Towards Reasoning About Context in the Presence of Uncertainty

Dan Chalmers *
d.chalmers@imperial.ac.uk

Naranker Dulay
n.dulay@imperial.ac.uk

Morris Sloman
m.sloman@imperial.ac.uk

Dept. of Computing, Imperial College, London, U.K.

ABSTRACT

In this paper we show how the relationships between real world actors and contextual information can be formulated in the presence of uncertainty. In particular, we show how relationship functions can be defined for uncertain numerical context values using interval arithmetic and analogously for context ontologies defined as trees. We illustrate the functions with a small example from our work on contextual mediation, where context enables specifications of behaviour.

Author Keywords

Context awareness, context models, context types, uncertainty

INTRODUCTION

The notion of context awareness, from the point of view of mobile or ubiquitous computing, has been a topic of interest since 1992 [5, 11, 9]. Context is a wide ranging concept, which includes location, user identity, time, device, network connection, I/O devices, social setting and what people and devices are nearby. We use a definition of context based on Dey and Abowd [4] and the Oxford English Dictionary:

> Context is the circumstances relevant to the interaction between a user and their computing environment [2].

In the worlds of mobile and ubiquitous computing, where a wide range of devices are applied in many different situations to support many tasks, the context of use will have a substantial impact on the appropriate behaviour of applications, without being a primary input source.

Context is not static in definition or state – one of the properties of context is that it describes a changing relationship between users, systems and their environment. Describing these relationships is crucial in any model of context which seeks to address scenarios beyond isolated users. A key issue is the treatment of uncertainty in the relationships – the quality of the sensed context data will vary due to noisy sensors, erroneous readings, out of date data etc.

We identify six uses of contextual information, drawing from [1, 2, 3, 4, 9]:

1. *Context display* where sensed context is presented to the user, e.g. display of current location.
2. *Contextual augmentation* annotates data with the context of its generation, e.g. meeting notes can be associated with people attendees and location of the meeting [1].
3. *Context aware configuration* e.g., to cause printing to be on the nearest printer, or cause selection of nearby proxies when needed.
4. *Context triggered actions* [9] such as loading map data for the next location predicted.
5. *Contextual mediation* is the use of context to modify services provided or the data requested to best meet the needs and limits arising from the context of the interaction [2].
6. *Context aware presentation* refers to the adaptation of the user interface or the presentation of data, e.g. adjusting interaction widgets according to the display device.

In designing a representation for context it is important to consider what will be done with that information. In the display of of context data the modelled values must be visualised (in this case we also abandon the idea that the context is not central to the application). In contextual augmentation the data must be stored so that any future indexing or searching is facilitated. In context aware configuration the model must help answer questions such as "find the nearest". In the last three cases (often collected together under the heading *contextual adaptation* [4]) context data may be matched to test conditions to enable or trigger specifications, policies or actions. Context may also be used as parameters to modify these responses. For instance, we may want to trigger policies relating to being "in a car" when driving, while also changing the behaviour of a navigation system depending on how far we are from our destination. In these cases we need well typed context data which can be used to make tests such as equality, difference and order; and also to be used as function parameters. In this paper we shall focus on defining relationships between context values to enable these comparisons.

---

*This work was partially funded by the EPSRC platform grant GR/R 95715/01, "Aedus".

We shall start with a model of values for individual context aspects, taking into account the inevitable uncertainty in sensed data. Our model addresses the need for both numerical values and more conceptual data. We then examine important functions for comparing these values, taking into account the uncertainty in the values. The definition and use of contexts which are formed from multiple aspects are considered next. Finally we apply these ideas to an example from our work, before relating this work to some key papers, concluding and noting future work.

VALUES OF CONTEXT

We think of context as a set of relevant aspects which have some value. The view of context as a set of name, value pairs underpins many of the existing approaches to context representation. This requires two key extensions for general application: firstly, the values should be typed; secondly, the possibility that the values from sensors may not be precise must be considered.

We illustrate two people interacting, each in some context, in figure 1. Our "two layer model" follows a simplified version of the entity-relationship diagram style in Henricksen et. al.'s work [6]. We have added a differentiation between actors / objects in the environment and their context, although other people and devices can be counted as context from the point of view of any one device. The actors and context objects are further described by their attributes and the relationships amongst them. In this way we show context, such as location and activity, differently from attributes of objects, such as name and type. Below the diagram we illustrate typical types for the values and how the relationships may be defined in terms of these. We have omitted identity for brevity, although this can be assumed to be a plausible attribute for most objects in the model. The type of the attribute can be numerical ($\mathbb{N}$) or a node from a hierarchy of concepts ($\mathbb{T}$). These two classes of context form natural description types: many sensors will provide raw numerical data, which is often directly useful. Other aspects of context are more readily expressed using conceptual values, such as activities being performed, while some numerical aspects may be processed to form classifications rather than exposing spurious accuracy, such as light levels as bright, moderate, dim. We have also omitted much of the detail in structure and origin of the relationships that Henricksen shows as a design tool.

Uncertainty in Context

Sensor error (both inherent granularity and due to false readings), out of date data and poor predictions will give rise to some uncertainty about sensed context in most cases. To some extent this may be mitigated by applying fusion to multiple readings [7], but some uncertainty will remain. If an application could describe the confidence it requires in the context data, the returned value can be a value range which the context awareness system believes includes the current context within the certainty constraint. It can be expected that a higher confidence can be given to a larger range of values, while a response with a smaller range may be given if one relaxes the need to include less likely possibilities. The underlying model of values can be abstracted by this mechanism: logs of historical values, Bayesian models and simple averaging models can all be arranged to return a value range for a confidence level, although the underlying model can be recovered if the source of uncertainty is significant. One advantage of this approach is that a trade-off between certainty and cost (power, network load etc) is possible where context sensing is distributed.

$T(\mathcal{C}_a)$ gives the type of the context aspect $\mathcal{C}_a$. So far we have worked with types of $(\mathbb{N}, u)$ for numerical values ($\mathbb{N}$) with units (u), and $(\mathbb{T}, t)$ for values from a tree ($\mathbb{T}$) describing a hierarchy of concepts, defined in an ontology (t). The numerical model is similar to that used in interval arithmetic. We use the notation $\mathcal{C}_{a,p,o}$ to denote a value of context for aspect, a, at a given certainty, p, describing an actor or object, o – which gives a range of values:

$$\begin{array}{ll} \text{if } T(\mathcal{C}_a) \in (\mathbb{N}, u) & \text{then } \mathcal{C}_{a,p,o} = (c^{min}_{a,p,o}, c^{max}_{a,p,o}) \\ \text{if } T(\mathcal{C}_a) \in (\mathbb{T}, t) & \text{then } \mathcal{C}_{a,p,o} = (c^{0}_{a,p,o}, \ldots, c^{n}_{a,p,o}) \end{array}$$

The certainty is a probability that the value range contains the correct value, $0 < p \leqslant 1$. Nodes on the tree have unique IDs and so the values for $\mathbb{T}$ can be interpreted in relation to t so their relationship to other values is still understood. We leave the definition for $\mathbb{N}$ with multiple ranges for the future.

It is also useful to describe an overall context for an object with a given certainty, $\mathbf{C}_{p,o}$. This is a set of values for various aspects, each of which is known with certainty p:

$$\mathbf{C}_{p,o} = \bigcup_{\forall relevant\ aspects,\ a} \mathcal{C}_{a,p,o}$$

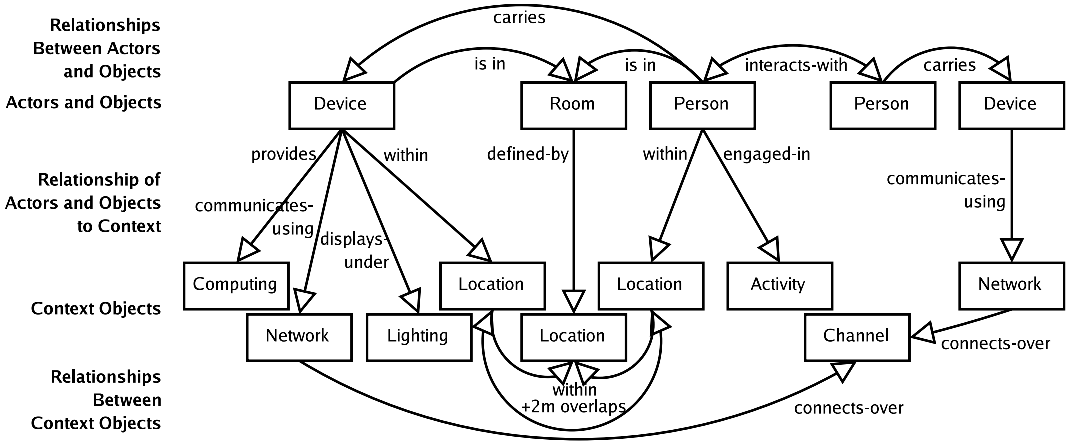
Note that we do not define the names of aspects or any order over them as the context which is relevant and can be sensed will be highly variable.

COMPARING CONTEXTS

Figure 2 illustrates the relationships between context values which we define more formally in the following. Here we illustrate relationships over both numerical and tree value types with uncertain values. The boxes with dotted lines indicate the range a value occupies. Some of these relationships are then specified in table 1.

Equals is a simple base case relationship, supporting questions such as: "does my location equal 'in a car?"'. We find that having acknowledged that context values may be uncertain a straightforward equality relationship is seldom used. The *within* relationship is often more useful, supporting questions such as: "am I in a vehicle?", "is my speed between 30 and 50km/h" and "who is within 5m of me?" (by expanding the current location by 5m and performing a within test on this). With this we test whether a sensed value range is within a test range, e.g. if the sensed speed is 29-31km/h this would be within the range 25-35km/h, but not within 30-35km/h.

A less stringent test is the *overlaps* relationship. This gives a *degree* which is the proportion of the first value by which



Relationships Between Actors

Actor	Relationship	Actor	Typical Relationship Test
person	carries	device	within distance
person	interacts with	person	related roles, history of communication
device	is in	room	device.location within room.location

Relationships Between Actors and Context

Actor	Relationship	Context	Type and Attributes of Context	Typical Relationship Test
device	displays under	lighting	level ($\mathbb{N}$), type of light ($\mathbb{T}$)	within range, equals type
device	communicates using	network	type (protocols) ($\mathbb{T}$)	has / equals, find best
device	within / at	location	coordinates ($\mathbb{N} \times \mathbb{N}$), room id ($\mathbb{T}$)	within nearest known point
device	provides	computation	cpu power, memory etc. ($\mathbb{N}$ or $\mathbb{T}$)	has / equals
person	engaged in	activity	ontology ($\mathbb{T}$)	within class / equals
person	within / at	location	as above	within / nearest known point
network	connects over	channel	bandwidth, round trip time etc ($\mathbb{N} or \mathbb{T}$) .	has / equals

Figure 1. Relationships Between Actors and Context Objects in A Context Model

the two values overlap. With this we test whether two value ranges have some common values. Taking the speed example again, 29-31km/h overlaps both 25-35km/h and 30-35km/h. In the first case the degree of overlap (with 29-31km/h as the first term) is 1, while in the second it is 0.5. Using a tree based value we count the proportion of overlapping nodes in the first term.

Note that as *overlaps* is not reflexive then an overlap of one is equivalent to a *within* relationship:

$$\mathcal{C}_{a,p,o} \sqsubset \mathcal{C}_{a,q,o'} \;\rightarrow\; \mathcal{C}_{a,p,o} \sqcap \mathcal{C}_{a,q,o'} = 1$$

The best match between a context value and two test contexts which both satisfy within can also be found by the degree of overlap. While $\mathcal{C}_{a,p,o} \sqcap \mathcal{C}_{a,q,o'} = 1$ in both cases $\mathcal{C}_{a,q,o'} \sqcap \mathcal{C}_{a,p,o}$ will be highest in the case with the closest match, which may be useful for selecting the most appropriate action.

The *overlaps* relationship for trees of abstract values is slightly less intuitive in use as it assumes all nodes on the tree have equal weight. The *degree* is then the number of nodes which are present in both values divided by the number of nodes is the first value. In the future we may wish to examine trees where the nodes are weighted, to reflect semantic weight, e.g. room size, or to reflect the certainty that a value is correct (within the certainty specified), so a more likely value within the range has a higher weight.

Using within we can say that a loss of certainty will give an equal or broader range of context values:

$$\mathcal{C}_{a,p,o} \sqsubseteq \mathcal{C}_{a,p',o} \text{ where } p \geqslant p'$$

A loss of certainty will preserve both a within and an overlaps relationship although the degree of overlap may change. Only a loss of certainty in the second value ($\mathcal{C}_{a,q,o'}$) can be guaranteed to preserve or increase the degree of overlap:

$$\text{Given } d = \mathcal{C}_{a,p,o} \sqcap \mathcal{C}_{a,q,o'} \text{ and } d' = \mathcal{C}_{a,p',o} \sqcap \mathcal{C}_{a,q',o'}$$
$$d \leqslant d' \text{ if } q < q'$$

Over an increase in certainty neither within nor overlaps relationships can be guaranteed to hold.

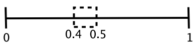
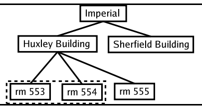
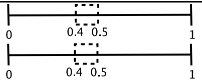
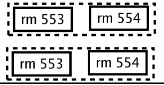
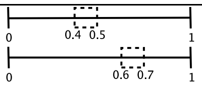
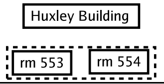
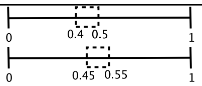
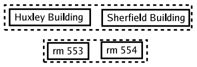
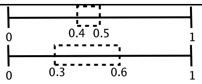
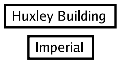
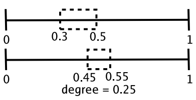
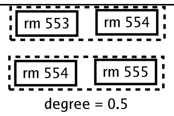
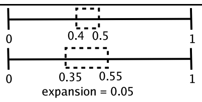
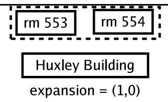
Relationship	Example with $\mathbb{N}$	Example with $\mathbb{T}$	Notes
Example Values in Type			A number between 0.4 and 0.5; a room which is either “rm553” or “rm554” within “Huxley Building”.
Equality			The values are exactly equal.
Strict Order (top $<$ bottom)			All values in top are less than all values in bottom.
Relaxed Order (top $<$ bottom)			All values in top are less than some value in bottom.
Within (top $\sqsubset$ bottom)			All values in top are within the range in bottom.
Overlaps (top $\sqcap$ bottom)			Some values in top are within the range in bottom. The degree is the proportion of top by which they overlap.
Expand (expand top to give bottom)			Expand top by expansion degree to give bottom (degree for $\mathbb{T}$ as for difference).

Figure 2. Relationships for Numbers and Trees with Uncertain Values

Similarly, while functions may transform context values between types (changes of unit and/or representation class), at best these are lossless but they cannot add precision:

$$\mathcal{C}_{a,p,o} \sqsubseteq f'(f(\mathcal{C}_{a,p,o}))$$

where f' performs a reverse direction of translation to f

In figure 2 we illustrate two forms of order: *strict order* – where the ranges may not overlap and *relaxed order* – where the ranges may overlap. *Expand* modifies the range of a value, enabling tests for proximity etc. Space does not allow us to expand on these relationships here.

Compound Contexts

As discussed earlier context is formed of many aspects and we may wish to consider several aspects at once when evaluating a response to context. We show a key relationship between compound contexts in table 2: *more specific context*. This is used to identify a context which refers to a less wide ranging set of conditions than another. $\mathbf{C}_{p,o}$ is equal or more specific than $\mathbf{C}_{q,o'}$ when three conditions hold: 1) where values for all aspects in $\mathbf{C}_{p,o}$ are either within or equal to some value in $\mathbf{C}_{q,o'}$ or that aspect is not present in $\mathbf{C}_{q,o'}$; 2) the value for all aspects of $\mathbf{C}_{q,o'}$ contain or equal that aspect's value in $\mathbf{C}_{p,o}$; 3) that the value for some aspect in 1 is not equal but either within / not present in $\mathbf{C}_{q,o'}$. An example use of this is to enable a general behaviour while “at work” but have more specific behaviours defined for “in a meeting room” and “in my office”.

Where an aspect of the context becomes more certain this will result in a more specific context, and vice-versa.

Example: Contextual Mediation

We motivated these relationships in terms of a design model and simple examples above. We now describe their use at an implementation level, drawing on our work on contextual mediation of maps [2, 3].

A location server makes a location prediction for a user and causes a new map to be loaded. The context is then used to adjust the map which is loaded, for instance: If the user is travelling slowly near their destination the desired map will be small scale, showing high detail and will refresh more frequently. If the user is further from their destination on a main road the map will have a larger scale, showing less detail of small features in order to show a greater area and so giving a longer term view and give preference to major roads. The selection of extra features, such as shops, restaurants etc. may depend on who else is present, whether the journey is for work or leisure, acceptable delay and level of detail on the screen etc.

Context has two functions here: Firstly, to trigger the loading of new map data and selection of the relevant map area and scale (context triggered action and indirect context display). The relevant area is found by expanding the current or predicted location. Secondly, to control the selection of data: according to stated preferences for types of data, download deadlines and screen clutter (contextual mediation).

In our work on contextual mediation specifications of preferences were enabled according to context matching functions. The specifications reflected the context they matched against, e.g. behaviour for different speeds of travel could be separated from behaviour for different modes of trans-

Within
$\mathcal{C}_{a,p,o}$ is within $\mathcal{C}_{a,q,o'}$ is defined as:

$$\begin{array}{lll}
\mathcal{C}_{a,p,o} \sqsubset \mathcal{C}_{a,q,o'}\ , & \text{where } (c^{min}_{a,q,o'} < c^{min}_{a,p,o} \wedge c^{max}_{a,p,o} \leqslant c^{max}_{a,q,o'}) \vee (c^{min}_{a,q,o'} \leqslant c^{min}_{a,p,o} \wedge c^{max}_{a,p,o} < c^{max}_{a,q,o'}) & \text{if } T(\mathcal{C}_a) \in (\mathbb{N}, u) \\
\mathcal{C}_{a,p,o} \sqsubset \mathcal{C}_{a,q,o'}\ , & \text{where } c_a \prec c_b & \text{if } T(\mathcal{C}_a) \in (\mathbb{T}, t)
\end{array}$$

$\mathcal{C}_{a,p,o}$ is within or equal to $\mathcal{C}_{a,q,o'}$ is defined as:

$$\begin{array}{lll}
\mathcal{C}_{a,p,o} \sqsubseteq \mathcal{C}_{a,q,o'}\ , & \text{where } (c^{min}_{a,q,o'} \leqslant c^{min}_{a,p,o} \wedge c^{max}_{a,p,o} \leqslant c^{max}_{a,q,o'}) & \text{if } T(\mathcal{C}_a) \in (\mathbb{N}, u) \\
\mathcal{C}_{a,p,o} \sqsubseteq \mathcal{C}_{a,q,o'}\ , & \text{where } c_a \preccurlyeq c_b & \text{if } T(\mathcal{C}_a) \in (\mathbb{T}, t)
\end{array}$$

Overlaps
Overlaps is defined as the proportion of $\mathcal{C}_{a,p,o}$ which overlaps $\mathcal{C}_{a,q,o'}$.
If $T(\mathcal{C}_a) \in (\mathbb{N}, u)$ overlaps is defined as:

$$\begin{array}{llll}
degree & = \mathcal{C}_{a,p,o} \sqcap \mathcal{C}_{a,q,o'} & = 0 & , \text{where } c^{max}_{a,p,o} < c^{min}_{a,q,o'} \vee c^{min}_{a,p,o} > c^{max}_{a,q,o'} \\
 & & = \frac{min(c^{max}_{a,p,o}, c^{max}_{a,q,o'}) - max(c^{min}_{a,p,o}, c^{min}_{a,q,o'})}{c^{max}_{a,p,o} - c^{min}_{a,p,o}} & , \text{otherwise}
\end{array}$$

If $T(\mathcal{C}_a) \in (\mathbb{T}, t)$ overlaps is defined as:

$$\begin{array}{lll}
degree & = \mathcal{C}_{a,p,o} \sqcap \mathcal{C}_{a,q,o'} & \\
 & = 0 & , \text{where } \forall c^x_{a,p,o} \in \mathcal{C}_{a,p,o},\ \forall c^y_{a,q,o'} \in \mathcal{C}_{a,q,o'},\ c^x_{a,p,o} \not\sqsubseteq c^y_{a,q,o'} \\
 & = \dfrac{|\forall c^x_{a,p,o} \in \mathcal{C}_{a,p,o} \text{ where } \exists c^y_{a,q,o'}\ ,\ c^x_{a,p,o} \sqsubseteq c^y_{a,q,o'}|}{|\mathcal{C}_{a,p,o}|} & , \text{otherwise}
\end{array}$$

$|\mathcal{C}_{a,p,o}|$ here indicates a count of nodes in the set.

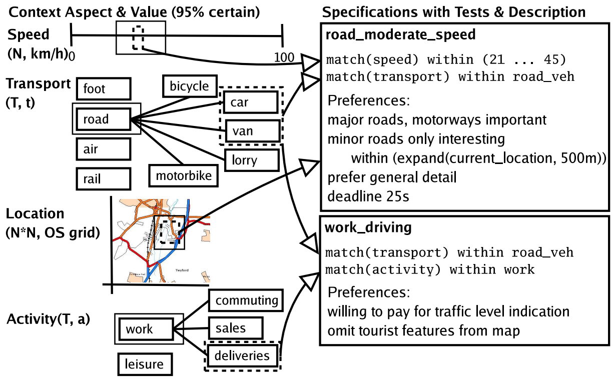
Table 1. Relationships for Numbers and Trees with Uncertain Values

$\mathbf{C}_{p,o}$ is more specific than $\mathbf{C}_{q,o'}$ (no relationship between p and q assumed) is defined by:

$$\begin{array}{lll}
\mathbf{C}_{p,o} \sqsubseteq \mathbf{C}_{q,o'} & \leftrightarrow & \forall \mathcal{C}_{a,p,o} \in \mathbf{C}_{p,o}, ((\exists \mathcal{C}_{a,q,o'} \in \mathbf{C}_{q,o'}, \mathcal{C}_{a,p,o} \sqsubseteq \mathcal{C}_{a,q,o'}) \vee (\neg \exists \mathcal{C}_{a,q,o'} \in \mathbf{C}_{q,o'})) \wedge \\
 & & \forall \mathcal{C}_{a,q,o'} \in \mathbf{C}_{q,o'}, (\exists \mathcal{C}_{a,p,o} \in \mathbf{C}_{p,o}, \mathcal{C}_{a,p,o} \sqsubseteq \mathcal{C}_{a,q,o'}) \wedge \\
 & & \exists \mathcal{C}_{a,p,o} \in \mathbf{C}_{p,o}, ((\exists \mathcal{C}_{a,q,o'} \in \mathbf{C}_{q,o'}, \mathcal{C}_{a,p,o} \sqsubset \mathcal{C}_{a,q,o'}) \vee (\neg \exists \mathcal{C}_{a,q,o'} \in \mathbf{C}_{q,o'}))
\end{array}$$

Table 2. Equal or More Specific Compound Contexts

port. This is illustrated in figure 3. Dotted boxes represent the sensed context, thin boxes highlight the context the specifications match against.



```
road_moderate_speed
match(speed) within (21 ... 45)
match(transport) within road_veh
Preferences:
major roads, motorways important
minor roads only interesting
    within (expand(current_location, 500m))
prefer general detail
deadline 25s

work_driving
match(transport) within road_veh
match(activity) within work
Preferences:
willing to pay for traffic level indication
omit tourist features from map
```

Figure 3. Relationships in Specifications for Mediation

A typical specification used a conjunction of *within* relationships over a small number of context aspects (the *circumstances relevant* – or "conditions that might affect an action" (Oxford English Dictionary)). We also used the *more specific context* relationship to create a precedence over various specifications which might be enabled at the same time addressing related contexts. The specifications reflecting more specific contexts were given priority in a manner similar to object oriented inheritance. This allowed us to define general specifications, e.g. relating to "driving a vehicle", which would not suffer due to slight uncertainty in activity sensing. These were then extended where more specific context had further requirement, e.g. for "driving a lorry". If the more specific specification was erroneously not enabled the behaviour would still be broadly acceptable.

We have presented definitions of relationships which are applicable to realistic context values, using numerical and tree based unit systems with uncertain values. Through example we have demonstrated their use as general purpose descriptors for relationships between the context of different objects, or between sensed context and specifications for behaviour.

RELATED WORK

In [6] Henricksen et. al. present a model of context information for use in systems design. They highlight the following characteristics as important in modelling context: alternative representations (typing and abstraction); imperfections; temporal characteristics; and interrelation of aspects. Alternative representations can be achieved through transformation functions, which our typed system facilitates. We abstract imperfections as a lower level issue by capturing alternative values at a certainty level, without separating out the cause. We have not yet considered temporal issues, although we expect to do so. The interrelation of aspects, such as ownership and proximity of a PDA device for a person and also where various aspects are derived from one source, e.g. time, coordinates, height and speed from GPS, are not currently captured by our representation of context. In a diagram, some form of grouping box or connection between relationship arcs might be used. In the model of values references to the source of information from context objects may help describe some of these concerns.

Ranganathan et. al. [8] model context as predicates of the form context-aspect(predicates). In many cases the predicates are (actor/object, relationship, context value). The typing of the arguments is determined by the context aspect being addressed and are defined in an ontology. The typing of arguments is not restricted and so can be complex, but no handling of uncertainty is illustrated. Their first-order logic approach allows the use of universal and existential quantifiers. Our relationship descriptions could fit into a context model such as this, providing a common form for relationship expressions which accommodate uncertainty in the context values.

Our treatment of context values as ranges is inspired by Leondardt's work on location models [7]. In that work, location was treated as an area around a point, arising from the uncertainty resulting from inaccuracies in sensors, from sensor fusion where multiple sensors give different values and the inherent property of location that most objects occupy a space rather than a point. We believe that these principles extend to many aspects of context.

The approach of Schmidt et. al. in [10] could be used as an alternative to our "describe context at required certainty" approach. Here the context from sensors is processed and described in a vector of (symbolic context, certainty that it is correct) tuples. While this works for a fixed set of sensors in a pervasive computing environment remote sensors will be available. This will cause a trade-off between quality of data and the cost of obtaining multiple readings (in processing time and network and memory use), which our approach facilitates.

CONCLUSIONS

In this paper we considered some possible uses of context at run-time and also looked at the use of context in a design representation. In most of these cases the ability to define relationships between objects or to test the context of some object is vital. When one considers the issue of uncertainty in context, simple equality and ordering relationships become harder to apply. We proposed a typed model of context which accommodates both numerical and ontology-tree based values where there is uncertainty in the value. We then defined two key functions to relate contexts in this definition: within and overlaps. While these definitions are fairly intuitive their power suggests that they would have general use in ubiquitous systems. This generality is an improvement over models where the aspects of context have been limited, the treatment of error ignored, or relations have to be redefined for every aspect of context.

Our future work in this area will include: extending the types treated, e.g. to include weighted trees, ordered trees and compound numerical values; specifying further functions such as difference; and considering the effect of time.

REFERENCES

1. P. J. Brown, J. D. Bovey, and X. Chen. Context-aware applications: from the laboratory to the marketplace. *IEEE Personal Communications*, 4(5):58–64, 1997.
2. D. Chalmers. *Contextual Mediation to Support Ubiquitous Computing*. PhD thesis, Imperial College, London, UK, 2002.
3. D. Chalmers, N. Dulay, and M. Sloman. A framework for contextual mediation in mobile and ubiquitous computing. *Personal and Ubiquitous Computing*, 8(1):1–18, 2004.
4. A. K. Dey and G. D. Abowd. Towards a better understanding of context and context-awareness. In *Workshop on The What, Who, Where, When, and How of Context-Awareness, Conference on Human Factors in Computer Systems (CHI2000)*, 2000.
5. D. Duchamp. Issues in wireless mobile computing. In *Proceedings of 3rd IEEE Workshop on Workstation Operating Systems*, 1992.
6. K. Henricksen, J. Indulska, and A. Rankotonirainy. Modeling context information in pervasive computing systems. In *Pervasive 2002*, pages 167–180, Zurich, Switzerland, 2002.
7. U. Leonhardt. *Supporting Location Awareness in Open Distributed Systems*. PhD thesis, Imperial College, London, UK, 1998.
8. A. Ranganathan and R. Campbell. An infrastructure for context-awareness based on first order logic. *Personal and Ubiquitous Computing*, 7(6):353–364, 2003.
9. B. Schilit, N. Adams, and R. Want. Context-aware computing applications. In *IEEE Workshop on Mobile Computing Systems and Applications*, 1994.
10. A. Schmidt, K. Aidoo, A. Takaluoma, U. Tuomela, K. van Laerhoven, and W. van de Velde. Advanced interaction in context. In *1st International Symposium on Handheld and Ubiquitous Computing (HUC '99)*, Karlsruhe, Germany, 1999.
11. M. Weiser. Some computer science issues in ubiquitous computing. *Communications of the ACM*, 36:75–85, 1993.