Knowledge Tags Maintenance in Heterogeneous Web Content: The Repository

Karol Rástočný[*]
Institute of Applied Informatics
Faculty of Informatics and Information Technologies
Slovak University of Technology in Bratislava
Ilkovičova 3, 842 16 Bratislava, Slovakia
rastocny@fiit.stuba.sk

Abstract

Knowledge tags provide new layer of a lightweight semantics over the Web content, in which computer systems and the Web users share their knowledge about tagged content. To allow this type of collaboration, tags' data must be stored in form which is understandable for computer systems and provides flexible and fast access for these systems. The next issue is dynamic character of the Web, whose content changes in time. These changes can invalidate knowledge tags, their parts or anchoring in tagged content. In this paper we present basic ideas of automatized knowledge tags maintenance and our approach for knowledge tags repository.

Categories and Subject Descriptors

E.2 [**Data Storage Representations**]: Object Representation; H.2.4 [**Database Management**]: Systems—*distributed databases, query processing*

Keywords

knowledge tags, lightweight semantics, Open Annotation model, SPARQL, MapReduce

1. Introduction

Computer systems need Semantics in the Web for fulfilling their roles, e.g. for recommending. For this reason, they process documents (i.e. textual files of different type and structure) on the Web and assign knowledge tags, short metadata (e.g., identified keywords) to their parts [12]. By sharing of these knowledge tags, new layer of the lightweight semantics over the Web can be created collaboratively. But systems currently store these metadata to their private repositories in forms that are understandable only for them, so other systems could not use these metadata and they have to do this work redundantly. This inspired us to propose method for knowledge tags maintenance, which allows systems to share their metadata. To propose this method, we negotiate with two sources of complications:

- *Dynamicity of the Web* – the Web content is not stable. The Web documents arise, are deleted and what is a cause of the most of problems, they are changed without a notice. These changes in tagged documents invalidate anchored knowledge tags that have to be modified or deleted.
- *The repository* – knowledge tags repository has to store a large amount of knowledge tags in flexible open format which has to be generally understandable for computer systems and the repository has to still provide fast parallel access for a numbers of computer systems.

The main contribution of this paper is a proposition of a repository and a method, how to store knowledge tags in it. But to understand of requirements for the repository, we also present basics of our automatic knowledge tags maintenance approach, how it works with repository and our presumptions, how third part systems will create and use knowledge tags.

2. Related Work

Dynamicity of the Web affects both knowledge tags' content and anchoring. The knowledge tags' content maintenance is the most complicated task, because this part of a maintenance requires an understanding of knowledge tags' content and their meaning. Only after the understanding, changes in the Web content can be reflected to knowledge tags' content, e.g. a correction of a typing mistake in a phrase on a web-page has to be reflected to a knowledge tag with keywords by updating of a keyword with the corrected phrase. Because of the knowledge tags' content maintenance complexity, it is often performed only in its the simplest form, in which all knowledge tags (i.e. annotations or metadata) are anchored to

[*]Doctoral degree study programme in field Software Engineering. Supervisor: Professor Mária Bieliková, Institute of Informatics and Software Engineering, Faculty of Informatics and Information Technologies, STU in Bratislava. Work described in this paper was presented at the 8th Student Research Conference in Informatics and Information Technologies IIT.SRC 2012.

Rástočný, K. Knowledge Tags Maintenance in Heterogeneous Web Content: The Repository. Information Sciences and Technologies Bulletin of the ACM Slovakia, Special Section on Student Research in Informatics and Information Technologies, Vol. 4, No. 2 (2012) 59-61

specific version of a document and after updating of the document, all anchored knowledge tags are removed or invalidated in new versions of the document [10].

Knowledge tags' anchoring maintenance is obviously based on a choice of a suitable robust anchoring, which can autonomously (without necessity of original version of tagged documents) react on tagged document modifications of one of tree degrees [9]:

- *Without influence* – a document was edited after the knowledge tag's position;
- *Simple shift of a knowledge tag* – a document was edited before the knowledge tag's position;
- *Complex modification* – a document was edited at the knowledge tag's position. This type of modifications has straight influence to both the anchoring and also the knowledge tag's content. In this case it is necessary to make decision if we can update the anchoring or the knowledge tag was orphaned (we could not assign new position of the knowledge tag in tagged document) and also how the knowledge tag's content has to be updated.

The initiative of uniform robust anchoring method has been started up by Phelps and Wilensky that define basic criteria of anchoring robustness that are mainly focused on anchoring simplicity and its automatic correction from new version of document without necessity of a document's change-set [7]. They also proposed their own anchoring based on SGDOM (simple, general document object model), which describes tree structure of textual documents. Logical parts of document have their own identifiers in a SGDOM description, what Phelps and Wilensky utilize and propose three descriptors of an anchoring that are tolerant to a different complexity of changes in tagged document on the expense of computational complexity:

- *SGDOM identifier* – the simplest descriptor, but not robust to change in the identifier;
- *Path in a SGDOM tree* – robust to local modifications that change SGDOM element's identifier, but intolerant to more complex modifications that change structure of SGDOM tree;
- *Context* – tolerant to more complex changes, but is not trivially interpretable.

A similar anchoring based on tree structure of documents is useb by iAnnotate tool [8] which anchors users' annotations in webpages to DOM objects. The anchoring is focused on changes in a webpage presentation (e.g., zoom or change of resolution), to which iAnnotate reacts by obtaining new positions of DOM objects. iAnnotate stores annotations to relational MySQL store which has good performance, but is not web-scalable and it requires fixed annotations format.

Annotea system [4] uses DOM-based anchoring, too, but it stores annotations in RDF model, which gives great annotations structure flexibility. The model inspired Open Annotation (OA) Collaboration to propose flexible open annotation model, which allows systems to store and share their annotations in unified form. Annotations in the model consist of objects of three main types:

- *oac:Body* – represents an annotation's content, metadata that are stored in an annotation;
- *oac:Target* – represents an annotated document. An annotation can contain multiple targets;
- *oac:Constraint* – constrains an annotation's body or target to their parts. A constraint on a body defines a part of the body which is the real body of the annotation (e.g., a part of a web page in linking annotations). A target constraint specifies concrete part of the target, to which an annotation is anchored.

3. Knowledge Tags Repository

A core element of the knowledge tags maintenance approach is the knowledge tags repository, which straightly influences usability and a performance of the method. Additionally, the knowledge tags repository can be widely usable only if it implements flexible and generally acceptable knowledge tags model and provides effective and powerful access to non-trivial amount of stored knowledge tags.

We propose knowledge tags repository which stores knowledge tags in existing Open Annotation model, which is only in beta version, but it is already used by a numbers of systems and projects. The model is based on RDF and it is highly recommended to implement it by RDF triple databases and to support at least basic access (e.g., SPARQL querying) with RDF serialization output. But after standard use cases of annotation repositories and specific requirements of the maintenance method analysis we have found that a manipulation with whole knowledge tags and not only with their parts is main unit of almost all standard use cases. This is a consequence of the fact that knowledge tags have a sense only as complete information with knowledge tags' content and also their anchoring in tagged documents.

The finding is in disagreement with recommended RDF triple databases that have good deduction possibilities, but obtaining of complete information about an object requires processing of several simple queries and each query can take several seconds in large datasets [9]. To address this issue, we build efficient RDF-like repository for objects of one type (including types derived from this type, in our case *oac:Annotation*) which allows efficient access to complete objects and also supports basic SPARQL query processing with performance comparable to classical graph-based RDF stores.

We employ *MongoDB* which is in a correlation with our need of an access to whole knowledge tags, while it stores documents (objects) as one item and not sparse over several tables or collections. This allows fast access to whole objects without necessity of time expensive joins [11]. In addition, MongoDB provides efficient data access (loading and updating) and supports distributed data processing via *MapReduce* [3]. On the other hand, it does not provide support for SPARQL query processing.

MongoDB stores objects as BSON (binary-encoded serialization of JSON) documents and not as sets of RDF triples that are used by RDF databases, so we map these

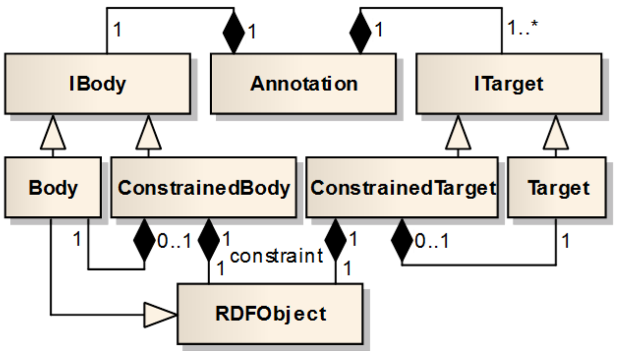

Figure 1: Structure of classes implemented in the knowledge tags repository. The core class *Annotation* represents RDF class *oac:Annotation* and stores knowledge tags. It aggregates one instance of *Body* or *ConstrainedBody* classes and at least one instance of *Target* or *ConstrainedTarget* classes. *RDFObject* class represents general RDF object. Its instances are employed as constraints.

RDF triples to instances of several specialized classes. In general we implement five specialized BSON classes for basic RDF classes (oac:Annotation, oac:Body, oac:Target, oac:ConstrainedBody, oac:ConstrainedTarget) and a class for general RDF classes (see Figure 1). The class specialization gives us possibility of more efficient data serialization and simplifies understanding of knowledge tags for new researchers.

As it is mentioned above, MondoDB supports distributed data processing via MapReduce and several approaches to SPARQL query processing via MapReduce [5][6] exists already, but they work with RDF triples databases and they are proposed for Apache Hadoop, which has some differences in processing of *Map* and *Reduce* phases. MongoDB additionally provides *Finalize* function for efficient finalizing results of reduce function. So we propose our algorithm for distributed SPARQL query processing [2], which is optimized for MongoDB and minimizes a count of necessary join operations to join results of all triple patterns in the query.

4. Evaluation and Conclusions

We evaluated proposed repository via a comparison of knowledge tags repository realisations based on MongoDB and based on Bigdata RDF triple database powered by NanoSparqlServer. We selected Bigdata because of its performance and horizontal scalability, what gives both realisations possibility to store non-trivial amount of knowledge tags.

We incrementally were loading one hundred of knowledge tags (each consists 16 RDF triples in OA model) anchored to one document during the evaluation and we measured times of a load, a retrieving of one knowledge tag by its URI, a retrieving URI list of knowledge tags anchored to a document, a retrieving of knowledge tags anchored to a document and an execution of simple SPARQL query with one join attribute. By a comparison of the measured values we find out that proposed knowledge tags repository is from 400 to 600 times faster than the repository based on Bigdata in use cases with whole knowledge tags manipulation. Less important operation, SPARQL query evaluation, took approximately same time.

Results of the preliminary performance evaluation of the repository indicates that proposed repository is more effective than classical RDF triple database four our use cases. But to the final proof we have to make evaluation with the repository deployed distributed over several nodes and we have to employ more RDF triple databases, including in-memory ones (e.g. Trinity or JPregel) that were not employed in preliminary evaluation, because they have good performance, but they have some issues with horizontal scalability and data persistence.

The final evaluation will be performed in a domain of PerConIK (Personalized Conveying of Information and Knowledge) project which is focused on enterprise application development support by viewing the software system as a web of information artifacts [1]. In the project several agents collect and process documentations, source codes, developer blogs, etc. and use knowledge tags for sharing their knowledge and collaboration activities.

Acknowledgements. This work was partially supported by the Slovak Research and Development Agency under the contract No. APVV-0233-10 and by the Scientific Grant Agency of Slovak Republic, grant No. VG1/0675/11.

References

[1] M. Bieliková, P. Návrat, M. Barla, J. Tvarožek, and M. Tvarožek. Collaborative programming: The way to "better" software. In *6th Workshop on Intelligent and Knowledge Oriented Tech.*, WIKT '11, pages 89–94, Košice, 2011. EQUILIBRIA, s.r.o. (in Slovak).

[2] M. Bieliková and K. Rástočný. Lightweight semantics over web information systems content employing knowledge tags. Lecture Notes in Computer Science. Springer Berlin / Heidelberg, 2011. (to be published).

[3] J. Dean and S. Ghemawat. Mapreduce: simplified data processing on large clusters. *Commun. ACM*, 51(1):107–113, Jan. 2008.

[4] J. Kahan and M.-R. Koivunen. Annotea: an open rdf infrastructure for shared web annotations. In *Proceedings of the 10th international conference on World Wide Web*, WWW '01, pages 623–632, New York, NY, USA, 2001. ACM.

[5] H. Kim, P. Ravindra, and K. Anyanwu. From sparql to mapreduce: The journey using a nested triplegroup algebra. *PVLDB*, 4(12):1426–1429, 2011.

[6] J. Myung, J. Yeon, and S.-g. Lee. Sparql basic graph pattern processing with iterative mapreduce. In *Proceedings of the 2010 Workshop on Massive Data Analytics on the Cloud*, MDAC '10, pages 6:1–6:6, New York, NY, USA, 2010. ACM.

[7] T. A. Phelps and R. Wilensky. Robust intra-document locations. *Comput. Netw.*, 33(1-6):105–118, June 2000.

[8] B. Plimmer, S. H.-H. Chang, M. Doshi, L. Laycock, and N. Seneviratne. iannotate: exploring multi-user ink annotation in web browsers. In *Proceedings of the Eleventh Australasian Conference on User Interface - Volume 106*, AUIC '10, pages 52–60, Darlinghurst, Australia, Australia, 2010. Australian Computer Society, Inc.

[9] R. Priest and B. Plimmer. Rca: experiences with an ide annotation tool. In *Proceedings of the 7th ACM SIGCHI New Zealand chapter's international conference on Computer-human interaction: design centered HCI*, CHINZ '06, pages 53–60, New York, NY, USA, 2006. ACM.

[10] R. Sanderson and H. Van de Sompel. Making web annotations persistent over time. In *Proceedings of the 10th annual joint conference on Digital libraries*, JCDL '10, pages 1–10, New York, NY, USA, 2010. ACM.

[11] S. Tiwari. *Professional NoSQL*. Wrox Programmer to Programmer. Wiley, 2011.

[12] M. Šimko. Automated acquisition of domain model for adaptive collaborative web-based learning. *Information Sciences and Technologies Bulletin of the ACM Slovakia*, 4(2), 2012.