Context Evidence and Location Authority: the disciplined management of sensor data into context models

C.W. Johnson
Dept of Computer Science
The Australian National University
Canberra, ACT, Australia
Chris.Johnson@anu.edu.au

David Carmichael, Judy Kay, Bob Kummerfeld
School of Information Technologies
University of Sydney
NSW, Australia
{dcarmich,judy,bob}@it.usyd.edu.au

Rene Hexel
School of Computing and Information Technology
Griffith University
Queensland, Australia
r.hexel@griffith.edu.au

ABSTRACT
Ubiquitous computing requires that data be handled across a very wide range of data rates and weights of associated meaning. A suitable system architecture is layered from event data through simple sensors, smart sensors, and smart environment agents. Upward through these layers the issues for representation and management of the data shift from the distribution and fast, bulk storage of very frequent simple data, to relatively infrequent logical deductions made against relatively complex models. The lower layers of systems for event data will be reused in different applications. In addition, abstract models of location may have a lot in common between different applications, which a common Location Authority can represent as a common model of the physical environment that changes only slowly (through manual administrative maintenance). On the other hand models of the devices and sensors within locations, and the moving population of people, are far more dynamic, require automatic updating, and different applications choose quite different attributes and relationships to model. We describe how the Merino/Personis architecture for an Intelligent Environment context supports the changes of representation of knowledge across this range and the different programming styles suited to the different levels.

Keywords
context-awareness system architecture, smart sensors, accretion-reduction, location authority

INTRODUCTION
A significant challenge for working with context in ubiquitous computing is to handle large amounts of disparate information in scalable fashion. Intelligent Environments aim to augment the physical environment with many computational devices and sensors to assist people in their activities. They must therefore combine the use of potentially large volumes of sensor data with much smaller amounts of refined data in models of the user and the environment, and enable programs to reason with the data and the models. In an Intelligent Environment there may be thousands of known places, devices, sensors and services. Its implementation must manage data at several scales: very frequent, high volume, low quality data from low level sensors; lower frequency, higher quality, more meaningful data from intermediate level "smart" sensors; and even more meaning-loaded, personalised and localised models.

The appropriate computer systems and programming tools differ across these levels. In our view, the significant components are:

- a layered IE architecture which encompasses sensors, smart sensors, and smart environment agents
- a uniform modelling representation for models of users, locations, places, sensors and other devices, and services

Location models in particular should be shareable by many applications. While modelling for specific users, and the sensors and resources at their activity locations, must take into account the privacy of individual users and personal safety associated with knowledge of their location, we wish to share the more stable, public knowledge about the locations and their relations that constitute a functional map of the physical environment. But this knowledge is not necessarily best shared as a library of functions or as a query-response service: it may take different forms better suited to particular applications and their need for simpler or richer relationships. The recent calls for a common Location Authority [8] or a common application programming interface for location sensing [5] support the need for a common knowledge base, but we do not find the evidence to support their more specific suggestions for this to be represented as an API or library, or as federations of

database services. A structured, disciplined framework approach is possible, but a solution of a single information service will remain unreachable for good reasons.

STRUCTURING CONTEXT INFORMATION AND MODELS

A disciplined approach to structuring context models is followed if there are strong guiding principles. Figure 1 illustrates the service layers of our Merino Intelligent Environment architecture[4]. The figure has two main parts. At the left, we have concentric ellipses for the layers of abstraction. These manage the movement and transformation of information between the raw physical hardware devices within the environment and the application programs that implement digital artefacts, which users can access within the Intelligent Environment. The other main part of the figure is the Context Repository and User Model at the right.

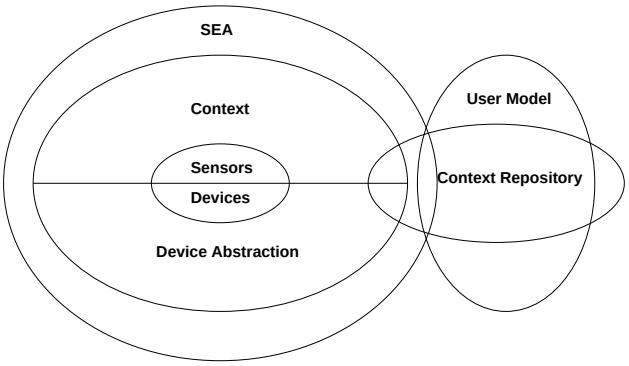

Fig 1 The Merino context architecture

Since these main elements are tightly coupled, we will explain the architecture by first showing how raw sensor information is managed, making its way to the Context Repository and User Model.

We begin with the Sensors Layer, shown in the upper part of the innermost service layer. This represents the range of sensors, which interrogate the physical and virtual environments. For example, some sensors may detect the location of a particular mobile phone. Others may detect a user's keyboard activity. A simple sensor might detect movement in an area. This layer is the part of the architecture that is directly connected to the variety of devices that are able to collect information that is needed for an Intelligent Environment.

We now move from this low level sensor to the next layer in the figure, the Context Layer. It must perform the core tasks of filtering and aggregating the raw sensor data from the lower layer. It collects data from one or more sensors and converts this to a higher level, closer to the needs of the digital artefacts. For example, there may be several temperature sensors in a room. Each of these produces data at the core Sensors Level; the Context layer combines these, perhaps with a majority voting scheme to provide a single consensus temperature value for the room. A smart location sensor may combine evidence from a number of Bluetooth device proximity sensors, desktop computer keyboard activity sensors, and prior history, to resolve an answer to the location of a particular person at a given moment.

The Context Repository is a key element, that unifies and manages the whole environment. The Context Layer interacts with the Context Repository. This holds collections of arbitrary context objects. Taking a very simple example, the Sensors Layer may detect that the user's keyboard is active. This is passed to the Context Layer which lodges the information directly into the Context Repository.

Context objects contain information about a sensor, a room, a device or any entity that has a role in the system. Taking a very simple example, a movement detector has an associated object containing its status: movement detected or not. For a mobile phone, there may be an object that contains information about the device such as its phone number, its Bluetooth MAC address, its owner and other capabilities.

The outer layer, the Smart Environment Agent Layer, takes information from the Context Layer, via the Context Repository and the User Model to construct higher level context information.

This Smart Environment Agent Layer is the level of abstraction in the architecture is where the highest level of context information is handled. The same information from the lower layers may be reused by several Smart Environment Agents which operate at this level. Different agents may make different interpretations of the same information. For example, one digital artefact might interpret location information by very rigorous standards: it may require high levels of certainty about the user's location before it will treat the user's location as known. Another artefact might require lower certainty standards. This would mean that the same sensor data, interpreted by the Context Layer, then stored in the Context Repository would be regarded as giving the user's location for one artefact while the other would consider the user's location as unknown.

The figure shows the User Model overlapping the Context Repository. Indeed, our architecture treats these as quite similar. The main difference is that the User Model contains information about people that is tagged to their identity. By contrast, the Context Repository holds context information from the Context Layer. This includes information from sensors which is used to model the environment. It also includes information from sensors that can model aspects of people. For example, information from movement detectors, Bluetooth detectors and keyboard activity monitors all provide information about people. While that information is anonymous, it can be kept in the Context

Repository. Once it can be associated with an individual, it should be kept in the User Model.

The reason that we distinguish these two forms of information is that the User Model has to be treated as personal information about people. This means that it is subject to additional constraints. In particular, the security and privacy of this data will need to conform to the requirements of emerging legislation, both national and international. Associated with this, we need to ensure that it is managed in ways that enable the user to scrutinise the information kept and the ways that it is managed, interpreted and used. The user must also be able to maintain a sense of control over these aspects of their user model.

This architecture has been implemented by a combination of sensor-event distribution (using a publish/subscribe service provided by the Elvin message routing scheme[1]), and accretion-resolution for the smarter sensors[2,3], feeding to a set of smart environment agents that work to an outer layer of applications realised as agents, communicating by blackboard systems.

ACCRETION-RESOLUTION FOR MODELLING CONTEXT

The accretion-resolution representation of our models has two basic operations[2]. The first, accretion, involves collection of uninterpreted evidence about the user. Each piece of evidence includes the source and the time associated with it. The second operation is resolution, the interpretation of the current collection of evidence when we need to know the user's location or the values of other aspects of the user model.

The accretion step allows the accumulation of raw sensor data at various interested nodes of the context model, as the data is distributed. For example, wireless proximity "sightings" of mobile devices by fixed wall sensors will be distributed to the model nodes corresponding to those devices; sensing human keyboard or mouse activity on a fixed desktop computer will be distributed to the model nodes corresponding to the logged-on or owning user of the computer.

The resolution step allows for the reduction of the amount of information flowing from low level sensors, and the merging of disparate evidence streams into single conclusions on demand. It is also the first appearance of intelligence: a resolution decision-procedure is able to resolve conflicting evidence from various sensors whether of the same or different types. For example, when the user's location is needed, a resolver is invoked to interpret the available evidence. A very simple resolver might always take the most recent sighting as indicating the true location. A slightly smarter resolver can treat one particular sensor as more reliable than another, and resolve apparent conflicts of reported location in its favour. A smarter resolver again might include a knowledge of the phenomenon of jitter between adjacent sensors, reporting a stationary subject that is apparently hopping about between regions that in fact have slightly overlapping ranges, and resolve the apparent conflict as indicating a location in the smallest known region that encloses the two sensors' detection range.

LOCATION AUTHORITY AND LOCATION CONTEXT INFORMER: STRUCTURING LOCATION CONTEXT INFORMATION

Context-aware computing is characterised as in part depending on "where", among the four Ws of "who, where, what, and when". Designating the "where" as being "location-aware" computing is a descriptive term that appeals to our human experiences, and brings with it suggestion of all the rich semantics we associate with locations, places, situations: the flavours of social situations, our mental maps of cities, buildings, landscapes, countries, districts, personal routes, conscious and unconscious navigation. But collectively we are guilty of trying to overload the notion of "location" in location-aware computing with too many different objectives. The "computing" of location-aware computing therefore presents an enormously difficult challenge, if we allow this rich range of associations that we have with "place" in everyday life to lead us into trying to do all of this with computing models of location. Mark Weiser's vision for ubiquitous computing [9] appears as grand and open-sided as any general goal of mimicking of human intelligent behaviour of the early decades of AI, and we need to exercise conscious restraint in our choice of targets in order to make our small steps of progress not disappointingly short of the vision. Even if we reduce our sights from *awareness,* to computing systems with location-*appropriate* behaviours, we should still take care to restrain our modelling ambitions. Keeping the software models simple and application-specific will be the better engineering for some time to come.

An information model of location for location-aware computing has two distinct aspects:

1. A Location Authority: an authoritative aspect, that describes the relatively static properties and simple static relationships related to a spatial region in one of two ways (a coordinate system or a place naming system—or a third, a landmark system)
2. A Location Context Informer: an informational aspect that provides information about the existence and location of entities within the region of interest: the entities are probably mobile (subject to changes in location), may be identified or anonymous, may be typed

These aspects can be incorporated in different ways into computing systems that implement Intelligent Environments.

The Location Authority may represent location data as coordinates, as named entities, or as spatial relationships to named landmarks. For two different location authorities these systems of identification may have no overlap, and

need no explicit correspondence that is of interest or of practical value to particular location-aware applications. For example, in a home or office situation it is the names of rooms and zones of the building, and possibly navigational connections between them, that are sufficient. We need not know the spatial alignment in coordinates between rooms on one floor or another, nor the precise size of the room, nor whether the corners are square. More precision in coordinate mapping is simply wasted here, and may be too expensive to sense and to map. Coordinates do not tell a location-aware system the useful relationships of how to travel between rooms, or which printer is accessible, has a short print queue, and is close by, as the person walks. Attempting to overlay one location authority map on another may also be fruitless. Coordinate space may be too rich for our simple-minded computing systems: we can still proceed more usefully with more abstract, simplified models.

A Location Authority should also report some notion of precision, which may be high—or may usefully be low, in that a particular authority may usefully allow or encourage a degree of overlap between separate named entities without explicitly noting the relationship. Activity zones or occupational areas within a building or a city are examples: the precision of rooms with boundary walls encourage us to think that all locations can be separated like that, until we start to include corridors and stairways and circulation or connection zones in the model, or to consider the useful approximations that can be made in modelling human activities. A location authority also comes with some notion of its dates or times of validity, and for any life expectancy beyond a few months the authority should identify a method for maintaining its accuracy as a map of the physical world. By physical and geological processes land masses and tectonic plates displace by centimetres a year, buildings slump, crack and slide; by human processes rooms and floors are reconfigured, rooms are relabelled, access doors are opened and closed, stairwells and corridors are blocked for maintenance; roads are opened, rerouted, closed, in the span of a few months to years. As we approach ubiquitous computing systems it will be too expensive to remap location authorities from scratch, and our needs are too specialised to depend on other mapping systems.

The second aspect of location knowledge is the Location Context Informer. Although Shafer lays out a common set of functions for the Location Authority, we see a need to distinguish between applications by such properties as their public or private availability, their knowledge of people or of inanimate things, and the embedding of the information user within the physical spaces and entity sets about which information is maintained. The most important classes of Location Context Informer answer different questions:

1. where am I?
2. where is Fred (some other person)?
3. what entities are at location L?

Again, the implications for a variety of information gathering, modelling, and dissemination processes are so strong that we see good reasons for being able to separate these functions rather than strive always to combine them in a super-system, and for location-aware systems to provide only one class of information, in some cases. The question "where am I?" has few privacy or security connotations. It may be answered by the user's handheld device identifying fixed beacons in its proximity, against a publicly available or pre-stored gazetteer. It therefore requires no dynamic information to leave the user's carried device, and does not have the implications of the system tracking the user. This may be very significant for user acceptability of a system.

The question "where is Fred?" is more loaded with cautionary principles of tracking people in real time: simple privacy, and physical security, if the system is so open that the potential questioners include Black Hats, those with evil intent. The answer may be a counter-question: "so who's asking?", requiring the Informer to have both authentication of the querier and knowledge of security or privacy control relationships between the two parties. The context informer may need a way to obfuscate locations consistently, blurring time or space to protect the object of the inquiry.

The location context informer has further characteristics that distinguish variants' properties: an answer may be a set of locations with associated probabilities, interpreting information from unreliable sensors or time snapshot samples of a rapidly moving subject. Alternatively, uncertainty may be resolved by formulating a reply naming a location that physically contains all of the most likely locations, or that socially or behaviourally contains them. If knowledge of the location is required to determine behaviour such as the ringing volume of a mobile phone for incoming calls for social reasons, an uncertainty in location should be resolved to the one with more socially conservative behaviour. An application that has safety as its prime concern and seeks to increase the volume of the phone used as a safety communicator in noisy environments might take the opposite resolution.

Other variations in requirements such as the temporal accuracy of the answer as well as its spatial accuracy, and the timeliness of producing replies to these questions, also lead to quite different implementations being appropriate for location context informers. In many cases a spatial accuracy of around several metres, or a room-sized space, is appropriate and sufficient for modelling user activities, and does not require the precision knowing particular books on a shelf or position across a virtual whiteboard [6,7].

The characteristic of our approach is that we are keeping our models within the information requirements of simple scenarios, rather than trying to take a overly general toolkit approach to location. Humans make such a variety of uses

of locational relationships in their world models of information abstractions, human dealings within cities, and in their own cognition, that implementing a single general model appears impossible.

ENGINEERING CONTEXT SOFTWARE

When we consider the design and management of software that applies context models, the resolution procedure is a pivotal point in system construction, enabling the designer to consider the substitution of meaningful variants of function: variants which are more efficient, more accurate, or more accommodating all have their place in different applications. For example, a simple system may tolerate inaccuracies in location as long as the answer comes quickly; a location-aware system for a more security conscious operation may require a high degree of accuracy or an explicit report that the person's location is unknown at some time. The research literature reports many such variant systems, which the accretion-resolution architecture would enable to be readily constructed as variants on a single system.

The generality and flexibility of the articulated Merino/Personis/Location Authority model enables very flexible system construction, including easy redistribution of components between distributed computing elements. This flexibility and ability to reuse parts of models is underlined by our use of a specific design notation for the elements that we have described here: sensors, evidence accretors, resolvers (or smart sensors), location authority gazetteers (location and entity relation tables), and the like. An example is shown in figure 2, below.

ACKNOWLEDGMENTS

We acknowledge the support of the Smart Internet Technology Cooperative Research Centre, Australia. This work was done within the SITCRC Intelligent Environments research program and its Architecture project..

REFERENCES

1. Fitzpatrick, G., Mansfield, T., Kaplan, S., Arnold, D., Phelps, T., Segall, B. Instrumenting the Workaday World with Elvin, *Proceedings of the ECSCW'99*, Copenhagen, Denmark, 1999, p 431-451
2. Kay, J, Kummerfeld, B, Carmichael, D.J. *Consistent modelling of users, devices and environments in a ubiquitous computing environment*, TR-547, School of Information Technologies, University of Sydney, NSW, Australia, June 2004.
3. Kay, J, Kummerfeld, B, Lauder, P. Managing private user models and shared personas, *UM03 Workshop on User Modeling for Ubiquitous Computing* 2003 p 1-11.
4. Kummerfeld, B, Quigley, A, Johnson, C.W., Hexel, R. Merino: Towards an intelligent environment architecture for multi-granularity context description, *UM03 Workshop on User Modeling for Ubiquitous Computing* 2003.
5. Patterson, C.A., Muntz, R.R. and Pancake, C.M. Challenges in Location-Aware Computing, *IEEE Pervasive Computing*, 2(2), April-June 2003, p 80-89.
6. Priyantha, N, Chakraborty, A. and Balakrishnan, B. The Cricket location-support system, *Mobile HCI* 2002, p 45-59.
7. Scott, J and Hazas, M. User-Friendly Surveying Techniques for Location-Aware Systems, in *Ubicomp 2003*, pp. 44-53.
8. Shafer, S.A.N. Location Authorities for Ubiquitous Computing, *Proc. 2003 Workshop on Location-Aware Computing*, p 13-15.
9. Weiser, M. The computer for the twenty-first century, *Scientific American*, 265(30) p 94-104, September 1991.

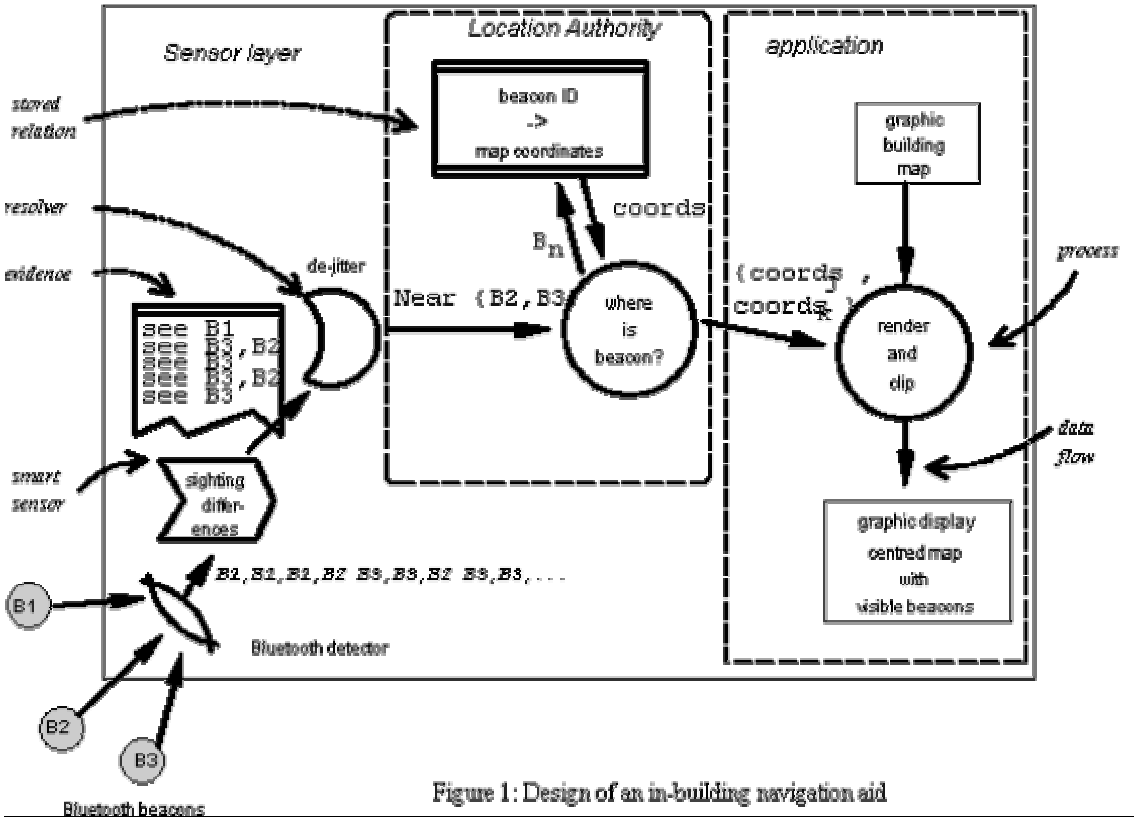

Fig 2. Design notation for sensors, evidence, resolvers