Towards a Common Context Model for Virtual Community Applications

J. Indulska[1], K. Henricksen[2], T. McFadden[2], and P. Mascaro[2]
[1] *School of Information Technology and Electrical Engineering, The University of Queensland*
Email: jaga@itee.uq.edu.au
[2] *Distributed Systems Technology Centre*
Email: {kmh,mcfadden,mascaro}@dstc.edu.au *

Abstract. Instrumentation of home appliances and devices with sensors and actuators enables the creation of smart, context-aware environments which can intelligently support users in a variety of tasks, including tasks that support independent living of the elderly. This paper describes how the complexity associated with designing and implementing context-aware applications can be reduced through both context reuse and programming methods that allow context evaluation to be decoupled from applications. The paper describes a family of applications for smart homes that fall into the category of virtual community applications for the elderly, and demonstrates that these applications can be supported by a common, evolvable set of context and preference models.

1 Introduction

Homes are equipped with a variety of appliances and devices, and can be further instrumented with sensors and actuators. This, coupled with a variety of wired and wireless communication technologies, creates a heterogeneous computing and communication infrastructure suitable for enabling the creation of context-aware home environments which can intelligently support users in a variety of tasks. This support may range from hiding heterogeneity of devices and communication technologies (e.g., by providing common and self-explanatory interfaces to entertainment units and appliances) to complex support for independent living of the elderly. The latter may not only include seamless control of sensors, actuators, appliances and other devices, but also a variety of context-aware applications supporting daily activities (and remote help with these activities from family members or the local community), health monitoring, and building of virtual communities [1, 2].

Due to the variety of user activities which should be intelligently supported by a smart home, and the heterogeneity of devices, communication technologies and interaction protocols between devices, most smart home applications require a rich set of context information in order to re-configure and self-adapt when the context or user preferences change. Such

*The work reported in this paper has been funded in part by the Co-operative Centre for Enterprise Distributed Systems Technology (DSTC) through the Australian Federal Government's CRC Programme (Department of Education, Science and Training).

context information may include information about device characteristics, device location, user location, and user activity.

Many challenges in designing smart home environments are akin to those faced when designing pervasive systems in general. Mobility of users, a variety of user activities, heterogeneous communication technologies and heterogeneous devices introduce similar requirements for context-awareness as those seen in other application domains of pervasive systems. Such systems require models for describing and evaluating context, adaptability methods which can be applied in response to context changes (or context-aware applications which can adjust their behaviour when notified about context changes), and appropriate middleware/infrastructure responsible for context gathering, management, evaluation and dissemination [3, 4].

It is commonly acknowledged that the design and implementation of context aware applications are difficult tasks, and that new approaches are required which can support developers of context-aware applications. Context-aware applications often need to evaluate similar context facts and, more often, contextual situations built from context facts. Therefore, techniques that allow reuse of situations have the potential to considerably simplify the development of new applications. It should also be noted that most of the current context-aware applications have context evaluation hardwired into the application logic, which implies that any changes in the context model or user preference model require re-programming of the applications. If context and preference evaluation logic can be separated from the applications it is possible to create evolvable applications which will work with modified types of context information or user preferences.

In this paper, we discuss applications for smart homes that fall into the category of virtual community applications, and demonstrate that these applications can be supported by a common, evolvable set of context and preference models. In defining the common models, we draw on our previously developed modelling techniques [5, 6], which are sufficiently rich to allow us to capture diverse types of context information, including imprecise, incomplete and ambiguous information, and to describe context-dependent user requirements and preferences. We demonstrate benefits that arise from context and preference reuse, which include reduced complexity associated with constructing new virtual community applications, reduced effort required for customisation and fine-tuning of applications, and more consistent behaviour across a user's set of applications.

The structure of the paper is as follows. Section 2 shows example virtual community applications and provides an informal description of the context information needed for such applications. This description provides insight into the potential for context reuse when building such applications. Section 3 introduces a formal common context model for the applications, while Section 4 presents a common preference model. Section 5 briefly describes how our infrastructure for context-aware computing allows evolution of the context and preference models without re-programming applications that rely on these models, and Section 6 concludes the paper.

2 Example virtual community applications

The virtual community applications that we consider have diverse goals, but are primarily concerned with providing support for independent living in communities of elderly people. They include applications that support:

- assistance for the elderly with everyday tasks by remote family or community members;
- dynamic formation of groups of people who are interested in activities such as shopping or playing chess, based on their preferences and availability; and
- awareness of the activities of others in nearby smart homes using abstract visual representations, helping to minimise isolation and provide assurances of others' well-being.

2.1 Selected applications

The following three example applications have been selected for discussion in this paper:

1. Instant communication with a family member, friend or health worker, achieved using context-aware choice of communication channel (telephone, SMS, e-mail, etc.) according to the current activity, devices (and their communication modes) and preferences of communicating persons.
2. Servicing of sensor-based alarms/events by family members, community members or health workers. Any event/alarm which requires external help (e.g., a burnt out light bulb or the elderly person lying on the floor) is evaluated based on the current context and policies/preferences to determine who should be informed. Context-aware communication (described above) is used to deliver information about the event.
3. Organisation of social activities (e.g., shopping, theatre, or a bridge game). When an elderly person wants to organise a social activity, context information is used to determine which people are preferred for the activity, and context-aware communication is used to contact them.

An informal characterisation of the context information required by these three applications is presented in the following subsection.

2.2 Required context information

The simplest of the three applications, application 1, requires context information about family members, health workers, and community members, including information about their activities, locations, and communication devices. Context information about devices should include modes of operation (types of communication channel) and location. The information about the elderly person's communication devices and the callee's activities and devices are the basic context facts used in this application. These basic facts must be supported by additional information about both the elderly person's and the callee's preferences for communication devices and communication modes.

Application 2 is more complex as it requires all of the context information used by application 1 (to support context-aware communication), as well as information about a variety of sensors available in the house, their types and states, and their positions within the house. Context facts (e.g., sensor states) can be used to define more complex contextual situations (expressions built from context facts). One example is an emergency health situation (e.g., person lying on the floor in the kitchen and not moving), which needs to be deduced from the states of several sensors. Such situations need to trigger some help actions, and therefore

the set of preferences defined for application 1 also needs to be extended for application 2 to include preferences about recipients of alarms or notifications about particular situations.

Application 3 requires all of the context information defined for application 1, as well as additional information about people's hobbies and interests and preferred social groups.

2.3 Other applications

The next section presents a formal context model for the described three applications, discussing reuse of context facts and context situations in the applications. Due to space limitations we cannot formally show that this context model can be used for a much larger number of virtual community applications. However, even from the informal model described in the previous subsection, it can be clearly seen that further applications of this type can be based on the same context model (with some small extensions). Additional applications that we have considered include:

- An application providing cooking advice from a family member or health worker. This application may be triggered by the elderly person or the smart home (the latter in the case when a medical evaluation of the cognitive abilities of the elderly shows that the person needs help with particular activities in the kitchen). It is assumed that there is a video camera allowing the helping person to observe activities in the kitchen. When the application is triggered, a person able to help is found based on the current context, and context-aware communication is used to communicate the request.
- An application supporting a distributed activity (e.g., remote bridge game). The application uses context-awareness to adapt to location/device changes of the participants.
- Ambient communication which provides awareness of the activities of others. When some community members are involved in particular popular activities (e.g., walking a dog), others are informed using a mode of delivery (e.g., a visual representation on an "ambient community screen" or a scrolling message at the bottom of a TV) that is appropriate to the current context and the recipient's preferences.

3 Common context model

A common model of context fact types for applications 1-3 is presented in Figure 1. The notation used is discussed in [5], and is an extension of the Object Role Modeling (ORM) notation [7]. The context information has various origins and quality. It can be sensed from physical or logical (software) sensors, derived from other context information or defined by users. It is important that context models address the issue of context imperfection, incompleteness and ambiguity, as this allows developers to create robust and reliable context-aware applications that are able to deal with conflicting context information or temporary lack of sensor data. The presented model captures context facts (e.g., person using device, home contains room) of various types (static, profiled, derived, sensed), and also quality, histories and relationships of context information.

Based on the defined context facts, more complex situations can be described. Both facts and situations can be reused in other applications. Figure 2 shows some example situations from application 1 which are reused in applications 2 and 3. Figure 3 presents examples of

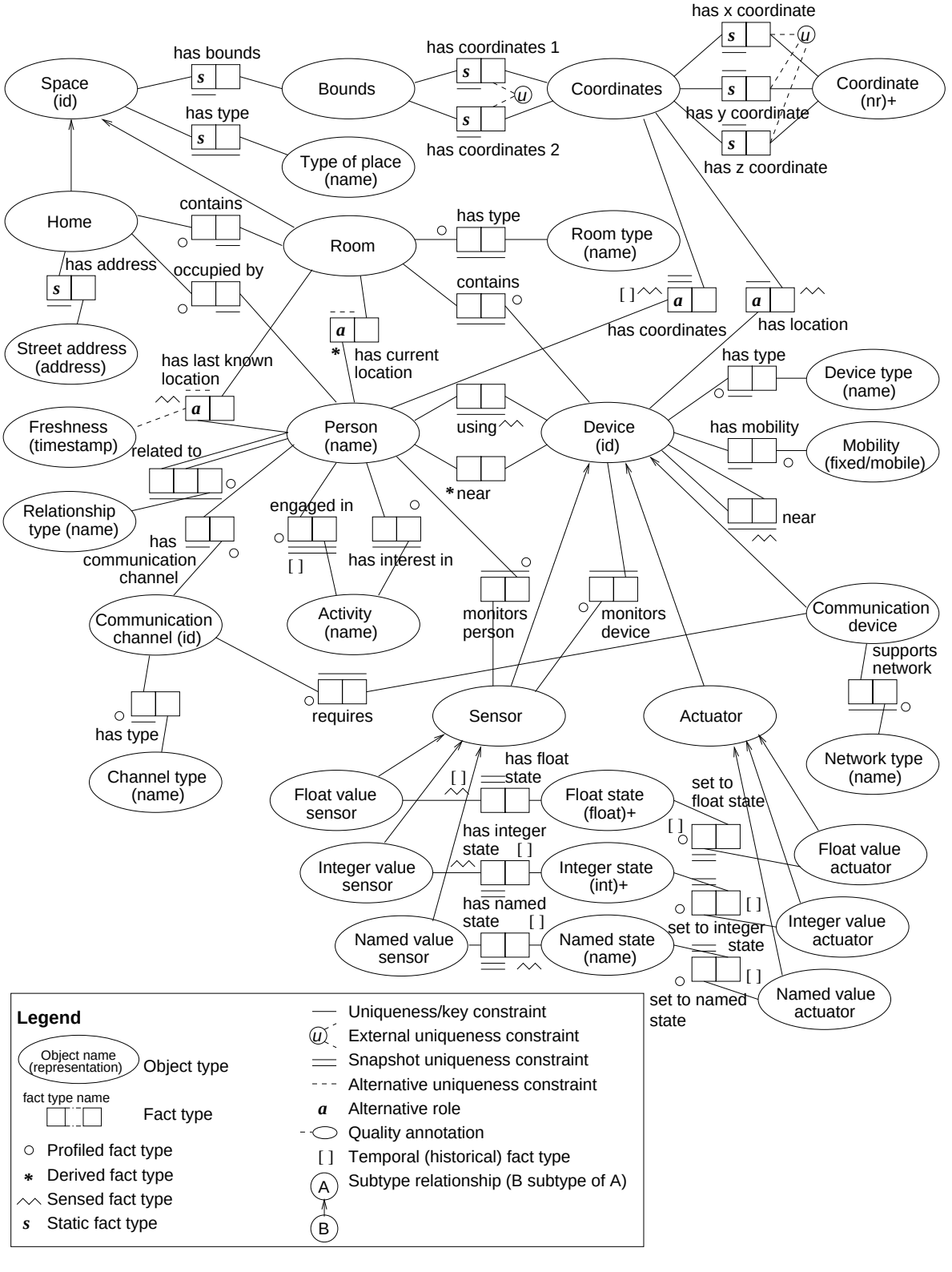

Figure 1: Common context model.

additional situations which are required for application 2. The examples are of two kinds: a situation which should trigger a home maintenance notification (when a light bulb is not working) and a situation which should trigger a health emergency alarm (person lying down

```
• IsChildOf(child, parent):
      PersonRelatedTo[child, parent, 'relation.son'] or
      PersonRelatedTo[child, parent, 'relation.daughter'] or
      PersonRelatedTo[parent, child, 'relation.mother'] or
      PersonRelatedTo[parent, child, 'relation.father']
• CanUseChannel(person, channel):
      forall device
         . CommunicationChannelRequires[channel, device]
         . PersonNear[person, device] and
           not exists sensor
              . SensorMonitorsDevice[sensor, device]
              . SensorHasType[sensor, 'device.status.busy']
                and SensorHasNamedState[sensor, 'device.status.busy.true']
```

Figure 2: Example situations for application 1.

```
• RoomLightBurnedOut(room):
      exists lightSwitch
         . RoomContainsDevice[room, lightSwitch]
         . DeviceHasType[lightSwitch, 'light.switch']
           and ActuatorSetToNamedState[lightSwitch,'light.on']
           and not DeviceHasSensedNamedState(lightSwitch, 'light.on')
• PersonHasFallen(person):
      PersonIsHorizontal(person) and not PersonMoving(person, 5) and
      exists coordinates
         . PersonHasCoordinates[person, coordinates]
         . exists space
              . SpaceHasBounds[space, bounds]
              . BoundsContainsCoordinates(bounds, coordinates) and
                not SpaceHasType[space, 'space.rest.position']
```

Figure 3: Example situations for application 2.

and not moving and not in a typical rest space). The situations `DeviceHasSensedNamedState` (used in the `RoomLightBurnedOut` situation), `PersonIsHorizontal` and `PersonMoving` (used in the `PersonHasFallen` situation) are omitted due to space limitations.

4 Common preference model

The goal of our preference model is to address a need for new techniques for capturing user requirements and preferences in a context-dependent fashion. The model is based on the situation abstraction and enables candidate choices (such as communication channels that can be used for interactions between users in the case of a communication application) to be rated according to their suitability for the current context. Figure 4 presents example preferences for the context-aware communication application, which forbid the use of communication channels when the required devices are not available, and assign higher precedence to telephone than email when the elderly person is contacting one of their children about a matter of high priority. Application 2 can reuse these preferences, but also requires preferences that describe how each type of event should be handled. Some examples are shown in Figure 5 for the `RoomLightBurnedOut` and `PersonHasFallen` situations. Application 3 requires ad-

```
• when not CanUseChannel(callee, channel) or
       not CanUseChannel(caller, channel)
  rate forbid
• when IsHigh(priority) and IsChildOf(callee, caller) and
       IsTelephone(channel)
  rate 1
• when IsHigh(priority) and IsChildOf(callee, caller) and
       IsEmail(channel) and AtWork(callee)
  rate 0.5
```

Figure 4: Example preferences for application 1.

```
• when RoomLightBurnedOut(room) and IsChildOf(contactPerson, occupant)
  rate 1
• when RoomLightBurnedOut(room) and IsNurseOf(contactPerson, occupant)
       and WorkingHours()
  rate 0.8
• when RoomLightBurnedOut(room) and IsNurseOf(contactPerson, occupant)
       and not WorkingHours()
  rate forbid
• when PersonHasFallen(room) and IsNextOfKinFor(contactPerson, occupant)
  rate oblige
• when PersonHasFallen(room) and IsDoctorFor(contactPerson, occupant)
  rate oblige
```

Figure 5: Example preferences for application 2.

ditional preferences about social groups and activities, but these are not shown here because of space limitations.

5 Support for context evolution

Context-aware applications have to be supported by an infrastructure able to gather, process, and evaluate context and preference information. This infrastructure must provide powerful context and preference query capabilities as well as notification mechanisms. We have developed an infrastructure for context-aware applications that meets these requirements, and allows modelling and evaluation of context facts (and their quality, ambiguity, history, and dependencies), situations and user preferences [5]. Context and preference information can be discovered and reused in newly created applications. This support for reuse simplifies the development of context-aware applications and allows for highly flexible behaviour.

Most of the current approaches to construction of context-aware applications assume that context evaluation constitutes a part of the application logic. Applications built in this way must be re-programmed for any changes in the context model (and/or preference model), which means that they cannot gracefully evolve as the context and/or preference model evolves.

The novel programming model supported by our software infrastructure [5] allows insertion of context- and preference-dependent decision points into the normal flow of application

logic. These decision points are implemented as calls to an API which supports context-dependent choice amongst sets of alternatives (e.g., available communication channels). User preference information forms the link between the context and the chosen action(s). Preferences assign ratings to the alternatives according to the context and other application parameters, and, based on these ratings, the application selects and invokes one or more associated actions. This solution is very flexible and enables modification and fine-tuning of context and preferences when required (even at run-time).

6 Conclusions

This paper presented a common context and preference model designed for a family of context-aware, virtual community applications which can support elderly persons. Our modelling approaches allow us to capture a rich and diverse set of context information, including ambiguous and otherwise imperfect information. They also enable reuse of context facts, contextual situations built from facts, and user preferences. Moreover, our approach to programming context-aware applications separates the application from the underlying context and preference information, allowing the latter to be easily evolved over time to support changes in user requirements and environment resources (e.g., addition of new sensors) with minimal impact on the application.

The presented virtual community applications require a rich set of contextual information about a number of persons (the elderly, family members, friends, health workers and community members). Some of these types of context information naturally raise privacy concerns, and, to address these, we are currently developing extensions to our context modelling approach to incorporate ownership information and context-dependent privacy preferences (expressed using the preference model described in this paper).

References

[1] E.D. Mynatt, J. Rowan, S. Craighill, A. Jacobs, Digital family portraits: Providing peace of mind for extended family members, Proc. of the ACM Conference on Human Factors in Computing Systems (CHI 2001), Seattle, ACM Press, 333–340.

[2] M. E. Pollack, L. Brown, D. Colbry, C. E. McCarthy, C. Orosz, B. Peintner, S. Ramakrishnan, I. Tsamardinos, Autominder: An Intelligent Cognitive Orthotic System for People with Memory Impairment, Robotics and Autonomous Systems, **44**(3-4), (2003), 273–282.

[3] Anind K. Dey, Daniel Salber, Gregory D. Abowd, A Conceptual Framework and a Toolkit for Supporting the Rapid Prototyping of Context-Aware Applications, Human-Computer Interaction, **16**(2-4), (2001), 97–166.

[4] Guanling Chen, Ming Li, David Kotz, Design and Implementation of a Large-Scale Context Fusion Network, Proc. of the 1st International Conference on Mobile and Ubiquitous Systems: Networking and Services (Mobiquitous), Boston, (2004).

[5] K. Henricksen, J. Indulska, A Software Engineering Framework for Context-Aware Pervasive Computing, Proc. of the 2nd IEEE International Conference on Pervasive Computing and Communications (PerCom), Orlando, (2004), IEEE Computer Society, 77–86.

[6] K. Henricksen, J. Indulska, A. Rakotonirainy, Modeling Context Information in Pervasive Computing Systems, Proc. of the 1st International Conference on Pervasive Computing (Pervasive), F. Mattern, M. Naghsineh (eds). Lecture Notes in Computer Science, Springer Verlag, LNCS **2414**, (2002), 167–180.

[7] T. A. Halpin, Conceptual Schema and Relational Database Design, Prentice Hall, Sydney, 2nd edition, (1995).