Using OWL in a Pervasive Computing Broker*

Harry Chen
University of Maryland
Baltimore County, USA
hchen4@cs.umbc.edu

Tim Finin
University of Maryland
Baltimore County, USA
finin@cs.umbc.edu

Anupam Joshi
University of Maryland
Baltimore County, USA
joshi@cs.umbc.edu

ABSTRACT

Computing is moving toward a pervasive context-aware environment in which agents with limited resources will require external support to help them become context-aware. In this paper, we describe an agent based architecture called Context Broker Architecture (CoBrA) to help these agents to acquire, reason about and share context knowledge. A key component in our architecture is an explicit context ontology defined using the Web Ontology Language (OWL). This ontology models the basic concepts of people, agents, places, and presentation events. We also describe a use case scenario and prototype design of CoBrA in an intelligent meeting room environment.

1. INTRODUCTION

Pervasive computing is a vision for the near term future in which devices, services, and software agents will seamlessly integrate and cooperate in support of human objectives – anticipating our needs, negotiating for services, acting on our behalf, and delivering services in an anywhere, any-time fashion. To realize this vision, an important next step is the development of an infrastructure that can help ubiquitous agents, services, and devices become aware of their contexts (including the ability to reason about and to share this knowledge). We are developing a new pervasive context-aware computing infrastructure called Context Broker Architecture (CoBrA) [3] that will help agents behave in an intelligent, context-aware manner.

By context, we mean an understanding of a location, its environmental attributes (e.g., room temperature, noise level, and light intensity), and the people, devices, objects, and software agents it contains. This understanding necessarily extends to modeling the activities and tasks that are taking place in a location as well as the beliefs, desires, commitments, and intentions of the human and software agents involved.

In the past, a number of distributed systems have been developed with a common goal to support pervasive computing (e.g., Intelligent Room [5], Context Toolkit [16], and Cooltown [12]). Although the research of these systems have made tremendous progress in advancing pervasive computing, the resulting implementations, however, have weaknesses in supporting knowledge sharing and context reasoning due to lacking an explicit representation of context ontologies [3, 4].

An explicit representation of ontology consists of a formal model of vocabularies (i.e., classes and properties) and associated semantics (i.e., the relationships between different classes and properties). Without a well-defined ontology, knowledge sharing and context reasoning is often difficult in the previous systems [3].

In the previous systems, ontologies are often defined based on *ad hoc* representation schemes such as a set of programming language objects or data structures. There are two problems with this approach: (i) the use of *ad hoc* representation schemes lacking shared vocabularies can hinder the ability of independently developed agents to interoperate (i.e., to share context knowledge), and (ii) the use of objects and data structures of low expressive power provides inadequate support for context reasoning.

In order to help agents to discover, reason about and communicate contextual information, we must define explicit ontologies for context concepts and knowledge. In this paper, we present a set of ontologies that we have developed to support ubiquitous agents in the CoBrA system. Our ontology (CoBrA ontology) is defined using the Web Ontology Language (OWL) [18], an Semantic Web language being specified by the W3C. The current version of the CoBrA ontology models the basic concepts of people, agents, places and presentation events. It also describes the properties and relationships between these basic concepts including (i) relationships between places, (ii) roles associated with people in presentation events, and (iii) typical intentions and desires of speakers and audience members.

The rest of this document is structured into nine sections. In the next section, we briefly review the Semantic Web and the OWL language. In Section 3, we describe the key features of CoBrA and its design rationale. In Section 4, we present a typical use case scenario of the CoBrA system and point out how CoBrA can help agents to reason about contexts and share knowledge. After our discussion, in Section 5, we describe the key ontology concepts in the latest version of our CoBrA ontology. In Section 6, a prototype system design of the Context Broker Architecture is presented. A brief discussion of related work and some concluding comments are given in Section 7 and Section 8, respectively.

2. THE SEMANTIC WEB AND OWL

Semantic Web is a vision of the next generation World Wide Web in which information is given well-defined meaning, better enabling computers and people to work in cooperation [2]. Research on the Semantic Web is driven by the

*This work was partially supported by DARPA contract F30602-97-1-0215, Hewlett Packard, NSF award 9875433, and NSF award 0209001.

need for a new knowledge representation framework to cope with the explosion of unstructured digital information on the existing Web. Current Semantic Web research focuses on the development of ontology languages and tools for constructing digital information that can be "understood" by computers [2].

The origin of the Semantic Web research goes deep in the roots of Artificial Intelligent research (e.g., knowledge representation and ontology). However, the recent publicly known Semantic Web research begins with the DAML (DARPA Agent Markup Language) effort in the US[1] and the OIL (Ontology Inference Layer) effort in Europe[2]. In late 2000, the original DAML language is combined with many of the ontology modeling features from the OIL language, and the result is the DAML+OIL language.

In late 2001, the World Wide Web Consortium (W3C) established the Web Ontology Working Group with the goal of introducing Semantic Web technologies to the main stream web community. The group has specified a language OWL that is based on DAML+OIL and shares many of its features (e.g., using RDF as the modeling language to define ontological vocabularies and using XML as the surface syntax for representing information [18]).

We have chosen to define our ontology for contexts for three reasons. First, it is much more expressive than RDF or RDF-S allowing us to build more knowledge into the ontology. Second, we chose to use OWL over DAML+OIL because OWL has been designed as a standard and has the backing of a well known and regarded standards organization. Third, from a system implementation point of view, the emergence of ontology inference engines in support for the OWL language (e.g., FaCT [9], RACER [19] and Bubo [20]) leads us to believe adopting OWL will create new opportunities for building more advanced intelligent systems.

3. CONTEXT BROKER ARCHITECTURE

CoBrA is an agent based architecture for supporting context-aware computing in intelligent spaces. Intelligent spaces are physical spaces (e.g., living rooms, vehicles, corporate offices and meeting rooms) that are populated with intelligent systems that provide pervasive computing services to users [11].

Central to our architecture is the presence of an intelligent *context broker* (or broker for short) that maintains and manages a shared contextual model on the behalf of a community of agents. These agents can be applications hosted by mobile devices that a user carries or wears (e.g., cell phones, PDAs and headphones), services that are provided by devices in a room (e.g., projector service, light controller and room temperature controller) and web services that provide a web presence for people, places and things in the physical world (e.g., services keeping track of people's and objects' whereabouts [12]).

In a large-scale intelligent space (e.g., a campus or a building), multiple brokers can form a *broker federation*. Individual broker in a federation is responsible for managing parts of the intelligent space (e.g., a room in a particular building). Brokers are related to each other in some organizational structure (e.g., peer-to-peer or hierarchical) in a federation, and they can periodically exchange and synchronize context knowledge, enabling fault-tolerance (similar to the *persistent broker team* described in [13]).

[1]DAML web site:`http://www.daml.org`

[2]OIL web site: `http://www.ontoknowledge.org/oil/`

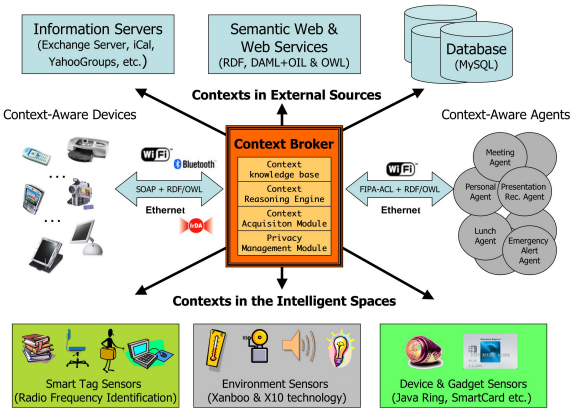

Figure 1: An intelligent context broker acquires context information from devices, agents and sensors in its environment and fuses it into a coherent model that is then shared with the devices and their agents.

In an intelligent space, the primary responsibilities of a broker are to (i) acquire context information from heterogeneous sources and reason about this information to maintain a consistent context model, (ii) help distributed agents to share context knowledge through the use of common ontologies, agent communication languages and protocols, and (iii) protect the privacy of users by establishing and enforcing user-defined policies while sharing sensitive personal information with agents in the community.

A context broker has four main functional components:

1. **Context Knowledge Base**: a persistent store for context knowledge in an intelligent space. This knowledge base provides a set of API's for other components to assert, delete, modify, and query stored knowledge.

2. **Context Reasoning Engine**: a reactive inference engine that reasons over the knowledge base. Its main function is to deduce additional knowledge from information acquired from external sources and to maintain the consistency of the knowledge base.

3. **Context Acquisition Module**: a collection of predefined procedures for acquiring information from the external sources. It serves as a middleware abstraction for acquiring contexts from heterogeneous sources (e.g., physical sensors, web services, databases, devices and agents).

4. **Privacy Management Module**: a set of communication protocols and behavior rules that the broker follows when performing privacy management tasks (i.e., negotiate privacy policies with new users and enforcing these policies when sharing information with agents in the community).

4. ONTOLOGIES IN COBRA

In a pervasive computing environment, individual agents may have limited resources to acquire, reason about and share contexts. In CoBrA, the role of a broker is to help

these resource-limited agents – e.g., to reason about contexts and share context knowledge.

In the next two sections, we will describe a typical multi-agent scenario of CoBrA (Section 4.1) and point out how explicitly represented ontologies enable knowledge sharing and context reasoning (Section 4.2).

4.1 An Intelligent Meeting Room Scenario

R210 is an intelligent meeting room with RFID sensors[3] embedded in the walls and furniture for detecting the presence of the users' devices and clothing. As Alice enters the room, these sensors inform the R210 broker that a cell phone belonging to her is present and the broker adds this fact in its knowledge base.

As she sits, the agent on Alice's Bluetooth enabled cell phone discovers R210's broker and engages in a "hand shake" protocol (e.g. authenticates agent identities and establishes trust [10]) after which it informs the broker of Alice's privacy policy. This policy represents Alice's desires about what the broker should do and includes (i) the context information about Alice that the broker is permitted or prohibited from storing and using (e.g., yes to her location and roles, no to the phone numbers she calls), (ii) other agents that the broker should inform about changes in her contextual information (e.g., keeping Alice's personal agent at home informed about her location contexts), and (iii) the permissions for other agents to access Alice's context information (e.g., all agents in the meeting room can access Alice's contexts while she is in the room).

After receiving Alice's privacy policy, the broker creates a profile for Alice that defines rules and constraints the broker will follow when handling any context knowledge related to Alice. For example, given the above policy, the profile for Alice would direct the broker (i) to acquire and reason about Alice's location and activity contexts, (ii) to inform Alice's personal agent at home when Alice's contexts change, and (iii) to share her contexts with agents in the meeting room.

Knowing Alice's cell phone is currently in R210 and having no evidence to the contrary, the broker concludes Alice is also there. Additionally, because R210 is a part of the Engineering building, which in turn is a part of the Campus, the broker concludes Alice is located in Engineering building and on campus. These conclusions are asserted into broker's knowledge base.

Following the profile, the broker informs Alice's personal agent of her whereabouts. On receiving this information about Alice, her personal agent attempts to determine why Alice is there. Her Outlook calendar has an entry indicating that she is to give a presentation on Campus about now, so the personal agent concludes that Alice is in R210 to give her talk and informs the R210 broker of it's belief.

On receiving information about Alice's intention, the R210 broker shares this information with the projector agent and the lighting control agent in the ECS 210. Few minutes later, the projector agent downloads the slides from Alice's personal agent and sets up the projector, the lighting control agent dims the room lights.

4.2 Knowledge Sharing & Context Reasoning

An explicit representation of ontologies can be used to enable knowledge sharing and provide a means for reasoning. In the scenario above, three distinct but related types of context information are used: (i) location contexts ("Where is Alice?"), (ii) activity context ("What is she doing?"), and (iii) agent contexts ("What might she want to do in this context?"). Note that an understanding of these contexts is only possible because of all different types of agents (including sensors and devices) share information with each other using common ontologies, and the broker is able derive additional information about Alice's location because it has a model of the rooms and buildings involved and has a rudimentary model of spatial relationships.

Context reasoning may also take place in detecting inconsistent beliefs about certain contexts. For example, in a pervasive computing environment, information sensed from the physical world can be both noisy and ambiguous (e.g., sensors may report that the same person is present in two different room at the same time). With ontologies, it is possible to guide the context reasoning process to detect and resolve this inconsistent knowledge (e.g., in our CoBrA ontology, containment relationships between different classes of locations can be used to detect this type of knowledge inconsistency).

5. THE COBRA ONTOLOGY

This section describes key ontology concepts in the current version of the CoBrA ontology (v0.2)[4]. This ontology defines a vocabulary for describing people, agents, places and presentation events for supporting an intelligent meeting room system on a university campus. It also defines a set of properties and relationships that are associated with these basic concepts.

Figure 2 shows a complete list of the names of the classes and properties in the CoBrA ontology. In v0.2, there are 41 classes (i.e., RDF resources that are type of `owl:class`) and 36 properties (i.e., RDF resources that are type of either `owl:ObjectProperty` or `owl:DatatypeProperty`). These ontologies are expressed using the OWL/XML syntax [17].

Our ontology is categorized into four distinctive but related themes: (i) concepts that define physical places and their associated spatial relations (e.g., containment, social and organizational properties)[5], (ii) concepts that define agents (i.e., both human agents and software agents) and their associated attributes, (iii) concepts that describe the location contexts of an agent on a university campus, and (iv) concepts that describe the activity contexts of an agent, including the roles of speakers and audiences and their associated desires and intentions in a presentation event. In the rest of this section, we discuss each of these four themes.

5.1 Places

The notion of a place in CoBrA is presently restricted to a set of physical locations that are typically found on a university campus. These locations include *campus*, *building*, *room*, *hallway*, *stairway*, *restroom*, and *parking lot*. These physical locations are all assumed have well-defined spatial boundaries (e.g., all locations can be uniquely identified by geographical coordinates – longitude and latitude). In addition, all locations on a university campus have identifiable

[3]RFID stands for Radio Frequency Identification (see also `http://www.rfid.org`)

[4]A complete version of the ontology is available at `http://daml.umbc.edu/ontologies/cobra/0.2/cobra-ont`

[5]In v0.2, only containment relations are defined, additional properties will be included in the next version of the ontology.

CoBrA Ontology Classes		CoBrA Ontology Properties	
"Place" Related	Agents' Location Context	"Place" Related	Agent's Location Context
Place AtomicPlace CompoundPlace Campus Building AtomicPlaceInBuilding AtomicPlaceNotInBuilding Room Hallway Stairway OtherPlaceInBuilding Restroom Gender LadiesRoom MensRoom ParkingLot	ThingInBuilding SoftwareAgentInBuilding PersonInBuilding ThingNotInBuilding SoftwareAgentNotInBuilding PersonNotInBuilding ThingInRoom SoftwareAgentInRoom PersonInRoom	latitude longitude hasPrettyName isSpatiallySubsumedBy spatiallySubsumes accessRestricted-ToGender lotNumber	locatedIn locatedInAtomicPlace locatedInRoom locatedInRestroom locatedInParkingLot locatedinCompoundPlace locatedInBuilding locatedInCampus
"Agent" Related	Agent's Activity Context	"Agent" Related	Agent's Activity Context
Agent Person SoftwareAgent Role SpeakerRole AudienceRole IntentionalAction ActionFoundInPresntation	PresentationSchedule EventHappeningNow PresentationHappeningNow RoomHasPresentationHappeningNow ParticipantOfPresentation-HappeningNow SpeakerOfPresentationHappeningNow AudienceOfPresentationHappeningNow PersonFillsRoleInPresentation PersonFillsSpeakerRole PersonFillsAudienceRole	hasContactInformation hasFullName hasEmail hasHomePage hasAgentAddress fillsRole isFilledBy intendsToPerform desiresSomeone-ToAchieve	participatesIn startTime endTime location hasEventHappeningNow invitedSpeaker expectedAudience presentationTitle presentationAbstract presentation eventDescription eventSchedule

Figure 2: A complete list of the names of the classes and properties in the CoBrA ontology (v0.2).

string names that are assigned to them by some official bodies (e.g., by the university administration).

When modeling physical locations, we define a class called `Place` which generalizes all type of locations on a campus. This abstract class defines a set of properties that are common to all concrete physical location classes, which consists of `longitude`, `latitude` and `hasPrettyName`.

`Place` classes (including subclasses) participate in containment relations. These relationships are defined by two related object properties[6] called `spatiallySubsumes` and `isSpatiallySubsumedBy`. The former describes the subject of this property spatially subsumes the object of this property (e.g., a building spatially subsumes a room in the building), and the latter describes the subject of this property is spatially subsumed by the object of this property (e.g., a room in the building is spatially subsumed by the building). In the context of the OWL language, these two properties are defined as an inverse property of each other.

Note that in the current version of the ontology, the domain and the range of both `spatiallySubsumes` and `isSpatiallySubsumedBy` properties are of the class type `Place`. In other word, these two properties cannot be used to make statements about the containment of a person, agent or object in a physical place. In Section 5.2, we will describe alternative constructs for expressing this type of statements.

In addition to containment relationships, physical places may also have events and activities associated with them (e.g., a meeting may be taken place in a room, or an annual festival may be taken place on a university campus).

[6]This refers to the `owl:ObjectProperty` property

In order to make statements about some events that are currently associated with a particular place, we introduce an additional object property called `hasEventHappening-Now`. The domain and range of this property are of the class `Place` and the class `EventHappeningNow`, respectively. The `EventHappeningNow` class represents a set of all events that are currently taking place (details of this class is discussed in Section 5.4).

```
<owl:Class rdf:ID="Place">
  <owl:unionOf rdf:parseType="Collection">
    <owl:Class rdf:about="#AtomicPlace"/>
    <owl:Class rdf:about="#CompoundPlace"/>
  </owl:unionOf>
  ...
</owl:Class>

<owl:Class rdf:ID="AtomicPlace">
  <rdfs:subClassOf rdf:resource="#Place"/>
  <rdfs:subClassOf>
    <owl:Restriction>
      <owl:onProperty
          rdf:resource="#isSpatiallySubsumedBy"/>
      <owl:allValuesFrom
          rdf:resource="#CompoundPlace"/>
    </owl:Restriction>
  </rdfs:subClassOf>
  <rdfs:subClassOf>
    <owl:Restriction>
      <owl:onProperty
          rdf:resource="#spatiallySubsumes"/>
      <owl:cardinality>0</owl:cardinality>
    </owl:Restriction>
  </rdfs:subClassOf>
  <owl:unionOf rdf:parseType="Collection">
    <owl:Class rdf:about="#AtomicPlaceInBuilding"/>
    <owl:Class
        rdf:about="#AtomicPlaceNotInBuilding"/>
  </owl:unionOf>
</owl:Class>
```

```
<owl:Class rdf:ID="CompoundPlace">
  <rdfs:subClassOf rdf:resource="#Place"/>
  <owl:disjointWith rdf:resource="#AtomicPlace"/>
  <rdfs:subClassOf>
    <owl:Restriction>
      <owl:onProperty
          rdf:resource="#isSpatiallySubsumedBy"/>
      <owl:allValuesFrom
          rdf:resource="#CompoundPlace"/>
    </owl:Restriction>
  </rdfs:subClassOf>
  <rdfs:subClassOf>
    <owl:Restriction>
      <owl:onProperty
          rdf:resource="#spatiallySubsumes"/>
      <owl:allValuesFrom
          rdf:resource="#Place"/>
    </owl:Restriction>
  </rdfs:subClassOf>
  <owl:unionOf rdf:parseType="Collection">
    <owl:Class rdf:about="#Campus"/>
    <owl:Class rdf:about="#Building"/>
  </owl:unionOf>
</owl:Class>
```

Figure 3: A partial ontology definition of the `AtomicPlace` & `CompoundPlace` classes in OWL/XML syntax

5.1.1 *AtomicPlace*

From the list of concrete physical locations that we have

mentioned (i.e., campus, building, room, hallway, stairway, etc.), some of these locations usually do not contain (spatially subsume) other physical locations. For example, hallways, stairways and rooms in a building usually are not usually considered to be a type of physical place that contains other places.

For this reason, we introduce an abstract class called `AtomicPlace` to represent a set of all physical places that do not contain other physical places. This class inherits all properties from its superclass `Place`. However, it puts restrictions on the range of the two properties `spatiallySubsumes` and `isSpatiallySubsumedBy`. In this `AtomicPlace` class, the cardinality of the property `spatiallySubsumes` is 0, indicating all instances of this class do not contain any other physical places. On the other hand, the range of the property `isSpatiallySubsumedBy` is restricted to the class `CompoundPlace`, which is a subclass of `Place`. This `CompoundPlace` class represents all physical places that may contain other physical places. Figure 3 shows partial representation of these classes in OWL/XML syntax.

Some subclasses of the `AtomicPlace` class include `Room`, `Hallway`, `Stairway`, `Restroom`, `LadiesRoom`, `MensRoom` and `ParkingLot`.

5.1.2 CompoundPlace

The `AtomicPlace` class represents a set of places that contains zero number of `Place` instances, the `CompoundPlace` class represent places that contain at least one `Place` instances. This class is also a subclass of `Place`.

Being a subclass of the `Place` class, `CompoundPlace` inherits all properties from its parent class. In order to express all instances of the `CompoundPlace` class should only be spatially subsumed by instances of other `CompoundPlace`, the range of this class's property `isSpatiallySubsumedBy` is restricted to have class type `CompoundPlace`. This restriction excludes all instances of the `CompoundPlace` class to be spatially subsumed by instances of the `AtomicPlace`.

5.2 Agents

The agent class in CoBrA represents both humans agents and software agents. Human agents are users in an intelligent space. Software agents, on the other hand, are autonomous computing entities that provide services to users (either directly or indirectly) in an associated space.

All agents have associated properties that describes their contact information, which includes uniquely identifiable names, URLs for their home pages, and email addresses. In addition, agents are assumed to have certain roles in different events and activities (e.g., a person can have the speaker role in a presentation event, and device agents in the close vicinity may take on the presentation assistant role during the presentation session). Different roles may give rise to different desires and intentions of an agent.

In the CoBrA ontology, the notions of desire and intention are both associated with actions[7]. Specifically, the notion of desire is defined as an agent's desire for some actions to be achieved by other agents (e.g., a person with the speaker role may desire some service agents to dim the lights when his presentation starts), and the notion of intention is defined as an agent's commitment to perform some particular actions (e.g., a person with the audience role may intend to download a copy of the slides after attending a presentation event).

To begin our ontology modeling for agents, we introduce a general class called `Agent`, which is a set of all human agents and computational agents. We define the class `Person` to represent human agents and the class `SoftwareAgent` to represent computational agents (both of which are subclasses of the `Agent` class and disjoints with each other). All agents in our ontology are associated with properties that describe their contact information. To generalize properties that serve as descriptions of contact information, we define an object property called `hasContactInformation`. From this property, we further define sub-properties of contact information, which consist of `hasFullName`, `hasEmail`, `hasHomePage` and `hasAgentAddress`.

5.2.1 Role

In our ontology, the class `Role` represents a set of all roles that can be associated with an agent. In other words, it is an abstract class that generalizes all possible types of agent roles in the CoBrA ontology. In v0.2 of the ontology, pre-defined subclasses of `Role` consist of `SpeakerRole` and `AudienceRole`.

To associate roles with an agent, the object properties `fillsRole` and `isFilledBy` are defined. In the context of the OWL language, these two properties are inverse property of each other – `fillsRole` has domain `Agent` and range `Role`, and `isFilledBy` has domain `Role` and range `Agent`.

```
<owl:Class rdf:ID="Role"/>

<owl:ObjectProperty rdf:ID="fillsRole">
  <rdfs:domain rdf:resource="#Agent"/>
  <rdfs:range rdf:resource="#Role"/>
</owl:ObjectProperty>

<owl:ObjectProperty rdf:ID="isFilledBy">
  <owl:inverseOf rdf:resource="#fillsRole"/>
</owl:ObjectProperty>

<owl:Class rdf:ID="SpeakerRole">
  <owl:intersectionOf rdf:parseType="Collection">
    <owl:Class rdf:about="#RoleInPresentation"/>
    <owl:Restriction>
      <owl:onProperty
rdf:resource="#intendsToPerform"/>
      <owl:hasValue
rdf:resource="#GivePPTPresentation"/>
    </owl:Restriction>
  </owl:intersectionOf>
</owl:Class>

<owl:Class rdf:ID="AudienceRole">
  <owl:intersectionOf rdf:parseType="Collection">
    <owl:Class rdf:about="#RoleInPresentation"/>

    <owl:Restriction>
      <owl:onProperty
        rdf:resource="#intendsToPerform"/>
      <owl:hasValue
        rdf:resource="#AttendPresentation"/>
    </owl:Restriction>
  </owl:intersectionOf>
</owl:Class>
```

```
<owl:Class rdf:ID="IntentionalAction"/>

<owl:Class rdf:ID="ActionFoundInPresentation">
  <rdfs:subClassOf rdf:resource="#IntentionalAction"/>

  <owl:oneOf rdf:parseType="Collection">
    <owl:Thing rdf:about="#GivePPTPresentation"/>
    <owl:Thing rdf:about="#SetupPPTPresentation"/>
    <owl:Thing rdf:about="#AdjustLightIntensity"/>
    <owl:Thing rdf:about="#DistributeHangouts"/>
    <owl:Thing rdf:about="#AttendPresentation"/>
    <owl:Thing rdf:about="#AcquireHangouts"/>
    <owl:Thing rdf:about="#ExchangeContact"/>
  </owl:oneOf>
</owl:Class>

<owl:ObjectProperty rdf:ID="intendsToPerform">
  <rdfs:domain>
    <owl:Class>
      <owl:unionOf rdf:parseType="Collection">
                <owl:Class rdf:about="#Role"/>
                <owl:Class rdf:about="#Agent"/>
      </owl:unionOf>
    </owl:Class>
  </rdfs:domain>
  <rdfs:range rdf:resource="#IntentionalAction"/>
</owl:ObjectProperty>
<owl:ObjectProperty rdf:ID="desiresSomeoneToAchieve">
  <rdfs:domain>
    <owl:Class>
      <owl:unionOf rdf:parseType="Collection">
                <owl:Class rdf:about="#Role"/>
                <owl:Class rdf:about="#Agent"/>
      </owl:unionOf>
    </owl:Class>
  </rdfs:domain>
  <rdfs:range rdf:resource="#IntentionalAction"/>
</owl:ObjectProperty>
```

Figure 4: This is a partial definition of the concepts related to roles, intentions and desires in an intelligent meeting room system.

5.2.2 Intentional Actions

All actions in CoBrA are defined as instances of the class `IntentionalAction`. Informally, intentional actions are actions that an agent performs intentionally and with certain goals in mind. In our design, we assume domain applications will extend this class to define specialized subclasses and instances. To support the construction of intelligent meeting room systems, we have pre-defined a set of concrete instances of `IntentionalAction` that are common in presentation events (see Figure 4).

All instances of the `IntentionalAction` class (or its sub-

[7]the semantics of an action is not formal defined in the current version of the ontology. In v0.2, all instances of actions are assumed to be atomic.

classes) can be associated with either an instance of the `Role` class or the `Agent` class through object properties `intendsToPerform` or `desiresSomeoneToAchieve`. The domain of these two properties are defined to be a union of the class `Role` and `Agent` (see Figure 4).

5.3 Agent's Location Context

In the last two sections, we have described a set of CoBrA ontology concepts for physical locations and agents. In this section, we will discuss additional concepts for modeling the location context of agents.

By location context, we mean a collection of dynamic knowledge that describes the location of an agent. In the context of the OWL language, this knowledge is a collection of RDF statements that describe the location property of an agent. To model the location property of an agent, we introduce an object property called `locatedIn`, which has range `Place`[8].

Physical locations, as we have discussed previously in Section 5.1, are categorized into two distinctive classes namely `AtomicPlace` (e.g., hallways and rooms) and `CompoundPlace` (e.g., campus and building). Following the semantics of these two classes, *no agent can locate in two different atomic places at the same time, but any agent can locate in two or more compound places at the same time.* In the context of the OWL language, any two instances of the `AtomicPlace` class are different if and only if both instances have distinctive object values[9] for the same class property.

To capture the notion an agent can be in an atomic and a compound place, from the `locatedIn` property we define two sub-properties called `locatedInAtomicPlace` and `locatedInCompoundPlace`. The former restricts its range to the `AtomicPlace` class, and the latter restricts its range to the `CompoundPlace` class. From these two properties, we can define additional properties that further restricts the type of physical place an agent is located in. For example, `locatedInRoom`, `locatedInRestroom` and `locatedInParkingLot` are sub-properties of `locatedInAtomicPlace`; `locatedInCampus` and `locatedInBuiding` are sub-properties of `locatedInCompoundPlace`.

Since all agents in CoBrA are associated with different types of location properties, we can generalize subsets of these agents in according to their location properties. To do so, we define `PersonInBuilding` and `SoftwareAgentInBuilding` to represent a set of people and software agents who are located in a building, respectively. The complement of these classes are `PersonNotInBuilding` and `SoftwareAgentNotInBuilding`.

5.4 Agent's Activity Context

An agent's activity context is similar to its location context – a collection of dynamic knowledge about certain aspects of an agent's situational condition. While location context describes the location at which the agent is situated, activity context describes activities in which the agent participates. In the current version of the ontology, the notion of an activity is restricted to represent a set of all typical group activity events in a meeting room (meeting, presentation and discussion)[10].

Activity events are assumed have schedules. For presentation events, we have defined `PresentationSchedule` class to represent their schedules. Presentation schedules are defined to have `startTime`, `endTime` and `location` properties, and each of which respectively represents the start time of a presentation, the end time of a presentation and the location of a presentation event. Each presentation event has one or more invited speaker and an audience. These two concepts are defined using the `invitedSpeaker` and `expectedAudience` properties. In addition to start time, end time and location, the schedule of a presentation usually includes a title and an abstract of the presentations. To model these two concepts, we introduce `presentationTitle` and `presentationAbstract` properties.

An agent's activity context is usually associated with activity events that are currently happening. For example, the activity context of a speaker includes the presentation event that he/she is giving the presentation at. To model this, we introduce the `PresentationEventHappeningNow` class. This class is a subclass of the `EventHappeningNow` class which models an event with the time predicate "now".

For a presentation that is currently happening, we can specialize the type of rooms at which the event takes place. For example, a room that has an ongoing presentation event is defined as `RoomHasPresentationEventHappeningNow`, which is a subclass of `Room` and restricts the range of its `hasEventHappeningNow` property to the class `PresentationSchedule`. In addition, we can also specialize people who has various roles in an on-going event. For example, a set of all people who have the speaker role of some on-going presentation event is defined as the `SpeakerOfPresentationHappeningNow` class. Similarly, we define the `AudienceOfPresentationHappeningNow` class to represent a set of all people who have the audience role of some on-going presentation event.

6. A PROTOTYPE SYSTEM DESIGN

In the last section, we have describe the key concepts in the CoBrA ontology. In this section, we present a prototype system design for realizing the intelligent meeting room scenario that we have described in Section 4.1.

In our prototype design (see Figure 5), there are five major system components: (i) a context broker, (ii) a personal agent of the user, (iii) a projector agent, (iv) a Bluetooth-enabled cell phone that the user carries, and (v) devices and clothes tagged with RFID tags. The context broker, the personal agent, and the projector agent will be implemented using the JADE agent development framework [1], a FIPA compliant Java agent development library. These agents will communicate with each other using the standard FIPA-ACL. When these agents communicate for the purpose of sharing context knowledge, the OWL language will be used as the content language for encoding context knowledge. Because all agents in CoBrA are assumed to share a common ontology (i.e., CoBrA ontology) for representing context knowledge, both the projector agent and the personal agent can interoperate with the context broker even if their internal implementations are independently designed

[8]The domain of this property is `owl:Thing`, indicating any thing may be located in some physical place.

[9]an object value refers to the object in an N-triple statement (i.e. (<subject>, <predicate>, <object>)

[10]In v0.2 of the ontology, we have only included concepts related to presentation events. In the future version, we will extend the ontology to includes other activity events

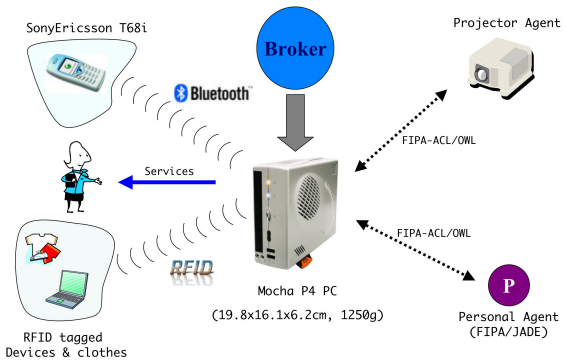

Figure 5: In our prototype system, the context broker will operate on a MOCHA PC, an all-in-one design mini book-size PC. It will be integrated with Bluetooth wireless communication and RFID readers for detecting people presence. The broker will communicate with software agents via FIPA-ACL/OWL.

(i.e., the interoperability of these agents are not completely pre-defined but rather achieved through ontology sharing).

In addition to the agents, our design also includes the use of a SonyEricsson T68i cell phone, which is component (iv). On this Bluetooth-enabled cell phone, users will store their personal policies (e.g., similar to the policy described in Section 4.1). As users enter the meeting room, they will submit their personal policies to the broker through Bluetooth networks. These policies will be expressed using the XML/OWL language syntax following the CoBrA ontology[11].

In order to detect the presence of users, we will label user devices and clothing with RFID tags, similar to the approach used in the CoolAgent RS [4] system. We are planning to deploy a number of RFID readers in our prototype environment, for example, placing readers next to the meeting room's door and underneath the tables in the meeting room.

7. RELATED WORK

Our work is closely related to other pervasive and context-aware computing research such as Intelligent Room, Context Toolkit, Cooltown, One.World [8] and Centaurus [11]. In comparison to the previous systems, our design of the Context Broker Architecture takes a knowledge representation approach to build ontologies of contexts and attempts to use Semantic Web language (i.e., OWL) as the content language in agent communication.

An explicit representation of context ontologies distinguishes CoBrA from other context-aware systems. While the previous systems rely on *ad hoc* representations of contexts to exchange information, CoBrA takes a knowledge representation approach allowing context knowledge to be reasoned and shared through a well-defined ontology model.

As in other systems (e.g., Context Toolkit and Cooltown), the CoBrA ontology models concepts for describing user locations. These concepts are useful for guiding the decision making of context-aware applications [16, 5, 11]. Nevertheless, none of the previous systems have explored the space and spatial relationship aspects of the location contexts (i.e., information that describes the whole physical space that surrounds a particular location and its relationship to other locations). Modeling space and spatial relationships are important in CoBrA. We currently have a simple model of space and spatial relationships (see Section 5.1). In the DAML+OIL community, recent discussions on the daml-spatial mailing list have initiated the work to develop a Semantic Web version of the spatial ontology based the SUO [14] and Cyc [6]. In the future, we plan on using, if possible, or at least mapping to, if feasible, one of these consensus ontologies for space.

In addition to using OWL as an ontology language for modeling contexts, we also attempt to use OWL as the content language in agent communication. Our approach is similar to the use of OWL in the TAGA system (Travel Agent Game in Agentcities) [22]. In TAGA, collections of agent communication primitives (e.g., action, result, query, sender, and receiver) are defined using OWL, forming ontologies for agent communications[12]. Using these ontologies, agents can express their reasons for communicating with other agents (i.e., making propositions and querying for information).

Using OWL as the content language in agent communication will allow the underlying agent implementations to be better integrated with Semantic Web technologies (e.g., ontology inference engines and Semantic Web query languages [7]). In contrast to the use of other content languages such FIPA-SL, KIF and XML, the use of OWL as a content language helps to simplify the underlying implementations for composing communication messages (e.g., avoiding multi-level parsing implementations for translating contents into internal knowledge representation).

8. CONCLUSION AND FUTURE WORK

Computing is moving towards pervasive, context-aware, environments in which resource-limited agents will require external supports to help them to become aware of their contexts. The Context Broker Architecture described in this paper will help these agents to acquire, reason, and share contextual knowledge. A key component of this infrastructure is an explicit representation of context ontologies expressed in the OWL language. Without this ontology, inconsistent and ambiguous context knowledge cannot be easily detected and resolved, and acquired context knowledge cannot be shared with independently developed agents.

We are developing an OWL reasoning engine called F-OWL[13] to support the use case described in Section 4.1. This reasoning engine is implemented using Flora-2 in XSB [15], an object-oriented knowledge base language and application development platform that translates a unified language of F-logic, HiLog, and Transaction Logic into the XSB deductive engine [21].

We plan to prototype an intelligent context broker and integrate this broker with the Centaurus systems (a framework for building pervasive computing services developed

[11]In the next version of the CoBrA ontology (v0.3), we will use concepts from the policy language REI [10]

[12]FIPA OWL content language ontology is available at `http://taga.umbc.edu/taga2/owl/fipaowl.owl`

[13]F-OWL web site: `http://umbc.edu/~hchen4/fowl/`

at UMBC) [11]. The objective is to create a pervasive context-aware meeting room in the newly constructed Information Technology and Engineering Building on UMBC's main campus[14].

9. REFERENCES

[1] F. Bellifemine, A. Poggi, and G. Rimassa. Developing multi agent systems with a fipa-compliant agent framework. *Software - Practice And Experience*, 31(2):103–128, 2001.

[2] T. Berners-Lee, J. Hendler, and O. Lassila. The semantic web. *Scientific American*, may 2001.

[3] H. Chen. An intelligent broker architecture for context-aware systems. PhD. dissertation proposal. `http://umbc.edu/~hchen4/`, 2003.

[4] H. Chen, S. Tolia, C. Sayers, T. Finin, and A. Joshi. Creating context-aware software agents. In *Proceedings of the First GSFC/JPL Workshop on Radical Agent Concepts*, 2001.

[5] M. H. Coen. Design principles for intelligent environments. In *AAAI/IAAI*, pages 547–554, 1998.

[6] Cyccorp Inc. *The Upper Cyc Ontology*, 1997. `http://www.cyc.com/cyc-2-1/cover.htm`.

[7] R. Fikes, P. Hayes, and I. Horrocks. *DAML Query Language (DQL)*, 2002.

[8] R. Grimm, T. Anderson, B. Bershad, and D. Wetherall. A system architecture for pervasive computing. In *Proceedings of the 9th ACM SIGOPS European Workshop*, pages 177–182, 2000.

[9] I. Horrocks, U. Sattler, and S. Tobies. Practical reasoning for expressive description logics. In H. Ganzinger, D. McAllester, and A. Voronkov, editors, *Proceedings of the 6th International Conference on Logic for Programming and Automated Reasoning (LPAR'99)*, number 1705 in Lecture Notes in Artificial Intelligence, pages 161–180. Springer-Verlag, 1999.

[10] L. Kagal, T. Finin, and A. Joshi. A policy language for a pervasive computing environment. In *IEEE 4th International Workshop on Policies for Distributed Systems and Networks*, 2003.

[11] L. Kagal, V. Korolev, H. Chen, A. Joshi, and T. Finin. Centaurus : A framework for intelligent services in a mobile environment. In *Proceedings of the International Workshop on Smart Appliances and Wearable Computing (IWSAWC)*, 2001.

[12] T. Kindberg and J. Barton. A Web-based nomadic computing system. *Computer Networks (Amsterdam, Netherlands: 1999)*, 35(4):443–456, 2001.

[13] S. Kumar, P. R. Cohen, and H. J. Levesque. The adaptive agent architecture: Achieving fault-tolerance using persistent broker teams. In *Proceedings of the Fourth International Conference on Multi-Agent Systems*, pages 159–166, 2000.

[14] I. Niles and A. Pease. Towards a standard upper ontology. In *Proceedings of the 2nd International Conference on Formal Ontology in Information Systems (FOIS-2001)*, 2001.

[14]ITE building construction live feed: `http://www.cs.umbc.edu/ITE/ITE.html`

[15] K. Sagonas, T. Swift, D. S. Warren, J. Freire, P. Rao, B. Cui, and E. Johnson. *The XSB Programmers' Manual*, 2002.

[16] D. Salber, A. K. Dey, and G. D. Abowd. The context toolkit: Aiding the development of context-enabled applications. In *CHI*, pages 434–441, 1999.

[17] M. K. Smith, C. Welty, and D. McGuinness. Owl web ontology language guide. `http://www.w3.org/TR/owl-guide/`, 2003.

[18] F. van Harmelen, J. Hendler, I. Horrocks, D. L. McGuinness, P. F. Patel-Schneider, and L. A. Stein. Owl web ontology language reference. `http://www.w3.org/TR/owl-ref/)`, 2002.

[19] R. M. Volker Haarslev. Description of the racer system and its applications. In *Proceedubgs International Workshop on Description Logics (DL-2001)*, 2001.

[20] R. Volz, S. Decker, and D. Oberle. Bubo - implementing owl in rule-based systems. 2003.

[21] G. Yang and M. Kifer. *Flora-2: User's Manual*. Department of Computer Science, Stony Brook University, Stony Brook, 2002.

[22] Y. Zou, T. Finin, L. Ding, H. Chen, and R. Pan. Taga: Trading agent competition in agentcities. In *IJCAI-2003 Trading Agent Conmpetition Workshop*, 2003.