A Minor Prototype of Personal Dataspace Management System

Tanvi Shree
Asst.Prof IT Dept
IMSEC Ghaziabad
tnvshree@gmail.com

ABSTRACT

A personal dataspace includes all personal data of a user. This personal data is generally heterogeneous, rough and tumble in nature which is distributed over various data sources. So, a single system (Personal Data Space Management System) is needed to manage this data. In this paper, we have presented a prototype of Personal Dataspace Management System which is built on the vision presented in [1,2,3].It represents data from different sources uniformly into triples that are similar to RDF. This paper presents the architecture of PDSMS, a MongoDB to SPARQL conversion tool and performance of MongoDB as a persistent storage.

Keywords

Dataspace, personal dataspace management system, triple model, triple store, MongoDB.

1. INTRODUCTION

An explosively increment in the amount of personal data and the limited time and capability of people has evolved PDSMS (personal dataspace management system). It is built on the visions of dataspace presented in[1,2,3] which advocates dataspaces as a new abstraction for information management that keeps heterogeneous data from heterogeneous sources in pay-as-you-go fashion.

A PDSMS stores, manages personal data, and provides a logical query mechanism. Personal data is all data of a user keeps like data on desktop, laptop, PDA, mobile phones, camera, email, network drives, webserver, etc. These are highly heterogeneous data consist of files, email, music, pictures, calendar data, MS excel and so on.

The main issue regarding designing of a PDSMS is "how to represent personal data", i.e. data modeling. Various data models like idm, Triple model[5]etc. are present to represent data in a PDSMS.

A Triple model is a flexible and simple way to represent personal data. It is based on RDF but is more suitable to dataspace applications. Information from heterogeneous data sources are decomposed into chunks of data unit with the help of wrappers. These chunks of data units are represented by independent data units called triples. A triple composes of Subject, Predicate, and Object.

To provide a system with persistent storage, triples are stored in triple store. Triple store is a database used for storing and querying triples. There are various triple stores available in market which are purposely built for triple store like sesame, mulgara etc. But existing databases like mySQL and Oracle are also being used as triple store.

This paper presents a minor prototype of PDSMS that uses MongoDB[16] as triple store. MongoDB is a noSQL, document oriented database. MongoDB use light weighted BSON (binary JSON) documents. In JSON, data is represented as key-value pair. It is schema-less, provides scalability to the model. MongoDB bridges the gap between key-value stores (which are fast and scalable) and relational database (which have rich functionality).

2. SYSTEM OVERVIEW

In this section, we discuss the architecture of PDSMS that we have implemented. A PDSMS comprises of many components, but the components that are covered in this system are- data extraction, data modeling, data storage and query execution. The figure given below describes architecture of this system.

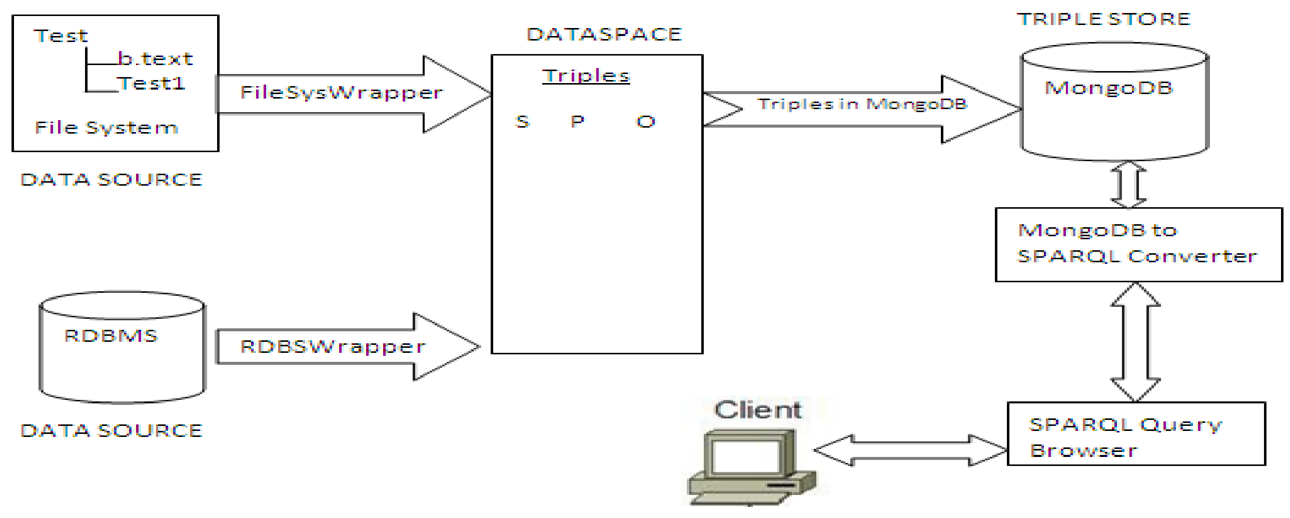

Figure 1: Architecture of the system

2.1 System Components

2.1.1 Data Source

Data sources of personal data are heterogeneous i.e. a similar domain data exists at physically separated sources or on a single data source but in different schemas.

2.1.2 Wrappers

Wrappers are constructed corresponding to each data model. It extracts useful information from datasource and decomposes it into smallest autonomous unit of information. Wrappers are constructed by encoding DRS (Decomposition rules set) into it. A DRS is defined as a set of rules built to decompose data into small data units for specific data source.

2.1.3 Dataspace using Triples

PleaseDataspace is that space in which information from different data sources can be uniformly represented. Data from different sources co-exist instead of integrated. In our system, data from different sources are represented uniformly in dataspace in triples, all triples are loosely connected without reconciling the semantic heterogeneity.

A triple T is a 3-tuple (S, P, O), where S is Subject component, P is Predicate component, and O is Object component. We define each component of triple as follows:

S is an integer, which is the id of an item.

P is a 2-tuple (l, p), where l is a finite string that represents the label, and p is also a finite string that represents the data type.

O is an array of bytes, which stores the data.

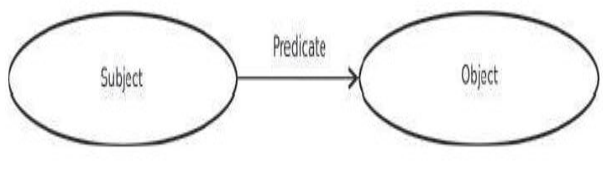

Figure 2: A Triple

Following syntax is used to create a model and insert triples into it. Triples can be viewed as resources in jena.

```
// create an empty graph
model = ModelFactory.createDefaultModel();

// create the resource
r = model.createResource();

// add the property
r.addProperty(RDFS.label, "11")
.addLiteral(RDFS.label, 11);

// write out the graph
model.write( System.out);
}
```

Figure 3: Create a Model and insert Triples using Jena.

A graphical representation of model can be generated using a tool "RDF gravity" which gives output as in figure 4.

2.1.4 MongoDB as a Triple Store

For persistent storage, triples are being stored in some database, which is known as triple store. In this system, we have stored triples in MongoDB which is a noSQL, document oriented database. MongoDB use light weighted BSON (binary JSON) documents. In JSON, data is represented as key-value pair.

Document-oriented key-value stores such as MongoDB are different than triple stores. The key-value store consists of two terms: keys and values, the triple store consists of three terms: subjects, predicates and objects. It is difficult to map a key-value pairs to RDF triples.

In this system, we have stored triple inside a document, each document contain a triple as shown in figure 5.

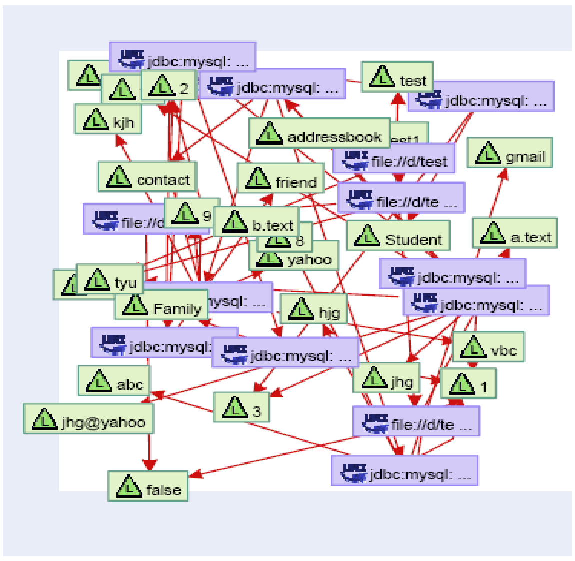

Figure 4: Graphical Representation of Triples

{ subject:"_:a",

Predicate:"<http://xmlns.com/foaf/0.1/name>",

Figure 5: Representing a Triple in MongoDB

These documents are residing inside a single collection. All triple of a graph in jena are corresponding to documents of a collection.

2.1.5 Query Converter from SPARQL to MongoDB

It is proposed to introduce a triple store based on document-oriented MongoDB. RDF has different query languages. The official and most widely used is the SPARQL [25] which is explained in section.

MongoDB uses own query syntax and do not support SPARQL. Mongo Query Language is an object-oriented imperative language. SPARQL is a domain-specific declarative language. Therefore, it is difficult to replace the SPARQL to MongoDB Query Language. Therefore, we need to design and implement a tool for SPARQL to MongoDB query language conversion. The tool first receives SPARQL query which is parsed, analyzed and then converted into Mongodb Query language. After executing the mongo query, we return back the result to the initial SPARQL query. The working of the converter is shown in figure 6.

2.1.5.1 SPARQL Processor
It provides an abstract view by hiding all the complexity within queryProvcessor() method. It converts SPARQL query into XML and then passed it to the next component.

2.1.5.2 SPARQL Parser
A SPARQL parser parses the SPARQL query by using the ARQ module [4] of Jena framework [5] which although constitute a complete system supporting RDF storage and SPARQL execution, I use only their SPARQL parsing function.

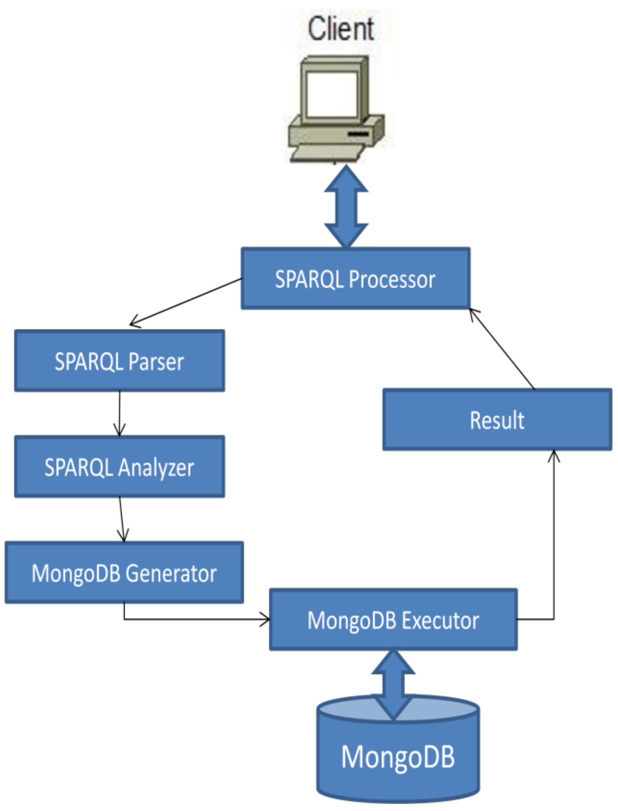

Figure 6: SPARQL - MongoDB Converter

2.1.5.3 SPARQL Analyzer
This component takes the SPARQL query parsed earlier and traverses its object representation .During this process a new data structure is built to represent the query.

2.1.5.4 MongoDBquery languageGenerator Analyzer
This component is responsible for producing a MongoDB query based on the SPARQL query and on selected RDF to DB mapping.

2.1.5.5 SPARQL Query Browser
This Query browser provides a GUI to the user where one can enter query in SPARQL and gets result back. It hides complexity of Triple store from user.

3. ASSESSING PERFORMANCE
For persistent storage, we use MongoDB as triple store. To assess performance of MongoDB, MongoDB is compared to MySQL i.e. triples are also stored in MySQL. Jena support storage of triples in MySQL and also contains SPARQL to SQL converter.

I take one database “Addressbook” from MySQL and two folders from File system for simplicity.
Total 130 triples are generated which are stored in MongoDB and MySQL databases. The table below shows the load time (time taken to write the data to MongoDB and MySQL).

Table 1. Load Time

Rounds	MongoDB(ms)	MySQL(ms)
1.	1828	6297
2.	1234	5594
3.	750	5609
4.	890	6437
5.	750	6282

ForAs we can see, MongoDB loads about 164 triples in one second, where as MySQL loads about 6 to 7 triples in one second.Once I had these datasets loaded I then set up a number of SPARQL endpoints and ran a SPARQL query on triples stored in MongoDB, MySQL and a RDF file stored in system and achieved the following results:

Table 2 .Query Execution Time

Round	MySQL	In-memory	MongoDB
1.	1688	1578	1469
2.	1687	1547	1297
3.	1718	1578	1312
4.	1719	1546	1297
5.	1703	1578	1281

MongoDB takes least time than MySQL and File to execute a query. The reason behind the better performance of MongoDB from MySQL is that MongoDB does not require joins like in case of relational database MySQL.

4. CONCLUSION

In this paper, we have tried to build a prototype of Personal Dataspace Management Systems (PDSMS). It is built on the bases of vision presented personal dataspace [1, 2,3]. This dissertation has advocated the working of MongoDB as a triple store. By including MongoDB as triple store, its advantages of schema-less, document oriented database and using light weighted JSONdocument have been optimally exploited. As we can see, performance of MongoDB is better than MySQL in both load time and query execution. Therefore, MongoDB can be used as triple store. This system can be amended by including features like data integration and pay-as-you-go.

5. REFERENCES

[1] Halevy A , Franklin M and Maier D.” Principles of dataspace systems”, In Proceedings of the twenty-fifth ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database System(2006).

[2] Franklin M, Halevy A, and Maier D.” From databases to dataspaces: A New Abstraction for Information Management”, ACMSIGMOD Record, 34(4):27-33, 2005.

[3] Y. Li and X. Meng.” Research on Personal Dataspace Management”,ACM SIGMOD phd workshop on innovative Database Research(IDAR 2008), june 13 ,2008.

[4] Dong X,”Providing Best-Effort Services in Dataspace Systems”, Phd Thesis, University of Washington, 2007.

[5] William Jones and Jaime Teevan.”Personal information management”, University of Washington Press, Seattle, WA, 2007.

[6] Ming Zhong , Mengchi Liu and Qian Chen.”Modeling Heterogeneous Data in Dataspace” IEEE IRI 2008, july 13-15 2008,Las Vegas, Nevada USA.

[7] http://jena.apache.org/

[8] http://en.wikipedia.org/wiki/Triplestore

[9] Dominik Tomaszuk.”Document-Oriented Triplestore based on RDF/JSON”, Studies in Logic, Grammar and Rhetoric year: 2010, vol: , number: 22(35), pages: 125-140.

[10] ARQ-A SPARQL Processor for Jena, http://jena.sourceforge.net/ARQ/

[11] Jena – A Semantic Web Framework for Java, http://jena.sourceforge.net/

[12] X.Dong, A Halvey, E. Nemes,S.B. Sigurdsson, and P Domingos, “SEMEX: Toward On-the-fly Personal Information Integration”, In IIEWB 2004.

[13] Dong X, Halvey A ”A Platform for personal information management and integration”, Proceedings of CIDR 2005, pp 119-130.

[14] J. Dittrich, L Blunschi and M. Farber,”From Personal Desktops to Personal Dataspaces: A Report on Building the iMeMex Personal Dataspace Management System”, conference on “Database System for Business, Technology and Web”, pp 292-308.

[15] J.-P. Dittrich,” imemex: A platform for personal dataspace management”, In SIGIR PIM workshop, 2006.

[16] K. Chodorow. MongoDB manual. 10gen. 2009.

[17] Steve Francia, ”MongoDB and PHP”,O’Reilly.

[18] Elmasri R, Navathe S B, ”Fundaments of Database Systems”, 5th edition, Pearson Education, 2008.

[19] Googledesktop search toolkit. http://desktop.google.com/, 2004.

[20] Yahoo! Desktop Search. http://desktop.yahoo.com/, 2007.

[21] Windows Desktop Search. http://www.microsoft.com/windows/desktopsearch/default.mspx, 2007.

[22] F. Garzotto, P. Paolini, and D. Schwabe. Hdma model-based approach to hypertext application design. ACM Trans. Inf. Syst., 11(1):1–26, 1993.

[23] W3C. Rdf. http://www.w3.org/TR/rdf-concepts/.

[24] W. Jones and H. Bruce. A report on the nsf-sponsored workshop on personal information management. In NSF PIM Workshop, Seattle, January 2005.

[25] W3C. Sparql. http://www.w3.org/TR/rdf-sparql-query/.

[26] E. Prud’hommeaux and A Seaborne, ”SPARQL Query Language for RDF”, World Wide Web Consortium, 2008.

[27] D. D. Chamberlin and R. F. Boyce, ”ISO/IEC 9075:1992 – Database Language SQL”, International Organization for Standardization, 1992.

[28] J. J. Carroll, I. Dickinson, C. Dollin, D. Reynolds, A. Seaborne and K. Wilkinson,” Jena: implementing the semantic web recommendations”, International World Wide Web Conference, Proceedings of the 13th international World Wide Web conference on Alternate track papers & posters, 2004.

[29] D. Beckett, “RDF/XML Syntax Specification (Revised)”, WorldWideWeb Consortium, 2004.

[30] D. Beckett, T. Berners-Lee, “Turtle – Terse RDF Triple Language”, World Wide Web Consortium, 2008.

[31] T. Berners-Lee,” Notation3”, World Wide Web Consortium, 2006.

[32] J. Broekstra, A. Kampman and F. van Harmelen, ”Sesame: A Generic Architecture for Storing and Querying RDF and RDF Schema”, The Semantic Web ISWC 2002, Springer, 2002.

[33] R. Oldakowski, C. Bizer, D. Westphal,” RAP: RDF API for PHP”, In Proceedings International Workshop on Interpreted Languages, MIT Press, 2004.

[34] S. Harris, N. Lamb and N. Shadbolt,"4store: The Design and Implementation of a Clustered RDF Store", The 5th InternationalWorkshop on Scalable Semantic Web Knowledge Base Systems, 2009.

[35] A. Muys," Building an Enterprise Scale Database for RDF Data".

[36] A Fnewman," Getting Started with JRDF", 2010.

[37] A. Newman, J. Hunter, Y. Li, C. Bouton and M Davis," A Scale-Out RDF Molecule Store for Distributed Processing of Biomedical Data", Semantic Web for Health Care and Life Sciences Workshop, 2008.

[38] J. Aasman, "Allegro Graph: RDF Triple Database", Technical Report 1, Franz Incorporated, 2006.