Ontologies in a Pervasive Computing Environment

Anand Ranganathan, Robert E. McGrath, Roy H. Campbell, M. Dennis Mickunas

Department of Computer Science
University of Illinois, Urbana-Champaign
ranganat@uiuc.edu, mcgrath@cs.uiuc.edu, rhc@uiuc.edu, mickunas@cs.uiuc.edu

Abstract

Ontologies are entering widespread use in many areas such as knowledge and content management, electronic commerce and the Semantic Web. In this paper we show how the use of ontologies has helped us overcome some important problems in the development of pervasive computing environments. We have integrated ontologies and Semantic Web technology into our Pervasive Computing infrastructure. Our investigations have shown that the Semantic Web technology can be integrated into our CORBA-based infrastructure to augment several important services. This work suggests a number of requirements for future research in the development of ontologies, reasoners, languages and interfaces.

1 Introduction

There is growing evidence for the potential value of Semantic Web technology for Web Services and other open, distributed systems [Goble et al., 2002; Peer, 2002]. This paper presents a case study of the use of Semantic Web technology in a Pervasive (or Ubiquitous) Computing Environment, GAIA [Roman et al., 2002].

Pervasive (or Ubiquitous) Computing Environments are physical environments saturated with computing and communication, yet gracefully integrated with human users [Lyytinen et al., 2002]. These environments involve the construction of massively distributed computing systems that feature a large number of autonomous entities (or agents). These entities could be devices, applications, services, databases, users or other kinds of agents. Various types of middleware (based on CORBA, Java RMI, SOAP, etc.) have been developed that enable communication between different entities. However, existing middleware have no facilities to ease semantic interoperability between the different entities.

Ontologies have been widely used in many areas such as knowledge and content management, electronic commerce and the semantic web. In this paper we show how the use of ontologies has helped us overcome some of the challenges in constructing and managing a pervasive computing environment. Of course, these problems are not unique to pervasive computing, but are faced by any multi-agent software system. We believe that our solutions to some of these issues can be extended to any multi-agent system.

This work has considered three major issues that confront the development and deployment of Pervasive Computing Environments. These are:

- Discovery and Matchmaking
- Inter-operability between different entities
- Context-awareness

This section briefly describes these three tasks in the domain of a pervasive computing environment.

In the future, we will extend this work to augment the configuration and management of multiple Spaces, and to augment additional services, such as the Quality of Service infrastructure [Wichadakul et al., 2002].

1.1 Discovery and Matchmaking in a Pervasive Computing Environment

A Pervasive Computing Environment has one or more registries to keep a real time state of the system, i.e., the entities currently present and available; and should have a protocol for discovering the arrival and departure of mobile entities, for advertising current availability, and for notifying interested parties of changes. A registry with these protocols is termed a "Discovery Service" [McGrath, 2000]. *Matchmaking* [Trastour et al., 2001] uses the Discovery Service to discover not only what entities are available, but what sets or combinations meet certain criteria, i.e., the requirements and preferences of the parties.

In the Discovery Service, standard schemas are needed to describe many kinds of entities, including people, places, and things. Furthermore, the system has policies, constraints, and relationships which may need to be discovered as well. For a robust system, it is necessary to have a flexible mechanism for exchanging descriptive information of many kinds.

1.2 Inter-operability between different entities

New entities may enter the environment at any time; these new entities have to interact with existing entities. The interaction must be based on common, well-defined concepts, so that there is no misunderstanding between the entities. The entities must have a common understanding of the various terms and concepts used in the interaction.

For autonomous entities to interact with one another, they need to know, beforehand, what kinds of interfaces they support and what protocols or commands they understand. In a truly distributed scenario, such as a pervasive computing environment, it may not be reasonable to assume that such agreement exists.

Similar mechanisms are needed for humans to interact with different entities. Humans need to understand what various entities do, and they need to understand the relationships between such entities. It is essential for humans to form an accurate conceptual model of the environment so that they can interact with the environment easily.

1.3 Context-Awareness

Applications in pervasive and mobile environments need to be context-aware so that they can adapt themselves to rapidly changing situations. Applications in pervasive environments use different kinds of contexts (such as location of people, activities of individuals or groups, weather information, etc.).

The various types of contextual information that can be used in the environment must be well-defined so that different entities have a common understanding of context. Also, there needs to be mechanisms for humans to specify how different applications and services should behave in different contexts. These mechanisms need to be based on well-defined structures of different types of context information.

1.4 Ontologies in Pervasive Computing Environments

In order to tackle the problems described above, we apply the technologies from the emerging "Semantic Web" [Berners-Lee et al., 2001; W3C, 2003a]. While the Semantic Web was designed to enhance Web search and agents, we show that it is well suited to some of the requirements of Pervasive Computing Environments.

We have incorporated the use of ontologies in our prototype pervasive computing environment, GAIA [Roman et al., 2002]. The ontologies are written in DAML+OIL [daml.org, 2003], describing various parts of the GAIA environment. An Ontology Server manages a composite ontology that describes the entities of the system and performs operations on the ontologies. This composite ontology is built by composing different ontologies specified in DAML+OIL files. The ontologies are validated into a Knowledge Base (KB), built on the CORBA FaCT Server [Bechhofer et al., 1999; Horrocks et al., 1999].

Ontologies are used for describing various concepts in the GAIA Pervasive Computing Environment. We have developed ontologies that describe the different kinds of entities and their properties. These ontologies define different kinds of applications, services, devices, users, data sources and other entities. They also describe various relations between the different entities and establish axioms on the properties of these entities that must always be satisfied.

A second use of ontologies is to describe different types of contextual information in GAIA. The ontology defines standard descriptions for locations, activities, weather information, and other information that may be used by context-aware applications.

The ontologies that describe the pervasive environment greatly help in the smooth operation of the environment. Some of the ways in which we use ontologies in our pervasive environment are:

- Checking to see if the descriptions of different entities are consistent with the axioms defined in the ontology. This also helps ensuring that certain security and safety constraints are met by the environment
- Enabling semantic discovery of entities
- Allowing users to gain a better understanding of the environment and how different pieces relate to each other
- Allowing both humans and automated agents to perform searches on different components easily
- Allowing both humans and automated agents to interact with different entities easily (say, by sending them various commands)
- Allowing both humans and automated agents to specify rules for context-sensitive behavior of different entities easily
- Enabling new entities (which follow different ontologies) to interact with the system easily.

In the following sections, we describe how ontologies are used within our Pervasive Computing Environment, GAIA. We first introduce GAIA; we then describe how the Ontology Server fits into the GAIA framework. We then briefly describe the kinds of ontologies that have been developed and how they are used within GAIA. Finally, we evaluate our approach and suggest important areas for future research.

2. GAIA: A Pervasive Computing Environment

GAIA is an infrastructure for Smart Spaces, which are pervasive computing environments that encompass physical spaces [Roman et al., 2002]. GAIA converts physical spaces and the devices they contain into a programmable computing system. It offers services to manage and program a Space and its associated state. GAIA is similar to traditional operating systems in that it manages the tasks common to all applications built for physical spaces. Each Space is self-contained, but may interact with other Spaces. GAIA provides core services, including events, entity presence (devices, users and services), discovery and naming. By specifying well-defined interfaces to services, applications may be built in a generic way so that they are able to run in arbitrary Smart Spaces. The core services are started through a bootstrap protocol that starts the GAIA infrastructure. GAIA uses CORBA to enable distributed entities to communicate with one another. GAIA has served as our test-bed for the use of ontologies in Pervasive Computing Environments.

We have used GAIA to manage rooms in our Computer Science building. GAIA helps make these rooms smart and responsive to the needs of different users. There are a wide variety of devices that exist in these rooms. These include authentication devices like fingerprint sensors and smart card readers, display devices like large plasma screens, video walls, handheld devices, wearable devices like smart watches and smart rings, various input devices such as touch screens and microphones, etc.. In addition, there are large number of applications and services like music-playing applications, presentation applications and drawing applications. Ontologies provide a very easy way to manage this diversity in our environments.

Unlike many uses of ontologies, we have tightly integrated Semantic Web services into the infrastructure. We implemented an Ontology Server, which is a standard system service that provides a generic interface to manage DAML+OIL ontologies, a Knowledge Base, and logic queries. Any entity in the system can use the Ontology Server.

We have created ontologies (in DAML+OIL) to classify and describe many concepts of the Pervasive Computing Environment. The ontologies include entities of the system (not limited to software) and context information.

The ontologies and Ontology Server are used to augment system services, including:

- Configuration management
- Discovery and matchmaking
- Human Interfaces
- Interoperation of components
- Context Sensitive behavior

The ontologies and Knowledge Base are tightly integrated into the whole system.

2.1 The Ontology Infrastructure in Gaia

We have integrated the use of ontologies in our smart Spaces framework, GAIA. (Figure 1) All the ontologies in GAIA are maintained by an Ontology Server [McGrath et al., 2003]. The Ontology Server asserts the concepts described in the ontologies in the CORBA FaCT Reasoning Engine [Bechhofer et al., 1999; Horrocks et al., 1999] to make sure that they are logically consistent and for answering logical queries. Other entities in GAIA contact the Ontology Server to retrieve descriptions of entities in the environment, meta-information about context or definitions of various terms used in GAIA. It is also possible to support semantic queries (for instance, classification of individuals or subsumption of concepts). Such semantic queries are resolved using the FaCT Reasoning Engine. The Ontology Server registers with the CORBA Naming Service so that it can be discovered by other entities in the environment.

One of the key benefits in using ontologies is that they aid interaction between users and the environment since they concisely describe the properties of the environment and the various concepts used in the environment. With that aim in mind, we have developed an Ontology Explorer, which is a graphical user interface that allows users to browse and search the ontologies in the Space. The Ontology Explorer also allows users to interact with other entities in the Space through it. This interaction with other entities is governed by their properties as defined in the ontology. The Ontology Explorer and Ontology Server are described in more detail in [McGrath et al., 2003].

2.2. Kinds of Ontologies in Gaia

We use ontologies to describe various parts of our pervasive environment, GAIA. In particular, we have ontologies that have meta-data about the different kinds of entities in our environment. We also have ontologies to describe the different kinds of contextual information in our environment. The ontologies used in GAIA are described in more detail in [McGrath et al., 2003].

2.2.1. Ontologies for different entities

Pervasive computing environments have a large number of different types of entities. There are different kinds of devices ranging from small wearable devices and handhelds to large wall displays and powerful servers. There are many services that help in the functioning of the environment. There are different kinds of applications such as music players, slide show viewers, drawing applications, etc.. Finally, there are the users of the environment who have different roles (student, administrator, etc.).

Ontologies help formalize and make available the informal and implicit taxonomy of the different kinds of entities in the system. We have developed ontologies that define the different kinds of entities, provide meta-data about them and describe how they relate to each other. These ontologies are written in DAML+OIL.

A Pervasive Computing Environment is very dynamic; new kinds of entities can be added to the environment at any time. The Ontology Server allows adding new classes and properties to the existing ontologies at any time, by merging new concepts into the system ontology. To do this, a new ontology describing the new entities is first

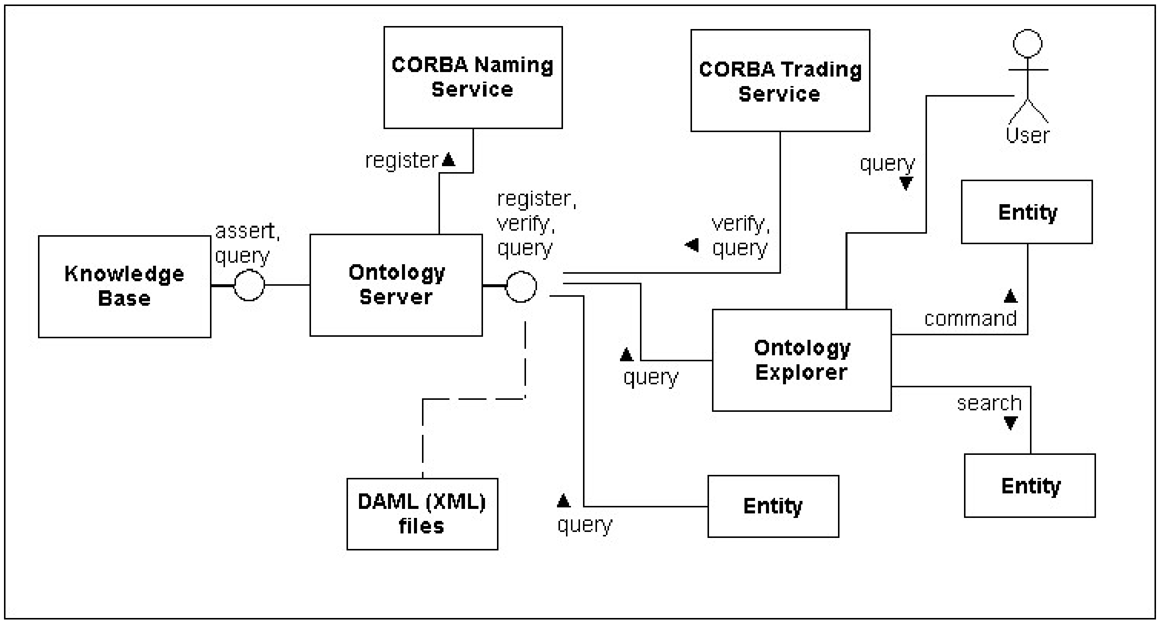

Figure 1. The Ontology Infrastructure of GAIA

developed. The new ontology is then added to the shared ontology using bridge concepts that relate classes and properties in the new ontology to existing classes and properties in the shared ontology. These bridge concepts are typically subsumption relations that define the new entity to be a subclass of an existing class of entities. For example, if a new kind of fingerprint recognizer is added to the system, the bridge concept may state that it is a subclass of "AuthenticationDevices". These bridge concepts are written by the developer of the ontology manually. We have no way at present of automatically generating these bridge concepts, although that would be very useful.

2.2.2. Ontologies for context information

GAIA has a context infrastructure [Ranganathan et al., 2003] that enables applications to obtain and use different kinds of contexts. This infrastructure consists of sensors that sense various contexts, reasoners that infer new context information from sensed data and applications that make use of context to adapt the way they behave. We use ontologies to describe context information. This ensures that the different entities that use context have a common semantic understanding of contextual information.

There are different types of contexts that can be used by applications. These include physical contexts (location and time), environmental contexts (weather, light and sound levels), informational contexts (stock quotes, sports scores), personal contexts (health, mood, schedule, activity), social contexts (group activity, social relationships, whom one is in a room with), application contexts (email, websites visited) and system contexts (network traffic, status of printers). We represent contexts as predicates. We follow a convention where the name of the predicate is the type of context that is being described (like location, temperature or time). An example of a context predicate is Location(Chris, in, Room 2401). This predicate is true if Chris is indeed in Room 2401. Ontologies essentially define the vocabulary and types of arguments that may be used in the predicates.

The use of ontologies to describe context information is helpful in checking the validity of context information. It also makes it easier to specify the behavior of context-aware applications since we know the types of contexts that are available and their structure. We can thus easily construct rules governing application behavior using these well-defined context predicates.

2.3. Uses of Ontologies in a Pervasive Computing Environment

The Ontology Server can be used by any application, component, or service in the GAIA environment. The ontologies that describe entities and context information are used to enable different parts of the pervasive environment interact with each other easily. The ways in which ontologies are used in GAIA are described in more detail in [McGrath et al., 2003].

2.3.1. Configuration Management

A pervasive computing environment is very dynamic, the configuration must change as activities change, and as people and devices enter and leave. Configuration management is very challenging, especially because:

- New entities, never before seen, may enter
- Components need to automatically discover and collaborate with other components

- Entities and components are heterogeneous and autonomous.

Without ontologies, the GAIA environment is configured with scripts and ad hoc configuration files [Roman et al., 2002]. Ontologies can replace these mechanisms with a standard, formal XML language.

Each entity is associated with an XML file that describes its properties. When a new entity is introduced into the system, its description is checked against the existing ontology to see whether it is satisfiable. If the description is not consistent with the concepts described in the ontology, then either the description is faulty (in which case the owner of the entity/context has to develop a correct description of the entity/context), or there are safety or security issues with the new entity or context. For example, the ontology may dictate that all electrical and electronic devices that are to be introduced in an environment (like a smart room) must accept 110V AC power. In that case, if somebody tries to install a new TV that is made for Europe and only takes 220V power, then the description of the new TV would be inconsistent with the ontology and a safety warning may be generated.

Formal ontologies also increase the capability to use descriptions from different, autonomous sources. The DAML+OIL ontologies can be published, to enable autonomous developers and service providers to describe their products with the correct vocabulary. Conversely, autonomous entities can specify the correct formal vocabulary to be used to interpret their descriptions by referring to the relevant DAML+OIL ontology. These actions require more than the URL: the formal semantics defined for DAML+OIL ensures that ontologies from different sources can be used together.

2.3.2. Semantic Discovery and Matchmaking

In our environment, the Ontology Server performs the tasks of semantic discovery and matchmaking. It poses logical queries involving subsumption and classification of concepts to the FaCT Server [Bechhofer et al., 1999; Horrocks et al., 1999], which has knowledge of all concepts used in the environment. Such queries are useful in finding appropriate matches. Other entities in the environment query the Ontology Server to discover classes of components that meet their requirements.

2.3.3. Improved Human Interfaces

Ontologies can be used to make better user interfaces and allow these environments to interact with humans in a more intelligent way. Ontologies describe different parts of the system, the various terms used and how various parts interact with each other. All classes and properties in the ontology also have documentation that describes them in greater detail in user-understandable language. Ontologies enable semantic interoperability between users and the system.

For example, we have defined the term "meeting" as a subclass of "GroupActivity". A meeting is defined to have a location, a time, an agenda (optional) and a set of participants. It also has the following human-understandable comment:
"A meeting is an activity that is performed by a group of people. A meeting involves different people coming together at a particular time or place with a common purpose in mind".
Thus, both humans and automated entities in the environment can get a clear understanding of the term "meeting" by looking it up in the ontology.

We have developed a GUI called the Ontology Explorer that allows users to browse the ontology describing the environment. A user can search for different classes in the ontology. He can then browse the results – for example, he can get documentation about the classes returned, get properties of the class, etc.. The Ontology Explorer is similar to a class browser, but it may browse information about all concepts in the system (like context information, applications, services, terms), not just the software objects.

2.3.4. Improved Inter-operability between entities

The description of the properties of different classes of entities thus allows both users and other automated agents to interact with them more easily by performing searches on them or sending them various commands. This has proved to be one of the major advantages to using ontologies in a pervasive computing environment since it helps simplify the user's and the agent's interaction with such complex systems.

Entities that support searches have their schemas described in the ontology. The ontology also specifies which fields in the query are required and which are optional. Thus any other entity (including users) can browse the ontology to learn the schema and query formats supported by the searchable entity. They can then frame their query and get the results. For example, we have an MP3 Server that exposes a query interface for searching for MP3 files. The schema for querying contains fields like artist name, length of song, etc. This schema is described in the ontology. Other entities, thus, know how to query the MP3 Server.

The same idea is used to let entities interact with one another i.e. by sending commands. Different entities allow different types of actions to be performed on them. For example, the MP3 Server described above allows different commands to be sent to it –start, stop, pause, change volume, etc. In our framework, entities specify the commands they support and the parameters of these commands in an ontology. Other entities can thus send commands to these entities.

2.3.5 Context-Sensitive Behavior

An ontology can improve the robustness and portability of context-aware applications. It is not possible to design in all possible contexts—or even to know what contexts may be used. Ontologies for context information are an important mechanism for adapting to environments. The application specifies rules for context-sensitive behavior using a specific set of context concepts and events (a vocabulary). When the application moves to a new space, the context may be different. This might be due to different sensors, different versions of services, or localizations. If the differences are terminological, an ontology may allow the rules to be "translated" and then work correctly in the new environment.

Context-aware applications in GAIA have rules that describe what actions should be taken in different contexts. In order to write such a rule, an application developer must know the different kinds of contexts available as well as possible actions that can be taken by the application. We have ontologies that describe the different kinds of context information – location, time, temperature, activities of people; and also different applications and what commands can be sent to them.

These ontologies greatly simplify the task of writing rules. We have developed a tool that uses these ontologies to allow a developer to write rules easily. The tool allows him to construct conditions out of the various possible types of contexts available. It then allows him to choose the action to be performed at these contexts from the list of possible commands that can be sent to this application as described in the ontology. Developers can, thus, very quickly, impart context-sensitive behavior to applications by associating context expressions (involving context predicates) with actions. An example of such a rule for a context-sensitive application is:

IF Location(Manuel, Entering, Room 2401) AND Time(morning) THEN play a rock song.

3. Lessons Learned

We have integrated ontologies and Semantic Web technology into our Pervasive Computing infrastructure. This work suggests a number of requirements for future research and development of ontologies.

Our investigations have shown that the Semantic Web technology can be used with CORBA-based infrastructure to solve some important problems for a pervasive computing environment. Our Ontology Server provides a standard interface to a Knowledge Base and logic engine. Ontologies for descriptions of entities and relationships are developed within a Knowledge Engineering Environment and stored as DAML+OIL XML files. Components of the system use the CORBA-based infrastructure to update and query the Ontology Server. In the future, we will extend this work to augment the configuration and management of multiple Spaces, and to augment additional services, such as Quality of Service infrastructure [Wichadakul et al., 2002].

Our system has integrated semantic services to an unprecedented degree. This was enabled partly by the availability of the CORBA FaCT server [Bechhofer et al., 1999] and the Java classes from OILed [Bechhofer et al., 2001; OilEd, 2002]. These packages are a model for what is needed in future software standards:

- A standard API for DAML+OIL (or, more likely, OWL [W3C, 2003b]
- A standard interface for generic Knowledge Base services

Alternative logic engines and Knowledge Bases should be wrapped in a generic interface, so they can be plugged in to infrastructure services. For example, the Open Knowledge Base Connectivity (OKBC) [Chaudhri et al., 1998] could be extended to support DAML (OWL). The Java Theorem Prover (JTP) [Fikes et al., 2003] is a promising step in this direction.

Ontologies and semantic services will play a key role as we develop more sophisticated tools to construct and manage multiple Spaces. It will be important to simplify the construction and maintenance of ontologies, perhaps with a repository of standard ontologies. It will also be important to integrate ontologies with the software generation and management, perhaps using ontologies to semi-automatically generate interfaces. In short, it will be necessary to incorporate successful developments of Knowledge Engineering Environments into the software engineering and configuration management tools of the pervasive computing environment.

A standard upper ontology for services, such as DAML-S [Ankolekar et al., 2002] is a good first step. We foresee the need for standard ontologies for many aspects of the Pervasive Computing Environment, including devices, software services, events, people, places, and things.

Merging (composing) ontologies from multiple autonomous sources is critical for the Pervasive Computing Environment. In particular, it is necessary to incorporate descriptions of new classes of entities (devices, services, components, and so on) and new types of context information (e.g., new sensors) as they are introduced. It should be possible to develop frameworks and editors to assure that the creators of new entities can create descriptions that can be easily merged into system ontologies. Successful research results in merging ontologies must be implemented in standard services and libraries, in order to be integrated into the infrastructure.

A Pervasive Computing Environment poses significant challenges for the architecture and implementation of reasoners and query engines. DAML+OIL (and in the future, OWL [W3C, 2003b]) has proven to be quite useful, especially in combination with a programming interface. However, it seems clear that the DAML and the Description Logic (DL) underlying DAML are necessary but not sufficient for ubiquitous computing applications. Specifically, Description Logics are not suited for some critical aspects of pervasive computing: DL does not deal well with quantitative concepts; including order, quantity, time, or rates. Unfortunately, this kind of reasoning is essential to certain aspects of ubiquitous computing, including, for instance, Quality of Service management [Wichadakul et al., 2002], resource scheduling, and location tracking. Ontologies for pervasive computing environments will require logical models that include spatial and temporal logic, geometry, and other quantitative reasoning. It is unclear whether DAML+OIL should be extended to include additional logical concepts, or whether other kinds of markup languages should be developed for expressing concepts involving these quantitative aspects.

The Pervasive Computing Environment is a long-running, open, real-time system. Maintaining an ontology in real-time as the system evolves presents important challenges for the design and implementation of ontologies and Knowledge Bases. In particular, the system needs to address the issues of:

- Large scale (many thousands of concepts and relations), many hundreds of services using the ontology and KB).
- Incremental updates (add, delete, or modify a few concepts in a large, active KB).
- Persistence and fault-tolerance
- Federation of multiple local Knowledge Bases

Another area that requires investigation is security, privacy, and access control. The Semantic Web as a whole is largely conceived as a completely open system, in which everything is published for everyone to see. It is far from clear how access control could or should be applied, e.g., to the information in an ontology or a KB. Reasoning engines typically can’t enforce security policies, and the DAML language, for instance, has no facility to limit visibility of concepts or attributes. This topic must be addressed in future research.

Acknowledgements

This research is supported in part by the National Science Foundation grant NSF 98-70736, NSF 9970139, and NSF infrastructure grant NSF EIA 99-72884. Important aspects of this study used software from Iona [IONA Technologies Inc., 2001] and University of Manchester [Bechhofer et al., 1999; OilEd, 2002]

References

[Ankolekar et al., 2002] Ankolekar, Anupriya, Burnstein, Mark, Hobbs, Jerry R., Lassila, Ora, Martin, David, McDermott, Drew, McIlraith, Sheila A., Narayanan, Srini, Paolucci, Massimo, Payne, Terry, and Sycara, Katia. DAML-S: Web Service Description for the Semantic Web. In *First International Semantic Web Conference (ISWC)*, Sardinia, 2002.

[Bechhofer et al., 2001] Bechhofer, Sean, Horrocks, Ian, Gable, Carole, and Stevens, Robert. OilEd: a Reason-able Ontology Editor for the Semantic Web. In *KI2001: Advances in Artificial Intelligence*, Vienna, 2001.

[Bechhofer et al., 1999] Bechhofer, Sean, Horrocks, Ian, Patel-Schneider, Peter F., and Tessaris, Sergio. A proposal for a description logic interface. In *International Workshop on Description Logics (DL'99)*, Las Vegas, 1999.

[Berners-Lee et al., 2001] Berners-Lee, Tim, Hendler, James, and Lassila, Ora. The Semantic Web. *Scientific American* 284(5):35-43, 2001.

[Chaudhri et al., 1998] Chaudhri, Vinay K., Farquhar, Adam, Fikes, Richard, Karp, Peter D., and Rice, James P. *Open Knowledge Base Connectivity 2.0.3--Proposed* Stanford Research INstitute, April 9, 1998.

[daml.org, 2003] daml.org. The DARPA Agent Markup Language Homepage, 2003. http://www.daml.org

[Fikes et al., 2003] Fikes, Richard, Frank, Gleb, and Jenkins, Jessica. *JTP: A System Architecture and Component Library for Hybrid Reasoning.* Knowledge Systems Lab KSL-03-01 Stanford University, 2003. http://ksl.stanford.edu/KSL_Abstracts/KSL-03-01.html

[Goble et al., 2002] Goble, Carole, and De Roure, David. The Grid: An Application of the Semantic Web. *ACM SIGMOD Report* 31(4):65-70, 2002.

[Horrocks et al., 1999] Horrocks, Ian, Sattler, Ulrike, and Tobias, Stephan. Practical Reasoning for Expressive Description Logics. In *International Conference on Logic for Programming and Automated Reasoning (LPAR'99)*, Tbilisi, 1999.

[IONA Technologies Inc., 2001] IONA Technologies Inc. ORBacus Trader, Version 2.0.0, 2001. http://www.iona.com/products/orbacus_trader.html

[Lyytinen et al., 2002] Lyytinen, Kalle, and Yoo, Youngjin. Issues and Challenges in Ubiquitous Computing. *Communications of the ACM* 45(12):62-65, 2002.

[McGrath et al., 2003] McGrath, Robert, Ranganathan, Anand, Campbell, Roy, and Mickunas, Dennis. *Use of Ontologies in Pervasive Computing Environments* UIUCDCS-R-2003-2332 UILU-ENG-2003-1719 Department of Computer Science, April, 2003.

[McGrath, 2000] McGrath, Robert E. *Discovery and Its Discontents: Discovery Protocols for Ubiquitous Computing* UIUCDCS-R-99-2132 Department of Computer Science University of Illinois Urbana-Champaign, March 25, 2000.

[OilEd, 2002] OilEd. OilEd, 2002. http://oiled.man.ac.uk/

[Peer, 2002] Peer, Joachim. Bringing Together Semantic Web and Web Services. In *First International Semantic Web Conference (ISWC)*, Sardinia, 2002.

[Ranganathan et al., 2003] Ranganathan, Anand, and Campbell, Roy.H. A Middleware for Context-Aware Agents in Ubiquitous Computing Environments., In ACM/IFIP/USENIX International Middleware Conference, Rio de Janeiro, Brazil, June 16-20, 2003.

[Roman et al., 2002] Roman, Manuel, Hess, Christopher K., Cerqueira, Renato, Ranganathan, Anand, Campbell, Roy H., and Nahrstedt, Klara. GAIA: A Middleware Infrastructure to Enable Active Spaces. *IEEE Pervasive Computing* 1(4):74-83, 2002.

[Trastour et al., 2001] Trastour, David, Bartolini, Claudio, and Gonzalez-Castillo, Javier. *A Semantic Web Approach to Service Description for Matchmaking of Services* HPL-2001-183 HP Laboratories Bristol, July 30, 2001. http://www.hpl.hp.com/techreports

[W3C, 2003a] W3C. The Semantic Web, 2003. http://www.w3.org/2001/sw

[W3C, 2003b] W3C. *Web Ontology Language (OWL) Reference Version 1.0.* W3C Working Draft, 31 March, 2003. http://www.w3.org/TR/2002/WD-owl-ref-20021112/

[Wichadakul et al., 2002] Wichadakul, Duangdao, Gu, Xiaohui, and Nahrstedt, Klara. A Programming Framework for Quality-Aware Ubiquitous Multimedia Applications. In *ACM Multimedia*, Juan Les Pins, 2002.