End-User Configuration of Ambient Intelligence Environments: Feasibility from a User Perspective

Panos Markopoulos[1], Irene Mavrommati[2], and Achilles Kameas[2]

[1]Industrial Design, Eindhoven University of Technology, Den Dolech 2, 5600MB Eindhoven, The Netherlands
P.Markopoulos@tue.nl

[2] Research Academic Computer Technology Institute, R. Feraiou 61, 26221 Patras Greece
{Mavrommati, Kameas}@cti.gr

Abstract. We report research into concepts and technology for enabling end-users to configure Ambient Intelligent environments. In this paper we focus on the feasibility and acceptability of this endeavor from an end-user perspective. We describe a conceptual model and an experimental enabling technology that illustrates the viability of these concepts and a multi-faceted evaluation of these concepts from an end-user perspective. Our work suggests the need for a flexible approach in letting users choose how much should be observable of system structure and function or of the processes of system learning and adaptation. Directions for future research in this field are described in the form of some provisional principles for shaping the interaction with end-user configurable Ambient Intelligence environments.

1 Introduction

The vision of Ambient Intelligence (AmI) promises that the environments where we work, relax or commute will be furnished with an increasing number of computationally augmented artifacts. AmI technology must fit seamlessly into the lifestyle and life-patterns of very different individuals and to adapt to situations and configurations unforeseen by their designers and developers. One potential solution is to support users to construct and customize their computational environments, as argued in [7]. This solution offers the benefits of an incremental and personalized construction of AmI environments, which empowers end-users.

The concepts and the technology discussed below extend the notion of component-based software architectures to the world of physical objects. Objects in peoples' everyday environment are augmented with autonomous computational artifacts, the e-Gadgets, which can be used as building blocks of larger systems. The computational environments formed by such artifacts are intended to be accessed directly and to be manipulated by untrained end-users.

Apart from the serious technical challenges pertaining to this vision, that are discussed in [4], an important research question that emerges is whether untrained end-

P. Markopoulos et al. (Eds.): EUSAI 2004, LNCS 3295, pp. 243–254, 2004.

users will be capable and inclined not only to program individual interactive systems but also to configure the environments they live or work in. The extrovert gadgets project (e-Gadgets) has developed concepts, a model and a prototype implementation to support this activity (see www.extrovert-gadgets.net). Below we describe this technology briefly and we focus upon an evaluation of the e-Gadgets concepts from a user perspective that tries to answer the research question raised.

2 Concepts and Terminology

An *e-Gadget* is defined as an everyday physical object enhanced with sensing, actuating, processing and communication abilities. A *GadgetWorld* is a functional configuration of associated e-Gadgets that collaborate in order to realize a collective function.

GAS-OS is the middleware that enables the composition of GadgetWorlds [4]. It is a component framework that manages resources shared by e-Gadgets, determines their software interfaces and provides the underlying mechanisms that enable interaction among e-Gadgets. The current version of GAS-OS is written in Java. GAS-OS supports IP-based communication using (without being bound to) IEEE 802.11g. For the prototype implementation, we used iPAQ handheld computers to execute GAS-OS. A special board has been designed that includes the hardware required to interface GAS-OS with the sensors embedded in artifacts.

A software tool, the GadgetWorld editor, has been developed to facilitate the composition of GadgetWorlds. The purpose of the editor is threefold: (1) to indicate/make visible the available e-Gadgets and GadgetWorlds (2) to form new GadgetWorlds (3) to assist with debugging, editing, servicing, etc. Two versions of the editor have been created. One with richer functionality runs on a laptop personal computer and is intended for the e-Gadget 'professional' designer. The second and simpler one runs on an iPaQ handheld computer and is intended for the untrained end-user.

GAS-OS supports the composition of e-Gadgets, without having to access any code that implements their interfaces. This approach separates the computational and compositional aspects of an application, leaving only the second task to the end-user. In this way, domain and system concepts are specified in the generic architectural model and are offered ready to the application designer and the end-user-programmer.

Plugs are software classes that make an e-Gadget's capabilities and services visible to people (through an editor) and to other e-Gadgets. For example, some of the services that the alarm clock can provide are: time, hour, minute, day, alarm on/off, sound; the lamp can provide such plugs as lamp on/off, light level. Composition is effected through the definition of *synapses* (links) between pairs of two (compatible) plugs. (See Fig. 1).

For example, consider a person who wants to achieve the following collective behavior from the objects in his/her environment: when the alarm clock sounds, the lamp on the ceiling of his/her room should be automatically switched on and the

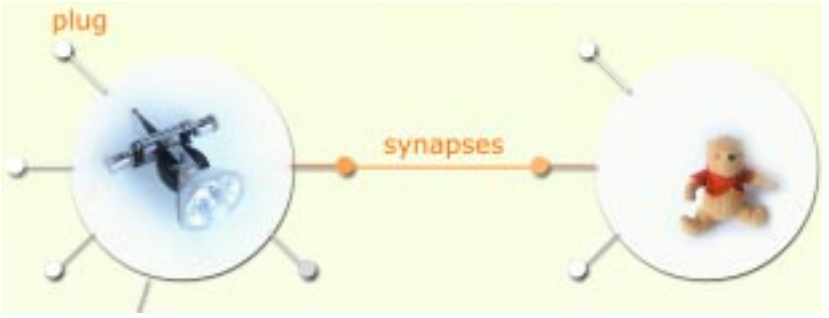

Fig. 1. E-Gadgets represented as circles, making synapses through their plugs to form GadgetWorlds

heater should start preparing the water for a shower. A novice e-Gadgets user could associate the alarm on/off plug of the alarm clock to the on/off plug of the lamp and to the on/off plug of the heater; a more experienced user could also use the light-level plug of the lamp or the temperature plug of the heater, thus programming the behaviour associated with a synapse.

Physically, the Plug/Synapse model is implemented with a peer-to-peer architecture. Each end of a synapse is managed by the GAS-OS running on each e-Gadget. A synapse serves as the abstraction of a communication channel between peers. The GAS-OS of the two e-Gadgets participating in a synapse will exchange events and data in a way specified by the adopted communication protocol; however, this occurs only when they have 'discovered' each other. Discovery is twofold: (1) on demand by an editor and (2) proactively carried out by an e-Gadget, after a synapse request. For example, if the light bulb of the ceiling-lamp is burnt-out, then the alarm clock e-Gadget may look for another e-Gadget that offers the lighting service; it can then switch this light on when the alarm goes off.

One important function of the editor is to identify and present to the user the e-Gadgets in its vicinity. It also helps inspect the capabilities offered by each device. Such capabilities have a direct relationship with the actuating / sensing capabilities of the objects and the functions that are intended by the appliance manufacturers. Some of these capabilities, especially the more complex ones, may not be obvious to people, apart from via the editor. In addition, the editor identifies the current configurations of e-Gadgets in its vicinity and displays them for supervision. The links between the compatible capabilities of appliances are visualized and can be manipulated through the editor. Associations between certain capabilities of appliances/objects can be formed thus creating configuration sets for a certain purpose. Once such a set of Synaptic associations is established, the part of the operating system that runs on each participating device will ensure proper operation. The set that is created remains operational as long as is required unless there's technical inability to maintain its functionality. Eventually a user may deactivate a GadgetWorld by destroying related associations.

Parallels can be drawn between the e-Gadgets editor and the Interstacks interface discussed in [6] for enabling end-users to integrate specialized hardware devices and

to control the flow of information amongst them. Both projects aim to support end-user configuration of a set of hardware components and component languages inspire both. Putting aside some significant conceptual differences from an AmI perspective that pertain to our ambition to convert every-day objects to computationally enabled artifacts, we note that from a user perspective there will be very similar concerns in designing appropriate editor interfaces. As no evaluation from a user perspective of the Interstacks project has been published to date, we hope that the evaluation discussed in section 3 will provide useful lessons for the design of editor interfaces like that of Interstacks or other comparable platforms.

In the course of the project 12 sample e-Gadgets have been created. Further, at the University of Essex, within the iDorm space (a laboratory resembling a student dormitory) several furnishings and devices run the e-Gadgets architecture. These e-Gadgets have been used as a test implementation of embedding the proposed platform into everyday objects.

Apart from users composing GadgetWorlds manually through an editor, the project has created prototypes where intelligent agents learn from people's use of a predefined GadgetWorld and proactively modify synapses between gadgets.

2.1 Example GadgetWorld Scenario

Lets assume a story involving the use of everyday artifacts. John 21, a student in economics, has recently created his Study Application (a simple AmI environment), with a new GadgetWorld he purchased. He wants his light to turn on automatically when he studies at his desk. This functionality can be supported by a GadgetWorld consisting in four e-Gadgets: a desk, a desk-lamp, a book and a chair. The book is equipped with a light sensor whose reading is made available through the plug 'luminosity'. When the lighting is lower than a certain threshold and someone sits on the chair and the book is open and is on the desk, the synapse with the on/off plug of the lamp will cause the lamp to switch on.

3 Evaluation of E-gadgets

An expert review workshop and an analysis based on the Cognitive Dimensions framework [2] were carried out to assess the concepts prior and during the prototype implementation. When a working system became available we sought expert feedback during demonstrations and we organized a formal evaluation where test-users created and modified their own GadgetWorlds. This work is summarized below.

3.1 Expert Review

Three invited experts in human-computer interaction (academics) participated in an early evaluation workshop, which had a general format of a focus group discussion. First, experts discussed the general concepts of the e-Gadgets project and a collection of 4 scenarios. Then they were given a problem-solving exercise for designing their

own GadgetWorld on paper. This exercise assumed 4 e-Gadgets: a desk, a lamp, a chair, a mat and a MP3 player. The experts sketched solutions for controlling the lamp and the players through the other e-Gadgets. After they designed such configurations they were asked to comment regarding the anticipated usage and acceptance problems of e-Gadgets technology. We then demonstrated two 'horizontal' prototypes of a GadgetWorld editor, i.e., providing an overview of the system rather than a fully functional segment of it. One was an HTML prototype running on a laptop and the latter was a video prototype demonstrating possibilities for a tangible interface to the eGadgets editor.

A wealth of formative feedback was generated in the expert workshop. In broad terms, the experts were concerned about how users would observe the invisible boundaries of GadgetWorlds and the logical connections between physical objects. Skepticism was expressed regarding the technical complexity given to users and the acceptability of agent technology in modifying the environment where we live and work in.

An interesting observation regarding the abstractions adopted by the project, was that human activity is modeled through information and behavior of e-Gadgets equipped with sensing behavior. On the other hand, humans themselves are a central part to any description of human activity and the context of operation for the e-Gadgets, so they should appear as "first class" abstractions in a vocabulary for describing and configuring AmI environments.

3.2 Analytical Evaluation Using Cognitive Dimensions

The Cognitive Dimensions framework [2] is a "broad-brush" technique for evaluating information artifacts, e.g., notations and interactive systems. It helps expose trade-offs made in the design of information artifacts with respect to the ability of humans using them to capture their concepts and intentions and to manage and comprehend the artifacts they create. Some of the most important conclusions from this analysis are summarized below, noting the relevant cognitive dimension where appropriate.

- The GadgetWorld editor should aim to bridge the gap between architectural descriptions of a GadgetWorld and the user's own conceptualizations, that might be rule-based, task-oriented, etc. (improving the Closeness of Mapping dimension).
- E-Gadgets require few conventions to be learnt (low terseness) and have an abstraction gradient favoring the non-trained programmer.
- E-Gadgets introduce hidden dependencies between the behaviors of apparently unrelated objects that should be made observable through the editor.
- An object may belong to several GadgetWorlds and its function is difficult to understand from its physical appearance (low Role Expressiveness).

With respect to the last two points, this analysis corroborated the opinion expressed by the experts that untrained users of e-Gadgets are handed programmers' tasks, so it would be inappropriate not to provide them with corresponding tools that help programmers carry out those tasks, e.g., libraries, debuggers, object inspectors, etc.

3.3 Demonstration Feedback at Conferences

The e-Gadgets technology was demonstrated at two international events attracting experts in human-computer interaction. Around 30 delegates experienced the demonstration at the "TALES of the Disappearing Computer" (Santorini, Greece, May 2003) and 10 completed the feedback forms. This event attracted delegates studying aspects of Ambient Intelligence (e.g., computer scientists, industrial designers, human-computer interaction experts) representing both industry and universities. Approximately 70 people visited the demonstration at the British HCI conference (Bath, September 2003) and we received 29 completed feedback questionnaires. The latter is a very specialized venue for Human Computer Interaction researchers, primarily representing the academic world.

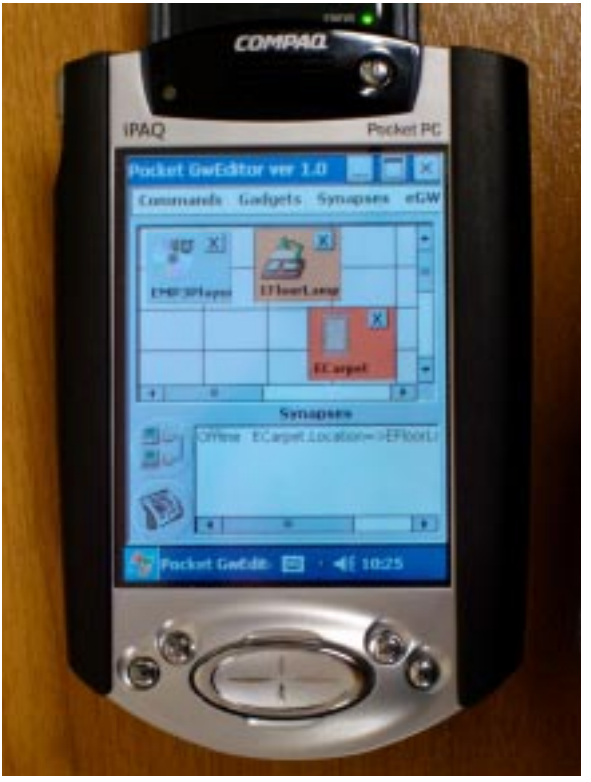

Fig. 2. The GadgetWorld Editor running on an iPaQ

The demonstration featured the handheld editor running on the iPaQ that supported the discovery of 3 e-Gadgets: a Mathmos "Tumbler" light (that resembles a luminous brick), a MP3 player and a pressure sensitive floor-mat. After a short explanation of 2-3 minutes, delegates were able to create a GadgetWorld, e.g., for controlling the volume and genre of music played from the position of a person on the mat or by flipping the Mathmos Tumbler on its different sides.

A wealth of comments was collected by written questionnaires; here we present only the general reaction to the e-Gadgets concepts, rather than more detailed comments about the demonstration set-up, the specific implementation or minor 'usability bugs' in the editors graphical user interface.

Some comments by respondents were conflicting: 13 delegates found that this technology will not be used because it is too complex, while 16 noted as a positive impression that is very easy to create and modify GadgetWorlds or that it is very easy to learn. Clearly, both are reasonable expectations that depend on the context of deployment and the targetted users. A lot of controversy was caused by the use of intelligent agents as an aid to configure environments. Some experts with Human Computer Interaction expertise rejected agent technology outright, others were enthusiastic and others pointed at some well-known caveats for adaptive systems from an end-user perspective (pertaining to the loss of control caused by agents as a result of reduced predictability and observability of system behaviour). 5 respondents suggested that a PDA is not a good platform to run the editor on, as it offers very limited display size. A couple of respondents pointed at two issues that run deeper into the concepts of the e-Gadgets project:

- The e-Gadgets abstractions refer to the system structure rather than the tasks of the user, so they have the onus of translating their intentions into architectural elements
- In actual life, an editor like the one shown during the demonstrations, would serve not only to inspect and to form one's own Gadgetworlds, but for people to inspect and understand Gadgetworlds purchased ready made or made by another member of the household. This means that a Gadgetworld architecture should not only be understandable to its creator but to other individuals as well.

A general point that relates to the latter observation is that the tasks of comprehending and modifying pre-defined configurations of components should be included in user testing of configurable AmI environments.

3.4 Evaluation at the iDorm

The iDorm facility is a specially constructed student dormitory that has been set up within a computer laboratory at the University of Essex, for experimenting with sensing technologies and intelligent agents. It is equipped with several sensing and actuating components, which for this study were controlled through GAS-OS.

The user study was a combination of short tests and a single trial that took place overnight. The short tests aimed to gauge how potential users grasp the fundamental concepts of e-Gadgets described above and whether they can create or modify their own GadgetWorlds. The overnight trial aimed to get a more realistic test of e-Gadgets. The user had to experience, albeit for a short time, the effects of 'programming' the environment. Because of practical constraints, we did not attempt repeated overnight tests, which however would have been preferable from a research perspective.

Participants

3 pairs of paid participants were recruited locally for the short tests. We looked for 'technophile' users with familiarity to computers. The pre-test questionnaire showed that participants were university students, 4 of whom computer science students, who were not familiar with the project. Only one participant was over 35 years of age. Only one participant was not a mobile-phone user. The rest were frequent users of personal computers, e-mail, SMS and mobile phones. In summary, all participants had a higher level of education and familiarity with computing than one would currently expect from the general public but pretty representative of the capabilities of potential early adopters for e-Gadgets technology.

Materials

The following e-Gadgets were made available for the user test:

1. Occupancy: Senses if the room is occupied or not.
2. LightLevel: Measures the ambient light in the room.
3. Chair: Senses if someone is sitting on it or not.
4. Bed: Senses if someone is on the bed or not.
5. Temperature: Senses the room temperature.
6. RoomLights: Switches room-lights on and off.
7. DeskLight: Switches desk light on or off.
8. BedLight: Switches bed-light on or off.
9. Blinds: Opens or shuts (completely) the blinds and lets you set the angle of the blades.
10. MP3 player: Sarts or stops playing music, sets the volume and lets the user choose a genre of music.
11. Clock: Tells the time or raises an alarm.

One of the authors acted as a facilitator and the other as an observer/note-keeper. The experimenters introduced the subjects to the experiment, explained the set-up and the nature of their involvement and obtained informed consent for videotaping. Participants filled in a pre-session questionnaire, describing their familiarity with computer technology in general and more specifically with handheld devices. A brief oral explanation plus a minimal demonstration of the system was then provided.

Participants were given the editor running on an iPaQ handheld computer (see Fig. 2). This editor supports discovery of devices that appear as a list. Through a set of pop-up menus operated with the stylus, the user can connect the plugs of two e-Gadgets to create a synapse. After creating a few synapses the user can actually run the configuration made. Note, that while some of the tasks involved control of light, sensing of movement, etc., which are typical examples for home automation our emphasis was different. We did not wish to test the acceptance of the home-automation functionalities used or to compare against purpose-specific software like X10. Rather we wanted to assess the way in which e-Gadgets are put together to form functional configurations.

One of the two participants would take control of the editor and was given advice how to operate it. Tasks were given one by one in cards. Due to technical malfunctions we adapted some tasks on the fly (during the tests the Blinds and the BedLight e-Gadgets ceased to operate). After the first task had been completed in this way, this 'trained' participant explained the operation to their peer who performed tasks 2-5 for

Fig. 3. The experimenter instructs a participant how to execute the first (training) scenario. Then she takes over to explain the editor to the second participant. Subsequently, the second participant will take over the editor.

the experiment (see Fig.3). This procedure is an adaptation of the Co-Discovery [5] and Peer Tutoring [3] methods for obtaining verbalisation data from usability testing. Getting this data was crucial for our case, where it was difficult for us to observe what they were doing on the handheld device (because of the small size of the screen). Some of the functionalities they managed to create were, for example, changing the genre of music depending on whether they would sit on the chair or not, switching the light off when there was nobody in the room, etc.

A mini-structured interview was conducted at the end of the session after which participants filled-in a written questionnaire with similar questions. This format (interview based on questions and then completing the questionnaire) was adopted to make sure questions were understood, to encourage participants to bring out opinions in the open but also so that they would use the opportunity to better formulate their thoughts in writing after the discussion.

Results of the iDorm Test

The short evaluation sessions went very smoothly, despite occasional minor technical failures. The overnight test was less successful, due to a network failure so few conclusions can be drawn from it, other than the importance of robustness and graceful degradation of AmI systems.

Test users, including the non-computer science students, were surprisingly capable in completing their tasks and surprised us with their enthusiasm. Participants reported that they enjoyed the type of programming activity; one mentioned that she liked "messing around with her furniture". We could also note the enthusiasm in their non-verbal behavior. All participants, despite some shortcomings of the editor's graphical user interface, found configuring a GadgetWorld straightforward.

This apparent contradiction to the opinion of several experts can be explained in two ways:

1. Our test participants were unusually well educated and familiar with technology (university students, some trained in computer science)
2. HCI experts overestimated the difficulty of concepts and interaction through a handheld editor. A possible explanation could be that younger adults (like our test-participants) are very adept with handheld technology and are as familiar with relevant interface conventions as most people are with graphical desktop interfaces.

Because of technical difficulties only one pair of subjects experienced the adaptation effected by the agent. An interesting observation we made at that point was that as soon as an agent was present, the test participants were not interested anymore in the structure of the system itself: Whether a synapse is there or not was not an issue anymore. Rather, understanding how the agent learns and what model it assumes about the user is more important. This feature of learning software agents seems to address the requirement for a task oriented language to communicate with the system and to help simplify the whole ontology that needs to be communicated to users. Subjects did not worry about the existence of the agent and the fact that their actions were being 'watched' by the device, but were disconcerted that they could not be aware of how much the agent has learned at any moment. Also, as one subject stated,

"...I don't really know how much control over it and if I cannot control it I would be afraid to use it. If I don't understand it I cannot control nor understand what it is doing..."

A range of comments was given by the participants, particularly suggesting improvements to the interface: e.g., referring to the terminology used in the interface, and making the workings of the system more observable and predictable and allowing also for task rather than structure orientated descriptions of system behaviors.

All but one participant said they enjoyed using the system and were very effective in achieving their tasks. One person felt that she needed more practice to grasp the concepts. We note that she was one of the least positive users about the whole experience. Invariably, users complained that not enough aspects of the physical gadgets were controllable by their digital manifestation: Once they had control over some properties they expected this to be extended to the rest. It seems an important and hard design challenge to convey the scope of the Gadgetworld both in terms of the editor interface and in terms of product design features for the e-Gadgets.

4 Discussion

The evaluation reported in the previous section has concluded in some provisional design principles, setting a direction for future work, in the domain of end-user configurable AmI environments:

- There should be several alternative ways for the users to articulate their intentions, e.g., both task based and structure oriented descriptions.
- End-user programming environments should offer representations of humans as first-class abstractions in the editing environment.

- An AmI environment should not surprise the user: Automation or adaptation actions should be visible and predictable, or at least justifiable.
- Intelligence should be applied only to simplify complex tasks.
- End-user programming of the user environment should be supported with similar tools as are offered to programmers, e.g., debuggers, object browsers, help, etc.

This list of principles focuses on the way in which end-user configuration of AmI environments should be supported. In this way it is complementary to works that attempt to characterize interaction with perceptive environments as, for example, the discussion presented by Bellotti and her colleagues in [1].

I noted the skepticism among HCI experts regarding the ability of end-users to grasp the concepts we proposed. The user-tests performed in the iDorm seem to appease the fears of an impossibly complexity, especially for new generations of users growing up surrounded by technology. However, scaled up user tests are required, both in scope and duration to gain more confidence in such a conclusion.

This research has made several inroads in the effort to empower people to actively shape AmI environments. It has demonstrated the feasibility of letting end-users architect AmI environments, though significant advances are still needed in engineering enabling technology.

From a researcher's perspective, we have demonstrated the value of the Cognitive Dimensions framework, as a tool in understanding interaction with AmI environments and we recommend its uptake in this field. Finally, the experiences reported suggest that an architectural approach where users act as composers of predefined components or by interacting with intelligent agents are two worthy and complementary approaches. Future work should explore their combination in a scheme that lets users choose and develop their strategy for composing a personalized AmI environment.

Acknowledgements. The e-Gadgets project was funded by the EU, the IST "Disappearing Computer" initiative. We thank fellow e-Gadgets researchers, the experts and test-users involved in our studies and the University of Essex for hosting the user tests.

References

[1] Bellotti, V., Back, M., Edwards, W.K., Grinter, R., Henderson, A., Lopes, C.: Making Sense of Sensing Systems: Five Questions for Designers and Researchers. Proceedings CHI 2002, ACM Press (2002) 415-422.

[2] Green,T.R.G., Petre, M.: Usability analysis of visual programming environments: a cognitive dimensions framework. Journal of Visual Languages and Computing, **7**, (1996) 131-174.

[3] Höysniemi, J., Hämäläinen, P., Turkki, L.: Using Peer Tutoring in Evaluating the Usability of a Physically Interactive Computer Game with Children. Interacting with Computers, **15(2)**, (2003) 203-225.

[4] Kameas, A., Bellis, S., Mavrommati I., Delaney K., Colley M., Pounds-Cornish A.: An Architecture that Treats Everyday Objects as Communicating Tangible Components. Proc. PerCom03, IEEE, Forth Worth (2003).

[5] Kemp, J.A.M., van Gelderen, T.: Co-discovery exploration: an informal method for iterative design of consumer products. In: Jordan, P.W., Thomas, B., Weerdmeester, B.A. McClelland, I.L.(eds.) Usability Evaluation in Industry. Taylor & Francis, London, (1996) 139-46.

[6] Lucas, P.: Interstacks. End-User "Scripting"for Hardware. CHI'99 Extended Abstracts, ACM Press (1999) 25-26.

[7] Newman, W., Sedivy, J., Neuwirth, C.M., Edwards, K., Hong, J.I., Izadi, S., Marcelo, K., Smith, T.F.: Designing for serendipity: supporting end-user configuration of ubiquitous computing environments. Proceedings DIS'02, ACM Press (2002) 147-156.