Presented at the Intelligent Campus International Symposium (IC'12), 4 - 7 December 2012, Macau, China

Personalising the iCampus; an End-User Programming approach

Jeannette Chin
Computing and Technology
Anglia Ruskin University
Cambridge, United Kingdom
jeannette.chin@anglia.ac.uk

Vic Callaghan
Computer Science and Electronic Engineering
University of Essex
Colchester, United Kingdom
vic@essex.ac.uk

Adrian Winckles
Computing and Technology
Anglia Ruskin University
Cambridge, United Kingdom
adrian.winckles@anglia.ac.uk

***Abstract*— This paper explores the possibility of facilitating better end user engagement with the iCampus by providing a platform (Pervasive-interactive-Programming) to program the functionality of iCampus intelligent environments. We first introduce Pervasive-interactive-Programming (PiP), explaining the principles and presenting some recent results. By way of an example, we discuss how end-user programming could be used to configure the functionality of a student campus dormitory. We then consider how these techniques might be expanded to cater for other iCampus' areas. Finally, we comment on the future direction of our research.**

Keywords; Pervasive Interactive Programming; End User Customisation; Smart Intelligent Environments; iCampus; iClassrooms; e-learning; Virtual Appliances; Virtual Applications; Smart Laboratories

I. INTRODUCTION

The advance of the Internet has dramatically changed our view of the world. Today, services that we once had to travel physically some distance for are delivered to us via the World Wide Web. Take banking as an example, there is no longer a need to try and make closing time or wait patiently in a long queue. Similarly, food and clothing is all available for delivery to our home at the click of a mouse. New online services are appearing all the time and it's us, the end users, that ultimately decide when, where and how to use the services being offered. This includes education, which is no longer restricted to being delivered to college or university based classrooms, rather online learning has made it possible for education to be taken anyplace, anytime.

The Internet also changed our view of living environments too! Digital homes and smart student dormitories have been born thanks to the trend of making domestic appliances network enabled. We are living in an electronic space that our grandparents could never dream of. Our living environments today are populated with networked appliances, sensors and actuators, be it static or mobile, that are constantly exchanging information with one another via the services offered by these networked devices. It is possible to create an intelligent environment such that, by coordinating the actions of networked devices or services, the environment will behave in a holistic and reactive manner to satisfy the end users' evolving needs. Furthermore, it is possible to construct a new user defined networked -appliances or -applications, by combining different network services together in novel ways, to create the so-called '*virtual appliance*' [1]. This principle can be extended to decompose and re-compose software applications allowing users to create their own bespoke applications. Collectively, such user created entities are referred to as Meta –appliances or -applications, more generally abbreviated to MAps.

Deconstruction, reconstruction and the user customized MAps, that describe network aggregated applications, raise exciting possibilities for end users of future intelligent environments and set significant research challenges [2]. But how can MAps be constructed and managed by ordinary non-expert end users who may have privacy concerns about fully automated systems [5] or who may wish to be the designer themselves? This question has led many

researchers to investigate what is termed '*end-user programming*', a methodology aimed at allowing non-technical people to personalise their own digital spaces with aggregations of network enabled embedded-computer based devices. Historically, programming has only been accessible to well-qualified professionals, such as computer scientists, or the outcome of self-programming autonomous intelligent agents. The challenge for achieving an end-user programming vision is to devise programming methodologies that are usable by non-technical people who, in the case of the iCampus, would be students and staff.

II. CONTEMPORATY WORK

Empowering end users has become a crucial factor in every aspect of designing modern network enabled environment, such as an iCampus, for a very simple reason – to 'serve' the users better by allowing them to customize their environments to suit their individual needs! Various approaches to allow end users to customise their space have been investigated by researchers, ranging from intelligent agents [4] through to user driven methods [3].

To date, most of the work has been based around the use of rules, which form the basis of governing the behaviour of the systems. Zamudio [12] reported rules are fundamental to determining how a given networked appliance or service interacts and Chin [3] proposed a taxonomy based on 'rule formation' that is divided into three categories: Pre-programmed rules [11][7][6], Agent-programmed rules [8][5][12] and User-programmed rules [9][13][10][3], as a way of describing customised networked environment programming.

III. END USER PROGRAMMING FOR THE ICAMPUS

The modern university campus is a computer network intensive environment, with computer based devices being used in a variety of applications from building services (heating lighting), through digital signage, administration to learning content and delivery [15]. The introduction of computer networks and devices is gradually transforming the campus into an increasingly virtualized world where physical and soft entities intermingle in an ever-richer information and electronic space to create new lifestyles and opportunities. Computer technology is programmable, a feature that allows it (and what it controls) to be readily customized to match the stakeholders evolving needs. However, the complexity of the systems means that programming, and reprogramming, has become the preserve of those people with appropriate technical skills, frequently requiring computer science degrees to master. This often results in such systems not serving their stakeholders as well as they might, with people invariably accepting the functionality pre-programmed into the system, even if it falls short of their needs.

In this paper we present a possible solution to the problem of how to enable non-technical iCampus stakeholders to "program" future fully networked iCampus systems. There are various pillars in an iCampus system [14], in this paper we direct our attention at the building services area, as that is a topic that has a large impact on the most basic parameters of user comfort and the institution's energy cost. In other papers we explore applying end-user programming to the other iCampus pillars such as teaching and learning [16].

IV. PERVASIVE INTERACTIVE PROGRAMMING

A. *Terminology*

Pervasive-interactive-Programming (PiP) is a programming paradigm that allows non-technical end users to customise their environments without requiring them to write any programming code but instead they simply demonstrate the desired functionalities to the "system".

A fundamental concept underpinning PiP is the notion of deconstruction and reconstruction of networked services or appliances which has lead to the creation of so called '*Virtual Applications*' (alternatively referred to as '*Virtual Appliances*'), an aggregation of networked services composed by the end users with a goal of performing a particular task or a series of actions depending on a given situation [1]. At a higher level, we use an abstraction referred to as a *Meta-Appliances/Applications* (MAps), which can be

viewed as soft objects that define the membership and behaviour of a collection of networked services. At a logical level, a MAp can be referred as a collection of rules that govern the behaviour of a community of coordinating networked devices and services. In terms of the iCampus, these networked services range from physical devices (such as lightings or heating systems) to software applications (such as word processors or email clients), which provide a service via the network. From the end-users' points of view, a MAp is an icon that captures the description of how the environment should behave according to their needs. MAps can be created under the direction of end users, through show-me-by-example techniques [3], to provide the functionalities the users' desire in the environment. By employing a flexible representation (MAp), together with an intuitive end user interaction mechanism (show-me-by-example), PiP is well suited to iCampus environments.

B. System Architecture

PiP was designed to work in real-time within small to medium sized pervasive networked environments (such as a offices or class rooms). The model was designed as an event-driven architecture to encourage hands-on activities. The middleware is based around UPnP, a framework that facilitates communication between networked applications/services (see figure 1).

Figure 1. PiP high level Architecture

Unlike macro languages, PiP is independent of logical sequence, which makes it more compatible with the human tendency to complete job sub-tasks in a variety of orders. It achieves this by employing a rule policy comprising "*a set of conditions that are satisfied if the conditions defined within the context of this set are all satisfied*". For instance this statement: "*if the telephone is ringing and if the audio is playing then stop the audio and raise the light level*" will have the same logical meaning as the following statements:

- "if the telephone is ringing and if the audio is playing then raise the light level and stop the audio"
- "if the audio is playing and if the telephone is ringing then stop the audio and raise the light level"
- "if the audio is playing and if the telephone is ringing then raise the light level and stop the audio".

PiP is based on a modular framework, made up of six core components which support real-time coordination of network-based devices –

1. Interaction Execution Engine (IEE) - manages communication between PiP components, from network to user level. This component features a 2-way function that, for in-bound functions, makes calls to the software component upon receiving networked events and, for the out-bound functions, packages network related actions which are sent as requests to the network.
2. Event Handler (EH) - manages the passing of events between interested parties, from network level to user level
3. Knowledge Engine (KE) - maintains records of all MAps as well as the information about the environments including devices and services. This component is responsible for updating the current status of device and maintaining up-to-date content of the knowledge bank.
4. Real-time MAp Maintenance Engine (RTMM) – This component is responsible for managing, assembling and instantiating MAps
5. Real-Time Rule Formation Engine (RTRF) – The main task for this component is to assemble rules formed by monitoring user interactions with the environment
6. GUI (PiPView) - a means for users to interact with the system. It utilises the "look and feel" of PC editing applications and "drag drop" methods. The implementation employs a multi-threaded approach to make concurrent activities possible.

C. *Testing and Evaluations*

PiP was evaluated in the Essex iSpace, a campus based high-tech student dormitory test-bed. The PiP trials were conducted in two phrases, both of which involved participants completing a set of questionnaires. The first phrase was an online survey conducted whereas the second was physical experiments in which participants were invited to use the system. Their behaviour was monitored and recorded plus their views were collected via a questionnaire.

The online survey was designed to provide a high-level insight into people's preferences for programming digital environments. Three approaches were evaluated (1) Coding (manual coding) (2) Analogy (coding via manipulating graphical representations) (3) Reality (coding via physical demonstration). In brief, the results showed that only 7% of the 69 participants thought that programming using real objects (Reality) was difficult to understand while 45% thought the other programming approaches were more difficult. Another aspect that was evaluated was the mental effort involved in each approach. 59% of the online participants concluded that manual coding (Coding) required a lot of mental effort whilst only 11% thought programming via using physical actions (Reality) to create functionality was difficult. Concerning the physical experiments, out of the 18 participants, 83% were able to use PiP to customise their personal space with little or no assistance and reported they found PiP very intuitive to use. The results were published in greater detail in an earlier paper [2] and showed conclusively that ordinary people found programming via physically demonstrating required system behaviour to be relatively easy compared to the other approaches.

D. *Discussions*

The results from both trials, demonstrated that, although there is room for improvement, the proof-of-concept prototype we employed had sufficient fidelity to show that the concept of end-user programming of iCampus services by non-specialist is viable.

Whilst we only experimented with PiP in respect to one aspect of the iCampus, a student dormitory, it is clear that PiP technology could be applied to any part of an iCampus building where there are similar networked appliances such as lights, HAVACs, fans, window blinds, media services etc. For example users in an iCampus could use PiP to customise a smart-classroom, office, sports hall etc to their likings and their personal preferences could be stored and reused in other environment as they move about the iCampus. By way of a scenario, consider a smart computer laboratory where students' workstations are essentially virtual desktops or machines that could be located anywhere within the cloud. The settings/profiles for each virtual desktop that also define users' preferences are encapsulated within the system. These settings/profiles could give the student a "read" only style of experience while at the same time, providing the academic a view of what the student is doing or he or she could gain full control and participate fully in his/her user experience. In future work we would like to explore the usefulness of this approach in a more fine grained evaluation, differentiating between differing groups of stakeholders (eg students, maintenance staff, academics etc) and different types of building (laboratories, offices, lecture rooms, dormitories etc). We are especially interested in moving our work into programming smart classrooms or laboratories for providing a platform to facilitate end users engagement and improve their experience (see figure 2).

Figure 2 – The Essex iClassroom

We would also like to explore how MAps might be shared between different iCampus stakeholders and even between University

campuses. In addition, we would like to explore how the *cloud*, via the Internet, might be used to open up the possibility of sharing MAps or functioning as a store of user preferences, which could be mined by management to improve the iCampus services.

V. CONCLUSSION

In this paper we have introduced a new approach to programming the iCampus called Pervasive-interactive-Programming (PiP). We have described how it works and presented a summarized version of an evaluation focused on a student dormitory. Our aim was to determine whether an end-user approach, based on programming-by-demonstration or, show-me-by-example, could enable non-specialist stakeholders of an iCampus to customize their building services. Our results have shown that using an end-user programming paradigm, PiP in our case, it is possible for non-technical stakeholders of an iCampus to customize their environments. Furthermore, we compared our approach to two other standardized approaches to programming, manual and graphical programming and found that our approach, involving physically demonstrating required building service behaviour, was preferred. There remains more work to be done to make our evaluation and conclusions more robust, but at this stage we can report that the results are encouraging and we intend to move forward with refining our system with the hope that end-user programming can play an important role in future programming of iCampus systems and services.

REFERENCES

[1] J. Chin, V. Callaghan, G. Clarke, "End-user Customisation of Intelligent Environments". In the handbook of Ambient Intelligence and Smart Environments, Editors: H. Nakashima, J. Augusto, H. Aghajan, Springer, 2010, Spring, pp 371-407, ISBN 978-0-387-93807-3

[2] Chin J, Callaghan V, Clarke G, "Soft-appliances: A vision for user created networked appliances in digital homes", Journal of Ambient Intelligence and Smart Environments 1 (2009) 65–71, DOI 10.3233/AIS-2009-0010, IOS Press, 2009

[3] Chin J, Callaghan V, Clarke G, "A *Programming-By-Example Approach To Customising Digital Homes"* IET International Conference on Intelligent Environments 2008, Seattle, 21-22 July 2008

[4] Callaghan V, Clark G, Colley M, Hagras H Chin JSY, Doctor F "Intelligent *Inhabited Environments"*, BT Technology Journal , Vol.22, No.3 . Klywer Academic Publishers, Dordrecht, Netherlands, July 2004

[5] Callaghan V, Clarke G, Chin J "Some Socio-Technical Aspects Of Intelligent Buildings and Pervasive Computing Research", Intelligent Buildings International Journal, Vol 1 No 1, Published Autumn 2008, ISSN: 1750-8975, E-ISSN: 1756-6932

[6] Barry Brumitt, Brian Meyers, John Krumm, Amanda Kern and Steven A.Shafer, "EasyLiving: Technologies for Intelligent Environments", Proc of the 2nd international symposium on Handheld and Ubiquitous Computing, 2000, 12 – 29

[7] G. Chen and D. Kotz. "A survey of context-aware mobile computing research", Paper TR2000-381, Department of Computer Science, Darthmouth College, November 2000.

[8] D.J. Cook, M. Huber, K. Gopalratnam and M. Youngblood, "Learning to Control a Smart Home Environment", Innovative Applications of Artificial Intelligence, 2003

[9] Humble, J. et al "Playing with the Bits", User-Configuration of Ubiquitous Domestic Environments, Proceedings of UbiComp 2003, Springer-Verlag, Berlin Heidelberg New York (2003), pp 256-263

[10] Gajos K., Fox H., Shrobe H., "End User Empowerment in Human Centred Pervasive Computing", Pervasive 2002, Zurich, Switzerland, 2002

[11] C. Kidd, R. Gregory, A. Christopher, T. Starner. "The Aware Home: A Living Laboratory for Ubiquitous Computing Research", In Proceedings of the Second International Workshop on Cooperative Buildings (CoBuild'99), October 1999

[12] Zamudio V , Callaghan V , "Facilitating the Ambient Intelligent Vision: A Theorem, Representation and Solution for Instability in Rule-Based Multi-Agent Systems" International Transactions on Systems Science and Applications (special issue on "Agent based System Challenges for Ubiquitous and Pervasive Computing"), Vol 4, No. 1, 2008

[13] Truong, KN.et al " CAMP: A Magnetic Poetry Interface for End-User Programming of Capture Applications for the Home", Proceedings of Ubicomp 2004, pp 143-160.

[14] Jason Ng "The Intelligent Campus – iCampus – end-to-end learning lifecycle of a knowledge ecosystem", The 6th International Conference on Intelligent Environments, Monash University (Sunway Campus), Kuala Lumpur, Malaysia , 19-21 July, 2010

[15] K. Guild, M. Paredes Farrera, R. Martin, R. Almeida, A. Bontozoglou, M. Patel, K. Yang, V. Callaghan, "STUDENT: Scenarios, Technologies & Users within the Digital Essex Network Testbed", 6th International Conference on Intelligent Environments (IE'10), Kuala Lumpur, Malaysia. 19-21st of July 2010

[16] J. Chin, V.Callaghan "A Show Me By Example Approach to Teaching Programming the Internet-of-Things in Immersive Education**"** Immersive Education 2012 (iED'12), Paris, France, 26th and 27th November 2012.

