A Generic Transformation of HL7 Messages into the Resource Description Framework Data Model

[1,2] Fabian Prasser, [1,2] Florian Kohlmayer, [2] Alfons Kemper, [1] Klaus Kuhn

[1] Department of Medicine, Technische Universität München
[2] Department of Computer Science, Technische Universität München

firstname.lastname@tum.de

Abstract: The inherent flexibility of the RDF data model has led to its notable adoption in many domains, especially in the area of life-sciences. In some of these domains, there is an increasing need to integrate data from various distributed sources. In translational medical research, an emerging domain of high relevance, the access to biomedical data sources that contain important primary data (e.g., clinical information systems or research databases) is a crucial requirement. In patient-care, information systems exchange information via a standardized application level protocol, called HL7. This paper presents a generic component for transforming such message streams into an RDF representation and loading them into an RDF database system. This allows to seamlessly integrate clinical data into biomedical Semantic Web applications.

1 Introduction

Translational medical research is an emerging concept which aims at transforming discoveries from basic sciences into diagnostic and therapeutic applications. In the opposite direction, clinical data are needed for feedback and as stimuli for the generation of new research hypotheses [Zer05]. This process is highly data-intensive and centered around the idea of integrating data from basic biomedical sciences, clinical sciences and patient care. The complexity and heterogeneity of the involved data is constantly growing with increasing scientific progress. To cope with this complexity, structured domain knowledge, e.g. from knowledge bases, is often required in order to adequately understand and interpret results [PES09]. Security and privacy questions are relevant, but out of the scope of this article. Typical use cases include the linkage and mapping of phenotypes and genotypes from research or clinical systems, IT support for clinical trials, the provision of data for statistical analyses and the integration of knowledge bases.

The *Resource Description Framework* (RDF) data model is well suited to cope with these domain-specific requirements [PWL+11]. Firstly, data is modeled as a network of objects, which makes RDF well-suited for the canonical representation of heterogeneous datasets and structures. Secondly, RDF provides explicit formal semantics which allow to decompose an RDF dataset into comprehensible atomary statements, even if there is no thorough understanding of the data. Thirdly, RDF enforces the explicit definition of entities, identifiers and relationships. For this reason, RDF data can be easily combined with other datasets. At the same time RDF is characterized by its consistency, as data, metadata

and semantics can be represented within one model. In recent years, RDF has attracted increasing attention from the bioinformatics community (e.g., [BNT+08]).

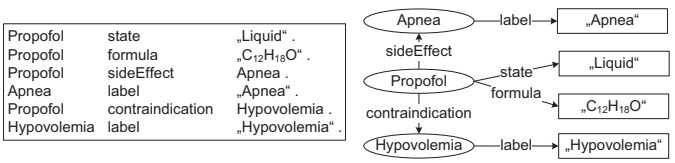

Figure 1: Example set of RDF triples and the resulting RDF graph[1]

RDF is a graph-structured data model in which information is modelled as a set of triples. A triple (Subject, Predicate, Object) states that the *Subject* has a property *Predicate* with value *Object*. Subjects and predicates are always *Resources* whereas objects can also be *Literals*. Resources are identified by *Uniform Resource Identifiers* (URIs) which are a proper superset of URLs. Literals are atomic values with optional type information (e.g., integer). An RDF graph is a directed labeled graph in which subject and object are nodes and predicates are edges ranging from the subject to the object. Figure 1 shows an example set of RDF triples as well as the resulting RDF graph.

```
MSH|^~\&|SENDER|099|RECEIVER||20090618122708||ADT^A01|42513321|P|2.3
EVN|A01
PID|||00048441934||Prasser^Fabian^Herr|Prasser|29533212|M|||^^^^^D||||D|||||||||||D|||00000000000000
NK1||Prasser||^^^^^D
PV1|||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||^^^^^^^^^^^^^^^^^^^^NP0100
```

Figure 2: Example HL7 V2 message of type ADT-A01

HL7 is a widespread messaging standard for information exchange between clinical information systems [HL712]. It is an application-level protocol, which is typically managed via an integration engine. For translational medical research, HL7 messages contain several important types of data. Firstly, it can be easier to extract clinical data from HL7 messages than it is to directly access the originating information system. Secondly, administrative information often contains important metadata. An example are reconciliation events, which occur when two different identifiers have been found for the same patient. In this case, the according patient identifiers are reconciled and HL7 messages are sent. Clean identifying data is highly relevant for data integration systems, as it is often required to ensure consistency in replicated data.

The HL7 messaging standard defines different message groups, each of which contains different message types. Each message type itself is defined by a set of segments which again consist of different groups and fields. For example, a message group exists for the admission, discharge and transfer (ADT) of patients. If a patient is admitted, a message of type ADT-A01 is sent to all subsystems. In version 2.X of the HL7 standard (referred to as HL7 V2), messages are encoded in plain text and separator characters are utilized to encode fields. An example HL7 V2 message is shown in Figure 2.

2 Related Work and Contribution

Many solutions for accessing differently structured data from within Semantic Web applications have been proposed. Those solutions cover a broad design spectrum, as has

[1] URIs have been abbreviated for better readability

been investigated in [GC07] and [SSJ+09]. Most work focuses on the widespread relational model, but approaches for other data models also exist, e.g., XML. A transformation solution for HL7 messages can be characterized analogously, as a set of messages can also be seen as a database with a schema which implements the HL7 specification. A *database-centric* transformation creates an RDF graph from the underlying database schema. The result closely resembles the original schema, whereas *ontology-centric* approaches have been developed for cases in which a database needs to be mapped onto an existing ontology. Some approaches not only allow to *materialize* the transformed dataset, but also provide means for on-demand *query rewriting*. In the latter case queries against the virtual RDF dataset are rewritten to queries against the target database system. Transformation solutions can also be categorized by whether they allow for an *automatic* or *semi-automatic* mapping definition or require a completely *manual* process. In contrast to database-centric approaches, ontology-centric solutions always require at least some manual work due to the complexity of the underlying mapping problem. Some approaches are *domain-dependent* and allow to include existing domain ontologies into the mapping definition and data transformation process.

A lack of metadata and well-defined semantics is a general problem in earlier versions of HL7 (see Figure 2). Therefore, HL7 V3 has been redeveloped from scratch utilizing a common domain model, the Reference Information Model (RIM). It is object-oriented (designed with UML), also covers clinical documents (Clinical Document Architecture, CDA), makes extensive use of wide-spread terminologies, and the data formats (e.g., for messages and documents) are based upon XML. Some previous work have aimed at bringing HL7 V3 to the Semantic Web. In the context of [HCL12] and in [HL7RIM] RDF representations of important parts of HL7 RIM have been developed. Some work (e.g., [KRA06, GRD12]) have utilized XSLT to derive RDF representations of XML-based HL7 V3 messages and integrate them with HL7 RIM. As there is currently no comprehensive RDF representation of HL7 RIM, these approaches only implement limited scenarios. Other work (e.g., [JS12]) aim at the opposite direction and utilize RDF to simplify the process of mapping clinical data to HL7 V3 to foster interoperability. In comparison to HL7 V2, deriving an RDF representation is less complex for HL7 V3 messages, as these are XML-based and contain meaningful metadata. Unfortunately, HL7 V3 is not compatible to previous versions. Because a migration is highly complex, it is still only rarely used.

This work focusses on deriving an RDF representation from HL7 V2 messages in order to integrate the contained data into Semantic Web applications. Although the approach is oriented towards HL7 V2, it can also handle HL7 V3 messages. The major challenges adressed in this work are 1) to obtain meaningful metadata for the data items in HL7 messages, and 2) to ensure that the solution is able to handle real-world data volumes.

3 Transformation Process

This section presents a *fully-automatic*, *domain-dependent* transformation approach for HL7 messages, which implements *materialization*. The approach is able to automatically transform valid HL7 messages into an RDF representation. It is domain-dependent and implements a concept in-between the design space of *database-centric* and *ontology-centric* solutions. The resulting RDF graph is materialized in a dedicated RDF database system.

The overall system architecture is shown in Figure 3. Here, the *HL7 Transformer* is registered as a receiver at the integration engine. It transforms all incoming messages into an RDF representation and incrementally maintains a definition of the resulting schema by utilizing the RDF Schema Description Vocabulary (RDF/S).

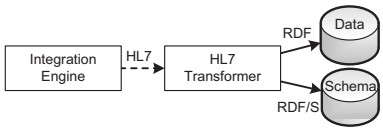

Figure 3: Architecture of the HL7 to RDF transformer

In HL7 V2, the semantics of a data item (field name) is defined implicitly by the message type, the segment and the position of a group or field within the segment. In order to create meaningful predicates in the RDF representation, a machine-readable format of the HL7 specification is required. Such a specification is provided by the HAPI project [HAP12] which implements a generic HL7 parser for Java. HAPI is utilized to parse each message and Java reflection is used to traverse the resulting object model, which provides field names. In this process, only those segments, groups and fields are transformed for which data exists within the message. Therefore, the resulting RDF representation is very compact as can be seen in Figure 4. In a real-world scenario, a large number of segments, groups and fields specified for the different message types are never used. Simply generating an RDF/S schema definition out of the HL7 specification would therefore result in a large volume of redundant metadata. In contrast, an incremental approach is implemented which always updates a global RDF/S schema description after processing a message.

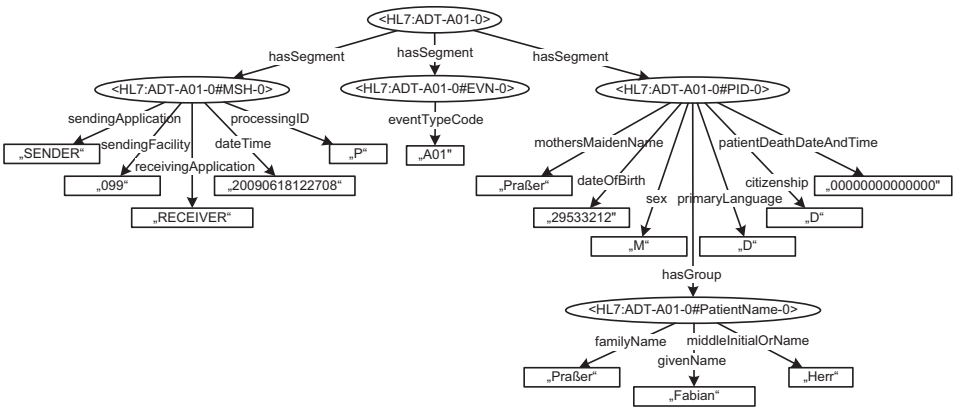

Figure 4: Excerpt of the RDF representation of the HL7 message from Figure 2 [1]

UML activity diagrams for the described processes are shown in Figure 5. Messages are transformed in batches. To this end, the transformation can either be executed periodically or when a pre-defined number of messages has been received. When a batch is processed, each message is transformed into an RDF representation and the RDF database is updated. Afterward processing a batch, the database is reorganized. This step is specific to the database system used by our implementation and will be explained in the next section.

[1] Links to the associated schema description and complete URIs have been omitted for better readability

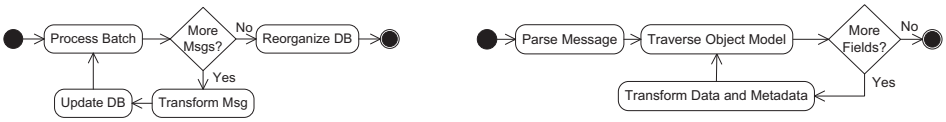

Figure 5: Processing (1) a batch of messages and (2) an individual message

When processing an individual message, it is first parsed into a Java object. The resulting object model is traversed recursively and for each method it is checked whether it returns a valid result (e.g., not `null`). If it does return a primitive value, it is materialized in the resulting RDF graph by deriving a meaningful predicate from the method name. Otherwise, the returned object is traversed. In order to only retrieve data which is part of the message, we exclude some methods from this process (e.g.,`hashCode()` or `toString()`). Identifiers for objects (see Figure 4) are generated incrementally, with identifiers for segments, groups and fields being defined relative to the current message identifier.

4 Evaluation

In this section we evaluate the performance of our solution with realisitic data characteristics. Because the data is extracted from a continuous message stream, the bottleneck in this process is the insertion of new data into the underlying database. The overhead induced by the transformation itself is negligable in this context. The experiments were performed on a Dell laptop with a 4-core 1.6 GHz Intel Core i7 CPU with 6 MB cache and 4 GB of memory running a 64-bit Linux 2.6.35 kernel. The system is able to perform sequential reads and writes on the local hard disk with about 100 MB/s.

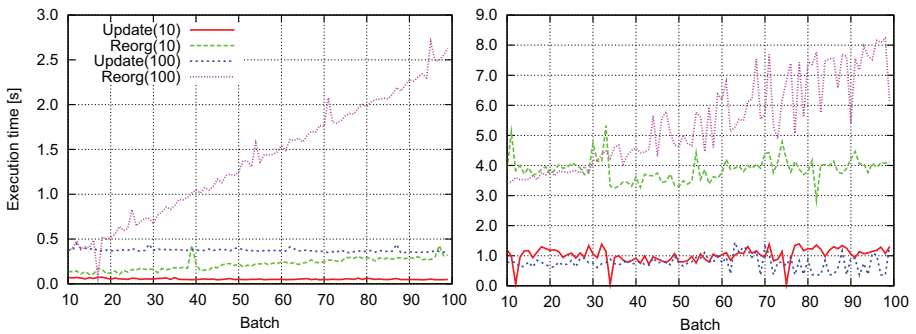

Figure 6: Batches of size 10 and 100 on (1) an empty database, (2) a database with 200k triples

As an RDF database system we used the RDF-3X triple store, because it offers excellent performance [NW10]. As RDF-3X maintains a highly compressed index of all possible permutations of the triples' subjects, predicates and objects, updating data is not straightforward. The system implements a two-step process. 1) When data is written to the database (*Update*) it creates so-called differential indexes for the new data. 2) A reorganization step merges the differential indexes back into the main indexes (*Reorg*).

The experiments consisted of randomly created batches of HL7 messages which were sent to the transformation component. The benchmarks were performed on an empty database, as well as on a database with an initial size of 200.000 triples. Each experiment contained 100 batches with batch sizes of 10 and 100 messages. On average, one message resulted in about 30 triples. An experiment with the larger batch size corresponds to the volume of ADT data which is created within a large maximum care hospital in one month. As can be

seen in Figure 6, the execution time of the update process mainly depends on the batch size and grows only slightly with the size of the database. In contrast, the execution time of the reorganization step clearly increases with the overall data volume. The solution scales very well and is easily able to handle realistic data volumes.

5 Conclusion

We have presented a generic and scalable component for transforming HL7 message streams into the RDF data model. The described approach is able to support the cleansing of identifying data, which is an essential requirement for biomedical integration systems. Furthermore, it implements a lightweight approach for accessing clinical information as RDF data, because it allows to automatically transform clinical message streams. This can be utilized by biomedical Semantic Web applications.

The presented component is part of our incremental ontology-based integration platform for translational medical research [PWL+11]. It builds upon the RDF data model and offers a schema-relaxed environment for integrating clinical data, research data and knowledge bases. In this context, the flexibility of RDF allows to integrate heterogeneous and differently structured data, such as HL7 messages and biomedical knowledge. Additionally, the expressiveness and inherent semantics of RDF and its related ontological vocabularies are utilized to annotate, transform and semantically integrate the data by means of ontology-alignment and semantic reasoning techniques.

References

[BNT+08] Franois Belleau et al. Bio2RDF: Towards a mashup to build bioinformatics knowledge systems. *J Biomed Inform*, 41(5):706–716, 2008.

[GC07] Raji Ghawi and Nadine Cullot. Database-to-Ontology Mapping Generation for Semantic Interoperability. In *InterDB*, 2007.

[GRD12] GRDDL Use Cases: Scenarios of extracting RDF data from XML documents. http://www.w3.org/TR/grddl-scenarios/, June 2012.

[HAP12] Hapi - The Open Source HL7 API for Java. http://hl7api.sourceforge.net/, June 2012.

[HCL12] W3C HCLS Interest Group. http://www.w3.org/2001/sw/hcls/, June 2012.

[HL712] Health Level Seven International. http://www.hl7.org/about/index.cfm, June 2012.

[JS12] Priya Jayaratna and Kamran Sartipi. HL7 v3 message extraction using Semantic Web techniques. *IJKEDM*, 2(1):89–115, 2012.

[KRA06] Jan Kunze et al. Eine Schnittstelle für Arztpraxisdaten mittels einer Ontologie auf Basis von HL7 Version 3. In *Tagungsband XML-Tage*, September 2006.

[NW10] Thomas Neumann and Gerhard Weikum. The RDF-3X engine for scalable management of RDF data. *VLDB J.*, 19(1):91–113, 2010.

[PES09] Philip R O Payne, Peter J Embi, and Chandan K Sen. Translational informatics: enabling high-throughput research paradigms. *Physiol Genomics*, 39(3):131–140, 2009.

[HL7RIM] Health Level Seven. http://protege.stanford.edu/ontologies/HL7RIM/, June 2012.

[PWL+11] Fabian Prasser et al. Inkrementelle ontologiebasierte Informationsintegration für die translationale medizinische Forschung. In *GI Edition: LNI*, 192:157–157, 2011.

[SSJ+09] Sören Auer et al. Triplify: light-weight linked data publication from relational databases. In *WWW*, 2009.

[Zer05] Elias A Zerhouni. Translational and clinical science–time for a new vision. *New Engl J Med*, 353(15):1621–1623, 2005.