On the Role of Ontological Semantics in Routing Contextual Knowledge in Highly Distributed Autonomic Systems

John Keeney, David Lynch, David Lewis, Declan O'Sullivan

Knowledge & Data Engineering Group,

Department of Computer Science,

Trinity College Dublin,

Dublin, Ireland.

{John.Keeney | Dave.Lewis | Declan.OSullivan }@cs.tcd.ie

Abstract

Much recent research has focused on applying Autonomic Computing principles to achieve constrained self-management in adaptive systems, through self-monitoring and analysis, strategy planning, and self adjustment. However, in a highly distributed system, just monitoring current operation and context is a complex and largely unsolved problem domain. This difficulty is particularly evident in the areas of network management, pervasive computing, and autonomic communications. This paper presents a model for the filtered dissemination of semantically enriched knowledge over a loosely coupled network of distributed heterogeneous autonomic agents. It also presents an implementation of such a Knowledge Delivery Network, which enables the efficient routing of distributed heterogeneous knowledge to, and only to, nodes that have expressed an interest in that knowledge for use as the operational or context information monitored in order to analyse to the system's behaviour as part of an autonomic control loop.

1. Introduction

Autonomic systems use knowledge of their operational state and operational context to self-manage, i.e. to self-configure, self-heal, self-optimise and self-protect, by monitoring state and context, planning and adapting. Though the need to self-manage was initially recognised as a challenge in dramatically reducing the operating costs of complex computing systems [15], increasingly complex networked systems are also seen as needing to self-manage. Applying autonomic approaches to networks and other highly distributed systems represents particular challenges in gathering operational knowledge from across the system, where operational network knowledge is operational state information about the system, and is accompanied by its meta-data, e.g. expressed as a management information model. The challenge arises because in a highly distributed autonomic system, the system elements that possess this knowledge are widely distributed, are purchased from different vendors, perform different functions, possess a wide range of knowledge meta-data and are operated by different organisations. Clark et al identified the central role of a knowledge-driven approach to support advanced AI techniques for monitoring and analysing Internet conditions in order to drive the planning of optimisation, protection or corrective strategies [12]. A variety of proposed autonomic solutions are using explicit knowledge models at run time to dynamically discover, handle and reason over newly encountered information. In particular standardised approaches to expressing ontological knowledge, as proposed by the Semantic Web community [2] are proving effective in implementing flexible autonomic solutions using AI planning [29], multi-agent [3] or other intelligent knowledge driven techniques [1]. This approach promises loose semantic coupling between autonomic applications, which is vital as new waves of applications increasingly rely on using the information and services offered by existing heterogeneous distributed applications. There has been some interest recently in developing middleware for accessing operational knowledge as the context for adaptive or autonomic systems in an ontological form [22][33]. To date, however, there has been no movement towards an inter-working consensus for these technologies or on how the knowledge required to make autonomic decisions is gathered from across a heterogeneous network, and particularly across administrative domains. In [16] we argued the need for an autonomic knowledge delivery service that can inherently scale to the size of the system it supports, including to Internet scales. To be a reliable medium for the dissemination of the knowledge needed by autonomic functions the implementation of a knowledge delivery service clearly needs to itself exhibit self-management.

In this paper we begin to address the challenge of establishing an Internet-scale knowledge delivery service for distributed, autonomous autonomic systems. Meeting this challenge demands we address both the extreme heterogeneity and rapid evolution of autonomic applications and context information, in combination with the need for high throughput, low-latency of messages between large, volatile populations of service clients. Clearly, any software-based event forwarding algorithm will struggle to match the hardware optimised performance of packet forwarding in IP routers. In this paper we describe the introduction of ontological reasoning into a content-based event delivery mechanism, measure its performance and discuss the implication for future event routing approaches. Basing the forwarding algorithm on today's ontological reasoners incurs a heavy computational load. We do not attempt to develop optimised reasoners for KBNs, instead we aim to explore the performance of ontological reasoning to better understand how it can effectively deployed in a knowledge delivery service. Ultimately we hope this will guide the evolution of intelligent clustering in event routing algorithms that are cognisant of the performance profiles of existing reasoners and of the semantics being exchanged by client applications, and can thereby off-set this relatively poor forwarding algorithm performance.

2. State of the Art

Publish-subscribe (Pub-Sub) event systems [20] might be considered as the basis for the proposed knowledge delivery service as they avoid close coupling between producers of events and one or more event consumers that have expressed an interest in an event type. Currently, publish-subscribe systems, e.g. IBM MQSeries, are used widely as a communication bus to flexibly integrate business functions. However, such Pub-Sub systems require agreements on message types between the developers of producer and consumer applications. This places severe restrictions on the heterogeneity and dynamism of client applications.

Pub-Sub systems that filter events based on matching client subscriptions to message attributes rather than the full message type, known as Content-Based Networks (CBN), facilitates still looser coupling between producer and consumer applications. Several CBN solutions and prototype exist, e.g. Siena [7], ELVIN [30], HERMES [28], XNET [9] and Gryphon [34], however their scalability is not yet proven to Internet scales. Widespread CBN deployments have been slow to emerge partly due to the difficulty in reaching a general compromise between the expressiveness of event types and subscription filters and the need both to match these efficiently at CBN nodes and to efficiently maintain forwarding tables by aggregating new subscriptions with any existing ones that cover a superset of matching messages [6]. As a result current CBNs only support a very limited range of data types and operators (typically integers, strings, Booleans), which falls well short of supporting the heterogeneity and flexibility that an autonomic knowledge delivery service requires. Selecting a more expressive language involves a difficult trade-off, since higher level features, e.g. set functions, introduce more complexity into a CBN node, and may only be of use to a subset of applications. We must aim therefore to have a CBN message and subscription language that can be expanded incrementally to meet the requirements of specific autonomic application domains without placing unnecessary overheads on the network as a whole.

A CBN based on messages containing semantic mark-up and queries is potentially far more flexible, open and reusable to new applications. We call such a semantic-based CBN a Knowledge-Based Network (KBN), and we propose this be the mechanism by which the Knowledge Delivery Service be implemented

Recent experimentation by the authors has evaluated the performance of a basic KBN based on the integration of mapping based semantic interoperability with the Elvin CBN [14]. In this paper we examine the integration of ontological equivalence and subsumption into the SIENA CBN event/subscription matching algorithms.

3. KDSv1: An Extension of the Siena Content Based Networking System

Though our initial measurements described in [14] used the Elvin CBN, this was a centralised system and our scalability goals required us to consider a decentralised CBN scheme. The design presented here we opted to build upon the Siena CBN [7] due to source code availability and an abundance of associated technical reports and papers, and in addition, its focus on expressiveness in a wide-area distributed environment.

A Siena notification is a set of typed attributes. Each attribute is a triple consisting of a name, type and a value, where the type is limited to one of "string", "time", "float" and "integer". A filter is constructed from a set of constraints which are each applied to the content of notifications. A constraint is a triple, consisting of the attribute name, a constraint operator, and a value. Where multiple constraints exist in a single filter they are evaluated as a conjunction. A filter "covers" a notification or event if that event satisfies each constraint applied to it by the content filter. An event or notification **n** is delivered to an interested party **X** if **X** has submitted a subscription filter that covers the notification. Also, a filter **f** "covers" another filter **f′** where together the set of constraints in **f** are more general than all of the individual constraints in **f′**, and so all of the notifications that would be delivered or forwarded for **f′** would also be delivered or forwarded for **f**, i.e. **f** is more general than **f′**.

Optimising Subscriptions and Notifications

In the current implementation of Siena, notification routers are arranged in a hierarchy of nodes, where each node maintains a tree structure that keeps track of subscriptions and so informs the notification forwarding strategy for that node. In this tree structure general subscriptions are at the top and more specific covered subscriptions are arranged as subtrees.

Each node in the hierarchical topology may have any number of incoming connections, other than clients, but only one outgoing connection to its parent node. Conceptually, the nodes have a client server relationship. Thus, a hierarchical node need only propagate information it receives to its parent node in the form of root subscriptions and publications. The main routing principle behind Siena is to push notifications as close as possible to parties that may be interested in that information. Known as downstream replication, this can be achieved both by subscription forwarding and advertisement forwarding. Subscription forwarding is the method used for routing in the Siena hierarchical implementation.

The tree of subscriptions is used to assist in pruning the number of subscriptions forwarded. Essentially, root subscriptions are the only ones sent. As such, subscriptions covered by previously forwarded subscriptions are pruned and network traffic is kept to a minimum. In order to ensure consistent notification across the network, Siena employs publication forwarding to master nodes, and leaves further notification beyond that of root subscriptions to the nodes on which the more specific subscriptions reside.

When the Siena node acting as the server to a notification producer **X** receives a subscription filter **f** from **X**, the subscription tree is searched starting at each root subscription. If a subscription is found that covers the filter **f** and contains **X** in its subscriber set the search terminates. Otherwise, if the filter **f** already exists in the subscription tree, **X** is simply placed in the subscriber set of that particular filter. Finally, should neither of these apply a new subscription is inserted under the most specific covering filter, possibly a leaf node, with **X** added to its subscriber set. If no covering filter exists, the subscription is inserted as a root subscription. All root subscriptions are forwarded to master nodes right to the top of the Siena node hierarchy, with sub nodes acting exactly like subscribers.

Upon reception of notifications at a Siena router node (either from the notification producer or a super-node) the set of clients or other sub-nodes with subscription filters covering the notification are sent that notification. If the master server was not the source of the notification than a copy of this notification is also sent to the master server. In fact, the relationship between a Siena node and its master is very similar to that of a subscriber client and the Siena node itself. The net effect of this is that no matter where a publication, or subscription, takes place on the network the correct subscriber subset is notified.

Extending the Siena Subscription Language

One of the primary contributions of the design of this implementation is to enhance the Siena subscription language. The main change to the subscription language was the addition of three new ontological operators: Subsumes, Subsumed by, and Equivalent. The subsumption relationship describes how an ontological entity is more general than another ontological entity. For example, as seen in the Wine ontology [38] (Figure 1), the ontological type "wine" subsumes the type "white wine", or "white wine" is subsumed by "wine" since "wine" is less specific than "white wine". Equivalence refers to the relationship between two ontological types that refer to the same type of entity yet may be different ontological classes.

As seen in the Wine ontology, an excerpt of which is shown in Figure 1 after it has be reasoned over by the Pellet ontology reasoner [25], the class "DryWine" has been found to be equivalent to "TableWine", and so share subclasses, despite the absence of an explicit statement stating this equivalence. If an event consumer was interested in receiving events with some field containing the name of a wine ontology class, where the class is less specific than "CotesDOr" but more

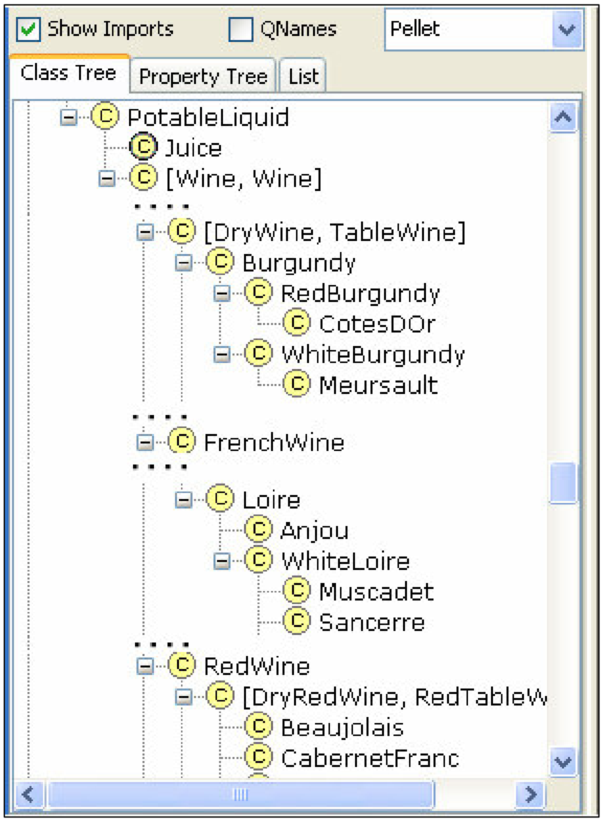

Figure 1: A graphical representation of an excerpt from the Wine ontology

specific than "Burgundy" than that subscriber would receive events where the specific field contained the name of the "RedBurgundy" class or an equivalent class. While this may seem to make the subscription specification more difficult for simple subscriptions, the advantages become apparent for more extensive ontologies. In addition, since the standard subscription language for Siena, and most content-based networking systems, allow filters to be defined using base data types, and only as a conjunction of filters (i.e. filter constraints are combined using the Boolean AND operator and so the failure of one constraint in a filter results a match failure for that filter), the specification of flexible subscriptions using ontological classes would entail the specification of multiple individual subscriptions to match for each class type specified as a string comparison, with no inbuilt consideration for equivalent classes.

Maintaining the Subscription Tree

While remaining at an abstract level it is necessary to discuss enhancements and modifications to the Siena subscription tree structure and subscription forwarding architecture at the design stage. The main consideration behind enabling ontology based subscriptions in such a manner is the preservation of the covering relation between filters. In particular, the partial ordering between subscriptions within the subscription tree structure must be maintained. In order to accomplish this we must define a covering relation between our enhanced subscriptions.

Consider two filtering constraints ***A*** and ***B***, such that ***A*** is given as (**x** *op* **a**), and ***B*** is given by (**x** *op* **b**), where *op* is one of EQU (equivalent to), MORESPEC (more specific than, or is subsumed by), or LESSSPEC (less specific than, or subsumes). The variable **x** is the variable for the field in each notification to be compared to the constant ontology class names **a** or **b**, given in the filter specification. Table 1 describes when filter constraint **A** covers filter constraint **B,** i.e., when the set of possible notifications matching filter constraint **A** is a superset of the set of notifications matching filter constraint **B**. In this design it should be noted that the subsumption and reverse subsumption relationships between two classes do not hold if they are equivalent, i.e. if class **a** is equivalent to class **b**, then **a** is not more or less general than **b**.

A Covers	*B*	*iff*	
x EQU **a**	**x** EQU **b**	***never***	*1*
x MORESPEC **a**	**x** EQU **b**	*if (* **a** LESSSPEC **b** *)*	*2*
x LESSSPEC **a**	**x** EQU **b**	*if (* **a** MORESPEC **b** *)*	*3*
x EQU **a**	**x** MORESPEC **b**	***never***	*4*
x MORESPEC **a**	**x** MORESPEC **b**	*if (* **a** LESSSPEC **b** *)*	*5*
x LESSSPEC **a**	**x** MORESPEC **b**	***never***	*6*
x EQU **a**	**x** LESSSPEC **b**	***never***	*7*
x MORESPEC **a**	**x** LESSSPEC **b**	***never***	*8*
x LESSSPEC **a**	**x** LESSSPEC **b**	*if (* **a** MORESPEC **b** *)*	*9*

Table 1: Covering relationships between new Siena ontological operators

A number of observations can be drawn from Table 1 that may not be immediately obvious. Lines 1, 5 and 9 show that a constraint does not cover itself or an equivalent constraint. This is to avoid the situation where ***A*** covers ***B*** and ***B*** covers ***A***, which would lead to circular references and infinite looping in the optimisation of a node's subscription tree. It should also be noted that (**x** MORESPEC **y**) is equivalent to (**y** LESSSPEC **x**). For any filter **f** with multiple filtering constraints combined as a conjunction, **f** is covered by **f'** only if all of the filtering constraints in **f** are covered by filtering constraints in **f'**. The covering relationships for the other Siena operators are given in [7][32], and remain completely unchanged by the addition of the three new operators described here.

4. Effects and Evaluation

In order to demonstrate the effects of adding support for ontological operators to the Siena subscription language a number of factors were evaluated. These include: the time taken to load, parse, and reason over a number of ontologies; the effect on scalability and end-to-end time of incorporating ontological lookups in the notification forwarding algorithm; and comparing a sample ontological subscription to an equivalent subscription which only operates on class names using string comparison operations.

We envisage that our initial distributed autonomic system will rely on the same management information bases that network and enterprise management system use currently. For this reason we have used the Distributed Management Task Forces Common Information Model CIM [11] as a standardised example of management knowledge. In CIM, when a management event occurs, its occurrence is signalled to a registered set of interested parties by the creation and dispatch of an "indication" objects to those parties. These indication types, as with all CIM classes, are specified in CIM Managed Object Format (MOF) formatted files, which essentially act as management information models, and which are then loaded and parsed by a CIM object manager component (CIMOM).

For these experiments, a new indication type JK_SampleEvent was defined as seen in Figure 3. This event is inherited from the standard indication type CIM_AlertIndication, an indication type used to describe error or alert type events.

```
#pragma include ("CIM_Core27.mof")
#pragma include ("CIM_Event27.mof")
[Indication, ... , Description ("A new event!")]
class JK_SampleEvent : CIM_AlertIndication{
  [Description ("A string variable")]
  string StringVar;
};
```

Figure 3: Excerpt of the MOF file declaring the JK_SampleEvent CIM indication

Pioneering work from Universidad Politécnica de Madrid (UPM) [18] demonstrates the value of modelling management information models in the OWL ontological format to support interoperability between models originally conceived in different management information languages. By making use of a conversion utility resulting from this research, the MOF file described in Figure 3 was converted to an OWL ontology that includes the imported CIM Core Model and the CIM Event Model, an excerpt of which is shown in a graphical format in Figure 2.

For the purposes of evaluating the overhead involved in the load time parsing and inference of an ontology, three ontologies were compared with three levels of reasoning. Firstly the very simple Service ontology [31], with only four classes and no individuals, shown

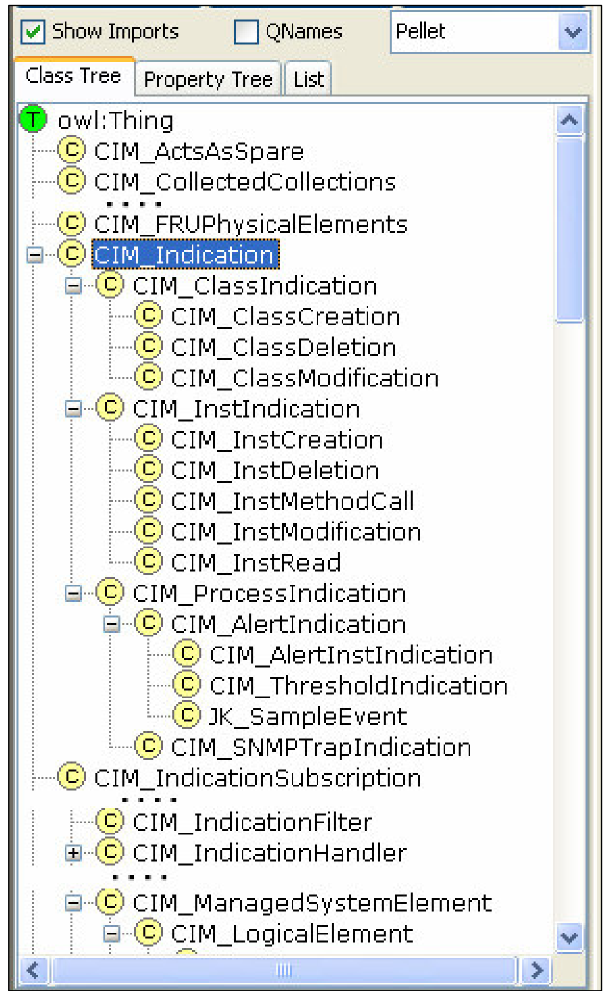

Figure 2: A graphical representation of an excerpt from the JK_SampleEvent ontology

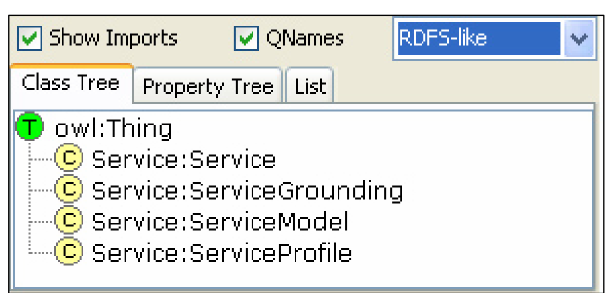

Figure 4: A graphical representation of an excerpt from the OWL Service ontology

in graphical format in Figure 4; next the complex Wine ontology, already shown in Figure 1, with 138 classes of which 61 are imported from another ontology, and 206 individuals of which 45 are imported; and finally the large but relatively simple JK_SampleEvent ontology discussed above, with 147 classes and 563 individuals, shown in Figure 2. These ontologies were loaded, parsed, and reasoned over using the Jena framework [13] with three different reasoners. The first reasoner, "OWL_MEM_NONE", supplied with Jena, performed no reasoning. The second "OWL_MEM_RDFS_INT", also supplied with Jena, performs RDFS entailment reasoning. The third reasoner, Pellet [25] performs full OWL DL reasoning. The results of these comparisons are given in Tables 2 and 3.

As can be seen from these results, the operations to load an ontology, and especially reason over its contents, are expensive operations. However, in the case where the set of ontologies to be used are known a priori, the loading and reasoning can be performed at initialisation time rather than during the operation of the system. When the set of ontologies used changes during runtime, such changes must be minimised to maintain satisfactory performance. In addition to the size of an ontology, the time to reason over an ontology is dependent on the level of reasoning required for correct interpretation of the ontology, which can be dependent on the complexity of the particular ontology. For a complex ontology, such as the Wine ontology, a more functional reasoner like Pellet is required to obtain a correct class hierarchy, however, for a relatively simple ontology, the full support of Pellet is not required to obtain a correct class hierarchy and can be provided by the less functional OWL_MEM_RDFS_INT reasoner, with a substantial time saving. For this reason it is necessary to carefully tune the specific level of reasoning required to each specific ontology on an application by application and a case by case basis. Further information on the comparative performance of a number of reasoners is available from [26].

Loading Only	Service	Wine	JK_SampleEvent
OWL_MEM_NONE	57.60 ms *std. dev. (9.21)*	338.84 ms *std. dev. (15.41)*	430.67 ms *std. dev. (11.78)*
OWL_MEM_RDFS_INT	55.37 ms *std. dev. (4.38)*	346.38 ms *std. dev. (10.91)*	444.68 ms *std. dev. (15.00)*
PELLET	58.91 ms *std. dev. (11.46)*	364.48 ms *std. dev. (18.20)*	441.5 ms *std. dev. (18.50)*

Loading and Reasoning	Service	Wine	JK_SampleEvent
OWL_MEM_NONE	52.63 ms *std. dev. (2.27)*	329.64 ms *std. dev. (9.95)*	455.38 ms *std. dev. (24.88)*
OWL_MEM_RDFS_INT	61.32 ms *std. dev. (2.67)*	366.56 ms *std. dev. (6.93)*	579.35 ms *std. dev. (29.40)*
PELLET	97.98 ms *std. dev. (9.69)*	1391.39 ms *std. dev. (29.30)*	1064.54 ms *std. dev. (48.18)*

Tables 2 and 3: The times taken to load, parse and reason over three different ontologies using three different reasoners

To further evaluate the impact of adding ontological operations to the subscription matching and notification forwarding algorithm in each Siena node, it was necessary to determine how such operations affect the scalability of the Siena network and end-to-end time taken for notifications to be delivered.

For this experiment an open source CIM object manager (CIMOM) [37] was extended to additionally publish a standard Siena notification each time an event occurred. This notification message included the ontological class name of the particular CIM event indication instance created to signify the occurrence of the event. This Siena notification was then published to a testbed Siena network. The ontology used was the JK_SampleEvent ontology discussed earlier and shown in Figure 2. A notification subscriber with specific interest in CIM event indications was then connected at varying locations within the Siena network in a manner to force the Siena notification to traverse a specific number of Siena router hops.

Firstly a simple single subscription filter was created to subscribe to all CIM event indications by subscribing to all notifications where the ontological class name was more specific (MORESPEC) than CIM_Indication, since as can be seen from Figure 2 all CIM event indications are sub classes of CIM_Indication. This subscription requires that at every Siena node, for every message containing a field with the ontological class name of a CIM event indication, the JK_SampleEvent ontology must be queried to determine if the event should be forwarded towards the subscriber. The end-to-end time for notification delivery and the scalability consequences of these operations are presented in Figure 5.

Secondly, in order to duplicate the same experiment without the use of ontological operators, a number of equivalent string based subscriptions were formed. Since string comparisons for multiple strings in a single subscription filter constraint are not supported in a single Siena subscription filter, multiple filter constraints are required. Furthermore, since multiple filter constraints in a single subscription filter are joined by conjunction (using the Boolean AND operator), a disjunction of constraints (using the

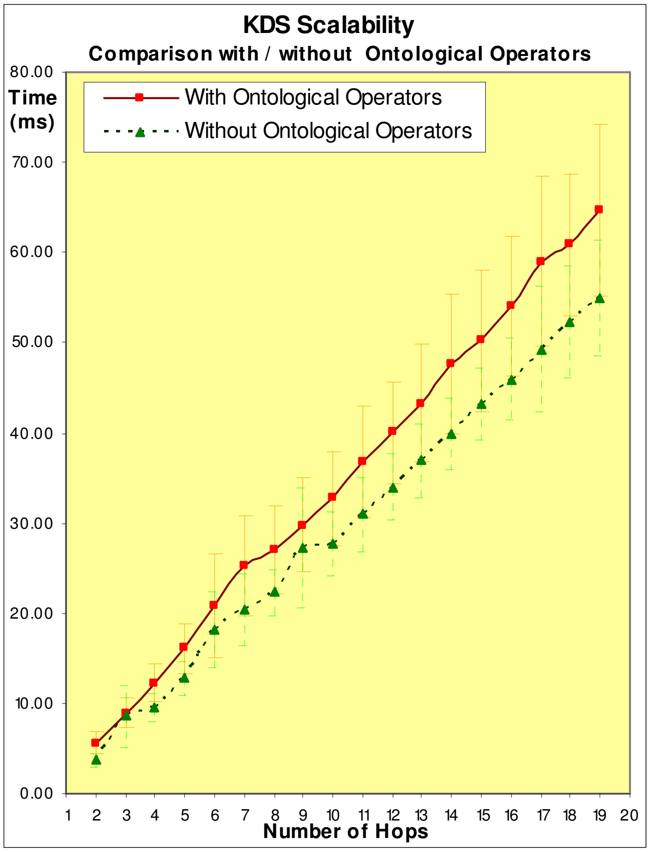

Figure 5: KDS Scalability and notification end-to-end timing

Boolean OR operator) can only be specified using multiple distinct subscriptions. This requires that in order to subscribe to notifications containing names of any of the 16 subclasses of CIM_Indication shown in Figure 2, 17 distinct subscriptions are required, each causing churn within the Siena network as described in section 3 above. The end-to-end delivery times of the same CIM event notifications, to a similar client, subscribing using these 17 subscription filters instead of the single ontological subscription, are also given in Figure 5 as a comparison to the use of ontological operations in the subscription and forwarding algorithm. Furthermore, if a new subclass of CIM_Indication is added at runtime, in addition to modifying the ontology at each Siena router node, the subscription code for each subscriber application will then need to be changed to explicitly subscribe to that new class, rather than allowing it to be automatically categorised for the ontological operators.

5 Discussion

As can be seen from the results presented in the previous section incorporating ontological operators into the Siena subscription language results in a small but definite performance decrease caused by the use of ontological queries in the forwarding algorithm of each Siena node, in the subscription algorithm, and in the loading, parsing and reasoning of the ontologies themselves. However, in normal operation, this performance decrease can be seen as acceptable when offset against the increased flexibility of the system, particularly over a short number of hops.

A number of other observations can also be drawn. Firstly, the loading of new ontologies into a reasoner embedded in a KBN node is computationally expensive due to load-time inference, so the frequency of additions to the ontological base of a given KBN node must be minimised. Secondly, ontological reasoning is memory intensive and memory usage is proportional to the number of concepts and relationships loaded into the reasoner so reasoning latency can be controlled by limiting this number in any given KBN node. However, once loaded and reasoned over, the querying of such an ontological based is relatively efficient, with performance relative to size of the ontological base. These axioms will therefore form the basis of semantic clustering policies used to partition the routing mechanism, as discussed below.

6. Related Work

There has been little examination of the use of ontology-based semantics in content-based networking in the scientific literature. In [27], an extension to the Toronto Publish/Subscribe System (ToPSS) is described that proposes extending the event/subscription matching function of this CBN to include class equivalence, ontological sub-class and super-class relationships (i.e. subsumption) and semantic mapping based relationships, which is equivalent the CBN extensions carried out in [14] and in this paper. More significantly, however, no report of an implementation or evaluation of this proposal has yet emerged. In [17] a semantic publish/subscribe system is presented, but it is based on a centralised pub/sub bus implementation and thus is limited to enterprise scale and does not offer true CBN capabilities.

Considering approaches to route management in CBN, the simplest approach is flooding, where a node requests all other nodes for relevant routes, but this is not scaleable to large numbers of nodes [21]. This is addressed in the Siena CBN through the static set up of spanning trees [8] from producers to all possible consumers. However, these are then costly to recalculate in the event of configuration change or failures, thus failing our requirements for robustness and self-configuration. The HERMES CBN [28], ToPSS [23] and the REBECCA CBN [36] have all addressed these issues by applying peer to peer distributed hash table (P2P DHT) mechanisms to the formation of routing tables in CBN nodes. P2P DHTs such as CAN [5], CHORD [10] and Pastry [24] have well known properties of scalability, robustness and self-organisation. It should be noted that though P2P systems are concerned with efficiently routing queries to matching information sources, they not address the CBN concern of optimally routing a sequence of asynchronous replies back to the querying, or in CBN terms, the consuming client. P2P DHTs provide efficient routing by using a cost metric keyed to the physical topology of the network resulting in average hop-counts for a route in the order of the log of the number of nodes in the network i.e. O(log(N)). It is the demonstrated strengths of DHT-based routing protocols for CBNs that indicate the appropriateness of peer-to-peer Semantic Overlay Networks as a routing mechanism that meets our requirements for an Internet-scale KBN.

There are several attempts at applying P2P DHT techniques to the retrieval of distributed ontology encoded knowledge information, e.g. in RDF, in semantic overlay networks [35][4][19]. In supporting an ontology-driven DHT-based P2P routing mechanism for the KBN, the approach outlined in [19] seems most promising due to its support for peer clustering.

7. Conclusions and Further Work

This work is significant in addressing issues of attaining Internet scalability in the use of the standardised ontological semantics to support a highly expressive knowledge delivery service for autonomic systems. By supporting arbitrary semantics in the structuring of messages and the construction of consumer subscriptions, the KBN would provide a stable basis for the long term evolution of new autonomic solutions.

In our future work we aim to develop a dynamic P2P based routing infrastructure for our KBN which will use policy-based clustering to reduce the reasoning load at any one routing node in a way tailored to the performance characteristics we are starting to gather. This will also involve characterising a wider range of reasoners in this role. By addressing asynchronous messaging over a highly decentralised network this work uniquely attempts to reconcile Internet engineering values and knowledge engineering solutions, thereby exploiting the new efficiencies yielded by clustering KBN nodes based on semantic distance. Clustering thereby both increases the scalability of ontology-based routing and supports the deployment of routing schemes tailored to specific application domains, thus allowing a wide range of strategies to co-exist.

Acknowledgements

This work was partially funded by the EU under the ACCA project, by Science Foundation Ireland under the CTVR project, and by the Irish Higher Education Authority under the M-Zones programme.

References

[1] Abecker, A., Stojanovic, L., Stojanovic, N., Studer, R., "Ontology-based correlation engine", in proc 1st Int'l Conference on Autonomic Computing, New York, 2004, pp 304-305

[2] Berners-Lee, T., Hendler, J., Lassila, O., "The Semantic Web", Scientific American, May 2001

[3] Bonino, D., Bosca, A., Corno, F., "An agent based autonomic semantic platform", in proc 1st Int'l Conference on Autonomic Computing, New York, 2004, pp189-187

[4] Cai, M., Frank, M., "RDFPeers: A scalable distributed RDF repository based on a structured peer-to-peer network", in proc of WWW conference, May 2004, NewYork, USA, pp 650-657

[5] Ratnasamy, S., Francis, P., Handley, M., Karp, R., Shenker, S., "A scalable content addressable network", in proc of the 2001 Conference on applications, technologies, architectures, and protocols for computer communications. ACM Press New York, NY, USA, 2001.

[6] Carzaniga, A., Rosenblum, D., Wolf, A.L., "Challenges for Distributed Event Services: Scaleability vs. Expressiveness" in proc of Engineering Distributed Objects (EDO '99), ICSE 99 Workshop, Los Angeles CA. May, 1999.

[7] Carzaniga, A., Rosenblum, D. S., and Wolf, A. L., "The Design and Evaluation of a Wide-Area Event Notification Service", ACM Transactions on Computer Systems, Vol. 19, Issue 3, August 2001

[8] Carzaniga, A., Wolf, A. L., "Forwarding in a Content-Based Network" in proc SIGCOMM'03, August 25-29 2003, Kaelsruhe, Germany, ACM Press

[9] Chand, R., Felber, P.A., "A Scalable Protocol for Content-Based Routing in Overlay Networks", Second IEEE International Symposium on Network Computing and Applications, April 2003, Cambridge, MA

[10] Stoica, I., Morris, R., Liben-Nowell, D., Karger, D., Kaashoek, M. F. , Dabek, F., Balakrishnan, H., "Chord: A scalable Peer-to-Peer Lookup Service for Internet Applications", IEEE Transactions on Networking, Vol. 11 Issue 1:17–32, 2003.

[11] Common Information Model v 2.10.1, DMTF 2005: http://www.dmtf.org/standards/cim/cim_schema_v2101

[12] Clark, D., Partridge. C., Ramming, J.C., Wroclawski, J.T. "A Knowledge Plane for the Internet", in Proc. of SIGCOMM'03, 25-29 August 2003, Karlsruhe, Germany.

[13] Carroll, J., Dickinson, I., Dollin, C., "Jena: Implementing the Semantic Web Recommendations", in

Proc. of World Wide Web Conference 2004, 17-22 May 2004, New York, NY, USA. http://jena.sourceforge.net/.

[14] Keeney, J., Lewis, D., O'Sullivan, D., Roelens, A., Boran, A., Richardson, R., "Runtime Semantic Interoperability for Gathering Ontology-based Network Context", to appear in proc 10th IEEE/IFIP Network Operations and Management Symposium (NOMS 2006), April 3-7, 2006, Vancouver, Canada.

[15] Kephart, J., Chess D., "The vision of autonomic computing," IEEE Computer, vol. 36, no. 1, pp. 41–50, 2003.

[16] Lewis, D. O'Sullivan, D. Power, R. Keeney, J. "Semantic Interoperability for an Autonomic Knowledge Delivery Service", in Proc of the 2nd IFIP TC6 International Workshop on Autonomic Communication (WAC 2005), Oct. 3-5, 2005, Vouliagmeni, Athens, Greece

[17] Li, H., Jiang, G., "Semantic Message Oriented Middleware for Publish/Subscribe Networks", Proc of SPIE, Volume 5403, pp 124-133, 2004

[18] López de Vergara, J.E., Villagrá, V.A., Berrocal, J, "Semantic Management: advantages of using an ontology-based management information meta-model", Proceedings of the HP Openview University Association Ninth Plenary Workshop (HP-OVUA'2002), distributed videoconference, 11-13 June 2002

[19] Loser, A., Naumann, F., Siberski, W., Nejdl, W., Thaden, U., "Semantic overlay clusters within super-peer networks", in proc Int'l Workshop on Databases, Information Systems and Peer-to-Peer Computing in Conjunction with the VLDB 2003

[20] Meier, R., Cahill, V., "Taxonomy of Distributed Event-Based Programming Systems", The Computer Journal, vol 48, no 5, pp 602-626, 2005

[21] Muhl, G., Fiege, L., Gartner, F., Buchman, A., "Evaluating Advanced Routing Alogorithms for Content-Based Publish/Subscribe Systems", in proc of the 10th IEEE Int'l Symp. On Modelling, Analysis and Simulation of Computer Telecommunications Systems (MASCOT'02), IEEE Computer Society 2002

[22] Mulvenna, M., Zambonelli, F., "Knowledge Networks: the nervous system o fan autonomic communication infrastructure", in proc of 2nd IFIP WG6.3, WG6.6 Workshop on Autonomic Communications, Athens Greece, Oct 2005

[23] Muthusamy, V., Jacobsen, H.A., "Small–scale peer-to-peer publish/subscribe" in proc 2nd Int'l Workshop on Peer-to-Peer Knowledge Management, San Diego, USA, July 2005

[24] Rowstron A., Druschel, P., "Pastry: Scalable, Distributed Object Location and Routing for Large-Scale Peer-to-Peer Systems", In proc of IFIP/ACM International Conference on Distributed Systems Platforms (Middleware), pp 329–350, Heidelberg, Germany, November 2001.

[25] MINDSWAP. Pellet: An OWL Reasoner. http://www.mindswap.org/pellet

[26] "Pellet Performance", http://www.mindswap.org/2003/pellet/performance.shtml

[27] Petrovic, M., Burceaa, I., Jacobsen, H.A. "S-ToPSS – a semantic publish/subscribe system" in proc VLDB, Berlin, Germany, September 2003

[28] Pietzuch, P., Bacon, J., "Peer-to-Peer Overlay Broker Networks in an Event-Based Middleware". 2nd International Workshop on Distributed Event-Based Systems (DEBS'03). In conjunction with the ACM SIGMOD/PODS Conference, San Diego, California, June 2003

[29] Ranganathan, A., Campell, R.., "Autonomic pervasive computing based on planning" in proc of the 1st Int'l Conference on Autonomic Computing, New York, 2004, pp 80-87

[30] Segall, B., Arnold, D., Boot, J., Henderson, M., Phelps, T., "Content-Based Routing in Elvin4", In Proceedings AUUG2K, Canberra 2000.

[31] Service Ontology, Part of "OWL-S: Semantic Markup for Web Services", The DAML Service Coalition, http://www.daml.org/services/, October 2002, http://www.daml.org/services/owl-s/1.1/Service.owl

[32] Covering relationships in Siena, http://serl.cs.colorado.edu/~carzanig/siena/forwarding/simp/namespacesiena.html#a1

[33] Stevenson, G., Nixon, P., Dobson, S, "Towards reliable wide-area infrastructure for context-based self-management of communications", in proc of 2nd IFIP WG6.3, WG6.6 Workshop on Autonomic Communications, Athens Greece, Oct 2005

[34] Strom, R., Banavar, G. , Chandra, T., Kaplan, M., Miller, K., Mukherjee, B., Sturman, D., Ward M., "Gryphon: An Information Flow Based Approach to Message Brokering", In International Symposium on Software Reliability Engineering 1998

[35] Tempich, C., Staab, S., Wranik, A., "REMINDIN': semantic query routing in peer-to-peer networks based on social metaphors" International World Wide Web Conference (WWW), New York, USA, 2004.

[36] Terpstra, W.W., Behnel, S., Fiege, L., Zeidler, A., Buchmann, A.P., "A peer-to-peer approach to content-based publish/subscribe", in proc of 2nd Int'l workshop on Distributed event-based systems, pp 1-8, ACM Press 2003

[37] WBEM Services, http://wbemservices.sourceforge.net

[38] W3C: The Wine Ontology http://www.w3.org/TR/owl-guide/wine.rdf