Ontonym: A Collection of Upper Ontologies for Developing Pervasive Systems

Graeme Stevenson
Systems Research Group
School of Computer Science
and Informatics
UCD Dublin, Ireland
graeme.stevenson@ucd.ie

Stephen Knox
Systems Research Group
School of Computer Science
and Informatics
UCD Dublin, Ireland
stephen.knox@ucd.ie

Simon Dobson
Systems Research Group
School of Computer Science
and Informatics
UCD Dublin, Ireland
simon.dobson@ucd.ie

Paddy Nixon
Systems Research Group
School of Computer Science
and Informatics
UCD Dublin, Ireland
paddy.nixon@ucd.ie

ABSTRACT

Pervasive systems present the need to interpret large quantities of data from many sources. Context models support developers working with such data by providing a shared representation of the environment on which to base this interpretation. This paper presents a set of requirements for a context model that addresses *uncertainty*, *provenance*, *sensing* and *temporal properties* of context. Based on these requirements, we describe *Ontonym*, a set of ontologies that represent core concepts in pervasive computing. We propose a framework for evaluating ontologies in the pervasive computing domain by combining recognised techniques from the literature, and present a preliminary evaluation of Ontonym using these criteria.

1. INTRODUCTION

Pervasive computing is an evolution of the desktop computing paradigm, whereby almost any object, from home furnishings and appliances, to cars, to clothing, even to coffee mugs and credit cards can be embedded with sensing and processing capabilities [1]. Through networks of these devices, information about people and their surrounding environment is combined and used to provide personalised services across application domains as diverse as assisted living, environmental monitoring, and ambient information systems.

The information relevant to a particular system or application is referred to as its *context*. Dourish describes context as a “slippery notion” [2]; it is highly domain specific, and has given rise to definitions too restrictive to obtain consensus and too broad to be meaningful. While the lack of an accepted definition precludes the specification of a “complete” model of context (that is, a symbolic model of the environment against which developers can write software), an extensible model that describes key pervasive computing concepts provides a basis for sharing data and supporting interactions between systems.

In this paper we review previous work in the area of context modelling and propose an extended set of requirements for context modelling in pervasive environments including: *uncertainty*, *provenance*, *sensing* and *temporal properties* of information (section 2). To our knowledge, no single model currently addresses all these concepts. We then describe *Ontonym*, a set of *upper ontologies* for pervasive computing (section 3). In the spirit of Baumgartner and Retschitzegger [3], we regard upper ontologies as describing high-level, but domain specific (i.e., pervasive computing) concepts, as opposed to *top-level ontologies*, which model the fundaments of the world. Ontonym provides semantic interoperability between pervasive systems by describing core pervasive computing concepts from the literature. These concepts act as point from which application-specific concepts can extend. From analysis of the literature we propose a strategy for evaluating ontologies that falls into three categories: *design principles*, *content* and *evaluation for purpose* (section 4), and conclude by presenting a partial evaluation of Ontonym from the criteria identified (section 5).

2. MODELLING CONTEXT

Strang and Linnhoff-Popien [4] and Henricksen et al. [5] set out several requirements for a context model, covering technical requirements of the modelling technique, capturing the quality of data, and supporting the representation of past and future states. Building on this work, we present an updated set of requirements for modelling context. In addition to capturing recognised properties of sensed data we add the requirement for modelling properties and capabilities of sensors. We further define the set of temporal properties that a context model should capture, add the need to model the provenance of data, and identify support for meta-modelling as way of capturing orthogonal concerns.

2.1 Uncertainty

Both Strang and Linnhoff-Popien and Henricksen et al. identify the need to model *incomplete*, *ambiguous*, and *imprecise* information. We put these under the umbrella heading of uncertainty, using the definitions of Henricksen et al. [6], as they relate to the value of an attribute belonging to an entity. Incompleteness occurs when not all information about the attribute is known; ambiguity when two or more data sources provide contradictory information about the attribute; and imprecision when the attribute's value is an approximation of the real world. The nature of imprecision varies between data types, but is usually characterised using quality parameters. For example, location data may be characterised by its granularity, precision, and accuracy.

2.2 Provenance

Producers and consumers of data are often separated by a middleware layer that transforms raw data into a form better matching application requirements by removing noise and applying fusion algorithms. *Provenance* is concerned with being able to determine the origin of a piece of data by recording from where data was sourced, and the role played by intermediate components in its derivation [7]. For any value in the data model, it should be possible to trace the complete transformation history of the data from sink to source. Provenance plays a roll in error detection, debugging, and in managing uncertainty.

2.3 Sensing

As information in pervasive computing systems is largely discovered through sensors observing the environment, modelling *data sources* is as important as modelling data itself. The representation of a sensor can be used in tracking provenance, and, where aspects of readings or its meta-data are constant, these properties can be modelled as part of the sensor rather than repeated across each of its readings. From an implementation perspective, it is not always sufficient to know if something has been sensed, but if something can be sensed (e.g., if the presence of bluetooth devices will be detected within a room). Capturing *sensor capability* allows us in many cases to determine if an application's needs can be met by a particular environment.

2.4 Temporal Properties

There are four temporal properties that affect the interpretation of context, and may inform the design of the system that manages and provides access to it: *dynamism*, *temporal dimension*, *observation time*, and *sampling period*. Henricksen et al. [8] note that the rate at which dynamic data changes can be highly variable – relationships between colleagues may endure for years, while a person's location can change momentarily. The temporal dimension of data is concerned with whether data describes past or planned state (e.g., a prior event or a scheduled meeting), while observation time records when the data was sampled. Finally, sampling period describes the frequency of a sensor's observations.

2.5 Modelling Capabilities

We adopt three of Strang and Linnhoff-Popien's requirements directly: *distributed composition* states that no central authority should be responsible for the administration of a context model and its data; *partial validation* requires that it is possible to validate the structure and content of knowledge against a model, even if complete knowledge is not available in a single place at the same time; and a *level of formality* ensures that each participant in an interaction shares the same interpretation of the data exchanged.

Although some consider additional requirements to be core to pervasive systems, such as ownership of information [9] and belief of agents [10], we treat these as orthogonal concerns in that they are not particular to context models and can be applied equally well to any knowledge model. We therefore argue that such requirements can be supported by using a modelling technique that supports *meta-modelling*, i.e., the ability to annotate statements in the model.

2.6 Summary

Building on existing work, we have presented an updated set of requirements for a context model that addresses uncertainty, provenance, sensing, temporal properties of context, and modelling capabilities. Given these requirements, we now introduce our work, the Ontonym ontologies.

3. THE ONTONYM ONTOLOGIES

Although context is an application specific notion, context-aware applications exhibit overlapping data requirements; the most common being the need to represent time, location, people (or identities), and events (or activities) [11]. Temporal relationships support the modelling of historic, current, and predicted states; location has several representations, each suited to particular classes of application; people are the actors in pervasive systems; and events frequently model the actions of people in a particular location at a particular time. The ubiquity of these concepts in the literature suggests their role in forming upper ontologies in a context model, from which application specific contexts can extend.

In this section we describe the Ontonym ontologies for pervasive computing, which combines these concepts with a model for provenance, and for sensors and sensed data to meet our modelling requirements. We compare Ontonym with existing ontologies for pervasive computing in section 5.

3.1 Time

There exist well established models for representing and working with time. The ISO 8601 standard for date and time representations covers lexical representations of Gregorian dates, time of day, and time intervals – a subset of which is adopted by XML Schema. Allen [12] proposed the theory that the world can be described by a set of temporally qualified assertions outlining what is known about the past, present, and future, and proposed an interval-based temporal logic to support this. Allen defines seven relationships between intervals (during, starts, finishes, before, after, meets, and equals) and their inverses for a total of 13 (equals has no inverse). Based on Allen's model, the W3C OWL-Time ontology [1] provides a comprehensive vocabulary

[1] OWL-Time: www.w3.org/TR/owl-time

for expressing facts about topological relations among instants and intervals, together with information about durations and date-time information (including support for XML Schema representations). Rather than reinvent the wheel, we use OWL-Time to represent all temporal information in Ontonym. In addition to being a standard, many technologies support working with XML Schema date-time representations (e.g., the SPARQL query language).

3.2 Location

The data generated by positioning systems can be categorised as *physical* or *symbolic* [13]. Physical positions, such as those generated by the Global Positioning System (GPS), take the form of a 2D or 3D numeric array. Symbolic positions, provided by systems like the Active Badge [14], describe locations using human-friendly descriptive names, such as "Coffee Area", that may be organised into a hierarchy of granularities (e.g., coordinate, room, and building). Jiang et al. fused both these models into a single representation by representing spaces symbolically and describing the geometry for each symbol with respect to a co-ordinate system [15]. Later, Ye et al. improved on this model, introducing two forms of *relative representation* that can be used to describe geometric boundaries relative to a location, or the relative position of a location from another by means of distance and standard compass directions [16].

Ontonym's location ontology is an implementation of Ye's model. Spaces are represented using a combination of *SymbolicRepresentation*, *GeometricRegion*, and *RelativeLocation* ontology classes. The model defines four types of spatial relationship: *containment*, *adjacency* and *overlap* are as per their suggested meaning, while *connectedness* is a particular case of adjacency, where it is possible to pass from a space to its adjacent space. Coordinates are defined with reference to a coordinate reference system (CRS), and translations from one CRS to another can be specified using the scheme described by Jiang et al. [15]. Listing 1 gives a typical example of a space description.

3.3 People

Excluding application-specific data, the types of information about a person used in pervasive systems can be broken down into four categories: identity, device ownership, personal details, and social relationships. Identity allows applications and services to differentiate one user from another. Device ownership is important as devices often act as a proxy for a user [17], for example, a keycard for authentication or a locatable tag[2] as an indicator of a user's location. Personal details include name and contact information, while social relationships between people are increasingly useful to applications.

OWL's URIs provide the notion of identity. We adopt the Friend-of-a-Friend (FOAF)[3] convention of using the MD5 hash[4] of a person's email address[5] in addition to any "human readable" URIs assigned. This provides users with a means of easily identifying themselves to applications, who can then hash the email address to index into the model.

Listing 1 A description in Notation3 of a cubicle and its relationships to other spaces. The entity denoted *map:bobsCubicle* is defined as having both symbolic and geometric components and specifies containment, adjacency, and connected relationships to other locations. The space's geometric region is defined as a cuboid whose bounds are specified by two coordinates (omitted for brevity, but expressed similarly to that in listing 4).

```
map:bobsCubicle
  a location:GeometricRegion,
    location:SymbolicRepresentation ;
  location:locationName "Bob's Cubicle" ;
  location:adjacent map:alicesCubicle,
                    map:corridor3f ;
  location:connectedTo map:corridor3f ;
  location:containedBy map:researchCubicles ;
  location:contains map:bobsDesk ;
  location:hasGeometricRegion
    [ a  location:Cuboid ;
      location:hasCoordinates [...], [...]
    ] ;
  location:hasGranularity map:cubicle .
```

We borrow terms from vCard[6], W3C PIM[7] and FOAF in our model including: date of birth, gender, language, and contact profiles (default, home and work) containing postal and email addresses, telephone and fax numbers, and web presence, formalising each as required (e.g., using OWL-Time for date of birth and the IANA language subtag registry for language preferences). We omit some terms, e.g., organisational roles, instant messaging usernames, documents, images, and project groups as being broad enough in scope as to warrant separate ontologies.

The modelling of names presents an interesting problem. Current approaches adopt the {*firstname, lastname*} approach to name modelling, which applies only to a small proportion of the world's population. In Ontonym's person ontology, components of a name are specified using classes that denote their origin (e.g., *GivenName*, *ReligiousName*, *PatronymicName*, and *ProfessionalTitle*) with names structured as a list of components[8]. We provide three name related predicates: *name* - the standard representation of a person's name, *completeName* - the complete form of a person's name, and *shortName* - a single name component used as an informal greeting.

The same problem arrises with the representation of postal addresses. The current version of Ontonym adopts the familiar street, region, city, country, postal code approach, which we recognise as inadequate. In the future we hope to adopt a standard arising from ongoing research [18].

We adopt FOAF's *knows* property, used to define relation-

[2] e.g., the Ubisense positioning system: `www.ubisense.net`

[3] The FOAF ontology: `www.foaf-project.org`

[4] The MD5 hash function - Internet standard RFC 1321

[5] This is not ideal, as a user's email addresses may change throughout their lifetime.

[6] vCard ontology: `www.w3.org/2006/vcard`

[7] W3C PIM: `www.w3.org/2000/10/swap/pim/contact`

[8] Based on proposed name representations for FOAF: `wiki.foaf-project.org/NamesInFoaf` (accessed 15th June 2008)

ships between people, keeping the requirement of "reciprocated interaction" (which can be modelled using symmetric or inverse properties), and borrow from Davies and Vitiello's work[9] to define properties, covering genetic, working, romantic, residential, and friendship relations. Listing 2 shows a description of a personal profile document.

Listing 2 A portion of a personal profile document containing some personal details. Device ownership is specified using the *owns* property, while a couple of personal relationships, *worksWith* and *husbandOf*, are also illustrated.

```
example:bob
  a person:Person ;
  owl:sameAs
    personMD5:41ec56ef7cbadea324acff8d2c1efa42 ;
  person:gender person:male ;
  person:shortName
    [ a person:GivenName ;
      rdfs:label "Bob"
    ] ;
  person:dateOfBirth
    [ a time:DateTimeInterval ;
      time:hasDateTimeDescription
        [ a time:DateTimeDescription ;
          time:day "23" ;
          time:month "2" ;
          time:unitType time:unitDay ;
          time:year "1981"
        ]
    ] ;
  device:owns example:bobsPhone ;
  person:workProfile
    example:bobsProfile ;
  person:husbandOf example:sarah ;
  person:worksWith example:alice .
```

3.4 Sensing

Ontonym's sensing ontology is concerned with the description of sensors and the data they generate. We represent the characteristics and uncertainty of sensed data by using a quality matrix. The core quality matrix defines four properties: *frequency*, *coverage*, and a set of *accuracy* and *precision* pairs. Frequency is defined as the sample rate – how often the sensed data is updated. Coverage is the amount of the potentially sensed context about which information is delivered. Precision defines a value range and accuracy is the percentage of how often the precision is achieved. Precision and accuracy can have different semantics depending on the sensor under consideration. For example, a temperature sensor might be specified as 99.9% accurate within 0.05 degrees, while a positioning sensors precision might be based on distance from the true location. e.g., 60% accuracy within 2 metres and 80% accuracy within 5 metres. Sensors may specify as many precision and accuracy pairs as required. If precision and accuracy vary with respect to the value then the sensor readings, instead of the sensor, are annotated.

The quality matrix associated with each sensor is referenced by all data inserted by that sensor into the model. Each sensor reading consists of observation-specific information, meta-data characterising the observation, a reference to the sensor that generated the reading (*observedBy*), a timestamp indicating when the observation was made (*observedAt*), *temporalDimension* properties to indicate the instants or time periods over which the value is considered "true", and a *rateOfChange* property that characterises the dynamism of the observed value. The data associated with a reading is considered to be the union of the observation and sensor properties, with the observation properties taking precedence. This allows static properties of an observation to be specified as part of a sensor's properties. For example, the position generated by a stationary bluetooth spotter is described by a radius centred on a fixed point.

The general quality matrix we describe models sensors at a high level and should be extended to characterise each type of sensor being used. Listing 3 describes a Ubisense sensor, while listing 4 shows a sample observation it might generate. Distances are described using the Measurement Units Ontology[10], and an OWL representation of the Unified Unified Code for Units of Measure[11].

Listing 3 A partial description of a Ubisense sensor. The *coverage* property defines the region of space over which the sensor operates and the *frequency* property defines the period between observations. The resolution of location data produced by the sensor is described by the *granularity* property, while *precisionAccuracy* pairs further characterise the quality of its observation.

```
example:CASLUbisense
  a sensor:Sensor ;
  sensor:coverage map:3f , map:4f ;
  sensor:frequency [...] ;
  sensor:granularity map:coordinateGranularity ;
  sensor:precisionAccuracy
    [ a sensor:PrecisionAccuracy ;
      sensor:accuracy "0.6" ;
      sensor:precision
        [ a muo:QualityValue ;
          muo:measuredIn ucum:meter ;
          muo:numericalValue "2"
        ]
    ] ;
  sensor:precisionAccuracy [...] ;
  sensor:rateOfChange [...] .
```

3.5 Provenance

The provenance ontology in Ontonym models three things: the creator or author of data, the time at which data is created or modified, and the data from which new data is derived. Creation records are catered for using the *createdAt* and *createdBy* properties, while modifications are recorded using an instance of a *ModificationRecord* class with associated *modifiedAt* and *modifiedBy* properties indicating the time of and entities responsible for each modification. Specification of derivation is supported using three properties:

[9] Davies and Vitiello's relationship vocabulary for FOAF: `vocab.org/relationship`

[10] The MUO ontology: `idi.fundacionctic.org/muo`

[11] The UCUM ontology: `unitsofmeasure.org`

Listing 4 An example of a partial reading produced by the Ubisense sensor described in listing 3. The *about* property relates the reading to a Ubitag, the *observedBy* property identifies the originating sensor, the *observedAt* and *temporalDimension* properties indicate the time of the reading and the time period to when the value applies (detail omitted), and the *value* property indicates the position at which the entity is observed – in this case, a 3D coordinate in a local coordinate system.

```
example:reading
  a sensor:Observation ;
  example:about ubitag:010131789 ;
  sensor:observedAt [...] ;
  sensor:temporalDimension [...] ;
  sensor:observedBy example:CASLUbisense ;
  sensor:value
    [ a location:Coordinate ;
      location:referenceCoordinateSystem
        example:ubisenseCoordinateSystem ;
        location:x "1.15" ;
        location:y "3.67" ;
        location:z "21.35"
    ] .
```

derivedFrom indicates data sources, *derivedBy* indicates the entity responsible for the derivation, and *derivationMethod* is used to specify the process by which the data was derived, which is application specific.

3.6 Events

The event ontology provides a means of describing activities that have (at least) a temporal dimension. These are implemented using the concepts of an *InstantEvent* and *IntervalEvent*, along with temporal predicates *atTime* and *timeSpan* as described by Hobbs and Pan [19]. A further two classes *SpatioInstantEvent* and *SpatioIntervalEvent* represent temporal events with an associated location. The event ontology also defines the *Role* class and *containsRole* property to identify roles that are played by entities in the activity, and a generic *playsRole* property to associate an entity (person, device, etc.) with an event. All of these concepts are designed for extension to describe domain-specific events. For example, a meeting and its participants, or a conference and its delegates.

3.7 Minor Ontologies

Ontonym contains a couple of additional minor ontologies. The *device* ontology serves, at the upper level at least, as a way of specifying the identifiers of devices that act as a proxy for a person. Similar to how a person's URI is derived from their email address, we take the approach of describing a device's URI using its (quasi-) uniquely identifiable components (e.g., a MAC address). Based on the Dublin Core[12] meta-data element set, the *resource* ontology provides a container for describing resources including their file format, language, web location, and associated rights (e.g., intellectual property).

4. PRINCIPLES OF ONTOLOGY EVALUATION

Representing knowledge is a highly subjective process – what is fit for one purpose may not be for another. Analysis of approaches to ontology evaluation in the literature reveals that no-single approach is comprehensive. In summarising best practice from across the literature, we propose an evaluation for ontologies in the pervasive computing domain covering three aspects: design principals, content, and evaluation for purpose.

4.1 Design Principals

The literature provides ontology engineers with five core design principals [20–23]. *Extensibility* requires that new classes and properties should be easily integrated with an ontology to meet the requirements of a given application domain, with minimal modification. Ontologies should be defined with *orthogonality* in mind; the concepts defined in any two given ontologies should be loosely coupled to the greatest extent possible, in order to ease maintenance and support reuse. In practice, this involves making trade-offs between the number of ontologies, and the coverage of each. Ontology engineers should avoid *encoding bias* – the use of representations purely for the convenience of an application. The adoption of standards should be preferred, so as to ease the integration of models and increase the likelihood of available tool support. Engineers should adopt and adhere to *naming conventions* and, while perhaps self-evident, ontologies should be *validated* against the rules of their implementation language. Most important is the inclusion of comprehensive *documentation*, without which ontologies are inaccessible to those who might otherwise use it. In addition to general information about the ontology, all classes and properties should be accompanied by a natural language description, the role of auxiliary ontologies and data formats should be described, and tutorials giving examples of correct use should be provided.

4.2 Content

The literature also provides guidelines on content [20,23–26]. *Clarity* relates to the specification of concepts in an ontology and requires that for each concept, all necessary and sufficient conditions that distinguish it from another concept are represented. *Consistency* requires that terms map to their real-world understood meanings, while *syntactic correctness* ensures that no incorrect terms or documentation appear. In the case of an upper ontology, *completeness* requires that an ontology should make just enough claims on a domain so it remains general enough to be usable in any application. Classes, associated properties, and constraints should serve for all general problems in the domain. Finally, *conciseness* is the don't-repeat-yourself principle of ontology engineering; existing terms should be reused where possible and redundancy should be eliminated.

4.3 Evaluation for Purpose

Determining from a set of ontologies which is best suited for a particular purpose is difficult. A common technique is to compare an ontology with some domain-specific corpus or "gold standard" (usually an existing ontology in the area), using lexical analysis to identify missing terms and relationships. An example of a gold standard may be a collection of

[12] The Dublin Core Metadata Initiative: `dublincore.org`

texts concerning the domain. Brewster et al. describe a process where this is achieved by performing a lexical analysis to generate a similarity measure [27].

A second approach is application, rather than domain, specific and involves evaluating ontologies by "slotting" them into the same application. This can judge the "fit" of an ontology to an application. While providing a more concrete evaluation than the lexical approach, the ontology can only be evaluated to the extent that the application is a good representation of the domain of interest. It may be difficult to generalise the use of data from one application to the next. More effort is required to manually plug-in ontologies into the same application than to compare terms against an automatically generated corpus.

From the perspective of modelling context (as it relates to pervasive computing), there is a large body of literature that could be used for lexical analysis. Comparing ontologies with the requirements set out in section 2, with the core concepts identified by the community and with existing ontologies, provides a similar, albeit more subjective, evaluation. Scenarios for application based evaluation could, in principle, be selected from the literature. However, the effort required to build these, to prove they are representative examples, and to apply multiple ontologies to them may be prohibitive in practice. Work towards realistic, nontrivial demonstrator scenarios, covering the broad spectrum of factors that affect pervasive systems would help in evaluating not only ontologies, but all aspects of pervasive systems [28]. As a halfway measure, we propose that ontologies for sensing may be evaluated by applying them to examples from live sensor feeds or published data sets.

5. AN INITIAL EVALUATION OF ONTONYM

In this section we present an initial evaluation of Ontonym based on the criteria laid out in the previous section. We are yet to perform a complete evaluation "for purpose", however, as a first step we compare Ontonym against the requirements set out in section 2 and with other ontologies in the domain.

5.1 Design Principals and Content

Ontonym is explicitly designed to support application specific concerns via extension. For example, The concepts of *Student* or *AudienceMember* are natural extensions of the *Person* concept. Similarly, an auxiliary ontology for instant messaging can link to the *Person* concept by providing a *hasIMProfile* property. Locations can be extended to represent their use in a particular application domain e.g., *SpeakerArea* and *AudienceArea*, and so on. The event ontology is designed solely for application extension – Ontonym provides only a general framework for their representation.

We have aimed to maximise orthogonality between ontologies, however, overlaps remain. For example, modelling a person's date of birth requires use of the time ontology. One technique we have used is to exploit the fact that individuals may belong to multiple classes. For example, rather than defining properties in the Person and Device ontologies to handle location, the Location ontology defines *LocatableEntity* and *LocatableFeature* classes (and associated properties) that represent physically locatable entities (e.g., people) and objects that act as proxies for their location (e.g., mobile phones and Ubitags).

We have avoided encoding bias and retained formality by adopting standards where possible. For example, using XML Schema data types, OWL-Time for temporal information and using the MUO ontology for unit representation. We have aimed to achieve clarity through the use of property restrictions and expression of disjoint classes. Consistency has been achieved by adopting terms directly from the literature, from existing ontologies, and through using documentation to remove any ambiguity. We have also aimed to achieve consistency with existing ontologies by mapping between terms using the *owl:sameAs* and *owl:equivalentProperty* concepts. This, for example, enables a subset of profile information expressed in FOAF to be immediately compatible with applications that use our ontologies.

Naming conventions have been adhered to throughout the ontologies, and Ontonym has been validated using the Pellet reasoner[13]. We have attempted to address the requirement of completeness by including only those concepts that are regarded to be core to pervasive computing. With each concept we have tried to only model data that is generally useful to all applications. The accompanying documentation covers all terms in the ontology and we are in the process of developing sample code and tutorial material as a step-by-step guide describing how new adopters should use Ontonym.

5.2 Modelling Requirements

The modelling requirements of distributed composition, partial validation, and formality are implicitly met through the use of OWL. Distributed fragments of model and data can be easily merged, with numerous tools available that support validation. As all terms in an OWL ontology are uniquely identifiable through the use of namespaces, formality is also provided. Use of an RDF-based model also supports metamodelling through *reification*. This allows a single statement in the model to be associated with any number of meta-data statements (e.g., representing ownership, security, or assertion of error).

Ontonym's sensing model supports both the representation of sensors and the data they observe. Observations link to the sensor that generated them, and meta-data is applied to the sensor and observation to characterise the sensing process. Sensors indicate their sampling frequency and coverage (e.g., a positioning sensor covers a portion of a map), while observations indicate the observation time, dynamism of the sampled data, and whether the data refers to past or future state. Our sensing model supports modelling of imprecision though the use of precision-accuracy pairs that are associated with a sensor or its individual observations. Adopters are encouraged to better characterise data by extending the sensing ontology with domain-specific properties. Ambiguous data can be represented in the model through associating an object with multiple instances of a property, while the model's semi-structured nature means that unknown properties can be omitted without error.

[13] The Pellet reasoner: `clarkparsia.com/pellet`

5.3 Comparison with Related Ontologies

The Context Ontology Language (CoOL) [29] focuses on providing a comprehensive description of services and context interaction, but does not fully represent people or time, and fails to address the fundamental design principles of coherence, orthogonality and encoding bias [11]. Global Smart Spaces (GLOSS) [30] focuses on providing location-based services. The location model in Ontonym is more extensive than that of GLOSS, containing a vocabulary for representing spatial relationships and relative locations. Other aspects of the GLOSS model are less well documented. For example, a person may have a profile associated with them, but details of the profiles contents are not provided.

SOCAM [31] contains a single upper ontology that defines the concepts of activity, location, person, time, computational entity, and merchandise under the umbrella concept of *ContextEntity*. SOCAM defines few properties, but includes some, relating to noise and temperature, that are application specific. Dates and times in SOCAM are are represented using simple strings. These attributes relate to the requirements of orthogonal design, completeness, and encoding-bias which Ontonym addresses. SOCAM also contains logical inconsistencies; for example the *spatialContains* relationship relates a location to other types of *ContextEntity*, which includes the concept of time. Gu et al. define four quality parameters they use to characterise data [31], however, these do not appear in the published ontologies.

SOUPA [10] provides an agent centric set of ontologies covering the main concepts in pervasive computing. Ontonym follows SOUPAs lead in using established standards for representing time and events [19]. SOUPA's person ontology is an extension of FOAF, which we believe poorly matches the domain due to the inclusion of application specific properties (e.g., online gambling accounts), and a single, vague concept (knows) of a relationship between people. FOAF also contains inconsistent definitions; for example, properties such as age and gender have the domain *Agent*, the definition of which includes groups, software, and physical artefacts in addition to people. SOUPA draws from several vocabularies for representing location, but there is no documentation on how they are combined or used in practice.

To the best of our knowledge, no existing ontologies for pervasive computing support the representation of sensors or provenance information. SOCAM is the only work to describe the capture of quality parameters, although this is not present in their published ontologies. With the exception of GLOSS, documentation of these ontologies is sparse or non-existent, making them less accessible to new developers.

6. CONCLUSION

In this paper we introduce Ontonym, a set of ontologies for pervasive computing that supports an updated set of requirements for context representation taken from the literature. Ontonym covers the core concepts of *location*, *people*, *time*, *event*, and *sensing*, and is designed to be extended to model application specific concerns. By publishing[14] these ontologies along with documentation and tutorials, we hope to provide developers with a clear, structured approach to modelling pervasive systems.

We propose a strategy for evaluating ontologies, by combining approaches from several sources, that falls into three categories: *design principles*, *content* and *evaluation for purpose*. A partial evaluation of Ontonym is carried out from these perspectives.

In the future, we will perform a user study to evaluate Ontonym from an applied perspective. Another strain of work will apply reasoning techniques to the Ontonym data model to investigate approaches to situation recognition.

7. ACKNOWLEDGMENTS

This work is partially supported by Science Foundation Ireland under grant numbers 07/CE/I1147, "Clarity, the Centre for Sensor Web technologies", 04/RPI/1544, "Secure and Predictable Pervasive Computing", and 05/RFP/CMS0062, "Towards a Semantics of Pervasive Computing".

8. REFERENCES

[1] Mark Weiser. The computer for the 21st century. *Human-computer interaction: toward the year 2000*, pages 933–940, 1995.

[2] P. Dourish. What We Talk About When We Talk About Context. *Personal and Ubiquitous Computing*, 8(1):19–30, February 2004.

[3] N. Baumgartner and W. Retschitzegger. A survey of upper ontologies for situation awareness. In *In Proceedings of the 4th IASTED International Conference on Knowledge Sharing and Collaborative Engineering*, pages 1–9, St. Thomas, US Virgin Islands, November 2006.

[4] T. Strang and C. Linnhoff-Popien. A context modeling survey. In *Proceedings of the Workshop on Advanced Context Modelling, Reasoning and Management as part of UbiComp 2004 - The Sixth International Conference on Ubiquitous Computing*, Nottingham, UK, September 2004.

[5] K. Henricksen, J. Indulska, and T. McFadden. Modelling context information with orm. In *Proceedings of On the Move to Meaningful Internet Systems 2005: OTM Workshops*, pages 626–635, 2005.

[6] K. Henricksen and J. Indulska. Modelling and using imperfect context information. In *Proceedings of the Second IEEE Annual Conference on Pervasive Computing and Communications Workshops*, page 33, Washington, DC, USA, 2004. IEEE Computer Society.

[7] P. Buneman, S. Khanna, and W. C.Tan. Data provenance: Some basic issues. In *Proceedings of the 20th Conference on Foundations of Software Technology and Theoretical Computer Science*, pages 87–93, London, UK, 2000.

[8] K. Henricksen, J. Indulska, and A. Rakotonirainy. Modeling Context Information in Pervasive Computing Systems. In *Proceedings of the International Conference on Pervasive Computing 2002 (Pervasive 2002), Zurich, Switzerland*, 2002.

[9] K. Henricksen, R. Wishart, T. McFadden, and J. Indulska. Extending context models for privacy in pervasive computing environments. In *Proceedings of*

[14] The Ontonym ontologies are available from `ontonym.org`

the 2nd International Workshop on Context Modelling and Reasoning (CoMoRea), pages 20–24. IEEE Computer Society, 2005.

[10] H. Chen, F. Perich, T. Finin, and A. Joshi. SOUPA: Standard Ontology for Ubiquitous and Pervasive Applications. In *International Conference on Mobile and Ubiquitous Systems: Networking and Services*, Boston, MA, August 2004.

[11] J. Ye, L. Coyle, S. Dobson, and P. Nixon. Ontology-based models in pervasive computing systems. *The Knowledge Engineering Review*, 22:315–347, December 2007.

[12] J. F. Allen. Towards a general theory of action and time. *Artificial Intelligence*, 23(2):123–154, 1984.

[13] J. Hightower and G. Borriello. Location systems for ubiquitous computing. *IEEE Computer*, 34(8):57–66, August 2001.

[14] R. Want, A. Hopper, V. Falcao, and J. Gibbons. The active badge location system. *ACM Transactions on Information Systems*, 10(1):91–102, 1992.

[15] C. Jiang and P. Steenkiste. A hybrid location model with a computable location identifier for ubiquitous computing. In *Proceedings of The Fourth International Conference on Ubiquitous Computing, UbiComp 2002*, pages 246–263, London, UK, 2002.

[16] J. Ye, L. Coyle, S. Dobson, and P.Nixon. A unified semantics space model. volume 4718, pages 103–120, Oberpfaffenhofen, Germany, September 2007.

[17] Roy Want. You are your cell phone. *IEEE Pervasive Computing*, 7(2):2–4, 2008.

[18] S. Coetzee, A. K. Cooper, M. Lind, M. M. Wells, S. W. Yurman, E. Wells, N. Griffiths, and M. J. L. Nicholson. Towards an international address standard. In *Proceedings of the Tenth International Conference for Spatial Data Infrastructure Conference*, Trinidad and Tobago, 2008.

[19] F. Pan and J. T. Hobbs. Time in owl-s. In *Proceedings of the AAAI Spring Symposium on Semantic Web Services*, pages 29–36, California, USA, 2004.

[20] T. R. Gruber. Towards principles for the design of ontologies used for knowledge sharing. In N. Guarino and R. Poli, editors, *Formal Ontology in Conceptual Analysis and Knowledge Representation*, Deventer, The Netherlands, 1993.

[21] M. Uschold and M. Grüninger. Ontologies: Principles, Methods, and Applications. *Knowledge Engineering Review*, 11(2):93–155, 1996.

[22] R. M. Colomb. Quality of Ontologies in Interoperating Information Systems. Technical Report ISIB-CNR-TR-18-02, Padova, Italy, 2002.

[23] A. Gomez-Perez. *Handbook On Ontologies*, chapter 13. Springer, 2003.

[24] R. Porzel and M. Rainer. A task-based approach for ontology evaluation. In *Proceedings of the ECAI Workshop on Ontology Learning and Population*, 2004.

[25] J. Brank, M. Grobelnik, and D. Mladenic. A survey of ontology evaluation techniques. In *Proceedings of the Conference on Data Mining and Data Warehouses (SiKDD 2005)*, 2005.

[26] M. Sabou, V. Lopez, and E. Motta. Ontology selection for the real semantic web: How to cover the queenÕs birthday dinner? *Managing Knowledge in a World of Networks*, 4248/2006:96–111, 2006.

[27] C. Brewster, H. Alani, and A. Dasmahapatra. Data driven ontology evaluation. In *Proceedings of the International Conference on Language Resources and Evaluation*, Lisbon, Portugal, 2004.

[28] S. Neely, G. Stevenson, C. Kray, I. Mulder, K. Connelly, and K.A. Siek. Evaluating pervasive and ubiquitous systems. *Pervasive Computing, IEEE*, 7(3):85–88, July-Sept. 2008.

[29] T. Strang, C. Linnhoff-Popien, and K. Frank. CoOL: A Context Ontology Language to enable Contextual Interoperability. In *Proceedings of 4th IFIP WG 6.1 International Conference on Distributed Applications and Interoperable Systems (DAIS2003)*, LNCS 2893, pages 236–247, Paris, France, November 2003.

[30] J. Coutaz, A. Dearle, S. Dupuy-Chessa, G. N. C. Kirby, C. Lachenal, R. Morrison, G. Rey, and E. Zirrintsis. Working Document on Gloss Ontology. Technical Report D9.2, Global Smart Spaces Project IST-2000-26070, 2003.

[31] T. Gu, X. H. Wang, H. K. Pung, and D. Q. Zhang. An ontology-based context model in intelligent environments. In *Proceedings of Communication Networks and Distributed Systems Modeling and Simulation Conference*, pages 270–275, 2004.