Utilizing Context in Adaptive Information Services for Pervasive Computing Environments

Ruaidhri Power, Declan O'Sullivan, Owen Conlan, Dave Lewis, Vincent Wade

Knowledge and Data Engineering Group
Trinity College, Dublin, Ireland
{Ruaidhri.Power, Declan.OSullivan, Owen.Conlan, Dave.Lewis, Vincent.Wade}@cs.tcd.ie
http://kdeg.cs.tcd.ie

Abstract. Ubiquitous and pervasive computing environments have the potential to provide rich sources of information about a user and their surroundings. Such context information may form a basis upon which Adaptive Information Services may be adapted. However, the nature of context information means that it is usually gathered in an ad-hoc and distributed manner with many devices and sensors storing potentially relevant data. The reconciliation and reasoning across this information presents a research challenge, but has the possibility of yielding valuable insights about users. In an ad-hoc pervasive computing environment, determining context information cannot rely on a fixed meta-data schema. This work shows how an ontology driven context service architecture will perform distributed open schema queries over heterogeneous context sources in order to provide information service adaptation.

1 Introduction

The vision of pervasive 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, but rather are blended into the background and assist us when needed [1]. In order to do this, pervasive 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.

Traditionally, Adaptive Information Services, such as Adaptive Hypermedia Systems [2] and personalized eLearning Services [3,4,5], have used a centrally stored and managed user/learner model as the basis for adaptation. These models tend to store pre-defined types of information and are built using explicit techniques, such as direct user querying, and implicit techniques, such as monitoring user interactions [6]. Context-based systems, however, provide a different approach to acquiring and storing modeling information. This approach, and more precisely the information model used, is more ad-hoc and potentially richer than the modeling techniques applied in most Adaptive Information Systems. The precise syntax and semantic descriptions of context information may not be known beforehand.

Our context management architecture uses an ontology-driven approach to bridge the heterogeneity of context information sources in pervasive 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[1], the Web ontology language and DARPA's DAML[2].

This paper describes the potential of context information when applied to Adaptive Information Services. Section 2 introduces context and the challenges that are involved in context management. Section 3 explores the potential for context to support Adaptive Systems and section 4 provides an overview of the context service architecture that we propose to support adaptive information services. Section 5 illustrates the architecture through exploration of an example scenario. Finally Section 6 presents conclusions and highlights future work.

2 Context in Pervasive 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 an inter-organizational scenario.

Context information for pervasive 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. 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 pervasive 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 pervasive 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.

1 http://www.w3.org/TR/owl-guide/
2 http://www.daml.org/

Finally, these environments should present context information in terms that the user can relate to, in other words the information should be user centric. Context information will almost certainly be managed by the entity to whom that context information relates (for example a business or an ordinary individual), rather than being managed globally. This is due to a few factors: the sheer volume of data that will compose context will make a global view of all context information impossible. Privacy and security concerns will also prompt people to manage their own context information. Perhaps most importantly, the set of information that will compose this context is so dynamic that it will never be standardized, so mechanisms will have to be developed to promote interoperability between context systems. These mechanisms will translate context into a form where it can be understood by each organization's context system. This is particularly the case for roaming applications, where context information must be supplied and received for a roaming user or device to avail of services within another environment. 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.

3 The potential for Context in Adaptive Systems

This section outlines three important mechanisms for delivering service adaptivity for pervasive computing environments, namely service composition, policy-based management and adaptive hypermedia, and outlines how these might make use of context.

Service Composition
Pervasive computing environments will exhibit a large amount of heterogeneity in the components from which they are constructed. Any adaptive system supporting pervasive computing will therefore face major interoperability and integration challenges in combining the adaptivity of user services with adaptive resource management. There is increasing interest in automating the service composition process, so that the service offered to users appears to be adaptive[9]. In a pervasive computing environment, the automatic composition of service needs to be driven by both the task required by the user and the context in which the task is to be performed [10]. Ontological representations of context [11] can be used in automating the selection of existing services in a service composition using an ontological representation of the service. For instance, when composing wireless multimedia presentation services for wireless PDAs in particular location the video compression capabilities of candidate constituent video streaming services need to be compared to context information on the decoding capabilities of the PDA and the current capacity of the wireless link. Equally, context information may be used in accepting or rejecting plans for sub compositions during the planning process, by comparing it to the planned composite service properties.

Policy-based Management
Another adaptive technique, which is seeing increased deployment in managing the adaptive behavior of network and services, is policy-based management [12]. It uses expressive rule languages to determine behavioral rules for how a system should respond to predetermined events and system conditions. In pervasive computing environments, anyone entering the space may possess or use resources that may be shared. Policies provide a way of managing such ad hoc collections of resources, but need to employ flexible means of binding resources and policy subjects to rules at runtime [13]. As this requires matching terms in policy rules to ad hoc information, there is increasing interest in using ontologies for policy definitions [14], which in turn allows us to enforce of policy rules a run-time with ontology-based context information used for resolving events and conditions in a policy to equivalent event and condition in the pervasive computing environment.

Adaptive Hypermedia
Presentation-centric adaptive systems use explicit user models to tailor information to different users. Data is collected for the user model from various sources, e.g. contact lists, schedules, terminal capabilities, application usage histories, security, cost, navigational and presentational preferences. The user model is the basis of the adaptation effects, and thus in a pervasive computing environment it needs to be resolved against current contextual information. One area of strong research into personalized adaptive systems is Adaptive Hypermedia systems, which are typically applied to areas of learning, such as museum guides or eLearning. These offer an alternative to the traditional "one-size-fits-all" approach by employing user models that allow personalization in hypermedia systems. The benefits of such personalization include relevancy, reduced time to learn and improved retention and recall. We have already developed a sophisticated generic adaptive engine that has been applied successfully to personalized eLearning hypermedia [3], which we are now extending to support dynamic context information acquisition to populate the user model.

4 Proposed Architecture for Context Services

One of the driving forces behind the design for a context system proposed in this paper is to minimise 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 from scratch, or it may be possible to reuse an existing ontology such as CoBrA-ONT [15]. This ontology is registered with the context service as belonging to the application, and is stored in the ontology repository.

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 the ontology mapping repository.

The internal architecture of a context-aware device is shown in Figure 1. Each box within the device represents an autonomous piece of software, with their interactions described by arrows. 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**, a reference to the query language used **L**, and a reference to the ontology **O** that the query refers to. This query is taken by the context service which examines the query and ontology used. Combining these with mappings from the ontology mapping repository, the context service can then compose a new query that can be routed to other context services, which will attempt to return a corresponding result.

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

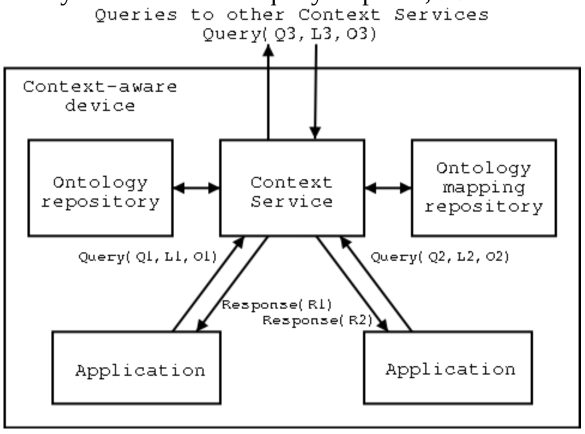

Figure 1: Context-Aware Device Architecture

The internal structure of a context service is shown in Figure 2. The first major functional section of a context service are its query interface/query analysis modules. The query analysis module also handles query decomposition. The decomposed queries are then passed to the query routing module.

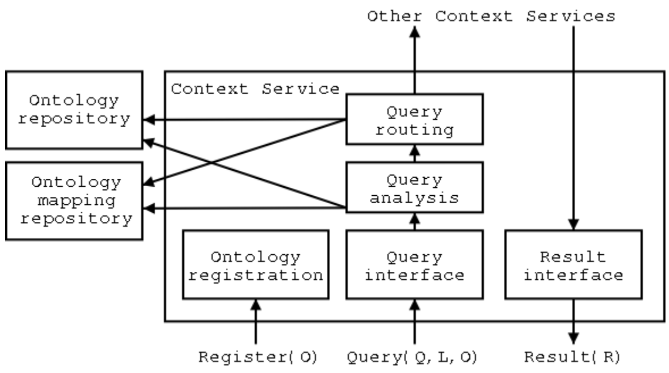

Figure 2: Context Service Internal Design

Queries are routed to the query interface of the appropriate context service by the query routing module. Results are returned to and combined appropriately by the Result Interface.

5 Using Context in Adaptive Information Services

In this section we present a scenario in which information delivery to a user is adapted , based on available context information. The scenario considered is that of college students who take part in an arranged lecture and subsequently wish to review the lecture material at home. In this scenario, the many students taking part in the lecture all have separately constructed user models describing their competencies, preferred learning styles, and so on.

We consider the case of a particular student who is attending the lecture. Her user model has been exposed through a context service running on her PDA, and this model has been constructed using terms from a well-known user modeling ontology. Because this ontology is well-known, mappings exist between it and the ontology being used by the college's information services, which may be highly tailored to the college's specific needs.

When an information service needs access to these models it poses a query based on terms within its ontology to its local context service that then distributes the queries to the relevant remote context services in terms of their ontologies. The responses to these queries are then translated into the terms that the information service that posed the query has in its ontology. This mechanism allows the information services in the lecture theatre to present adapted content to the student based on preferences in her user model. For example, we could imagine an information service that provides lecture notes that would adapt the notes delivered based on the learning style of the user, or provide them in a different natural language if the user's context indicated that their native language was not English. A key feature of this approach is that any information that the students make available as part of their context does not need to be captured through either explicit or implicit techniques for user modeling, as it is automatically retrieved from context services as needed.

In addition to a user model, relevant information such as the characteristics of the student's display device (a PDA) is available as part of her context. This allows the lecture theatre software to deliver a set of notes that is text-based, rather than one that uses large diagrams, for display on the PDA. Once our student has returned home, she wishes to review the material covered in the lecture on her laptop. After she transfers the notes she downloaded during the lecture to her laptop, her laptop software can then use the stored reference to the context of the lecture to file the notes appropriately, and also contact the college information service to retrieve a set of notes that are better suited to its larger display. Because of concepts mastered during the course of the lecture, software on the laptop can now adapt the notes presented based on the updated user model as part of the student's context. This is true despite the internal model in the laptop's software being independently authored, as the context service will again, based on ontology mappings, translate responses to queries into terms that the laptop software will understand.

While this scenario relies on some aspects of pervasive computing such as location awareness, it demonstrates how information services in general can be given access to context information - user models and any other information - that can be used to perform adaptation.

6 Conclusion and Future Work

This paper has described context information and highlighted its use in Adaptive Information Services through a detailed scenario. It has also presented an ontology-based architecture for a distributed context management system. A state of the art in ontology based integration (e.g. KRAFT), content based routing, and distributed querying using P2P (e.g. Edutella) has been completed. An initial implementation of the context management system to verify the design presented in this paper is already underway.

Context information has the potential to *fill in the gap* left with traditional modeling techniques. However, the challenge arises in successfully leveraging this information as part of Adaptive Information Services. Context information will not necessarily conform to a pre-defined schema – the Adaptive Information Service may have to discover, and understand, the schema. In addition, the semantics of the information described may not be fully understood by the Adaptive Information Service – again this may need to be discovered, possibly with the aid of concept ontologies. For these reasons it is our belief that the integration of an adaptive system with a context management system which is ontology based potentially provides a powerful solution. Thus it is planned to integrate the ontology driven context management system outlined in this paper with such an adaptive system (APeLS [3]) to explore the potential of using context as a rich source of adaptation of information services.

References

1. Weiser M., “Some computer science problems in ubiquitous computing,” Communications of the ACM, July 1993.
2. Brusilovsky, P. (2000) Adaptive hypermedia: From intelligent tutoring systems to Web-based education (Invited talk). In: G. Gauthier, C. Frasson and K. VanLehn (eds.) Intelligent Tutoring Systems. Lecture Notes in Computer Science, Vol. 1839, (Proceedings of 5th International Conference on Intelligent Tutoring Systems,ITS 2000, Montreal, Canada, June 19-23, 2000) Berlin: Springer Verlag, pp. 1-7.
3. Conlan, O., Wade, V., Bruen, C., Gargan, M. (2002) Multi-Model, Metadata Driven Approach to Adaptive Hypermedia Ser-vices for Personalized eLearning. In the Proceedings of Second International Conference on Adaptive Hypermedia and Adaptive Web-Based Systems, AH 2002, 100-111.
4. De Bra, P., Santic, T., and Brusilovsky, P. (2003) AHA! meets Interbook, and more... In: A. Rossett (ed.) Proceedings of World Conference on E-Learning, E-Learn 2003, Phoenix, AZ, USA, November 7-11, 2003, AACE, pp. 57-64.
5. De Bra, P. (2001) “AHA! Adaptive Hypermedia for All”, project proposal. http://www.nlnet.nl/project/aha/200106-aha-proposal.html
6. Kobsa, A. (1993). User modeling: Recent Work, Prospects and Hazards. In Adaptive User Interfaces: Principles and Practice, M. Schneider-Hufschmidt, T. Kühme, and U. Malinowski, (eds.). 1993, North-Holland: Amsterdam.
7. Berners-Lee T., Hendler, J. Lassila, O. “The semantic web.” Scientific American, May 2001.
8. Mitchell K., A Survey of Context-Awareness, Available: http://www.comp.lancs.ac.uk/~km/papers/ContextAwarenessSurvey.pdf [2002, Nov. 27]
9. Ponnekanti, S.R., Fox, A. (2002), "SWORD: A Developer Toolkit for Web Service Composition", To appear in The Eleventh World Wide Web Conference (Web Engineering Track), Honolulu, Hawaii, May 7-11, 2002
10. Masuoka, R., Labrou, Y., Parsia, B., Sirin, E., “Ontology-Enables Pervasive Computing Applications”, IEEE Intelligent Systems, Sept/Oct 2003, pp 68-72
11. Chen H. and Finin T "An Ontology for Context-Aware Pervasive Computing Environments", Workshop on Ontologies and Distributed Systems at 18th International Conference on Artificial Intelligence, August 2003, Acapulco, Mexico.
12. Kagal, L., Finin, T., Anupam J.,“A Policy Language for Pervasive Systems”, Fourth IEEE International Workshop on Policies for Distributed Systems and Networks, Lake Como, 4-6 June, 2003
13. Sloman, M., Lupu, E. “Security and Management Policy Specification”, IEEE Network, Vol. 16, no. 2, 2002, p 10-19
14. Tonti, G., Bradshaw, J.M., Jeffers, R., Montanari, R., Suri1, N., Uszok, A., “Semantic Web Languages for Policy Representation and Reasoning: A Comparison of KAoS, Rei, and Ponder" Proceedings of 2nd International Semantic Web Conference (ISWC2003), October 20-23, 2003, Sanibel Island, Florida, USA
15. Chen H, Finin T, and Joshi A, “An ontology for context-aware pervasive computing environments,” Special Issue on Ontologies for Distributed Systems, Knowledge Engineering Review, 200