An Evaluation Model for Semantic Enablement of Virtual Research Environments

Tristan O'Neill, Trina Myers, and Jarrod Trevathan

***Abstract*—**The *Tropical Data Hub* (TDH) is a virtual research environment that provides researchers with an e-research infrastructure to congregate significant tropical data sets for data reuse, integration, searching, and correlation. However, researchers often require data and metadata synthesis across disciplines for cross-domain analyses and knowledge discovery. A triplestore offers a semantic layer to achieve a more intelligent method of search to support the synthesis requirements by automating latent linkages in the data and metadata. Presently, the benchmarks to aid the decision of which triplestore is best suited for use in an application environment like the TDH are limited to performance. This paper describes a new evaluation tool developed to analyze both features and performance. The tool comprises a weighted decision matrix to evaluate the interoperability, functionality, performance, and support availability of a range of integrated and native triplestores to rank them according to requirements of the TDH.

***Keywords*—**Virtual research environment, Semantic Web, performance analysis, tropical data hub.

I. Introduction

THE *Tropical Data Hub* (TDH) is a *Virtual Research Environment* (VRE) for the collaborative collection, management and reuse of research data [1]. The TDH is designed to complement existing data repositories while providing researchers with a single virtual location for research data from tropical regions. The hub provides researchers, managers and decision-makers with access to an extensive amount of data from disparate data sources for a more accurate holistic view of the current state of the tropics. This holistic view is possible with a cross-disciplinary "horizontal" approach rather than the "vertical" paradigm of research silos. Horizontal research spans a cross-connect through disciplines, research methods, data resources and experimental techniques to enable synthesis of a diverse range of disciplines and data. A semantic layer incorporated into the TDH would enable this data linkage ability between internal and external data and metadata.

Semantic Web (SW) technologies allow for a flexible scalable environment to model abstract and concrete concepts in a way that is "understandable" to the machine [2]. Ontologies are the basis of SW technologies and can be defined using the *Resource Description Framework* (RDF) and the *Web Ontology Language* (OWL) [3]. Ontologies provide the means to describe real world concepts by well-defined descriptions to automatically make latent connections between entities [3]. Ontologies are formed using triples (subject-predicate-object) written in RDF and OWL to form the structure of a triplestore.

A semantic triplestore is a purpose-built knowledge base for the storage and retrieval of triples [4]. It presents the opportunity to provide researchers with a new level of data and metadata storage and retrieval technology. The triplestore is a SW technology, which allows researchers to expand their research questions into different disciplines and across disparate data sources. Traditional information retrieval techniques are inadequate because they are mainly based on the keyword search, not on the contextual information of the search used in semantic inference [5, 6]. Currently, there are numerous RDF triplestores available with differing levels of functionality, supported features and stages of development.

This paper proposes an *evaluation model* that identifies and compares each triplestores functions, features and performance components that influence the decision of which is best to implement. A complex *Weighted Decision Matrix* (WDM) comprises the evaluation model [7]. The context for these comparisons is in consideration when applying a semantic layer to a data portal such as the TDH. The requirements of a triplestore for the TDH include the categories interoperability, reasoning and inference functionality, performance and support. Here, the most current triplestore versions were analyzed and compared using an analytical approach to determine a viable candidate. Specific criteria that were tested included: minimal maintenance; multi-disciplinary queries; disparate data source queries; distinct data storage platforms; timely query responses; timely reasoning response; and accuracy in output. This approach has produced a decision matrix based on the specific criteria and the prioritization of the TDH requirements.

This paper is organized as follows: Section II describes the problem motivation and research goals. Section III details the methodology for the study. Section IV presents an analysis and results of the triple stores evaluated and Section V discusses the results. Section VI offers some concluding remarks and avenues for future work.

II. Background and Considerations of Semantic Enablement

The TDH is being developed as a platform to store, aggregate, selectively process and serve significant tropical

T.O'Neill and T. Myers are with the School of Business (Information Technology), James Cook University, Townsville, Queensland, 4811, Australia (phone: 6107 4781 6908; e-mail: Tristan.Oneill@my.jcu.edu.au, Trina.Myers@jcu.edu.au).

J. Trevathan is with the School of Information and Communication Technology, Griffith University, Brisbane, Queensland, 4111, Australia (phone: 6107 3735 5046; e-mail: j.trevathan@griffith.edu.au).

data sets in an open collaborative environment. The TDH philosophy is to span traditional "vertical" research disciplines and enable "horizontal" research. Specifically, vertical research is the traditional discipline and data-specific research paradigms are conceptual silos of concentrated research efforts. In contrast, horizontal research spans a cross-connect through disciplines to enable analysis and synthesis of the available, yet disparate data.

The prevailing vertical method of research correlation and analysis requires the researcher to submit queries to multiple data stores independently to obtain results. The results need to be analyzed independently to determine the potential probability of an event occurring. This event may be migration patterns or purchasing trends. These results may then be the basis of another research question, which would then need to be compared or queried against another data set. For example, to find out the average rainfall for towns in North Queensland, Australia that consisted of more than 10,000 residents one query to resolve this would not currently be possible as the data stores for sociological growth and meteorological analysis are not linked. The use of triples dictates how the data from disparate data stores relate to each other.

Currently, the TDH does not support automated "linkage" between data stored inside the repository to achieve this cross-connect between data and metadata. When data is linked, hidden connections between related data, people and processes can be automatically revealed. Semantic technologies can automate linkages between TDH data sets and metadata and make possible intelligent searching and alerting (Fig. 1). The potential benefits of these data linkages include the discovery of potential collaborative partners or organizations and the discovery of, and connection to, open data sets external to the TDH [1].

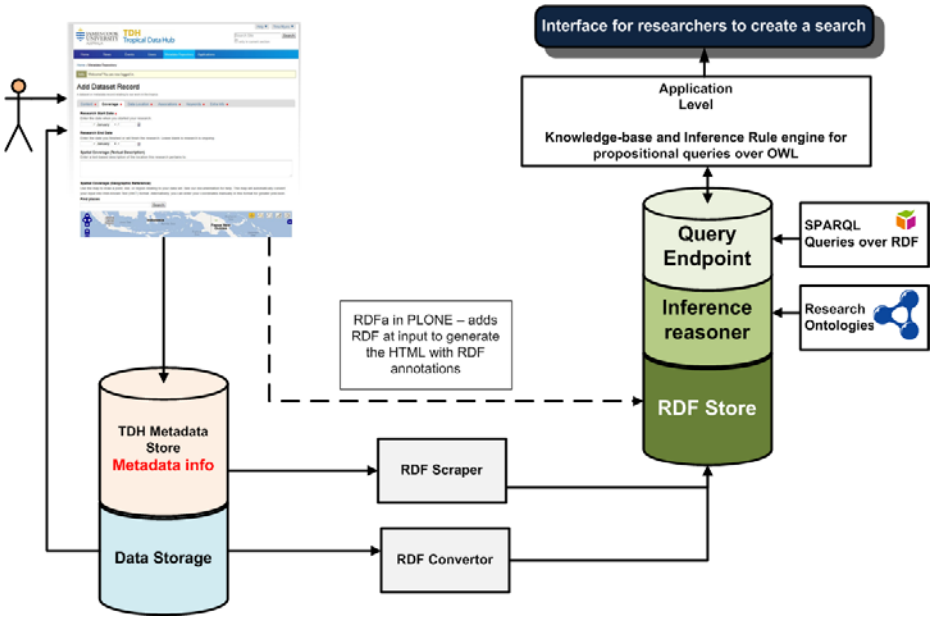


Fig. 1 The end-to-end semantic layers for the TDH framework

The SW uses *Universal Resource Identifiers* (URIs) to map terms between objects in a global graph data structure [2]. The RDF and OWL make use of URIs to identify concepts, objects and relationships within an ontology in the form of triples [3]. A triple can be associated with one or more schemas, which define the associated classes and properties. *Resource Description Framework Schema* (RDFS) is a framework of rules to define classes, subclasses, properties and sub properties. A triplestore offers an infrastructure to support RDF, RDFS and OWL reasoning and inference. A triplestore can be used to identify how the data from different data stores relate to each other. A query supplied to a triplestore can evaluate responses from across a variety of data stores on various platforms.

Current triplestores differ in extensibility, interoperability, capacity and performance [8]. Most are in varying stages of development and provide a variety of extensible frameworks. Types of triplestores differ, as they can be native or have an integrated relational data base backend. The support for both RDF and OWL layers and the functionality provided for reasoning and inference engines differs. The interoperability of the triplestore is important and requires consideration as to which *Operating System* (OS) platforms are supported (e.g., Linux, Windows, Macintosh, etc). In addition, the processing efficiency and memory management are diverse amongst the different initiatives. The final consideration is the amount of support provided for each triplestore through online documentation, discussion forums and the amount of activity present on these sites.

Current work on benchmark standards for triplestore mainly focuses on performance and capacity [8-10]. This focus is paramount to solving the limitations of contemporary semantic knowledge bases and triplestores. However, there are requirements during implementation that are important to the

ongoing use of these frameworks in a deployed environment [11]. Ease of use and support are also considerations but are not usually a part of a benchmark study.

The determination of which triplestore best suits an open VRE, where data submitted can be of any format, is not an easy question to answer. Presently there are no current and extensive benchmarks for triplestores for use in an application environment like the TDH. To conform to the TDH requirements, the implementation must:

- Not require extensive maintenance;
- Be able to query across disciplines;
- Be able to query disparate data sources;
- Be able to query data sources on any platform;
- Provide a response in an accurate and timely manner;
- Allow users to submit their own research data to the triplestore; and
- Provide a simplified tagging method (or enforce data stores are submitted in accordance with the TDH metadata schemas).

The resolution of two distinct questions were the grounding for this research on semantic enablement of a VRE:

1. What are the major criteria, considerations and functionality required to enable semantic capabilities in a VRE to support data integration, management and reuse; and
2. What is the state of the art in semantic technologies and to what degree do they support these criteria.

III. Methodology

This paper analyzed a number of triplestores to benchmark the effective and efficient methods to complete inference and queries. The triplestores were tested on a variety of performance measures including, but not limited to, response times, storage space, data read, usability, extensibility, development, efficiency, memory management, platform compatibility and support. A set number of queries were applied to gauge different aspects of each triplestores' capability against the performance criteria.

The response times tested included the speeds at which data is read from the triplestore. The read speeds tested involved:

- Data extraction directly from the triplestore on the remote storage device;
- Data extracted during a *cold run* where the query is run before the triplestore has loaded any data into memory;
- Data extracted during a *hot run* where the query is run again after a cold run.

After these tests were performed, an analysis on the triplestores yielded the varying strengths and weaknesses of each triplestore for read query execution.

The calculated storage space was determined by the amount of space required by the triplestore to store the data. The storage of data can occur in both volatile (main) memory and in permanent storage (hard disk drive space).

The triplestores' usability factors included:

- Its capacity to handle varying data formats (Text, CSV, GIS, etc);
- How difficult it is to implement;
- How intuitive and user-friendly is the user interface; and
- How efficient is the navigation provided to traverse the triplestore.

A. Criteria Development for the Weighted Decision Matrix

A review of 29 triplestores was conducted to examine viable use for the TDH. The most appropriate triplestore was determined by identifying which ones were currently available and how they adhered to the requirements outlined in Section II. Research into what directly affected the implementation and performance of each triplestore derived a set of criteria to meet the requirements. A WDM was constructed using the identified criteria to determine which triplestore is the most suitable (Tables I-III).

The WDM consists of a unique criteria specified for meeting the desired requirements. The WDM was separated into four distinct categories: interoperability (Table I), functionality (Table II), performance and support (Table III). Each category identifies specific aspects encapsulated by the use of a triplestore.

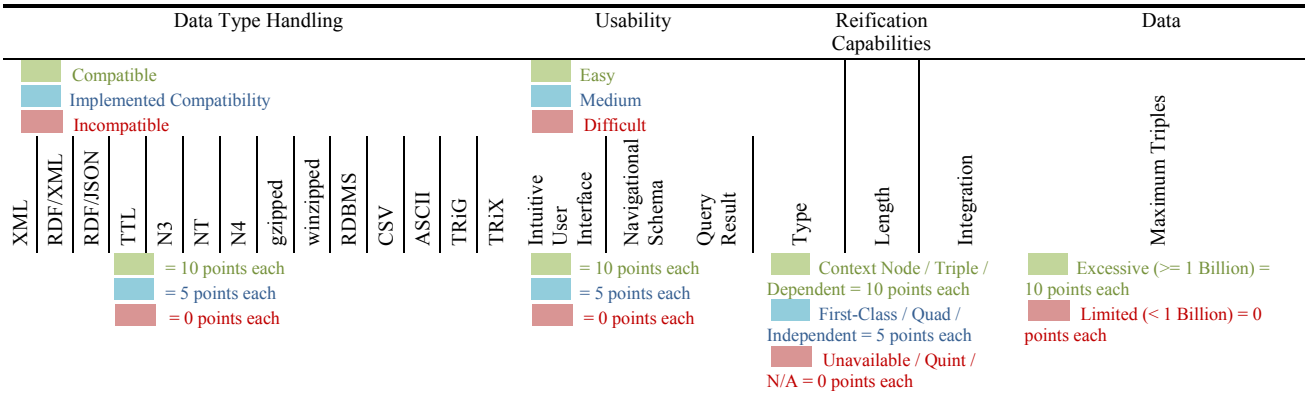
The features and the weighted criteria are grouped and listed within each table for examination as a matrix. For example, in Table I, interoperability, the key criteria are:

- OS compatibility;
- Open or closed sourced;

TABLE I
INTEROPERABILITY

Operating System			Integration		Supported API Languages											Open Source Reasoners Supported							Commercial Reasoners Supported			Open Source Inference Engines Supported							Commercial Inference Engines Supported			
Linux	Windows	Mac	Native	Integrated	PHP	Java	Perl	Lisp	JavaScript	Net Framework	Lua	REST API	Python	C#	C++	C	Ruby	FaCT	FaCT++	HermiT	Hoolet	KAON2	LarKC	Pellet	Dlog	OntoBroker	RacerPro	SHER	BaseVISor	Cwm	Drools	Jess	Prova	RIF	SweetRules	SWRL	Bossam	OWLIM-Lite	OWLIM-SE
Compatible			Is a		Compatible											Lisp	C++	Java	Java	Java	Java	Java	C	Java	Lisp	Java	Java	Python	Java	Java	Java	Java	Java	Java	Java	Java	Java	Java
Incompatible			Isn't a		Incompatible																																	
= 10 points each			Native = 10 points		= 5 points each											= 10 points each							= 5 points each			= 10 points each							= 5 points each			
= 0 points each			Integrated = 0 points		= 0 points each											= 0 points each							= 0 points each			= 0 points each							= 0 points each			

TABLE II
FUNCTIONALITY



- Native or integrated backend;
- *Application Programming Interface* (API) languages;
- Open and closed source reasoners supported;
- Open and closed source inference engines supported.

Table II and Table III show a similar format for the functionality, performance and support criteria.

The results from the analysis were then converted into the WDM. The WDM allowed for a set criteria to be established across numerous similar subjects, which could then be used to compare, analyze and rank the elements of each triplestore against the other triplestores. The criteria were graded according to a numerical scale and each triplestore is ranked relative to all the other analyzed triplestores. Applying a scaling factor (multiplier) to the results of each criterion provides identification of the strengths and weaknesses of each triplestore. The scaling factor is significant enough to alter the ranking of the triplestores, but does not excessively benefit any particular element. The triplestore located at the top of the list was then considered to be the most viable.

B. Interoperability Measures

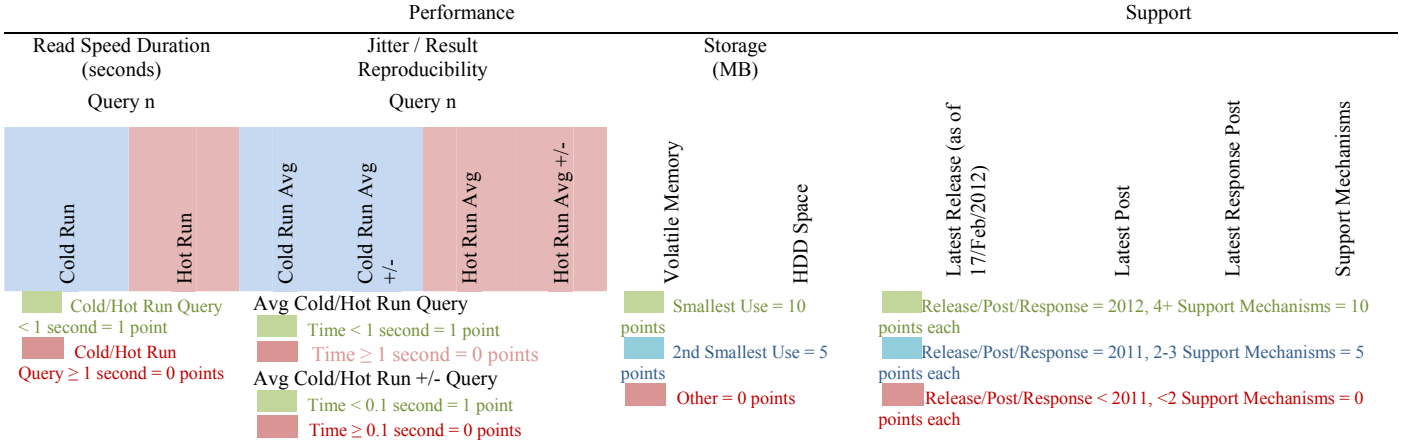
The level of required interoperability (Table I) by a triplestore begins with how compatible the triplestore is with various OSs. Ideally, the triplestore should be able to function on the most prevalent platforms that are identified in the WDM. Triplestores that were incapable of meeting the first criterion of operating on a Linux platform were removed from the remainder of the analysis (Linux is the underlying platform of the TDH infrastructure).

Information was collected from research papers and the triplestore's associated websites to determine interoperability. This information included the OS compatibility, backend integration, supported API languages, reasoning, and inference engines.

C. Data Storage Measures

Triplestores are capable of storing information natively or integrated with a different data model (e.g., relational, object oriented, etc.) (Table I). The TDH is an open portal available via web access so the data (while stored permanently in one place) should be accessible from any location without the need to duplicate and store that data locally. Triplestores that store data natively are more inclined toward the needs of the TDH. However, while this form of data integration is preferred, integrated data stores have faster performance [9]. Since the preference is for a native triplestore to run in conjunction with the legacy relational data base system, the integrated triplestores have not been removed from the matrix but rather

TABLE III
PERFORMANCE AND SUPPORT



incur a scaling penalty.

Application Programming Interface (API) (Table I) - The next criterion identifies which API languages are supported by the triplestores. There is no preferential language required so this criterion identifies the extensibility each triplestore has with the number of reasoners and inference engines that can be supported. Reasoning engines offer inference mechanisms via description logics, a subset of first-order (predicate) logic, over the available data [12]. The axioms specified in the ontologies are reasoned over to infer logical consequences. Both open-source and commercial reasoners exist that can be implemented in a triplestore [13]. This project's scope implies that only open-source reasoners will be utilized, however commercial reasoners were also researched in the event that free versions of these reasoners may become available.

Inference engines are used to apply propositional logic syllogisms to infer knowledge [2, 14]. The inference engines available are open source or proprietary. The proprietary inference engines with a free version release were included in this analysis. The API and programming languages of the different reasoning and inference engines were noted in the analysis to determine their compatibility with the triplestores.

D. Functionality Measures

The functionality category's first criterion identifies which data formats are supported (Table II). The more formats a triplestore is capable of supporting the less conformity is required by varying disciplines who are interested in storing or linking data.

The level of usability is the next criterion. Some triplestores are designed capable of being manipulated by other client-end software. This criterion is designed to meet the interactive usability needs of the user based on intuitiveness, navigation and result display.

The final criterion in this category identifies the limit of triples each triplestore can handle. This is an important attribute as the more triples capable of being stored in the models held by the triplestore; the more data that can be kept in main memory and the faster queries can be performed.

The data type handling and triples limitation of the triplestores was determined based the experience of implementing and performance testing each individual triplestore.

E. Performance Measures

There are three areas which were evaluated in the performance measure: read speed for thirteen different queries; result reproducibility (jitter) of the queries that were conducted; and the storage space required by each triplestore to store the data (Table III).

The first criterion is indicative of how well each triplestore handles a series of queries. The queries were designed to test the transactional efficiency of the triplestores. These queries ranged from extracting the number of triples stored in a single model to obtaining results for a specific predicate value found in any of the models. Some queries also included filter and optional clauses to test these aggregate functions. The results from these queries could then be averaged and the distance between either the fastest or slowest query and the average time would determine the jitter that occurs within each triplestore.

The query times (read speeds) for the test data were collected using one of three means. The first method involved the query time being output to the screen by the triplestore (e.g., Mulgara offers this function). The second method involved coding a PHP segment to determine the duration of the query (e.g., ARC2). The third method involved running a JavaScript stopwatch. The JavaScript stopwatch is not considered to be highly accurate, however due to the limitations associated with the use of the triplestore no other means were available to obtain the query time (e.g., Joseki). From these results the jitter could be extrapolated.

Jitter is used to determine if there is any significant fluctuation in results from a repetition of events under the same conditions. To test for jitter each query set (one cold run and three hot runs of one query) was performed three times. To ensure that the test conditions were the same, the server was rebooted between query set runs. From these results, the average time and variance for both cold and hot runs were calculated.

A measurement of the space used by the triplestore upon loading was recorded. This measurement was used to determine how efficient the triplestore was at storing the data. The data were stored in two different ways depending on the type of triplestore used (native/integrated). Hard disk drive space (physical memory) was calculated by reading the total used disk space before and after the data was loaded into each triplestore. Volatile memory (RAM) was calculated using the *top* command under a Linux terminal while the triplestore was running.

F. Support Measures

The support category refers to the assistance available from the triplestore's developers and user community (Table III). This criterion includes:

- How recently the triplestore version was updated;
- Who were the current users of the particular triplestore;
- What support mechanisms are in place to assist users of the triplestore; and
- How active the discussion forums were.

The latest release date of the triplestore indicates how recently a patch or update has been applied. This criterion indicates how quickly the developers of the triplestore update the code to conform to the requests of the users or to evolving standards.

The *Agent Use* identified in the WDM is an indicator of which end-users are using the triplestores and under what circumstances. To understand the extensibility and design of the individual triplestore, the end-user circumstances included home, commercial, enterprise and development environments.

The support mechanisms are an indicator of how the developers of the triplestores have provided the users with the means to acquire assistance. The types of assistance vary from emails to online documentation to wikis. These forms of

assistance are not indicative of a good use of such mechanisms and therefore the *Latest Post* and *Latest Response* activity were added to ensure the recent use of these mechanisms.

The release dates for the criteria outlined in the support category were obtained through the download pages of each respective triplestore. The support criterion was determined by analyzing the linked web pages of each triplestore's homepage.

G. Limitations and Constraints

The tests were performed on a Linux server. The server was running Ubuntu 10.04 LTS on an AMD AthlonTM II X2 255 Processor, Dual-Core with 4GB RAM and 167GB HDD space.

Data for this project were used from three distinct sources: the Smart Environment and Monitoring Analysis Technologies (SEMAT) project [15], the Australian Institute of Marine Science (AIMS) [16], and the Geographic Information Systems (GIS) [17]. The AIMS data were obtained in XML format, the GIS data were in ASCII format and the SEMAT data were in CSV format. The data were converted into XML/RDF format using a personalized PHP conversion script. The new data formats consisted of:

- AIMS: 44,600 triples from 3.21MB of data;
- SEMAT: 279,204 triples from 16MB of data; and
- GIS: 1,132,144 triples from 144MB of data.

The conversion of data resulted in a total of 1,455,948 triples from 163.21MB of raw data.

Three triplestores, 3store, C-Store and YARS, were identified as no longer active and removed from any further analysis. The Pointrel System, RAP, RDF-3X, RDF::Core and RDFBroker triplestores were also removed from further consideration because there had not been a software release in over three years.

IV. Analysis and Results

A. Interoperability

Ten points were given for every OS that the triplestore supported. There were four triplestores found to be compatible with each OS type, ARC, Jena SDB, Jena TDB Joseki and Mulgara. Then, a check to determine which of these triplestores met the native triplestore criterion was performed. The triplestores that were not reliant on a RDB backend were assigned ten points. ARC2 is based on an RDB backend and was therefore received zero.

The API languages supported by each triplestore allowed the identification of which programming languages can be used to construct interfaces. Triplestores that support a variety of languages provide a wider API usage base so if found compatible with an API language they were allocated five points.

The support for openly available reasoning engines and inference engines is a high priority for the requirements of the VRE. Ten points were allocated for each open source reasoning or inference engine compatible with the triplestore. Five points were allocated for every commercial reasoner or inference engine.

Virtuoso is the lead triplestore in interoperability with an extensive range of compatibilities with a wide variety of programming languages. AllegroGraph and Mulgara ranked second and third respectively with the Jena backed triplestores placing fourth. Each of these triplestores had a diverse compatibility with reasoner and inference engine support. ARC2 ranked tenth due to the lack of support for a variety of programming languages.

B. Functionality

The functionality analysis for each triplestore investigated their extensibility for data types, capacity and usability. Triplestores with the ability to handle numerous data types (such as RDF/XML, RDF/JSON, N3, CSV, ASCII, TTL, etc) prevent additional work to convert data into a format that is compatible. The primary criterion was the capability to handle RDF/XML formatted data. However, the triplestores that were capable of handling additional formats earned extra points in the WDM. Ten points were allocated for every data type found to be directly supported by the triplestore. If the data type required additional software implementation to be compatible, only five points were allocated.

The capacity, or maximum number of storable triples possible, is an integral part of a triplestore evaluation [8]. There is no pre-defined quantity of triples to be met in the criteria. However, each triplestore must be capable of storing a flexible and possibly excessive number of triples to enable dynamic data integration within the TDH. The potential lines of enquiry would require a minimum of one billion triples to be an acceptable amount for each triplestore. Ten points were allocated to each triplestore that met the minimum triple capacity.

Usability refers to the learning curve and user-friendliness of the triplestore determined during the implementation and testing. An intuitive user interface, navigational schema and query end point were the three functions evaluated. Ten points were assigned for each triplestore that was easy to put into operation and five points for the triplestores requiring a moderate amount of time to understand and implement. Triplestores that required an excessive amount of time to comprehend or failed to execute were allocated zero points. Mulgara, ARC2 and Jena TDB received the highest points due to their straightforward implementation.

Overall, Mulgara placed first in the functionality category due to its extensibility in the handling various data formats, capacity and user-friendly interface. Jena SDB and Jena TDB scored lower due to the support for less than one third available data formats. ARC2 scored lowest due to the difficult interface that required programmatic coding to implement.

C. Performance

The performance analysis for each triplestore evaluated their query runtime, jitter and use of memory and storage. Query runtime performance identified which triplestores were capable of meeting the read-speed performance criterion of a

maximum timeframe of one second. None of the triplestores managed to meet this criterion on every query due to the various levels of complexity. However, Mulgara and ARC2 managed to achieve this result on a majority of the queries posed.

While each query is expected to be performed in under one second, future similar queries should have little or no variance in performance. This variation is known as jitter. The amount of jitter that occurs within each triplestore is determined by averaging out the queries based on whether they were cold or hot runs. A score of one point was allocated for each triplestore that had an average query run-time of less than one second or an average query jitter value of less than one tenth of a second,.

A measurement of the storage space used by the triplestore upon loading was noted. Data can remain in various physical locations and only be loaded into volatile memory when there is a need to query the data, which is the point of applying a triplestore. Therefore, the efficiency of physical and volatile memory usage by each triplestore is important to ensure the resourceful querying of data. Ten points were assigned to the triplestores that used the least amount of either physical or volatile memory. The triplestores that used the next *least* amount of space was allocated a value of five points.

Mulgara was the lead triplestore for query execution time. ARC2 failed to score better in this criterion due to the inability to execute four of the 13 queries. Jena SDB and Jena TDB both failed to complete any query in under the one second limitation with a significant amount of jitter in query times.

D. Support

The amount of available support for open source products is an important deployment factor when deciding which to implement. The point allocation in this section is based on how recent the software was updated, how many forms of support were available and the activity on the community forums. The triplestores were allocated ten points each for having a recent version release, for having four or more support mechanisms and for having the latest post or latest response dates during 2012. Five points were assigned if the latest release, latest post and latest response dates were from 2011 or there were only two or three available support mechanisms. Any release, post or response dates pre-dating 2011 or less than two support mechanisms did not earn points.

The support results are indicative of how well each triplestore is supported by the development community. Jena SDB, Jena TDB, Mulgara and Virtuoso have active development communities providing assistance to end-users of their respective triplestores. The commercial triplestores have not generated new software releases and there were very little recent online support for end-users.

V. Discussion

The overall results indicate the triplestore's viability for use with a VRE such as the TDH. A summary of these results and ranks can be seen in Table IV. The leading candidate triplestore is Mulgara (score of 531) with significantly better results in functionality and performance than the other triplestores. Virtuoso ranked second based on extensive interoperability. There were eleven triplestores with scores between 200 and 400 indicating significant progress in the development and support of triplestores in general. ARC2 had a significant result from the performance analysis but failed to achieve a significant rank due to a lack of extensibility of data type handling and programming language support in the interoperability category.

Notably, each development community created their own variation in the SPARQL command structure. These variations include the structure of how to query across numerous graphs/models and the number of SQL commands incorporated into the structure (HAVING, FILTER, SELECT). Mulgara limits the use the HAVING, FILTER and SELECT clause to only one within each query so there is no support for nested queries. To obtain a result from a complex query that involves multiple use of these clauses requires the generation of numerous sub-graphs until the final sub-graph can be queried with a single instance of each command.

The following seven conditions justified the criteria chosen to determine the best candidate triplestore for implementation:

1. Maintenance of a triplestore must be minimal;
2. Queries can be processed across numerous disciplines;
3. Queries can be applied to disparate data sources;
4. Queries can be run on data stored on any platform;
5. Query response times are under one second;
6. Researchers can submit their own research data to the triplestore; and
7. A simple tagging method for metadata is provided.

These criteria were consolidated into the four main categories Interoperability (4, 6), Functionality (2, 3, 7), Performance (5) and Support (1). Extrapolating out these requirements, the effectiveness of each triplestore under these conditions was identified. Each triplestore was then analyzed

TABLE IV
OVERALL RESULT TOTALS FOR TOP SEVEN OPEN SOURCE TRIPLESTORES

Name	Version	Interoperability Result Subtotals	Functionality Result Subtotals	Performance Result Subtotals	Support Result Subtotals	Overall Total	Overall Rank
ARC	2	35	75	88	50	248	10
Bigdata	1.1	185	95	0a	50	330	5
Jena (SDB)	1.3.4	200	50	22	60	332	4
Jena (TDB) (Joseki)	0.8.10	200	85	22	55	362	3
Mulgara	2.1.11	215	115	141	60	531	1
Sesame	2.x	180	90	0a	40	310	6
Virtuoso	6.1.4	275	85	0a	55	415	2

[a] Attribute not analyzed

according to these requirements and ranked. The ranking system allows for an easier understanding of how effective each triplestore would be if implemented under the given conditions. Due to a lack of online documentation on implementation, there were some triplestores that were not able to be performance analyzed. The final results are not definitive due to the incomplete analysis of all triplestores. For example, Virtuoso ranked second because a performance analysis could not be conducted.

Mulgara proved highest in the overall ranking of a triplestore due to a significant number of extensible developments within the software. The advantages of utilizing Mulgara include the following.

- OS compatibility with all three main OSs;
- An extensive amount of support for reasoner and inference engine compatibility;
- The ability to handle seven of fourteen data source types and an additional data source type through extra software implementation;
- A simple yet efficient interface for querying data;
- Faster query run-time performance;
- Low amount of jitter;
- A large quantity of recent online resources for support.

Mulgara's disadvantages include a significant lack of API language support and a limited (less than one billion) storage capacity for triples.

VI. Conclusion

Presently, there are no current and extensive benchmarks for triplestores for use in VREs. This paper identified the major considerations and functionality associated with the semantic enablement of the TDH. These considerations and functionality extended into the environment's ability to support data integration, management and reuse. After determining these components, an evaluation of the current state of the art of semantic technologies was compared to the technologies available of each triplestore. The triplestores were tested against interoperability, functionality, performance and support criteria. These categories outlined how each triplestore met the requirements of practicality for implementation and extensibility for the evolution of the TDH.

The WDM evaluation model developed here provides a benchmarking standard for future triplestore evaluations. The model is openly available to view [7]. This standard provides researchers with a consistent model without the need to design new models for every new application of a triplestore. This evaluation provides the necessary information to assist the decision of which triplestore is most appropriate for a collaborative VRE such as the TDH. The semantic layer will define a faster and more efficient means for inferring new knowledge over extensive amounts of data and metadata within the TDH.

Future implementations of triplestores in VREs should undergo assessments as outlined in this evaluation model to assist their decision. This process will provide an in-depth evaluation of any triplestore capabilities interested in being implemented. The greater the number of triplestores analyzed provides opportunity for locating the most suitable triplestore for a given project. This analysis reviewed 35 triplestores, where 29 were open-source and six commercial.

Future work would identify any additional triplestores released. Performance testing will be done in real-time as opposed to a test bed, providing a more accurate assessment of the triplestores ability to handle the environment.

This evaluation model provides the underlying framework and is scalable and flexible. If there are any additional criteria that a project intends to pursue, the model can be modified to incorporate these new requirements.

Acknowledgment

The authors wish to thank Professor Ron Johnstone from the SEMAT project, University of Queensland, the Centre for Tropical Biodiversity and Climate Change, James Cook University, Townsville, the Australian Institute of Marine Science for the use of their datasets. The authors wish to thank Professor Ian Atkinson, Associate Professor Ickjai Lee and Associate Professor Bruce Litow for their interest, helpful discussions and feedback.

References

[1] T. Myers, J. Trevathan, and I. Atkinson, "The Tropical Data Hub: A Virtual Research Environment for tropical science knowledge and discovery," International Journal of Environmental, Cultural, Economic and Social Sustainability, January 2012.

[2] G. Antoniou and F. van Harmelen, A Semantic Web primer (2nd edition), 2nd ed. Cambridge, MA, USA: The MIT Press, 2008.

[3] D. Allemang and J. Hendler, Semantic Web for the working ontologist, 2nd Edition: effective modeling in RDFS and OWL. Burlington, MA, USA: Morgan Kaufmann, 2011.

[4] W3C. (2012) The Linking Open Data Project [Online]. Available: http://esw.w3.org/topic/SweoIG/TaskForces/CommunityProjects/LinkingOpenData

[5] T. Heath and C. Bizer, "Linked Data: Evolving the Web into a Global Data Space," in Synthesis Lectures on the Semantic Web: Theory and Technology. vol. 1, J. Hendler and F. vanHarmelen, Eds.: Morgan & Claypool Publishers, 2011, pp. 1-136.

[6] H. Alani, W. Hall, K. O'Hara, N. Shadbolt, M. Szomszor, and P. Chandler, "Building a pragmatic Semantic Web," IEEE Intell. Syst., vol. 23, pp. 61-68, May-June 2008.

[7] T. O'Neill. (2012) Weighted decision matrix triplestore evaluation model [Online]. Available: https://docs.google.com/folder/d/0B7rjeZeCV0l-Z3VxckphM2tQVzQ/edit

[8] W3C. (2012) RDF Store Benchmarking [Online]. Available: http://www.w3.org/wiki/RdfStoreBenchmarking

[9] M. Mohamed, L. Jens, A. ren, and N. Axel-Cyrille Ngonga, "DBpedia SPARQL benchmark: performance assessment with real queries on real data," in Proceedings of the 10th international conference on The Semantic Web (ISWC2011). vol. 1 Bonn, Germany: Springer-Verlag, 2011, pp. 454-469.

[10] C. R. Rivero, A. Schultz, C. Bizer, and D. Ruiz, "Benchmarking the Performance of Linked Data Translation Systems," in Linked Data on the Web (LDOW2012) Workshop at WWW2012. Lyon, France. 2012.

[11] G. Yuanbo, "A Requirements Driven Framework for Benchmarking Semantic Web Knowledge Base Systems," IEEE Transactions on Knowledge and Data Engineering, vol. 19, pp. 297-309, 2007.

[12] F. Baader, D. Calvanese, D. L. McGuinness, D. Nardi, and P. F. Patel-Schneider, "The Description Logic handbook theory, implementation and applications (2nd ed.)," 2nd ed. ed, F. Baader, D. Calvanese, D. L. McGuinness, D. Nardi, and P. F. Patel-Schneider, Eds. Cambridge, USA: Cambridge University Press, 2007, p. 545.

[13] T. Myers, J. Trevathan, R. Johnstone, and I. Atkinson, "Supporting dynamic hypothesis modelling and alerts in marine environments." in Coast to Coast 2012 Brisbane, Australia, 2012.
[14] S. Liang, P. Fodor, H. Wan, and M. Kifer, "OpenRuleBench: an analysis of the performance of rule engines," in 18th international conference on World Wide Web (WWW'09) Madrid, Spain: ACM, 2009.
[15] J. Trevathan, R. Johnstone, T. Chiffings, I. Atkinson, N. Bergmann, W. Read, S. Theiss, T. Myers, and T. Stevens, "SEMAT – The next generation of inexpensive marine environmental monitoring and measurement systems," Sensors, vol. 12, pp. 9711-9748, 2012.
[16] AIMS. (2012) AIMS Data Centre - Weather Observing System [Online]. Available: http://data.aims.gov.au/awsqaqc/do/start.do
[17] J. Vanderwal. (2012) Australia decadal climate change predictions (~5km resolution) [Online]. Available: https://eresearch.jcu.edu.au/tdh/data/a308884e-87a1-49f4-b7f2-be19d3123018