Back to our Data —- Experiments with NoSQL Technologies in the Humanities

Tobias Blanke
Centre for e-Research
Department of Digital Humanities
King's College, London
Email: tobias.blanke@kcl.ac.uk

Michael Bryant
Centre for e-Research
Department of Digital Humanities
King's College, London
Email: michael.bryant@kcl.ac.uk

Mark Hedges
Centre for e-Research
Department of Digital Humanities
King's College, London
Email: mark.hedges@kcl.ac.uk

***Abstract*—In this short paper we discuss our work on developing a data infrastructure for Holocaust research. Faced with the challenge of integrating data with widely varying characteristics, we decided to pursue an approach based on the property graph model and corresponding graph databases. These provide intuitive modelling capabilities and the ability to fluently evolve in structure to meet the needs of the data but require more basic implementation work, as the technology is less mature.**

I. Introduction

The European Holocaust Research Infrastructure (EHRI) is a European Union funded Integrated Infrastructure project that aims to bring together collections and materials with researcher practices on the Holocaust. It includes over 20 partners, many of which are key archives and other collection holding institutions in the field, from Yad Vashem in Israel, to the Dutch NIOD, the Shoa Memorial in France and the Wiener Library in London. Details of the kind of research we want to enable and general requirements are described in [1]. This paper presents the technical approach we are taking to creating an environment that is both flexible enough to allow for the integration of heterogeneous material and that is sufficiently social to allow researchers to discover and analyse their material and make new connections.

One of the main components of the EHRI infrastructure will be an integrated research environment for Holocaust scholars to explore the data sets and collection descriptions that we provide. Our user requirements work has made it clear that while there are commonalities, such as easy and fast access to the material, the questions researchers are likely to ask of the data have great variability. Other work we have done in the field of arts and humanities e-Research with different kind of user groups supports this conclusion [2].

The requirement of a dynamically evolving research environment can be seen not just for arts and humanities research but across several other disciplines where the focus is on interacting with the research data. In fact, this focus on the ability to organise, analyse, and create exchanges around data has become one of the few general commonalities that has bound together disciplines in the various e-Science programmes. In 2008, Dave de Roure [3], one of the pioneers of the UK e-Science programme, consequently announced a new e-Science in a keynote at the annual IEEE e-Science meeting, which concentrated on enabling this new kind of interaction which data intensive research requires. We would like to suggest that the new e-Science needs to enable flexible interaction and collaboration with data. We need data infrastructures that can adapt to the new questions that e-Science wants to ask without losing data or making complicated changes to data formats and schemas. A fundamental feature of research applications is that it is hard to foresee what kind of data we need to integrate next, and that realistically we must expect it to be unique in its setup and formats. Holocaust studies works with a great variety of sources and our infrastructure needs to facilitate this. For EHRI, the technical design challenge was to innovate a dynamic, research-driven environment, where new material is constantly discovered, added and analysed.

One solution that we did not want to follow is to limit the amount of information we provide to a lowest common denominator. We started to investigate a new kind of infrastructure that would allow us to focus on the content as a researcher would need it and take the data as we found it, heterogeneous and often incomplete, from different sources but even in this form useful to the researcher.

Neither requirement fits well to traditional data stores such as relational databases, which typically require a predefined understanding of the underlying data structure which must be elaborately migrated to a new schema should a new understanding emerge. While relational databases are more than capable of storing extremely generic data structures, doing so is a trade-off against both ease of access and semantic coherence. We were keen to avoid the 'fallacy of prescience', the pitfall described by Smith, Deshaye and Stoicheff where initial assumptions limit what can later be done with the data [4].

This paper analyses our work with a new data infrastructure based around graph-based NoSQL technologies. Our initial experience has made us confident that NoSQL has significant potential to help mitigate some of the traditional challenges in our domain. In Section III, we analyse the technical design decisions to set up a graph-based data infrastructure. However, since many of these technologies are relatively new (in implementation, if not theory) their adoption still brings a range of challenges to be overcome, which we analyse in Section IV.

II. Background — NoSQL Graph Databases

Our work is based on our own research into how to integrate humanities data, from data grids [5] to Linked Data environments [6]. In this paper we present how the integration

978-1-4799-1293-3/13/$31.00 ©2013 IEEE

of historical data can benefit from graph databases, a mature technology often used within the social media domain [7]. While many NoSQL databases are concerned with the challenges of big data, graph databases address a use-case that traditional relational databases do not typically handle well; a larger number of heterogeneous records that are heavily interconnected in a free form manner [7].

Graph databases, however, are hardly used in e-Science or e-Research environments, with the exception of the life sciences, where [8] has successfully deployed graph databases to integrate data from several large data sets such as UniProt KB. Life science data is even more relationship-heavy than humanities data and [8] are approaching the benchmark of 1 billion relationships and 100 million nodes stored and processed in their database. Compared with the work in [8], this paper contributes a new use case for graph databases and a new understanding of the importance of its potential for other other research domains.

At present, the most common expression of the graph database data model is the *property graph*, which has three simple elements: nodes/vertices (V), edges that represent relationships between nodes (E) and properties that can be assigned either to nodes or relationships. In the course of our investigations, we found that our source components can be straightforwardly mapped onto this model. In EHRI, archival descriptions are modelled as nodes and connected via edges with other descriptions or their repositories, researcher annotations are nodes that can be linked to these descriptions and the material they refer to, and typical archival thesaurus terms are again nodes where the edges between them represent the typical narrower and broader relationships between thesaurus terms. As long as it can be mapped onto the node relationship model, any new data model can be added to the existing data without explicitly modifying a schema.

More common than graph databases in e-Research in general [9], as well as in the humanities and heritage domains in particular, is the use of triple stores in the context of semantic web technologies as discussed in [6]. Next to this work within humanities e-Research, there are the well-known efforts by large digital heritage providers such as Europeana or the Library of Congress to publish all their records as Open Linked Data. Whilst in EHRI we share a strong appreciation for the many advantages of semantic web technologies and the various standards (such as CIDOC-CRM) that build upon them, we felt that the graph databases model offered many compelling advantages over triple stores as an implementation medium, and additionally, possessed intriguing potential for enabling useful forms of access and exploration.

As analysed in [10], graph databases are more oriented to small- to medium-scale processing and analysis requirements, and because the property graph model makes a clear distinction between the intrinsic data belonging to individual entities (properties) and their external connections to other entities, the most meaningful data is pushed closer to the foreground. With this emphasis on relationships, graph databases are particularly well suited for historical research in particular and humanities research in general.

Our overall aim working with NoSQL technologies is to put researchers in touch with the data as they need it. Graph databases allow for exactly the kind of browsing activities and building relationships that are so typical to humanities research [6] and they allow to add new information dynamically and in the form most natural for the researcher. Graphs seem to model best how humanities researchers think about their data and its relationships, and in EHRI we felt that the property graph model struck a better balance at representing these structures in a pragmatic and intuitive manner than the low-level everything is a triple approach of RDF.

III. Technical Architecture

The EHRI technical architecture broadly consists of three components: a registry, an administration interface, and a user-facing search and annotation front-end. The registry also contains a significant amount of business logic that enables EHRI's role-based access and data management capabilities, as well as its rich annotation and entity-linking features. Placing such generic business logic within the database itself allows us to deal with the complex data management challenges in a highly efficient manner that greatly simplifies the implementation of client applications. Figure 1 summarises the architecture. It shows the architectural layers that make use of our graph databases and the other data stores we use. In this section, we concentrate on our added features and design decisions to enable efficient graph database processing of archival information, i.e. how we firstly model archival information as graphs, how we secondly make them persistent and thirdly how we integrate user interaction with the archival data.

The most important decision we had to make was how to model archival information as a property graph and how to enable the kind of dynamic data-driven environment we describe above. As with other attempts to mediate archival standards such as ISAD(G) with broader ontological conceptual models such as CIDOC-CRM, we recognised a clear distinction between structural and what [11] refer to as generic elements. The EHRI registry only concerns itself with structural elements in the underlying archival data, and places no restrictions on what information can be attached to graph entities.

We learned early on in the project that being able to easily alter the attribute-level schema of domain entities was a key requirement to this kind of environment, since the data management needs of our archival survey team, for instance, were difficult to graft into existing tools without making backwards incompatible changes to their database schemas that would make ongoing maintenance a burden. A database technology that incorporated property-level flexibility into its design was, through this experience, of much greater importance to us.

Making changes to the data held by domain entities is therefore highly flexible, and can be altered without making any schema-level modifications to the registry itself. This is achieved by using the aforementioned graph database, for which we have chosen the popular and well-supported Neo4j database.[1] The database itself is, for the large majority of operations, interfaced with via an abstraction layer called Blueprints,[2] which provides access to common primitive operations such as creating vertices (graph nodes), creating edges

[1] http://www.neo4j.org

[2] http://www.tinkerpop.com

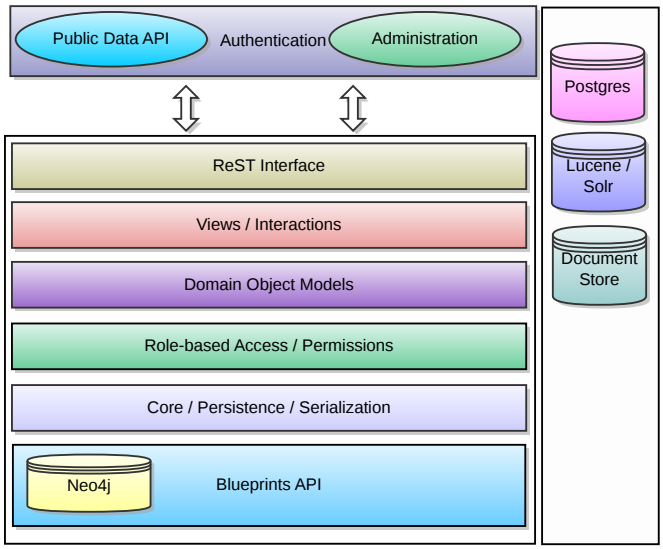

Fig. 1. *The EHRI registry multi-tier architecture.*

(the relationships between them), and traversing the graph. In addition to providing a number of useful tools for interfacing with the graph, such as a powerful traversal API named Gremlin, Blueprints allows us to separate our domain-specific logic from the underlying database engine, and, if necessary, replace it with one of several alternatives.

As a second feature to enable efficient processing of archival information, the EHRI persistence layer deals with subtree data structures. The purpose of this special handling for subtrees, as opposed to more free-form sub-graphs, is that the entities that EHRI handles – for example, archival descriptions of documentary units – are translated not to a single graph node but to a composite structure comprised of several nodes. In fact, we consider it to be one of the major advantages of the graph database that there is such a close relationship of its own data model with the hierarchical organisation of information that is typical to archival descriptions. In relational databases, these objects would reside in several tables linked by foreign keys, with constraints that automatically delete the subordinate rows when the parent row is itself deleted. Being able to create, update, and delete these composite subtree data structures is an important feature of the EHRI registry architecture.

Since putting the user in touch with the data was a primary objective in our system, the third design feature to deal specially with archival information is the 'views' layer. It models the interactions between users and groups in the system (also modelled as nodes) and data objects and is special as it can take full advantage of the graph database's fluency at traversing hierarchical relationships. Users in our system can belong to an arbitrary number of groups, and groups themselves can also be nested within other groups. Access controls and permissions for data objects also have a hierarchical structure, mirroring the manner that non-virtual archives organise information.

The graph model comes into its own when integrating the relationships between researchers and their interactions with the data itself — through making new connections, comments, and annotations. EHRI differs from many other archival aggregation projects in that it is explicitly focussed on allowing researchers themselves to create what they may feel are relevant connections between archival material, in a manner which may be private, shared with a limited audience, or public. Users can 'follow' the public activity of other users with whom they may share common research interests, and collaborate with them in groups. The views layer in the EHRI architecture mediates a researcher's perspective on the underlying archival data with their social connections, allowing them to view contributions from other user's they may trust, or with whom they have a collaborative relationship.

A ReST interface, finally, is the primary entry point of the registry and the last feature we would like to discuss here. It provides two key pieces of EHRI-specific functionality. Firstly, while Neo4j's native ReST interface deals only in graph primitives (nodes and relationships), our ReST extension handles the subtree data structures described above, that encapsulate composite objects. This allows us to create, update and delete composite objects in a transactional, idempotent manner that either succeeds completely, or leaves the graph unchanged.

Secondly, the EHRI ReST extension provides a way of accessing and serialising customisable sub-graphs of data in a single HTTP request. For example, an archival description stored within EHRI's registry would typically comprise at least three nodes: the logical object itself, the description of the logical object (of which there can be several) and temporal metadata attached to the description (dates of creation or aggregation, which are stored as separate nodes.) Using the graph database's basic ReST interface directly to gather this information would take upwards of seven separate HTTP requests. EHRI's ReST interface uses metadata provided in the domain layer to automatically traverse and serialise nodes that we know belong together, and form a logical composite whole, into a single sub-graph data structure.

Whilst the ReST interface is designed to simplify the process of accessing the composite entities that it contains, it is not intended to be the sole entry-point to the graph. On the contrary, trusted users have full access to a raft of other graph traversal mechanisms provided by both the Blueprints layer, and the Neo4j database we currently build upon. This includes the aforementioned Gremlin tool, which can be seen as an imperative language for exploring graph structures with lazy semantics, and a powerful declarative graph traversal language called Cypher that is integrated into Neo4j.

These technologies give us great flexibility to 'walk' the graph and explore the registry data in useful ways. For example, using Cypher's 'shortest path' algorithm, we can trivially determine if two users have interacted with any of the same items, by white-listing a set of relationships for the algorithm to traverse until it either finds one 'user' node from another, or runs out of relationships. We plan to leverage these powerful traversal and exploration mechanisms to complement the EHRI portal's search capabilities with additional serendipitous discovery behaviours.

IV. Discussion — Advantages and Disadvantages

One of the biggest advantages of using graph databases for working with archival material is the natural fit with the domain data. In our experience, access patterns that are optimised for open-ended traversal greatly simplify the experience of dealing with deeply nested hierarchical structures. In Section III, we described how the EHRI registry's ReST interface handles the automatic serialisation of certain key relationships modelled within our domain. One of the relationships that is automatically serialised is that between an item and its parent, for example an archival description representing a particular series, and its parent fonds. Because we can infer that the number of ancestors a given hierarchically-positioned item has will not be unbounded, we know it is safe to automatically serialise an item's parent, and its parent's parent, up until the top level.

Retrieving the full context to which a documentary unit belongs requires therefore only a relatively trivial recursive traversal through the parent item nodes, likewise gathering their contextual metadata in turn. Open-ended traversals of this kind are what graph databases such as Neo4j are optimised for and is a central feature of graph query languages such as Gremlin and Cypher. Not only are these actions therefore fast, but, almost as importantly, they do not require complicating the data model realization in the manner that relational database implementations, such as the adjacency list and nested set patterns, tend to do. As described in III, these benefits extend to many other aspects of the EHRI registry, such as modelling of SKOS concept trees and role-based access control.

While we found graph databases very useful for working in our own domain, which is primarily concerned with relationship-heavy metadata, they are less well suited for systems which deal with stable and potentially long-term persistence of research data. We recognise that a polyglot persistence approach, leveraging multiple solutions each suited for different aspects of how the data is used by individual applications, have an important role to play in current and future digital research infrastructures. Although the graph is its central component, the full EHRI stack makes use of multiple persistence solutions: an Postgres database for user authentication data, Apache Lucene for multilingual indexing, and a document store for harvested XML. While our graph database is capable of exposing its underlying data as RDF and can be queried via SPARQL, we are considering utilizing a dedicated triple store to enable additional kinds of large-scale consistency reasoning and analytics, as well as the publication of EHRI data sets as Linked Data.

It has to finally be acknowledged that at the present time the technological ecosystem around graph databases is still considerably less mature than that of more established persistence solutions. For EHRI, this has meant more time spent dealing with the 'plumbing' of applications (such as access control) that could have been expended elsewhere (although, conversely, we have been able to create a system that fully leverages the advantages of the graph database.)

Further details on the research are described in [12], where we also present how we use an external SOLR that stores the textual and linguistic properties of archival description data index to retrieve relevant collections for the portal. Retrieving the full context of the documentary unit can then be achieved by walking through the graph from there. In terms of computational complexity this traversal is constant (O(N)) and independent of any new relationship that has been added to the graph outside the context of the documentary unit.

V. Conclusion

The paper presented our work on a dynamic data infrastructure for research on Holocaust data. Rather than relying on more established solutions such as relational databases and semantic web environments, we experimented with graph databases. Though these are relatively technically mature by now, they are still not widely used in research infrastructures. We were rewarded with a rich flexibility and ability in working with the very heterogenous data that we are dealing with in Holocaust research. On the downside, we needed to do more basic implementation work, but we expect this to be less of a problem once graph databases and other NoSQL persistence solutions are more established within and beyond research applications.

References

[1] R. Speck and P. Links, "The missing voice: Archivists and infrastructures for humanities research," *International Journal of Arts and Humanities Computing*, Forthcoming.

[2] T. Blanke and M. Hedges, "Scholarly primitives: Building institutional infrastructure for humanities e-science," *Future Generation Computer Systems*, vol. 29, no. 2, pp. 654–661, 2013.

[3] D. de Roure, "The new e-science," http://www.slideshare.net/dder/the-new-science-bangalore-edition, December 2008.

[4] J. Smith, J. Deshaye, and P. Stoicheff, "Callimachusavoiding the pitfalls of xml for collaborative text analysis," *Literary and linguistic computing*, vol. 21, no. 2, pp. 199–218, 2006.

[5] M. Jackson, M. Antonioletti, A. Hume, T. Blanke, G. Bodard, M. Hedges, and S. Rajbhandari, "Building bridges between islands of data-an investigation into distributed data management in the humanities," in *e-Science, 2009. e-Science'09. Fifth IEEE International Conference on*. IEEE, 2009, pp. 33–39.

[6] T. Blanke, G. Bodard, M. Bryant, S. Dunn, M. Hedges, M. Jackson, and D. Scott, "Linked data for humanities researchthe spqr experiment," in *Digital Ecosystems Technologies (DEST), 2012 6th IEEE International Conference on*. IEEE, 2012.

[7] P. J. Sadalage and M. Fowler, *NoSQL Distilled: A Brief Guide to the Emerging World of Polyglot Persistence*. Addison-Wesley Professional, 2012.

[8] P. P. Tobes, "Bio — data graph db," http://bio4j.com/, Feb 2013.

[9] K. OHara, T. Berners-Lee, W. Hall, and N. Shadbolt, "4.3 use of the semantic web in e-research," *World Wide Research: Reshaping the Sciences and Humanities*, p. 130, 2010.

[10] M. Rodriguez, "On graph computing," http://markorodriguez.com/2013/01/09/on-graph-computing/, Feb 2013.

[11] M. Theodoridou and M. Doerr, "Mapping of the encoded archival description dtd element set to the cidoc crm," *FORTH-ICS Technical Report*, vol. 289, 2001.

[12] T. Blanke and C. Kristel, "Integrating holocaust research," *International Journal of Arts and Humanities Computing*, Forthcoming.