

Models of People, Places and Devices for Location-Aware Services

Mark Assad, Judy Kay, and Bob Kummerfeld *

The University of Sydney, Sydney, Australia

Abstract. The Active Model (AM) is a novel approach to supporting context-aware ubiquitous services. At its core are active models for people, sensors, devices and places. These make it possible to quickly build and augment flexible ubiquitous personalisation services. We use a prototype implementation of a music presentation system to illustrate the active, distributed models and associated resource discovery.

1 Introduction

Context-aware systems aim to “provide relevant information and/or services to the user, where relevancy depends on the user’s task” [1]. So, for example, a context-aware music system would select music to please the people present: if the two people in a room like Mozart, but have no other common music preferences, it should play Mozart. We use this example to illustrate the AM approach, which can support a range of context aware services, like “Place-its” [2], people finders [3, 4], and status information within homes [5]. AM grew from our work on modelling users so they can maintain control over the personalisation [3]. We have generalised this to manage all elements of a ubiquitous environment: sensors, services, devices, rooms and people. This contrasts with work on other systems [1, 6] that focus on managing data from sensors. Like such systems, AM manages context information from such entities, processing and collating it as necessary, but this is totally integrated with the models of the people.

The following overview of AM describes its core elements: an overview of the models; the subscriptions that make them active; their distributed operation; disconnected operation; and the associated resource discovery that enables AM-based systems to find the relevant models.

2 System Description

Our system uses an *accretion/resolution* representation, implemented in the *Personis*¹ user modelling system [3, 7]. This provides a hierarchy of *contexts*, each acting as a namespace for the *components* modelled. Figure 1 shows a set of very simple models, described in detail in the figure caption.

* This research was part funded by the Smart Internet Technology CRC Australia.

¹ Personis Pty Ltd

The user models have two basic operations: **ask** requests the value of a set of components and **tell** sends a piece of evidence to support reasoning about the value of a component.

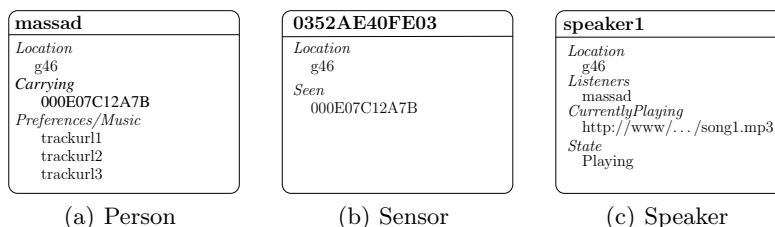


Fig. 1. Example models. Subscriptions on components make them *active*. Fig. 1(a) is for a person *massad* with components for his *Location*, a room called *g46*, what he is *Carrying*, a phone with identifier *000E07C12A7B* and a context *Preferences* which contains his *Music*. Fig. 1(b) shows the model for a sensor: it has just its *Location* and the identifiers for devices it has *Seen*. Fig. 1(c) is a model for the device that plays the music: this reflects its *Location* is also *g46*, the *Listeners* are *massad*, the url of the music it is *Currently Playing* and its *State* is *Playing*.

We now describe the core of our **active** models. When a person enters our music room, the music application needs to be notified so it takes account of their preferences. AM addresses this problem by allowing an application to attach *subscriptions* to components of a model. Subscriptions are written in the form **value** ~ **pattern** : **action** where the **value** is an **ask** operation on a model, the **pattern** is a regular expression and the **action** can be either a **tell** operation or a **notify** specifying a server and a message to send. Figure 2 illustrates the active models, with their subscriptions and associated information flows.

Figure 3 shows the processes driving the selection of music to match the modelled preferences of the people in a room. The models are similar to those in Figures 1 and 2, with the addition of the user *bob*. Building this system involves just seven subscription statements, the three in Figure 2 and the four shown as the bold components in Figure 3. The other parts, the Playlist Manager and Speaker Manager, are <100 line programs.

To be able to map the names of the models to the hosts that they are served from, AM uses Apple’s Bonjour [8] **service discovery**. Bonjour is used to find model servers on the local area network and across the Internet. New models are registered, locally with the multicast DNS server, and remotely on the **personis.org** domain name server. The service discovery is built on top of standard DNS queries, so does not require any changes to current network configurations. As the address of the server is not directly tied to the name of the model, a user’s model can be **distributed** across different computers.

The final aspect is **disconnected operation**, critical to the export of a subset of a model’s components as an external **partial model**. This can then

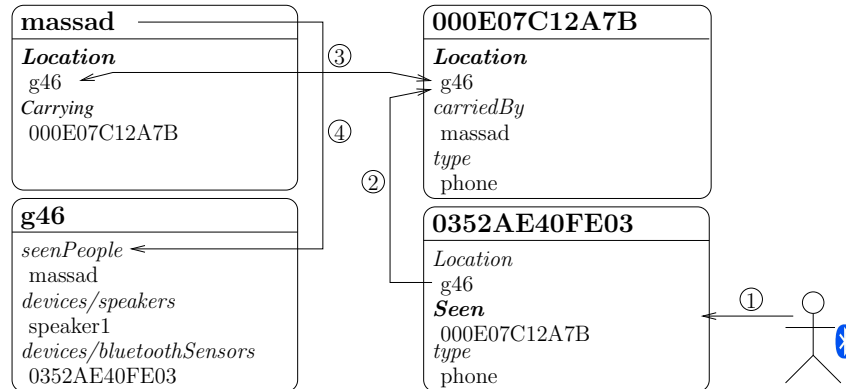


Fig. 2. Subscription events after a user walks within range of a Bluetooth sensor. Components with associated subscriptions are in bold. ① When *massad* enters the room, with phone (*000E07C12A7B*), the sensor does a **tell** to the *Seen* component. The subscription on it fires. ② This **tells** the phone model's *Location* component the value *g46*. ③ This *Location* component's subscription **tells** the user model for *massad* the location of his phone. The resolver on this component interprets his location as *g46*. ④ This triggers the subscription adding *massad* to the *seenPeople* component of *g46*.

be accessed and modified by other programs, and later imported back into the original model. A user can carry an exported model on a portable device like a PDA or mobile phone and changes can be made at remote locations without the need for Internet connectivity. Also, by allowing partial models to be exported the user is able to select which part of their model they wish to present to a 3rd party. This supports a degree of anonymity with the release of a partial model that has no identifying information.

3 Conclusions and Contributions

We have used AM to build applications that provide presence information, including the music prototype described in this paper. Important contributions of AM are : its simple but powerful active user models; subscriptions attached to components to drive information flow between models and other programs in context-aware applications; service discovery that enables models of sensors, devices, people and places to be stored at arbitrary machines; support for disconnected operation. AM enables the creation of flexible and powerful context-aware applications.

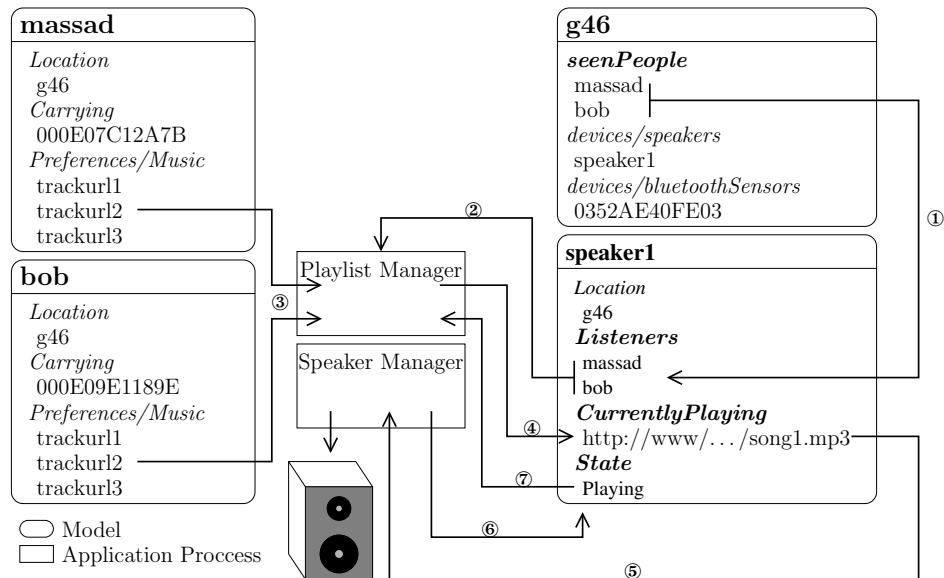


Fig. 3. Subscription events for the music system. Two people enter the room, updating the **g46** model's *seenPeople* component. ① Its subscription **tells** the *speaker1*'s *Listeners* component. ② This causes a **notify** about the two users near *speaker1* to the Playlist Manager program. ③ This program **asks** for music preferences in the users' models. ④ It chooses a music track and **tells** this to the *speaker1*'s *CurrentlyPlaying*. ⑤ The subscription **notifies** the song to the Speaker Manager which plays it. ⑥ As the song is played the *State* of *speaker1* is updated. When it is finished playing, the *State* changes to *Stopped* and its subscription **notifies** ⑦ the Playlist Manager. This selects the next track, restarting the cycle from ④ to ⑦.

References

1. Dey, A.: Providing Architectural Support for Building Context-Aware Applications. PhD thesis (2000)
2. Sohn, T., Li, K., Lee, G., Smith, I., Scott, J., Griswold, W.: Place-its: A study of location-based reminders on mobile phones. In: Ubicomp. (2005) 232–250
3. Carmichael, D., Kay, J., Kummerfeld, R.: Consistent modeling of users, devices and environments in a ubiquitous computing environment. *User Modeling and User-Adapted Interaction* **15** (2005) 197–234
4. Iachello, G., Smith, I., Consolvo, S., Abowd, G., Hughes, J., Howard, J., Potter, F., Scott, J., Sohn, T., Hightower, J., LaMarca, A.: Control, deception, and communication: Evaluating the deployment of a location-enhanced messaging service. In: Ubicomp. (2005) 213–231
5. Consolvo, S., Roessler, P., Shelton, B.: The carenet display: Lessons learned from an in home evaluation of an ambient display. In: Ubicomp. (2004) 1–17
6. Hightower, J., Consolvo, S., LaMarca, A., Smith, I., Hughes, J.: Learning and recognizing the places we go. In: Ubicomp. (2005) 159–176

7. Kay, J., Kummerfeld, B., Lauder, P.: Personis: A server for user models. In: Adaptive Hypermedia and Adaptive Web Based Systems. (2002) 203–212
8. Cheshire, S., Krochmal, M.: DNS-Based Service Discovery (2005)