

Context Life Cycle Management in Smart Space Environments*

Wan-rong Jih
wanrong@ntu.edu.tw

Chi-Chia Huang
chifatty@gmail.com

Jane Yung-jen Hsu
yjhsu@csie.ntu.edu.tw

Department of Computer Science and Information Engineering
National Taiwan University

ABSTRACT

Life cycle management plays an essential role in context-aware systems, it needs to aware surrounding contexts and adapt to changing contexts in highly dynamic environments. Consequently, context management is different from traditional approaches due to the characteristics of contextual information are dynamic, transient, and fallible. Understanding the life cycle of a context is the key issue to maintain consistent contextual information. Accordingly, we propose an ontology-based model for representing, deducing, and managing consistent contextual information. In addition, an agent-based architecture supports the process of context life cycle management.

Categories and Subject Descriptors

K.6.3 [Management of Computing and Information Systems]: Software Management—*Software process*; I.2.11 [Artificial Intelligence]: Distributed Artificial Intelligence—*Intelligent agents, Multiagent systems*; H.1 [Models and Principles]: Miscellaneous

General Terms

Management, Design

Keywords

Context-aware systems, Context management, Ontology, multi-agent architecture

1. INTRODUCTION

Researches on smart spaces, such as Aware Home[1], Place Lab[17], Smart Meeting Room[8], and Smart Vehicles[18], provide intelligent and adaptive services for assisting users

*This work was partially supported by grants from the National Science Council, Taiwan (NSC 96-2628-E-002-173-MY3), Excellent Research Projects of National Taiwan University (97R0062-06), and Intel Corporation.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

AUPC'09, July 13–17, 2009, London, United Kingdom.
Copyright 2009 ACM 978-1-60558-647-2/09/07 ...\$10.00.

to concentrate on their specific tasks. In these smart space environments, contextual information can be sensed in richly equipped and networked environments. Moreover, users are not necessary to carry any extra device because services will be delivered to vicinity of them.

Context-awareness is the essential characteristic for building smart space environments. Software architecture for context-aware systems must have abilities to perceive the change of external environment or internal state, understand the current context, and provide services to right place at right time. Consequently, support of various devices deployment, manipulation of contextual information, and discovery of services, are basic considerations while design a context-aware system.

A multi-agent system is composed of multiple interacting intelligent agents. Each agent has several important attributes, includes the ability to perceive and react to the near environment, the possession to achieve individual goals, the facility to communicate with other agents, the capability to perform actions with some level of autonomy, and the capacity to provide services. These attributes make agent paradigms suitable for building context-aware applications. Different agent supports for different entities and each agent concerns itself with simple task, that is, some agents carry out contextual information from the vicinity of the user; some agents manipulate contexts; and some agents deliver services.

Many challenges have been encountered while building a context-aware system. We address three issues in our research. The first one is information representation. In a smart space, raw contexts are gathered from various devices, and consequently a context-aware system needs to understand the meaning of a context. Therefore, a model to represent contextual information is the first issue of developing context-aware systems. Second, high-level contexts, such as environmental situations and users' states, cannot directly be acquired from sensors. Accordingly, the capability to entail high-level contexts from the existing knowledge is required in context-aware systems. Third, inconsistent contexts may appear due to the rapid change of contextual information that will cause a context-aware system fail to deliver correct services. Therefore, a mechanism to maintain consistent knowledge bases is an import issue.

In this research, we leverage semantic web technologies that provide means to express context and use abstract representations to derive usable context for proactively delivering context-aware service to users. We propose an ontology-base model for supporting context management which can

provide high-level context reasoning and detect the knowledge inconsistency.

2. RELATED WORK

Gu *et al.*[15] propose CONON context model, which assigns different temporal characteristics to different contexts. For example, the sensed contextual information has its lifetime with persistence of seconds to hours. Chen *et al.*[9] introduce COBRA architecture that utilizes the predefined schedule to handle the context consistency problem. According to the scheduled time of a user, system assumes the user is stay in a specific place. Using an abductive reasoning mechanism, COBRA maintains its belief of user's location, even though the sensor perceives the movement of a user. However, the demonstration domains of these researches are limited and the rules are tightly defined the specific relationship among user's intentions, activities, and services.

In the past decade, the services offered to users are interested in context information related to the users' current position. Nowadays, the design principles in context-aware systems are mainly focus on addressing special requirements of isolated areas (*e.g.* smart homes[1], offices artifacts[23], *etc.*) in an ad-hoc manner. Meyer and Rakotonirainy[20] make an exhaustive survey on context-aware homes. Department of Architecture research group at Massachusetts Institute of Technology proposes the House_n¹ research, which includes a "living laboratory" residential home research facility called the PlaceLab[17]. Hundreds of sensing components are installed in nearly every part of the house. Interior conditions of the house can be captured by using these sensors, such as temperature, light, humidity, pressure, electrical current, water flow, and gas flow sensors. Eighty wired switches can detect the opening of the refrigerator, the shutting of the linen closet, or the lighting of a stovetop burner events. Cameras and microphones are embedded in the house for recording the resident's movement. Twenty computers collect all the data streams from these devices and sensors to provide multi-disciplinary research, for example, monitoring the resident's behavior, activity recognition, and dietary status.

3. TECHNOLOGY OVERVIEW

An overview of the context models, context reasoning, and ontology are introduced in this section.

3.1 Context Models

A context model is needed to define and store contextual information, which includes temporal relations, geo-spatial entities and relations, user profiles, personal schedule, and actions taken by the people. Develop a model to represent the wide range of contexts is a challenging task. Strang and Linnhoff-Popien[22] summarized the most influential context modeling approaches according to their data structures. Chen and Kotz[5] classified context models into four typical categories, which are key-value, markup scheme, object-oriented, and logic-based models.

Many context-aware systems concentrate on location aware services. Ye *et al.*[24] use lattice model to represent spatial structure, which can deal with syntactic and semantic labels. This general spatial model provides both absolute and

¹http://architecture.mit.edu/house_n/

relative references for geographic positions, both the containment and connection relationships can be determined as well. MINDSWAP Group geoStuff² to express basic geographic features such as countries, cities, and relationships between these spatial descriptors.

There are two basic temporal models, the point-based and interval-based time models. A time point can be considered as an instant of time. In contrast to that, a time interval is a temporal primitive with an extent. It can be specified by two time points or by a time point plus a duration. Traditional time structure is based on a set of points, Bry *et al.*[3] introduce CaTTs and treat the cultural calendars as interval-based time. Ma and Hayes[19] analyze the temporal interval-based models in a recent literature report. The OWL-Time³ and ISO 8601 define the time structure and standard.

The RFC 2445⁴ provides the definition of iCalendar for exchanging calendaring and scheduling information across the Internet. Google Calendar⁵ is a popular web-based calendar which supports iCalendar standard and users can share their own personal activities with others. These human activities are related to people, time, and location. Consequently, the contents of persons' schedules can help us to derive their location at a given time.

3.2 Ontology

The term ontology has its origin in philosophy, and has been applied in many different ways. The core meaning within computer science is a model for describing the world that consists of a set of types, properties, and relationship types. In the knowledge engineering model, ontology is applied to model declarative knowledge. Strang and Linnhoff-Popien[22] concluded that the ontologies are the most expressive models. Gruber[14] defines ontology as an "explicit specification of a conceptualization". Ontology is developed to capture the conceptual understanding of the domain in a generic way and provide a semantic basis for grounding the fine-grained knowledge. The COBRA-ONT[7] provides the key requirements for modeling context in a smart meeting application. It defines concepts and relations of physical locations, time, people, software agents, mobile devices, and meeting events. The SOUPA[10] (Standard Ontology for Ubiquitous and Pervasive Applications) is proposed for supporting pervasive computing applications. SOUPA uses some other standard domain ontologies, such as FOAF⁶ (Friend of A Friend), OpenGIS, the spatial relations in OpenCyc⁷, ISO 8601 date and time formats⁸, and time ontology[16].

3.3 Context reasoning

The processes of context reasoning can infer new contexts from the existing contexts. In a smart space, if systems are unable to reason and share context, the intelligence of context-aware systems will be limited and cannot meet

²<http://www.mindswap.org/2004/geo/geoStuff.shtml>

³<http://www.w3.org/TR/owl-time/>

⁴<http://tools.ietf.org/html/rfc2445>

⁵<http://calendar.google.com>

⁶<http://xmlns.com/foaf/spec/>

⁷<http://www.cyc.com/cycdoc/vocab/spatial-vocab.html>

⁸<http://www.w3.org/TR/NOTE-datettime>

users' requirements, which will cause the user abandon the system.

Design and implementation of context reasoning can vary depending on the types of contextual information involved. The early context-aware systems[4]are tightly coded the logics of context reasoning into the behavior of the systems. Implementation for understanding the contextual information is bound into the programs for guiding the context-aware behavior of the systems, therefore, the developed applications often have rigid implementations and are difficult to maintain.

Rule-based logical inference can help to develop flexible context-aware systems by separating high-level context reasoning from low-level system behaviors. Context modeling languages are used to represent contextual information and the rule languages are used to enable context reasoning. In most cases, these two types of languages have different syntax and semantic representations; it is a challenge to effectively integrates these distinctive languages to support context-aware systems. A mechanism to convert between contextual modeling and reasoning languages is one of the solutions for this challenge. Gandon and Sadeh[12, 13] propose e-Wallet that implements ontologies as context repositories and uses a rule engine Jess[11] to invoke the corresponding access control rules. The e-Wallet using RDF⁹ triples to represent contextual information and OWL¹⁰ to define context ontology. Contextual information is loaded into the e-Wallet by using a set of XSLT¹¹ stylesheets to translate OWL input files into Jess assertions and rules.

Ontology models can represent contextual information and specify concepts, subconcepts, relations, properties, and facts in a smart space. Moreover, ontologies reasoning can use these relations to infer the facts that are not explicitly stated in the knowledge base. Ranganathan *et al.*[21] propose that ontologies can make it easier to develop programs for reasoning about context. Chen[6] proposes that the OWL language can provide a uniformed solution for context representation and reasoning, knowledge sharing, and meta-language definitions. Anagnostopoulos *et al.*[2] adopt the Description Logic for expressing and reasoning contextual knowledge. The OWL DL was designed to support the existing Description Logic business segment and has desirable computational properties for reasoning systems. Typical ontology-based context-aware application is EasyMeeting that uses OWL to define the SOUPA ontology and OWL DL to support context reasoning. Gu *et al.*[15] propose an OWL encoded context ontology CONON in Service Orientated Context Aware Middleware (SOCAM). CONON consists two layers of ontologies, an upper ontology that focuses on capturing general concepts and a domain specific ontology. EasyMeeting and SOCAM both use an OWL DL reasoning engine to check the consistency of contextual information and provide further reasoning over low-level context to derive high-level context.

4. AGENT-BASED ARCHITECTURE

Figure 1 shows the sequence diagram of our agent-based architecture for supporting context-aware service. In a smart space environment, equipped devices and applications can

⁹<http://www.w3.org/TR/rdf-concepts/>

¹⁰<http://www.w3.org/TR/owl-features/>

¹¹<http://www.w3.org/TR/xslt>

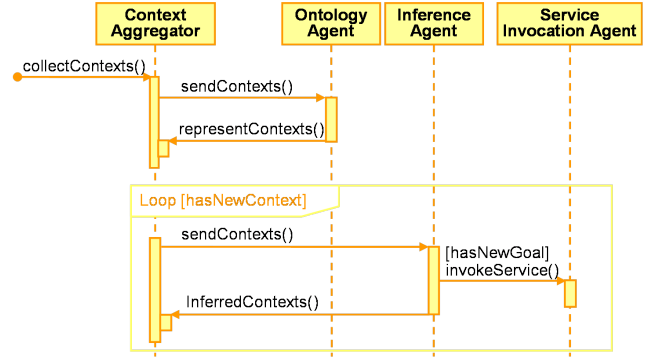


Figure 1: Sequence Flow Diagram

provide contextual information and deliver context-aware services. Context Aggregator collects raw sensing data from these context sources of the smart space. According to the definition of context ontology, Ontology Agent interprets the context into a semantic representation. Inference Agent will continuously infer high-level contexts and may derive the service goal for the user.

5. CONTEXT PROCESSING

A context-aware system must have the capability to precede continuously changing contexts and proactively provides services to the user. Figure 2 shows procedures of processing contextual information. The detailed procedures

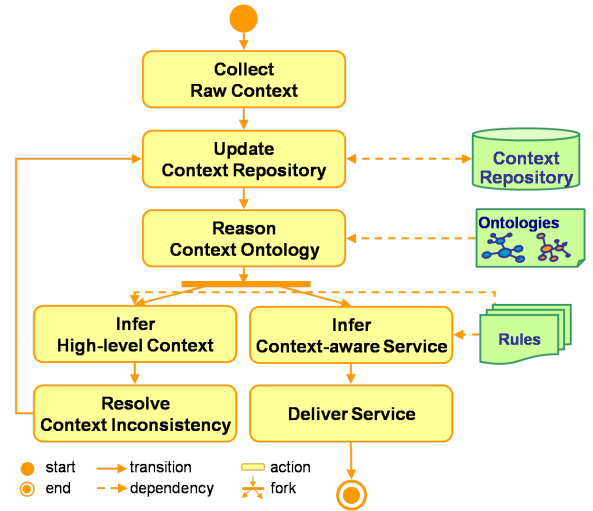


Figure 2: Context Activity Diagram

of every action are described as follows:

(1) **Collect Raw Context**

Context Aggregator captures raw sensing data from context sources in the smart space environment. A context source can be either provided by hardware or software, such as RFID reader, GPS, RSS feed, personal calendar, weather forecasts, food preference, *etc.*

(2) **Update Context Repository**

Contexts are stored in Context Repository and will

be dynamically updated for reflecting changes of contexts. There are two types of context: the raw context and the high-level context. A raw context refers to the sensing data that directly obtained from a context source. High-level contexts are deriving from a rule-based approach, which matches facts and contexts with horn clause rules.

(3) Reason Context Ontology

Ontology Agent loads the OWL context ontology and parses the contents into RDF triples, which makes other agents able to represent and share context in the system. Context ontology defines structures and relationships between contexts. These relationships can support the ontology reasoner to deduce high-level contexts. Moreover, utilizing the ontology structures, a subsumption reasoning approach is applied for derive the hierarchical concept of a raw context. For example, to deduce the superclasses of a specified class and to decide whether one class subsumes another, *e.g.*, a building may spatially subsume a room.

(4) Fork actions

Two operations of rule-based inference are occurring at the same time. Both of them use the rules to infer results. One is for deducing high-level contexts and the other one is for deriving context-aware service.

(a) Infer High-level Context

Rules for a rule-based system serve as IF-THEN statements. For example, a rule for detecting the user's location can be shown as follows:

```
[Person_Location:
  (?cellphone isBelongTo ?person)
  (?cellphone isMovingTo ?place)
->
  (?person isLocatedIn ?place)
]
```

Patterns before `->` are conditions matching by a specific rule, called left hand side (LHS) of the rule. On the other hand, patterns after `->` are statements that may be fired, and it is called right hand side (RHS) of the rule. If all the LHS conditions are matched, patterns in RHS will be asserted.

The rule `Person_Location` can infer the user's location. If a person `?person` has a cell phone `?cellphone` and the mobile phone tracking system can transmit its location `?place`, we can deduce that `?person` is located in `?place`.

(b) Infer Context-aware Service

RHS statements can also be the goal of a service request.

```
[Find_Nearby_Restaurant:
  (?person isLocatedIn ?place)
  (?person like ?food)
  (?restaurant styleOfFood ?food)
  (?restaurant hasAddress ?address)
  (?place isNearby ?address)
->
```

```
(?restaurant isRecommendedFor ?person) ]
```

Rule `Find_Nearby_Restaurant` can deduce nearby restaurants for a food recommend system. For example, if Henry is located in downtown and he likes Chinese food, this rule will find all the nearby Chinese food restaurants.

(5) Resolve Context Inconsistency

We must ensure that incorrect or outdated contexts are not existed in Context Repository. If a raw context is changed, some of inferred high-level contexts may also be changed. For example, if Henry moves from the library to a restaurant, high-level context asserts by rule `Person_Location` will be changed from `(Henry isLocatedIn library)` to `(Henry isLocatedIn restaurant)`. For one given time, a person is appeared at one location. Therefore, we define the property of a person's location `isLocatedIn` is one-to-one relationship; that is, the system will replace the location context of Peter. Therefore, we need to maintain consistency of current contexts so that the system can deduce correct high-level contexts and derive the right service for the user.

Because high-level contexts are deduced from raw contexts, if we keep a consistent raw contextual information, we can also maintain the consistency of high-level contexts and achieve the context consistency. We use RDF-triple to represent contextual information and utilize the property of predicate to decide whether the raw context should be kept or not. If the predicate of a raw context is a functional property, the new one will replace the original context. Otherwise, the new raw context will be added into Context Repository. An OWL DL reasoner performs the consistency checking, which ensures that ontology does not contain any contradictory facts. If there exist inconsistency, this means some outdated raw contexts existed and they should be removed.

(6) Deliver Service

When a service request has been inferred, the corresponding service will be triggered.

6. CONTEXT MODEL

Context-aware applications need a unified context model that is flexible, extendible, and expressive to adapt the variety of context features and dependency relations. The ontology models can fulfill these requirements. Therefore, we deploy an ontology context model to represent contextual information in smart space environment.

6.1 Context Repository

Context Repository stores a set of consistent context, which including location, person, and activity information. Classes of contexts and relationships between instances of context objects are defined by context ontology. A RDF-triple contains a subject, a predicate, and an object. The subject is a resource named by a URI with an optional anchor identity. The predicate is a property of a resource. The object is the value of a property for a resource. For example, the following triple represents "Peter is sleeping".

```

<http://...#Henry>
<http://...#isLocatedIn>
<http://...#restaurant>

```

Where **Henry** represents a subject, **isLocatedIn** is a predicate, and the location **restaurant** is an object. We use subject and predicate as the compound key of Context Repository. When a context has been updated, the associated timestamp will be changed accordingly.

6.2 Context Life Cycle

In a smart space environment, many contexts may appear at the same time and some contexts may suspend or disappear according to the environment change. Therefore, it is important to manage the life cycle of contextual information to provide timely and accurate contexts. The life cycle of a context can help context-aware systems to maintain context consistency. Moreover, it can be used for providing context-aware services.

The contextual information may appear and vanish at anytime. The life of a context can be either a time point or a time interval. For example, “Henry enters a restaurant at 12:00PM” is a time point event. On the contrary, some contexts can be created and endure for a time period, such as “Henry is eating”. We define every context has its life cycle which is the duration of validity.

Figure 3 is the diagram of the context state transition. We classify states of a context into three states: Active,

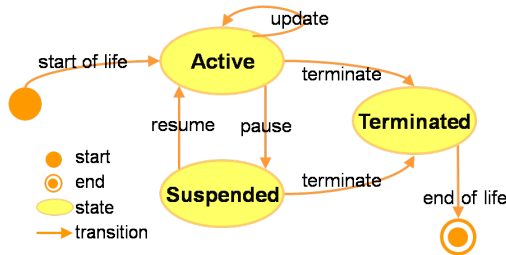


Figure 3: Context State Transition

Suspended, and Terminated.

Active.

Every context is starting from the entry point to the active state. A context becomes active as soon as the system receives new contextual information. For example, when Henry is eating in a restaurant, “Henry is eating” and “Henry is in a restaurant” are the contexts in active state. A context remains in active state when the user is still performing the same activity. Therefore, it only needs to update the duration of context. When the activity is stop, the state of context transfers from active to terminated.

Suspended.

If the activity that the user currently performed has been disturbed, the state of the context will change to suspended. For example, when Henry is eating and the phone is ringing, a new context of “Henry answers the phone” is active and the context “Henry is eating” changes to suspended state. If the previous activity has been resumed, the corresponding context changes from suspended to active state.

Terminated.

A context is in a terminated state when it is not valid anymore. When an activity has been terminated, the corresponding context changes its state to ‘terminated’ as well.

7. CONTEXT MANAGEMENT

Changes of environmental contexts are transient in the sense of that any context may appear and vanish at anytime. Algorithm 1 shows how to manage the contextual information in Context Repository.

Algorithm 1 Maintaining Context Repository

```

1: Input:  $c$  is the new context
2:  $C$ : Context Repository
3:  $rdfi$ : RDF-triple  $(s_i, p_i, o_i)$  of a context  $i$ 
4:  $key_i$ : key of context  $i$  in Context Repository
5: for all  $i \in C$  do
6:   if  $isOutdated(i)$  then
7:      $delete(i)$ 
8:   end if
9: end for
10: if  $\exists i \in C$  s.t.  $key_i = key_c$  and  $isFunction(p_i)$  then
11:    $update(key_c, c)$ 
12: else
13:    $insert(key_c, c)$ 
14: end if

```

We use RDF-triple $rdfi$ to represent a context i . The (s_i, p_i, o_i) represents subject s_i , predicate p_i , and object o_i . In Context Repository, the key key_i of a context i comprises the subject s_i and predicate p_i . When a new context c is arrival, we use the key key_c to query Context Repository. Given a context i , the property of the predicate p_i is used to decide whether the new context should be kept or not. Consequently, if the predicate p_i is a functional property, function $isFunction(p_i)$ returns *true*, otherwise, returns *false*. If a context i in Context Repository and its associated predicate p_i represents one-to-one relationship, the new context will replace the old one. Otherwise, the new context will be inserted into Context Repository. Functions $update(key_c, c)$ and $insert(key_c, c)$ perform the context replacement and insertion, respectively. When a context i is vanished, it should be removed. function $delete(i)$ can remove the specified context from Context Repository. We use a decay function to determine the existence of a context. Different context is associated with a different decay function. This function can either be an objective function for predicating a specified activity or simply be a constant function. The function $isOutdated(i)$ applies the context decay function to decide whether the context is existed or not.

8. CONCLUSION AND FUTURE WORK

This research presents a context management mechanism in a smart space. We integrate context-aware technologies, semantic web, and logical reasoning to provide context-aware services. An ontology-based model provides context reasoning, which can deduce high-level contexts and detect context consistency. Each contextual information is associated with a state of life cycle for maintaining the context consistency and supporting the context-aware services.

9. REFERENCES

- [1] G. D. Abowd, C. G. Atkeson, A. F. Bobick, I. A. Essa, B. MacIntyre, E. D. Mynatt, and T. E. Starner. Living laboratories: the future computing environments group at the georgia institute of technology. In *Proceedings of Conference on Human Factors in Computing Systems (CHI '00): extended abstracts on Human factors in computing systems*, pages 215–216, New York, NY, USA, 2000. ACM Press.
- [2] C. B. Anagnostopoulos, A. Tsounis, and S. Hadjiefthymiades. Context awareness in mobile computing environments. *Wireless Personal Communications: An International Journal*, 42(3):445–464, 2007.
- [3] F. Bry, F.-A. Ries, and S. Spranger. CaTTS: calendar types and constraints for web applications. In *Proceedings of the 14th international conference on World Wide Web (WWW '05)*, pages 702–711, Chiba, Japan, 2005. ACM Press.
- [4] L. Capra, W. Emmerich, and C. Mascolo. CARISMA: Context-aware reflective mIddleware system for mobile applications. *IEEE Transactions on Software Engineering*, 29(10):929 – 945, 2003.
- [5] G. Chen and D. Kotz. A survey of context-aware mobile computing research. Technical Report TR2000-381, Dartmouth College, Hanover, NH, USA, 2000.
- [6] H. Chen. *An Intelligent Broker Architecture for Pervasive Context-Aware Systems*. PhD thesis, University of Maryland, Baltimore County, 2004.
- [7] H. Chen, T. Finin, and A. Joshi. An ontology for context-aware pervasive computing environments. *The Knowledge Engineering Review*, 18(3):197–207, September 2003.
- [8] H. Chen, T. Finin, A. Joshi, L. Kagal, F. Perich, and D. Chakraborty. Intelligent agents meet the semantic web in smart spaces. *IEEE Internet Computing*, 8(6):69–79, November – December 2004.
- [9] H. Chen, F. Perich, D. Chakraborty, T. Finin, and A. Joshi. Intelligent agents meet semantic web in a smart meeting room. In *Proceedings of the Third International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS '04)*, volume 2, pages 854–861, Los Alamitos, CA, USA, 2004. IEEE Computer Society.
- [10] H. Chen, F. Perich, T. Finin, and A. Joshi. SOUPA: Standard ontology for ubiquitous and pervasive applications. In *The First Annual International Conference on Mobile and Ubiquitous Systems: Networking and Services (MobiQuitous'04)*, pages 258–267, August 2004.
- [11] E. Friedman-Hill. *Jess in Action: Java Rule-Based Systems*. Manning Publications, Greenwich, CT, USA, 2003.
- [12] F. L. Gandon and N. M. Sadeh. A semantic e-wallet to reconcile privacy and context awareness. *Lecture Notes in Computer Science: The Semantic Web (ISWC 2003)*, 2870:385–401, October 2003.
- [13] F. L. Gandon and N. M. Sadeh. Semantic web technologies to reconcile privacy and context awareness. *Journal of Web Semantics*, 1(3):241–260, 2004.
- [14] T. R. Gruber. A translation approach to portable ontology specifications. *Knowledge Acquisition*, 5(2):199–220, June 1993. Special issue: Current issues in knowledge modeling.
- [15] T. Gu, H. K. Pung, and D. Q. Zhang. A service-oriented middleware for building context-aware services. *Journal of Network and Computer Applications*, 28(1):1–18, 2005.
- [16] J. R. Hobbs and F. Pan. An ontology of time for the semantic web. *ACM Transactions on Asian Language Information Processing (TALIP)*, 3(1):66–85, 2004. Special Issue on Temporal Information Processing.
- [17] S. S. Intille. Designing a home of the future. *IEEE Pervasive Computing*, 1(2):76–82, April 2002.
- [18] G. Look and H. Shrobe. A plan-based mission control center for autonomous vehicles. In *IUI '04: Proceedings of the 9th international conference on Intelligent user interfaces*, pages 277–279, New York, NY, USA, 2004. ACM Press.
- [19] J. Ma and P. Hayes. Primitive intervals versus point-based intervals: Rivals or allies? *The Computer Journal*, 49(1):32–41, 2006.
- [20] S. Meyer and A. Rakotonirainy. A survey of research on context-aware homes. In *Proceedings of the Australasian information security workshop conference on ACSW frontiers 2003*, pages 159–168, Darlinghurst, Australia, Australia, 2003. Australian Computer Society, Inc.
- [21] A. Ranganathan, J. Al-Muhtadi, and R. H. Campbell. Reasoning about uncertain contexts in pervasive computing environments. *IEEE Pervasive Computing*, 3(2):62–70, 2004.
- [22] T. Strang and C. Linnhoff-popien. A context modeling survey. In *Workshop on Advanced Context Modelling, Reasoning and Management at The Sixth International Conference on Ubiquitous Computing (UbiComp 2004)*, Nottingham, England, 2004.
- [23] H. Yan and T. Selker. Context-aware office assistant. In *Proceedings of the 5th international conference on Intelligent user interfaces (IUI '00)*, pages 276–279, New York, NY, USA, 2000. ACM.
- [24] J. Ye, L. Coyle, S. Dobson, and P. Nixon. A unified semantics space model. In J. Hightower, B. Schiele, and T. Strang, editors, *Proceedings of the 3rd International Symposium on location- and Context-Awareness (LoCA 2007)*, volume 4718 of *Lecture Notes in Computer Science*, pages 103–120, September 2007.