Enabling end users to compose Ubiquitous Computing applications - the e-Gadgets experience

Irene Mavrommati, Achilles Kameas

Abstract

A set of concepts which supports end-users in composing and configuring ubiquitous computing applications is described. The technology that implements the model is briefly presented; all have been developed and evaluated in end-user trials in the course of research during the e-Gadgets (IST-FET) project (www.extrovert-gadgets.net). This research has made several inroads in the effort to empower people to actively shape Ambient Intelligence environments; it has demonstrated the feasibility of letting end-users architect AmI environments. The experiences reported suggest that an architectural approach where users act as composers of predefined components is one worthy approach.

Introduction

The research project “extrovert-Gadgets (e-Gadgets)” [6] was conducted under the IST-FET-Disappearing Computer initiative. The research presented extends the notion of component-based software architectures to the world of physical objects, thereby transforming everyday objects into autonomous artefacts, or ’e-Gadgets’, which can be used as building blocks for larger systems. The computational environments formed by such artefacts are intended to be accessed directly and manipulated by people [2]. e-Gadgets has provided a set of concepts, a user model, an architecture and prototypical implementations to support them.

The vision of Pervasive and Ubiquitous Computing promises that our future environments will be furnished with an increasing number of computationally augmented artefacts. Ubiquitous Computing technology will need to be deployed and used in an immense range of different contexts to fit into the lifestyle and life-patterns of very different individuals, and must do so unobtrusively. Consequently, Ubiquitous Computing services and applications must be able to adapt to varying and changing situations, mainly determined by the places where they will be deployed and their specific conditions, which, for all possible instantiations are unforeseeable by their designers and developers. One potential solution is to enable users to configure, customise or even construct their Ubiquitous Computing applications [4], [2]. This solution offers several benefits: (a) application are made to fit in the best possible way to users’ own requirements; (b) applications can be incrementally improved by their very users; (c) people are able to shape their own environments instead of being mere recipients of ubiquitous technology. The first steps towards realising this approach is the design of a set of concepts that is common both to designers and users, and the provision of architectures and tools to implement them.

Examples of use

Imagine a large family house where the last person to leave has to lock the door and set the alarm. Instead of checking...
has to lock the door and set the alarm. Instead of checking all the rooms to see if someone else is still in, a link could be established between pressure-sensitive floors and the front door locking mechanism, whereby one could set the front door not to lock if someone is still somewhere in the building.

Another example taking place in the same setting is the baby that has just started to walk. Certain areas in the house should be beyond its access limits as they could be dangerous (i.e. the cellar that has a staircase). The floor can be set up so that when it senses movement in the ‘dangerous’ zone it sends out an alert to the parents, so they can make sure their child is safe.

Another tailor-made home facilitation would be for someone to establish associations between the alarm, curtains and coffee maker: for example, when the alarm clock rings, the bedroom curtains open to let the daylight in, and at the same time the coffee maker gets the signal to switch on and make fresh coffee (figure 4).

Now, let’s take a look at the life of Patricia, a 27-year old single woman, who lives in a small apartment near the city centre and studies Spanish literature at the Open University. A few days ago she passed by this store, where she saw an advertisement about “extrovert Gadgets”. Pat decided to enter. Half an hour later she had given herself a very unusual present: a few furniture pieces and other devices that would turn her apartment into a smart one!

Next day, she was anxiously waiting for the delivery of an e-Desk (capable of sensing light intensity, temperature and weight upon it), an e-Chair (which can tell whether someone is sitting on it), a couple of e-Lamps (capable of being remotely turned on and off), some e-Book tags (which are attached to a book, can sense whether the book is open or closed, and determine the amount of light that falls on the book), and an e-Carpet. Pat had asked the store employee to pre-configure some of the e-Gadgets, so that she could create a smart studying corner in her living room. Her idea was simple (she felt a little silly when she spoke to the employee about it): when she sat on the chair, drew it in towards the desk and then opened a book, the study lamp would be switched on automatically. If she closed the book or stood up, then the light would go off (she hadn’t thought of any use of the carpet, but she liked the colours).

For research purposes a number of domestic objects and furniture have been augmented with computation and communication capabilities and turned into e-Gadgets. This process of adding custom middleware to artefacts is a stepwise methodology that ensures that all the necessary hardware and software modules are installed in the object and initialised correctly.

In order to turn an everyday object into an e-Gadget, firstly one has to attach to it a set of sensors and actuators. For example, in order for the e-Desk (figure 1) to be able to sense weight, luminosity temperature and proximity, it has to be equipped with pressure pads, luminosity sensors and an infrared sensor. Pressure pads cover the underneath of the desk-top; luminosity and temperature sensors are evenly distributed on the surface. Infrared sensors are placed on the legs of the table and on the desk-top itself (figure 2).

The behaviour requested by Pat requires the following set of e-Gadgets: e-Desk, e-Chair, e-Lamp, e-Book. The collective function of this application can be described as:

When the particular CHAIR is NEAR the DESK, AND ANY BOOK is ON the DESK, AND SOMEONE is sitting on the CHAIR, AND the BOOK is OPEN, THEN TURN the LAMP ON.

In order to achieve the collective functionality required by Pat, the employee in the store had to create a set of ‘synapses’ among the e-Gadgets ‘plugs’ (figure 3). This type of functionality and component structure is created,
type of functionality and component structure is created, inspected and modified through the 'editor' (figure 6). For example, Pat can subsequently define the intensity of the e-Lamp when it's being automatically switched on. Or, if an intelligent agent is used, it could adjust the light intensity each time, based on the overall amount of light in the room, as it is recorded by luminosity sensors distributed on objects in the room.

Basic concepts and constructs

An e-Gadget is an everyday physical object enhanced with sensing, acting, processing and communication abilities. An e-Gadget is a functionally autonomous object, capable of managing its own resources (including power, processor, memory, sensors etc.) and of engaging in communication actions with other associated e-Gadgets.

e-Gadgets express their capabilities via the software construction of 'plugs' (figure 3). Plugs are software classes that make the e-Gadget's capabilities visible to people and to other e-Gadgets. Plugs can be connected together, by people, agents, or other actors, to form associative links (synapses) between e-Gadgets. A 'synapse' is therefore an associative link between two 'plugs'. This is briefly referred to as the "Plug-Synapse" model (figure 5).

A GadgetWorld is a dynamic functional configuration of associated eGadgets, which collaborate in order to realize a collective function. GadgetWorlds are the equivalent of software applications consisting of interacting components, exhibit collective behaviors and appear as functionally unified entities. A GadgetWorld can be considered as a distributed, ubiquitous computing application. Plugs and Synapses constitute the logical architecture of a GadgetWorld. In the general case, eGadgets form an ad-hoc network (hence the ability to support changing behavior), which becomes a P2P network when synapses are established.

A synapse serves as an abstraction of a communication channel between peers, as long as they have 'discovered' each other. Discovery may be forced upon an eGadget by an user creating the synapse with an Editor, or proactively carried out by an eGadget (for example when a light source breaks down, the switch connected to it may look for another light nearby, that can replace it). The option of an intelligent agent can be used to optimize GadgetWorld applications, by learning from the ways people use them.

The Gadgetware Architectural Style (GAS) [1] constitutes a generic framework shared by both artefact designers and users for consistently describing, using and reasoning about GadgetWorlds. GAS defines the concepts and mechanisms that will allow people (eGadget users) to define and create GadgetWorlds out of eGadgets, and use them in a consistent and intuitive way. GAS is expressed in the Plug-Synapse model, that is a conceptual abstraction that enables uniform access to eGadget services / capabilities / properties and allows users to compose applications; users form Synapses by associating Plugs, thus composing GadgetWorlds.

GAS-OS [1] is the operating system / middleware that manages resources shared by eGadgets, determines their software interfaces and provides the underlying mechanisms that enable communication (interaction) among eGadgets. GAS-OS also provides synapse management, discovery and routing services at the GadgetWorld level. Each end of a Synapse is managed by the GAS-OS running on each eGadget, thus implementing a peer to peer architecture [5].

Editing tools were developed, which are applications that support the establishment and management of ubiquitous applications [10]. The Editor was developed as an end-user tool to facilitate the composition of ubiquitous applications with eGadgets. It visualizes the available eGadgets and their functional configurations (GadgetWorlds); it can form
new application configurations; it can assist with debugging, editing, servicing, etc. With the Editor, people can supervise eGadgets and create/edit synaptic links between them [3].

This approach provides a separation between computational and compositional aspects of an application, leaving only the second task to application designers and end-users. The benefit of the approach is that, to a large extent, system design is provided ready to the end user or the application designer, because the domain and system concepts are specified in the generic architecture.

More than twelve sample eGadget artifacts and two sample interface versions of the Editor have been created. The eGadgets and Editors have been tested as stand-alone applications as well as within the setting of an intelligent space. One Editor runs on a p.c. / laptop while a second condensed version runs on an iPAQ handheld computer. Together with the explicit Plug Synapse model, the Editor is a tool that supports people’s creativity and innovation, by enabling end user programming of ubiquitous computing applications.

Apart from the serious technical challenges pertaining to this vision, an important research question that emerges is whether untrained end-users will be capable and inclined not only to program individual interactive systems but also to configure the environments they live in. The ‘extrovert Gadgets’ project has developed concepts, a model and a prototype implementation to support this activity (see www.extrovert-gadgets.net). After having described this technology briefly and we will focus upon an evaluation of the e-Gadgets concepts from a user perspective that tries to answer the above research question.

System abstraction related to physical affordances

In the Plug-Synapse model (figure 5) we adapt the physical objects to be able to act as components of an augmented Ubiquitous computing environment. We take advantage of their physical properties and characteristics (and translate them into the ‘plugs’ of their digital shelf), as well as of their physical affordances (that can be used via the ontology, which provides a second level of semantic interpretation of the physical characteristic). Thus the system abstractions of Plugs and the Ontology relate to the affordances given by the physical characteristics of the object.

By using the abstractions of Plugs and Synapses, one can combine the augmented objects and build new types of applications (by connecting their plugs together in synaptic associations, and describing the properties of these associations).

Objects can range, as well as the applications that can be created with this model. Objects can range from small scale (keys, lights, door handles) to bigger ones (Stereos, TV sets. desks, carpets) up to large ones (such as rooms, buildings, city squares, etc). Games, tailor made small scale applications, home applications for home automation, control, as well as larger scale applications for buildings (for example safety and security in communal buildings or hotels can all be addressed by this model.

By giving to the physical objects the possibility to act as components of an augmented environment, and to interconnect (via synaptic links) to form Ubiquitous applications, the objects acquire in fact new affordances, that are given to them via their digital self: the affordance to act as components, to act together in functional clusters (that are formed via the synaptic links). This new affordance of augmented objects is indicated to people via the Editor. In addition, as a designers choice this affordance of component-connectivity can also be indicated by elements of the physical design of object, such as physical, auditory, haptic or visual design elements for control and
feedback that visualizes the interface of the digital self of the object. Such elements depend on the exact nature of the object and result from its industrial/interaction design (this involves the specific design briefing and is dependant on the design constraints and the designers expertise).

**Evaluation from an end user perspective**

During its course, this research was evaluated in the course of user and expert trials. At first, an expert review workshop and an analysis based on the Cognitive Dimensions framework [7], were done to assess the concepts in the preliminary phases of the prototype implementation [8]. Then feedback was collected using a hands-on Demonstration (with three artifacts and an editor), that was shown in 2 conferences (DC-Tales and BSCHI 8, 2003). From both events, 29 completed feedback questionnaires were received from both these events, while aprox.100 people visited the stands spending 5-10min. to experience for themselves this technology. Finally, a short user evaluation was help in the iDorm: a specially constructed student dormitory that has been set up within a computer laboratory at the University of Essex, and is equipped with several sensing and actuating components, which for this study were controlled through GAS-OS. The study was a combination of short user tests (6 users in pairs have used eGadgets and an Editor to create applications in the iDorm, they had 2 hours per pair) and a single trial that took place overnight. This evaluation aimed to gauge how potential users grasp the concepts and whether they can create or modify their own applications (GadgetWorlds). The combined outcome of the evaluation indicated that the understandability of the Plug-Synapse model was high among users, and most were able to utilize the system and make simple configurations by using the concepts and tools provided.

One of the conclusions of the evaluations is that End-users programming their environment should be supported with similar tools as programmers, e.g., debuggers, object browsers, help, etc. Multiple means to define user intentions should be supported by the Editor, as people conceptualize their intentions in a variety of ways, which are not necessarily structural abstractions of the system. In the first evaluations skepticism was noted among HCI experts regarding the ability of end-users to grasp the concepts we proposed. The short user-tests seem to rest the fears of an impossible to use complexity. Although scaled up & longer user tests are required, to gain more confidence in this conclusion, this can hold true especially for new generations of users growing up surrounded by technology. All evaluation participants that had a hands-on experience using this technology familiarized with the concepts very quickly, within only a few minutes (5-10min.) of explanation. The majority of subjects succeeded in creating simple applications for themselves (using 2-3 objects with 2-3 connections), using the editor provided, in spite of the fact that the editor interface was at a very preliminary stage (the editor was used as a functional tool so as to test the research concepts, providing an appropriate level of robustness).

This user research has demonstrated the feasibility of letting end-users architect AmI environments, though significant advances are still needed in engineering enabling technology. In addition, we have demonstrated the value of the Cognitive Dimensions framework [7], as a tool in understanding interaction with AmI environments and we recommend its uptake in this field [9]. 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.
Value of the approach

The solution proposed by extrovert-Gadgets is communication (hence the term "extrovert"), as opposed to mere message exchange. Communication, according to Habermas, aims to achieve a shared understanding. Habermas says that one communicates in order to make known a desire or intention; then others can respond to the suggestion. In the e-Gadgets approach, a Synapse is formed as a result of negotiation among eGadgets. Negotiations and subsequent data exchange are based on ontologies possessed by the eGadgets; the only intrinsic feature of an eGadget is the ability to engage in structured interaction.

The value of the overall approach is that, via the e-Gadgets tools, artefacts are treated as reusable components (for designers and people). Ubiquitous computing artifacts that follow the proposed architectural style can be reused for several purposes, in order to build a variety of Ubiquitous computing applications.

This research has made several inroads in the effort to empower people to actively shape AmI environments and has demonstrated the feasibility of letting end-users architect AmI environments. The proposed model is easily comprehensible; therefore by the appropriate use of tools, the e-Gadgets technology can be usable by designers of Ubiquitous computing systems, but also by untrained end-users. Subsequently this approach opens possibilities for emergent uses of ubiquitous artefacts whereby the emergence occurs from people’s own use. Potentially it can enable the acceptability of Ubicomp technology into people’s environments, as well as enable the making of emerging niche applications.

References


This paper describes the set of concepts that underlie the Plug-Synapse model, which supports endusers in composing and configuring ubiquitous computing applications. The technology that implements the model is briefly presented; all have been developed in the course of research during the e-Gadgets (IST-FET) project.