Programming Home Care

Claire Maternaghan and Kenneth J. Turner
Department of Computing Science and Mathematics
University of Stirling, Stirlng
Scotland, UK
Email: cma, kjt @cs.stir.ac.uk

***Abstract*—The home is composed of many different devices, services and technologies. These rarely communicate with one another, and require various different computer systems and applications to be able to interact with them all remotely. A challenge within telecare is being able to exploit the functionality of these devices within the home and offer a common means of control, monitoring and programming, either locally or remotely. Homer, a home system designed and developed at the University of Stirling, can communicate with any device within the home and then expose the functionality to a range of different interfaces on different platforms and devices. This paper introduces Homer, describing how it communicates with the devices within the home, a brief description of the system architecture, and finally describes its user interfaces for the home. Home requirements are introduced at the beginning of the paper, explored throughout, and finally evaluated at the end.**

I. Introduction

A. Context

The world population is gradually aging [1]. As a result, there is increased pressure in most countries to provide adequate support for older people. Although technology is only part of the solution, telecare (remote support of home care) has been enthusiastically promoted as a way of helping older people to continue living independently in their own homes. Telecare involves some kind of computer-based system in the home that monitors for undesirable situations such as falls, bed wetting or overflowing baths. The home provision is supplemented by a link to a call center for dealing with alerts and calls for help.

However, telecare technologies are still relatively undeveloped. Commercial systems often do not incorporate the latest research advances. More seriously, telecare systems are usually relatively fixed in function. Where changes are possible, they normally require specialised technical expertise and often reprogramming. As a result, telecare systems can be hard to customise for individual circumstances, and can be hard to adapt as these change over time [2].

Home automation has a longer history going back several decades. However, most approaches are relatively unsophisticated. Indeed, home *control* rather than *automation* would often be a better designation. Much of the commercial effort in this area is concerned with capabilities such as being able to stream audio and video around the home. Although some home systems do offer programmability, this usually requires specialised technical expertise and is aimed more at the hobbyist rather than ordinary householders.

This paper describes Homer – a home system that is designed to meet the needs of both telecare and home automation. Both applications share a common core of capabilities, though they also require specialised devices and services in each application. The study in [3] discovered that users would like the ability to control the home (though they would not wish this to seem like programming). As the target users have very limited technical knowledge, a home system needs to be made easy to use. However, the system also needs to offer more sophisticated capabilities to specialists (e.g. a care professional or a home system installer). Simple tasks must therefore be easy, while complex tasks must be possible.

B. Related Work

This paper touches on many related fields: telecare, home automation and smart homes, component architectures, policy-based management and end-user programming. As a result, only a high-level overview of related work is feasible here.

Telecare Commercial telecare solutions are available from companies such as Cisco, General Electric, Initial, Intel, OmniQare, Philips and Tunstall. Current telecare systems are relatively unsophisticated, and generally require specialised installation expertise (especially if they have to be modified). OmniQare is unusual in being a framework for third parties to add telecare services. As telecare is a fairly recent development, standards are still in their infancy. The Continua Health Alliance (www.continuaalliance.org) and the European Telecommunications Institute (www.etsi.org) are working towards telehealth and telecare standards, but interoperability among different devices and systems is still a long way off.

Home Automation At device level, several standards have evolved to support home automation. These include infrared (home appliance control), KNX (building management and domestic applications) and X10 (mains appliance control). More interesting are packages that aim to offer higher-level control over home devices. These include Control4 (a widely adopted framework), Cortexa (rule-based, but not flexible or simple enough), Girder (technical knowledge needed to define input-output event mappings), Home Automation Inc. (designed for installers rather than end users), and HomeSeer (particularly focused on control via remote devices). In general, these approaches lack either the sophistication needed for full home automation or the simplicity required by non-technical users.

Component Frameworks Many component architectures have been developed. In the context of home systems, rel-

PervasiveHealth 2011, May 23-26, Dublin, Republic of Ireland

DOI 10.4108/pervasivehealth.2011.246066

evant approaches include Atlas (home sensor/actuator platform), Jini (distributed network architecture), Open Services Gateway initiative (service platform, www.osgi.org), Service Component Architecture (implementation-independent component interconnection, www.osoa.org), and Service Oriented Device Architecture (device inter-working, www.eclipse.org/ohf/components/soda). Of these, approaches based on Service Oriented Architecture have proven particularly popular.

Policy-Based Management Policies are automated rules for controlling systems, with many possible applications. Examples of the many approaches include ACCENT (domain-independent policies [2]), Drools (business rules, www.jboss.org/drools), Police (emphasis on distributed policies) and Ponder (distinctive features such as domains, conflict handling and refinement [4]). Although simple rules are supported by some commercial home automation packages, the richer field of policies applied in the home has not been widely explored (ACHE [5] and [6] being a few examples).

End-User Programming This allows both technical and non technical people to express their desires in a logical format. There are currently four main techniques being researched. Visual programming (e.g. [7], [8], [9]) which is arranging pictorial representations in meaningful ways to represent logic; this requires a basic level of logical thinking from the user, but is relatively simple for the system to parse. Natural language (e.g. [10], [11], [12], [13], [14]) uses natural language to express desired functionality; this requires little logical thought from the user, but can be very challenging for the system to parse. Tangible programming (e.g. [15], [16], [17], [18]) is combining physical representations to form rules; this requires a level of logical thought from the user and is a rather restrictive means of programming, but is relatively simple for the system to parse. Finally, programming by demonstration (e.g. [14], [19]) is physically demonstrating what you would like the system to do by manipulating real world objects; relatively simple for users, but can lead to ambiguity when parsing as it is hard to know which current environmental conditions or user actions are meant to be included (for example, the demonstration takes place after 6pm, whilst it is raining outside, and the user walks through a door to turn on a lamp. Which of these actions and conditions are relevant?).

C. Requirements

An extremely important feature of a home system is coping with the very large range of different devices, technologies and protocols currently on the market and in people's homes. The system must be able to easily and dynamically support the addition of new devices as they become available. Currently very few of the systems described above can handle new technologies, as they support only custom, in-house components. It therefore becomes very difficult for these companies to keep up with the new devices that come on the market.

There are three main features of any home system: monitoring, control and automation. Monitoring features allow residents, friends, family and/or care professionals to view the current state of the connected devices within the home. Some examples include viewing the surveillance camera feed for the front porch, checking if the front door is locked, or if the resident has taken their medication. Secondly, it is useful to be able to control these devices, either locally by the resident or remotely by friends/family/care professionals. Some examples include turning on and off individual lights, adjusting the heating, or setting a reminder to take medication. Finally, it is very useful to automate particular tasks for the resident. Examples of such automation include turning on lights in an occupied room when it starts to get dark, reminding the resident to take their medication, and sending a message to a caregiver if the resident misses their medication for a day. These rules could be set up at installation time by the installer or care professional, and managed by the care professional and/or the resident.

Currently companies focus on the monitoring and controlling aspects of home systems, but for telecare the automation aspects are vital. A home which is kitted out with technology to enhance and prolong the resident's stay should be there to offer help and ease their daily lives. This requires a degree of automation which should be customisable and controllable by the resident and/or care professionals involved. The few existing systems which offer automation are often programmed by the installers and require technical knowledge to set up.

It is crucial that the system is designed to be as simple as possible to live with, offering easy means to monitor, control and automate the home. If a home system is overly complex the average resident is highly likely to avoid using it. Also, since the average resident, friends, family or care professionals will have very little programming experience, the system must be designed so as not to require any technical background knowledge. Existing systems generally offer a very high standard of user interface for monitoring and controlling features of a home. However programming features, if they exist, are generally designed for technically experienced individuals.

A summary of requirements for a home system is therefore: support legacy devices, support new devices easily and dynamically allow the user, friends, family and care professionals to monitor, control and automate the home and the devices within it. Finally, all these tasks must be simple for any user.

D. Overview

This paper discusses Homer, an OSGi (www.osgi.org) home system developed by the authors, which aims to meet the requirements discussed in section I-C. The design of Homer components is described in section II, along with some examples of currently implemented components and how components work with policies to automate the home. The architecture is introduced in section III. User interfaces and end-user programming techniques are discussed in section IV. Finally, the section V evaluates Homer with regard to the requirements, explores future work and concludes what has been discussed in this paper.

II. HOMER COMPONENTS

'Component', within the context of this research, is the term used to mean a device or user service within the home. Examples of devices include lights, medicine dispensers and televisions. Examples of user services include SMS, weather forecasts and Twitter. This section discusses the design of Homer components, gives some examples and finally shows how components can be automated through the use of policies.

A. Design

Homer components are lightweight, loosely-coupled modules that can be installed, modified and removed from Homer at run-time. This capability is intrinsic to OSGi. It is important to develop a home system which can dynamically install and uninstall devices within the home as they become available and unavailable, without interfering or restarting the home system. Within telecare, devices may be added and removed frequently as newer models of existing devices become available or the user develops new health problems that would benefit from new devices. Because devices, and the services that they offer to the home, are volatile it is crucial that the system does not have dependencies on the components themselves, and indeed that the components do not have dependencies on each other.

A component represents a device or a user service. As simple examples, a medication dispenser provides usage information, a thermostat can check the room temperature, and a lamp module offer actions such as turning a lamp on, off or to some dim level. Homer categorises these aspects of a component as triggers, conditions and actions.

A trigger reports something that happens externally to Homer, e.g. the front door is opened. A condition checks the state of a component, e.g. whether the front door is open. An action allows the user to request a change external to Homer, e.g. to lock the front door. A component must state what triggers, conditions and/or actions it can support. By doing so it is offering a guarantee that it will post the relevant triggers and, on request, evaluate conditions and support actions.

Components are not allowed to communicate directly with other components; shared functionality must be provided by a Homer service. Components should be simplistic, with no intelligence or complex logic of their own. This cleanly separates the core devices and services from the logic and applications that build on these.

B. Component Examples

A range of devices have been integrated with Homer to demonstrate its functionality. Examples include the following:

Camera: The camera component allows for movement detection and photos/videos to be recorded on request; these can also be emailed or sent to a digital display within the home. This offers security features for residents, e.g. to check who is at the door or to check for a prowler outside the house. The camera offers communication features to allow residents to keep in touch with friends and family. It also offers peace-of-mind features, e.g. to allow informal carers to know that the resident is up and about the house.

Email This supports exchange of email on behalf of other components.

Infrared Most audiovisual devices have infrared remote controls. With ageing, users may lose dexterity in their hands so that traditional remote controls become difficult to use. The Homer infrared controller extends the variety of home appliances that can be controlled. For example, programs can be recorded automatically and appliances can be used through a simple touch screen.

Momento A very important aspect of telecare is communication. Older people, on the whole, like to feel close to their friends and family: photos are a good way of doing this. The Homer component for the i-mate Momento wireless digital photo frame (www.momentolive.com) has its own email address, allowing friends and family to email photos for immediate display.

Nabaztag The Nabaztag 'Internet rabbit' (www.nabaztag.com) has been adapted as a user-friendly interface device. As a non-threatening interface to technology, this is ideal for technophobic or technically inexperienced users. The rabbit provides an interface which supports speech recognition, RFID tag recognition, text-to-speech conversion, and audible, visual or gestural alerts.

Oregon Scientific Homer can monitor the home environment using wireless devices produced by Oregon Scientific (www.oregonscientific.com). These are mostly used for information such as room temperature and humidity level. This information can be used to control the household environment.

Tunstall For telecare, Homer supports a range of home devices produced by Tunstall (www.tunstall.com). This includes basic devices such as flood detectors, gas detectors, movement detectors and pressure mats, as well as more specialised devices such as medicine dispensers and door entry systems.

Twitter Support for Twitter (twitter.com) helps to maintain communication using short messages. These can be used for status updates and alerts.

SMS Similar to the email component, this supports sending and receiving SMS messages.

Visonic These sensors (www.visonic.com) are mostly for monitoring home activity, including door, window, motion and gas sensors.

WiiMote The WiiMote (a hand-held controller, www.nintendo.com/wii) has been given a Homer component wrapping. The WiiMote can be used for gestural input; for example, it can mimic nodding or shaking the head in response to questions. It also has buttons which can be used for control functions. This is a good example of how a mass-market device, originally for a completely different purpose, can be adapted for use in telecare or home automation.

X10 This widely used technology for controlling mains appliances and lighting allows Homer to manage many devices around the home.

C. Policy-Based Control

Homer policies allow different features and functionality of the various components within the home to be interconnected

in logical ways. This results in a system which can be fully automated by interconnecting component functionality at a higher level. By handling this automation at a higher level there is no direct dependency on the components and allows for dynamically piecing together different features and services that the components offer within the home. This dynamic and loosely coupled approach is vital for a homecare system where devices will come and go.

A policy for Homer contains triggers and conditions in a 'when' clause, and actions in a 'do' clause (for more details see [20]). A primitive example is:

when the front door opens **do** turn on the hall lamp.

This is a policy which involves the Visonic and X10 components. The Visonic component offers a trigger for when a door is opened, here the front door. The X10 component offers an action to turn on a lamp, here the hall lamp. By putting these two pieces of functionality together we can create a policy which will automatically turn on the hall lamp whenever the front door opens.

A slightly more sophisticated example is:

when Mary is going to bed **do** set the bedroom temperature to $24\,^{\circ}\mathrm{C}$.

This policy has to know how to evaluate the trigger in order to be able to evaluate the policy. The trigger is not from a component, instead it is from another policy:

when Mary is at home **and** Mary turns off the television **and** the time is between 2130 and 2230 **do** tell Homer Mary is going to bed.

This policy similarly make use of other policies. This notion of gathering multiple triggers to produce a meta-trigger is called sensor fusion. By extracting commonly used logic into separate policies the user can write everyday policies at a higher, more easily articulated level.

D. Sample Policies

The following illustrates how policies can support telecare:

Sleeping Problems:

- **when** Tom gets out of bed at night **and** opens the front door within 5 minutes **do** activate his neighbour's bedside alarm
- **when** John gets out of bed at night **do** turn on the hall and toilet lights

Memory Problems:

- **when** Brian is late in taking medication **do** provide a reminder
- **when** Brian does not take medication for a whole day **do** send an email alert to the surgery **if** it is a weekday
- **when** the time is between 0500 and 1200 **and** Mary is in bed **and** the diary has an event in an hour **do** activate Mary's alarm clock
- **when** the living room is unoccupied for 5 minutes **do** turn off the television

Mobility Problems:

- **when** a person with a valid RFID tag arrives at front door **do** open the door **and** alert the user **and** display the visitor's photo

Hearing Problems:

- **when** music is playing **and** (the telephone rings **or** the doorbell rings) **do** reduce music volume by 90%

Comfort Features:

- **when** the living room is occupied **and** the living room light level falls below 60% do turn on the lamp
- **when** Mary is getting up **or** Mary is going to bed **do** set the bedroom temperature to a comfortable level
- **when** movement is detected in a room **do** set the room temperature to a comfortable level
- **when** the weather forecast predicts very cold weather during the night **do** turn on the heating
- **when** an SMS is received from Mary saying 'warm the house' **do** turn on the heating
- **when** the TV in the living room is turned off **and** the time is after 9:30pm **do** turn on the electric blanket in the master bed **and** tell Mary 'electric blanket has been turned on'

Safety Features:

- **when** the fire alarm is activated **and** no one is home **do** send an SMS alert to a neighbour
- **when** movement is detected outside **and** the time is between 2300 and 0600 **and** the house is in sleep mode **do** turn on the outside light **and** turn on the outside security camera
- **when** flooding is reported in the bathroom **do** turn off the water **and** send a recorded message by phone to a neighbour

III. Architecture

Homer acts as the middleware platform between users and the devices and services in their home, offering all three features (monitoring, controlling and automating). Figure 1 shows the high level architecture of Homer, showing how Homer sits between the components and the user. Various aspects of the architecture are as follows:

Components The components within the home are connected using their respective technologies, and given a Homer wrapping to expose the features and functionality of the component for Homer. The details and design of components were discussed in section II.

Web Server The web server exposes the functionality of Homer through HTTP, using a custom API and JSON (JavaScript Object Notation, www.json.org) for data interchange. This allows external applications to be developed which provide the user with the ability to monitor, control and/or automate the home. Example applications have been developed for the iPhone and iPad [20].

Homer Central Framework The internal Homer framework is composed of three main parts: the policy server, the database and the OSGi event broker for message exchange. It is responsible for managing components, requests from the web server, handling the evaluation and execution of policies, and most importantly the communication among all these entities. The home framework is discussed further in [20].

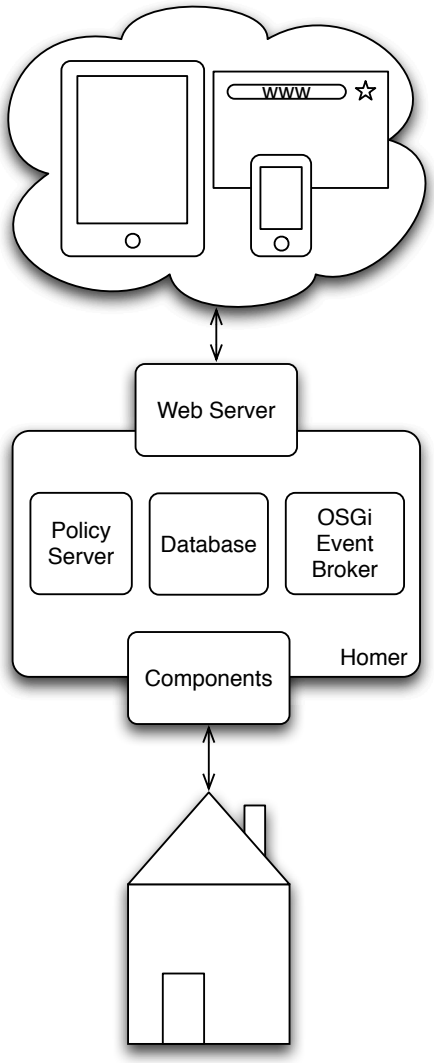

Fig. 1. Homer Architecture

IV. User Interfaces

This section discusses how the residents and their care professionals, friends and families can interface with the home through Homer. It firstly looks at the technologies and devices available for interacting with Homer, then describes a new technique for simplifying the home system called 'perspectives'. Finally, Homer's novel approach to end-user programming of the home is described.

A. Devices

Homer has been designed to support any hardware or platform by exposing the home through an HTTP API, using JSON for data communication. A whole host of possible user interfaces exist, including Google Android phones, Apple iPhone/iPad/iPod Touch devices, web browsers, televisions, PCs and tablet computers. To demonstrate this a web page (using Google Web Toolkit) has been written, along with an iPhone and iPad application (see figure 2 for the iPhone application).

It is important that telecare systems offer a range of interfaces to help all the people involved (resident, friends, family, care professionals) to choose a mode of interaction which feels most comfortable to them. By doing this it encourages use of the home system, offering a higher degree of ease than if they were forced to use only one particular mode. An advantage of offering an open API is that any manufacturer could create devices which offer Homer functionality. For example, a remote control manufacturer could make a Homer remote control for all the lights in the home. A system which encourages third-party participation allows users of the home system to benefit greatly through a wider and larger range of devices, components and services.

B. Perspectives

People conceptualise their environments in different ways, so it is important that a home system can cater for this. Homer offers the ability for users to access devices, services and conditions from various 'perspectives': device type (e.g. television, mobile phone, fan), location (e.g. living room, kitchen, work), personal (e.g. me, Mary, the kids) and time (e.g. sunset, every weekday, a birthday). There are perspective crossovers, where items in one perspective can equally be viewed from another perspective. As an example of crossover: "lamp $\rightarrow$ Mary's bedside lamp" (type), "bedroom $\rightarrow$ Mary's bedside lamp" (location) and "Mary $\rightarrow$ my bedside lamp".

By offering perspectives users are able to conceptualise their home in a way that is familiar to them. Any household member can immediately start using the home system from their perspective without changing configurations or, more importantly, the way they visualise the home.

This technique has been applied throughout the home system, from monitoring and control to the programming of rules for automation. As a simple example an iPhone application was developed to demonstrate that all the devices and services within the home could be accessed through their location or their device type. Figure 2 shows the application displaying devices and services by their type, with another tab allowing the user to browse using locations instead.

C. End-User Programming

End-user programming is an extremely challenging field, as introduced in section I-B. Because it is rare for non-technical individuals to be able to express their desires in an organised, logical and constrained manner, it is difficult to build systems that support communication between such users and computer systems. All efforts must be made to simplify the language the user must work with to communicate with the system, yet remain constrained and unambiguous for the system to translate.

Homer uses a hybrid of natural language and visual programming techniques. Natural language was chosen due to its simple and easily understood nature for users. Visual programming was chosen due to its visually attractive style

Fig. 2. iPhone Application

and ease of interpretation by the system. An example of the success that can be achieved combining these techniques is found in CAMP [21], which uses a magnetic poetry metaphor and allows users to visually piece together snippets of natural language to make simple rules for a video capture and access system.

The possible triggers, conditions and actions for devices and services within Homer can be composed to form policies, as described in II-C. Each term of a policy is a particular trigger, condition or action associated with a particular device or service instance. Each term has an associated location, device type and, in some cases, person or time. Using the notion of perspectives it is therefore possible to browse terms from different perspectives.

D. Disguised Programming

Programming is a task that is often avoided by non-technical people. A technique used within Homer is the notion of disguising the programming aspects of the home application.

To demonstrate this notion, a prototype iPad application for the home is described. The main display is an interactive plan view of the home. The user may choose to navigate to their television by touching the living room on the plan view, then choosing the television. The television page offers monitoring and control of the device, for example showing if the television is currently on or off, and being able to change its channel, volume or power. On this same page there would be additional buttons saying things such as 'turn off when...', 'turn on when...' or 'lower the volume when...'. These would then act as templates for policies, where the user can simply fill in the 'when' part of the policy to describe when the event should occur. On this same television page the user is able to view what policies affect this particular television. The possible triggers, conditions and actions are grouped together with policies listed for each. For example:

Turns on when:

- Mary gets home from work.
- The DVD player is turned on.

V. Conclusions

This section concludes the paper by evaluating the research carried out and concluding what has been discussed.

A. Evaluation

The requirements in section I-C listed important aspects needed for a home system that are now revisisted.

1) Support New and Legacy Devices: Due to the existing wide array of devices and technologies for the home, and the high number of new devices continually arriving on the market, it is crucial that a home system has an architecture which can support these devices. Homer exploits a service component architecture, which allows the different devices and software services to be treated as components, offering services to the system. Components within Homer have a plug-in style whereby they can be added, altered and removed at run-time.

Homer has been designed to support components written by third-party developers and companies. In the commercial world this would encourage others to write Homer components for their devices (similar to the highly successful Control4 model, www.control4.com). Writing a component for Homer is relatively simple, requiring little knowledge about Homer itself. The component must be written in Java, with the only requirement being to implement a HomerComponent class, and the required methods to advertise the components triggers, conditions and actions.

Due to these design decisions Homer can easily support new and legacy devices being added, edited and removed at run-time, and being developed by third-party companies.

2) Support Monitoring, Controlling and Automation: The three main features of a home system are monitoring, controlling and automation. As descibed in the requirements section very few companies offer all three. Homer, however, fully supports all three features, and exposes these through an open API for developers. This allows any third-party developer to write or create devices or applications which offer any or all of these features.

3) Suitable for Non-Technical People: A home system must be designed for all possible users, who are most likely going to be non-technical. This was taken into consideration throughout the design and development stages of Homer, resulting in a home system which has been designed for a user with minimal technical knowledge. Through new methods such as perspectives and disguised programming, Homer offers a truly

unique set of interfaces for a home, specifically designed for the non-technical user.

B. Future Work

Homer scales well to hundreds of policies. However, a potential issue with many policies is that they may contradict each other, especially when various different groups of users could be writing and editing policies for the same household. Work is under way to adapt conflict handling techniques from ACCENT [2] for use with Homer, focusing on how best to present and handle policies within a home environment for non-technical users.

Another possible problem with many policies is that it might become difficult to discover why the home took certain actions. Techniques from expert systems are being investigated as a means of explaining to the user the reasoning that led to particular actions. This will be particularly important when conflict handling is introduced, since the user might wonder why some policy was not applied.

C. Users

Homer is aimed at the active aging population, who could benefit from some automated tasks or health related help and monitoring. However, due to the nature of a home system there could be many different potential users. For that reason Homer is designed to be as simple as possible for all functionality, offering optional advanced features to those who desire.

Once research and development have been carried out on policy conflict handling and policy explanation, as described in section V-B, a full user trial is planned. This will evaluate the usability and design aspects of Homer in a real-life environment.

D. Conclusion

The paper has introduced the current problems with home systems for both telecare and home automation, and then discussed some novel concepts for system design and programming the home to help with these problems.

It has been explained that Homer aims to meet the needs of both telecare and home automation through core capabilities coupled with support for more specialised devices and services. A flexible architecture has been introduced that allows components to describe key features of themselves. For example the triggers, conditions and actions supported by a component make user control and configuration easy. New and existing devices can readily be added or removed at runtime, allowing the home system to evolve. Components are also dynamically integrated with policies as a means of letting users manage how the home should behave.

The functionality of Homer is exposed through a platform-neutral interface that makes it possible to develop a wide range of user interfaces. The application to telecare has been described, with components appropriate to home care illustrated.

Homer has been designed to be simple for both technical and non-technical people to be able to use both locally and remotely. By using novel techniques of perspectives and disguised end-user programming, Homer offers a significant benefits for both home care and home automation.

ACKNOWLEDGMENT

Claire Maternaghan is supported by the Scottish Informatics and Computer Science Alliance (SICSA), the University of Stirling, and the MATCH project (Scottish Funding Council, grant HR04016). The authors are grateful to their colleagues on the MATCH project for their advice and support.

REFERENCES

[1] L. A. Gavrilov and P. Heuveline, "Aging of population," in *The Encyclopedia of Population*, P. Demeny and G. McNicoll, Eds. London, UK: MacMillan, Jan. 2003, pp. 27–50.

[2] K. J. Turner, L. S. Docherty, F. Wang, and G. A. Campbell, "Managing home care networks," in *ICN'09*, R. Bestak, L. George, V. S. Zaborovsky, and C. Dini, Eds. Los Alamitos, USA: IEEE Computer Society, Mar. 2009, pp. 354–359.

[3] C. Maternaghan, "How do people want to control their home?" University of Stirling, UK, Tech. Rep. CSM-185, Dec. 2010.

[4] N. Damianou, E. C. Lupu, and M. Sloman, "The Ponder policy specification language," in *Policy Workshop 2001*, ser. LNCS. Berlin: Springer, 2001.

[5] M. C. Mozer, "The neural network house: An environment that adapts to its inhabitants," in *Proc. AAAI Symp. on Intelligent Environments*, M. Coen, Ed. AAAI Press, Mar. 1998, pp. 110–114.

[6] C. Leong, A. Ramli, and T. Perumal, "A rule-based framework for heterogeneous subsystems management in smart home environment," *IEEE Transactions on Consumer Electronics*, vol. 55, no. 3, pp. 1208–1213, Aug. 2009.

[7] T. Sohn and A. K. Dey, "iCAP: An Informal Tool for Interactive Prototyping of Context-Aware Applications," in *CHI '03*. New York, New York, USA: ACM Press, 2003, p. 974.

[8] Y. Li, J. Hong, and J. Landay, "Topiary: a tool for prototyping location-enhanced applications," in *UIST'04*. ACM, 2004, pp. 217 – 226.

[9] M. W. Newman, A. Elliott, and T. F. Smith, "Providing an Integrated User Experience of Networked Media, Devices, and Services through End-User Composition," *Pervasive*, pp. 213–227, 2008.

[10] L. Kagal, T. Finin, and A. Joshi, "A policy language for a pervasive computing environment," *POLICY'03*, vol. 2003, 2003.

[11] J. Rimmer, T. Owen, I. Wakeman, B. Keller, and J. Weeds, "User Policies in Pervasive Computing Environments," *User Experience Design for Pervasive Computing*, 2005.

[12] J. Weeds, B. Keller, D. Weir, I. Wakeman, J. Rimmer, and T. Owen, "Natural language expression of user policies in pervasive computing environments," *OntoLex'04*, 2004.

[13] M. Knoll, T. Weis, A. Ulbrich, and A. Brandle, "Scripting your home," *LNCS*, vol. 3987, p. 274, 2006.

[14] K. Gajos, H. Fox, and H. Shrobe, "End User Empowerment in Human Centered Pervasive Computing," in *Proc. Pervasive 2002*. Citeseer, 2002, pp. 134 – 140.

[15] S. R. Klemmer, J. Li, J. Lin, and J. A. Landay, *Papier-Mache*. New York, New York, USA: ACM Press, 2004.

[16] R. Ballagas, M. Ringel, M. Stone, and J. Borchers, "iStuff: A Physical User Interface Toolkit for Ubiquitous Computing Environments," *CHI'03*, vol. 16, pp. 2–4, 2003.

[17] S. Greenberg and C. Fitchett, *Phidgets: Easy Development of Physical Interfaces through Physical Widgets*. New York, New York, USA: ACM Press, 2001.

[18] T. Rodden, A. Crabtree, T. Hemmings, B. K. J. Humble, K.-P. Åkesson, and P. Hansson, "Configuring the ubiquitous home," in *COOP'04*. Amsterdam: IOS Press, May 2004, pp. 215–230.

[19] A. K. Dey, R. Hamid, C. Beckmann, I. Li, and D. Hsu, "a CAPpella: Programming by demonstration of context-aware applications," *SIGCHI'04*, p. 40, 2004.

[20] C. Maternaghan, "The homer home automation system," University of Stirling, UK, Tech. Rep. CSM-187, Dec. 2010.

[21] K. Truong, E. Huang, and G. Abowd, "CAMP: A magnetic poetry interface for end-user programming of capture applications for the home," *UbiComp'04*, pp. 143–160, 2004.