4	1
Sum(classificationHierarchy mappings)	25	1	2	4	1

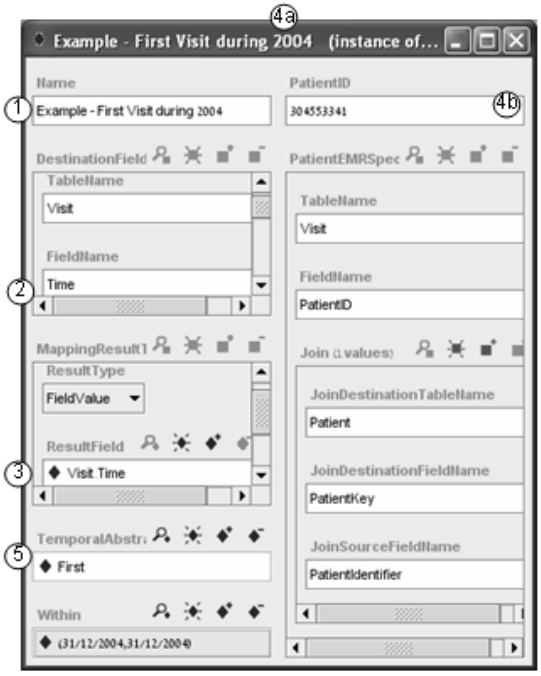
Guidelines / Mapping Instances	Extubation [41]	Depression [44]	Flu [33]	Immunization [37]	Lower Back Pain [45]	Acute Low Back Problems [43]
BinaryLogicalMapping	0	1	6	13	3	0
BinaryLogicalMapping + TemporalAbstractionMapping	3	1	7	4	0	2
TemporalAbstractionMapping	0	1	0	0	0	6
ClassificationHierarchyMapping	0	0	6	10	4	0
BinaryLogicalMapping + ClassificationHierarchyMapping	0	0	1	24	1	0
DirectOneToOneMapping	0	2	8	2	4	1
ExtendedLogicalMapping	0	0	5	0	1	1
ExtendedLogicalMapping + ClassificationHierarchyMapping	0	0	0	0	0	0
ExtendedLogicalMapping + TemporalAbstractionMapping	0	0	2	0	0	0
TemporalAbstractionMapping + ClassificationHierarchyMapping	0	0	0	1	0	0
TemporalAbstractionMapping + ClassificationHierarchyMapping + BinaryLogicalMapping	0	0	5	83	0	0
Unsuccessful mapping	**0**	**0**	**0**	**0**	**0**	**0**
Total mappings (=data items)	3	5	40	137	13	10
Sum (composed mappings)	3	1	15	112	1	2
Sum(classificationHierarmapps)	0	0	12	118	5	0

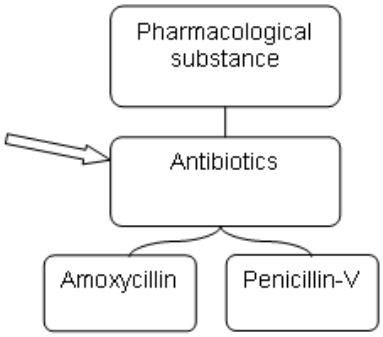
Guidelines / Mapping Instances	Breast Mass[a] [38]	Pneumonia[a] [34]	Alzheimer[a] [40]	Headache[a] [35]	AIDS[a] [42]
BinaryLogicalMapping	5	0	0	0	3
BinaryLogicalMapping + TemporalAbstractionMapping	1	0	0	0	0
TemporalAbstractionMapping	0	0	0	2	0
ClassificationHierarchyMapping	0	33	12	0	14
BinaryLogicalMapping + ClassificationHierarchyMapping	0	3	6	3	14
DirectOneToOneMapping	13	1	0	3	7
ExtendedLogicalMapping	0	0	0	0	3
ExtendedLogicalMapping + ClassificationHierarchyMapping	0	0	0	2	0
ExtendedLogicalMapping + TemporalAbstractionMapping	0	0	0	0	0
TemporalAbstractionMapping + ClassificationHierarchyMapping	0	0	0	0	4
TemporalAbstractionMapping + ClassificationHierarchyMapping + BinaryLogicalMapping	0	0	1	0	10
Unsuccessful mapping	**0**	**0**	**0**	**0**	**0**
Total mappings (=total #data items)	19	37	19	10	55
Total mappings (=total #data items)	1	3	7	5	28
Sum(classificationHierarchy mapps)	0	36	19	5	42

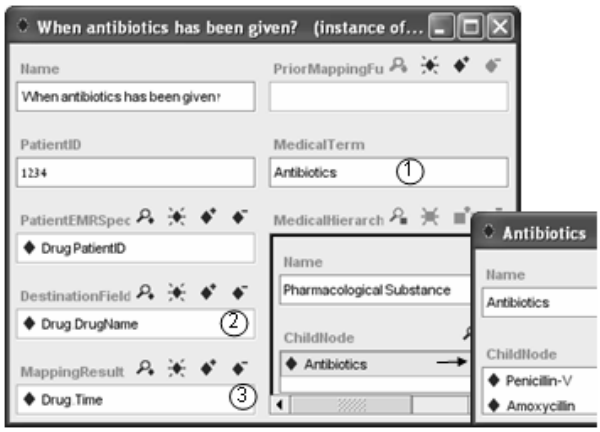
[a] The original encoded guideline did not contain explicitly defined data items. We created the mapping functions based on the decision criteria and not based on existing data items.

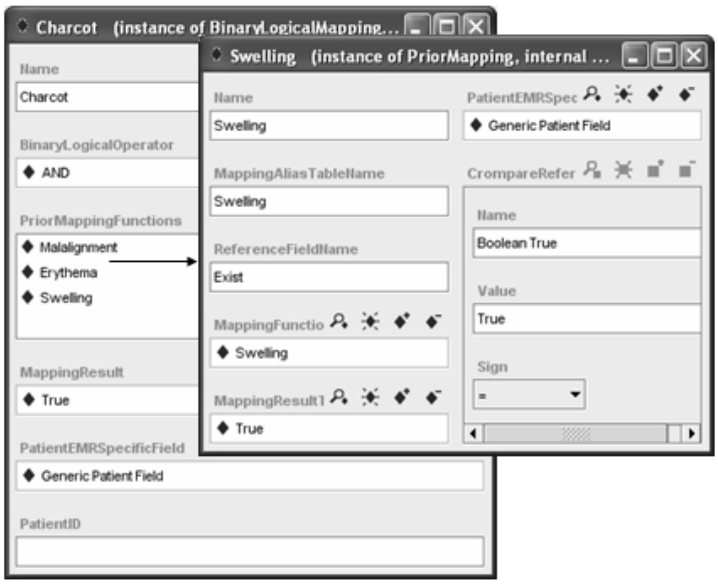

EMR
1
EMR
2
EMR
n
Connecting Model
Database Views
(RIM Standard)
CIG
EMR
1
EMR
2
EMR
n
CIG

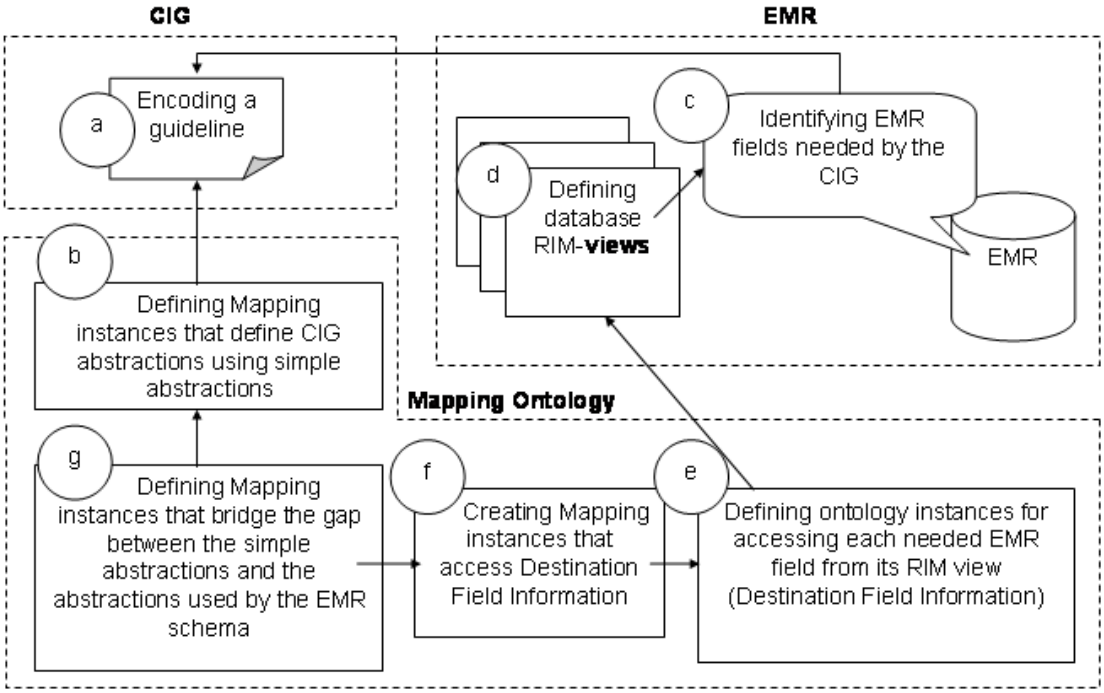

ResultType
Result_field: DestinationfieldInfo
ResultType: {field_value, true, count..
Name: String
Mapping
Name: String
Patient_id: String
Patient_EMR_specific_field: DestinationfieldInfo
Mapping_result_type:ResultType
Prior_mapping_function: PriorMapping[]
DestinationFieldInfo
Field_name:String
Table_name:String
Join:JoinTable
Name: String
One To One Field Mapping
Destination_field: DestinationfieldInfo
One To Many Field Mapping
Prior Mapping
Mapping_function: Mapping
Reference_field_name: String
Mapping_alias_table_name: String
Compare_Referene_Field: comparison
Direct One To One
Compare_to: Comparison
Classification Hierarchy
Medical_term: String
MedicalHierarchy:Hierarchy
Binary Logical
Binary_logical_op : {and, or}
Extended Logical
Extended_logical_op : {at least, at most}
extendedLogicalOperator
Count :int
Temporal Abstraction
TemporalAbstractionOperator :{Exists, Current, First}
Unary Logical
Unary_LogicalOperator: {not}
Comparison
Sign: {<,>,=, ≤,≥, !=}
Value: String

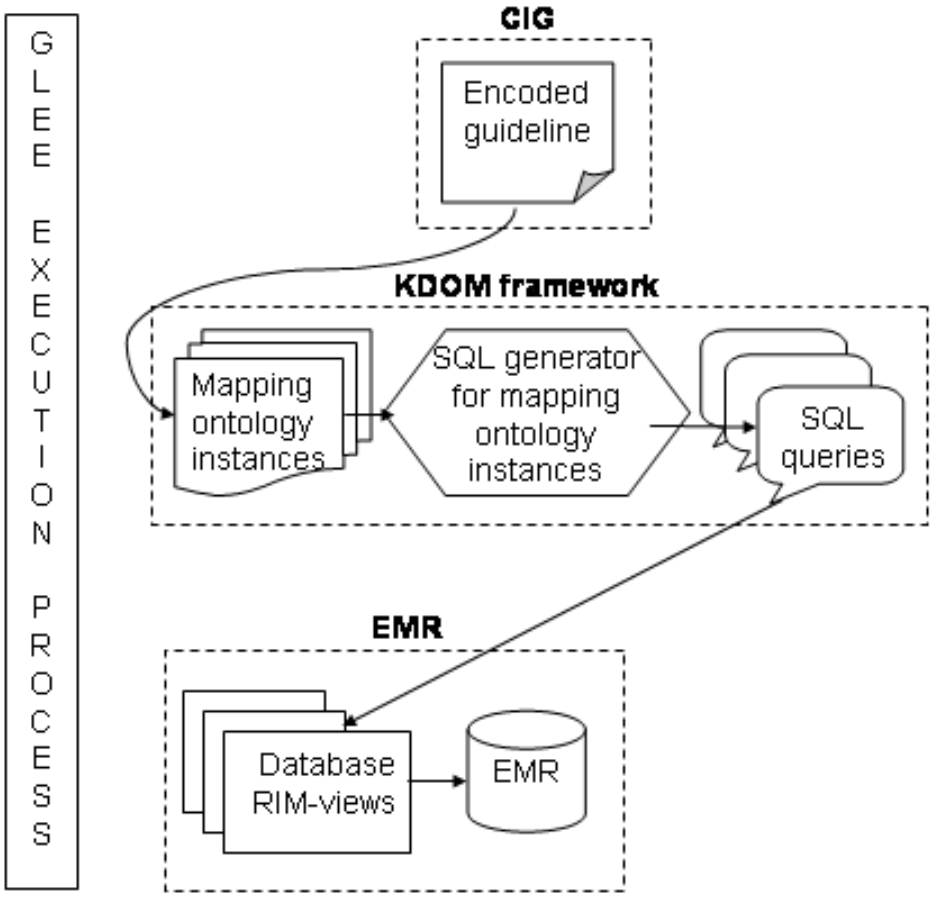

4a
Example - First Visit during 2004 (instance of...
Name
Example - First Visit during 2004
PatientID
304553341
4b
1
DestinationField
TableName
Visit
FieldName
Time
2
PatientEMRSpec
TableName
Visit
FieldName
PatientID
MappingResultT
ResultType
FieldValue
ResultField
Visit.Time
3
Join (1 values)
JoinDestinationTableName
Patient
JoinDestinationFieldName
PatientKey
JoinSourceFieldName
PatientIdentifier
TemporalAbstr
First
5
Within
(31/12/2004,31/12/2004)

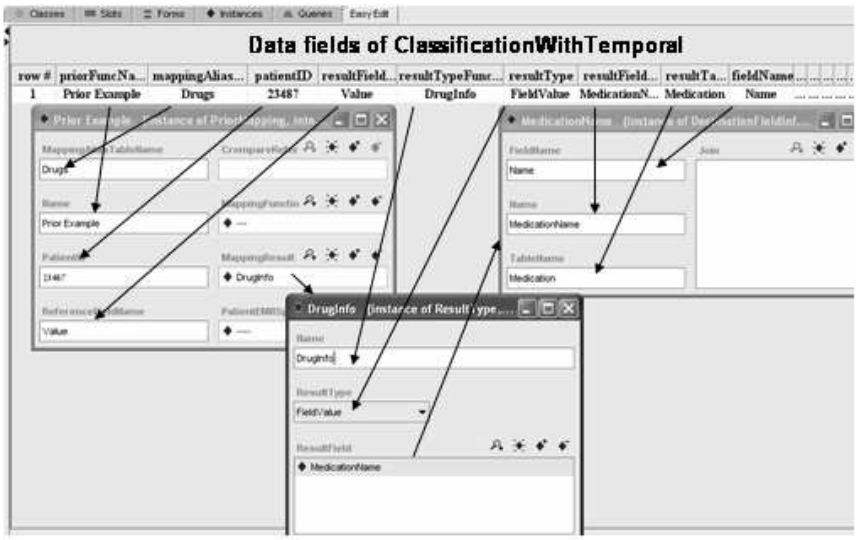

Pharmacological substance
Antibiotics
Amoxycillin
Penicillin-V


When antibiotics has been given? (instance of...
Name
When antibiotics has been given
PriorMappingFu
PatientID
1234
MedicalTerm
Antibiotics
1
PatientEMRSpec
Drug.PatientID
MedicalHierarch
DestinationField
Drug.DrugName
2
Name
Pharmacological Substance
ChildNode
Antibiotics
MappingResult
Drug.Time
3
Antibiotics
Name
Antibiotics
ChildNode
Penicillin-V
Amoxycillin


Charcot (instance of BinaryLogicalMapping...
Name
Charcot
BinaryLogicalOperator
AND
PriorMappingFunctions
Malalignment
Erythema
Swelling
MappingResult
True
PatientEMRSpecificField
Generic Patient Field
PatientID
Swelling (instance of PriorMapping, internal ...
Name
Swelling
MappingAliasTableName
Swelling
ReferenceFieldName
Exist
MappingFunctio
Swelling
MappingResultT
True
PatientEMRSpec
Generic Patient Field
CrompareRefer
Name
Boolean True
Value
True
Sign
=


CIG
EMR
a
Encoding a guideline
c
Identifying EMR fields needed by the CIG
d
Defining database RIM-views
EMR
b
Defining Mapping instances that define CIG abstractions using simple abstractions
Mapping Ontology
g
Defining Mapping instances that bridge the gap between the simple abstractions and the abstractions used by the EMR schema
f
Creating Mapping instances that access Destination Field Information
e
Defining ontology instances for accessing each needed EMR field from its RIM view (Destination Field Information)


GLEE EXECUTION PROCESS
CIG
Encoded guideline
KDOM framework
Mapping ontology instances
SQL generator for mapping ontology instances
SQL queries
EMR
Database RIM-views
EMR


Data fields of ClassificationWithTemporal
row #
priorFuncNa...
mappingAlias...
patientID
resultField...
resultTypeFunc...
resultType
resultField...
resultTa...
fieldName...
1
Prior Example
Drugs
23487
Value
DrugInfo
FieldValue
MedicationN...
Medication
Name
Prior Example
Drugs
23487
Value
DrugInfo
Name
MedicationName
Medication
DrugInfo
FieldValue
MedicationName