Use cases for triple stores and graph databases in scalable data infrastructures

© Vasily Bunakov
Science and Technology Facilities Council, Harwell Oxfordshire, United Kingdom
vasily.bunakov@stfc.ac.uk

Abstract

Various types of NoSQL databases may present a sound alternative to relational databases in eInfrastructures that require managing and analysing data supplied from disparate sources. This work considers a few use cases where particular types of NoSQL databases – triple stores and graph databases – may be a natural choice for scalable data services. The purpose of this work is to brief on the experiments performed and to provide a roadmap for further technology evaluation. Cases suggested are mapped to EUDAT common services [11] yet should be of interest to other eInfrastructures.

1 Introduction

EUDAT project www.eudat.eu builds common data sharing, data preservation, data discovery and data analysis services for multidisciplinary and cross-border European research communities. Services under development or in their pilot phase now are:

- B2SHARE - a data publishing service,
- B2SAFE - a secure and reliable data replication service,
- B2FIND - a data discovery service,
- B2STAGE - a service for the delivery of data to high-performance computation,

with further services for semantic annotation, provenance and data retrieval that are under consideration.

There are indications that some mainstream platforms chosen for pilots of the aforementioned services during the previous phase of EUDAT, often underpinned by relational databases, may not be the best for scalability; also there are cases when having relational back-end means performing an excessive mapping in order to adopt certain metadata structures which more naturally fit into graph representation.

2 Use cases explained

2.1 Triple store as a back-end to data catalogue

The pilot version of B2FIND service relies on CKAN platform [7] which has an advantage of configurability and enjoys support of a thriving community of the platform adopters and software developers. Experiments showed though that CKAN, even after all the tuning recommended by the platform developers like switching off database triggers before bulk data upload, is not particularly powerful with data ingest. Taking in a few hundred thousand metadata records for B2FIND data catalogue using CKAN API could take over one full day [12]; this might present a problem in production environments with their needs of occasionally moving data, or restoring it from back-ups, or data replication.

Of course, it is possible to bypass CKAN API and ingest data records directly in a relational database that underpins CKAN instance, yet this solution defies the initial reason for choosing CKAN as a ready-to-use platform for the data catalogue. Also, although the metadata schema offered by CKAN fits the initial B2FIND requirements, it may present difficulties for expanding it with semantically meaningful links to richer metadata in domain-specific external repositories.

In search of alternatives to CKAN, EUDAT B2FIND looked into the results of triple stores evaluation performed earlier by Europeana project [13] and selected Jena TDB triple store [14] for experiments on scalability.

To populate the Jena TDB instance with data, RDF triples exported from CKAN B2FIND instance were used; this ensured that experimentation was done on the metadata of the same complexity. About 25 thousand unique RDF triples have been harvested with from B2FIND CKAN instance, then multiplied by factors of 10, 20, 30 and 40 to simulate EUDAT catalogue with 250K, 500K, 750K and 1M records respectively (so that some records – up to 40 of them – differed only by their IDs with all other attributes being the same). An average number of RDF triples per B2FIND record happened to be 33.3 so the test data resulted in corresponding RDF graphs of 8.5M, 17M, 25.5M and 34M triples.

The evaluation results obtained [1-2] showed higher performance of Jena TDB for data records ingest compared to CKAN. Jena TDB performance for data search was lower than CKAN when search requests required ordering of search results; when no ordering of search results was requested, Jena TDB demonstrated high performance on par with CKAN. This suggests that a triple store can be a competitive back-end for faceted search with graphs pre-indexed by certain attributes but less so in cases when search results ordering should be specifically defined by the user.

Overall, the experiments showed linear performance both for data ingest and data search in Jena TDB in the

Proceedings of the XVII International Conference «Data Analytics and Management in Data Intensive Domains» (DAMDID/RCDL'2015), Obninsk, Russia, October 13 - 16, 2015

range of up to 1 million B2FIND records that should be enough to satisfy the current needs of B2FIND.

CKAN still has the aforementioned advantage of high configurability with multiple ready-to-use modules developed by CKAN user community, so it remains the EUDAT B2FIND engine for the time being. Another B2FIND concern why a triple store might have disadvantage before CKAN was a need in a configurable user interface to the B2FIND back-end which CKAN can provide out-of-box. To meet this need of having a GUI to triple store, Elda platform [12] which is a Java implementation of Linked Data API Specification [16] was installed atop of the Jena TDB instance, and tested with EUDAT B2FIND data records. This provided a GUI and an ability to get search results in a variety of popular data formats: RDF Turtle, JSON, XML.

CKAN and triple store-based solutions are likely to co-exist in EUDAT B2FIND, with the decision about full service migration to a triple store back-end made later on depending on volumes of data records acquired, as well as further evaluated usability and performance of triple stores across a few instances of differently configured infrastructure, to clearly distinguish between effects of the infrastructure quality and performance of the database engines. The Figure 1 presents the current vision of RDF technology in EUDAT B2FIND technology stack.

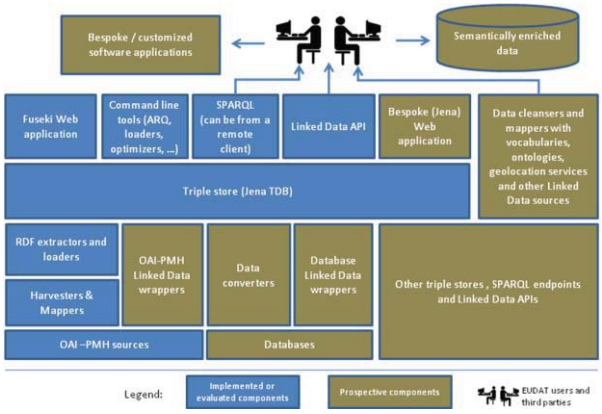

Fig. 1. EUDAT B2FIND technology stack.

Hence the current use case for a triple store in B2FIND is using it as a supplement to the existing data catalogue, to cater for machine agents that use SPARQL endpoints or LOD API in order to support third-party information services. These services can be data cleansing and enriching in spirit of "five stars" model of Web content quality [15], or mash-ups that mix EUDAT B2FIND triples with those for DBpedia entries and other established Linked Open Data sources.

A further development and a specific incentive for the adoption of triple stores and associated RDF technology in EUDAT B2FIND could be the exploration of a federated search using remote SPARQL endpoints, opposed to currently adopted stance of harvesting data records for a centralized B2FIND data catalogue. SPARQL allows mixing up requests to local stores with those to remote stores; this logical scalability as well as the actuality of data records retrieved from the source of their origin may prove attractive for certain EUDAT user communities and third-party software developers, even taking into account all physical communication overheads of sending requests to remote hosts.

Further experiments planned for B2FIND involve setting up the neo4j graph database [18] as a back-end to a triple store. The advantage of this can be using the same piece of infrastructure (scalable graph database) for more than one EUDAT service that, unlike B2FIND, may benefit from using native graph database methods (where B2FIND is likely to be interested in only RDF representation). Neo4j has been tried out on a relatively small (a few thousand) number of records from B2SHARE with good signs of scalability, so B2SHARE and B2FIND could be the good candidates for being backed by one graph database instance. The disadvantage of a graph database with a triple store built upon it could be lower performance, compared to a native triple store like Jena TDB. So thorough performance measuring is required, as well as balancing the benefits of a unified infrastructure (where graph database supports all EUDAT services) against higher requirements to the infrastructure that are potentially required for graph database.

Simplification of the technology stack when a triple store or a graph database can replace CKAN backed by a relational database will depend on further technology evaluation, software licensing considerations and business sustainability model chosen for EUDAT B2FIND service.

2.2 Data provenance and semantic logging

Another opportunity for the triple stores use in EUDAT is B2SAFE service that requires collection and management of data provenance records.

There are a few semantic models that can support data provenance use case in EUDAT: the group of PROV recommendations developed by W3C [20], Open Provenance Model [19], or CERIF with its semantic representation under way [6], as well as simpler models suitable for the definition of granular research activities, including related to data moves and transformations [3]. All these models can be supported by a triple store back-end with a potential need to get graph databases employed, too, as they can present additional means of naming and manipulating graphs so that a data provenance record can be clearly defined as a persistent chain (graph) of all actions performed over a particular dataset. An example of where a graph database may have an advantage before a triple store for the use case of data provenance is the quick extraction of provenance subgraphs and calculating their properties with graph-optimized algorithms.

A specific case of data provenance is data movement between EUDAT services. One of the scenarios could be User placing "long tail" research data in B2SHARE which automatically pushes it then to

B2SAFE for long-term preservation and to B2FIND to get it registered in a common data catalogue. The data then can be retrieved by request coming through B2SHARE, B2FIND or B2STAGE (for computation); a user or a machine agent that retrieved data may be interested in its origins, checksum and other parameters typically associated with the notion of provenance.

One way to achieve this is the construction of requests, using the dataset PID, to each of the EUDAT services involved, and the construction of a provenance chain on-the-fly; if built upon SPARQL endpoints or other sorts of APIs to EUDAT services, this will require sending requests to each of them. Another way is writing down the granular actions of data movement between EUDAT services in a log, and then building data provenance chains (graphs) upon information obtained from the log.

In the former case of on-the-fly inquiries about data provenance, a triple store-based engine sending federated requests to multiple EUDAT services will be more appropriate; in the latter case of producing permanent data provenance records, graph database may suit it better, perhaps accompanied by a triple store top-up for harnessing the power of the mentioned semantic models based on RDF technology.

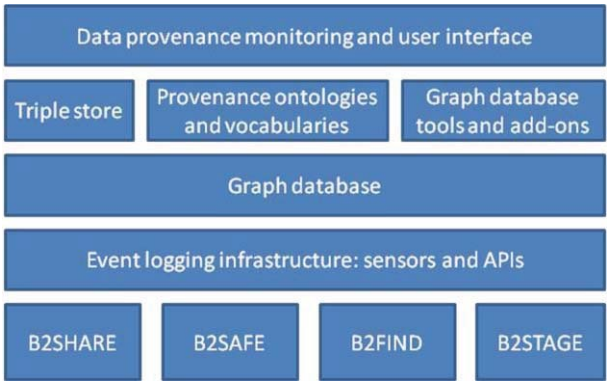

Fig. 2. Data provenance technology stack for EUDAT services.

Another reason for using graph databases can be the availability of mature open source frameworks for them such as TinkerPop [21] which can be harnessed for the development of middleware, so that even if a graph database may not present a data modeling advantage over a triple store, there may be technological considerations that make the choice of a graph database more reasonable. The technology stack for the hybrid data provenance platform is presented by the Figure 2. Data provenance can be seen as a special case of a more common use case of "semantic logging". Such big players as Microsoft started offering software development frameworks that allow sensible recording of events within software applications, and feeding these events into the event tracing services on an OS level (ETW – Event Tracing for Windows in Microsoft case). The back-end for capturing application-specific events can be a flat file, a relational database, or Azure Table Storage [17].

If one foresees any kind of machine reasoning over application events captured, then a graph database with a semantic top-up or a triple store can be a more natural choice than the mentioned back-ends; one of the key factors for the actual adoption of triple stores and graph databases in semantic logging will be their performance for capturing (writing down) application-specific and service-specific events.

An EUDAT candidate service for the adoption of semantic logging can be B2STAGE, so that all data supplied for high-performance computation as well as resulted from it are supplied with clear provenance and contextual information, for its inclusion in the events chain/network that is common with other EUDAT services. More candidates will be third-party services that use other EUDAT common services (B2FIND, B2SHARE) and are willing to share or mix up their internal event logs with those generated by EUDAT services.

2.3 Data retrieval via PIDs and semantic annotation

Using persistent data identifiers for data citation is becoming a commonplace across many research disciplines; there are good services that help researchers with minting data PIDs, e.g. DataCite [9] or CrossRef [8]. However, what is called "data" for the purposes of citation is highly specific to a particular research domain or actual practices of data centres that mint PIDs [4-5], so a reasonable idea to use data PIDs for automated data retrieval presents a real challenge.

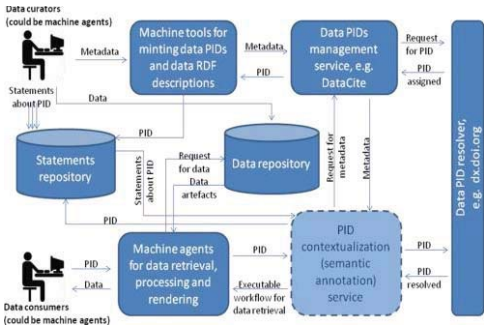

Fig. 3. A place of data PID semantic annotation service in PID curation and data retrieval workflow.

One of the possible responses to this challenge is using semantic annotation in order to describe the context of data PIDs including the actual protocol for data retrieval [5]. The Figure 3 presents a flow of information entities in data retrieval service based on data PIDs semantic annotation.

Triple store seems a natural choice to support this service, yet a hybrid solution that combines a graph database topped-up with triple store and semantic reasoner (in spirit of the Figure 2) may prove a more viable solution if we want persistence layer for data retrieval protocols (executable workflows) presented as identifiable graphs. Data retrieval via HTTP using data PIDs can be used by either B2STAGE service, or by third-party services built by EUDAT user communities who exploit other EUDAT services (B2FIND,

B2SHARE).

Semantic annotation of data PIDs is just one case for semantic annotation in EUDAT. There may be more cases suggested by EUDAT user community or discovered via EUON (European Ontology Network).

3 Conclusion

EUDAT started considering triple stores and graph databases in support of existing and emerging requirements of EUDAT infrastructure. The major use cases, with cross-links and generalizations between them outlined earlier, are:

- Massive migration of data records from B2FIND and B2SHARE data catalogues
- Machine access to B2FIND, with the ability of third parties to build their own information services supplementing the "mainstream" EUDAT B2FIND Web interface
- Federated search in B2FIND using requests to remote sources of data records, opposed to the current data records harvesting
- Data provenance within particular EUDAT services (B2SHARE, B2SAFE, B2STAGE) and across them
- Semantic logging in software applications, which can be used by EUDAT B2STAGE or by third-party applications calling other EUDAT services
- Data retrieval via data PIDs using semantic annotation, which can be used by EUDAT B2STAGE or by third-party applications calling other EUDAT services
- Other use cases for semantic annotation required by EUDAT services, or suggested by EUDAT user community, or identified through information practitioners' networks, specifically EUON

This work refers to already performed technology evaluation and aims to identify sensible use cases that contribute to the technology evaluation roadmap, with more experiments to be performed in support of existing and prospective eInfrastructure services.

Acknowledgements

This work is supported by funding from EUDAT project www.eudat.eu The author would like to thank his colleagues in EUDAT for their input although the views expressed are the views of the author and not necessarily of the project.

References

[1] Vasily Bunakov. Triple store evaluation. Presented in EUDAT project meeting, Amsterdam, Netherlands, 13-15 Jan 2014. http://purl.org/net/epubs/work/11477713

[2] Vasily Bunakov. Triple store testing on DKRZ virtual machine. EUDAT internal report, May 2014.

[3] Vasily Bunakov. Core semantic model for generic research activity. In 15th All-Russian Conference "Digital Libraries: Advanced Methods and Technologies, Digital Collections" (RCDL 2013), Yaroslavl, Russia, 14-17 Oct 2013, CEUR Workshop Proceedings (ISSN 1613-0073) 1108 (2013): 79-84.

[4] Vasily Bunakov. Investigation as a member of research discourse. In 16th All-Russian Conference "Digital Libraries: Advanced Methods and Technologies, Digital Collections", Dubna, Russia, 13-16 Oct 2014. CEUR Workshop Proceedings Vol-1297 (2014): 160-165.

[5] Vasily Bunakov. Service for data retrieval via persistent identifiers. In DATA 2015: 4th International Conference on Data Management Technologies and Applications. Colmar, Alsace, France, 20-22 July, 2015, pp.177-182.

[6] CERIF ontology. http://www.eurocris.org/ontologies/semcerif/1.3

[7] CKAN open source data portal software. http://ckan.org/

[8] CrossRef. http://www.crossref.org/

[9] DataCite. http://www.datacite.org/

[10] Elda: the linked-data API in Java. http://www.epimorphics.com/web/tools/elda.html

[11] EUDAT project. http://www.eudat.eu/

[12] Binyam Gebrekidan Gebre. CKAN evaluation. EUDAT internal report, September 2013.

[13] Bernhard Haslhofer, Elaheh Momeni, Bernhard Schandl, and Stefan Zander. Europeana RDF Store Report. The results of qualitative and quantitative study of existing RDF stores in the context of Europeana. March 2011. https://eprints.cs.univie.ac.at/2833/

[14] Jena TDB. https://jena.apache.org/documentation/tdb/

[15] Tim Berners-Lee. Is your Linked Open Data 5 Star? http://www.w3.org/DesignIssues/LinkedData.html

[16] Linked Data API Specification. https://code.google.com/p/linked-data-api/wiki/Specification

[17] Microsoft semantic logging: patterns & practices. https://msdn.microsoft.com/en-us/library/dn775006.aspx

[18] neo4j graph database. http://neo4j.com/

[19] Open Provenance Model. http://openprovenance.org/

[20] PROV-Overview. An Overview of the PROV Family of Documents. http://www.w3.org/TR/2012/WD-prov-overview-20121211/

[21] TinkerPop: An Open Source Graph Computing Framework. http://tinkerpop.incubator.apache.org/