EAI Endorsed Transactions
on Context-aware Systems and Applications

Research Article EAI.EU

User Control Support in Smart Homes

M. Leghari[1,*], L.D. Dhomeja[1] and S.A. Memon[1]

[1]Institute of Information & Communication Technology, University of Sindh Jamshoro, Pakistan

Abstract

A smart home is a context-aware system that adapts itself autonomously in response to context to satisfy user needs and to improve safety, security, resource use, etc. On the one hand, software autonomy serves the basic purpose of pervasive computing by reducing interaction with the users, easing the use of the system, and reducing the user distraction. On the other hand, it takes control away from the users of the applications, making users feel loss of control over their context-aware applications. The situations including applications may not behave as expected, user preferences may change over time or users may want to add new behaviors, etc, may arise and require smart home users to interact with the applications to control their behavior. This research addresses this issue and proposes an approach, which would provide a wider support of user control by exposing and manipulating (1) application parameters, (2) adaptation logic(s) thus allowing users to add new behaviors. Using this approach a complete system is developed in order to see its effectiveness; furthermore the system is tested on three different context aware applications and a preliminary usability study is done to evaluate the system effectiveness.

Keywords: User control, Context awareness, Smart homes, Adaptation logic modifications, Jess, User Control Manager.

Received on 21 January 2020, accepted on 15 May 2020 , published on 22 May 2020.

doi: 10.4108/eai.13-7-2018.164666

*Corresponding author. Email: mehjabeen.leghari@usindh.edu.pk

1. Introduction

A smart home is a context-aware system that adapts itself autonomously in response to context (location, activity, user presence, temperature, etc) to satisfy user needs and to improve security, resource use, etc.

Dey and Abowd [1] describe context as "Any information that can be used to characterize the situation of an entity. An entity is a person, place, or object that is considered relevant to the interaction between a user and an application, including the user and the applications themselves". In the case of a smart home the application may blend into the background, as Weiser envisaged [27], but the definition remains valid.

Context-awareness can manifest itself in various ways [5], but in a smart home it will tend to result in a greater degree of autonomy in the control systems adaptation of the home's behavior. The application's autonomy means that there is no interaction involved between a user and the application when it is being executed. This software autonomy takes control away from the users of the applications. While software autonomy serves the basic purpose of pervasive computing by performing users' tasks without requiring their attention, the users may develop feelings of loss of control over their applications. There might be situations when the users of the home may need to interact with the applications. For example, the home applications may not behave as expected and hence require interacting with the system to correct their behavior.

Beyond day-to-day use, there will be times when a user may want to reprogram or edit the behavior of the home. Preference may vary: either in tuning the definition of the context triggering them, or in the resulting behavior. New applications or behaviors may be identified and programmed in. New appliances may be installed and require integrating into the control. New aspects of context to be sensed may be installed and suitable responses woven into the existing behaviors. In each case there may be some change to the encoding of preferences or logic. Above mentioned situations demand the need for providing the users of home with suitable mechanisms to control the behavior of the applications. However, it is important that context-aware applications must provide mechanisms to provide a suitable balance between software autonomy and user control as discussed in [17]. In [4] a survey is presented to determine whether users prefer autonomous context aware applications or preference based context aware applications. In the light of survey results, giving user control in a context aware application is appreciated by most of the users. [26] suggested that giving 100 % self adaptivity to context aware application may make its users anxious.

While there has been intensive research on context-awareness and a summary of contributions made in this field can be found at [6], [1], the research on providing user control in context-aware application has just started getting attention from research community. Our literature survey (discussed in Section 2) indicates that some approaches to user control in context-aware applications have been studied, which include End User Programming [7], [9], [10], Preference Based User Control [13], [17], [18], [19], [16] and Enactor Model [11].

Unlike traditional programming approaches where application developers define or develop applications, end user programming enables end users to define the behaviour of context-aware applications. The approaches based on end user programming normally provides better user control than traditional software engineering approaches but they are mostly used when the application behaviour is very simple like controlling lights or bells, etc [17]. In preference-based approaches the user control is achieved by configuring user preferences, which are provided during design time only, restricting the user control to a limited number of user preferences. In an enactor model [24], user control is provided by exposing and manipulating application parameters. This approach provides a better support of user control by exposing and manipulating all application parameters than providing a few personalization options as in preference based systems. We argue that there may come times when control of application behaviour may involve more than just setting the parameters, i.e., modifying adaptation logic or contextual information of the application or user may want to add new context-aware behaviours that were not identified at the design time. This kind of user control is not possible with enactor model. Although in [11] and [24] using preference model the authors have proposed an approach that will expose application information directly to the users for controlling application behaviour, but in their proposed model it is too difficult to understand the preferences by the end users specially novice users as these are expressed as high level If-Then-Else clauses. Our research addresses all these issues and proposes an approach to supporting user control which exposes and manipulates not only application parameter but also contextual information and adaptation logic of the application. We tested and evaluated our approach by implementing on smart home hypothetical example scenarios.

2. Literature Review of User Control Approaches

A number of approaches to providing user control in context aware applications have been proposed in the literature which can be categorized as:

[1] End user programming

[2] Preference based user control

[3] Extension to context toolkit

2.1. End User Programming

End user programming [9] allows end users to define the behavior of context-aware applications and add behaviors that were not foreseen by system developers. A number of systems have been developed, using end user programming approach. For example, CAPpella [10] and Topiary [23] are end user programming systems, which are based on a paradigm called Programming by Demonstration [7]. These systems require end users to define system behaviors by performing the desired operations manually. For example in CAPpella a medicine taking scenario is discussed in which end users demonstrate the system different medicine taking activities. The system uses machine learning techniques to recognize those activities. Later if user takes the medicine on time its record is logged on to medical journal, else user is reminded using alarm after certain time to take the medicine. Unlike end user programming with programming by demonstration, Jigsaw editor [20] enables end users to develop context-aware applications by assembling components together (i.e. arranging components such as sensors, applications and devices). Using this editor, users can arrange the components accordingly to achieve their desired tasks in the real world.

Although end user programming approach is very powerful technique since end users can understand, define and demonstrate their behaviors/requirements better than developers and designers as discussed in [18], these approaches have been used for initial training of the systems by end users. These are not preferred for the situations where users' preferences change over time.

2.2 Preference Based User Control

Preference-based decision support is a well-known approach to supporting user control in traditional computer applications that involves incorporating preferences and personalization mechanisms. Cell phone is a very popular example of that in which users control their devices by setting their required preferences e.g. date, time, calendar, themes and sound etc. This approach has also been explored in context aware applications. Various preference based approaches have been developed by researchers for providing personalized user control in context aware applications. The user control approach in [19] allows exposing preference information to the users for setting their preferences of context aware applications. The persona system [22] receives information from the environment and presents it to the application using preference information of particular user. For example aware mirror application uses artifacts including sensor based tooth brush to recognize users and a mirror to show the user relevant information of weather, news etc. if a user don't want to see any of such information then she can use GUI or voice based commands to personalize these applications. Although persona approach is so much robust in that it provides real end-user interaction of the system, it is limited in that only a set of preference attributes can be modified by the users no logic internals are exposed to users. In [15], [12] authors have proposed a preference modeling approach for supporting intelligibility and control in context aware applications. Using preference model they have exposed applications behaviors directly to end-users for controlling any unwanted behavior of application. Although their approach is better in providing user control support than previously discussed approaches as end users can directly manipulate context aware application's behavior, their chosen preference model is much complicated and difficult to be understood by novice end users. Their approach enables end users to control the context aware applications by changing their preference information. This is achieved by developing appropriate feedback forms by the developers. The changes/modifications performed by users are mapped back to the context aware rules by using different logics/algorithms (Predicate Logic and Defeasible Logic) [14]. Although their approach is promising providing user control and intelligibility, the feedback forms developed are not much flexible and user friendly and require too much low level options to be performed by end users. Also the feedback form is used just to change the preference information of a user; there is no mechanism to provide adaptation logic or rule level manipulation access to end users. Our approach on other hand differs from their preference model in that it allows exposition and manipulation of various parts of context aware application, through which users could be able to change the actual application logic or contextual information dynamically. In [16] users can create semantic web rules to make ontological models and context aware applications according to their preferences.

2.3 Extension to Context Toolkit

Another very notable approach to supporting user control of context-aware application has been proposed and implemented by [24]. In their approach, Context Toolkit [8] has been extended by introducing an enactor model. Developing context-aware applications using their approach requires coding enactor component, which encapsulates application logic that includes adaptation logic and parameters. The user control support is provided by exposing parameters of the context-aware applications and allowing the user to manipulate these parameters to control the application behavior. The important point to note about their approach [11] is that the user control is provided by exposing parameters only and manipulating them. However, we argue that there may come times when control of application behavior may involve more than just setting the parameters, i.e., modifying adaptation logic or contextual information of the application. Also, their proposed approach is inefficient for providing user control in context aware applications, when users want to add any new functionality into the system that is determined later by users.

Our literature survey provided above suggests that no existing approaches provide users control in context-aware applications in such a manner that all parts of the application could be exposed and manipulated to support a wider user control with an easy to use technique. To address this issue we have proposed an approach that will provide a wider user control support by allowing exposition and manipulation of application parameters, adaptation logic and contextual information. Additionally, end users will be able to add context-aware behaviours identified later. All this would be performed by using a user friendly GUI of the application.

3. Proposed Approach

We present a novel approach to supporting user control in smart home environments through which users can themselves control the application behavior directly without being too much technical. Proposed approach allows users to control context-aware home applications by:
exposing and manipulating application parameters
exposing and manipulating application adaptation logic
The usefulness of the proposed approach could be better understood in the light of following hypothetical smart home light example scenario that has been modified from [12]. The scenario is that:
"When a user enters a room (e.g. room1), the light of the room turned ON and adjusted to last used light intensity value (e.g.75% light intensity), the dark adjoining rooms' light is raised to 10% light intensity value. The system senses the light level of room1 and if it is 10% below or above the user's preferred light value (e.g. 65% light intensity) for room1 then it is adjusted to the user's preferred light value. If user stays in the room for a certain time period, (e.g. for two minutes) adjoining rooms' lights turns OFF.
Through this example we demonstrate how our proposed approach supports user control at both levels: (1) application parameters and (2) application adaptation logic.

3.1 Parameter Modification

A context aware application may contain a number of parameters or preference information, on which its behavior can be dependent. For example in the above scenario we can see a number of parameters involved in the overall application's functionality as listed below:
User's preferred light intensity can be stored as an integer variable and set to a value e.g. 75% light intensity.
Adjoining rooms' light value may be another integer parameter set to 10% light intensity value.
Adjoining rooms' light status, whether or not to turn them ON could be stored as a Boolean variable set to e.g. "True".
Similarly the time information after which adjoining rooms must be turned off could be stored as an integer variable and set to e.g. "2 minutes".
User control could then be achieved by fine tuning any of these parameters. Let say initially a user's preferred light intensity value for room1 was 65%. Afterwards she feels that she needs to use more light intensity e.g. 85% for certain activities in room1. She can achieve such change just by changing her preferred light value variable for room1 e.g. "user1room1intensity=85".
Next we can see initial or default value for adjoining rooms' light intensity is 10%, if an elder user's preferred light intensity is 90% and she wants to use 20% adjoining rooms light then this too could be achieved by changing 2nd variable's value to 20 i.e. "adjoining_room_lights_user1=20%".
Further if a user wants to control behavior of application such that lights of the adjoining rooms should remain off instead of their light values being adjusted to 10% intensity value. This can be achieved by setting the Boolean variable value to OFF.
Similarly, if user wants to change the behavior of the application such that if the user stays in the room2 for a minute instead of 2 minutes, adjoining rooms light remained turn OFF. This can be done by modifying time variable's value to "1 minute". This enables smart home users to control application behavior at any time.

3.2 Adaptation Logic Modification

In addition to application parameters, a context aware application may contain a number of rules in the form of Event Condition Action (ECA) rule[4]. As name implies an ECA rule is composed of 3 parts i.e. Event, Condition and Action. When an event triggers in a context aware application and certain conditions are satisfied then actions are to be taken for the user. Through the same example scenario discussed in section 3 we demonstrate how user can control application behavior by modifying application's adaptation logic/ ECA rules using our approach.
In the above scenario there are two rules involved as shown here:
Rule1:
Event(s)/Context(s): User Presence user_id= "User1", location= "Room1"
Action(s): set the light intensity to last used value e.g. (75% light intensity value).
Turn ON adjoining rooms light and set their light values.
Rule2:
Event(s)/Context(s): Room1 light intensity value
Condition: If ((user's preferred light value + 10) > (Room1 light) OR (user's preferred light value – 10) < (Room1 light))
Action: set the light intensity of Room1 to user's preferred light intensity value.
Rule3:
Event(s)/Context(s): duration= "02 minutes"
Condition: If (User still in Room)
Action: Turn OFF adjoining rooms' light
Now if user wants modifications in application behavior such that when user enters a room, initially light of the room must be set to the user's preferred light intensity value instead of last used light intensity, this can never be achieved by changing application's parameters, it requires that now every time for a particular user's entry in a room his preferred light intensity must be adjusted instead of last used light intensity value, which is a modification in the action part of context aware rule i.e. Rule1. Using our approach user can perform such modification by changing

action part of the Rule1. After such modifications Rule1 can be re-written as:
Rule1:
Event(s)/Context(s): User Presence user_id= "User1", location= "Room1"
Action(s): set the light intensity of Room1 to user's preferred light intensity value.
Turn ON adjoining rooms light and set their light values.

4. System Architecture and Implementation

In order to provide a wider user control support, context aware applications must be designed using decoupled architecture. Our proposed system provides a wider user control support and its high level architecture along with context aware application is shown in Figure 1. It has a main component: **User Control Manager (UCM).** UCM has two sub components i.e. Parameter Manipulator and Adaptation Logic Manipulator.

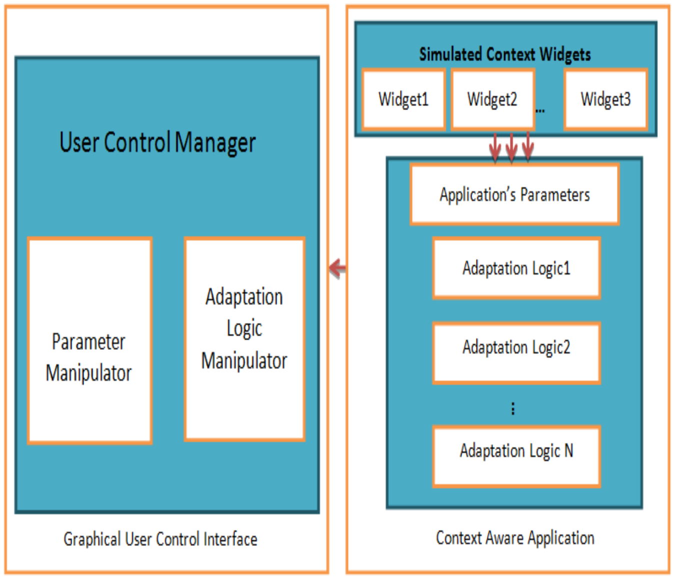


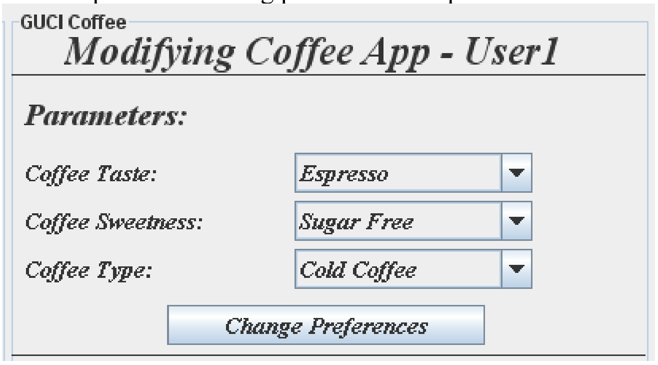
Figure 1. High Level System Architecture

4.1 User Control Manger (Ucm)

UCM is a Java's Swing based main component of the system which works as bridge between user and the context aware application. It gets all the preference and adaptation logic information of a context aware application dynamically and presents that to user using a graphical user interface. Later if the user has performed any changes to their preferences and/or logic internals of application, UCM is responsible to map those changes back to context aware application's rule base and profile parts to change application's behavior for the users.
There are two subcomponents of the User Control Manager:
Parameter Manipulator
Adaptation Logic Manipulator

Parameter Manipulator

The first component called parameter manipulator is responsible for fetching parameters of a context aware application. End users can manipulate the parameters/preferences of a context aware application using parameter manipulator, according to their desires to fine tune application's behavior.
For example in a smart home light application there could be a number of preferences of a user like:
Light intensity of a particular room when a user enters into it.
Adjoining rooms' light status upon user's entry into room.
The intensity at which adjoining rooms turned ON.
Or may be about after how much time adjoining rooms fade back OFF?
In a context aware application this preferences information is stored as application parameters. Every time a user is detected by the application it adapts its behavior according to the parameters stored in the application. If user performs any modification in the preference information then these changes are performed dynamically by parameter manipulator which enables end users to fine tune application behavior. Figure 2 shows parameter manipulator for smart coffee application. It depicts GUI implementation for coffee application. As we can see parameters for a coffee app could be: coffee type, whether hot or cold and suger free or sweet. Users can set parameters for such an app by changing values of these parameters using parameter manipulator.


GUCI Coffee
Modifying Coffee App - User1
Parameters:

Coffee Taste:	Espresso
Coffee Sweetness:	Sugar Free
Coffee Type:	Cold Coffee

Figure 2. Parameter manipulator for coffee application

Adaptation Logic Manipulator

Users may want to further control the behavior of a context aware application then just setting application's parameters. Second component of UCM called Adaptation Logic Manipulator is responsible for providing such type of user control in context aware applications. Adaptation logic manipulator is responsible to fetch adaptation logic internals to users for modification, and to map back all the changes to context aware applications.
There are 3 subparts of adaptation logic manipulator and those are designed according to the different parts of a rule in a context aware application. Context aware applications' rules are composed of 3 parts called Event, Condition and Action. So using Adaptation Logic Manipulator all three subparts of a rule could be modified by end-users. In this way it enables end users to change behavior of context

aware applications at a wider level. Figure 3 depicts the Adaptation Logic Manipulator for smart AC application using which users can change application logic as well. Like when to turn ON AC and at which temperature etc.

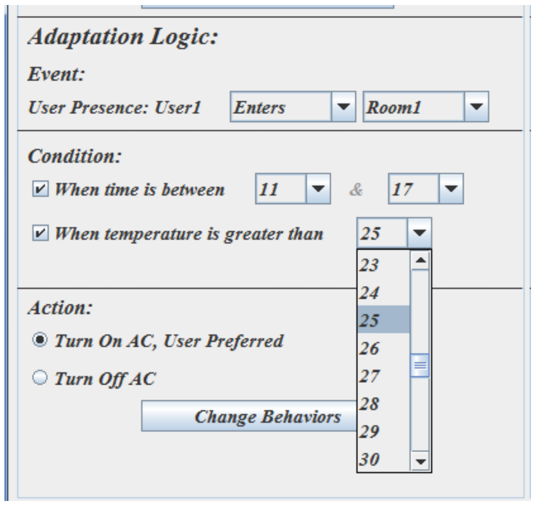

Adaptation Logic:
Event:
User Presence: User1 | *Enters* | *Room1*
Condition:
☑ *When time is between* *11* & *17*
☑ *When temperature is greater than* *25*
23
24
25
26
27
28
29
30
Action:
● *Turn On AC, User Preferred*
○ *Turn Off AC*

Figure 3. Adaptation Logic Modification in Smart AC Application

4.2 Context Aware Application

We have developed three different context aware application's prototypes (i.e. Smart home light application, smart home AC control application and smart coffee application) to be used with the proposed system architecture.
Simulated context widgets are used to present context inputs to various applications. Context aware applications may consist of a number of application's parameters and adaptation's logic(s). UCM works in conjunction with context aware application to provide user control to end users. We have used Jess [21] a rule based system to create context aware applications. As discussed in section 4 providing wider support of user control require decoupled architecture of context aware applications. It is easy to use Jess for such type of decoupled context aware applications, because Jess rules could be modified dynamically.

4.3 Sequence Diagrams

Figure 4 depicts the message sequence diagram of the parameters' manipulation in the proposed system. Here initially context widgets are responsible for providing context to the rule engine, which in turn triggers actions in response to that context using application's actuators. It is the responsibility of parameter manipulator to fetch user related parameters from the rule engine and presents them to users to let they fine tune application behavior. Whatever changes users made to the parameters using interface are mapped back to rule engine with the help of parameter manipulator, hence it works as the bridge between application and user.

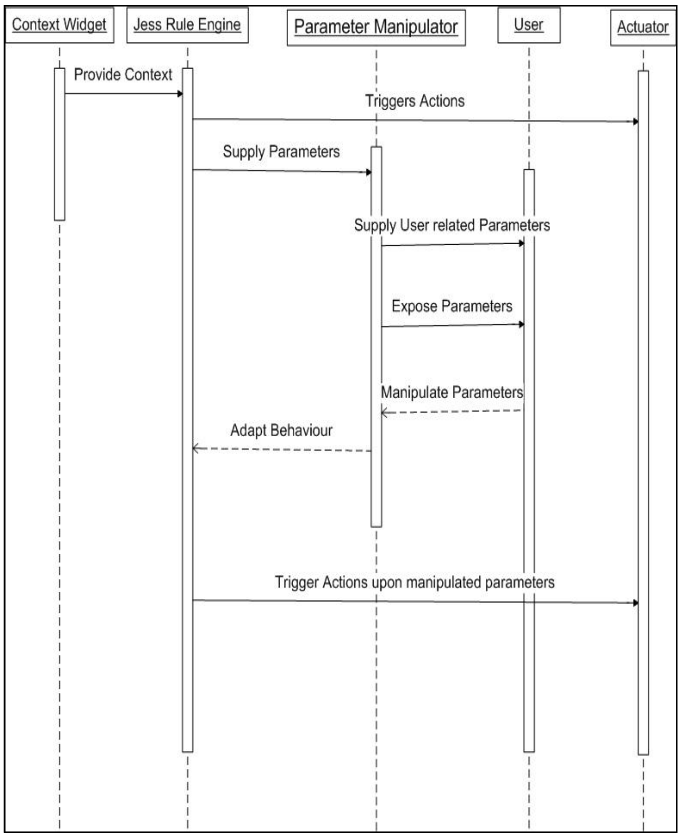


Figure 4. Sequence Diagram for the Parameters Manipulation

Similarly figure 5 illustrates the sequence of messages in overall adaptation logic manipulation using our system. Here the overall working is same as of parameters manipulation; the difference here is that adaptation logic manipulator is responsible for fetching context aware rule and to present its different parts i.e. events, conditions, and actions to the users using GUI to let they control the behavior of context aware application.

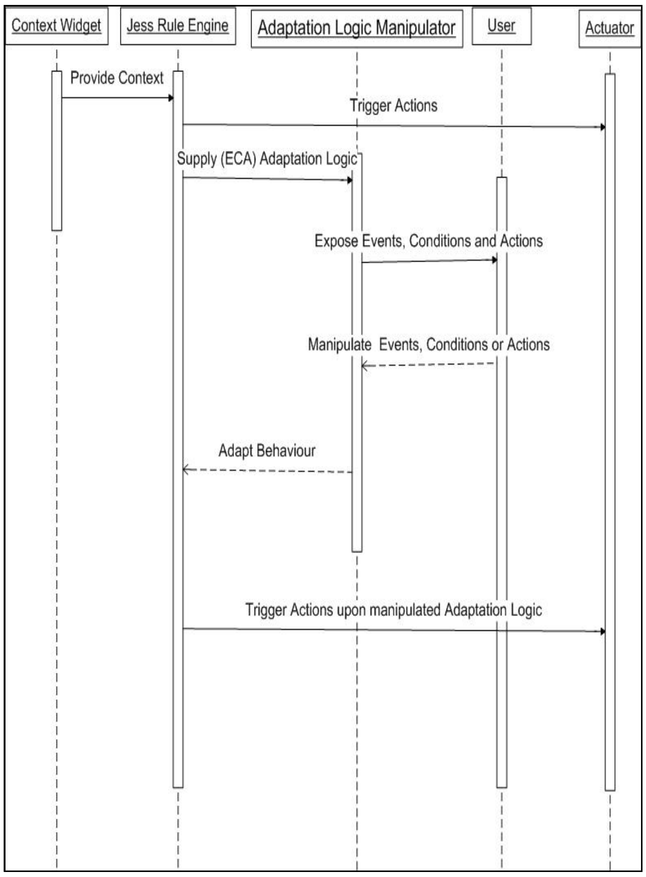

Context Widget
Jess Rule Engine
Adaptation Logic Manipulator
User
Actuator
Provide Context
Trigger Actions
Supply (ECA) Adaptation Logic
Expose Events, Conditions and Actions
Manipulate Events, Conditions or Actions
Adapt Behaviour
Trigger Actions upon manipulated Adaptation Logic

Figure 5. Sequence Diagram for the Adaptation Logic Manipulation

Context widgets provide the context information to the rule engine which fires appropriate rules suitable for the current situations. The rules present in the rule engine stimulate actuators to perform the intended actions for the smart home users. The main components of the proposed approach work in between rule engine and user. Rule engine supplies parameters and logic information to UCM (User Control Manager), which presents that information for further exposition and manipulation to the end users. Users change their desired actions and preferences using GUI, then UCM turn those changes back to rule engine for modification of application's behavior.

5. Evaluation and Results

We have performed usability study [25] to evaluate the proposed system. As discussed earlier we have implemented our system on three different smart home applications. We have used those three applications for evaluation of our system.

5.1 Usability Study Participants

We have selected a number of participants from diverse community of University of Sindh (Mirpurkhas Campus), for performing evaluation of the developed system. For this purpose we recruited 52 users of University from different disciplines. The demographic information of the participants is given in Table 1.

Table 1. Participants' Demographic Information

Gender	Percentage
Male	60%
Female	40%
Age	**Percentage**
18 - 25	86%
26 or Above	14%
Education	**Percentage**
Higher Secondary Education	86%
Bachelor	10%
Masters	4%
Have an Idea about Smart Environments	**Percentage**
Yes	69%
No	31%
Experienced Change in Preferences	**Percentage**
Yes	71%
No	29%

5.2 Test Procedure

We have performed usability study on three different smart home applications to demonstrate the usefulness of our proposed approach. All participants were given three applications to run with and without our proposed system. For those participants who were unaware of context aware applications, we have arranged an introductory session to introduce them about context aware applications. Later they were asked to use the same methods to test the applications as that of other familiar users.
After testing the proposed system, users were asked to give their feedback. We used questionnaire as a tool to to see the satisfaction of users on the implemented applications. We used 7-scale Likert score metrics based questionnaire for this purpose as suggested by [5].

5.3 Graphical User Interface (GUI) Evaluation

Users were very first asked to evaluate the usefulness of Graphical User Interface of the proposed system. In this part of evaluation users were supposed to evaluate GUI of Parameter Manipulator and Adaptation Logic Manipulator. Figure 6 shows the users overall satisfaction score for Parameter Manipulator and Adaptation Logic Manipulator on a 7-scale rating score. We can see users are satisfied changing parameters as well as adaptation logic of context aware applications using our system with the satisfaction score of 5.66 and 5.16 respectively.

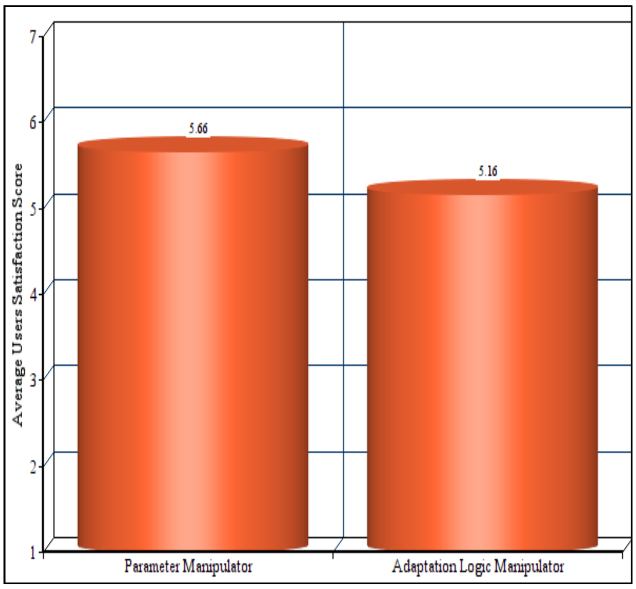


Figure 6. Users Satisfaction Score for User Control Manager

5.4 User Control Evaluation in Smart Home Applications

After evaluating GUI, users were asked to evaluate overall User Control support in all three smart home applications. Figure 7 illustrates users score for User Control in three different Smart Home applications.

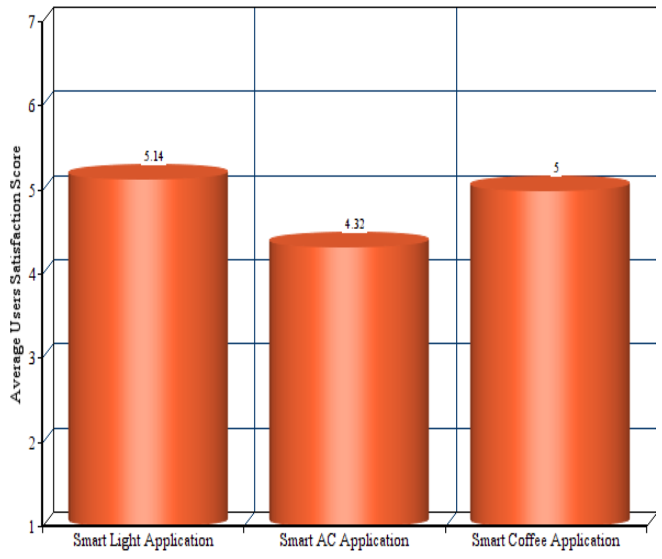


Figure 7. Users Satisfaction Score for Smart Home Applications

6. Conclusion

Smart home applications adapt themselves and perform users' tasks autonomously based on context. While software autonomy plays a vital role in realizing pervasive computing vision, users feel loss of control over their applications. We have identified various situations that require users to control the applications. These include home applications may not behave as desired; the users may want to reprogram or edit the behavior of the home; new application behaviors may be identified and integrated with existing behaviors, etc. We have conducted a literature survey on user control approaches, which suggests that existing approaches provide user control by exposing and manipulating parameters of the applications only. While addressing this issue we proposed an approach and build a system on it. We have implemented three different smart home applications and performed a usability study which shows the proposed system provides a wider support of user control by exposing and manipulating other parts of the applications e.g. adaptation logic(s), in addition to application parameters, and thus allows adding new application behaviors by end users not foreseen during design time.

7. Future work

Existing approach have few limitations as only the predefined set of rules and context internals are seen exposed using our approach. While there could be a better work to extend the existing approach by letting users add the behaviors entirely as they wish.
Second thing that needed to be enhanced is to provide another mechanism for controlling applications behavior e.g. voice based commands or text based commands.
Another notable future extension of the research work is regarding dynamic user control interface design i.e. the UCM should be smart enough to detect the context aware application parameters and rules and design the interface for users depending on those parameters and logics.

Acknowledgements. Part of the work in this paper is taken from the MPhil thesis of first author.

References

[1] Abowd, Gregory D., Anind K. Dey, Peter J. Brown, Nigel Davies, Mark Smith, and Pete Steggles. "Towards a better understanding of context and context-awareness." In *International symposium on handheld and ubiquitous computing*, pp. 304-307. Springer, Berlin, Heidelberg, 1999.

[2] Alferes J.J., Banti F., Brogi A. (2006) An Event-Condition-Action Logic Programming Language. In: Fisher M., van der Hoek W., Konev B., Lisitsa A. (eds) Logics in Artificial Intelligence. JELIA 2006. Lecture Notes in Computer Science, vol 4160. Springer, Berlin, Heidelberg

[3] Amando P. Singun , 2017. Usability Metrics for a Web-Based Test Blueprint System. *Journal of Engineering and Applied Sciences, 12: 6898-6903.*

[4] Barkhuus, Louise, and Anind Dey. "Is context-aware computing taking control away from the user? Three levels of interactivity examined." UbiComp 2003: Ubiquitous Computing. Springer Berlin/Heidelberg, 2003.

[5] Chalmers, Daniel. Contextual mediation to support ubiquitous computing. Diss. University of London, 2002.

[6] Chen, Guanling, and David Kotz. A survey of context-aware mobile computing research. Vol. 1. No. 2.1. Technical Report TR2000-381, Dept. of Computer Science, Dartmouth College, 2000.

[7] Cypher, Allen, ed. Watch what I do: programming by demonstration. MIT press, 1993.

[8] Dey, Anind K., Gregory D. Abowd, and Daniel Salber. "A conceptual framework and a toolkit for supporting the rapid prototyping of context-aware applications."Human–Computer Interaction 16.2-4 (2001): 97-166.

[9] Dey, Anind K., and Tim Sohn. "Supporting end user programming of context-aware applications." IST PROGRAMME (2003): 23.

[10] Dey, Anind K., et al. "a CAPpella: programming by demonstration of context-aware applications." Proceedings of the SIGCHI conference on Human factors in computing systems. ACM, 2004.

[11] Dey, Anind K., and Alan Newberger. "Support for context-aware intelligibility and control." Proceedings of the 27th international conference on Human factors in computing systems. ACM, 2009.

[12] Dhomeja, Lachhman Das. "Supporting policy-based contextual reconfiguration and adaptation in ubiquitous computing." PhD diss., University of Sussex, 2011.

[13] Fong, Johnson. "Intelligibility and user control of context-aware application behaviours." 7th International Conference on Pervasive Services (ICPS2010). ACM Press, 2010.

[14] Fong, Johnson, et al. "Defeasible preferences for intelligible pervasive applications to enhance eldercare." Pervasive Computing and Communications Workshops (PERCOM Workshops), 2012 IEEE International Conference on. IEEE, 2012.

[15] Fong, Johnson, Jadwiga Indulska, and Ricky Robinson. "A framework to support intelligibility in pervasive applications." Pervasive Computing and Communications Workshops (PERCOM Workshops), 2013 IEEE International Conference on. IEEE, 2013.

[16] Geihs, Kurt, and Christoph Evers. "User intervention in self-adaptive context-aware applications." In *Proceedings of the Australasian Computer Science Week Multiconference*, pp. 1-8. 2016.

[17] Hardian, Bob, Jadwiga Indulska, and Karen Henricksen. "Balancing autonomy and user control in context-aware systems-a survey." Pervasive Computing and Communications Workshops, 2006. PerCom Workshops 2006. Fourth Annual IEEE International Conference on. IEEE, 2006.

[18] Hardian, Bob. "Middleware support for transparency and user control in context-aware systems." Proceedings of the 3rd international Middleware doctoral symposium. ACM, 2006.

[19] Hardian, Bob, Jadwiga Indulska, and Karen Henricksen. "Exposing contextual information for balancing software autonomy and user control in context-aware systems." Proceedings of the Workshop on Context-Aware Pervasive Communities: Infrastructures, Services and Applications. 2008.

[20] Humble, Jan, Andy Crabtree, Terry Hemmings, Karl-Petter Åkesson, Boriana Koleva, Tom Rodden, and Pär Hansson. "“Playing with the Bits” User-configuration of Ubiquitous Domestic Environments." In *International Conference on Ubiquitous Computing*, pp. 256-263. Springer, Berlin, Heidelberg, 2003.

[21] Jess homepage. 2017 Available at [http://herzberg.ca.sandia.gov/jess/].

[22] Kawsar, Fahim, et al. "A portable toolkit for supporting end-user personalization and control in context-aware applications." Multimedia Tools and Applications47.3 (2010): 409-432.

[23] Li, Yang, Jason I. Hong, and James A. Landay. "Topiary: a tool for prototyping location-enhanced applications." Proceedings of the 17th annual ACM symposium on User interface software and technology. ACM, 2004.

[24] Newberger, Alan, and Anind Dey. "Designer support for context monitoring and control." IRB-TR-03-017, Intel Research Berkeley (2003).

[25] Sharp, H. Rogers. "Y., and Preece, JJ Interaction Design: Beyond HCI." (2007)

[26] Skillen, Kerry-Louise, Liming Chen, and William Burns. "VIPR: A visual interface tool for programming semantic web rules." In *2016 Intl IEEE Conferences on Ubiquitous Intelligence & Computing, Advanced and Trusted Computing, Scalable Computing and Communications, Cloud and Big Data Computing, Internet of People, and Smart World Congress (UIC/ATC/ScalCom/CBDCom/IoP/SmartWorld)*, pp. 277-284. IEEE, 2016

[27] Weiser, Mark. "The computer for the twenty-first century (pp. 94-100)."Scientific American, September Issue (1991).