MEDINFO 2007
HISA Ltd Australia
© 2007 The authors. All rights reserved.

Managing Differently Structured Data Using the Clinical Element Model

Craig G. Parker[a], Joseph F. Coyle[a,b], Yan Z. Heras[a,b], Thomas A. Oniki[a], Stanley M. Huff[a,b]

[a]*Intermountain Healthcare, Salt Lake City, Utah, USA*
[b]*Department of Medical Informatics, University of Utah, Salt Lake City, Utah, USA*

Abstract

One goal of many clinical information systems is to capture as much data as possible in structured form. This allows significant flexibility in using the data after it has been captured. While progress data standardization efforts are making this goal more achievable, the realities of clinical information present constant challenges. We present some of the challenges that we have encountered as we create clinical data representations using the Clinical Element Model. The majority of these challenges are due to either differences in information granularity or incompatible data types. We have devised strategies for dealing with these challenges. These strategies revolve around the use of two constructs within the Clinical Element Model. The Aggregate construct is used to address issues related to information granularity differences. The Alternative construct is designed to assist in situations where data types are not consistent.

Keywords:

clinical data models; information storage and retrieval

Introduction

Providing optimal care for patients in a modern healthcare setting requires the integration of many different types of information from many different sources. Assimilating large quantities of information and determining the role this information should play in caring for a patient is a demanding task for clinicians and those who support them [1]. Clinical information systems are designed to assist in this task by helping to capture and organize much of this information. When properly implemented, the organization of information in a clinical information system enables presentation and review of the stored information in a temporal and spatial arrangement that enhances a clinician's ability to provide appropriate care in a timely fashion. In addition to presenting back to the user useful arrangements of stored information, clinical information systems may also perform higher-level functions on the information such as making inferences about the state of a patient's health or recommending tasks for managing a disease.

Regurgitation of previously stored information in useful arrangements does not generally require much structurally from the information being handled. However, as a clinical information system strives to provide higher-level functionality, the structure of the underlying data becomes more important. Although natural language processing techniques can be applied to assist in the effort, it is generally more difficult to generate meaningful and accurate alerts on unstructured, textual documents than on well-structured information.

Clinical elements

We have previously described the Clinical Element Model [2], the data modeling formalism that we employ for structuring our clinical information. The Clinical Element Model defines a highly flexible and recursive data model. The foundational unit of the model is called a Clinical Element. A Clinical Element can primarily either (1) be assigned a data type and contain one piece of information conforming to that type, or (2) contain a group of other Clinical Elements. For example, in modeling vital signs, we would define a Clinical Element for respiratory rate with a data type of "physical quantity", likely named "RespiratoryRate". We would similarly define Clinical Elements for other components of a vital signs assessment such as heart rate, blood pressure, and temperature. We would then define a Clinical Element that would contain the components of the vital signs assessment and name it something like "VitalSignsPanel". Thus, the "VitalSignsPanel" Clinical Element would contain, amongst others, the "RespiratoryRate" Clinical Element. Clinical Elements can thus be represented as a simple tree structure with all internal nodes containing other Clinical Elements, and all leaf nodes containing data of an assigned type.

The data types that we use in the Clinical Element Model are derived from the data types that Health Level Seven (HL7) has defined for their Reference Information Model (RIM) [3]. They include types such as "time stamp", "coded concept", "physical quantity", "binary", and "string". While we have made a few changes to the HL7 versions of these types to support our needs, we strive to maintain a high level of compatibility to support semantic interoperability with other systems via HL7 messaging.

Within the Clinical Element Model, in addition to the primary type of data that a Clinical Element contains, every Clinical Element may also contain a set of qualifiers. Qualifiers are data items that provide additional informa-

tion relevant to the primary data. For example, the "RespiratoryRate" Clinical Element may contain a qualifier for the time at which the respiratory rate was measured. Structurally, the data captured in qualifiers is no different from other data in the model, and thus qualifiers are modeled as specializations of Clinical Elements. Therefore every qualifier may either be assigned a data type or may contain a group of other Clinical Elements. Likewise, qualifiers may contain other qualifiers. Although the decision as to whether a piece of data should be modeled as primary data or as a qualifier is not always straightforward, in practice these cases are the exception and do not pose significant problems for this modeling approach.

A fundamental aspect of our data management approach using the Clinical Element model is that we define a single standard representation for each type of clinical information that we use in our system. We then map all incoming information to these standardized structures. In developing the Clinical Element Model, we put significant effort into structuring the information such that it will support many higher-level functions that we deem to be of value. These functions included advanced clinical decision support and semantically interoperable exchange of information with other clinical information systems.

Differently structured data

Despite the effort that we put into modeling and mapping information appropriately, we recognize several situations where real-world concerns present us with information that does not fit neatly into our ideal models. When managing a system that receives clinical information from a large number of external sources, it is nigh impossible to have the power and influence necessary to ensure that all of the information that is received comes in a format that maps well into a standardized structure. Even when an enterprise strives to require high standards of data structure when negotiating with potential data sources, the realities of legacy data, unresponsive vendors, and lack of bargaining power often leave the enterprise with less than optimally structured inbound information.

Furthermore, even when external sources of information are cooperative, enterprises will encounter relevant and accurate clinical information that does not fit well in their internal data structures. The result of these situations is source data that is of different granularity than that of the target model, or that is of a form that is not compatible with the data type in the target. In the Clinical Element Model, we have developed several constructs that allow us to gracefully handle clinical information that does not fit nicely into the standard structures that we have defined.

Methods

Handling granularity issues with aggregate

The first construct we will consider is called Aggregate. As may be inferred from its name, Aggregate is used in situations when the granularity of the source information is not consistent with that of the target model. Specifically, we use aggregate when the inbound information is less granularly structured than the target Clinical Element. In these situations we use a system that we have created for decomposition of composite information into more granular pieces. Description of this decomposition system is beyond the scope of this paper, but the output of this system is information of appropriate granularity for use in the target Clinical Elements.

For example, in our inbound interface for an external microbiology system, we receive information about the body location, from which the relevant specimen was collected, in a single, unstructured, text field. The contents and format of the data in this field vary significantly between messages. In contrast, the model for body location defined in the Clinical Element Model is highly structured, with the ability to independently specify features such as body part, laterality, proximity, superiority, depth, and lesion type.

When the decomposition system is successful, it generates an appropriately populated set of Clinical Element instances containing the information from the source text. We have found it helpful, when capturing and storing these decompositions of clinical information, to store the original composite with each piece. It is useful to have this information available when we need to present back to the user what was originally received. One reason is that although a proper decomposition should contain all of the information content from the original source data, putting all of the pieces back in an order and format that is natural for a human to consume is not a simple task. In addition, we find it useful to have the original source data closely linked when we are verifying the results of our decomposition process.

The Aggregate structure in the Clinical Element Model is the place where we store the composite source information. Aggregate is a foundational part of the model, and therefore is available to all instances of Clinical Elements. Enhancing our description of Clinical Elements above, every Clinical Element will contain a primary data item (either a field of an assigned type, or a group of Clinical Elements), and may also contain qualifiers and an Aggregate. Since all information in the Clinical Element Model is stored as Clinical Elements, any piece of information can have an associated Aggregate. In addition to the textual source data in the microbiology interface use case described above, we have either real or expected use cases

for binary data and coded concepts as source data types. Thus, the Aggregate construct consists of an exclusive choice between a "coded concept" value and a "binary" value. These two data types support all of the used cases that we are aware of for composite data in our enterprise. Following the conventions of the HL7 data types, the "string" data type is a specialization of "binary", and is thus supported by Aggregate.

An example of when we would store binary data in Aggregate is when we receive information from an outside source as a document in a format such as Adobe Systems' Portable Document Format (PDF). If the PDF document contained relevant information that we would benefit from storing in a structured format, we would extract the information, creating and storing the appropriate Clinical Elements. Each of these Clinical Elements would be associated with the original PDF by placing the PDF in the binary value field of Aggregate.

We recognize that, in many cases, storing the original composition with each Clinical Element derived from it may result in large amounts of redundant data being stored and unnecessarily wasting space. To mitigate this condition, we have devised two strategies for referencing a single instance of the source data contained in Aggregate rather than copying it multiple times.

One of the properties of qualifiers in the Clinical Element Model is that they can be inherited by containment. This is similar to "context conduction" in the HL7 RIM. To illustrate how this works, consider two Clinical Elements, A and B, and a qualifier, C. In this example, Clinical Element A contains Clinical Element B (e.g. A may be a chemistry panel and B may be a serum sodium measurement). If the definitions of both A and B reference qualifier C, then we only need to specify C for A, and it automatically applies to B unless specifically overridden. For example, in a chemistry panel, we may define a qualifier for the specimen both at the panel level and within each individual chemistry in the panel. We can then specify the specimen at the level of the panel, and each of the individual chemistries will inherit this specimen unless they explicitly specify a different one. If the composite source data decomposes nicely into a series of Clinical Elements contained within a common parent, this same mechanism can be used with Aggregate to avoid repeating the composite source data multiple times.

Within the Clinical Element Model we have also defined a mechanism for specifying a link or reference from one data instance to another. Using this mechanism, we can store the composite source data as an independent, undivided, Clinical Element. We can then, from within the Aggregate, reference this composite Clinical Element by an indexed unique identifier rather than by inclusion of the source data directly.

Handling type discrepancies with alternative

Another construct within the Clinical Element Model which we use to manage data that doesn't fit well is called Alternative. In contrast to Aggregate, which was developed for managing granularity mismatches, Alternative was designed to handle data type mismatches.

As we develop our clinical models, we evaluate each piece of relevant data, define a Clinical Element for that data, and assign that Clinical Element a type. In most cases, assigning a data type is relatively clear-cut. Most laboratory data with numeric result values is handled nicely with the "physical quantity" data type. Points in time are represented with the "time stamp" type. And so forth. However, even when the assigning of data types appears simple, the realities of clinical data present interesting challenges.

The "physical quantity" data type consists of a numeric value and a coded concept representing the units of measure. Although most of the results for a given laboratory test may fit well in this structure, sometimes we will receive a result such as "below detectable limit". In a situation like this, the value we received does provide an answer to the question posed by the laboratory test, "how much of a given substance is present in the sample", but it does not fit nicely into our "physical quantity" data type.

To handle such cases, we use the Clinical Element Model's Alternative construct. Like Aggregate, Alternative is a foundational part of the model. It is available to any Clinical Element. Also like Aggregate, its structure consists of a choice of several data types. In the case of Alternative, the available data types are "coded concept", "binary" (which includes "string" as mentioned above), and "physical quantity".

Strictly speaking, to be appropriate for use in the Alternative construct, the source data must fulfill two requirements. First, the granularity of the source data must match that of the target Clinical Element. If there is a granularity mismatch, then Aggregate should be used instead of Alternative. Second, the source data, while structurally incompatible with the target data type, should be semantically consistent with the target Clinical Element. Expanding on the laboratory example above, "below detectable limit" answers the question of "how much" in a manner similar to a physical quantity of "2 mg/dL". However, if the value we received were "specimen hemolyzed", indicating that the test could not be performed because of an unfit specimen, this value does not answer the question of "how much" and therefore should be recorded elsewhere, leaving the value in the target Clinical Element null. In practice however, determining the semantics of data received in an alternative format is frequently a difficult task. Therefore information that this

stored in Alternative is not always as semantically congruent with the expected data as we would desire.

The reason for these requirements on the use of Alternative is that we want all of a given kind of information stored, and thus retrievable, in a common manner, even if the data type is not what is expected. When querying for a given laboratory test, we do not want our users to have to remember to query both for those stored in the expected "physical quantity" format as well as for those stored in other formats. Rather, we feel that is important that a query for a given kind of information retrieve all instances of that information regardless of data type. We recognize that this places an additional burden on the applications that receive the information to handle unexpected data types, but ultimately we are more comfortable forcing these issues to be dealt with up front rather than letting this useful information fall between the cracks.

The common scenarios for using the Alternative construct include data at the extremes of possible ranges, and data from unusual, infrequent, or uncontrolled sources. In addition to "below detectable limit", other examples of the use of Alternative include:

- We receive an optically scanned SOAP note from a referring physician when we have internally defined a well-structured model for a SOAP note.
- We receive a quantitative heart rate of "140 bpm" for the APGAR pulse component which is designed to contain one of the coded concepts for "no heart rate", "heart rate less than 100 bpm", or "heart rate at least 100 bpm".

We do not use the Alternative construct when the source data is of the expected data type, but needs to be translated, such as when we receive a concept from a different coding system than that which we use internally or when we receive a physical quantity with units that differ from our enterprise standard. For these cases, the "coded concept" and "physical quantity" data types, respectively, have built in mechanisms for handling translations.

Since the information captured in Alternative should be semantically compatible with the target Clinical Element, that information may sometimes be directly transformable to the target data type. For example, in the case of APGAR scores mentioned above, would could accurately generate a concept of "heart rate at least 100 bpm" from the physical quantity that we received. In situations where this is possible, and doing so would provide enough additional value, we would develop processes – either automated or manual – to populate the primary data field in addition to the Alternative construct, allowing the data to be handled more consistently with other instances of that kind of information.

Discussion

The Aggregate and Alternative constructs provide us with practical mechanisms for handling differently structured data. Some the principles behind this approach include:

- We want our data structured in ways that support robust clinical decision support and semantically interoperable data exchange.
- When transforming data into more useful, structured forms, we want to maintain the original form of the data.
- Even when data is not structured as expected, we do not want any data to be stored using obscure methods or constructs that make it easy to overlook.

The focus or our modeling efforts is largely on the common internal data models needed to support our traditional, "best-of-bread" approach. As described above, this approach, while addressing some concerns, creates its own set of challenges, particularly for applications that use the information stored as Clinical Elements. For applications over which we have influence, we are building in mechanism to support the capture, manipulation, and display of this differently structured data. For other applications, we have to address the difficulties on a case-by-case basis and choose the best solution for each.

Conclusion

The use of the Aggregate and Alternative constructs allows us to manage relevant data that doesn't fit our data model definitions in close proximity with its expected target. This enables us to maintain the integrity of the source data, support derivations into forms consistent with other data in the system, and keep the semantics of our data models well defined.

References

[1] McDonald CJ. Protocol-based computer reminders, the quality of care and the non-perfectability of man. N Engl J Med. 1976;295:1351-5.

[2] Coyle JF, Mori AR, Huff SM. Standards for detailed clinical models as the basis for medical data exchange and decision support. Int J Med Inf. 2003;69:157-74.

[3] Health Level Seven, Reference Information Model, Health Level Seven, Inc., Ann Arbor, MI, 2006, http://hl7.org

Address for correspondence

Craig Parker
4646 W Lake Park Blvd
Salt Lake City, UT, USA 84093