doi:10.3233/978-1-61499-808-2-122

Technical Environment for Developing the SNIK Ontology of Information Management in Hospitals

Konrad HÖFFNER[a,1], Franziska JAHN[a], Christian KÜCHERER[b], Barbara PAECH[b], Birgit SCHNEIDER[a], Martin SCHÖBEL[a], Sebastian STÄUBERT[a] and Alfred WINTER[a]

[a] *IMISE, University of Leipzig, Härtelstraße 16-18, 04107 Leipzig, Germany*

[b] *Software Engineering Group, Heidelberg University, Institute for Computer Science, Im Neuenheimer Feld 205, 69120 Heidelberg, Germany*

Abstract. The SNIK project converts textbooks about information management in hospitals to a domain ontology that provides a shared vocabulary for institutions to model and integrate processes, data and infrastructure. To accommodate user groups with different requirements and technical backgrounds, and to support incremental and cooperative development, we create a system architecture to publish, visualize, browse and query the ontology, as well as to evaluate and improve the data quality.

Keywords. Health Information Management, Ontology, Hospital Information System

1. Introduction

1.1. Introduction

Knowledge about hospital information systems and their management (HIM) is mainly available through textbooks, articles, frameworks, standards and experts working in this field. In order to structure and formalize the HIM knowledge, we developed *SNIK* (German: *Semantisches Netz des Informationsmanagements im Krankenhaus*), an ontology that integrates terms, definitions and relations of strategic, tactical and operational information management, Information Technology (IT) service management and IT governance with a special focus on healthcare [1]. SNIK is manually extracted from HIM textbooks and is modelled as a *Resource Description Framework (RDF)* graph, which represents machine processable knowledge about arbitrary concepts in the form of *triples,* which specify binary relations between resources. An *RDF resource* is uniquely represented by a *Uniform Resource Identifier (URI)* and can be an individual, a property (relation) or a class (set of individuals). Each triple has the form "(subject, property, object)". For example, we encode the sentence "Activity diagrams are part of UML." as the triple (ActivityDiagram, subClassOf, UmlDiagram). As the SNIK ontology relates to abstract textbook knowledge and not to any concrete hospital, SNIK does not contain individuals. SNIK consists of 1936 classes, 63 properties and 63916

[1] Corresponding Author, E-mail: konrad.hoeffner@imise.uni-leipzig.de.

triples in version 0.3.3. We used SNIK in teaching to help health informatics students acquire and practice the terminology of HIM [2]. SNIK also can be used for requirements engineering and software development [15]. In the following, we present the requirements, concept and implementation of the SNIK system architecture to provide quality assurance, visualizations and services for querying and browsing the ontology.

1.2. Requirements

Different technical challenges have to be considered when providing a complex ontology for user groups with different backgrounds, but also during its incremental and cooperative development: (1) Convert manually created extraction tables to an RDF ontology. (2) Integrate a new subontology into SNIK. (3) Enable users to interact with the ontology without installing specialized tools. (4) Assure the quality of the ontology.

2. State of the art

The system architecture of SNIK is inspired by successful Semantic Web projects: DBpedia [3] is the RDF conversion of Wikipedia. Due to its size and generality, there is a large amount of interlinks to DBpedia, making it a central part of the Semantic Web. Besides millions of instances, DBpedia also contains a handcrafted ontology. The data is available over a SPARQL[2] endpoint, an RDF browser and as RDF dumps. Healthcare science also has a prominent [14] status in the Semantic Web with transformations of databases such as PubMed [4], Drugbank [5] and SIDER [6].

3. Concept

We fulfill the requirements listed in section 1.2 by introducing a system architecture that consists of established libraries, Semantic Web tools, and development platforms. We thus also provide a blueprint for developers and maintainers of other large ontologies. In particular, (1) to enable RDF conversion, we configure a CSV to RDF conversion tool and (2) we use a link discovery framework to integrate new subontologies into SNIK. (3) A *graph-based visualization* gives an overview and emphasizes relations, while an *RDF browser* provides details about particular resources. A *SPARQL endpoint* answers formal queries and enables the usage and development of other tools and services. (4) To ensure a high data quality, quality management determines objective quality criteria that are used both to compute quality scores and to find offending triples.

[2] Recursive Acronym for *SPARQL Protocol and RDF Query Language*

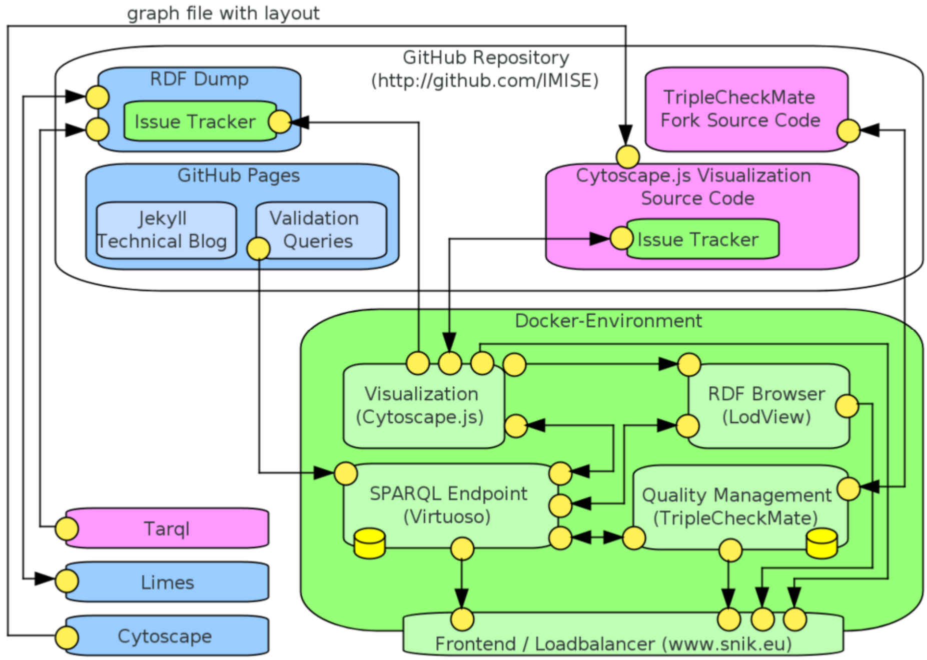

Figure 1. SNIK system architecture 3LGM² model [12]. Rounded rectangles depict application components, arrows pointing from circle to circle illustrate communication over interfaces. For initial setup, all application components are involved, but violet components have to run once. The green components are running at normal operation. To perform an update, the blue components are active and also the green components are affected.

4. Implementation

The components of the SNIK system architecture and their relations are shown in Figure 1. The applications are provided as Docker containers if possible to ease setup, deployment and portability. First, the knowledge extracted from information management textbooks to tabular data by domain experts needs to be converted to RDF. To set up a new subontology, we convert the tables using *Tarql*[3] with a mapping configuration file[4]. On update, we upload the serialized RDF to our public *Git repository* and *SPARQL endpoint*[5]. We serialize the ontology from the endpoint to a table, which we import and layout in Cytoscape, and then move to the visualization repository. We publish subsequent releases both periodically, after the initial release accompanying the quality assurance efforts, and also when significant user feedback is accumulated through the issue tracker and implemented. After quality control according to quality criteria defined in [7], we also create link candidates between the subontologies using *LIMES*, the *Link Discovery Framework for Metric Spaces* [8] with the *RDF dump* and a mapping configuration. A high confidence mapping consists of all class pairs that have a label in common. A lower confidence mapping consists of all class pairs having a label pair with

[3] http://tarql.github.io
[4] https://github.com/IMISE/snik-csv2rdf
[5] OpenLink Virtuoso 7.20.3217, supports SPARQL 1.1

trigram similarity of 0.8 or higher. Both mappings are manually verified where unrelated pairs are discarded. The remaining ones are categorized using relations from the Simple Knowledge Organization System (SKOS) [9]. Non-ontologists interact with the ontology over the RDF browser and the visualization. The browser queries the SPARQL endpoint to resolve the URI of a resource[6] to a HTML page with properties of the resource and links to others. A machine-processable RDF description of a single resource is available as well. It is implemented as a fork of LodView[7] with minor modifications.

The visualization is based on the Cytoscape.js library, a JavaScript version of Cytoscape [10]. The library shows the ontology as a graph and calculates the shortest paths between classes. For teaching, we created the “spiderworm”, which combines the shortest path between two classes with all connected nodes of the target class [2]. Additional custom functionality includes a search using exact string matching with the class labels from the SPARQL endpoint, a collection of filters to simplify the graph, and access to the RDF browser description of a class. Modelling errors can be reported to the RDF dump issue tracker by sending the user to a pre-filled issue template. Users can also report visualization problems itself through the issue tracker of the visualization.

While the issue tracker mostly handles issues that are incidentally noticed, we also offer a TripleCheckMate [11] instance for dedicated quality control of specific or randomly chosen classes queried from the SPARQL endpoint. TripleCheckMate was originally developed to evaluate the data quality of DBpedia instances, but it is configurable enough that we could adapt a fork with small changes to our ontology and to evaluate classes instead of instances. Evaluation results are saved in a SQL database. Domain and range statements as well as Web Ontology Language (OWL) restrictions are also automatically verified using SPARQL queries.

5. Lessons learned

Developing a JavaScript visualization based on Cytoscape.js instead of using the Cytoscape desktop application allowed us to have custom functionality with little development time, easy deployment and updates, and no installation requirement. Using Tarql with custom mapping configuration files saved us considerable costs for software development, adaption and maintenance compared to a self-developed solution. Trough hosting our code bases and ontologies on GitHub, we take advantage of public access, history, simultaneous editing, releases, issue trackers, and outside code patch suggestions. Collecting data quality issues automatically through SPARQL queries, from dedicated manual fact checking over TripleCheckMate, and from incidental observations over the issue tracker connection in the visualization, maximizes data quality improvements.

6. Conclusion

With the help of Semantic Web technologies and Docker, we developed a modular and flexible architectural environment to deal with a growing ontology. Users benefit from easy access over a multi-view web interface and different possibilities to interact with the ontology and to jointly improve its quality. Nevertheless, for using the provided

[6] such as http://www.snik.eu/ontology/meta/Top

[7] http://lodview.it/

features and maintaining the ontology, active users are needed. Therefore, we have to promote the SNIK ontology on data portals and social media for it to be actively used by medical informatics students, lecturers and software developers.

7. Conflict of Interest

The authors state that they have no conflict of interests.

8. References

[1] Schaaf M, Jahn F, Tahar K, Kücherer C, Winter A, Paech B. Entwicklung und Einsatz einer Domänenontologie des Informationsmanagements im Krankenhaus. In: INFORMATIK 2015 Lecture Notes in Informatics (LNI). Bonn: Gesellschaft für Informatik e.V. (GI); 2015. p. 753–65.

[2] Jahn F, Schaaf M, Kahmann C, Tahar K, Kücherer C, Paech B et al. An Ontology-Based Scenario for Teaching the Management of Health Information Systems. In: Exploring Complexity in Health: An Interdisciplinary Systems Approach: Proceedings of MIE2016 at HEC2016. Amsterdam: IOS Press; 2016. p. 359–63 (Stud Health Technol Inform; vol. 228).

[3] Bizer C, Lehmann J, Kobilarov G, Auer S, Becker C, Cyganiak R and Hellmann S. DBpedia - A crystallization point for the Web of Data. *Web Semantics: Science, Services and Agents on the World Wide Web. Amsterdam: Elsevier Science Publishers B. V.* 7 154–65; 2009

[4] Roberts R J. PubMed Central: The GenBank of the published literature. In: *Proceedings of the National Academy of Sciences of the United States of America* 98 381–2; 2001

[5] Law V, Craig K, Djoumbou Y, Jewison T, Guo A C *et al.* DrugBank 4.0: Shedding New Light on Drug Metabolism. *Nucleic acids research* 42 D1091-7; 2014

[6] Kuhn M, Letunic I, Jensen L J and Bork P. The SIDER database of drugs and side effects. *Nucleic acids research* 44 D1075-9; 10.1093/nar/gkv1075; 2016

[7] Zaveri A, Rula A, et al. Quality assessment for Linked Data: A Survey. S*emantic Web* 1 1–5; 2012

[8] Ngomo A-C N and Auer S 2011 LIMES - A Time-Efficient Approach for Large-Scale Link Discovery on the Web of Data *IJCAI-11: In: Proceedings of the twenty-second international joint conference on artificial intelligence.* Menlo Park: AAAI Press; *2011.* pp 2312–7

[9] Miles A. SKOS: Requirements for Standardization. In: *Proceedings of the 2006 International Conference on Dublin Core and Metadata Applications*; 2006. pp 55–64

[10] Schaaf M, Jahn F, Tahar K, Kücherer C, Winter A, Paech B. Visualization of Large Ontologies in University Education from a Tool Point of View. In: Exploring Complexity in Health: An Interdisciplinary Systems Approach: Proceedings of MIE2016 at HEC2016. Amsterdam: IOS Press; 2016. p. 349–53 (Stud Health Technol Inform; vol. 228).

[11] Kontokostas D et al. TripleCheckMate: A Tool for Crowdsourcing the Quality Assessment of Linked Data. *Knowledge Engineering and the Semantic Web.* Berlin Heidelberg: Springer; 2013.

[12] Winter A, Brigl B, Wendt T. Modeling Hospital Information Systems (Part 1): The Revised Three-Layer Graph-Based Meta Model 3LGM^2. Methods Inf Med 2003; 42(5):544–51.

[13] Merkel D. Docker: Lightweight Linux Containers for Consistent Development and Deployment. Linux Journal. Houston: Belltown Media; 2014 (vol. 239), 2.

[14] Cheung K, Prud'hommeaux E, Wang Y, Stephens S. Semantic Web for Health Care and Life Sciences: A Review of the State of the Art. : 111-113. Brief Bioinform; 2009. 10 (2): pp 111—113.

[15] Kücherer, C. Use of Domain Ontologies to Improve Requirements Quality. In: Requirements Engineering: Foundation for Software Quality. Berlin Heidelberg: Springer; 2017.