"Playing with the Bits" User-configuration of Ubiquitous Domestic Environments

Jan Humble*, Andy Crabtree, Terry Hemmings , Karl-Petter Åkesson*, Boriana Koleva, Tom Rodden, Pär Hansson*

*SICS, Swedish Institute of Computer Science, Box 1623, SE-164 29 Kista, Sweden.
{humble, par, kalle}@sics se

MRL Lab, Jubilee Campus, University of Nottingham, Nottingham, NG8 1BB, UK.
{axc, tah, bnk, tar}@cs.nott.ac.uk

Abstract. This paper presents the development of a user-oriented framework to support the user reconfiguration of ubiquitous domestic environments. We present a lightweight component model that allows a range of devices to be readily interconnected and an editor to support users in doing this. The editor discovers available ubiquitous components and presents these to users as jigsaw pieces that can be dynamically recombined. The developed editor allows users to assemble lightweight sensors, devices such as displays and larger applications in order to meet their particular needs.

1 Introduction

The emergence of increasingly interactive ubiquitous environments makes it important that we consider how devices are placed within an environment, how combinations of these devices are practically managed, and how these devices work as an ensemble. This is particularly true in domestic settings where inhabitants have considerably more control over the environment than in office or work settings. An essential element of domestic environments is their evolutionary nature. Domestic environments are open to continual change and the need to support this change is essential to the successful uptake of ubiquitous devices in domestic spaces. Previous studies have highlighted how inhabitants continually reconfigure domestic spaces and the technologies within them to meet particular demands [7]. Similarly, architectural historians, such as Brand [2], have highlighted the importance of change to allow inhabitants to appropriate and adapt domestic spaces to meet their evolving needs. The dynamics of change highlighted by Brand are particularly significant when we consider the nature of ubiquitous computing in domestic environments [8].

As Edwards and Grinter [4] point out, unlike the lab houses that largely provide the focus of current research in the area, the networked home of the future will not be custom designed from the start but will emerge in a piecemeal fashion. The evolutionary development and the piecemeal introduction of ubiquitous computing into the home environment demand that we consider how real world processes of adoption and use might be supported to permit the ongoing configuration and reconfiguration of interactive devices.

2 Configuring Devices within the Home

In developing technology for domestic settings we need to make sure that it is open to continual change. Inhabitants must be able to quickly place devices in the home, understand this placement and rapidly reconfigure device arrangements. If the cost of change is too high then devices are less likely to be added to the home and become part and parcel of the 'everyday stuff' that is routinely used and exploited by its inhabitants [10]. A central issue in the development of innovative ubiquitous devices for the home is the relationship between devices and the infrastructure they exploit. In fact, as interactive devices become increasingly ubiquitous the underlying infrastructure supporting them needs to become increasingly prominent and available to users.

While infrastructures such as Jini[1], UpnP,[2] and the Cooltown[3] reduce the cost of introducing new devices they are oriented towards the *developer* of new devices rather than the eventual *users* of these devices within a ubiquitous environment. The focus of these infrastructures has by necessity been on the development of appropriate protocols and techniques to allow devices to discover each other and make use of the various facilities they offer. Limited consideration has been given to how users may be involved within the dynamic configuration of these components. Two notable examples are the Speakeasy system [6], which has adopted a composition model based on typed data streams and services, and iStuff [2], which knits together a number of ubiquitous devices via a state based event-heap. Our composition model differs from that of Speakeasy and is closer to iStuff in that we seek to exploit a distributed shared state model. However, our challenge is to allow users to understand the arrangement of devices and manipulate to these in order to meet their changing household demands.

2.1 A User Oriented Component Model

Our user-oriented model exploits a component model based on JavaBeans[4] and shares bean properties across a distributed dataspace. Component based models seek to balance between the benefits gained from assembling existing components and the

[1] Jini - http://wwws.sun.com/software/jini/
[2] Universal Plug and Play - http://www.upnp.org
[3] Cooltown - http://cooltown.hp.com/cooltownhome/
[4] Java Beans - http://java.sun.com/products/javabeans/

loss of expression and generalization. The majority of component models have focused on software components with an emphasis on supporting the programmer. In our case we wish to develop a component model that embraces a heterogeneous collection of devices and is readily understood by inhabitants of the home. Consequently we have tended to focus on making the composition as simple as possible despite the reduction of expression.

The basis of our component model is a shared dataspace that allows real world devices to make information about the nature of the physical environment digitally available. Devices can use this dataspace to become aware of their context and represent this contextual information to other devices and to make this manifest in the physical world. The aim of devices within the physical environment is either to make information from the physical available within the digital or to make digital information have a corresponding physical manifestation.

The fundamental aim of components is to ensure the convergence of the physical and the digital environment. Thus each component can be though of as a **digital/physical transformer**. There are three main classes of transformer component.

- *Physical to Digital Transformers* take physical effects and transform them into digital effects. Each transformer measures a physical effect and transforms it into a corresponding digital property that is shared through the dataspace.
- *Digital to Physical Transformers* represent the complementary set of transformers. Their job is to make digital information physically manifest by transforming the values of shared properties to drive some sort of physical device.
- *Digital Transformers* act upon digital information and effect digital information. This class of components provides a way to present to users deeper semantic reactions to changes in the environment.

This model is analogous to the one proposed within iStuff which provides developers with a set of discrete devices that can be assembled through publication within the event-heap. In this paper we build upon this work by considering how components such as the devices in iStuff and the ways in which they are configured might be exposed to users. Consequently, our emphasis is on the development of user-oriented editors that allow the dynamic composition and assembly of arrangements of devices. Below we focus on the design and development of one users editor – the jigsaw editor -that we have developed in partnership with users through a series of focused cooperative development exercises.

3 From Transformers to Jigsaw Pieces

Our starting point for the development of a user-oriented approach to reconfiguration was the formulation of a simple editing metaphor based on the notion of assembling simple jigsaw like pieces. Our choice of the 'jigsaw pieces' metaphor is based on the familiarity evoked by the notion and the intuitive suggestion of assembly by connecting pieces together. Essentially, we allow users to connect components and compose various arrangements through a series of left-to-right couplings of pieces. Constraining connections in a left to right fashion also provides users with the sense of a pipeline of information flow.

Our development of the jigsaw model exploited a paper-based 'mock up' approach [5] married to 'situated evaluation' [11]. In order to structure our investigation we presented users with a set of initial seed scenarios elaborating various transformers and their potential arrangement. These reflect different levels of abstraction and provide a *starting point allowing users to reason about* the editor, the complexity of configuration, and the nature of ubiquitous computing in the context of their everyday lives. The seed scenarios were drawn from earlier ethnographic studies [3], and some initial prototype development. An illustrative selection of scenarios and components are presented below.

Seed Scenario #1. A Common grocery item is missing from a kitchen cupboard

Using the pieces shown below, **GroceryAlarm** is connected to **AddToList,** which is then connected to **SMSSend**. **GroceryAlarm** reports the missing item after a certain time interval and the missing item is added to an electronic shopping list. This list is periodically sent via SMS to a mobile phone.

GroceryAlarm: Generates names of missing groceries in the cupboard. It detects groceries moving in and out and if one is away more than 30 minutes it is said to be out.

AddToList: Takes an element string and adds it to the list it publishes into the dataspace.

SMSSend: Takes a message string and sends this as SMS to the given phone.

Seed Scenario #2. Reminders can be directed to a number of outputs

A reminder application lets the user enter textual or auditory reminders using a touch display and microphone. This can be connected to a display, sent to a mobile, etc. The **Reminder** piece can be connected to the either the **KitchenTableDisplay** or **SMSSend** to allow users to receive reminders where they are most appropriate.

Reminder: Corresponds to a reminder application that provides an input GUI, manages the reminder alarms, and publishes reminders as URLs when reminders are due.

KitchenTable Display: Takes a URL and displays the associated web page.

SMSSend: Takes a message string and sends this as SMS to the given phone**.**

Our mock up sessions suggest that users felt a need to understand how the devices within their home where interconnected but few of them expressed any desire to "program their home". Rather users expressed the need for a simple access point that allowed them to both view the assembly of devices and to reconfigure this assembly as they used these devices for different purposes. Responding to this request we constructed the Jigsaw Editor Tablet described briefly in the following section.

4 The Jigsaw Editor

The editor presents the transformers within the domestic environment as Jigsaw pieces on a tablet based display. The editor discovers a local dataspace where transformers are registered. In order to make itself available for use a transformer exports itself to the distributed dataspace. The transformer is introspected and the properties associated with it are made available as input and output points, each transformer also has a jigsaw piece property which exports how it should appear in the editor. Each room has a dataspace associated with it and components that can be accessed from that room are registered with the dataspace.

It is worth stressing that within this approach we are constraining the potential for expression. Our emphasis is on reconfiguration rather then programming and we do not seek the richness of programming expression allowed by iCap [9] or existing visual programming languages which emphasize the development of new applications and the understandably of procedural programming constructs. Rather we wish to allow users to understand their environment and how the devices within it are interconnected and to allow them to change these interconnections through an editor.

The Jigsaw editor is made available to users using a tablet PC that uses 802.11 to talk to the dataspace (see Fig. 1). The editor discovers the dataspace and is notified of the components available within the dataspace. The editor is composed of two distinct panels, a list of available components (shown as jigsaw pieces) and an editing canvas. Jigsaw pieces can be dragged and dropped into the editing canvas or workspace. When a jigsaw piece is dragged onto the workspace it clones itself and becomes a symbolic link to the underlying component it represents. The editing canvas serves as the work area for connecting pieces together and visualizing their activities.

Fig. 1. The tablet editor and the editor screen

Connecting jigsaw pieces together works by dragging a particular piece in the vicinity of a fitting target piece. When a jigsaw piece is first dragged, non-compatible pieces in the workspace are disabled and shadowed, indicating which pieces are plausible target connections. When a user places two components close to each other they snap together if the property types for the underlying components input and outputs match. Audible feedback is provided when they snap together. When a connection is made a link is registered with the dataspace and the property of the source component is connected to the compatible property of the target (or sink) component. This link is persistent and the assembly is displayed whenever the editor reconnects. Components have the option of containing several properties making multiple connections possible. For one-to-one matching the connection is done automatically. For multiple choices a dialog window is presented to the user asking for the preferred matching for each property.

The editor not only provides audio feedback on interaction, but also traffic feedback. When properties related to jigsaw pieces in the dataspace are updated, the corresponding jigsaw piece changes colour and a short audio clip is played. Users can monitor the connections between components in the home using these indications of activity. The editor allows simple sensors such as switches, more complex devices such as displays and on-line applications to be interconnected. In the rest of this section we illustrate some of the assemblies developed by users during our evaluation sessions..

4.1 Exploiting a Simple Sensor: The Doorbell

Responding to requests by users for a doorbell sensor we extended the set of components to provide a simple touch sensitive component. This component utilizes the Smart-Its toolkit,[5] a general-purpose hardware toolkit for ubiquitous devices. A Smart-Its device has a board equipped with sensors and attenuators and a communication board supporting wireless connection to a base-station. A multi-device event engine has been developed that maps readings from the sensors into Java events which can then be used to changes properties in a JavaBean that acts as a proxy for the sensor device. This arrangement is show in Figure 2, which also shows the Smart-It device and the corresponding jigsaw piece made available to the editor.

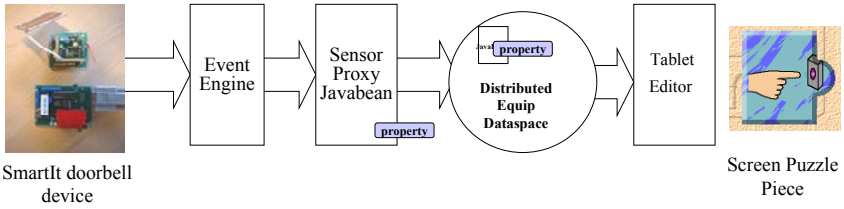

Fig. 2. Making simple lightweight sensors available to the dataspace

[5] Smart-Its - http://smart-its.teco.edu.

Once made available to the dataspace the Smart-Its device appears on the jigsaw editor and users can connect the sensor device to other components. For example, the sensor can be used to drive larger scale devices connected to the dataspace. Two such devices are the web camera and a portable display that is build using an iPAQ (a similar component exists for MMS messages to mobile phones).

4.2 Integrating Larger Devices: The Webcam and Display

Larger devices are made available to the system in a similar way as lightweight sensors and actuators. Essentially the driver used to connect to the device is 'wrapped' as a JavaBean. This JavaBean is then exported to the dataspace with the result that the corresponding property values can be shared across the dataspace. This means that the device is available to the jigsaw editor and can be combined with the inputs provided by the lightweight sensors. For example, the arrangement shown in Fig. 4 shows the pushbutton being used to signal a webcam to take a picture. The picture taken by the webcam is available as an output. Linking the webcam piece to a portable display means that this picture is then directed to that display. The arrangement as it appears on the editor screen is shown below.

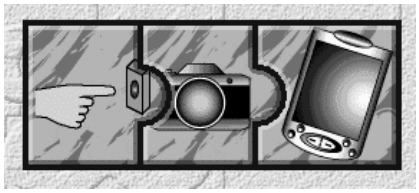

Fig. 3. Combining the doorbell the webcam and the portable display

4.3 Exploiting Applications

Users also sought to link sensors and devices to more abstract and complex entities. We address this issue by exploiting the ability of JavaBeans to make applications into components. By exporting the properties from the JavaBean used to represent an applications we can it available to users and combine the application with lightweight sensors and devices.

For example, to support remote monitoring users can connect a lightweight Smart-It motion sensor to a more complex webcam device and make the output from the device available to a weblog application. This configuration means that whenever motion is detected within a space this is used to take a picture that is then automatically added to the weblog. Users away from home can access the weblog to view the image via any web browsers..

5 Conclusions

In this paper we have presented the development of an editor that supports user-configuration of ubiquitous computing environments in the home. The editor exploits a jigsaw metaphor to make user-configuration of complex functions readily intelligible. It is worth reflecting on the heterogeneity of the components users wished to connect together. It was not unusual to see users develop assemblies that combined lightweight sensors with more traditional computer devices and larger applications and services. In order to allow user to construct these heterogeneous assemblies of devices we developed an editor that discovers components that have made properties available to a distributed dataspace. This arrangement allows quite distinct types of component to offer a very simple lightweight state based interface, which the editor can then, present to users to allow them to construct assemblies to meet there particular needs.

Obviously, the process of user-based development is ongoing, with each iteration leading to the further refinement of the technical infrastructure and toolkit of devices, software and applications that embed ubiquitous computing in the domestic environment to meet real user needs. We are currently in the process of placing the toolkit in a number of users' domestic environments for continued assessment. We envisage these trials raising significant issues of access control and management.

The current version of the toolkit including the Jigsaw editor is publicly available and may be downloaded from *http://www.sics.se/accord/toolkit.html*. This allows developers to wrap their particular sensor, devices or applications as JavaBeans, to provide an iconic representation of the device and to publish them to our dataspace. Once within the dataspace they become available for use through the Jigsaw editor. Our aim is to allow users more control over the assembly of the ubiquitous devices that share their environment in order that users within the home can readily situate ubiquitous technologies in the space they live in

References

1. Ballagas et al. (2003) "iStuff", *Proc. CHI '03*, Florida: ACM Press.
2. Brand, S. (1994) *How Buildings Learn*, New York: Viking.
3. Crabtree, A., Hemmings, T. and Rodden, T. (2002) "Pattern-based support for interactive design in domestic settings", *Proc. DIS '02*, London: ACM Press.
4. Edwards, K. and Grinter, R. (2001) "At home with ubiquitous computing" *Proc. Ubicomp '01*, pp. 256-272, Atlanta, Georgia: Springer-Verlag.
5. Ehn, P. and Kyng, M. (1991) "Cardboard computers", *Design at Work* (eds. Greenbaum, J, and Kyng, M.), pp. 169-195, New Jersey: Lawrence Erlbaum.
6. Newman, M. et al. (2001) "Designing for serendipity", *Proc. DIS '02*, London: ACM Press.
7. O'Brien, J. et al. (1999) "At home with the technology", *ACM Trans. on Computer-Human Interaction*, vol. 6 (3), pp. 282-308.
8. Rodden, T. and Benford, S. (2003) "The evolution of buildings and implications for the design of ubiquitous domestic environments", *Proc. CHI '03*, Florida: ACM Press.

9. Sohn, T. and Dey, A. (2003) “iCAP”, *Proc. CHI ’03 (interactive poster)*, Florida: ACM Press.
10. Tolmie, P., et al. (2002) “Unremarkable computing”, *Proc. CHI ’02*, pp. 399-406, Minneapolis: ACM Press.
11. Twidale, M., Randall, D. and Bentley, R. (1994) “Situated evaluation for cooperative systems”, *Proc. CSCW ’*94, pp. 441-452, North Carolina: ACM Press.