Ghanam, Y., Shouman, M., Greenberg, S. and Maurer, F. (2009)
Object-Specific Interfaces in Smart Homes. Research report 2009-937-16, Department of Computer Science, University of Calgary, Calgary, Alberta, Canada, August.

Object-Specific Interfaces in Smart Homes

Yaser Ghanam, Maha Shouman, Saul Greenberg, and Frank Maurer
Department of Computer Science
University of Calgary, Canada T2N 1N4
{yghanam, mshouman, sgreenbe, fmaurer}@ucalgary.ca

ABSTRACT
As smart homes become more widespread, the need for more natural forms of interaction in the home also increases. On the one hand, forcing a user to go to a central home controller at every required change does not make sense. On the other hand, having a different controller for each device is also quite overwhelming. Object-Specific Interfaces (OSIs) allow an interface customized to an object of interest to show up on a mobile device when in close proximity to it. The interface can be passive, thus showing a manual or information, or interactive and so would allow for device monitoring and controlling.

Author Keywords
Smart home, controller, monitor, object-specific, interface, proximity.

ACM Classification Keywords
H5.m. Information interfaces and presentation (e.g., HCI): H5.2 User Interfaces: Input devices and strategies.

1. INTRODUCTION

With the ever changing dynamics of how people live and interact with surrounding environments, there is a rising demand for enabling technologies that support ubiquitous and domestic computing. These technologies include both software and hardware components. They are expected to have specific qualities that make them convenient, secure and reliable, especially when used in home settings. The complexity of the technological component in a smart home is directly proportional to the complexity of the expected scenarios. Functionality can be as simple as turning lights off and on using a remote control panel, or may be as complex as determining what mood an inhabitant is currently in and choosing the appropriate theme of light and music accordingly. When brainstorming about possibilities in smart homes, there are no limits but those of our imagination. Advancements in this field are made as a result of not only ideas triggering development of enabling technologies, but also due to emerging technologies revealing new possibilities. These emerging technologies are often claimed to make our life easier and our interaction with devices and commodities more intuitive. To some extent, this might be a valid claim. But when the number of smart devices starts to grow gigantically, users are often overwhelmed with the many things they need to learn and operate. The spacious variety of devices and functionalities needed in a smart environment makes it difficult to integrate all controlling terminals together into a single terminal. Especially given that these devices are often provided by different manufacturers. This imposes a requirement to have at least one controlling unit for each set of logically or physically connected devices making interaction more complex and cognitively demanding.

If we take a closer look at an average smart home, this problem is inarguably evident. In the living room, for instance, residents can set the temperature, dim the lights and control the blinds. For each of these tasks, there is a different remote controller, let alone other remote controllers for the entertainment center. Figure 1 shows three different remote controllers that can easily be found in one room to control different aspects.

Figure 1 – Different remotes to control different aspects

Furthermore, some smart devices in the home might have to be monitored or controlled through hardware or software interfaces. These interfaces, yet again, vary in look, interaction techniques, functionality and complexity. Figure 2 shows an example of a hardware interface that usually resides on a table (like a phone), and another that is to be mounted on a wall.

Figure 2 – Different monitors in the room (to the left: on the table, to the right: mounted on the wall)

In this paper, we present the first steps towards an approach to address the issue of multiplicity of controllers

and terminals in smart homes. We propose the use of proximity-based Object-Specific Interfaces (OSI) to be delivered to a single mobile device.

The rest of this paper is structured as follows. Section 2 provides a review of relevant literature. Section 3 is a discussion of the proposed approach. Section 4 looks at the system architecture of our implementation of the proposed approach. Section 5 lists a number of preliminary observations and limitations.

2. LITERATURE REVIEW

There have been previous attempts at creating and studying object-specific interfaces. For instance, [7] focused on interfaces of devices in the home which require scheduling. The study involved comparing the usability of four touch-screen interfaces for simple scheduling tasks.

Other implementations used browser-style interfaces in a home automation network [1,3]. In [3], the interface allowed users to browse through different devices in the home and interact with them accordingly. This architecture allowed for a URL to be provided for devices with individual interfaces [1]. On the other hand, places more focus on web-based wireless automation. Both these architectures use web access to implement UI access.

In addition, there were quite a few papers [2,4] that use Bluetooth in the home for appliance control and monitoring. The use of Bluetooth allows for a home Personal Area Network that includes all home devices.

RFID is another technology that has found its place in many Intelligent Homes. [8] used RFID to make their test-home more ubiquitous and context-aware. Another well known application is the use of RFID in museums [6] allowing visitors to further interact and explore exhibits.

Phidgets [5], or physical widgets, also have major applications within the home, as they allow for physical user interfaces that are easy to install and use.

Device-specific interfaces were also found in the literature. One patent [10] describes an interface that shows device controls and common instructions. Furthermore, [9] implemented a Personal Universal Controller (PUC) that allows the use of PDAs or smart phones to remotely control appliances. The interface of the PUC is automatically generated based on the appliance being controlled.

Finally, programmable switches are described in a patent [11] as wall switches that may be programmed to schedule lighting circuits.

3. THE PROPOSED APPROACH

The problem of the intricacy of interactions required in smart homes between individuals and devices is increasingly significant. The effort presented in this paper is an attempt to tackle one specific aspect of this problem. Namely, how can we minimize the cognitive load required of the individual to interact with the many devices in smart homes? Within the context of this question, we define two main criteria that we think constitute an effective solution:

1. The number of monitoring and controlling devices in a smart home is to be minimized while keeping intuitiveness and convenience intact.
2. Users should be automatically and promptly assisted when setting up or operating complex devices.

Our proposed solution is to provide a mechanism that enables delivery of Object-Specific Interfaces (OSI) to a mobile device equipped with a reasonably large and high quality screen. The delivery of OSIs is based on proximity. That is, when the resident comes close to an object in the home, the mobile device detects proximity to that object, and consequently pops up an interface. The interface should be adequate enough to enable interaction with the object, or give guidance on how to interact with the object. Theoretically, given any object in the home, this object can be associated with an interface that can be categorized as *informative* or *interactive*.

3.1. Interactive interfaces:

This type of interfaces allows the user to monitor or/and control the device they are close to. They provide the proper Graphical User Interface through which the user can see the current status of the device, or change it. One condition for this to be feasible is that the device itself is accessible and configurable through APIs provided by the manufacturer. In the intelligent home industry, it is common for manufacturers of smart devices to provide APIs in different programming languages. We have developed two applications in an effort to prove this concept. These applications satisfy the first criterion we mentioned earlier for an effective solution: “The number of monitoring and controlling devices in a smart home is to be minimized while keeping intuitiveness and convenience intact.” This is achieved by using a single device (the mobile device) to deliver interactive interfaces to. These interfaces, as will be shown later, are simple and can be reused for multiple devices.

3.1.1. Monitor, control and schedule light modules

This application enables users to monitor the current light intensity of the light module (desk lamp or living room lamp) they are close to, adjust this intensity to the preferred level, and create schedules for automatic

adjustment of intensity. The technologies supporting this behavior will be explained in later sections.

Figure 3 shows a snapshot of the screen that appears on the mobile device when it is near the lamp.

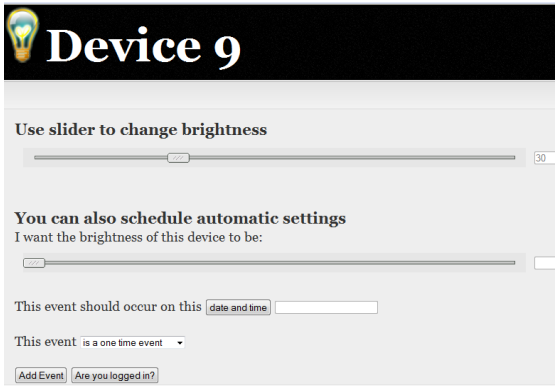

Figure 3 – Monitor and control light modules

Initially, the first slider shows the current intensity level of the device, along with an exact numerical value in a textbox. A light module, as specified by the manufacturer, can be off (intensity = 0), fully on (intensity = 99) or partially on (0 < intensity < 99). The user can change the intensity by dragging the slider to the right to increase intensity or to the left to decrease intensity. Through the same interface, the user can also configure the device to automatically adjust its intensity according to a specified schedule. The usefulness of this feature is twofold:

1. Some people often forget to switch lights off when they leave the living room to their bedrooms. They can schedule lights in the living room to go off at say 11:00 pm every day except in weekends.

2. There is a concern that when you go on vacation and leave your home empty, it might become a target for burglary. Scheduling lights to go on and off in a certain pattern gives the delusion that the home is not empty and thus decrease the possibility of a break-in.

Figure 4 shows a screenshot of how this feature is used. The user in this scenario wants the device to adjust its light intensity at 99 starting on December 11th, 2008 at 3:55 pm. This adjustment is to occur every two days until the user chooses to stop it.

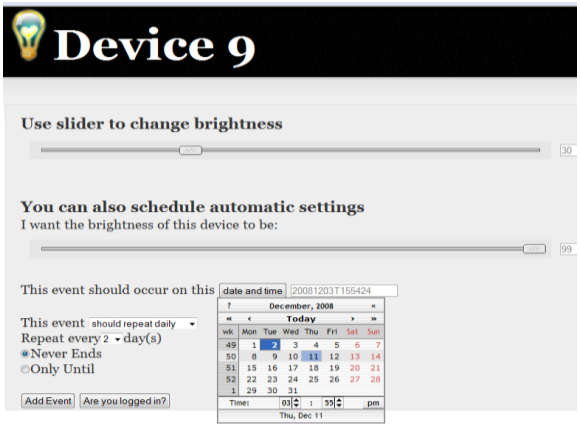

Figure 4 – Schedule light modules

In designing the scheduling interface, we were looking for simplicity and intuitiveness. Hence, we decided to use the same design style used in Google calendar [12]. We also used Google calendar as a backend to persist all scheduling events. This decision was made for a number of reasons. For one, since we chose to use the same UI style (same GUI elements and distribution of information on the screen) as Google calendar, it made sense to use their model to persist and retrieve the collected data rather than developing our own. Additionally, because scheduling entries are persisted and visualized as any other normal events in Google calendar, it is possible for users to remotely view, modify or delete these events using a normal web browser.

3.1.2. Programmable switches

In this subsection, we discuss another application that falls under the category of interactive interfaces. This application makes it possible for users to control the outcomes of changing the state of a logical switch in the smart home. By a logical switch we mean a switch that is not directly wired to an output terminal, but rather is connected through a logic box that acts as a multiplexer to deliver the output. Figure 5 shows an illustration of a two-state logic switch connected to two lamps. One setting, for instance, can be that when the switch is at state S1, the white lamp is on and the black lamp is off. When the state is S2, both lamps are off.

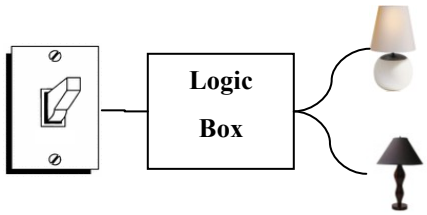

Figure 5 – A two-state logic switch

Giving the user the ability to easily configure a switch to produce a certain behavior can be very useful. Say there is a four-state switch in the living room that can control up

to 16 small ceiling lamps distributed in a 4 x 4 fashion. The user can define four different configurations: full lighting (all 16 on), no lighting (all 16 off), cinema mode (only 4 in the back on), and romance mode (side lights on).

In the application we developed, when the user comes near a switch, she is given the option (through an interface on the mobile device he carries) to configure that switch. Figure 6 shows a snapshot of this interface.

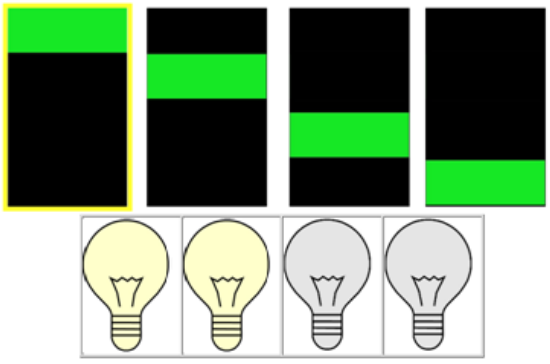

Figure 6 – Configuring programmable switches

This example shows a four-state switch that can be programmed to control four lamps in the living room (only four for simplicity). First, the user clicks on the state of the switch he wants to program. Then he chooses which of the four lamps should be on at this specific state. In this case, the user chose only the first two lamps to be on. The user can proceed to program other states in the same way. Since each one of the lamp icons is mapped to an actual lamp module in the living room, these settings take effect immediately. The architecture of this application will be explained in a separate section.

3.2. Informative interfaces:

The main characteristic of these interfaces is that the information flow is unidirectional. That is, these interfaces provide useful information about the object, but unlike interactive interfaces, they do not enable the user to monitor or control the status of that object. The reason these interfaces do not provide a mechanism to interact with the object can be either that:

(a) The object is natively static and does not change status like a statue.

(b) It is technically infeasible to interact with the object by any means other than the controller provided by its manufacturer. For example, consider an entertainment center in the living room. The user typically can use a control panel or a remote control to schedule the videotaping of his favorite TV show. The manufacturer of this entertainment center restricts interactions only via the provided controls of the same brand. Therefore, the OSI to be delivered upon proximity to that center is categorized as informative.

Informative interfaces can display instructions on how to deal with a specific device to achieve different objectives. Back to the previous example, coming close to the entertainment center, an interface automatically pops up on the user's mobile device like the one shown in Figure 7. The user chooses the functionality that he would like to get instructions on.

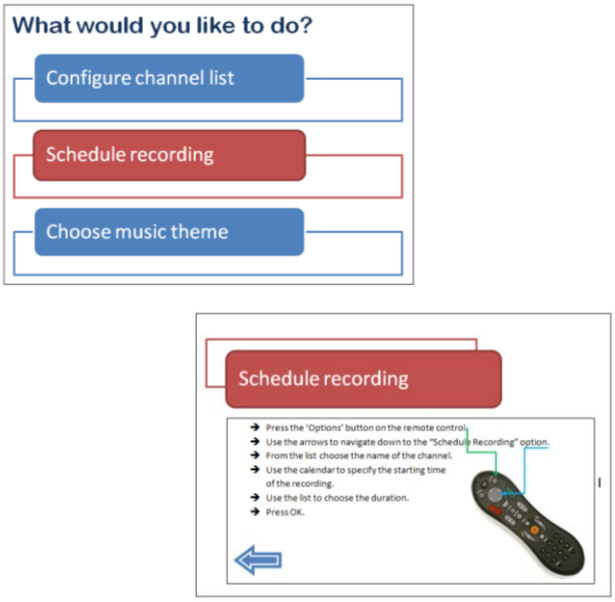

Figure 7 – An informative interface displaying instructions on how to schedule a recording

Informative interfaces serve the second criterion of the two we have previously mentioned: "Users should be automatically and promptly assisted when setting up or operating complex devices."

Beyond the focus of this paper, object-specific informative interfaces can be utilized to build a variety of domestic computer applications. For example, objects like souvenirs, mementos and statues can provide rich information on when, from where, and by whom the object was brought home. Another application would be associating reminders to specific objects like the front door or the kitchen table. When the person is close to these reminders, his mobile device displays the appropriate message on the screen.

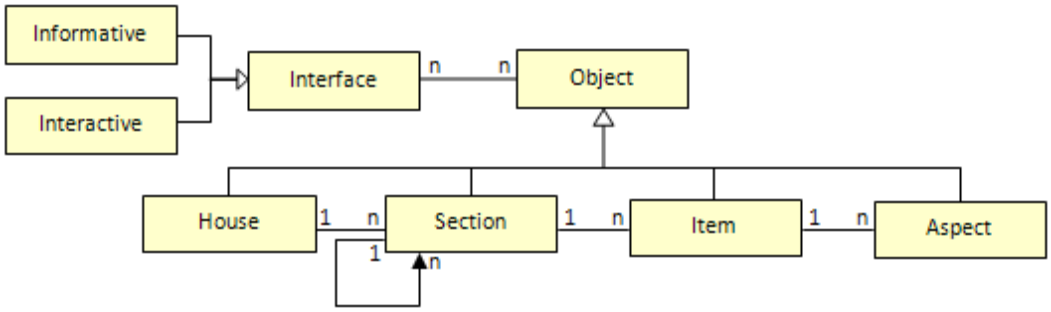

Figure 8 – OSI object model

3.3. Contexts:

So far in this paper, we have explained how OSIs can help decrease the cognitive effort required to learn and interact with the increasing number of devices in a smart home. We provided specific examples of relevant applications, where an interface is delivered to the mobile device when the user is closest to a given object in the home. The implicit definition of an object, as per the discussion so far in this paper, encompasses physical entities such as switches, light modules and entertainment centers. These entities are contained within other larger entities that are rather more stationary such as rooms. These rooms, along with a kitchen and a number of other facilities, are contained within the home. If we allow our definition of an object to be elastic enough to embrace these larger containers, then we probably can apply the same concept of OSI to rooms, facilities like the kitchen and the garage, and even the house as a whole. Figure 8 (at the end of the paper) is a conceptual model of objects that can be assigned OSIs. The main goal of this object model is to provide users with the level of context they deem necessary in a certain situation. An object can be the house itself, a section of the house (including floors, rooms, the kitchen, the garage, the basement… etc). Each section in turns can include one or more subsections (e.g. the basement has three rooms and a kitchen). Each section contains a number of items. These items can be statues, tables, lamps and other devices ranging in complexity. A single item might have a number of perspectives. A perspective of a given object can be an information holder that describes a specific aspect (or concern) of the object. Or it can be an aspect of the object that can be interacted with. To illustrate the idea of "contexts" according to our object model, let's discuss a concrete example.

When the user first enters the home, an OSI, similar to the one shown in Figure 9, is delivered to her mobile device giving her an overview of the home floor plan and its various sections.

Figure 9 – An OSI for the home object

The user can zoom into one of the sections, say Room 03, as shown in Figure 10. Notice that the same interface would automatically pop up on the mobile device when the user enters Room 03.

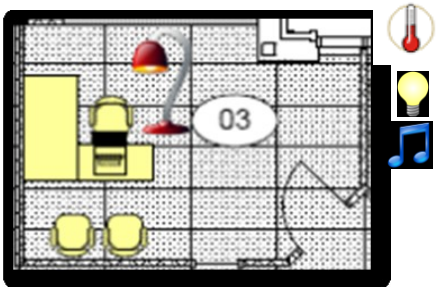

Figure 10 – An OSI for the room object

The user can choose to have a closer look at a specific aspect of the room such as the temperature, the lighting scheme or the music theme.

Moreover, the user should be able to see what items can be monitored or/and controlled in that room and navigate to them. The user can select the item of interest. If, for instance, the user selected the lamp, a screen similar to the

one shown in Figure 3 would be displayed. At any point, the user can zoom out to look at the larger context.

4. SYSTEM ARCHITECTURE

In this section, the technical details of the OSI implementations are described. All three architectures have two aspects in common:

a) *Identification*: All object identification, whether the object is a controllable device, a programmable switch, or a passive item, occurs through RFID. The objects in question are equipped with RFID tags, while the mobile devices have short-range RFID readers attached to them. When the mobile device is in close proximity to the object tag, a request is sent to the server.

b) *Network*: The mobile device and server are connected using an ad-hoc, wireless personal area network. The server is also connected to the internet.

4.1. Informative Interfaces:

Figure 11 shows the complete architecture for Informative Interfaces. As shown below, when the mobile device is within the range of an object's RFID tag, the mobile device places an HTTP request to the server.

Figure 11- Informative Interface Architecture

This HTTP request contains the tag ID the RFID reader has detected. The server then returns the appropriate manual or information, to be displayed as an HTML page on the mobile device browser.

4.2. Interactive Interfaces:

As was previously explained, there are two types of Interactive Interfaces. What they both share is that the majority of client-server communication occurs through AJAX, which contributes to providing the user with immediate and uninterrupted system interaction and control.

4.2.1. Device control and scheduling

This application starts out similarly to the previous one; a mobile device comes close to a tagged appliance, thus triggering an HTTP request to the server. This time however, the server sends back an interactive interface, rather than static information. This interface is also an HTML page.

Once the interface is displayed, the user has two options: control or scheduling. When the user moves the control slider, shown in Figure 4, an AJAX request is sent to a Web Service running on the server. As illustrated in Figure 12, the device in question is connected to a Z-Wave module, which allows for the control of the power input to the device. The server is connected wirelessly to this Z-Wave module, so that the web service called changes the device level according to the slider value the user has selected.

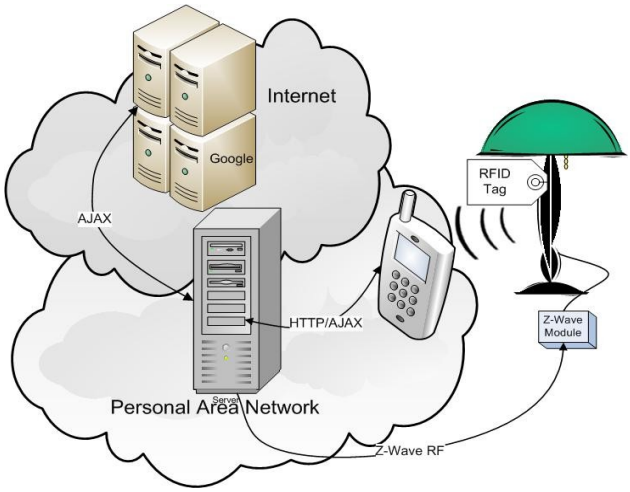

Figure 12- Device control and scheduling architecture

Scheduling the device is a slightly more complicated process. It starts out the same as control, with an AJAX request being sent to the server; however, the server response is different.

The web service called creates a new event in Google Calendar, using AJAX as well. The event is titled using the following format:

id'x' level'y'

Where 'x' is the device id and 'y' is the requested device level. In addition, recurrence options are added to the event depending on the user's specifications. Meanwhile, another process running on the server continuously polls the Google Calendar for current events. When the time for an event arrives, the server modifies the device level accordingly, also using Z-Wave Radio Frequency.

4.2.2. Programmable Switches

Programmable switches are somewhat more interesting since, unlike the previous application, the event input, i.e. the switch, and the event output, in this case the lights, are completely separate.

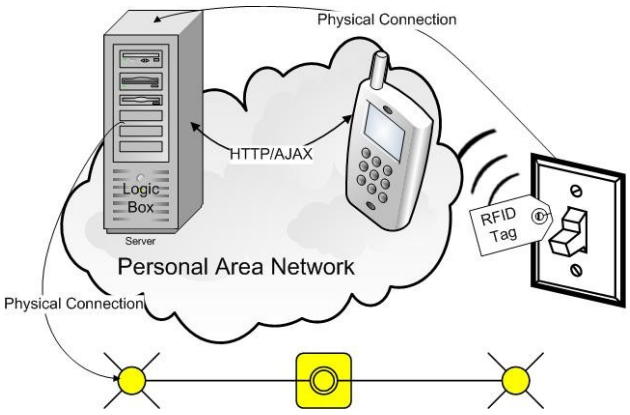

Figure 13- Programmable switch architecture

Events concerning switches are twofold. First of all, a user must program the switch output. This is done in a similar manner to the device scheduling, except that the switch settings are stored locally on the server.

The second part occurs when a user changes the state of the switch. This triggers the logic box listener on the server to lookup the previously stored switch setting and thus to modify the light outputs accordingly. As shown in Figure 13, both the switch and lights are connected to the server through physical wiring. However, there is no direct connection between the switch and lights.

5. DISCUSSION & OPEN QUESTIONS

As previously mentioned, the objective of the work presented in this paper is to help reduce the cognitive load associated with managing and operating the ever increasing number of devices and controllers in smart homes. In this section, we would like to reflect back on this objective within the context of the two-part solution we provided:

1. Minimize the number of monitoring and controlling devices while maintaining intuitiveness and convenience.
2. Provide timely assistance to users when setting up or operating complex devices.

Delivering interfaces to a mobile device that are specific to the object the user is closest to does indeed minimize the number of required controllers and monitors in the home. No longer does the user need to have a separate controller or monitor for each device (or group of devices in the home). Rather, the user uses her mobile device as a single terminal to monitor and operate devices of different complexities. This single terminal is also capable of automatically detecting which interface is to be displayed in a particular context. Additionally, it gives the user the flexibility to look at the specific object within a larger context, and from different aspects. In our judgment, this solution is more convenient than having to learn and use numerous controllers. Nonetheless, our solution assumes that the user should have his mobile device on him at all times to be able to interact with the smart home. We suspect that this might be an adoption barrier of the OSI technology. Having said that, it is too difficult at the moment to make accurate predictions of the acceptance of this technology, given that we have not tried it beyond a laboratory environment.

Furthermore, to a certain extent, the centrality of the mobile device in our solution might be another source of inconvenience. That is, what happens if, the user forgot where he put the mobile device? Traditionally, monitors and controllers usually do not leave the room where the associated object is. Controllers do get lost sometimes, but because every device is controlled through a separate controller, the loss of one controller is likely to affect only one device or in the worst case a group of devices. In our case, however, a malfunction or a loss of the mobile device means losing the ability to monitor and control all devices that rely on the OSI concept. Of course, this assumes that the traditional way of controlling objects and our solution are mutually exclusive. But this does not necessarily have to be the case. Our solution does not impose any constraints on the coexistence of multiple controllers for the same device. Also, if this solution is to be adopted, a wise step would be to have more than one mobile device in the home at any point of time.

Also, if we were to look closer into the intuitiveness of OSIs, we would notice a number of factors at play. For one, the design of the interface itself plays a key role of how intuitive the user would perceive the interaction. In our implementation, we tried to consider this issue by providing simple and self-explanatory interfaces. For example, in the design of the interface shown in Figure 4, we provided instructions on what can be done through this interface and how to do it. Specifically, in the scheduling part, we used a distribution of UI elements on the screen that looks similar to the one used by Google Calendar. However, we incorporated some changes to make the interface read more naturally such as:

(a) "This event should occur on this *date and time*"

(b) "This event *should repeat daily*"

Italic text shows changeable attributes presented as UI elements, namely, a calendar in (a) and a dropdown menu in (b). An open question that is yet to be addressed is how to give the user feedback on already scheduled events. A number of solutions were proposed, but we could not

determine which one was the most effective. One alternative was to show the scheduling entries as normal events in Google calendar. The problem with this solution is that it hides pieces of information that might be vital in the scheduling event. That is, when the user sees a weekly view of the calendar, he might not be able to tell if the event is a onetime event, a weekly recurring event or a monthly recurring event and so on. Also, the user cannot tell when a specific recurring event should stop. The other alternative was to provide the user with a list of scheduling events put in a natural language to give him a full view of all pieces of information, such as:

> "I want the brightness of the device to be at *50%*. This event should occur on *December 5th, 2008 at 12:00 pm*. This event *should repeat weekly* until *January 20th, 2009 at 8:00 am*."

Italic text resembles items that can be modified to update entries in the calendar.

The second aspect of intuitiveness is providing the user with timely feedback and immediate system response. We think that the current version of our implementation does not fully pass this criterion. Calling web services to execute commands that control devices seems to be too long a path. It takes several seconds for the command (e.g. moving the slider) to propagate through the network and get executed before any feedback can be observed on the light module. This delayed response sometimes wrongly gives the impression that the system is not responding. Users usually react by giving more commands (e.g. moving the slider again) to the system. Soon the server becomes overwhelmed with the many commands and the response becomes even slower. We believe that this is a purely technical issue that can be resolved in the future.

Regarding the second part of the proposed approach, OSIs do provide timely assistance when the user comes in close proximity to operate a complex device. An issue within that scenario that we did not touch on in our implementation is how to design and add these passive interfaces in the first place. Tool support might be needed to guide the user through a number of steps through which he can add a softcopy of a manual (maybe his own manual/comments) to the list of interfaces and associate these interfaces with the corresponding RFID tags. This is an open question to investigate in future research.

6. CONCLUSION

In this paper, we discussed a concept we called Object-Specific Interfaces. OSIs allow the user to use a single mobile device to monitor and operate different objects in the home. This is done through passive or interactive interfaces that are automatically delivered to the mobile device when the user is within close proximity to the object. As a proof of concept, we have developed a number of applications to better understand the capabilities as well as the limitations of our approach. While the proposed solution successfully addressed the issue of the increasing number of monitoring and controlling devices in the home, a number of questions are yet to be answered. A primary concern is with the intuitiveness of OSIs. Interfaces need to be simple and at the same time address the different needs of different devices. Moreover, technical implementation ought to be very efficient to smoothen the interaction.

The proposed approach is yet to be evaluated in real life settings. We believe that the current implementation is a first prototype that can be enhanced by a number of ways. But it is not very clear to us whether our system, in its current status, can endure thorough usability tests.

We think that our effort in this project is a cornerstone that can inspire the building of more sophisticated and usable applications in the near future.

REFERENCES

1. A. Alkar and U. Buhur, "An Internet based wireless home automation system for multifunctional devices," Consumer Electronics, IEEE Transactions on, vol. 51, 2005, pp. 1169-1174.
2. N. Sriskanthan, F. Tan, and A. Karande, "Bluetooth based home automation system," Microprocessors and Microsystems, vol. 26, Aug. 2002, pp. 281-289.
3. P. Corcoran and J. Desbonnet, "Browser-style interfaces to a home automation network," Consumer Electronics, IEEE Transactions on, vol. 43, 1997, pp. 1063-1069.
4. Y. Tajika et al., "Networked home appliance system using Bluetooth technology integrating appliance control/monitoring with Internet service," Consumer Electronics, IEEE Transactions on, vol. 49, 2003, pp. 1043-1048.
5. S. Greenberg and C. Fitchett, "Phidgets: easy development of physical interfaces through physical widgets," Proceedings of the 14th annual ACM symposium on User interface software and technology, Orlando, Florida: ACM, 2001, pp. 209-218; http://portal.acm.org/citation.cfm?id=502348.502388.
6. S. Hsi and H. Fait, "RFID enhances visitors' museum experience at the Exploratorium," Commun. ACM, vol. 48, 2005, pp. 60-65.
7. C. Plaisant and B. Shneiderman, "Scheduling home control devices: design issues and usability evaluation of four touchscreen interfaces," Int. J. Man-Mach. Stud., vol. 36, 1992, pp. 375-393.
8. T. Yamazaki, "Ubiquitous home: real-life testbed for home context-aware service," Testbeds and Research

Infrastructures for the Development of Networks and Communities, 2005. Tridentcom 2005. First International Conference on, 2005, pp. 54-59.

9. Nichols, J, and Faulring, A. "Automatic Interface Generation and Future User Interface Tools", ACM CHI 2005 Workshop on The Future of User Interface Design Tools, Pittsburg.
10. P. Gerety, K. Vij, Device interface controller having device specific and device common instructions separately stored, US patent 4800523, Patent and Trademark Office, 1989.
11. P. Chan, Programmable switch, US patent 4570216, Patent and Trademark Office, 1986.
12. Google Calendar, available at http://www.google.com/calendar. Last accessed Dec 5th, 2008.