Analysis and Requirements Gathering of User Interfaces for Home Automated Systems

Akash Bhrat Manilal[1]
[1]*Department of Computer Science and Engineering, Technical University of Lisbon, Lisbon, Portugal*
{*akash.manilal*}*@tecnico.ulisboa.pt,*

Keywords: User Interface, Usability, Automation Systems, Patterns, Interface Analysis.

Abstract: As Home and Building Automation Systems become more widespread, the need for interaction with devices increases. However, many of these technologies are brittle and do not adapt to the user's explicit or implicit wishes. The variety of devices and functionalities required in smart environments present plethora of User Interface elements that are difficult to integrate under a single interface. Indeed UI are confusing, overwhelming and hard to operate. Literature is scarce and there is no clear guidance when it comes to designing a UI for Building and Home Automation Systems. This document analyses existing User Interfaces for Home Automation Systems to uncover their fundamental aspects of interaction. Backed by well accepted heuristics on UI design, this work systematizes and validates the main aspects of UI for interactivity with HAS devices, defining a set of device behaviour feedback requirements.

1 INTRODUCTION

Home Automation Systems (HAS) provide advanced functionality to homes using distributed control systems. Demand for energy efficient solutions, enhanced security, increased venture capital funding, and greater convenience are some of the factors that have led to the growth of the home automation and controls market. Moreover, automation system vendors are proposing integrated virtual interfaces that enable actuating on devices from a single point of control; Nowadays, turning a light on, increasing the TV volume or even heating the oven can be accomplished from a single application. As a consequence, a larger number of controls is necessary to command all devices resulting in cluttered user interfaces with lack of usability.

This document will identify the most frequent feedback problems related to HAS User Interfaces (UI) and find solutions that the User Interface can provide in order to portrait accurate information regarding the devices status feedback. We will also analyse the highest ranked HAS, describing their functional requirements, levels of usability and usefulness of their user manuals, tutorials, forums and any other documentation they possess. An evaluation will also be made of each one in order to identify weaknesses and strengths of existing applications. Furthermore, functional requirements and layout specifications for each interface will be described so that we can design User Interface mockups to be later evaluated by users. Those mockups will include the solutions and specifications previously defined.

This project is being developed within the scope of the SMARTCAMPUS[1] project.

1.1 Motivation

HAS has been evolving in terms of functionalities, performance and types of technologies it can control to satisfy users. Nevertheless, there are still issues regarding User-Machine Interaction that have not yet been addressed. The main concern relies on the User Interface and the fact that they have not been evolving at the same pace of other fields. This section presents some scenarios that will illustrate problems which served as motivation for writing this document.

When a user makes a request to the system, it has to go through four different phases: The first one is acknowledging the request from the UI, then the request is sent to the respective gateway. In phase three the request reaches the right equipment and finally, if there is not any obstacle preventing the machine from completing the request, it is executed. Consider a command that turns on the lights of a certain room but it fails. Our first scenario focus on how are we go-

[1]http://greensmartcampus.eu/

ing to show the user where the error occurred. What kind of message can we show? How is the user going to understand where it went wrong? There is a need to show a progress and completion status for each request.

Another topic most of the existing HAS in the market do not take into account is distinguishing the types of request users can make. They might be aimed to the space around the user or in some cases, they would rather request a service and let the system decide. Consider a user that wants to read a book. If he makes a request to the system claiming that he needs light to read a book, the system itself can decide whether to open the blinds or turn on the lights.

All of these scenarios served as motivation to this work. We will present possible solutions for them. Additionally, we will show the concepts of other fundamental aspects of interaction with HAS.

1.2 Problem Definition

Most Home and Building Automation Systems focus on functionality gathering, house features, integrating with external systems or even improving response time. Even though they are important aspects, they conceal other requirements that are also of major importance to the correct functioning of systems and acceptance from users such as usability, ease of use, screen organization and device behaviour feedback.

Our biggest issue relies on the fact that existing HAS do not take into account those problems and the few that do, do not obey or follow any type of standard or rules because it does not exist.

We can define our problem with the following statement: There are no good practice guides regarding User Interfaces for Home Automation Systems. There is also no standard when it comes to feedback error handling given by the system.

What it is proposed in this document, in order to solve the issue we have in hands, is to analyse existing systems, retain their weaknesses and strengths, identify all possible feedback problems Home Automation Interfaces can have and along with information regarding functional requirements and layout specifications, build a User Interface that includes all the information previously gathered.

2 CONCEPTS

User Interface is the visual part of a computer application through which a user interacts with a software. It determines how commands are given to the computer or the program and how information is displayed on the screen. However, the field that studies how User Interfaces should be designed is much wider than its definition. In this section we will introduce the concept of Smart homes and Human-Machine Interaction. Both of them will be related with UI designs, referencing patterns and different type of User Interface usability evaluations.

2.1 Smart Homes

Smart homes go by several names, including Integrated Control Systems (ICS) and Home Automation Systems (HAS). By any name, these systems are used to control devices around the house. Home electronics and appliances including doors, lights, surveillance systems and consumer electronics are some examples (Humphries et al., 1997). Control Systems can also provide information, meaning, users can find out how much electricity they have used on specific appliances and utilities can read meters remotely (Han et al., 2010). An important goal of smart home research becomes how to appropriately expand system capabilities to produce more control – both perceived and actual (Davidoff et al., 2006). To provide mobility, usually, users are able to control with a smart-phone or tablet. However, these systems lack user friendliness and are neither intuitive nor realistic (Wimsatt et al., 2006).

Domestic technologies have been adopted at different rates throughout the years. To better understand the direction smart home technologies are taking and what programmers and scientists believe is the next evolution for HAS (Harper, 2003), it is important that we distinguish the concepts of *time-saving* goods and *time-using* goods (Bowden and Offer, 1994). The first ones are meant to reduce the time needed to complete a task, in order to have more time for oneself. Vacuum cleaners, washing and drying machines are some examples. The majority of these goods took decades to diffuse and all of them are clearly related to household income. As for time-using goods, they occupy free time and improve its quality by giving users more to be entertain with. Television, radio and video are the most common and well-known time-using goods for most people. Unlike time-saving goods, these have spread quickly and showed to be less related to household income.

The relation between these two goods rely on the time they occupy. The more people spend time using goods like television or radio, the less they have to complete house tasks. This can be considered the motivation needed to start thinking about smart homes.

2.2 User Interface Usability Evaluation

The term usability was firstly introduced to replace the term User Friendly which, around the early 1980's, had a more vague and subjective connotation (Bevana et al., 1991; Spencer, 2004). We can define usability as a function of the ease of use, learnability (when relevant), and acceptability of the product. It determines the actual usage by a user for a particular task in a specific context. Ease of Use determines whether a product can be used, acceptability decides if and how it will be used. Attributes of a product, performance and satisfaction measures the ease of use in a particular context. The context consists of the user, task and physical/social environment.

We can analyse UI's through analysis techniques, computerized procedures, empirical methods and heuristic methods (Nielsen, 1994a; Nielsen and Molich, 1990). However, not all of them have been referenced as real testing to usability when it comes to user interface. The following list shows a set of User Interface Usability Evaluation: Questionnaire for User Interaction Satisfaction (QUIS) (Chin et al., 1988) was designed to assess users' subjective satisfaction with specific aspects of the human-computer interface. The QUIS team successfully addressed the reliability and validity problems found in other satisfaction measures, creating a measure that is highly reliable across many types of interfaces; Perceived Usefulness and Ease of Use. What causes people to accept or reject information technology? Among the many variables that may influence system use, previous research suggests two determinants that are especially important. Perceived usefulness is defined here as "the degree to which a person believes that using a particular system would enhance his or her job performance while Perceived ease of use, refers to "the degree to which a person believes that using a particular system would be free of effort (Davis, 1989); Attributes of Usability, according to Jakob Nielsen, is defined by five quality components (Nielsen, 1994b): Learnability, Efficiency, Memorability, Errors and Satisfaction;

Nielsen's Usability Heuristics is one of the most used guidelines to verify whether the user interface is well evaluated, usability wise (Nielsen, 1994a); After Scenario Questionnaires is to be given to a study subject after he/she has completed a normal condition scenario (Lewis, 1995). The user is to circle their answers using the provided 7-point scale (the lower the selected score, the higher the subject's usability satisfaction with their system) and the Wizard of Oz user-based evaluation of unimplemented technology where, generally unknown to the user, a human or team is simulating some or all the responses of the system (Bernsen et al., 1994). Some advantages of this evaluation are testing future technologies without building an expensive prototype, "filling in" functionality that is not yet ready for a prototype, rapid iterations, particularly minor changes in wording or call flow, are immediately testable and it allows the system to be evaluated at an early stage in the design process.

2.3 Feedback Issues

Feedback is the process part of a chain cause-and-effect. Wondering what a software program is doing is a common way to confirm that UI lacks clear information. Feedback principles in a UI needs to be immediate and synchronized with user action. As for Home and Building Automation Systems (HBAS) there are specific sets of responses an UI should give to their users.

Hysterisis is the dependence of a system not only on its current environment but also on it past one. This dependence is due to conflicts on internal states. To avoid conflicts, its history must be known (Visintin, 1994). We can identify this problem every time a user requests a functionality that is exactly the opposite of the previous one in a short amount of time. Opening and closing blinds and turning the heater on and off are some examples of such problem. To an HBAS it is relevant to determine whether the application should not allow the user to complete his task or simply showing a warning message to acknowledge the user that his request might cause conflicts.

Other kinds of feedback issues come from Progress and Completion of tasks. For the first one we can show the user an accurate progression of the requested action by measuring the average time to complete the activity (Best Effort). Once the average time has passed, a message is shown to the user, confirming the completion of the task. This approach mislead the user since we cannot be certain the task is already completed. The second option is to give the Feedback Bus the responsibility of sending an acknowledge signal (ACK) to confirm when the task is completed. The last choice and the most common one, is using a motion sensor in order to get physical feedback. This approach can cause issues if an object is preventing the motion sensor to read correctly.

Manual Override is an action where the control is taken from an automated system and given to the user. When building HBAS, the main goal is to give the user as less work as possible to perform tasks. However, the system should be prepared to deal with exceptions. On extreme cases, giving the machine full responsibility for automations of all features of the

house can become an issue. If they follow rules installed by the manufacturer or required by law and refuse to cede control in some situations then the owners of the devices may feel less empowered, alienated and lacking true ownership.

3 RELATED WORK

User Interfaces have always been a topic of interest when related to HAS. Systems have to take them into account because they are the biggest way of interaction between users and machine. In this section we will address theoretical studies related to HAS characteristics and their User Interfaces. Firstly we will present the notion of portable systems and remote access. Then we will show some systems capable of controlling various house features with only one UI and discuss customisable UIs.

In 1996 a method for menu-driven UI, more particularly, for distributing menus throughout a home, was presented (Fujita and Lam, 1996). In order to make the software portable, functions could be implemented using personal computers, infra-red, remote controls, home distributed networks and television receivers. This invention was advantageous in a way that menus could be distributed to any place in a home, users could control the home system and outside services from any display location in the home and different menus could be distributed at the same time to distinct display locations.

In 2005, a mobile-based HAS solution was introduced (Van Der Werff et al., 2005). The design consisted on a mobile phone with Java applications, a cellular modem, and a micro-controller. User friendly graphical UI was provided on the mobile phone through applications developed in Java programming language. The controller board resided at home and worked as a home server, which carried out the task of operating and monitoring house appliances. The home server communicated with the remote control via cellular modem.

Liu and Xian built a system (Liu and Xian, 2007) so as to merge all house features. To enhance the experience, a voice controlled, User Friendly UI was carefully designed, so that user could use not only standard keyboard and mouse, but also the voice to control home environments including lights, TV, radio, VCD/DVD player, fan, and air conditioner. Furthermore, the self-designed software applied home automation control ideas to internet access and PC application software access with the features of surfing the internet, sending and receiving emails, using other PC software such as Microsoft Office.

Table 1: Systems Comparison. Each column represents the dimensions analysed while the lines represent the systems/works mentioned in the previous sections.

	Portability	Remote Access	All-in-One	Custom UI
Menu Driven UI	+	--	+/-	--
Mobile-based HAS	+	+/-	+	--
Zig-Bee	+	+	+	--
HAS for Disabled	+	--	+	--
HAS Contextual UI	--	+	+	+

A report by William Wimsatt went further and in addition to gathering features, developed a contextual UI (Wimsatt et al., 2006). Contextual Interface is a product of a design process in which the various interface features are selected to improve the users ability to operate the interface. In most graphical user interfaces these items remain static so that irrespective of the context, a control such as a *start* button, remains in the same place on a display with the same appearance and performs the same function.

3.1 Discussion

In order to identify the common issues and strengths of each work described during this section, a comparison was made. Table 1 shows the results of the analysis based on the details of each work. To facilitate the comparison, a convention of symbols with different meanings were defined: ”+” means the system fully supports the feature to be evaluated; ”+/-” means the system partially supports the evaluated feature; and ”–” stands for either the system does not support the feature that is being tested or the work does not present any information regarding the evaluated dimension.

Examining the table we can see that each dimension has different results. Regarding portability, we can verify that all of them, excluding the work of William Wimsatt (Wimsatt et al., 2006) (which does not present any information related to this dimension), are portable. The systems were built in a way that allows users to access it in different devices. Some of them use infra-red, remote controls and home distributed networks to accomplish this feature. In contrast, almost none of them are customisable.

Forcing a user to have a system that controls irrigation, another that controls water and another for illumination nowadays is not viable. Companies understood this problem and made an effort to join all those systems into one, requiring the user to adapt a single UI. According to Table 1 we can see that all of them took this problem into account and gather all the functionalities.

The last analysed feature was the remote access. In the last few years, companies have been trying to make their software as accessible as possible, using the network as an asset. According to the table, results

Table 2: Overall systems comparison. Horizontally, an individual score is presented, according to each dimension of evaluation. the last cell of each line represent the total score of each system. Vertically, the last cell represent the sum of all the points acquired in a specific dimension. Darker cell represents the best dimension and system. Grey cell represent the system and dimension with the lowest total score.

	Points of Evaluation				
Products	System Features	User Adapatability	Ease of Use	Help and Documentation	
mControl	5	3,5	4,5	3,5	**16,5**
HomeSeer	5	3,5	5	5	**18,5**
Control 4	5	3	4	2	**14**
PowerHome	4	2,5	2	4	**12,5**
Vivint	5	3,5	4	4	**16,5**
ActivateHome Pro	2	2	1,5	5	**10,5**
	26	**18**	**21**	**23,5**	

vary from work to work.

4 EVALUATION OF EXISTING SYSTEMS

HAS has been growing exponentially in the last years. The proof relies in the various Automation Systems available in the market. For the past few years there has been an increase of requests to implement Integrated Control Systems (or Smart homes) and at the same time, an increase of companies capable of providing and supporting those systems. In order to evaluate existing systems it was decided to choose six HAS that stood out among the others: mControl, HomeSeer, Control4, PowerHome2, Vivint and ActiveHomePro.

4.1 Results and Discussion

After the previous analysis it is clear that most of the systems include, either in modules or as a whole package, all the house features we were looking for. We need to take this information into account since our solution will be based on what most system possess.

According to Table 2, the low scores presented on user adaptability are due to the lack of flexibility regarding UI customization. None of the presented systems offers that option to the customer. Users should be able to customise as they see fit, in order to get to the functionalities they use the most the fastest way. The other reason making this point with an average score is the fact of not being session-based applications. Most of the systems allow the user to access the software remotely. However not all allow voice commands. Tabs seem to be the preferred way to organise functionalities.

Finally, the last point analysed, help and support, showed to be a bit controversial. There are some systems with low scores on system features, user adaptability and ease of use but offer great ways to solve problems. The less the user uses help and documentation, the better he knows how to use the software. This does not mean we can ignore documentation or warning messages. We have to let the user know what he is doing in order to avoid making him search for the answers.

5 SOLUTION

The knowledge acquired from researching the current literature and evaluating existing systems led us to define a set of requirements for device behaviour feedback and layout specifications, leading to the constructions of a interface that meets all requirements previously defined. Additionally, functional requirements and a possible flow of events inside an application were proposed.

5.1 Device Behaviour Feedback Requirements

Before designing or implementing any type of interface, we thought it would be necessary to understand the types of behaviour devices can take. With that being said, it is important to describe, for each type of device status, solutions that the User Interface can provide in order to portrait accurate information regarding the devices status.

When we talk about device status there are at least two that come into our mind: ON and OFF. However, and considering more complex devices, there are far more status devices can take. On that note, it is important to know how the equipments are going to send their status to the system in order to show in the interface. There are six types of status feedback devices can take: unknown, received long ago, to be confirmed, known but estimated, known and confirmed.

For each one of them the User Interface should present a different message so the user does not have doubts on what he is looking at. In this case, pop-up messages and tags were the best proposals we could present.

Regarding device status provenance, it is important to distinguish the various status into three different divisions: the source that sent the command that defined the current state of the machine, the type of command (manual, automatic, etc.) and the context in which it operates. For each one of them we can come across problems like: not knowing the source of the command, the command was triggered by another user or being part of a group command context. On a more extreme scenario, we can have problems like not knowing the context of where the command was

requested. For all those status, we endorse a solution that despite being simple, it can be highly effective: tags.

Lets focus in the effectiveness of the requests made by the users. Picture a scenario where the user sends a request to an equipment that he cannot confirm whether it was successfully completed or not. Assuming the command was properly sent by the interface, there are three main confirmation steps it has to go through in order to assure the action was effectively taken by the device Reaching the gateway, the device and Action taken and Acknowledged by the system. For any of these cases, our solution proposal is to show an animation of the route the command is taking.

Hysterisis, as defined in the concepts, is the dependency of the output of a system, not only with its current input but also on all the previous ones. The last ones are exactly the problem makers. With that being said, we need to define their status in order to accurately show the user what is going on with his request. We managed to identify the following device status: initializing, shutting down, fading in/ramping up, fading out/ramping down, in progress, finished and hysteresis period.

For error/warning conditions we have established that messages - either pop-up, sliding or any other type - are the best solution to let the user know the problem that is keeping the system from fulfilling his request.

Manual Override is a mechanism wherein control is taken from automated systems and given to the user. In the eventuality of a manual override, the system should take into account four different status a device can take: reversible override, irreversible, controlled by a timer or by external variables. For the first two status the solution we propose for the User Interface is a message stating that the device is being manually overridden and in case it is reversible, the UI should provide a reverse button to eliminate the manual override. If a timer is controlling the device it should indicated through tag with a time interval discriminated. Lastly, in case the device is being controlled by external variables, like sensors, there should be a tag indicating which variable is in control.

Sometimes a user might want to apply constraints to a machine or equipment. The solution we propose is a very simple one: icons. We need and unlocked and locked icon. The first one is when a machine is unlocked and does not have constraints while the second is in case a equipment cannot be used, hence the lock icon. In some cases the device can be partially locked, meaning, maximum and minimum values can be applied to prevent users from over using the machine. In that particular case, besides the lock icon, the values should also be indicated next to the device icon.

The last device behaviour to be analysed is related to energy consumption. Unlike the others types of behaviour, this one is only informative, i.e., the user cannot change it or act upon it. Regardless the information the system wants to present (device peak, average or lowest consumption) the solution we propose is a icon giving the idea that there is more information regarding the device is currently being used. In some cases, like light bulbs, we can show the current device consumption.

5.2 Interface Requirements Specification

Before building mockups , there was a need to specify requirements for each Interface. The goal of this work is to find solutions for feedback problems. However, we have decided to design a complete interface, simulating actions like a real Home Automation System and include features like most of them have.

For simulation purposes, we have assumed our system should have the following functional requirements: Distinguish Users inside the same house; House Consumption for each equipment; Illumination and brightness of all the lights inside the house; HVAC system; Scheduled Event that can go to a simple timer to turn off the TV to starting washing machines; Controlling the space around the user and not just one equipment; Gadgets and electronic devices like TV, sound system, washing machine or any other kind; and Multiple Triggers.

After specifying the functional requirements, we need to understand how many layouts we are going to design and get into the specifics on each of them. To make a believable Interface with features some of the existing HAS currently have, we have decided that it is necessary to develop six interfaces: welcome page (main panel), illumination page, HVAC, gadgets, scenarios and scheduled events.

The main panel will be the first interface the user will face every time he opens the application. This layout should let the user sign in (with username and password or just a unique number). It will have buttons, each of them representing functional requirements: Illumination button, HVAC, Scenarios, Gadgets and Scheduled Events. The layout will also have a settings button, which will be useful in case the user needs to adjust systems configuration, preferences, etc. It will also be in this page that the user will see the house consumption.

Illumination page, as the name suggests, will be

the interface where the user will be able to control the lights and brightness of the house. This page will also give information about all the lights that are turned on (the amount of watts it is spending, where it is and level of brightness).

Unlike many of the existing HAS interfaces, we have decided to develop a layout exclusively for electronic devices. Here, besides being able to turn devices on and off, the user will also be able to see which devices are working and set timers and schedule events with them. The user should also be able to turn on multiple devices.

Lastly, the schedule events layout. It is a page where the user can manage all the scheduled events. He should be able to see which ones are in progress, the ones that were concluded (successfully or not) and add more events. To create a schedule event the user needs to insert the date, time and the devices he wishes to turn on. Regardless the layout, the buttons referred in the main panel should be available at all times in all pages, as well as the settings button and the logo of the application.

5.3 User Interface Final Mockups

After defining the type of behaviour feedback, functional requirements and the flow of events the User Interface should take, mockups were designed so that we can have a visual idea of the solution that was proposed.

The version 2.0 was the last one developed by us. For our last version we have created 7 mockups that meet all the Device Behaviour Feedback Requirements. The design was completely changed, (regarding the previous versions) the colours were also changed and an Ipad-frame was designed to make it more realistic. The following descriptions will show how much thought we have put into this and how the requirements were met.

Figure 1 shows us the Gadgets mockup. Before getting into further details, at first we can see the following differences: Buttons are positioned vertically instead of horizontally. The icons representing each of the functionalities are different as well as the gadgets. On this mockup in particular we can see that the TV and Radio are turned on because both of them have the timeline all in green - representing the Device command confirmation feedback - and the ON is bold - representing the Device Hysteresis Feedback. On the right side of the TV icon I can see the channel I am currently watching, control the volume, switch channels or even set a timer. The same applies when it comes to controlling the radio but in this case instead of channels we are controlling the frequency.

Figure 1: Version 2 - Gadgets page updated version. From top left going vertically: five buttons representing each functionality like Illumination, Gadgets (has a different colour because it is selected), Scheduled events, HVAC system and Scenarios. Centre: TV and Radio are turned on and functioning properly. We can see that because of the 4 green lights below the icons and the controllers next to them. The radio image has a little next to it representing a schedule event. The fridge has a problem (does not mean is not turned on) hence the red dot bellow the icon. The last two gadgets (Microwave and Oven) are turned off (there are no green nor red dots).

On the other hand we have the microwave and oven turned off. On this case the controllers that allow you to handle those gadgets are not visible and the OFF is bold. The timeline is white, meaning there was not even an attempt to turn them on.

The fridge is a case where we can see the requirements of Device Error/Warning conditions an Device network status. was we can see the timeline has a red dot. That means somewhere along the request something went wrong. The issue here is that regardless of the problem the request might have suffered, we might still be able to control the fridge. So the details are shown but also a warning sign is. Notice that the cross next to the OFF sign is bold.

The Illumination mockup presented in Figure 2 has a simple design but important details that help users better understand what we can or cannot do.

Buttons are on the left side of the screen and organized vertically. The novelty in this case is on top - all the house compartments are discriminated. In this case the Living room is highlighted, meaning it is the one where the lights on the main screen are presented. Here we can see that light1 and light2 are turned on with an intensity of 80 watts and 30 watts, respectively. Light3 had a problem, hence the warning icon.

In order to make it simpler to dim the lights, there is a bar below each light bulb to regulate brightness.

Figure 2: Version 2 - Illumination page updated version. Starting from the top left of the image there are 5 buttons representing the features available in the mockup: Illumination (with a different colour because it is selected), Gadgets, Scheduled Events, HVAC and Scenarios. Right next to the Illumination button the mockup show all the house divisions where lights can be controlled. There is also a add button where we can add house divisions and set the number of lights to be controlled in that area. The living room is underlined because the lights of that division is shown below. the 3 light bulbs are available to be controlled. The first two are turned on while the third had a problem, hence the red dot and the warning sign right next to the ON/OFF signal.

Types of Behaviour	UI Mockups version 2.0				
	Illumination	Gadgets	S. Events	HVAC	Scenarios
Dev. Status Feedback	x	x	-	x	-
Dev. Status Provenance	-	-	x	-	x
Dev. Command Confirmation Feedback	x	x	-	-	-
Dev. Hysteresis Feedback	x	-	-	-	-
Dev. Network Status	-	-	-	-	-
Dev. Error and Warning Conditions	-	x	-	-	-
Dev. Override Settings	x	x	-	x	x
Dev. Constraint Feedback	x	-	-	-	-
Dev. Energy Consumption Feedback	-	-	-	x	-

Table 3: Crossover between types of behaviour and version two of the UI mockups. Table representing types of behaviour and the mockup pages o version two that contain the solution proposals for each of them. All pages marked with an 'x' mean they contain an type of solution to that type of behaviour, while all pages marked with an '-' mean they do not take that type of behaviour into account.

Another important detail is the light bulb icon itself. Light2 is only consuming 30 watts, so the icon is only half full, while the first one is completely full. This details allow the user to understand the level of consumption.

When the user wishes to turn on the lights we can do it by pressing the On button or by clicking in the light bulb.

Looking at Table 3 we can see that this final version of the mockup covers almost all types of behaviour and that the Illumination and Gadgets page are the ones that contain more solution proposals. Since we have developed images of possible User Interfaces and not a real system, that would explain why we were not capable of implementing the solutions proposed for the Device Network Status.

6 VALIDATION AND RESULTS

In order to validate the device behaviour feedback requirements and the interface requirements specifications we have planned a evaluation that will allow us to take more specific conclusions about our work done so far and it will give the users an opportunity to interact with the mockups and give their feedback. After performing the evaluation we will collect the results and analyse in order to take conclusions regarding all the work done by us.

6.1 Evaluation

The evaluation we have defined consists in six steps, all of them with different goals. The steps are:

1. Developing User Interface Mockups. For Validation purposes, we decided to design three different mockup versions, zero, one and two. The version 0 will not withhold any proposed solution from the Device Behaviour Feedback defined by us. Version 1 will contain some of the requirements and the last version will have most of them. The goal is to have three distinct versions with different requirements and interfaces in order for the user to differentiate from them and evaluate the one they like the most.

2. Preparing the Oz Paradigm evaluation (Bernsen et al., 1994). With the developed mockups, we are going to create HTML pages for each image and, through Javascript/JQuery, associate events for each "button" inside the mockups. These events will simply call another HTML page with the respective mockup. This step is the assemblage of the Oz paradigm. However instead of having a Wizard behind the mockups, we have decided to use web programming languages to better simulate a system.

   Another important aspect we have done in this step is the counter. For each button with an event associated, ie, for each html page, a counter will be associated to it in order to determine which pages the testers use the most. For each time a user goes to a determined page, the counter associated to that mockup will be incremented. This

way we can have an idea of which pages the user has been through and the one he used the most;

3. Setting up the questionnaires. Questionnaires are a great way to collect results to have a better idea the impact our work had to the users and testers of our program (Perlman, 1998). The questionnaires we used were some of the ones referred in the Concepts section. Although we did not change any of the questions or the rating of each of statement on the questionnaire, we needed to configure it in order to better collect the results. Using a customizable Web-based perl CGI script developed by Gary Perlman we were able to set, for each questionnaire, every time the user presses the send button, it will automatically forward the results to the inbox of the email we set. This way we can "grab" the results and analyse them on the spot. Besides the email, the script allow us to set the systems name and an auxiliary description to help users better understand the goal of each test;

4. Users evaluation - part one. Using the HTML pages developed on step one, users will be able to test each version of the mockups and give their feedback. Regardless of the mockup, the starting point will always be the welcome.html page. From there, users can freely tests everything and anything they want from the "system". The main goal of this test is for users to focus not only in usability but also the ease in understanding the messages, understand the meaning of the icons and judge the aesthetics of each mockup.

   After testing each version, users, through the respective links provided by us, will have to answer to questionnaires, the Questionnaire for User Interaction Satisfaction (Chin et al., 1988) and the Nielsen's Usability Heuristics (Nielsen, 1994a). As mentioned in step two, the results will be sent to the email account we have configured. With this step we intend to determine which version of the mockup users like the most, which one they thought it was aesthetically more appealing and, through the usability heuristics defined by Jakob Nielsen, determine if they respect all of them or not. Additionally we will also study the pages they have visited the most through the counter we have associated to each event in step one. In the end, the best version of the mockups will be the chosen one for further testing;

5. Analyse results so far. So far we have set up the Oz paradigm, we have chosen the questionnaires for users to answer and have collected the results of the first part of users evaluation. Now we have to collect all the results and analyse them. Taking into account our goal for this first evaluation we are now able to determine which mockup is the favourite and which pages were most visited by the testers. Another positive aspect of the questionnaires is that, besides the questions there was also room for the users to freely describe three positive and negative aspects found during the time they tested the "system". With those comments, we are also able to improve some of the mockups before going to the next step.

   With these improvements we are going to update the HTML pages and now we have all set for the next step.

6. Users Evaluation - part two. At this stage we already now which mockup version was the favourite one, with better results regarding the two questionnaires answered and analysed in previous steps. Now we have the chance to test deeply the changes we have made according to users evaluation and the capacity of our mockups guiding the users through some challenges. Given a scenario where the testers should execute or perform an action, we want to evaluate the easiness and the celerity to complete the task they have in hands. We have defined two different scenarios: in the first one the user will have to perform a simple task like scheduling an event in a certain day at a certain hour, while the second scenario will be related to editing and controlling illumination and brightness. It is worth mentioning that the improvements and the mockups developed so far were designed thinking about those two scenarios, taking into account that they do not represent a real system but a set of images capable of simulating one (thanks to JQuery library).

   Just like we did in step three, the user will start the task in the welcome.html page and, from there, they should be able to complete the tasks we have set for them. After each challenge, the users will have to answer a questionnaire containing only three questions and, like previous questionnaires, a chance to point out three positive and negative aspects of their experience. Through the After Scenario Questionnaire (Lewis, 1995) we are going to be able to determine whether the user thought it took to long to complete the tasks and the mains obstacles they had to overcome;

7. Collecting final results. With the final tests concluded and the evaluation from the users finished, we now get to the second phase of collecting and analysing the final results. At this point we have enough data to determine whether the chosen mockups are good when it comes to performing tasks and how well the users reacted to the

challenges they had in hands. With all the questionnaires we will also be able to build graphics that will help us better understand the results and take more conclusions regarding the effectiveness of our tests.

With these six steps we believe it will be enough to take conclusions regarding all the developed mockups and will help us validate everything we have worked so far. It will also be useful to know to which extent the mockups would be accepted in the market and the benefits of implementing them in any HAS.

6.2 Collecting Data and Analysing Results

After collecting all the data from the questionnaires, we have analysed all the relevant information and got to the following conclusions:

Through the QUIS we were able to conclude that version 2.0 was the one with the best results and feedback from the users. Version 1.0 was the second most favourite while the version 0 was clearly not satisfactory and got the lowest scores. It is important to understand that the values presented here correspond to an average of the total number of evaluations made. For example, in case of version 2, regarding the statement ”Reading characters on the screen”, as we can see, it is rated as 7. This does not mean the total of ratings was 7 but the average from all users evaluations gave that result.

Regarding the Usability Heuristics defined by Jakob Nielsen, none of the version got the perfect score but taking into account that the tests were made with mockups and not a real system we can consider them very good. Once again the version 2.0 was the one with the highest score. Version 1.0 had a very good feedback from users usability wise and version 0 was once again the least preferred for the users.

For each statement the user could evaluate it from 0 to 7, so the maximum scored allowed by this Questionnaire was 70 points. Once again it is important to remember that this graph shows the summations of the average calculated for each statement. Version 2.0 was the best one with a score of 43,5 points while the other two versions (0 and 1) got 23,1 and 38,4, respectively.

After collecting the data from the counters that were associated to each HTML page, we have concluded that regardless of the version of the mockup, people were more compelled to use the Illumination page, followed by the Gadgets page. It is important to mention that this result can be tricky, seeing as we just evaluated the number of times users go to each page and not the amount of time the spend on it. So if a user presses a button by mistake and goes back to the previous page, both counters will be incremented. Nevertheless, we believe that one of the reasons that people used these two pages the most it is because they are the ones with more interaction.

According to the analysis made we were able to see that version 1, unlike the other two, was the only one where the gadgets page was more visited. By a difference of almost 10 percent to the second most visited page, Illumination. As for version 0, the Scenarios page is clearly the one testers were not convinced by its design or features, since it was the one with the least percentage of visits. Illumination was by far the one users were more keen on using.

Analysing users feedback regarding positive and negative aspects, some of them pointed out for version 2: the Scenarios, HVAC and Gadgets page needed some improvements. The first one was said to be a bit confusing when editing a specific scenario, while the HVAC page was too simple with little information and users felt the page lacked ”attractiveness”. The Gadgets page was also too confusing, specially because machines that were not turned on, also had their commands visible, making the user puzzled with too much information that we did not need. All of these pages were improved and updated for the last phase of testing.

Analysing the data from the Scenarios test, we got to the conclusion that users, overall, had a little trouble completing the task in the first scenario. Testers had more difficulty to conclude the first scenario rather than the second and, consequently, were not very satisfied with the time they spent to complete the task. The column related to support information has to do with the informations users could find on the internet to better understand the scenarios.

Regarding specifically scenario one, most of them thought the time that took them to complete the task was not very satisfying but according to the feedback pointed out on the negative aspects, they thought there should be some redundancy when scheduling events. This leads us to conclude that we should also enable users to schedule events by choosing the gadgets or equipment they want to schedule. So instead of just scheduling events from that specific page, through the calendar, the

users should be able to do it by picking the equipment first and the date and time later;

Regarding Scenario 2, in general, users where satisfied with the challenge itself and how they managed to solve it. The time they spent to complete the task was good and the feedback both positive and negative was almost non-existent;

Overall , after all the tests and feedback from the users and after taking all conclusions pointed out before, we can clearly state the version 2 came to prove that the requirements we have presented in the Solution section, regarding functionalities and device behaviour feedback, plus the definition of different types of request, all this respecting the flowchart of events also defined in that chapter were correct and if we develop a system that is able to meet all of these requirements the acceptance of the users and in the market will be good.

7 CONCLUSIONS

According to literature and analysis made to User Interfaces of existing HAS it can be concluded that most of them do not take into account feedback issues. Most systems that are currently in the market are more concerned with gathering functional requirements rather than improving their interfaces, exposing these systems to lack of usability and ease of use. the net result is that feedback issues are getting bigger and making users think twice before installing an application for smart homes. Commonly, solutions are not the most adequate and tend to confuse users mostly because even when the Interface is able to give feedback, it does not fairly represent the problem the user have in hands. Another issue is lack of standards or User Interface patterns that could help solve a lot of problems regarding usability.

The followed approach is composed by three different steps. The first step consists in defining a set of requirements regarding feedback behaviour from devices connected to a HAS. These requirements take into account the type of behaviour devices can take, consider all the status they can have and present a interface level solution in order to accurately portrait the status the user is facing. Step two regards functional requirements for a typical HAS. Additionally it was also necessary to define a set of layout specifications. Lastly, we have created User Interface mockups containing our solution proposal so we could prove the effectiveness of our work.

The User Interface mockups were validated using a set of approaches previously studied. Before that, three other versions of UI were designed in order for the user to have a choice. The difference between them relies on the amount requirements they respect regarding the solution we propose. With the Oz Paradigm we were able to let the user interact with the designed mockups, making him believe it was in a fact a real system. Along with the experience, users were asked to answer a set of questionnaires related to usability, ease of use and after scenario feedback.

The evaluation results demonstrate that the User Interface that respected more layout specifications, functional and feedback requirements was the one users were more satisfied with.

This approach along with the solution we have defined, are of great importance and is a good starting point for systems aimed for home automation to have a standardized solution for feedback issues.

Acknowledgement

We would like to thank to my supervisor Paulo Carreira for the valuable guidance and advice given throughout this work and the designer Sara Nunes for her support regarding the validation. We would also like to thank the FCT - Fundação para a Ciência e a Tecnologia - for giving us the funds under the project PEst-OE/EEI/LA0021/2013.

Furthermore I would like to thank my colleagues and all the dissertation readers for giving feedback when needed.

Lastly I want to thank my family for all the love and support they gave me throughout this work.

REFERENCES

Alexander, C., Ishikawa, S., and Silverstein, M. (1978). A pattern language: Towns, buildings, construction (center for environmental structure series).

Barfield, L., van Burgsteden, W., Lanfermeijer, R., Mulder, B., Ossewold, J., Rijken, D., and Wegner, P. (1994). Interaction design at the utrecht school of the arts. *ACM SIGCHI Bulletin*, 26(3):49–86.

Bernsen, N. O., Dybkjær, H., and Dybkjær, L. (1994). Wizard of oz prototyping: How and when. *Proc. CCI Working Papers Cognit. Sci./HCI, Roskilde, Denmark*.

Bevana, N., Kirakowskib, J., and Maissela, J. (1991). What is usability? In *Proceedings of the 4th International Conference on HCI*.

Borchers, J. O. (2001). A pattern approach to interaction design. *AI & SOCIETY*, 15(4):359–376.

Bowden, S. and Offer, A. (1994). Household appliances and the use of time: the united states and britain since the 1920s1. *The Economic History Review*, 47(4):725–748.

Chin, J. P., Diehl, V. A., and Norman, K. L. (1988). Development of an instrument measuring user satisfaction of the human-computer interface. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 213–218. ACM.

Davidoff, S., Lee, M. K., Yiu, C., Zimmerman, J., and Dey, A. K. (2006). Principles of smart home control. In *UbiComp 2006: Ubiquitous Computing*, pages 19–34. Springer.

Davis, F. D. (1989). Perceived usefulness, perceived ease of use, and user acceptance of information technology. *MIS quarterly*, pages 319–340.

Fujita, Y. and Lam, S. P. (1996). Distribution system and method for menu-driven user interface. US Patent 5,500,794.

Ghanam, Y., Shouman, M., Greenberg, S., and Maurer, F. (2009). Object-specific interfaces in smart homes. Technical report, Citeseer.

Granlund, A., Lafrenière, D., and Carr, D. A. (2001). A pattern-supported approach to the user interface design process. In *Proceedings of HCI International*, volume 1.

Griffiths, R., Pemberton, L., Borchers, J., and Stork, A. (2000). Pattern languages for interaction design: Building momentum. In *CHI'00 extended abstracts on Human factors in computing systems*, pages 363–363. ACM.

Han, J., Yun, J., Jang, J., and Park, K.-R. (2010). User-friendly home automation based on 3d virtual world. *Consumer Electronics, IEEE Transactions on*, 56(3):1843–1847.

Harper, R. (2003). *Inside the smart home*. Springer.

Humphries, L. S., Rasmussen, G., Voita, D. L., and Pritchett, J. D. (1997). Home automation system. US Patent 5,621,662.

Lewis, J. R. (1995). Ibm computer usability satisfaction questionnaires: psychometric evaluation and instructions for use. *International Journal of Human-Computer Interaction*, 7(1):57–78.

Liu, D. and Xian, D. (2007). Home environmental control system for the disabled. In *Proceedings of the 1st international convention on Rehabilitation engineering & assistive technology: in conjunction with 1st Tan Tock Seng Hospital Neurorehabilitation Meeting*, pages 164–168. ACM.

Mayhew, D. J. (1999). The usability engineering lifecycle. In *CHI '99 Extended Abstracts on Human Factors in Computing Systems*, CHI EA '99, pages 147–148, New York, NY, USA. ACM.

Mills, D., Bonn, E. A., and San Juan, S. S. T. M. (1992). Macintosh human interface guidelines.

Molina, P., Meliá, S., and Pastor, O. (2002). User interface conceptual patterns. In Forbrig, P., Limbourg, Q., Vanderdonckt, J., and Urban, B., editors, *Interactive Systems:Design, Specification, and Verification*, volume 2545 of *Lecture Notes in Computer Science*, pages 159–172. Springer Berlin Heidelberg.

Nielsen, J. (1994a). Heuristic evaluation. *Usability inspection methods*, 17:25–62.

Nielsen, J. (1994b). *Usability engineering*. Access Online via Elsevier.

Nielsen, J. and Molich, R. (1990). Heuristic evaluation of user interfaces. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 249–256. ACM.

Nilsson, E. G. (2009). Design patterns for user interface for mobile applications. *Advances in Engineering Software*, 40(12):1318–1328.

Norman, D. A. (2002). *The design of everyday things*. Basic books.

Perlman, G. (1998). Web-based user interface evaluation with questionnaires. *Retrieved on*, 3(22):06.

Spencer, D. (2004). What is usability? *University of Melbourne, Melbourne*, pages 108–115.

Van Der Werff, M., Gui, X., and Xu, W. (2005). A mobile-based home automation system. In *Mobile Technology, Applications and Systems, 2005 2nd International Conference on*, pages 5–pp. IEEE.

Van Welie, M. and Trætteberg, H. (2000). Interaction patterns in user interfaces. In *7th Pattern Languages of Programs Conference*, volume 13, page 16.

Visintin, A. (1994). *Differential models of hysteresis*. Springer Berlin.

Wimsatt, W. et al. (2006). Home automation contextual user interface. US Patent 7,047,092.