GIDS: Global Interlinked Data Store

Gareth Jones & Dave Braines
Emerging Technology Services
IBM United Kingdom Ltd
Hursley Park, Winchester, UK
Email: {garethj,dave_braines}@uk.ibm.com

Paul Smart & Trung Dong Huynh
School of Electronics & Computer Science
University of Southampton
Southampton, UK
Email: {ps02v,tdh}@ecs.soton.ac.uk

Jie Bao
Dept. of Computer Science
Rensselaer Polytechnic Institute
Troy, NY, USA
Email: baojie@cs.rpi.edu

***Abstract*— This paper introduces the Global Interlinked Data Store (GIDS), a technique to support the easy creation and retrieval of interlinked semantic data within a web-scale distributed network environment such as the World Wide Web (WWW). By using the GIDS a web application developer can treat the network as a data store without worrying about files, databases or other traditional data storage concerns. The data that is created on the network can be subsequently accessed and navigated by end users and software agents alike. The GIDS proposes a novel three-stage data storage process which enables the data to be stored in up to three contextually relevant locations to enhance subsequent retrieval opportunities. We propose that the GIDS can provide a highly-scalable distributed capability to store and retrieve data directly on a network. We believe that the capability offered by the GIDS will be of significant use to rapidly formed diverse coalitions who wish to communicate and exchange semantic data in a large network environment such as the WWW. Based on commonly used Web standards, we have implemented the GIDS in a prototype which can be invoked via simple web service requests to read and write data as prescribed by the GIDS.**

I. Introduction

Research sponsored by the International Technology Alliance[1] (ITA) into Semantic Web technologies has recently yielded a new approach to the distributed network storage of interlinked semantic data which we describe in this paper. We believe that this represents a significant improvement in usability and accessibility of semantic data when compared to existing Semantic Web best practices and builds upon the recent successes in publishing large volumes of interlinked data made in the context of Linked Data Web[2]. This paper provides details of this concept, which we call the *Global Interlinked Data Store* (GIDS), along with the outline of an exemplary prototype implementation.

The GIDS proposes a technique for directly storing semantic data "on the surface" of a network and enabling the subsequent navigation of that data. This is achieved through the use of network resolvable addresses in the definition of the data, removing the need for separate indexes defining the location of these data. The GIDS also enables data to be distributed over referenced locations; allowing a contextually relevant distribution of data across the network, and provides a basis for multiple paths of access to the data for subsequent onward navigation. The GIDS is intended to be a supporting platform for the Semantic Web, providing a simple set of interfaces within which data can be written, read and navigated by software agents, applications, or human users. The GIDS does not in itself provide any of the higher level capabilities associated with the Semantic Web (such as inference/entailment, complex queries, etc) but it is designed to support these capabilities in the agents and applications which use the GIDS.

This paper is organised as follows: Section II outlines the various related current technologies within whose context the GIDS resides. We then provide a conceptual overview for the GIDS in Section III, along with a discussion of capabilities, advantages, disadvantages and assumptions. Section IV contains the technical details for a prototype implementation. Section V provides an outline of the perceived military relevance along with some examples of usage. We briefly describe some related work in Section VI before summarising with our conclusions and planned future work in Section VII.

II. Current technologies

This section provides contextualising information for those key areas which are directly relevant to the GIDS.

A. The Semantic Web and the Linked Data Web

The Semantic Web is anticipated to be the next major iteration of the current World Wide Web (WWW)[3][4], with an increased emphasis on the representation of the "meaning" of information. This is a significant change from the current document-centric approach of the WWW where information is frequently stated with little-or-no explicit semantics, and therefore no readily machine-processable "meaning". The current WWW is fundamentally designed to be understood by humans rather than machines, and as such the meaning of the data that is currently expressed is implicit in the natural language used, implied in the page layout, or is even missing altogether because it is considered contextual and commonly known.

The semantics of data on the Semantic Web take the form of ontologies which formally define relevant concepts and their relationships and are designed for the purpose of enabling subsequent machine-processing. These ontologies can be constructed using an appropriate ontology language (such as OWL[5], the Web Ontology Language) such that they rigorously define the conceptual model for a particular domain, enabling software agents and human readers to process the ontology to gain insight and understanding of the model and any associated data which is represented in relation to that model.

Since the original proposal by Tim Berners-Lee in 1999[6] this approach to building a Semantic Web has had some successes within specific domains[7][8] but has not gained widespread adoption on the WWW, potentially due to the complexity of the languages involved, the skills required to build accurate domain ontologies, and the effort required by content producers to conform to those ontologies[9]. There are also perhaps underlying philosophical considerations as to whether a single unambiguous definition of a domain can ever be universally accepted[10], which again act as a disincentive to the widespread adoption of ontologies and therefore the Semantic Web itself in its current form.

A more recent, emergent, approach to providing semantic content for the WWW has been that of the Linked Data Web[2], which builds upon the principles and technologies of the Semantic Web but has taken a more pragmatic approach, starting with the data rather than the ontologies or models which define that data. The Linked Data Web advocates the semantic markup of data on the WWW using RDF[11] (Resource Description Framework) as the data markup language[1]. The marked-up data of the Linked Data Web should conform to relevant models (or ontologies) but by taking this data-centric approach these models tend to be far simpler than the traditional domain ontologies prescribed by the Semantic Web. In fact the existing WWW community had already started to take steps in this direction through the loose definition of terms such as "Semantic HTML" and the more rigorous definition of Microformats[12] to describe types of data that are already frequently expressed in existing web pages. The Linked Data Web initiative represents a concerted effort to coordinate action in this area and create a vast interlinked collection of datasets[2] which conform to a set of emerging standard models in the various domains of interest. The specific interlinking of datasets is a fundamentally important aspect of the Linked Data Web initiative, with these links serving as joining points between sets of information from different sources with different focus areas. These interlinked knowledge networks represent epistemically potent contingencies and are of particular interest to the ITA research in terms of their analysis, engineering and management, using the tools and insights gleaned from a mature network science discipline.

B. URIs and URLs

Another key difference between the Semantic Web and the more recent Linked Data Web initiative is the different approach to the use of unique identifiers that unambiguously define various entities. The Semantic Web advocates the use of URIs (Uniform Resource Identifiers), simply mandating that they must be unique, but not requiring them to be processable in any way. The Linked Data Web advocates that the URIs that are used should be dereferencable[2], i.e. in addition to their uniqueness they should also be able to be requested from the network, with a meaningful response being returned, although the Linked Data Web does not explicitly specify the detailed form of the response. This difference means that the Linked Data Web disallows blank or anonymous nodes[13] which are capable of modelling some useful knowledge structures, e.g., existential quantifications and class restrictions.

[1]OWL, the Web Ontology Language, is defined using RDF.

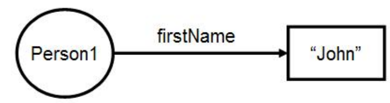

Figure 1. An example triple with a literal value

Figure 2. An example triple with a non-literal value

URLs (Uniform Resource Locators) are URIs that uniquely identify a dereferenceable resource and can therefore be considered equivalent to dereferenceable URIs as outlined above. However, in this paper we use the term "deferenceable URI" when discussing the conceptual aspects of the GIDS and reserve usage of URL for our GIDS prototype implementation since it has a specific parameterised URL composition which extends the core dereferenceable URI that defines the entity. This URL enables a user of the prototype to specifically request sets of information related to the underlying deferenceable URI. Section IV provides examples to highlight this.

C. Triples and Triple Stores

The Semantic Web makes use of existing data representation languages that express data in the form of triples. RDF is the most commonly used data representation language for expressing such triples, generally in the form of RDF/XML which is a serialisation syntax in XML for RDF. At the conceptual level a triple is comprised of three components which are known as *subject*, *predicate* and *object*. Each of these components can contain a value which is a URI (a reference) or a literal value (an absolute value such as a string or a number, e.g., "John" or 42). Example triples are shown in Figures 1 and 2. In both of these examples the triple is read from left to right. The subject is "Person1" (a URI[2]) and the predicates are "firstName" and "knows" respectively, both of which are also URIs. The object values are "John" (a literal) and "Person2" (a URI). Conceptually a triple can be read as an entity (the *subject*) having a relationship (the *predicate*) to a value (the *object*).

Triples are fundamental building blocks for the extensible representation and reference of data[3], and are the representational basis for the Semantic Web[14].

Triples as described above tend to be manifest in two major forms: as RDF statements located within a file or page, or as records inside a triple store. Triple stores are conceptually

[2]For simplicity the URIs used in these examples are not dereferenceable.

[3]Section VII includes some discussion of quadruples and named graphs, however the GIDS in its current form is fundamentally triple based.

similar to relational database systems, except that they do not encode a particular custom schema within which the data is stored. At a conceptual level a triple store can be considered to be the equivalent of a single table with three columns (*subject*, *predicate* and *object*) and a row for every unique triple that is stored[4]. The GIDS replaces the need for such a triple store since the data is accessed directly from the network via dereferenceable URIs.

This extant approach to storing triples in web pages, files or triples stores entails a knowledge of the location within which the triples are currently stored such that they can be accessed. When the triples are stored in a web page or file with a URL which does not correspond to the URIs in the triples themselves, the web page or file must be separately located. Alternatively, if the triples are known to be stored in a triple store then some kind of interface (possibly a SPARQL endpoint[5] or a standard database access mechanism such as ODBC[6]) must be accessed with the correct parameters in order to locate and return the required triples. In both of these cases the steps required to access the triples are non-trivial and often require additional information.

Having now contextualised the domain within which the GIDS is positioned, the remainder of this paper discusses the GIDS in detail, starting with the conceptual overview in Section III, then providing details of a prototype implementation in Section IV before concluding with a discussion of military relevance, related work and planned future work in Sections V, VI and VII respectively.

III. Conceptual overview

The GIDS aims to promote direct and easy access to semantic data in a network environment such as the WWW, through network resource requests based on the embedded URIs, avoiding concerns about indexes, specific file formats, file locations or access to triple stores.

Returning to our earlier description of a triple, we now introduce the concept of an *entity* which is core to the GIDS and common to the Semantic Web. Whilst a triple contains three elements (*subject*, *predicate*, *object*), any of those three elements can refer to an entity through the use of a dereferenceable URI. An entity can be comprised of one or more triples, and such multi-triple composition occurs when a number of triples exist, all with the same URI in the subject position. Figure 3 gives an example of a simple entity with the URI "Person1" which is composed from a number of individual triples, all of which have the same URI ("Person1") as the subject. This entity can be considered as being comprised of the attributes shown (the predicates) with the corresponding values (the objects) in Figure 3.

[4]In practice triple stores are more complex than this with numerous capabilities for efficiently storing and retrieving triples, but for the purpose of this contextual introduction this simplistic definition will suffice.

[5]SPARQL is a popular Semantic Web Query Language. A SPARQL endpoint allows semantic queries to be executed against a specific triple store, yielding a set of triple data as a response.

[6]The Open Database Connectivity software API

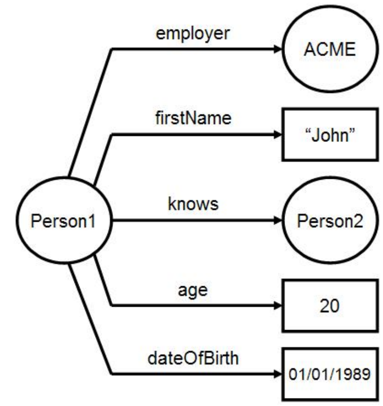

Figure 3. An example entity, comprised of numerous triples

The overall aim for the GIDS is to ensure that every entity (and every triple) is directly accessible[7] via the corresponding dereferenceable URI. This means that a simple network request to a dereferenceable URI will yield a response containing triples which in turn contain dereferenceable URIs, the details of which can be requested through subsequent network requests in a recursive manner.

Within the GIDS entities are identified using dereferenceable URIs as described previously. The act of dereferencing the URI yields all known triples which are related to that entity.

A. Core capabilities

The GIDS proposes three important additional capabilities which build upon the basis of the Semantic and Linked Data Web, specifically:

A defined response format: When dereferencing a URI the format and structure of the expected response is known and takes the form of triples which are related to the entity that corresponds to the requested URI. This is a significant extension to current best practice on the Linked Data Web, and enables subsequent automated processing of the data which is returned in the response. The GIDS does not specify what the format and structure of the response should be; it simply asserts that there should be a known structure and format which will be defined in any concrete implementation.

Direct triple access interface: The ability to directly read and write semantic data triples to/from the network through simple network resource requests (e.g., a HTTP GET request to a particular URL in the same way that a browser requests a web page). The GIDS does not specify the details of these requests since the GIDS could potentially be used in a number of different networks, however the prototype described in Section IV does provide details of these requests for this implementation.

Multiple storage locations: The GIDS proposes the ability

[7]Triples are accessible via their own dereferenceable URI through reification, described later in this paper.

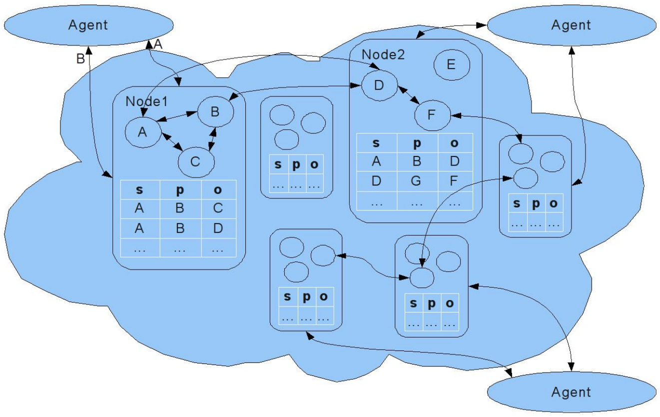

Figure 4. An example of triples distributed using the GIDS

to store semantic data triples in up to three logical locations[8] on the network. This builds upon the notion that every element of information is composed of a triple which is composed of three dereferenceable URIs (or literals). Each of the three components in the triple can be notified of the triple[9] when it is asserted through the execution of the triple access interface against the URI for the involved entity. Each of these three URIs can then optionally store the details of that triple for subsequent processing. This enables entities to store data which relates directly to them (i.e. that URI is used in the predicate or object position) rather than simply the data which is part of their definition (i.e. that URI is used in the subject position). This storage of data in multiple relevant locations supports the ability to initiate queries back across the network rather than always being forced to follow the links forward[10]. The GIDS proposes that each entity can choose whether or not to store each triple that it is notified of, and the decision to do so or not may either be universal or potentially based on contextual information such as the specific URIs involved, or the permissions of the requesting agent, etc. The GIDS does not prohibit the storage of triples in additional locations (e.g., for separate indexing or caching considerations), but such additional storage would be carried out specifically by the application using the GIDS rather than being a core GIDS capability and is therefore not discussed here.

The combination of the above three core capabilities yields this new approach which enables data to be easily stored *directly on the network* through simple network requests, to be retrieved in the same way with *a known response format* to facilitate subsequent automated processing, and enables each entity referenced by a triple to *store a separate copy of that triple* to facilitate improved navigation.

[8]The distinction between logical and physical location is important: each GIDS installation may serve part of a domain, a single domain or many domains and the physical location of the actual triple(s) is not of concern to the exercising application as long as the logical location can be dereferenced.

[9]Unless it is a literal, in which case it is not dereferenceable.

[10]e.g., requesting triples from a URI which refer to it as a predicate or an object rather than just the subject. Such queries are straightforward in centralised triple stores, but not in distributed environments.

Figure 4 shows a simple example of the GIDS being used to store data about a number of entities across a number of nodes within a distributed network environment. In this diagram a node is discriminated based on the hostname component of each URI, and it is anticipated that each node will run a separate GIDS implementation. Node 1 defines and stores information related to three entities (A, B, C) and Node 2 defines and stores information related to three entities (D, E, F). The information is stored in the form of triples, with s, p, and o being the subject, predicate and object respectively. In this example the dereferenceable URI of an entity can be calculated by concatenating the node and the entity id, e.g., http://Node1/A

The triple of particular interest *(A, B, D)* is actually stored against two separate nodes (Node1 and Node2). This is because Node1 defines entities A and B, whereas Node2 defines entity D. Using the GIDS each entity involved in a triple will be notified when the triple is asserted, and in this example both nodes therefore have stored a copy of the triple. The diagram also shows other nodes and other entities, and indicates various agents (software or human) who are interacting with the entities and triples via standard GIDS network requests.

The GIDS also provides capability for the triples themselves to be referred to as entities, providing the basis for an important capability known as "reification"[11].

When each triple is created it is assigned an unique identifier[12], with this identifier being returned in the response after the triple has been successfully asserted against one or more of the dereferenceable URIs which it contains. This triple identifier is also returned in the responses to all subsequent queries. This triple identifier is itself a dereferenceable URI and therefore is considered an entity in its own right, and when that URI is requested the details of its represented triple will be returned. This reification capability enables subsequent statements to be made about the original statement, for example: If the triple *"Person1 knows Person2"* has the URI *"Triple1"* then a subsequent triple can be created such as *"Person3 believes Triple1"* or *"Person4 created Triple1"* and so on. Using this simple reification technique in conjunction with dereferenceable URIs enables unlimited chains of reified information to be stated. This reification technique within the GIDS is designed to closely match the de-facto *rdf:Statement* approach to reification within RDF, although we acknowledge that the topic of reification in RDF and in the Semantic Web in general is still subject to extensive debate.

Whilst the GIDS works best in environments where all nodes and entities are able to respond to GIDS requests it is also designed to work with existing network accessible semantic data. For example GIDS can tolerate cases where a triple contains URI references that are either not dereferenceable, or are dereferenceable but do not return the expected data

[11]Additional techniques for identifying triples may also be implemented at the application level, e.g., through use of quadruples or HTML anchor tags.

[12]Each logical triple may have up to three separate physical triples stored. See Section IV for further details.

format or structure (because they are not running on a GIDS infrastructure).

B. Advantages

We have discussed the abstract GIDS concept in some detail, and some advantages of the approach are specifically enumerated here:

Data Visibility: As discussed previously, one of the problems associated with the Semantic Web is that of data visibility. As data in the GIDS is represented by dereferenceable URIs, all data can be accessed, modified and deleted through the interaction with these URIs. This enables easier data sharing, re-use and integration. It is also a significant extension to the current Linked Data Web approach mainly due to the known format and structure of the response and the ability to directly modify or delete data via this interface (as discussed previously). Data visibility also enables direct interaction with data via network level requests. Multiple implementations of the GIDS can therefore inter-operate within the same environment through exchange of data and network requests regardless of configuration or software components used.

Data Access: The availability of data from all entities associated with a data element (a triple) allows data access from any of these entities. This provides entry points to users of the data from either the subject, predicate, or object entity. In the "Person1 knows Person2" example, this triple may be retrieved by asking "Person1" who they "knows", by asking "Person2" who they believe "knows" them, or by asking "knows" which entities are related via the "knows" predicate. This does raise the prospect of privacy and trust concerns which are discussed in the Section III-C.

Data Ownership: Data in the GIDS is stored and owned by the entities associated with the data. In the example "Person1 knows Person2", both "Person1", "knows" and "Person2" are owners of their own data and therefore can store a copy of this triple when they are notified of the assertion of the triple. It is likely that popular predicates will generate potentially large volumes of related triples. So, if the "knows" predicate was managed by an organisation concerned with social connections, that organisation can choose to manage the scalability issues associated with recording all triples referring to the "knows" predicate if they see that data as valuable. Alternatively they can totally, or selectively, ignore these asserted triples as required. This approach enables automated capture of predicate related information to occur in addition to the more localised ownership of data relating to the subject and object entities.

A natural extension of this predicate-based data storage is to consider the various core modelling predicates themselves (such as *rdf:type*, *rdfs:subClassOf* and *owl:sameAs* for example). Clearly in their current locations and implementations these URIs will not be able to respond to GIDS requests and store the appropriate data, nor may they wish to since the volumes of data involved will be very large. It is possible in certain domains that GIDS requests to these URIs may wish to be intercepted with the intention of storing the resulting triples in a proxy location. This would then support use of GIDS techniques against these common predicates if suitable resources are provided to handle the large volumes of requests.

Data Redundancy: In GIDS, a triple can be replicated at the subject, predicate and object entities thus providing some data redundancy. If data is lost, was never stored, or is temporarily unavailable at one of the associated entities (as is the nature of an Internet-based infrastructure), data can be retrieved from other entities associated with the required triple. This consideration of data redundancy also applies to inferred triples, although inference and entailment capabilities are beyond the scope of the GIDS (See Section III-D).

C. Disadvantages

The GIDS architecture also has a few disadvantages that are discussed here:

Efficiency: Every request to create or delete a triple can be made to all three URIs that make up that triple, i.e. to the subject, predicate and object URIs. This means that every logical request may result in up to three physical requests across the network. This disadvantage is the corollary of the data access, ownership and redundancy advantages described previously. Existing WWW infrastructure components which provide proxy or caching capabilities may be useful in this context.

Reliability and trust: The nature of the WWW is that it is potentially unreliable and we specifically acknowledge that the ITA research context assumes a MANET (Mobile Ad-hoc Network) environment which will likely be significantly less reliable than the traditional WWW environment. Servers may become unavailable or resource-constrained, they may be unreliable, data may be inconsistent between different sources or may be untrustworthy, etc. Storing up to three copies of every triple does provide some mitigation, but it also introduces the problem of potential inconsistency. Another common reliability factor related to the WWW is that of changing URIs; these are entirely under control of the owner and hence can be changed at any time. There are several potential solutions to mitigate against such changes: existing network redirecting solutions; use of an appropriate *sameAs* predicate to relate relevant URIs; and data migration from one URI to another (whilst updating information known by related entities). This is the nature of the WWW today, and there is plenty of encouragement for people to choose stable URIs[15]. As the WWW has been proven largely successful in meeting a similar reliability challenge we are optimistic that this issue will not be prohibitively restrictive for the GIDS.

Privacy: The notification of data to each of the three associated URIs does raise concerns about privacy, since it may not always be appropriate to notify a URI that a related statement has been made. The GIDS does not enforce the notification to all associated URIs, so the application or user can decide which of the three to send. Also, the GIDS does not mandate that a URI which receives a request to assert a triple must act on that request. In both of these cases the application making or receiving the request can take into account contextually-relevant information (such as whether the request is authenticated) and use existing WWW conventions to aid the decision. Further investigation into the privacy aspects of the GIDS are planned in our future work.

D. Assumptions

The GIDS makes some assumptions, some of which are key enablers for the desired capabilities, whilst others are inherited from the Semantic and Linked Data Web context:

Dereferenceable URIs: As with the Linked Data Web, and as discussed earlier, a restriction is imposed on the use of dereferenceable URIs for all entities. Although it is possible for some entities to be stored that do not fit these requirements, they cause limitations in processing, but not failures. Consider the previous example triple "Person1 knows Person2". If "Person1" is identified by a non-dereferenceable URI or the URI is dereferenceable but does not conform to the GIDS interface, users of the data can no longer retrieve (or write) any data directly from "Person1" and hence cannot ask the question "Who has some kind of relationship with Person1?"[13]. We also acknowledge that many applications may operate above the GIDS infrastructure, for example potentially providing search engine or general registry/index capabilities in the same way as happens on the WWW today.

Anonymous/Blank Nodes: As mentioned previously the GIDS does not readily support blank or anonymous nodes since they do not have a globally guaranteed unique identifier to act as the dereferenceable URI. Since the GIDS is aimed at the storage of triple data we therefore delegate the problem of dealing with blank nodes to the application or agent which is using the GIDS and place the requirement on that application to generate appropriate URIs for any blank nodes it may wish to deal with.

Inference/Entailment Support: The GIDS itself does not offer any inferential or entailment capability since it is concerned purely with the storage and retrieval of triple data in a distributed network environment. Applications and agents using the GIDS may wish to provide such capabilities and these will be supported by the GIDS through the retrieval of the underlying data and the ability to assert new data via the same network level interface. The application must define appropriate algorithms to implement such inference on a GIDS infrastructure, for example considering the notification method for the assertion (or deletion) of inferred triples and whether such events will be notified to all URI entities involved in the inferred triple. It is possible that certain applications using the GIDS may wish to manage their own centralised triple store in a more traditional manner to afford more efficient and reliable inferences, but this should be constructed as a "cache" of the actual underlying triples retrieved from the GIDS.

IV. Implementation

This section provides outline details of a prototype GIDS implementation. URL construction rules, expected data formats and structures are given below, and these details are all provided in the context of the earlier conceptual overview of the abstract GIDS concept which underlies this implementation. The use of existing WWW standards enables us to reduce development cost, and provides other key capabilities such as scalability and caching (if required), authentication and security support, etc.

A. A RESTful architecture

The GIDS prototype uses HTTP based requests and is built on REST[16] (Representational State Transfer), a well known and acclaimed software architecture based on a simple set of principles. This software architecture was chosen as it fits well with the intentions of the GIDS, and it is tightly coupled with the fundamentals of the WWW itself, bringing many advantages.

The core principle of a RESTful architecture (particularly when associated with the WWW) is that the interaction with resources through the use of dereferenceable URIs. Interaction with resources is achieved using a stateless protocol such that state is managed and represented by the resources themselves, not as part of the client-server communications. With the WWW, REST implies use of HTTP as it was originally designed, with appropriate use of HTTP verbs, status codes and content negotiation.

There are many benefits associated with REST, a primary one being the support for caching data which can enable faster response times and reduced server load. The stateless nature of REST allows for better server scalability by reducing the need for session state maintenance and allows swapping of servers during sessions. As REST with the WWW is based on the basic protocols of HTTP, no extra client libraries are required, making development and client application support much simpler.

B. API

All actions related to a single triple require up to three separate requests to all non-literal URIs that represent the entities in that triple.

Actions
The following actions against entities or reified triples are supported through the use of specific HTTP verbs[14] executed against dereferenceable URIs: GET (to Retrieve), POST (to Create), DELETE (to Delete), PUT (to Replace).

URL parameters
The basic HTTP GET request to a dereferenceable URI will return the set of triples which are known by that URI where the URI is featured in either the subject, predicate or object position. Additional parameters can be specified when constructing the URL to filter the set of triples that are returned:

- *?s=(value)* Return only where subject matches *(value)*.
- *?p=(value)* Return only where predicate matches *(value)*.
- *?o=(value)* Return only where object matches *(value)*.

[13] "knows" may still be asked "Who does Person1 know?".

[14] Unknown HTTP verbs return "501 Not Implemented", and unsupported HTTP verbs return "405 Method Not Allowed".

In cases where multiple parameters are specified, the filter values are combined using the Boolean *AND* operator.

These parameterised URLs facilitate the expression of "atomic queries", i.e. a simple triple-pattern-matching query for a specific URI. Such queries can be composed (by the executing application) into larger and more complex queries to yield a SPARQL-like query capability using GIDS network requests.

Content Negotiation

When retrieving data from an entity, the GIDS prototype API allows the requester to specify the format of the data. This works by using the standard HTTP process of specifying an ordered list of preferred response formats in the HTTP Accept header. Responses are then returned with the HTTP Content-Type header specifying the chosen format of the data using MIME types. Currently supported formats in the GIDS prototype API are shown in the table below:

Name	**MIME Type**	**Notes**
XHTML	application/xhtml+xml	HTML table with RDFa
JSON	application/json	Talis proposal[15]
RDF/XML	application/rdf+xml	W3C specification[16]
Turtle	application/turtle	Turtle specification[17]
N-Triples	text/plain	W3C recommendation[18]

The embedded RDFa[17] within the XHTML response ensures the machine-processable semantic content is explicitly stated within the human-readable page that is returned.

C. Reification

As mentioned previously, each explicitly stated triple is assigned a unique id in the form of a dereferenceable URI so that it can be subsequently referred to as an entity. This is known as reification. Since the GIDS can send notifications to all entities involved in a triple this means that the network could contain three separate physical copies of the same logical triple[19] at three potentially separate network locations with different reified URIs. In the current GIDS implementation the only way to establish the logical equivalence of separate physical triples is to evaluate them to detect identical Subject, Predicate and Object composition. It is possible that specific specialised predicates could be used to record triple equivalence (e.g., similar to *sameAs*), although the unreliable nature of the network environment in which the GIDS is operating could not assure that all such relationships would be recorded consistently. With this limitation in mind we have currently chosen to leave these triples unrelated and specifically seek logical equivalence directly in the rare cases where reification needs to be used in this way. Alternatives that we have considered to support logical reification include the use of quadruples (as outlined in Section VII), the potential for a specific reification server/service or the delegation of this responsibility to the application layer. The resolution of this issue is a key focus for our future work.

[15]Specification at http://n2.talis.com/wiki/RDF_JSON_Specification
[16]Specification at http://www.w3.org/TR/rdf-syntax-grammar/
[17]Specification at http://www.dajobe.org/2004/01/turtle/
[18]Specification at http://www.w3.org/TR/rdf-testcases/#ntriples
[19]i.e. the triple with identical Subject, Predicate and Object values.

D. Software

This prototype implementation of the GIDS is built using the ARC[20] framework which runs in PHP and uses MySQL as a basic triple store. This choice of software stack was largely based on the good fit between our need for a lightweight and quick-to-use environment, along with built-in support for RDF and triples, multiple serialisation formats, and the desire for a RESTful interface as described earlier.

We also note Pubby[21] which is available to act as a Linked Data front end to existing SPARQL endpoints and may be a useful component to use when considering integration of the GIDS into the existing Semantic Web environment.

V. Military relevance

Earlier work from the ITA research program has yielded a scenario based around the fictional country of Holistan [18], and within this overall scenario a specific vignette has been constructed to facilitate exploration of the various opportunities for potential knowledge-based capability advancements in military coalition environments[19]. We envisage the GIDS being used as the fundamental information backbone within which any open data can be shared between military coalition partners and non-government organisations (NGOs) such as charitable or relief organisations, media companies, or individuals, etc. The open and accessible approach to data access and construction that is taken by the GIDS, coupled with the lightweight network level API should provide a compelling capability for agile organisations and individuals to directly interact with and contribute to the relevant data directly, without the need to purchase or construct complex applications or systems to do so. This low-level and flexible approach to data production and consumption means that the organisations, individuals and systems will be able to respond far more quickly to a diverse and changing set of requirements, and also to more rapidly integrate new information sources as they become available.

A specific example in the context of the Holistan vignette would be to *"support the rapid retrieval and integration of multiple types of situation-relevant information"*[19] in the context of the mission planning phase, for example through integration of available open meteorological information. Alternatively, data relevant to the region, such as de-mining information contributed by a humanitarian de-mining agency, could be made available via the GIDS, linked to other relevant information and subsequently consumed in the form of situation-relevant information during planning or execution phases. Through reification it is also possible for various agencies or individuals to comment on the available data, offering opinion or qualification of specific facts which can again be subsequently consumed to provide a broader, consensually authored, information set. The benefit of the storage of data at the three locations outlined earlier is especially beneficial in this latter case since it may be a common requirement to

[20]http://arc.semsol.org/
[21]http://www4.wiwiss.fu-berlin.de/pubby/

navigate to data from the predicate or the object entity rather than the more traditional subject-oriented route.

The GIDS could also be used in more secure military environments to achieve similar benefits for the appropriately authorised users and systems, but the greater emphasis on security, coupled with the increased prevalence and proprietary nature of existing systems in this space yield a less obvious case for immediate benefit.

VI. Related work

The Linked Data Web community continues to refine their preferred approach to the efficient publishing of linked data using dereferenceable URIs[13] and many of the implementation details of the GIDS follow the conventions and principles they propose. RDFPeers[20] and PAGE[21] are examples of recent work which has been undertaken to propose the distributed storage of RDF data in multiple locations, similar to the approach taken in the GIDS. These approaches however do not use dereferenceable URIs to determine the network locations to be used and instead use separate indexes and hash algorithms to calculate the unique identifiers for the network storage locations. Finally some recent work looking at a similar Peer-to-peer distribution of RDF data[22] is emerging from the University of Southampton, but again is not specifically focused on the use of dereferenceable URIs as the fundamental interaction and navigation mechanism.

VII. Conclusion and future work

The Global Interlinked Data Store (GIDS) concept and associated prototype provide a novel approach to support the creation and navigation of semantic data triples in a distributed network environment. The GIDS provides a number of extensions to the current Linked Data Web approach, and is designed to provide a lightweight and simple interface to semantic data stored in a network environment such as the World Wide Web. This paper describes a prototype implementation built according to a RESTful architecture. We also discuss the military relevance of this work in a web based coalition context and provide details on how the GIDS is related to other ongoing research and development activities in this area.

Further work will focus on the following areas: privacy concerns, scalability and performance study and optimisation techniques, additional support for complex queries, data import/export support, better integration with existing Semantic Web data sources and the development of tools to allow GIDS integration with existing Semantic Web tools. Also, as mentioned previously, we intend to investigate extant named graph[23] (quadruple) techniques to potentially offer better support for logical reification resolution, and the handling of representation syntaxes that do not directly render triples.

Acknowledgment

This research was sponsored by the U.S. Army Research Laboratory and the U.K. Ministry of Defence and was accomplished under Agreement Number W911NF-06-3-0001. The views and conclusions contained in this document are those of the author(s) and should not be interpreted as representing the official policies, either expressed or implied, of the U.S. Army Research Laboratory, the U.S. Government, the U.K. Ministry of Defence or the U.K. Government. The U.S. and U.K. Governments are authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation hereon.

References

[1] G. Cirincione and J. Gowens, "The international technology alliance in network and information science a U.S.-U.K. collaborative venture [very large projects]," *Communications Magazine, IEEE*, vol. 45, no. 3, pp. 14–18, March 2007.

[2] T. Berners-Lee, "Linked data - design issues," W3C, Design Issue, February 2006. [Online]. Available: http://www.w3.org/DesignIssues/LinkedData.html

[3] T. Berners-Lee, J. Hendler, and O. Lassila, "The semantic web: Scientific american," *Scientific American*, May 2001.

[4] N. Shadbolt, T. Berners-Lee, and W. Hall, "The semantic web revisited," *IEEE Intelligent Systems*, vol. 21, no. 3, pp. 96–101, 2006.

[5] M. Dean and G. Schreiber, "OWL web ontology language reference," W3C, W3C Recommendation, February 2004.

[6] T. Berners-Lee, *Weaving the Web: Origins and Future of the World Wide Web.* Texere, 1999.

[7] K.-H. Cheung, A. Smith, K. Yip, C. Baker, and M. Gerstein, "Semantic web approach to database integration in the life sciences," *Semantic Web: Revolutionizing Knowledge Discovery in the Life Sciences*, pp. 11–30, 2007.

[8] H. Alani, D. Dupplaw, J. Sheridan, K. O'Hara, J. Darlington, N. Shadbolt, and C. Tullo, "Unlocking the potential of public sector information with semantic web technology," in *The 6th International Semantic Web Conference (ISWC)*, 2007.

[9] S. Haustein and J. Pleumann, "Is participation in the semantic web too difficult?" in *In Proc. of 1st Int. Conf. on the Semantic Web (ISWC02*. Springer, 2002, pp. 448–454.

[10] F. S. C. da Silva, W. W. Vasconcelos, D. S. Robertson, V. V. B. B. Brilhante, A. C. V. de Melo, M. Finger, and J. Agustí-Cullell, "On the insufficiency of ontologies: problems in knowledge sharing and alternative solutions," *Knowl.-Based Syst.*, vol. 15, no. 3, pp. 147–167, 2002.

[11] O. Lassila and R. R. Swick, "Resource Description Framework (RDF): Model and Syntax Specification," W3C, W3C Recommendation, February 1999, see http://www.w3.org/TR/REC-rdf-syntax/.

[12] J. Allsopp, *Microformats: Empowering Your Markup for Web 2.0*. friends of ED, March 2007.

[13] C. Bizer, R. Cyganiak, and T. Heath, "How to publish linked data on the web," 2007. [Online]. Available: http://www4.wiwiss.fu-berlin.de/bizer/pub/LinkedDataTutorial/

[14] T. Berners-Lee, "Semantic web roadmap," On-line draft www.w3.org/DesignIssues/Semantic.html, 1998.

[15] T. Berners-Lee, "Cool URIs don't change," On-line draft http://www.w3.org/Provider/Style/URI, 1998.

[16] R. T. Fielding, "Architectural styles and the design of network-based software architectures," Ph.D. dissertation, University of California, Irvine, Irvine, California, 2000. [Online]. Available: http://www.ics.uci.edu/ fielding/pubs/dissertation/top.htm

[17] G. Velez, "Semantic web publishing with RDFa," *Linux J.*, vol. 2008, no. 171, p. 5, 2008.

[18] D. Roberts, G. Lock, and D. Verma, "Holistan: A futuristic scenario for international coalition operations," in *Conference of the International Technology Alliance (ACITA 2007)*, July 2007. [Online]. Available: http://www.usukita.org/papers/3365/RobertsLockVerma.pdf

[19] P. R. Smart, D. Mott, E. Gentle, D. Braines, W. Sieck, S. Poltrock, P. Houghton, A. Preece, M. Nixon, M. Strub, D. Roberts, D. Verma, and N. Shadbolt, "Holistan revisited: Demonstrating agent- and knowledge-based capabilities for future coalition military operations," in *Conference of the International Technology Alliance (ACITA 2008)*, July 2008.

[20] M. Cai and M. Frank, "RDFPeers: a scalable distributed rdf repository based on a structured peer-to-peer network," in *Proceedings of the 13th international conference on World Wide Web*, 2004, pp. 650–657.

[21] E. D. Valle, A. Turati, and A. Ghioni, "PAGE: A distributed infrastructure for fostering RDF-based interoperability." in *DAIS*, ser. Lecture Notes in Computer Science, F. Eliassen and A. Montresor, Eds., vol. 4025. Springer, 2006, pp. 347–353.

[22] J. Zhou, W. Hall, and D. De Roure, "Building a distributed infrastructure for scalable triple stores," in *Journal of Computer Science and Technology, 24 (3)*, 2009, pp. 1–15.

[23] J. J. Carroll, C. Bizer, P. J. Hayes, and P. Stickler, "Named graphs, provenance and trust," in *WWW*, 2005, pp. 613–622. [Online]. Available: http://www.informatik.uni-trier.de/ ley/db/conf/www/www2005.html#CarrollBHS05