Push-pins: design-by-user approach of home automation programming

Kentaro Fukuchi
Takeo Igarashi
Japan Science and Technology Agency.
Frontier Koishikawa Bldg. 1-28-1
Koishikawa, Bunkyo-ku, Tokyo
112-0002 JAPAN.
fukuchi@megaui.net

Maki Sugimoto
Charith Fernando
Masahiko Inami
Keio University
4-1-1 Hiyoshi, Kouhoku-ku,
Yokohama-shi, Kanagawa 223-8526 JAPAN.
sugimoto@kmd.keio.ac.jp
charith@kmd.keio.ac.jp
inami@inami.info

UbiComp 2009, Sep 30 – Oct 3, 2009, Orlando, FL, USA
ACM XXX-X-XXXXX-XXX-X/XX/XX.

Abstract
A tangible programming interface called ``Push-pin'' is proposed, which involves end-users in designing smart home programs for home appliance automation. Programming on the Push-pin system is based on a stimulus-response model in which an appliance connected to another appliance via a network gets activated when the other appliance is activated. To interconnect two appliances, the user puts a pin associated with an output appliance such as a lamp or a robot cleaner into an input module such as a switch or a motion sensor. The input appliance sends stimulus data with the ID of the pin as the destination address.

Keywords
Smart home, tangible user interface, ubiquitous computing, end-user programming

ACM Classification Keywords
H.5.2 Information interfaces and presentation (e.g., HCI): User interfaces.

Introduction
The concept of the “smart homes” (or home automation), which are expected to assist people in carrying out daily activities in the living or working environment, has emerged recently in the wake of

extensive research on ubiquitous computing and pervasive technology. Smart homes are typically equipped with a large number of sensors, cameras, actuators and electric appliances connected to a network and controlled by a central server [1].

There are basically two approaches to enable the “smartness” of smart homes. One approach is to have experts predefine the setup on the basis of their knowledge and established heuristics. However, in this approach, experts have to address all possible needs of the occupants at the design stage, which is difficult. Moreover, even after all possible needs are well studied in advance, the occupants may continuously and repeatedly reconfigure their environment after the implementation of the automation system (e.g., purchase new appliances) [2], which require additional help from experts to redesign the system.

The other approach is to implement an intelligent system that can learn and adapt to the occupants (e.g., [3]). However, it will be annoying if the system misidentifies patterns and provides unwanted services (e.g., Clippit of Microsoft Office). Mozer, who developed a smart home using a neural network that can learn and predict his behavior, commented: “The result is likely to be that the system mispredicts often and annoys the inhabitants more than it supports them. [4]”.

Intille et al. proposed a new approach in which the system does not activate any appliances automatically, but just provides information to occupants and educate them on how to control their environment [5]. For example, when the system detects that a window should be opened to control the room temperature, the system subtly illuminates a light bulb embedded in the window frame instead of opening the window automatically. This approach solves the problem of unexpected or undesirable automation, but prevents the occupants from enjoying certain automation, such as automatically performing specific tasks when the occupants are not at home.

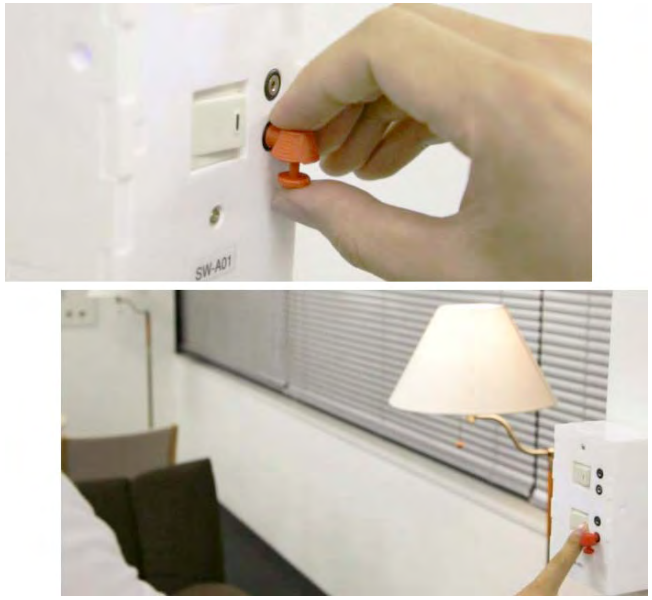

figure 1. Tangible pin associated with floor lamp inserted into a pin slot of switch, to control the floor lamp.

We introduce a tangible programming interface that enables occupants to program and configure smart systems themselves. This do-it-yourself approach has a number of advantages: (1) as the occupants are the ones most familiar with their own in-house activities and the structure of their dwellings, they will be able to

select the most appropriate solution for their home, (2) getting the occupants involved in the design and setting up process leads to a greater feeling of control of their homes and higher acceptance of the final design, and (3) very importantly, occupants carrying out the programming and configuration themselves is often more cost effective than involving the professionals [6].

The proposed interface enables users to specify the stimulus-response relationship between appliances using a physical tag called “push-pin” (Figure 1). Because modules activating appliances are explicitly selected and configured by the occupants, they are less likely to provide unwanted or unexpected services. If the occupants feel that a program is unsuitable, they can easily reconfigure it. Physical presence and direct interaction with appliances with pins helps users to identify and configure the relationships between appliances in situ. This can be done by using a computer using a standard graphical user interface (GUI). However, this can be difficult for people who are not familiar with computers. GUI operation also poses an additional cognitive overhead of recognizing the relationship between the icon on the display and the physical entity.

This paper describes the basic concept of the Push-pin system and our prototype implementation. We first introduce related studies and then describe our programming model and provide an overview of the system. We then present several usage scenarios, details of our current prototype implementation and a user study.

Stimulus-response model for smart home programming

The Push-pin system employs the stimulus-response model [7] to configure the relationship between appliances. All configurations are described using a “stimulus and response” pair. In the Push-pin system, a stimulus is generated in an input module when it is activated (e.g., when a sensor detects an occupant). An output module receives the stimulus, responds to it, and activates itself (e.g., turning on light). This relationship is configured by inserting a push-pin associated with the output module into the input module (Figure 2).

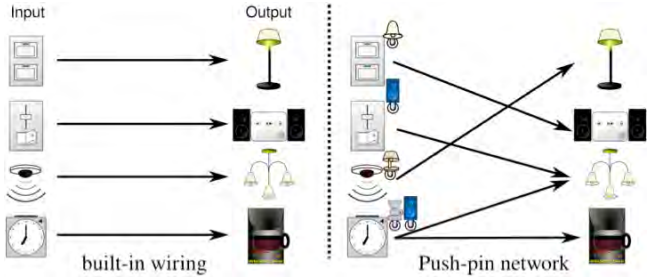

figure. 2. An illustration of stimulus-response relationship between input and output modules. Stimulus-response pairs can be found in traditional architecture, but their relationships are fixed. Push-pin system enables occupants to reconfigure them by putting push-pins to input modules.

The stimulus-response model is simple, easy to understand, and useful for designing home automation programs. Many home automation plans, such as “decrease the volume of the TV when I am on the phone”, “water the lawn when it’s not raining” and “show a reminder saying trash day every Monday.”[8] can be developed using this model with the help of various modules such as a calendar, a volume input,

and weather information service. The combination of sensors, home appliances, and stimulus-response rules enables feature-rich home automation using a simple programming model.

The Push-pin system can be used to program home appliances on the basis of the stimulus-response model using pins that are tangible and to specify logical relationships between home appliances. Because a pin is directly inserted into an appliance, the linking process is intuitive for the occupants.

figure 3. Push-pins prototypes. Right to left: floor lamps, circulator, TV and audio player.

SYSTEM OVERVIEW

The Push-pin system consists of modules, pins, and a network. There are three types of modules: output modules consisting of home appliances such as a lamp or a washing machine to be controlled, input modules consisting of devices such as a switch or a dimmer, and intermediate modules to specify advanced behaviors such as delay. All modules are connected to the network, and two modules are virtually linked to each other by a pin to establish a stimulus-response relationship between them.

Modules

Modules are basically categorized into two types, input modules and output modules. An input module consists of an input component (e.g., a button, a knob, a sensor), one or more pin slots, and a network adapter. An output module consists of functional devices, pins associated with the functions, and a network adapter. Each function of an output module has its own ID, and the same ID is embedded in the corresponding pin. By inserting the pin into a pin slot of an input module, it can be virtually connected to the output module.

When an input component is activated (e.g., a button is pressed), the input module reads the ID of the pin in its pin slot and transmits an event signal (stimulus) to the network with the corresponding ID. When an output module receives the signal from the network, it activates the corresponding device when the ID in the signal matches with its function ID (response).

Some appliances may function as both input and output modules because modules have multiple functions, and some of them stimulate while others respond. For example, a coffeemaker can start brewing when it receives a “turn ON” signal from its “start brewing” pin and may transmit a signal to notify another module linked to its “notify” slot when it stops brewing (Figure 4).

In addition to input/output modules, intermediate modules are used to design advanced automation program. Conditional branch, logic operation, or signal modulation can be implemented by introducing intermediate modules between input and output modules.

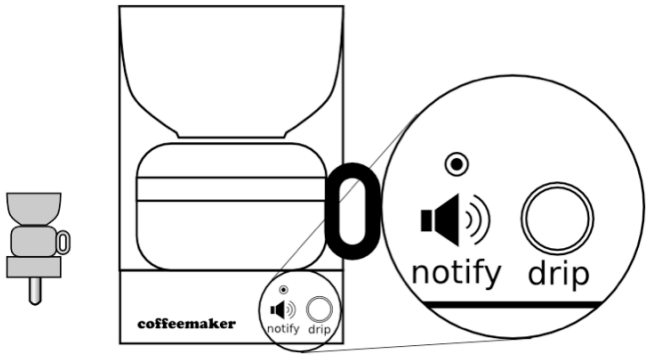

figure 4. Example of pin slot design of coffeemaker.

Pin/pin slot
Pins are provided with all output modules. The pins are designed such that the corresponding function can be easily distinguished by users. We envision that the size of the pin will be similar to that of real push pins. They will be neither too large to be obtrusive nor too small. Their size will be such that users can manipulate them by hand easily. Figure 3 shows the current prototype design of pins and modules.

An input module has one or more pin slots for each of its input components. When multiple pins are inserted into an input component, all of them will be activated in response to the same stimulus. An output module can have multiple pins to receive signals from them, but this setting can be ambiguous. In our current design, we interpret this as an OR condition. An event occurring at any input module activates the output module. AND condition is supported by a designated intermediate module.

Network and communication protocol
Modules communicate with each other via a wired or wireless network, or they may be bridged to each other as in INSTEON [9]. For a wired network, we plan to employ power line communication (PLC) that carries data along an electric power line like X10 and INSTEON. The advantages of PLC are that it uses the existing power lines in a house (no need to install additional wires) and provides power to modules easily. On the other hand, via a wireless network, modules can communicate with each other even if their power lines are different; their positions are not limited by the power lines, and they can controlled by a remote controller or mobile devices. In the current design of the message protocol, a message is represented as a packet, and it consists of a pin ID and payload data. The payload data of an event message consist of data-type information and a value. Currently, two types of data have been introduced. Binary status data are the signals transmitted from switches or motion sensors to turn appliances ON or OFF, play or stop an audio player, etc. Real value data are the signals transmitted from volume sliders or knobs to control the volume of audio players or the brightness of lamps.

CURRENT PROTOTYPE IMPLEMENTATION

Working prototypes of pins and input and output modules have been developed and tested using a stimulus-response model. Currently, pins and pin slots are implemented using 2.5 mm mono jack connectors. A resistor is embedded in a pin and its value is used as the pin's ID. Five unique IDs are implemented in the prototype. Current prototypes employ only wireless communication using ZigBee. Prototype modules have battery-powered control units. A control unit consists of

a micro controller (PIC 16F876), pin slots, and a wireless network module (Digi XBee Pro).

Modules

We developed switch panels, dimmer panels, PIR motion sensors, and clock modules as input modules and multipurpose outlets and melody modules as output modules. The current prototype clock module transmits an ON signal at a given time, but does not transmit an OFF signal. The PIR module transmits an OFF message after 10 seconds of the last motion is detected. To trigger multiple output modules simultaneously by an input module, multiple push-pins can be inserted in it. A multipurpose outlet module is used to control the appliance by closing and opening a relay in it according to the received signal. A miniature version of the connected appliance is used as its push-pin (Figure 3). The melody module plays a melody when it receives an ON signal.

CONCLUSION

In conclusion, we proposed a tangible programming system called “Push-pins” based on a stimulus-response model that enables end-users to program a home appliance network and home-automation systems using physical tags (push-pins). Using this system, users can interconnect appliances through pins. We presented the architecture of the Push-pin system, designs of programmable appliances. Some prototype modules have been developed to test this architecture.

Intelligent home appliances and built-in home automation system have become popular in the recent times. However, users find it difficult to reconfigure them or interconnect them cooperate. We believe in involving the occupants in setting up and programming their home appliances by using their knowledge and ideas because they are familiar with their living environment.

References

[1] M. Chan, D. Est`eve, C. Escriba, and E. Campo. A review of smart homes-present state and future challenges. Computer Methods and Programs in Biomedicine, 1(1):55–81, 2008.

[2] J. O’Brien, T. Rodden, M. Rouncefield, and J. Hughes. At home with the technology: an ethnographic study of a set-top-box trial. ACM Trans. Comput.-Hum. Interact., 6(3):282–308, 1999.

[3] M. C. Mozer. The neural network house: An environment that adapts to its inhabitants. In The American Association for Artificial Intelligence Spring Symposium on Intelligent Environments, pages 110–114, 1998.

[4] D. A. Norman. The Design of Future Things. Basic Books, 2007.

[5] S. S. Intille. Designing a home of the future. IEEE Pervasive Computing, 1(2):76–82, 2002.

[6] F. Kawsar, T. Nakajima, and K. Fujinami. Deploy spontaneously: supporting end-users in building and enhancing a smart home. In UbiComp ’08: Proceedings of the 10th international conference on Ubiquitous computing, pages 282–291, 2008

[7] D. Wolber and G. Fisher. A demonstrational technique for developing interfaces with dynamically created objects. In UIST ’91: Proceedings of the 4th annual ACM symposium on User interface software and technology, pages 221–230. ACM, 1991. 9

[8] J. Ledford. 25 Home Automation Projects for the Evil Genius. McGraw-Hill, 2006.

[9] P. Darbee. INSTEON: The Details. Smarthome Technology, 2005.