Modelling and Using Sensed Context Information in the Design of Interactive Applications

Philip D.Gray[1] and Daniel Salber[2]

[1] Dept of Computer Science, University of Glasgow,
Glasgow G12 8QQ, Scotland
pdg@dcs.gla.ac.uk
[2] IBM T.J. Watson Research Center, 30 Saw Mill River Road
Hawthorne, NY 10532, USA
salber@acm.org

Abstract. We present a way of analyzing sensed context information formulated to help in the generation, documentation and assessment of the designs of context-aware applications. Starting with a model of sensed context that accounts for the particular characteristics of sensing, we develop a method for expressing requirements for sensed context information in terms of relevant quality attributes plus properties of the sensors that supply the information. We demonstrate on an example how this approach permits the systematic exploration of the design space of context sensing along dimensions pertinent to software development. Returning to our model of sensed context, we examine how it can be supported by a modular software architecture for context sensing that promotes separation between context sensing, user interaction, and application concerns.

1 Introduction

Until very recently, most applications were used in a static setting using little more than user input to define what could be done and to drive the interaction forward. This situation has been transformed by the explosion of portable machines, embedded computation, wireless communications, distributed networks and cheap, plentiful sensors.

Hardware and software resources are running ahead of design and engineering models, tools and architectures. While it is easy to envision endless uses of context in interesting new applications, it is much harder to identify the issues involved in designing systems that use information sensed from the environment and also hard to incorporate sensors into applications. There is little support for systematic exploration of the design space of a context-aware application and for the evaluation of the consequences of design choices on architecture and implementation.

In this paper, we (1) propose a model of sensed context information that accounts for the complexity of sensing, as opposed to traditional user input; (2) present an approach to the design of context-aware applications that deals explicitly with

properties of sensed context; and (3) introduce a preliminary software architecture model that captures typical operations on sensed context and its properties. In section 2, we propose a definition of sensed context and analyze its constituents. Section 3 outlines our design approach and illustrates with an example the systematic exploration of a design space that it supports. In section 4, we propose a software architecture model for sensed context information. We illustrate its use with the example of section 3. Section 5 examines some related work, particularly in software architecture. We conclude and outline plans for future work in section 6.

2 A Model of Sensed Context

Context sensing in interactive applications refers to the acquisition of information from the surrounding environment. We first define sensed context in terms of properties of real world phenomena. We then analyze the features of sensed context and propose a model that captures its most relevant characteristics.

2.1 Defining Sensed Context

We are interested in the sort of information that:

- can be accessed via sensors,
- capture properties of real-world phenomena, and
- can be used to offer application functionality or to modify existing functionality to make it more effective or usable.

In each case the information is sensed from the physical context in which the application is being used. This is part of the overall context of use, which can also include information that is not sensed (typically, the user's emotional state, the social organisation of the artifacts, etc.). To reflect this relationship, we make a distinction between context and that subset of it that is capable of being sensed, viz., sensed context.

The term 'context' suffers from an embarrassing richness of alternative definitions. Dey, Salber and Abowd [1] provides a useful review and offers a version that is a useful starting point for our definition of sensed context:

context $=_{def}$ any information that can be used to characterize the situation of an entity, where an entity is a person, place, or object that is considered relevant to the interaction between a user and an application, including the user and the application themselves. Context is typically the location, identity and state of people, groups and computational and physical objects.

This definition needs some refinement to capture our notion of sensed context. Sensed context refers to that part of context that comes from the physical environment; i.e., that part of context that is accessible via sensors, in other words, the properties of phenomena. The term 'phenomenon' refers to "an occurrence, a circumstance or a fact that is perceptible by the senses." [2] This term comes close to expressing that set of things we wish to include as the subjects of sensed context information if we take the "senses" to include non-human sensing devices.

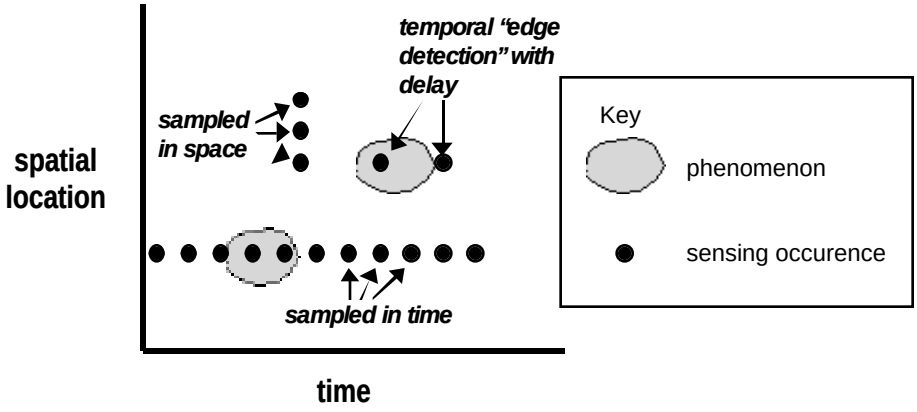

Fig. 1. Relationships of phenomena and sensing activities in space-time. The sensing activity at the bottom is performed by a fixed sensor that provides samples at regular intervals. The phenomenon at the top is sensed twice, but there's a delay in detecting the beginning and the end of the phenomenon, for instance because sensing occurs at fixed intervals.

The relationship of sensors to phenomena can be related in space-time as shown in figure 1. That is, phenomena "occupy" spatial and temporal locations. Often, occurrences of sensing provide "samples" of the phenomena or special events, such as the boundary of the phenomena in space or time. Many of the interesting design and implementation questions about sensed context relate to the sensor-phenomenon relationship, including the timeliness and accuracy of the sensed information and whether identity of entities across phenomena can be determined.

The notion of 'interaction' in the original definition is ambiguous; it is not clear whether this refers to what is achieved via the interaction (viz., tasks) or the means by which the tasks are performed (viz., concrete user interface, dialogue, etc). We wish to be non-restrictive and thus explicitly include both as legitimate aspects of human-computer interaction that sensed context may influence. Finally, it may be difficult or impossible to determine at a particular time if some sensed context is or is not actually relevant. What is important is its *potential* relevance to the interaction, whether or not someone considers it to be such.

Therefore, we propose the following definition:

sensed context $=_{def}$ properties that characterize a phenomenon, are sensed and that are potentially relevant to the tasks supported by an application and/or the means by which those tasks are performed

Sensed context is ultimately *derivable* from some sensory apparatus, or sensor(s). It doesn't follow, however, that it *is* so derived. The fact that Daniel speaks French may be acquired via some sensor (e.g., a multi-lingual speech recognition system) or via a database query, with no sensing involved. Also, sensed context may be derived from other sensed context via some transformation or interpretation. In the case of Daniel's speaking French, the input from the sensor (microphone) may have to go

through sophisticated processing; nevertheless, we wish to refer to Daniel's speaking French as sensed context if the source involved one or more sensors.

Note that the same physical apparatus might serve to provide conventional input and sensed context. For example, a keyboard might provide input strings for processing by the application (user input) and the same input events could be a sensor for user fatigue, based on the average time between keystrokes. Similarly, a sensor can be exploited by a user as an input device. Consider the case of two persons who have agreed that moving past a proximity sensor at their office door will be used as a signal to the other that it's time for lunch. The difference lies in the relationship to user's intentions, conscious or otherwise, not the way the information is subsequently handled by the system.

It can be difficult at times to draw a distinction between sensed context and user-generated input. Nevertheless, it can still often be useful in thinking about the impact on the user of different design alternatives; choices of user input versus independently sensed context can be critical to the feasibility and usability of certain application types.

2.2 Modelling Sensed Context

According to our definition, sensed context is information and we therefore model it as such. The particular aspects captured in the model are there because we believe them to be important for the purposes of reasoning about designs and, in some cases, for implementing them.

Sensed context is propositional in nature, typically of the form "phenomenon P has property p", e.g., "this device is at location y", "this probe has temperature t", "the time is t". Relational information is also possible, such as "person p is near landmark l", "group g is meeting in room r at time t". Being propositional, sensed context information can be formulated using a formal representation such as first-order predicate logic, composed into more complex sensed context expressions and have associated with them meta-propositional properties, such as judgements of the quality of the information (e.g., its probability) or the nature of its sensory source (e.g., the operational parameters of the sensor). In this paper we will not pursue the formalization of sensed context information, but will focus on the associated meta-information.

To summarize, sensed context has several important characteristics that we will utilize in our model, including

- information content, especially
  - the sensed properties of the phenomena
  - the subject(s) of the sensing
- meta-information, including
  - information quality attributes
  - information about the source of the content

We now discuss each of these characteristics in turn.

2.2.1 Information Content: Sensed Context Types

Although we place no limits on the type of properties or relationships that can be expressed in sensed context, certain types are particularly prominent because

- the information offers real benefits in functionality and usability and
- current sensor technology and interpretation techniques enable the information to be captured and transmitted cost-effectively.

These information types include:

spatial location

We include here propositions about physical location, such as the location of a person or artifact in a building or the position of a car in a road system. It may include location in terms of global or local reference systems and absolute or relative position. Abstract spaces may also be spatially located and thus, indirectly, treated as sensed context, but only if they are marked out in physical space.

time

Time may include interval and point references. It is relatively easy to acquire and is useful in association with other properties, as a time stamp.

identity

Typically, identification is performed by a sensor capturing distinctive features of the entity to be identified (e.g., iris configuration, a facial color patch, or the id of an associated "tag"). Often, it is sufficient to be able to assert that different sensing occurrences relate to the same entity; that is, sensed context enables assertions of identity of entities across time and space.

2.2.2 Information Content: The Subject of Sensed Context

If one takes sensed context to be propositional, then there must be a subject or subjects for the propositions. That is, sensed context is not just a property such as location, but an assertion that something is at a location. It may be the case that a given sensor simply delivers a location value, but the data becomes useful information when implicitly interpreted as the location of the device. Typically, it is not the location of the sensor that is of interest, but of something else (e.g., the person holding the GPS), but the further inference from "This GPS is at location x" to "Person P is at location y" depends on the initial interpretation of the data. During design it can be important to identify both the ultimate subject of the information content from the application's point of view and the information provided by the sensor(s), so that the transformational requirements are clear.

2.2.3 Properties of Sensed Context: Meta-Information

One of the distinguishing features of sensed context information, compared to other sorts of information utilised in an application, is the importance of the way it is acquired. Sensors are subject to failure and noise. Often, they only capture samples of phenomena, hence their output is approximate only. Furthermore the process of interpretation of sensed context is also subject to ambiguity and approximation. In addition, sensors may be used in conjunction with actuators that perform actions in the physical world.

These aspects are sufficiently important that we propose as a central feature of our model a set of meta-attributes of the information, including:

- forms of representation
- information quality
- sensory source
- interpretation (data transformations)
- actuation

As shall be discussed in section 3 below, this meta-information is crucial when reasoning about sensed context during design and can be valuable as a potentially controllable aspect of information flows in the run-time system.

We shall look at each of these categories in turn.

Forms of Representation. Many sensing devices, such as GPS and timing devices, are capable of offering their output in different data formats. Therefore, we wish to capture this potential as a meta-property of the information itself, so that it can be described, and reasoned about, even when the sensory source is not yet specified or even known. Furthermore, transformations can serve to change the form of representation of a property, for instance transforming GPS data into building names. Expressing the desired form of representation helps identify the transformations required.

Information Quality. We can identify the following information quality attributes:

- coverage – the amount of the potentially sensed context about which information is delivered
- resolution – smallest perceivable element
- accuracy – range in terms of a measure of the property
- repeatability – stability of measure over time
- frequency – sample rate; the temporal equivalent of resolution
- timeliness – range of the measure in time; the range of error in terms of the time of some phenomena; the temporal equivalent of accuracy

These properties are perhaps easiest to consider in the case of spatial location. For example, a piece of sensed context might provide its information in terms of British Ordnance Survey coordinate system, covering mainland Britain, to a resolution of 100m and an accuracy of +-10%. Similarly, if the information is also temporal, we can identify its form of representation (seconds), its frequency (5 kHz) and its accuracy (or timeliness) (±100 ms).

We believe these are well-defined attributes of quality of all, or at least many interesting, properties of sensed context. Thus, we might have a way of determining the language of a speaker from his or her speech. We can describe information quality attributes for this sensed information:

coverage: are all languages recognized? If not, what is the subset?

resolution: can a distinction be made between similar languages? Thus, if the system cannot resolve the difference between various Slavic languages (Russian, Polish), this is a resolution issue.

accuracy: in this case, it may be difficult to distinguish between the effect of resolution and accuracy problems. However, the distinction can be drawn: accuracy refers in this case to the probability that the system will identify the wrong language. This is different from it's not being able to distinguish between them.

repeatability: given the exact same input, e.g., a recorded utterance, is the determined language the same over successive trials?

frequency might be at the level of a single utterance, with *timeliness* measured as a delay of up to 5 seconds from the end of the utterance.

Sensory Source. So far we have focussed on the nature of sensed context as information. However, it is often necessary to think about where that information comes from.

Among the attributes of the acquisition process by which the information is acquired, we identify the following as candidates for our model:

- **reliability**
- **intrusiveness**
- **security or privacy**

Just as with sensed context, sensors can be characterized by meta-information about the apparatus. The set of properties of a sensor are difficult to fix globally. They are typically closely related to the physical, behavioral and operational characteristics of the sensor. However, we would expect most sensors to at least have the following:

- **cost**
- **operating profile**

Transformation. We may wish to specify the transformation process by which sensor output is transformed into usable sensed context information. At the design stage it may be useful to identify that a transformation is unreliable or computationally costly. A specification of the data transformations is of obvious benefit at the implementation stage.

Actuation. Although context-awareness at present is more concerned with acquiring and deriving information from sensed context, we need to keep in mind the possibility of acting on the real world through effectors. Actuation can also influence the sensing processes by shutting down a faulty sensor, or modifying its operating parameters, for example by reorienting a GPS antenna.

3 Supporting the Exploration of Sensed Context Design Options

Understanding the nature of sensed context and the particular role of meta-data allows us to proceed further and examine how this new understanding might be used for application design and development. In particular, we are interested in considering how these requirements might be used to support systematic exploration of the design space of a context-aware application. What follows is an initial and tentative investigation of how to utilise our model. This section is intended to be illustrative of what might be done.

Our approach relies on checklists that are intended to uncover design choices and the consequences these might have in terms of software development. We will use the example of a museum context-aware tour guide throughout this section. Besides representing one of the most common types of context-aware application reported upon in the literature, the application features some clear, simple requirements and a rich design space. It is not our intention to present a solution to the design problem, but simply to indicate the kind of issues that we believe our approach can uncover.

As shall become evident as we work through this example, the design space is very rich, with many trade-offs requiring more exploration than we can manage in a

relatively short paper. Consequently, we have focussed on a rather restricted view of the design process and its legitimate areas of concern. In particular, the example set out in section 3.1 below focuses on design issues related to the provision of (some) application functionality with no consideration of the wider contextual framework in which such a functional requirement might arise nor the context-related issues associated with the decisions about the method of delivery of that functionality. This may seem ironic in a paper devoted to a consideration of sensed context, but we believe this is a reasonable simplification of the design process, at least for an early investigation such as we are reporting here.

Activities Involved in the Approach. Our approach is based on a set of design-oriented activities that together enable issues of sensed context to be identified, related to other design considerations and explored. The activities include:

- identifying sensed context possibilities
- eliciting and assessing information quality requirements
- eliciting and assessing requirements of the acquisition process
- consideration of issues of
  - intrusiveness, security, privacy
  - transformations of the data from source to "consumer"
  - transmission and storage
- eliciting and assessing sensor requirements

The sequence of these activities in our list suggests a very rough order in which they might be addressed, but the order is not intended to be restrictive. One might well begin with any of the activities and work out to others, returning thereafter to re-assess earlier decisions.

The outcome of these activities will be a set of requirements related to sensed context and a set of design issues related to those requirements. At this stage of development our approach does not offer any particular assistance with how to resolve the set of issues raised. These have to be handled as any set of design problems that have alternative solutions and demand resolution of conflicting requirements.

3.1 Identifying Sensed Context Possibilities

We now consider our example of a context-aware museum tour guide. For the sake of brevity, we will concentrate on a single but representative feature, namely:

Deliver information in the language of the visitor that describes the exhibit that the visitor is attending to.

We will assume that the exhibits are laid out in a single exhibition space and that each exhibit has associated with it a set of descriptions, each with the same content but expressed in different languages. The set of languages supported is some subset of all natural languages. Visitors are people that wander around the museum (i.e., they are mobile), following a path from exhibit to exhibit, reading descriptions about the exhibit they are currently attending to.

Note that at this level of description, we are making no assumptions about the choice of input and output devices that are to be employed. Similarly, we are not specifying what specific sensors or sensing techniques we might use. Before we

identify the sensed context involved in this case, we may make assumptions (a visitor is attending to only one exhibit at a time) or impose constraints (users should not have to carry devices). We will not introduce any design constraints in this exercise because we want to show how we can explore the space generally, although typically constraints identified at this point can limit the space and make the design process more manageable.

In this example the primary entities of interest are: visitor, exhibit. For the *visitor*, we are interested in her language and for the exhibit, the description(s) that are available. Also, we are interested in the relation of *attending to* between the visitor and the exhibit[1] she is attending to. We now have three information needs and can ask of each if it is potentially sensed:

- *visitor's language* – we can envisage using speech recognition to identify the speaker's language (assuming visitors are talking amongst themselves, or are asked to speak to a device at the beginning of their visit) or via a tag attached to the visitor and readable by some sensor, either at an exhibit or elsewhere.
- *exhibit's description* – this is directly sensible, e.g., via an infrared broadcast or indirectly sensible via a transmitted context identifier that can be used for content lookup
- *visitor's attention* – this is potentially sensed in a number of ways, including proximity sensing, orientation capture and eye gaze tracking, using sensors on the visitor, the exhibit or globally in the exhibition gallery.

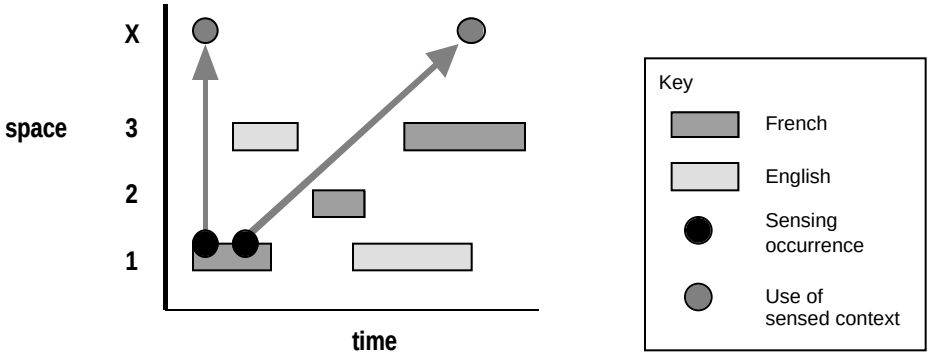

Fig. 2. Time-space diagram of phenomena (visitors speaking English or French attending to exhibits), sensing occurrences, and use of sensed context by the museum tour application. Locations labeled 1, 2, and 3 are locations of exhibits. In this example, sensed context is used by a centralized application running on a server whose location (X) is represented at the top of the diagram.

Once the primary entities and related information of interest are identified, additional characteristics of the potential context sensing can be investigated. Two characteristics are particularly useful: spatial and temporal footprints. Interestingly,

[1] The relationship might be expressed from the point of view of the visitor (the exhibit I am attending to) or the exhibit (the visitor attending to me). We shall refer to it from either point of view as appropriate.

these characteristics are easily captured if we reason in terms of phenomena and use the graphical representation of figure 2, showing three exhibits and two languages. Notice that the diagram captures the language need in relation to exhibit and time; the precondition of the visitor's attention is abstracted away. The diagram makes clear that it is the linguistic need that is central; the user's attention to the exhibit is just an enabling pre-condition. Additionally, it is evident that the identity of the user is not important in this case, a condition that might have been overlooked if focusing on the possibility of sensing the visitor's location and language.

The diagram also makes apparent the potential communications and storage requirements for sensed context. We have placed on the space-time diagram possible context sensing events, along with context use by the application. For each sensed context information that we want to acquire, if there is a spatial gap between the sensing event and the actual use of the information acquired, the information will have to be communicated. If there is a temporal gap, the context information will need to be stored.

3.2 Sensed Context Requirements Checklist

Context-aware application design must factor in the variable quality of sensed context data. One necessary step is to express information quality requirements for sensed context information needed to support the application functions.

Our approach relies on running sensed context information (that is, each predicate over subjects and features) through a checklist of the information quality criteria of section 2.2.3. Each criterion must be considered in turn for each piece of sensed context

As an example, we will consider two of three potential sensed phenomena: the visitor's language and the visitor's attending to an exhibit:[2]

Table 1. Information quality criteria for context proposition "visitor's language is X".

Coverage	All languages supported by the application
Resolution	Language (a lesser requirement would be: linguistic family)
Accuracy	100%
Repeatability	Stable (same language for successive inputs of a pre-recorded utterance)
Timeliness	Before first information delivery
Frequency	• Once, assuming (1) visitor won't switch languages during the visit and (2) we can attach the language info to the visitor and retrieve the visitor's identity from one display to the next • Else, once for each information delivery

[2] The third possibility, sensing the exhibit description, is a reasonable approach. One might, for example, have the exhibit broadcast all its descriptions, and have the visitor's hand-held display receive only the one in the appropriate language (the "teletext" strategy). We have decided to leave this option out of consideration solely for reasons of simplicity in our example.

Table 2. Information quality criteria for context proposition "visitor is attending exhibit Y".

Coverage	All exhibits
Resolution	A single exhibit
Accuracy	100%
Timeliness	Before visitor moves on (on the order of seconds)
Repeatability	100% stable
Frequency	Once per exhibit within timeliness requirements (on the order of seconds)

Notice that several design issues are identified by this exercise. We may have to be clearer about the relationship of languages to dialects. More importantly, since we have a quality requirement that only one language needs to be sensed per visitor (per visit), we can entertain the possibility of a single sensing occurrence in order to capture this information (although we may then need to store the information for later use).

The next two activities, assessing sensor requirements, transmission and storage needs are all closely linked to particular acquisition strategies or patterns. The following seem to be reasonable candidates for sensing strategies in this example:

- Determining the language of the visitor
  - Looking it up (preferences)
  - Asking the visitor
  - Associating a sensible tag with the visitor that contains a language id
  - Listening to speech
- Visitor is attending an exhibit
  - Near it (but might fail if user is turning her back to the display) -> Proximity
  - Facing it -> Proximity + relative orientation
  - Looking at it -> Gaze
  - Asking the visitor
  - Based on history

Arising out of this exploration are a number of consequent design issues. In the case of proximity sensing (location) there are related issues of whether the user senses the display or vice versa; in the former case, we are confronted with a question of power consumption and possible privacy issues if the identity of the visitor is also made available (perhaps via the sensed tag strategy of language sensing. Considering proximity-based location as a source of attention, it becomes apparent that proximity is not sufficient if one can be attending to two different exhibits from the same position or if the proximity sensors necessarily have overlapping coverage. If there is no overlap, proximity sensing is probably acceptable; if there is overlap, then it must be either replaced or enhanced by other means. Table 3 captures the most significant design issues for the two pieces of sensed context.

Table 3. Combined assessment of sensing strategies. This table summarizes the issues that must be addressed. Items in bold denote major issues.

item	*information quality*	*acquisition process*	*sensors*
language			
sensed tag	accuracy & repeatability	need stored language data	**if RFID tags: orientation of tag wrt. reader; RF interference**
listening to speech	**coverage & accuracy**	need multi-lingual recognition; assumes user is talking	**ambient noise**
ask user		**intrusive**	n/a
lookup		need stored language data	n/a
attending to			
proximity	**accuracy if overlaps between exhibits**	**privacy if linked to identity**	locus of sensing; **power if mobile**
orientation	accuracy	**not sufficient on its own; needs proximity**	
gaze	accuracy		**awkward headgear**

Checklists, or structured questions, of the sort we have presented here are, we believe, a step in advance of the unstructured, ad hoc approach typical now. It is a form of literate specification [3], helping to identify issues, make issue coverage visible, assist in the documentation of the design process and offer a means for traceability of design decisions. While we believe our small example suggests the technique is both feasible and useful, it remains to be evaluated in a realistic setting.

4 Mapping Sensed Context to a Software Architecture

Software modelling of sensed context within a context-aware application must take into account the complex nature of context information. The salient characteristics of sensed context information outlined in section 2 suggest a key distinction that we want to capture in the software model: both the actual information content of sensed context and meta-information that characterizes sensed context should be handled explicitly and distinctly. In addition, section 2 introduced the need for controlling sensors and sensing processes. Finally, we want to capture the ability not only to sense context, but also to act on the real world through actuators. Our software model for context-aware applications aims at emphasizing features that are specific to sensed context and that should be given particular attention. Furthermore, the model we

propose focuses on sensed context: we do not explicitly model traditional user interaction that may occur in the application. Other models exist for that purpose, and they may be used in conjunction with our model. We will show an example of dual use of our model along with existing user interface models.

4.1. Requirements for a sensed context architecture

Context sensing is still very much an exploratory domain. Developing efficient context sensing capabilities requires practical knowledge of sensors and associated software techniques, which might build upon disciplines as varied as signal processing or machine learning. The set of skills required is far different from that used to develop user interfaces or application logic. As a result, a paramount requirement of a context sensing architecture is to facilitate the separation of concerns between user interface, application logic, and context sensing. In addition, we want to support iterative design (typical of an exploratory domain), and thus emphasize modularity.

4.2 Global view of the architecture

To achieve separation of concerns, we introduce a set of functional components that focus on bridging the gap between sensors and applications. In an architecture model like Arch [4], this set of components is organized in two layers that correspond to two primary classes of operations on context: acquisition from the physical world, usually through sensors, and transformations of sensed information to extract information meaningful to the application. These operations occur at two different levels and can be represented as a supplementary branch in the Arch model as shown in figure 3. Arrows represent flow of control, data, and meta-data. This point will be clarified when we introduce the components that constitute the two context layers.

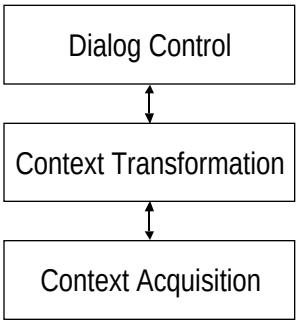

Fig. 3. Context handling as a supplementary branch in the Arch model. The classic user interface and application logic branches have been omitted for clarity.

This approach clearly emphasizes separation of concerns between the three branches of the extended Arch. This however, does not preclude the possibility that

context-handling components may have user interfaces to allow control by or feedback to the user. A major difference with the traditional Arch model should be mentioned. As noted in Salber *et al.* [5], context components are long-lived and typically precede and survive the application they serve. Indeed, context may be required at any time by an application and the relevant context data may have been acquired long before the application requests it. Consider a handheld application that helps the user remember where she last parked her car: The location of the car when the user steps out must be recorded even though the retrieval application is not running, and maybe never will. Thus, the context transformation and acquisition layers may be active independently of any application. Connections between the Dialog Control component and the context branch are established dynamically. In addition, several applications may require the same context information, possibly simultaneously. The handheld car park payment application may be notified that the car is in a parking meter area and propose to the user to negotiate a fair price. In this sense, context services are similar in principle to operating system services such as network demons, or low-level windowing facilities.

4.3 Context handling components

Using sensors, as opposed to user input devices, entails dealing with a lot of data. Sensed context is much richer and data-intensive than user input. Thus, our model should primarily be concerned with organizing the flow of data. The model we have presented in section 2 provides an important insight into a distinguishing characteristic of sensed context. Meta-information that describes the quality and source of context data should be considered as important as the actual sensed information. In addition, context transformation mechanisms may be controllable and transformations may result in actions on the environment. We want to reflect in our software architecture the three flows we have identified: data, meta-data and control.

We populate each of the two layers identified in the previous paragraph with modular, composable, context-handling components. They all share the same structure represented as an hexagon shown in figure 4. Each lower side of the hexagon represents inputs. Each of the three sides receives one of the three flows of data, meta-data and control. Upper sides of the hexagon represent outputs and generate data and meta-data flows, along with actions on the environment. For a given component, any input or output arrow might be omitted if it is not relevant. Conversely, a component may receive multiple input flows on any one of its lower sides, or generate multiple outputs.

Each context-handling component performs transformations on context data. Although we have isolated the acquisition layer in the global view of the model, it is populated with components that share the common structure of figure 4. The only difference is that an acquisition component acquires its inputs from sensors (or from a sensing component part of a library, when they become available). In the generic model of the context-handling component, the data input flow is processed and produces the data output flow. Meta-data input may be used to influence processing (e.g., discard inaccurate data). New meta-data that describes the data resulting of the transformation is generated. Depending on the transformation involved, this may

consist in updating the meta-data, or generating completely new meta-data. The control input allows other components to influence the transformation process itself. This may consist in modifying a parameter (e.g., the sampling rate of a sensor), requesting generation of the latest data (i.e., polling), starting or shutting down the component (e.g., shutting down a faulty sensor). The action output is the channel through which actions on the environment are performed. This may include turning on a light, changing the speed of a motor, driving a camera that tracks a person in a room, etc. Tight coupling between inputs and the action output is possible at the level of a single or a few components.

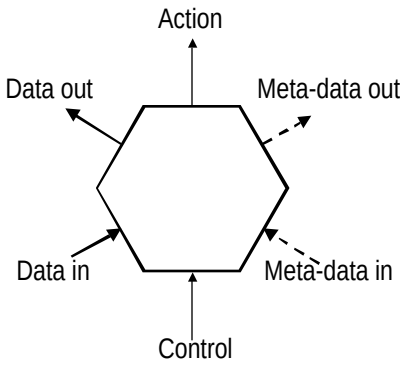

Fig. 4. The structure of a context-handling component.

The output of any context-handling components may be connected to the input of any other component. In general, information content outputs will be connected to the information input of another components, and similarly for meta-information and control/action. However, there may be connections from the meta-information input of a component to a second component's information input. The second component would then monitor the quality of the information provided by the first component, and possibly provide meta-information about its own monitoring function. We will give an example of this configuration in the guided tour example. In general though, control and action flows are kept separate from information content and meta-information flows. Control is also used for an important role: In a changing environment where sensors may appear and disappear, control drives the dynamic reconfiguration of the organization of components, establishing and breaking connections between a component and its peers. This issue is still being investigated but we believe it is crucial for handling context in a comprehensive way, especially when devices or sensors are mobile. Ideally, dynamic configuration, as well as transparent networking (including dynamic adaptation to the networking resources available) and other services such as service discovery and brokering between the application needs and the capabilities of the sensed context components should be handled by a middleware infrastructure, acting behind the scenes of our architecture. We know these services are required, but they fall outside the scope of our present concerns for this architecture model. We assume they are provided by the target

platform. Ultimately, we aim at eliciting precise requirements for such a platform. This goal is beyond the scope of this paper.

4.4 Example use of the architecture for the museum tour application

We have chosen to show how to use the architecture on a subset of the museum tour guide: the determination of the language of the visitor. Assessing the design issues of table 3, we use two complementary sensors in this example: speech recognition and RFID tag readers. Since each of these techniques has shortcomings under certain conditions, we combine the information they provide for better reliability.

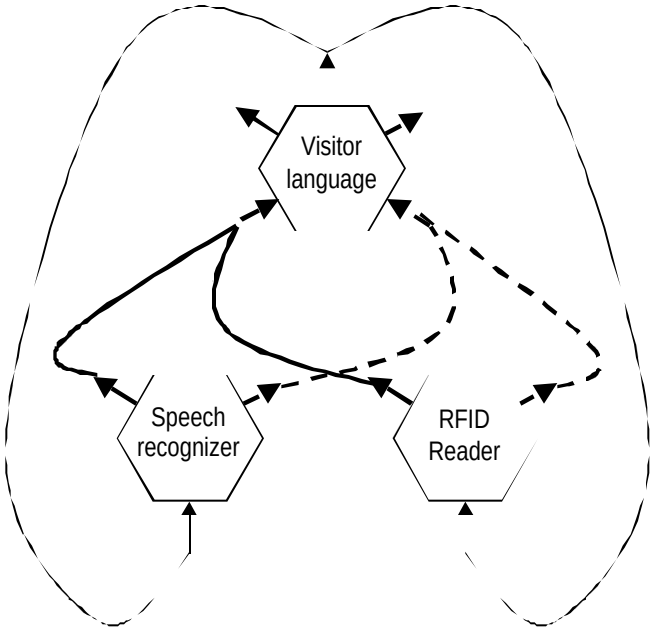

Fig. 5. Components involved in determining the visitor language in the museum tour guide application. Solid bold lines indicate sensed context information flow. Dashed lines indicate meta-information. Fine lines indicate control flow.

The two input sensors are modeled by a context-handling component as shown in figure 5. A third component is in charge of determining the visitor language and combines the information content based on the meta-information provided, which might include indications of failures to read a tag, and confidence factors from the speech recognizer. Based on the meta-information, the combining component might decide to shutdown or reconfigure one of the two sensors. This is expressed by the "Action" output being fed to the "Control" input of the sensor components. The

processes involved in selecting and combining the information content from the sensor components should of course be completely specified in a complete example.

We believe that the organization of components presented here is a potential candidate for a pattern of context sensing: combining information from several unreliable sensors and allowing on-the-fly reconfiguration of the low-level sensing components. Other potential pattern candidates include for instance the management of collections of context information or the monitoring of a single sensor by a dedicated monitoring component. We believe this is a promising area of research that our proposed architecture can support.

5 Related Work

In the field of context-aware computing, there is little work yet that aims at modelling sensors to takes into account the fact that sensed context information acquired from sensors is often of variable quality. For user input, Card, McKinlay and Robertson have devised a comprehensive model of input devices that shares interesting similarities with our approach [6]. They also felt the need to model device properties such as resolution to help designers make their requirements explicit. There have been several recent proposals of software architecture models for context-aware applications. Winograd, for example, presents a data-flow architecture based on networks of observers that abstract information from sensors [7]. Observers construct "context models", data structures that are stored in a "manager" blackboard and made available to applications. There are two interesting differences with our approach. First, Winograd's approach makes explicit the needs of applications: They provide the context model structure that observers populate. Second, the architecture doesn't account for ambiguous context information. Kiciman and Fox have a data-flow approach that introduces mediators which establish dynamic connections between components [8]. The context toolkit framework by Salber, Dey and Abowd was the basis of the architecture model presented here [5]. The context toolkit assigns specific roles to components, e.g., interpreters transform data, aggregators act as repositories. We have generalized the approach of the context toolkit by introducing a generic context-handling component that can be instantiated to play different roles. This approach is, we believe, more extensible (new classes of components can be defined) and allows for encapsulation of well-understood behaviors in a single component. But most importantly, the approach presented here is more expressive, in the sense that it captures meta-information that describes the quality of sensed context. Recent work by Dey, Mankoff and Abowd extends the context toolkit to allow manual disambiguation by the user of imprecise context data [9].

6 Conclusions and Future Work

We have presented a comprehensive view of how sensed context might be handled in interactive systems. In particular, we have presented a model for sensed context that accounts for the inherent uncertainty of data acquired through sensors or derived

by subsequent transformations. We have examined how this model can be used to explore relevant dimensions of the design space of a context-aware application. Finally, we have introduced a proposal that incorporates the characteristics of sensed context information in terms of a software architecture model.

The model of sensed context we have described in this paper still needs refinement. We have exercised it on small examples but we need to better assess its value and its shortcomings. We plan to use it for the design of two real context-aware applications in collaboration with designers. We expect these real-world design experiences to provide some insights on how to better structure our exploration of the design space. We also anticipate to be able to address other questions, such as appropriate ways to describe aspects of interactive system functionality relevant to the use of sensed context (e.g., using UML). Another expected result is a set of requirements for tools to support design, and documentation of the design process. A first step toward this goal will be to express our model as an XML DTD so that models of sensed context information can be used by software tools.

References

1. Dey, A.K., Salber, D., Abowd, G.D.: A Conceptual Framework and Toolkit for Supporting the Rapid Prototyping of Context-Aware Applications. Human-Computer Interaction (to appear) (2001)
2. The American Heritage Dictionary of the English Language, 3rd ed. Houghton Mifflin Company, Boston (1992)
3. Cockton, G., Clarke, S., Gray, P., Johnson, C.: Literate Development: Weaving Human Context into Design Specifications. In: Benyon, D., Palanque, P. (eds.): Critical Issues in User Interface Systems Engineering. Lecture Notes in Computer Science, Vol. 1927. Springer-Verlag, Berlin Heidelberg New York (1996) 227-248
4. Bass, L., Little, R., Pellegrino, R., Reed, S., Seacord, R., Sheppard, S., Szczur, M.R.: The UIMS Tool Developers' Workshop: A Metamodel for the Runtime Architecture of an Interactive System. SIGCHI Bulletin 24 (1992) 32-37
5. Salber, D., Dey, A.K., Abowd, G.D.: The Context Toolkit: Aiding the Development of Context-Enabled Applications. In: Williams, M.G., Altom, M.W., Ehrlich, K., Newman, W. (eds.): Proceedings of the ACM SIGCHI Conference on Human Factors in Computing Systems. ACM Press, New York (1999) 434-441
6. Card, S., McKinlay, J., Robertson, G.: The Design Space of Input Devices. In: Chew, J.C., Whiteside, J. (eds.): Proceedings of the ACM Conference on Human Factors in Computing Systems. ACM Press, New York (1990) 117-124
7. Winograd, T.: Towards a Human-Centered Interaction Architecture. http://graphics.stanford.edu/projects/iwork/papers/humcent/ (1999)
8. Kiciman, E., Fox, A.: Using Dynamic Mediation to Integrate COTS Entities in a Ubiquitous Computing Environment. In: Thomas, P., Gellersen, H.-W. (eds.): Proceedings of the Second International Symposium on Handheld and Ubiquitous Computing. Lecture Notes in Computer Science, Vol. 1927. Springer-Verlag, Berlin Heidelberg New York (2000) 211-226
9. Dey, A.K., Mankoff, J., Abowd, G.D.: Distributed Mediation of Imperfectly Sensed Context in Aware Environments. GVU Technical Report No. 00-14 (2000)