ICT for Health Science Research
A. Shabo (Shvo) et al. (Eds.)

doi:10.3233/978-1-61499-959-1-16

Query Translation Between openEHR and i2b2

Georg FETTE[a,b,1], Mathias KASPAR[b], Leon LIMAN[a], Georg DIETRICH[a], Maximilian ERTL[b], Jonathan KREBS[a], Stefan STÖRK[b], Frank PUPPE[a]
[a] *University of Würzburg, Chair of Computer Science 6*
[b] *University Hospital of Würzburg, Comprehensive Heart Failure Center*

Abstract. Secondary use of electronic health records using data warehouses (DW) has become an attractive approach to support clinical research. In order to increase the volume of underlying patient data DWs at different institutions can be connected to research networks. Two obstacles to connect a DW to such a network are the syntactical differences between the involved DW technologies and differences in the data models of the connected DWs. The current work presents an approach to tackle both problems by translating queries from the DW system openEHR into queries from the DW system i2b2 and vice versa. For the subset of queries expressible in the query languages of both systems, the presented approach is well feasible.

Keywords. Clinical data warehouse, query, i2b2, openEHR

1. Introduction

The secondary use of electronic health records (EHR) has become an important field in medical informatics. Routine clinical data is reused for various scientific purposes, like prospective estimation of study cohort sizes or support of study cohort acquisition. In order to support access to the EHRs, data warehouses (DW) have been developed. Routine data, which is often scattered in various data sinks in various heterogeneous data formats, is aggregated in a DW in a homogeneous form in a centralized data sink, which is accessible via a standardized query interface. Two popular architectures that can serve as such a DW are i2b2 (https://www.i2b2.org) and openEHR (https://www.openehr.org).

In order to increase the volume of underlying patient data for more expressive query results, DW installations at different institutions can be connected to distributed DW networks. Systems like SHRINE [1] for i2b2 or SNOW [2] for openEHR allow queries to be distributed to connected DW instances and to aggregate the returned results. Each network system, however, only allows DWs having the same query interface to be part of the network. If a DW with a different query interface has to be integrated into a network, the data from that DW has to be transferred (like in [3]) into a new dedicated DW installation fitting the networks query interface. However, parallel support of multiple DW systems at the same institution containing the same redundant data creates an overhead in support and hardware.

[1] Corresponding author: Georg Fette, University and University Hospital Würzburg, CHFC, Am Schwarzenberg 15, 97078 Würzburg, Germany; E-Mail: georg.fette@uni-wuerzburg.de

An alternative approach is to translate the queries of an incompatible DW system into the query language of the required DW system. The current work follows this approach by translating (when possible) i2b2 queries into openEHR queries (expressed in the query language AQL) and vice versa.

2. Methods

OpenEHR's query language AQL (Archetype Query Language) is SQL-inspired and like SQL a functional language. It consists of three parts: The *FROM* part defines which structural elements are queried, the *WHERE* part constrains those elements and the *SELECT* part defines which elements have to be returned in the results. OpenEHR's data model allows elements to be nested, so that an element with the same identifier can appear multiple times in the model. Therefore, in order to unambiguously identify elements in queries they have to be identified by their paths, which are concatenations of the identifiers of the nested elements (e.g. *Ehr/Observation[LabResult]/Value*). Nesting in openEHR can have arbitrary depth.

i2b2's query language is formulated in XML and is as well a functional language. i2b2 uses an Entity-Attribute-Value schema [4], which naturally only allows one single layer of attributes to be nested in a patient record layer. In i2b2, however, facts can additionally be nested in encounter and/or instance object layers. The nesting topology in i2b2 is controlled by so-called *panel_timing* rules. As in i2b2 all data model elements are unique they can be referenced in queries solely by their identifiers.

To translate a query, it is first parsed and transformed into a graph. For parsing AQL queries, the parser from the AQL-processor of the EtherCIS project (http://ethercis.org) was taken and combined with a graph builder written by the authors. The parser and graph builder for i2b2 were written by the authors. The graphs retain the data model structure, the constraints on data element values, and which data elements have to be contained in the returned results.

Before the translation into the target language, a graph can undergo several transformations. Currently there exist three types of transformations: TBox-ABox-Transformations, Path-Transformations and Concept-Code-Mappings (see Figure 1).

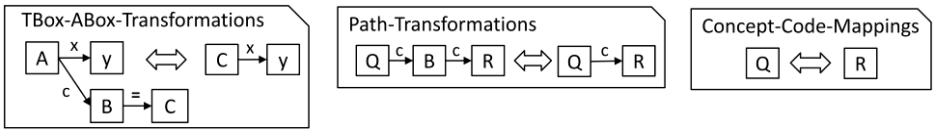


Figure 1. Current types of transformations applicable on query graphs.

TBox-ABox-Transformations are necessary because in i2b2 many data model elements (e.g. laboratory measurements) are modeled as explicit concepts, whereas in openEHR they are modeled using the same abstract concept, containing a field to parametrize the instances using terminology codes in order to represent specific measurement types. The transformation identifies a core node *A* containing a selector node *B* being equal to a specific selector equality value *C*. The core node *A* gets renamed to the selector equality value *C* and the attached selector node *B* is removed. The transformation can also be applied in the reverse direction: a node identified by a selector equality value *C* gets added an additional subgraph consisting of a selector node *B* and a

selector equality node *C* identified by the primarily matched selector node. The identifier of the primarily matched node is replaced with the core identifier *A*.

Path-Transformations enable the possibly deeply nested data model elements of openEHR to be mapped to i2b2's rather flat data model. Long paths can be reduced to short ones by removing intermediate nodes. The reverse operation inserts nodes in between short paths.

Concept-Code-Mappings map identifiers of the source data model to identifiers of the target data model.

Query graphs are translated into the target language via respective graph writers. When i2b2 is the target language, the nodes identified by *Patient*, *Encounter* and *Instance* are treated as special cases: *Patient* is mandatory and assumed to be the query graph root. The other two cases control the query's panel timings.

The proposed method was tested on manually designed AQL queries, on AQL queries contained in the AQL documentation [5] and on queries described in i2b2's query specification [6]. A query was translated into its respective counterpart and afterwards retranslated into its original language. After being retranslated into the original query language the retranslated query had to be identical (besides formatting differences) to the original query.

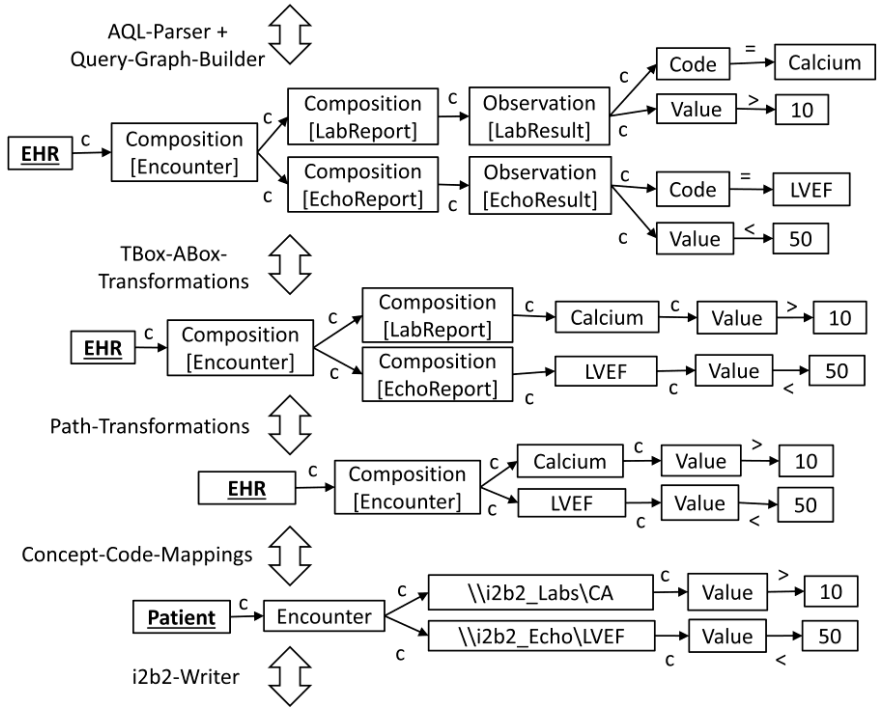
3. Results

Figure 2 pictures an example of an AQL query being transformed into an i2b2 query. The example contains a query for patients having two constrained measurements contained in different reports, which are contained in the same encounter. The AQL parse tree is translated into a graph by instantiating all archetypes as well as all archetype elements as nodes. Table 1 shows the transformation configurations used in the example. The TBox-ABox-Transformations exchange the two branches representing the *Calcium* and *LVEF* measurements by the required structures from the i2b2 data model. The Path-Transformations shorten the paths, so that the query matches the data model nesting capabilities of i2b2. The Concept-Code-Mappings exchange all openEHR identifiers with data model identifiers used in i2b2. Finally, the query graph is translated to an i2b2 query.

Table 1. Transformation configurations used in the example depicted in Figure 2. The characters in brackets behind the column headings indicate the transformation nodes in Figure 1.

TBox-ABox-Transformations		
CoreIdentifier (A)	SelectorIdentifier (B)	SelectorEquality (C)
Observation [LabResult]	Code	Calcium
Observation [EchoResult]	Code	LVEF
Path-Transformations		
SourceIdentifier (Q)	IntermediatePaths (B)	TargetIdentifier (R)
Composition[Encounter]	Composition[LabReport]	Calcium
Composition[Encounter]	Composition[EchoReport]	LVEF
Concept-Code-Mappings		
SourceIdentifier (Q)	TargetIdentifier (R)	
Calcium	\\i2b2_Labs\CA	
LVEF	\\i2b2_Echo\LVEF	
Composition[Encounter]	Encounter	
EHR	Patient	

```
SELECT e FROM EHR e CONTAINS COMPOSITION c[Encounter] CONTAINS
  (COMPOSITION d[LabReport] CONTAINS OBSERVATION a[LabResult]) AND
  (COMPOSITION e[EchoReport] CONTAINS OBSERVATION b[EchoResult])
WHERE a.code = ‚Calcium‘ AND a.value > 10 AND b.code = ‚LVEF‘ AND b.value < 50
```




```
<query_definition>><query_timing>SAME_ENCOUNTER</query_timing>
 <panel><panel_timing>SAME_ENCOUNTER</panel_timing>
  <item><item_key>\\i2b2_Labs\CA</item_key>
   <constrain_by_value><value_operator>GT</value_operator>
     <value_constraint>10</value_constraint></constrain_by_value>
  </item></panel>
 <panel><panel_timing>SAME_ENCOUNTER</panel_timing>
  <item><item_key>\\i2b2_Echo\LVEF</item_key>
   <constrain_by_value><value_operator>LT</value_operator>
     <value_constraint>50</value_constraint></constrain_by_value>
  </item></panel></query_definition>
```

Figure 2. Translation of an AQL query into an i2b2 query. Underlined nodes denote data elements that have to be included in the query result. c-relations denote nesting.

The translation experiments with queries contained in the language specifications of the respective systems succeeded only on a limited subset. The translation works only for queries containing exclusively operators, which are available in AQL as well as in i2b2. The intersection of both systems' operator sets are comparators on single values {>, >=, <, <=, =} and Boolean operators {*and, or, not*}. In i2b2 nesting of Boolean operators is not arbitrarily possible because queries have to be presented in conjunctive normal form (CNF). Furthermore, in i2b2 queries may contain at most one *Encounter* node which has multiple concepts attached, as *Same_Encounter* timings always hold for all concepts in the whole query. The same is the case for *Same_Instance* timings. The common intersection of potential return value types of both query languages is {*Patient, Encounter*}. AQL supports the return of arbitrary concept types and combinations of those, which is not possible in i2b2. i2b2, on the other hand, supports additional return

types incorporating aggregation operators (e.g. *PatientCount*), which are not supported by AQL. Furthermore, i2b2 supports some rather specific return values like *PatientRace* or *PatientAge* which could be expressed in AQL but which are not treated here because of their high specificity.

4. Discussion

We presented an approach to translate openEHR/AQL queries into i2b2 queries and vice versa and illustrated the approach in an example.

Queries expressible in both languages can be automatically translated, which allows instances of the two query systems to be transparently included in research networks of the other type. This possibility enables the fusion of formerly separated research networks of i2b2 and openEHR to much larger networks having larger pools of incorporated patient data. This would facilitate statistical evaluations of routine patient data a give results from these evaluations a much larger impact.

As i2b2 and AQL do not comprise the same sets of query operations and differ in model expressiveness, the translation capability of the presented approach is restricted to the subgroup of queries that are expressible in both languages. 'Expressible in a language' means that all involved operators exist in that language and that the data model supports the nesting structure of all data elements contained in the query.

To extend the translation capabilities of the proposed approach the set of graph transformations could be enriched by transformations that make unavailable operators be expressed by semantically equivalent combinations of other operators (e.g. *i matches(|10..20|)* could be substituted by *i >= 10 and i <= 20*). The restriction of i2b2 being limited to nested Boolean clauses in CNF could be mended with graph transformations that transform arbitrary Boolean clauses into a CNF.

An aspect not yet tackled is how query results are returned by the respective systems, as each system has its own syntax and control flow for delivering results. For that topic, proper adapters would have to be developed as well.

An implementation of the presented approach in Java is available at https://gitlab2.informatik.uni-wuerzburg.de/gef18bg/cdw_querymapper.

This research was funded by grant of German Federal Ministry of Education and Research (Comprehensive Heart Failure Center Würzburg, grants #01EO1004 and #01EO1504).

References

[1] Weber GM, Murphy SN, McMurry AJ, et al. The Shared Health Research Information Network (SHRINE): a prototype federated query tool for clinical data repositories. *J Am Med Inf Assoc.* 2009;16(5):624-30.

[2] Hailemichael MA, Marco-ruiz L, Bellika JG. Privacy-preserving Statistical Query and Processing on Distributed OpenEHR Data. *Stud Health Technol Inform.* 2015;210:766-70.

[3] Haarbrandt B, Tute E, Marschollek M. Automated population of an i2b2 clinical data warehouse from an openEHR-based data repository. *J Biomed Inform.* 2016;63:277-294.

[4] Dinu V, Nadkarni P. Guidelines for the effective use of entity-attribute-value modeling for biomedical databases. *Int J Med Inform* 2007;76(11-12):769-79.

[5] AQL language specification, https://www.openehr.org/releases/QUERY/latest/docs/AQL/AQL.html, accessed 10.10.2018

[6] i2b2 CRC messaging guide, https://www.i2b2.org/software/files/PDF/current/CRC_Messaging.pdf, accessed 10.10.2018