Developing an Ontology for the Domain Name System

Nickolas J. G. Falkner Paul D. Coddington Andrew L. Wendelborn
School of Computer Science
University of Adelaide,
Adelaide, South Australia 5005,
{jnick,paulc,andrew}@cs.adelaide.edu.au

Abstract

Ontologies provide a means of modelling and representing a knowledge domain. Such representation, already used in purpose-built distributed information systems, can also be of great value when applied to existing distributed information systems. The domain name system (DNS) provides a wide-area distributed name resolution system which is used extensively across the internet. Changing the type and nature of resource records stored in the DNS currently requires an extensive request for comment procedure which takes a substantial amount of time, as the change has to be made globally. We propose an ontology for a DNS zone file, to provide a machine readable codification of the DNS and a mechanism for allowing local changes to the stored and represented structure of DNS records, using the extensible nature of OWL to allow local variations without having to go through the manual RFC procedure. This ontologically based system replaces a slow manual procedure with a rapid, machine-realisable procedure based on a uniform ontological representation of significant DNS knowledge. This paper discusses the application of ontologies to the DNS and how such an application can be built using OWL, the web ontology language.

Keywords: domain name system, ontology, OWL, distributed systems, semantic web

1 Introduction

The Domain Name System (DNS) [3, 4] provides a mapping from the Internet Protocol (IP) addresses of computers to names, which allows the imposition of a human-friendly hierarchy of names on top of a sea of numbers. The IP addressing system is hierarchical, a design decision that facilitates subnet assignment and reflects the hierarchical relationship that exists in the implementation of networks. However, it is generally easier for users to recall the names of machines and domains rather than a multiple digit sequence, regardless of how appropriate or logically justifiable the scheme is behind the digit sequence.

The current wave of research into ontologically based technologies can enhance existing information systems through the addition of metadata, annotation and the use of more suitable service models. These techniques are largely applied to newly created information systems but we have seen an opportunity to take an existing complex and widely-used information system, the DNS, and recast it. The purpose of this paper is to show that such a recasting is a positive contribution.

The big advantage of extending the DNS with semantic annotation is that a large number of applications across the Internet could use this metadata to derive information that would normally be stored out-of band, as text files associated with DNS stored records, or that may not be obviously related to information which already exists in the DNS.

For example, in the current DNS, the presence of a record which maps a name to an IP address does not automatically create an entry which maps the IP address back to that name. While this is not always a desirable behaviour, and the DNS was deliberately created so that these assumptions were not made, there are many situations where this behaviour could be desirable. Instead of a logical and automated process, sites are left with a manual and user-driven one to alter two logically related but physically distinct data files (the DNS zone files) that store these mappings.

There is currently no mechanism that allows DNS behaviour to be altered on a site-by-site basis. In this context, a site is a logical entity that could contain one machine, several machines in one location or the machines of an entire company, spread nationally or globally. It is a close approximation to the notion of a virtual organisation (VO) referred to in Grid computing. Our proposal readily facilitates the description of such behaviour at whatever size or nature of site is required. From the previous example, a site could apply an ontological relationship that states that for every name-to-IP-address mapping there exists an IP-address-to-name mapping using the same pieces of data in an inverse

Proceedings of the 16th International Workshop on Database and Expert Systems Applications (DEXA'05)
1529-4188/05 $20.00 © 2005 **IEEE**

relationship. This should not be the default behaviour for all DNS systems but there are many sites, and subsets of sites, where such behaviour is desirable.

There are many similar, and relatively minor, changes that cannot be applied globally and would not be passed through the current RFC mechanism. There is an existing extension mechanism specified in the standards track for DNS, EDNS0 [5], but this is primarily to allow backward compatible mechanisms for protocol growth which reflects the limits of the original size of the fields used to describe opcodes (OPCODEs) and response/error codes (RCODEs). It does not include the dynamic, and potentially localised, inclusion of new resource record types. Even if a group of DNS administrators could agree on local changes and implement them they would effectively define the new behaviour in a form that could be captured ontologically. With the proposed mechanism, the new ontological information could be added to an existing, running system rather than having to be translated to source code for the nameserver daemon, tested, debugged and then deployed.

The semantic web allows the use of metadata to place a structure on the data stored and used within a system and to show the relationships between this data. Ontologies can then be used to interpret the metadata and classify it to provide a truly machine-interpretable form of the data. We believe that ultimately this can lead to the production of a 'semantic internet', where the semantic web, and semantically enhanced network services provide a much more useful environment for distributed computation. With this in mind, all services should be capable of either handling metadata natively or should, at worse, pass on such information without stripping it from the information stream. Given the importance of locating the correct service, and hence the correct machine with the desired characteristics, we believe that such information can be encoded on top of existing services, such as the DNS, to re-use existing software and hardware in the new semantically-enhanced network environment.

Although there has been some work on the domain name system with ontologies, it has focussed on providing an ontology for the management of DNS [2] rather than the information DNS contains, which is stored in zone files.

There has also been work [1] to provide a virtualisation of DNS to allow the use of RDBMS for storing this data, and hence support queries over this space. The mapping between the DNS and an RDBMS representation, while a possible outcome of what is proposed here, is not the focus. The ontology presented here provides a well-defined context for each data element and construct in the DNS so that the annotated data can then be used in the semantic web or semantic grid.

This paper shows how an existing service can be modified to fit into the semantic web and semantic grid frameworks without compromising the loosely-coupled and distributed nature of the service. We provide an elementary ontology for DNS zone files in order to show the benefits that such a data representation and organisation can bring to a well-established distributed system such as the domain name system. We also outline why such a solution can be deployed onto single servers without requiring a global changeover to the new system.

2 Motivation

In this section we will briefly discuss the underlying structure of the DNS and the reasons for using an ontology to map this data. We describe the core function of the DNS and, in this section and beyond, we describe how the logical structure of this information can be represented in an OWL-based ontology. There are many reasons for extending a system but it is essential that they are valid, such an extension does not damage the original function, and that they are meaningful, that this extension makes a positive contribution to the system. Section 3 lists the benefits of this approach, including the nature of the proposed extensions.

The origins of DNS, described in detail in RFC 1034 [3] and RFC 1035 [4], will not be further explored here other than to note that DNS is designed to be extensible, distributed, general purpose and capable of supporting local as well as global structure.

The DNS can provide name to IP address mappings (A records), IP address to name mappings (PTR records), canonical names (CNAME records) for IP addresses so that multiple names map to one IP address, facilities to support global e-mail (MX records), location of name servers (NS records) and, more recently, even the location of particular network services within a domain (SRV records) [7]. The key concept is that of abstraction: a user only needs to know a name and the correct host can be contacted once the DNS has resolved the name to an IP address.

DNS has three major components. These are:

- The domain name space and resource records, which are specifications for a tree structured name space and the data associated with these names.
- Name servers, which are server programs that provide information about a part of the tree structured name space. If they hold complete information for a part of the tree then they are an authority. Authoritative information is organised into zones, and is stored in zone files.
- Resolvers, which are programs that query name servers to resolve client requests. Resolvers are usually found at system level and are directly accessible by user programs.

The motivation for using an ontology to capture the conceptual relationship is, primarily, that the capture of DNS information in a semantically rich way allows the three components above to view and access data as required without having to encode the data in three different ways. The stored records, because they are annotated, can be displayed and accessed as most suits the application in question. Data, once written, can be read widely and shared easily. As discussed in the example from the previous section, the ontological framework allows the data to be used in both the way it was entered but, because the data now has both context and type information, it can also be re-interpreted to extend the knowledge contained in the system without having to capture any additional information.

3 Benefits

The major benefit of using any ontology is that it allows the relationships between data to be recognised and used by a wide range of applications and users. It also allows the expansion of an encoded knowledge domain through the use of inferred relationships - allowing a system to infer facts based on existing facts.

If a domain is seen as one entity with multiple attributes then it is useful to be able to extract the meaning of these attributes and use this meaning to assist with resource and service location and use. It is only recently that SRV records [7] in DNS have allowed the location of web servers and other services without having to use a *de facto* server name, such as www, to identify the host. SRV records are a meaningful extension of DNS to provide semantically rich metadata and shows that there is already a demand for the provision of such data as it abstracts away from a name dependent model to a feature oriented model.

OWL [9] is a logical way to provide a mechanism to capture and use more semantic metadata and does not require a DNS specific zone file parser once the domain of knowledge has been captured. The core functionality of the DNS can be stored in OWL and this 'reference' ontology can then be extended by local servers, or groups of servers, to provide the additional required behaviour. Note that even without extension, the type information and cardinality data associated with the entries in the DNS would provide far more information to the user and other applications than is currently available unless one resorts to reading the DNS server source code. Our expectations are that any additional overhead required to provide the ontological extensions would be offset by the additional functionality. It is also important to remember that caching is a strong component of the DNS and an initial query might have additional overhead but any further hits on the cache would not have that overhead. Our planned future work includes analysing efficiency and overhead issues in this system and will measure the comparative performance of enhanced servers. Service description languages such as OWL-S and WSDL can use such information to provide access to suitable services or act as wrappers for other services which are not yet metadata aware but can make use of the annotated DNS.

4 Method

It makes sense to start the iterative development of the DNS ontology classes from the top down because:

- The DNS class hierarchy is fairly simple since the domain is physically represented as zone files and zone files effectively serve as a container for resource records.
- Resource records (RRs), although referring to a single entity, do not strictly encapsulate each other. We can add RR types as we need them without having to know all of the available RR types. Given that one of our projected advantages is the ability to add additional, and hence unknown until later, RR types this is a significant fact.
- Even with a highly detailed classification of classes, the final depth of nested subclasses would be relatively shallow. Thus it can be sketched easily from a top down approach.

Space does not permit a full articulation of the mapping from the DNS to a DNS ontology so a short outline is provided, with an example of the mapping. The ontology describing the encoded subset of DNS used for this project is on-line [6] and an annotated version is also provided. The key issues in the production of the ontology were to first map the DNS as a knowledge domain and then determine what had to be mapped, how it was to be mapped and what would constitute a sufficient basis for the production of a prototype system without the need to incorporate all of the features of the DNS.

We have chosen to use OWL DL for expressing the ontology as OWL DL is guaranteed to be computable and decidable. Both of these characteristics are important to guarantee that an answer can be given to an enquiry. We have chosen to produce an initial, proof of concept, implementation of the DNS ontology omitting, for the time being, a number of more advanced concepts articulated in later RFCs. These more advanced concepts build on the core foundation of the DNS and do not require any conceptual changes to be made to the system if added later.

The domain of the ontology is derived from the DNS RFCs that define the zone files. We now consider the process of constructing the ontology. It is necessary to build the

ontology from scratch as there is currently no existing ontology which describes DNS zone files. Although the Dublin Core Metadata Initiative [10] could be used in this ontology, it requires an including ontology to be written in OWL Full and cannot be incorporated in an OWL DL ontology.

The important concepts that must be included in the ontology are DNS domains, zones, resource records and the types of these resource records (A, PTR, NS, SOA, CNAME, TXT, SRV, etc).

The classes and class hierarchy are based around DNS domains as the top level, which contain resource records. The resource record types all sit on the same conceptual level so the overall class hierarchy is shallow.

The properties that relate the classes, and subsequent instances, are critical to the ontology as a great deal of the DNS ontology information is encoded as properties since they can best be represented as the relationships between classes and their members. For example, an IP address is represented as a property that is found within resource records and has a value that takes the form of an IP address. The facts of the properties establish the type, cardinality and range of values and these must also be defined to allow static checking of the data and also to enable interoperability between different local ontologies.

Finally, instances are created which capture the data for a given entity. The instances provide flesh to the underlying ontology.

As an example of capturing data in the final ontology, consider the information associated with a single node in the domain 'example.com'. The basic information to capture is for the example domain 'example.com' (this information does not reflect the data for a real 'example.com' and is used for illustration). Within this there would be a set of zone files (at least one on the primary server). There is also a record for the IP address 12.20.40.77 which has the name 'an.example.com'. This machine is also known as 'www.example.com' and has an Internet Class value of 'IN'.

We create an instance of a resource record and call it 'an.example.com'. This has a property called 'locatedIn', which is transitive, and places it within the domain. Hostname and IP address properties are also defined, along with the Internet Class and CNAME properties

The OWL representation of the 'locatedIn' property can be seen in Figure 1 to illustrate how object properties are used within the defined classes to enable the capture of DNS information.

```
<owl:ObjectProperty rdf:ID="locatedIn">
<rdf:type rdf:resource="&owl;Transitive
Property" />
<rdfs:domain rdf:resource=
"http://www.w3.org/2002/07/owl#Thing" />
<rdfs:range>
<owl:Class>
<owl:unionOf rdf:parseType="Collection">
<owl:Class rdf:about="#Domain"/>
<owl:Class rdf:about="#Zone"/>
</owl:unionOf>
</owl:Class>
</rdfs:range>
</owl:ObjectProperty>
```

Figure 1. OWL representation of the locatedIn transitive property.

5 Extensions to DNS

The DNS is strongly controlled (through the RFC mechanism) as to which resource records may be defined and how they are defined. This strong control is to ensure interoperability between different servers and clients. However, under the current DNS modification scheme, this control also means that an RFC process has to be followed to bring about a global change and this is a slow process. With an OWL encoding of the DNS system, as described above, DNS modification is no longer a centralised, rigidly controlled and slow process while, at the same time, it can still provide a core functionality based upon the RFCs if desired. There is no reason why more resource record types could not be encoded in OWL and used to extend local functionality. An ontologically enhanced server only needs a new ontology to provide additional services to its user base. Two sites can collaborate without having to carry out detailed recoding and testing and, importantly, completely outside of the RFC mechanism. Potentially, the use of semantically enhanced systems can interact with the ontologically enhanced DNS to make ad-hoc changes as required by the system agents without having to involve any human agents at all. Such a system must employ strict controls and security to avoid compromise or exploitation.

An ontology also provides an excellent mechanism for advertising the structure and relationships of such new resource record types, and the existing DNS SRV [7] mechanism provides a means for advertising which new services are available. SRV records are already used in the existing DNS and allows ontological advertisements to be piggybacked without requiring all servers to be globally updated to handle a new ontological advertisement protocol. Our ontologically-based system with local extensions can coexist with traditional DNS services because of the transparency requirements described in RFC 3597 [8]. This RFC removes the requirement for nameservers to have the same software version, or capabilities, as the servers that they are exchanging data with.

With the introduction of extensions, it is important to maintain the core functionality of the DNS in order to allow interoperability and the continued functioning of one the world's critical distributed information systems. Our vision, firstly, is of a core ontology which encodes the RFC-prescribed DNS elements. Then, any extension ontologies start from that point and extend the core ontology. Thus two enhanced servers, even with different extensions, can still use the standard elements of the DNS even if they cannot understand each other's extension model.

A possible use for such an extension is the provision of additional information channels for local sites. Our proposed ITXT RR is a TXT record which is only viewable from within a given site. ITXT records can only be viewed from hosts within the domain that the nameserver is an authority for. The IP address of the host is resolved to a name and checked for spoofing. It is the role of network security to ensure that only hosts which are from within the network appear to be from within the network to the nameserver. Some network administrators already hand-annotate the zone files to include additional textual data but, for security reasons, this information is generally stored as comments in the file rather than data that can be accessed via the DNS. Such information could include the MAC address or office location of a given machine.

The ontology can also be used for reasoning, such that a query class could be introduced which extracted information from the ontology without having to directly access the DNS. This would improve the maintenance and data-mining capabilities of large sites since they could carry out context-rich searches across their own data without having to resort to text searches in editors or use large overhead RDBMS systems to store their data.

6 Conclusions

This paper has provided an elementary DNS zone file ontology. It has shown that, using relatively simple design tools and a straight-forward top-down approach, it is possible to capture the key aspects of the DNS zone file in a machine-interpretable way. It has also shown that, with the correct design approach, an ontological approach is suitable for this application.

The reason that we chose to use OWL is based on the increasing use of OWL for similar projects, the existence of tools (such as Protege) for editing OWL and the large community that is working to make OWL a good choice for ontologies. OWL, in its XML form, is also a good fit for the Semantic Web as it can be handled and manipulated using existing semantic web technologies - including using HTTP as a transfer mechanism.

This shows the benefits of using ontologies, even in existing systems, and how such systems can be represented in a way that allows them to develop new and interesting features, decades after they were initially introduced. It also shows that ontology languages, such as OWL, have reached a level of maturity where they can be effectively used without the user having to be an AI specialist or language designer.

References

[1] Barta, R,'Virtual and Federated Topic Maps,' *Proceedings of XML Europe 2004*, 18-21 April, Amsterdam, 2004.

[2] Chen, C-S et al, 'Building a DNS Ontology using METHONTOLOGY and Protege-2000' *Proceedings of 2002 International Computer Symposium*, Workshop on Artificial Intelligence, Vol.2, pp.1853-1860, Taiwan, 2002.

[3] Mockapetris, P., 'RFC 1034: Domain Names–Concepts and Facilities', 1987, *available from* http://www.dns.net/dnsrd/rfc/

[4] Mockapetris, P., 'RFC 1035: Domain Names–Implementation and Specification', 1987, *available from* http://www.dns.net/dnsrd/rfc/

[5] Vixie, P., 'RFC 2671: Extension mechanisms for DNS (EDNS0)', 1999, *available from* http://www.dns.net/dnsrd/rfc/

[6] Falkner, N., 'An ontology for DNS zone files', 2004, *available from* http://www.dhpc.adelaide.edu.au/ontologies/dns.html

[7] Gulbrandsen, A., Vixie, P. and Esibov, L., 'RFC 2782: A DNS RR for specifying the location of services (DNS SRV)', 2000, *available from* http://www.dns.net/dnsrd/rfc/

[8] Gustafsson, A.,'RFC 3597: Handling of Unknown DNS Resource Record (RR) Types', 2003, *available from* http://www.dns.net/dnsrd/rfc/

[9] Smith, M. K., Welty, C.and McGuinness, D. L., 'OWL Web Ontology Language Guide', 2004, *available from* http://www.w3.org/TR/2004/REC-owl-guide-20040210/

[10] Kokkelink, S. and Schwanzi, R., 'Expressing Qualified Dublin Core in RDF/XML', 2002, *available from* http://dublincore.org/documents/2002/05/15/dcq-rdf-xml/