A Context Information Service using Ontology-Based Queries

Ruaidhri Power, Dave Lewis, Declan O'Sullivan, Owen Conlan, Vincent Wade
Knowledge and Data Engineering Group
Department of Computer Science, Trinity College Dublin
Dublin 2, Ireland
+353 1 608 2158
{Ruaidhri.Power, Dave.Lewis, Declan.OSullivan, Owen.Conlan,
Vincent.Wade}@cs.tcd.ie

ABSTRACT
Ubiquitous computing environments have the potential to provide rich sources of information about a user and their surroundings. However, the nature of context information means that it must be gathered in an ad-hoc and distributed manner with many devices and sensors storing potentially relevant data. In an ad-hoc ubiquitous computing environment, retrieval of context information cannot rely on a fixed meta-data schema. This works shows how an ontology driven context service architecture may perform distributed open schema queries over heterogeneous context sources in a potentially decentralised manner.

Keywords
Ontologies, context information service

INTRODUCTION
The vision of ubiquitous computing is that computers will be integrated seamlessly into our daily lives. We already make much use of computers, however we can gain the most value from them when they are no longer things we interact with explicitly, rather are blended into the background and assist us when needed [1]. In order to do this, ubiquitous computing environments must be able to collect a wide range of information and use this information to work with the user in order to achieve the user's goals. This information is termed context information, and its collection and management is termed context management.

Our context management architecture uses an ontology-driven approach to bridge the heterogeneity of context information sources in ubiquitous computing systems. Ontologies are a technique for formally representing domain knowledge in an application independent way. Ontologies feature heavily in the Semantic Web initiative [7], which aims to provide ways of defining information so that it can be understood and processed by computers more easily. Examples of ontology languages are W3C's OWL , the Web ontology language and DARPA's DAML+OIL. The Semantic Web initiative has encouraged research into how ontology-based queries can be resolved in a distributed peer-to-peer manner between agents holding information with heterogeneous RDF-based semantics that are distributed over the web [9, 10, 11], though to date these have not been applied to context management. Our approach is driven by the dynamic ad hoc nature of ubiquitous computing environments and the resulting heterogeneity and lack of a priori schema knowledge for the context information that may be available to a context-aware application at any one time.

This paper describes a context information service that serves ontology-based context queries. It discusses the basic architecture, which is currently being prototyped, and the various issues of query analysis, decomposition and routing that effect the feasibility and scaleabilty of the proposed service.

CONTEXT IN UBIQUITOUS COMPUTING ENVIRONMENTS
One of the realities that must be faced in context management is that there will not be a globally standard model for representing context information. Many current approaches [8] to context management advocate predefined models for context information, which applications interact with using middleware platforms for querying and manipulation. However, context data will come in many different forms, from many different sources. Any attempt to formally structure all potential context information would be difficult at best in a controlled situation, within one organization for example, but almost impossible in roaming scenarios where applications encounter an array of different context information sources implemented in different technologies from different vendors.

Context information for ubiquitous computing environments has particular characteristics, which provide challenges in undertaking context management. Firstly, the information that can compose the context of these

environments is very broad, and can come from a variety of heterogeneous sources. Considering a ubiquitous, elearning scenario, a user's name, age, address, native language, current location and learning style could compose part of his context. Similarly, the people sharing a room with him or working in his office could be considered to be part of this context information, as could the current temperature and lighting conditions. Any system for context management must therefore be able to cope with information from a large variety of heterogeneous sources that will provide this information. Because almost any information could be considered context information from the point of view of some entity in a ubiquitous computing environment, there is very little information that we can discard as being irrelevant. Perhaps the most important characteristic of context information is that we cannot be entirely certain what information will be relevant in advance of constructing a system to manage this information. A useful solution to the problem of context management will therefore have a low impact on existing infrastructure, and cope well with heterogeneity. Such a system should also cope well with new forms of context information.

The second challenging characteristic of context information in ubiquitous computing environments arises from the fact that the environment will consist of a highly dynamic collection of users and computing devices. These devices must seamlessly integrate with whatever computing environment they are presented with, so that their users can make most efficient use of them. In this environment a roaming user or device is the norm, rather than the exception. These environments frequently make use of temporary, ad hoc connections between devices to accomplish tasks. Therefore, context information systems must be able to dynamically discover and connect to information sources in order to extract data and manipulate it into relevant context knowledge. This frequently changing environment can lead to uncertainty: where gathered information can quickly become stale; services and devices can also suddenly become available or unavailable due to changes in connectivity.

Finally, context information should be expressed in terms useful to the individual user and their application. This fits with the likelihood that the sheer volume of data that will compose context will make a global view of all context information impossible, so that mechanisms are required for interoperability between heterogeneous context sources. This is particularly the case for roaming applications, where context information must be supplied and received for a roaming user or application to avail of foreign services that are encountered. Privacy and security concerns will also prompt people to manage their own context information. A characteristic of a good solution will be that this information will be merged into each user's own view of the world, and redefined in terms that the user can understand.

CONTEXT INFORMATION SERVICE ARCHITECTURE

One of the driving forces behind the design for a context system proposed in this paper is to minimize the effort required to make a piece of software context aware. These software components will come in many forms, from the e-mail clients and office tools that are prevalent today, to tiny embedded operating systems with minimal processing power, to massive mainframe or cluster computers running large databases. For the purpose of this paper, any of these software components are referred to as applications. While this term may bring to mind today's software which is not context aware, throughout this paper it refers to any software component which wishes to make use of context information. These applications need to have easy access to more information about the environment in which they are operating, rather than being limited to the explicit input or information from hardwired data sources provided to current applications.

In our architecture, a ‘context service’ is the service provided to applications to make context information available to them. One role of a context service is to take queries from a context-aware client and to resolve those queries by acting as a mediator between the client and other information sources that the service has access to. As well as acting as consumers of context information (by executing queries), applications can also act as producers of context information by providing their context service with a description of the information they have available. If an application produces context information, a context service can advertise that information available to it to other context services.

An application can be designed as context-aware by defining an ontology that describes the domain of context information that the application is interested in querying, and also that it wants to make available to other applications. This ontology may be written by the application developer from scratch, or it may be possible to reuse an existing ontology such as CoBrA-ONT [20]. This ontology is registered with the context service as belonging to the application, and is stored in an ontology repository accessible to the context information service. The application developer has the option of providing mappings between concepts in the application’s ontology and equivalent concepts in other ontologies used within the system. This is however not a requirement, as this step may be done at a later stage. If mappings are provided, they will be stored in an ontology mapping repository, also accessible to the context information service.

The internal architecture of a network implementing the context service is shown in Figure 1. Starting from the bottom of the diagram, applications present queries to the context service. Each of these query messages contains the

content of the query Q and a reference to the ontology O to which the query refers. This query is taken by the context service which examines the query and ontology used. The context service is implemented by a set of peer context service nodes (CSN). Combining these with mappings from the ontology mapping repository, the CSN receiving the query from an application can then compose a new query that can be routed to other CSNs, which will attempt to return a corresponding result.

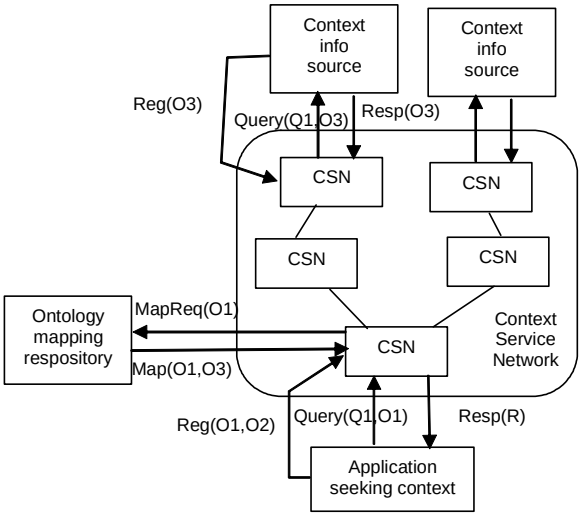

Figure 1: Context Service Network Architecture

Any results that are returned are translated back from the ontology of the remote CSN into the application's ontology before they are returned as a query response, R.

CONTEXT QUERYING

The context service offers a query interface, which will accept a query in a supported query language, such as SQL, XQuery , RDQL . The terms involved in the query must be a subset of those in the ontologies understood by the context service, for the service to be able to understand the query. Because each context-aware application has provided its own ontology to the context service, the local CSN is guaranteed at a minimum to be able to understand queries posed by its applications.

Associated with each CSN there is a repository of ontologies that describe the domains of knowledge that this node currently holds. These ontologies are provided to the CSN by the applications for which it acts as a source of context information, and can also be discovered through communication with other CSNs. The CSN must take each of these queries and communicate with other CSNs to resolve them. It does this by sending the query to some number of CSNs, after translating the query into terms that they will understand.

We consider an example scenario of a college's lecture theatre equipped as a ubiquitous computing environment. Software in the theatre has been developed for use in any meeting scenario and provides in its ontology the terms 'meeting' and 'document'. When the software was installed, the college provided mappings to the context service to say that these terms were equivalent to the terms 'lecture' and 'notes' (stored on a college context server) for their purposes. The application's query to its local CSN is phrased in terms of 'meeting' and 'document', allowing the software to remain unchanged when deployed in this scenario, but still communicate with a college CSN about which nothing was known when the application was designed. The lecture theatre CSN transforms a query for the lecture document, say, and passes it on to the college CSN which then returns the notes in question.

These CSNs will be potentially quite large in number and can be distributed around the network in whatever way seems most appropriate, most likely based on how devices are managed. For instance, if all applications on a PDA are provided by the same vendor, they may be happy to use a single CSN on the PDA. They are not limited to this configuration though, as for example some devices may not have the processing power to host a CSN and will rely on a remote service that will provide context to them over a network connection.

Once the client application's query has been received and understood, the first CSN must decide to where these queries should be routed. In the scenario presented above, the CSN knew of one location where it could find information to answer its query. However, there will be many situations in which the set of CSN which will be able to answer the query is not known a priori.

The CSN can decide where to send the queries based on a routing algorithm. In the simplest case, this is just to route the query to one other node that will understand the query (has equivalent terms in its local ontologies) and is therefore able to return a meaningful reply. This query can be understood by any other CSN that have equivalent terms in their own local ontologies. Because the lecture theatre's CSN knows that the college context service has these equivalent terms, it translates the original query into the new terms and forwards it on. When it receives a reply to this query, it can then translate it back into the original terminology and pass it on to the application.

Alternatively, this query could be multicast or sent over a content-based network to a targeted set of CSNs that will be able to answer the query. These approaches have different advantages: queries that are only sent to a small number of nodes are efficient in terms of bandwidth and processing costs in routing of queries, but queries sent to a large number of nodes may have the greatest chance of being answered satisfactorily. The trade-off is highly dependent on the accuracy of the current routing tables possessed by a CSN. A problem arises, for instance, when a large number of CSNs can understand a query. Obviously they should not all be sent the query (as a

broadcast), as this would be wasteful of bandwidth. The CSNs must decide how to route these queries to achieve the optimal effect. Queries posed by applications should be routed to those CSNs that can answer them, but should not be spread any further than is necessary.

Research into content-based networking (CBN) [5][6] has produced an enhancement to traditional publish/subscribe mechanisms that may prove useful in the design of a context system. In a content-based network, rather than messages being given an explicit destination address, they are routed to a set of hosts based on predicates applied to their content. Hosts are assigned `addresses' in the content-based network based on a function which describes the type of messages they are interested in. Content-based routers can then decide which of their neighbouring routers to send the message to, based on a routing table composed from these predicates.

Chan et. al. describe in [19] a method of content-based routing that is specifically tailored to XML documents. Consideration is again given to the space and efficiency characteristics of such a solution. Here, XPath is used as the XML pattern specification language. XML seems an appropriate choice as an information interchange markup, and XPath an effective way of querying those XML documents and specifying patterns of interest.

In [12] we discuss a CBN-based approach to an ontology-based query service that would be well suited to monitoring changes in context information over time. The proposal uses *persistent* ontology-based queries for defining the information being sought and shared, so that the range of supported application domains automatically reflects the ever-expanding range of domain ontologies that will be published for use in the Semantic Web. Internally the service uses Content-Based Networking techniques to efficiently deliver the meta-data of published information to interested parties, as well as to support the autonomic management and knowledge management needs of the architecture itself. Role based access control can be applied to advertisements and subscriptions in CBNs [17], though we favour a more flexible community-based approach [18] to defining access control policies to match the less structured and more dynamic organisational environments that will characterize ubiquitous computing.

Research into CBNs has shown promising scalability in the number of end systems involved (global CBN are already operated by companies such a Tibco) and in the churn of currently connected end systems [25] – an important feature in ubiquitous computing environments with mobile context querying clients.

DISCUSSION

We see a number of advantages to the approach outlined in this paper. Firstly, applications can be designed independently of the environment in which they are finally run, because of the encapsulation of their domain knowledge in an ontology. This is the minimum amount of work that an application programmer will have to do to enable his application to be context-aware. In addition, integration into a context system is done through mappings between equivalent terms. These mappings can be set up and also changed at a later stage, all independently of the original application programmer. These mappings will either have to be provided by the application programmer, or (we believe more likely) they will be provided by the people integrating the software in the environment. A further benefit is that mappings between equivalent terms can be stored and reused in future. In the simple example above, the equivalence between the lecture theatre's term ‘meeting’ and the college's term ‘lecture’ can be stored and deployed automatically in future, saving effort. Finally, a service that uses content-based networking for the routing of queries is potentially much more flexible and efficient than ones that use centralized architectures, or unfocussed multicasting of queries

This approach however, leaves many open question related to the limits of semantic based reasoning in such a content based network. Currently standardized ontologies are based on description logic, which is soon to be complemented with the Semantic Web Rule Language (SWRL) [14]. Though the latter will obviously aid in the representation of queries as ontologies it is far from clear that all form of context queries can be addressed with these ontological logics. Though a combination of OWL and SWRL may go some way to being able to reason about the many classes of queries (and then only with the support of ontologies for temporal logic), other logics suitable for feeding optimization algorithms may be also be needed. This in turn will require an extensible, modular structure for reasoners embedded in network nodes, similar to existing semantic application toolkits [15], but which can be scalably managed. For example within the semantic query CBN we envisage nodes dynamically subscribing with queries for logic problems which they encounter in resolving user queries in order to locate suitable downloadable code to conduct the required reasoning. Equally, ontologies capturing mappings between concepts in separate domain ontologies that appear in user queries can also be sought and obtained by semantic CBN nodes from ontology repositories using the CBN service itself.

A CBN-based context information service raises several further issues that require investigation in order to assess usability and scalability of this architecture for deployment on the Internet. We must perform a more detailed assessment of the performance possible with existing ontology-based matching algorithms, though in the long term we expect that optimized software and hardware support for OWL will emerge driven by its potential

popularity, as has already happened for XML processing. One possible optimization that will reduce the reasoning load on semantic routing nodes will be to decompose context queries based on known routes prior to submitting them as subscription queries to the service [16].

RELATED WORK

As part of its work on context management, the Aura Project[21] at Carnegie Mellon University provides contextual services built on top of a contextual service framework. These services provide applications with properties of both physical entities and available resources. A fixed schema which is used to represent contextual information in this system. This schema describes classes of entities which are used within the system: devices, networks, people and areas. Also represented are relationships between entities. Aura provides a single interface to all services to provide a consistent mechanism for client access and eliminate redundant code. However, any context-aware applications that wish to interoperate with Aura must be written so that their queries conform to the Aura information model (schema). This raises the barrier of entry for the applications and requires universal agreement on the Aura schema for universal adoption.

The Semantic Space [22] infrastructure uses the Semantic Web standards RDF and OWL to define context ontologies. These technologies are used for representing knowledge bases, for querying and for inference. Semantic Space uses ontologies to let web agents exchange and interpret information based on a common vocabulary. The context model is composed of an upper-level context ontology, which provides basic concepts designed to be common to all context-aware applications. This ontology can then be extended with further ontologies that cater for each application's specific requirements. This approach however requires agreement on the upper-level context ontology by all applications wishing to make use of Semantic Space. It is unclear how applications making use of extended context ontologies can interoperate with each other. This is because they introduce terms which are not understood by all context-aware applications, as they are not part of the upper-level context ontology. In contrast, the work presented in this paper does not rely on a fixed upper-level schema which must be agreed on by all applications, but rather faces the reality that such a schema will not exist. Hence, mechanisms are explored for enabling interoperability through mappings between independently-developed schemas.

The Nexus[23] project manages context using a "world model" service based on the "Augmented World Model" (AWM). This model describes real and virtual objects relevant to location-based applications. Nexus aims to support all kinds of context-aware applications by providing a shared global context schema. As part of this project, the researchers discovered that they had to adapt their context schema as time progressed, incorporating additional requirements from other projects which had not previously been foreseen. These changes to the underlying schema could invalidate existing applications which relied on the old model, requiring a considerable amount of effort to bring them up to date. In contrast, our work can cope with changes in information models due to the use of mappings between ontologies, which can be updated as models change, without requiring the application to be re-engineered.

The PACE[24] project provides a set of generic modelling constructs, allowing context modelling using well-defined programming abstractions. These abstractions allow mechanisms such as event notification and branching (context-dependent choice from a number of alternatives). However, this work does not tackle problems related to multiple independently-developed models of context information which need to work together in a context aware environment.

FURTHER WORK

Experimental work in progress focuses on implementing the architecture outlined in this paper, and applying it to a student project meeting scenario. Our initial implementation uses OWL as the ontology language, XQuery for the query language and Content-based networking algorithms for the query routing mechanism. However, this work will characterize desirable features in each technology, rather than prescribe particular technology selections, in order that they can be more easily be identified as new technologies arise. An initial experiment to verify the approach presented in this paper is underway, and is due for completion in September 2004.

Another strand of work is addressing the fact that responses to context queries may produce a definite answer, or they may return that the answer to the question is unknown. This will require application programmers to be aware that context-aware applications may operate in an environment where not enough information is available to answer their queries, and adjust their behaviour accordingly. A further area of study is the decomposition of complex queries into parts answered by different information sources. The field of distributed databases provides insight into how queries across distributed information sources as a plan of execution [2], possibly structured by some cost factor [3] or based on partially pre-computed plans [4]. These techniques need to be adapted to the open schema scenario addressed by the context setting.

In general, further experimentation will be required to evaluate the scalability and performance of such knowledge based networking against variations in numbers of information sources, sinks, advertisements, subscriptions and client join/leaves. More challenging is the need to assess scalability against growth in the number

and scope of ontology domains, ontology encoded logics and ontology mappings. In addition, the potential sensitivity of context information means access control and its use in enforcing privacy policies must be comprehensively addressed.

ACKNOWLEDGMENTS

This work was partially funded by the Irish Higher Education Authority under the M-Zones programme.

REFERENCES

1. Weiser M., "Some computer science problems in ubiquitous computing," Communications of the ACM, July 1993.
2. A. Ip, W. Rahayu, and S. Singh, "Query optimization in a non-uniform bandwidth distributed database system," in Proc. of the 4th International Conference on High-Performance Computing in the Asia-Pacific Region, vol. 2, (Beijing, China), pp. 818--823, May 2000.
3. L. Mackert and G. Lohman, "R* optimizer validation and performance evaluation for distributed queries," in Proceedings of the Conference on Very Large Databases (VLDB), Kyoto, Japan, 1988.
4. Z. Nie and S. Kambhampati, "Joint optimization of cost and coverage of query plans in data integration," in Proc of the 10th international conference on Information and knowledge management, (Atlanta, Georgia, USA), pp. 223-230, ACM Press, 2001.
5. A. Carzaniga and A. L. Wolf, "Content-based networking: A new communication infrastructure." NSF Workshop on an Infrastructure for Mobile and Wireless Systems, 2001
6. A. Carzaniga and A. L. Wolf, "Forwarding in a content-based network," in Proc of ACM SIGCOMM 2003, (Karlsruhe, Germany), pp. 163--174, Aug 2003.
7. Berners-Lee T., Hendler, J. Lassila, O. "The semantic web." Scientific American, May 2001.
8. Mitchell K., A Survey of Context-Awarenes Available: http://www.comp.lancs.ac.uk/~km/papers/ContextAwarenessSurvey.pdf [2002, Nov. 27]
9. Cai, M., Frank, M., "RDFPeers: A Scaleable Distributed RDF Repository based on a Structured Peer-to-Peer Network, in Proc. of World Wide Web Conference 2004, 17-22 May 2004, New York, NY, USA
10. Stuckenschmidt, H., Vdovjak, R., Houben, G.J., Broekstra, J., "Index Structures and Algorithms for Querying Distributed RDF Repositories", in Proc. of WWW Conf. 2004, 17-22 May 2004, New York, NY, USA
11. Tempich, C., Staab, S., Wranik, A., "REMINDIN': Semantic Query Routing in Peer-to-Peer Networks Based on Social Metaphors", in Proc. of WWW Conf 2004, 17-22 May 2004, New York, NY, USA
12. Lewis, D., Feeney, K., Tiropanis, T., Courtenage, S., "An Active, Ontology-driven Network Service for Internet Collaboration", in Proc of Semantic Web for Web Communities workshop, 2004, Valencia, Spain
13. Belokosztolszki, A., Eyers, D.M., Pietzuch, P.R., Bacon, J., Moody, K,. "Role-based access control for publish/subscribe middleware architectures". in International Workshop on Distributed Event-Based Systems (DEBS03), ACM SIGMOD, San Diego, CA, USA, 2003. ACM.
14. Horricks, I., Patel-Schneider, P., Boley, H., Tabet, S., Grosof, B., Dean, M. (2003), "SWRL: A Semantic Web Rule Language Combining OWL and RuleML", version 0.5, 19th November 2003,
15. Oberle, D., Staab, S., Volz, R., "An Application Server for the Semantic Web", in Proc of WWW'04, May 17-22, 2004, New York, USA, pp. 220-221.
16. Courtenage, S. "Specifying and Detecting Composite Events", In 1st International Workshop on Discrete Event-Based Systems, Vienna, 2002
17. Belokosztolszki, A., Eyers, D.M., Pietzuch, P.R., Bacon, J., Moody, K,. "Role-based access control for publish/subscribe middleware architectures". In International Workshop on Distributed Event-Based Systems (DEBS03), ACM SIGMOD, San Diego, CA, USA, 2003. ACM.
18. Feeney, K., Lewis, D., Wade, V. "Policy-based Management for Internet Communities", in proc of 5th IEEE International Workshop on Policies and Distributed Systems and Networks, 2004, pp 23-34
19. C.-Y. Chan, W. Fan, P. Felber, M. Garofalakis, and R. Rastogi, "Tree Pattern Aggregation for Scalable XML Data Dissemination." Bell Labs Technical Memorandum, February 2002
20. H. Chen, T. Finin, and A. Joshi, "An ontology for context-aware pervasive computing environments," Special Issue on Ontologies for Distributed Systems, Knowledge Engineering Review, 2003
21. G. Judd, P. Steenkiste "Providing Contextual Information to Ubiquitous Computing Applications", CMU Technical Report CMU-CS-02-154, July 2002
22. X. Wang, J. S. Dong, C. Chin, S. Hettiarachchi, D. Zhang "Semantic Space: An Infrastructure for Smart Spaces", IEEE Pervasive Computing, July-September 2004 (Vol. 3, No. 3), pp. 32-39
23. F. Dürr, N. Hönle, D. Nicklas, C. Becker, K. Rothermel, "Nexus - A Platform for Context-aware Applications", to appear in Proc GI-Fachgespräch "Ortsbezogene Dienste", Hagen, Germany

24. K. Henricksen, J. Indulska, "A Software Engineering Framework for Context-Aware Pervasive Computing", in Proc. of the 2nd IEEE International Conf. on Pervasive, Computing and Communications; Orlando, Florida, March 2004, IEEE Com. Soc., pp. 77-86

25. Chand, R., Felber, P.A., “A Scaleable Protocol for Content-Based Routing in Overlay Networks, In IEEE International Symposium on Network Computing and Applications (NCA'03), Cambridge, MA, April 2003.