SPOK, an End-User Development Environment for Smart Homes

Joëlle Coutaz[1], Alexandre Demeure[2], Sybille Caffiau[1]
J.R. Courtois[2], T. Flury[1], C. Gérard[2], C. Lenoir[2], K. Petoukhov[2], P. Reignier[2], C. Roux[1], G. Vega[1]
[1]Université Grenoble Alpes, CNRS, LIG F-38000 Grenoble, France, {firstname.lastname}@imag.fr
[2] Inria Grenoble Rhônes-Alpes, 38330 Montbonnot, {firstname.lastname}@inria.fr

ABSTRACT

We propose a live demonstration of SPOK, an End-User Development Environment for smart homes. SPOK (Simple PrOgramming Kit) runs on top of ApAM, a component-oriented middleware developed on top of OSGi, that supports the dynamic and resilient management of devices and web services as well as the integration of a diversity of protocols such as EnOcean, UPnP, and Watteco. Video of SPOK in action as of October 2013: http://iihm.imag.fr/demos/appsgate/appsgate2013.mp4.

Keywords

End-User Programming, End-User Development, Smart home, Smart environment, Ambient Intelligence, Ubiquitous Computing.

1. INTRODUCTION

SPOK (Simple PrOgramming Kit) is an End-User Development Environment (EUDE) that permits non-specialists in the area of ubiquitous computing to configure, define and control the behavior of their smart home using a pseudo-natural language. With SPOK, inhabitants are not doomed any more to consume the rigid solutions imposed by home automation. Instead, they can improve the quality of their everyday life by developing programs that fit their needs in an opportunistic manner.

The "Do-It-Yourself Smart Home" is becoming popular. For example, commercial home boxes such as the ZipaBox and the Vera, are intended for non specialists. In particular, the Scratch-based programming language [2] of the ZipaBox and the rule-based IFTTT propose an attractive graphics concrete syntax and stylistics. In the next section, we describe how SPOK and its infrastructure depart from the state of the art.

2. KEY FEATURES

SPOK and its companion services (e.g., context manager, devices and Web services abstracted in a uniform manner to hide heterogeneity) are ApAM-compliant components that are dynamically composed and maintained by the ApAM infrastructure.

2.1 ApAM Key Features

ApAM [1] is a component-oriented middleware that extends OSGi/iPOJO in two ways: (1) developers describe an application by intention using a dedicated language as opposed to explicitly specify components composition and bindings at design time; (2) from the abstract description of the application architecture, a concrete architecture is computed and updated incrementally by resolving the dependencies between the components currently available in the execution environment.

With the system described in this paper, a smart home application is composed of SPOK and of its companion components. Due to the incremental and dynamic construction and maintenance made possible by ApAM, the smart home application is resilient to the opportunistic installation or disappearance of devices, sensors, actuators, and Web Services.

2.2 SPOK Key Features

End-users express their programs using a rule-based programming language whose concrete syntax is a pseudo-natural language. The grammar is extensible dynamically with the grammar that comes with each new class of devices. If end-users are not pleased with the vocabulary, SPOK allows them to modify the terminals. The SPOK editor supports feedforward by providing end-users with a "Smart Keyboard" that contains the appropriate set of correct instructions and terminals for the current entry point.

The SPOK language supports encapsulation (a program can call another program) as well as parallelism at multiple levels of granularity: several programs can be executed in parallel, and several instructions within a program can be executed in parallel. For conditional statements, conditions may include events (e.g., the light has been switched off), or states (e.g., the light is off) in order to support both instantaneous (inchoative) and long-duration (durative) actions.

3. THE DEMONSTRATION PER SE

SPOK and its infrastructure will be demonstrated using a variety of devices and web services: EnOcean sensors and actuators, Philips Hue lights, email, weather forecast, fan, tablets, and the DomiCube. The DomiCube is a "home-made" graspable Bluetooth enabled cube that contains an accelerometer and a gyroscope, and whose faces can be dynamically linked to any home behavior.

First, we will demonstrate the dynamic and resilient management of devices and web services provided by the infrastructure. Then, we will show how to develop and execute programs with SPOK and let UbiMob participants build and run their own programs.

4. ACKNOWLEDGMENTS

This work is supported by the European Catrene AppsGate CA110 project. The DomiCube was built in the Creativity Lab of the EquipEx AmiQual4Home, ANR-11-EQPX-0002.

5. REFERENCES

[1] Damou, E. 2013. *ApAM : un environnement pour le développement et l'exécution d'applications ubiquitaires*. Thèse de doctorat de l'Université de Grenoble, spécialité informatique.

[2] Resnik, M., Maloney, J., Monroy-Hernández, A., et al. 2009. Scratch: Programming for all. *Communication of the ACM,* 52, 60–67.