homeBLOX: Making Home Automation Usable

Marcel Walch
Institute of Media Informatics
Ulm University
89069 Ulm, Germany
marcel.walch@uni-ulm.de

Florian Schaub
Institute of Media Informatics
Ulm University
89069 Ulm, Germany
florian.schaub@uni-ulm.de

Michael Rietzler
Institute of Media Informatics
Ulm University
89069 Ulm, Germany
michael.rietzler@uni-ulm.de

Björn Wiedersheim
Institute of Media Informatics
Ulm University
89069 Ulm, Germany
bjoern.wiedersheim@uni-ulm.de

Julia Greim
Institute of Media Informatics
Ulm University
89069 Ulm, Germany
julia.greim@uni-ulm.de

Michael Weber
Institute of Media Informatics
Ulm University
89069 Ulm, Germany
michael.weber@uni-ulm.de

UbiComp'13 Adjunct, September 8–12, 2013, Zurich, Switzerland.
ACM 978-1-4503-2215-7/13/09.

http://dx.doi.org/10.1145/2494091.2494182

Abstract
Home automation aims to increase convenience of residential living. The homeBLOX system uses a process-driven execution model to enable complex automation tasks with heterogeneous devices, while providing a user interface that abstracts from lower-level complexity. Complex automation tasks are created as sequences consisting of events and actions linked to physical and virtual devices, which are translated into BPEL code for execution. We outline the key concepts, architecture, and prototype of our system.

Author Keywords
Home automation; UPnP; BPEL; Arduino; UCD; usability.

ACM Classification Keywords
H.4.m [Information systems applications]: Miscellaneous; H.5.2 [Information interfaces and presentation]: User interfaces.

Motivation
Home automation aims to enhance residential living comfort by interconnecting household devices. However, most systems need to be set up by a specialist or require extensive tinkering by passionate users [4]. Recent projects aim to improve home automation. HomeOS [2] is a middleware providing programming interfaces and services

to manage a variety of devices. While developers can create applications leveraging those capabilities, end users must hope that some application addresses their use case. Busemann et al. [1] propose a flexible Connector system for sensor nodes, where a modular set of protocol adaptors abstracts from the sensors' heterogeneous implementation. NinjaBlocks[1] provides a Web-based rule engine for less tech-savvy users. The creation of a set of *if-this-then-that* rules utilizes events and actions offered by the user's devices. While this approach allows for individualized home control, use of many devices and scenarios likely results in a complex set of hard to maintain, potentially interfering rules. Guinard's extension of the ClickScript[2] mashup editor [3] utilizes the connection of device blocks to represent rule creation involving RESTful services. For the FHEM[3] home control server, multiple user interfaces (device lists, remote controls, device-floorplan models etc.) exist, which demonstrate the complexity of home automation interfaces. Thus, current systems face the dilemma of either limiting individual control through predefined but usable applications or providing highly flexible but complex controls. We propose homeBLOX to bridge this gap and facilitate usable, yet highly customizable home automation. Our approach combines a modular architecture for flexible integration of heterogeneous devices and protocols with a graphical abstraction that simplifies configuration of home automation tasks as processes, while supporting complex scenarios that exceed existing rule-based approaches.

Process-driven Home Automation

While rule-based approaches facilitate straightforward specification of simple rules, more complex automation scenarios result in difficult to maintain rule sets, due to the complexity of rule conditions. Our approach uses processes rather than rules to specify automation tasks. Processes have the advantage that they can model temporal dependencies: a single process can describe an automation sequence that combines events and actions of different devices at different steps. Thus, over the course of a process, context information is aggregated rather than describing the entire situation in a rule's condition part. However, while processes have the potential to simplify home automation, the user interfaces for configuring process engines are notoriously complex[4]. Therefore, we designed a graph-based process notation in which each node constitutes a device action and every edge a triggering event. We further developed step-by-step wizards to support users in the creation of automation sequences in order to make process creation less difficult and error-prone. We refined our interface through multiple iterations of user testing and feedback. Our system further abstracts from the heterogeneity of different home control protocols. Protocol-specific controllers forward incoming events from connected devices to the internal middleware and trigger actions on them. A light-weight driver concept reduces configuration when integrating new devices and abstracts from the specific protocol supported by the device. Drivers summarize and describe the abilities of a device to ensure consistent representation of all devices in the user interface.

homeBLOX Architecture

The homeBLOX system consists of a server for device management and a set of smart devices placed in the user's home. A tablet application enables users to manage automation sequences and interact with the system.

[1] Ninja Blocks Inc. - The Ninja Platform, http://ninjablocks.com/
[2] Lukas Naef - ClickScript, http://clickscript.ch
[3] Koenig, Haas and Droegehorn - FHEM, http://fhem.de/
[4] e.g. intalio|bpms, http://www.intalio.com/products/bpms/

Figure 2: Wizard to con gure the connection *Getting up* and *Co ee Maker*: event selection (top), action selection (middle), action con guration (bottom).

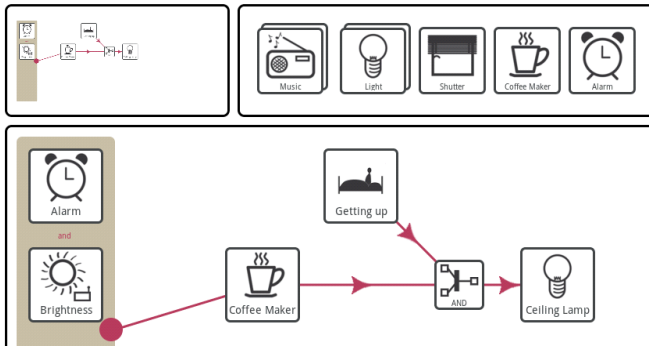

Figure 1: Sequence editor with an example sequence for automating co ee making and lamp control when getting up.

User Interface

An Android-based tablet app provides the homeBLOX user interface to create and manage automation sequences. In the sequence editor, the user creates sequences by dragging available devices, represented as block icons (blox), from the top tool bar onto the canvas and drawing connections between them. Figure 1 shows an exemplary sequence. The canvas has a dedicated start area on the left to de ne the triggers for a sequence. When multiple blox are placed in the start area, the user is asked to choose the start semantics (logical *AND* or *OR*). Drawing a connection line between two blox on the canvas prompts a wizard to con gure the connection, which consists of at most three steps: event selection (ES), action selection (AS), and action con guration (AC). Figure 2 shows the steps of an exemplary wizard sequence. Only relevant steps are shown, e.g., step A is not displayed for connections originating from the start area, as triggering events are already de ned there. Step C only appears if the selected action has con guration parameters, e.g., the number of cups a co ee maker should brew. Whenever a user connects blox with multiple incoming links, the

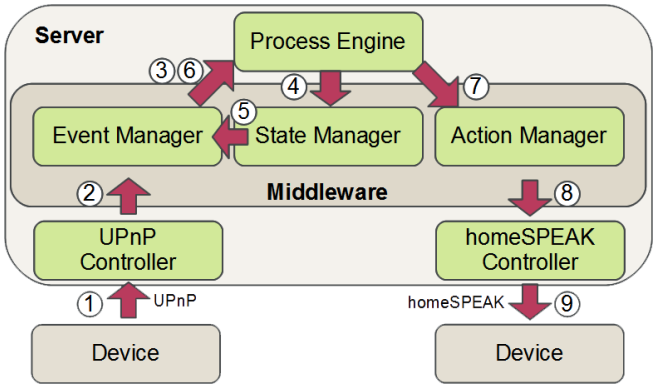

Figure 3: Execution of a home automation sequence.

system adds logical operation blox to visualize employed semantics. We followed a user-centered design process in the development of the sequence editor. Incremental user testing revealed, amongst other aspects, that users employ diverse strategies to create sequences: some users start by con guring a sequence s trigger events, while others drag all desired blox to the canvas rst, or begin with the outcome of the sequence. As a consequence, our sequence editor does not presuppose either strategy to equally support users regardless of their entry point.

Home Automation Server

The homeBLOX server receives events from connected devices and executes user-speci ed sequences. When a new device is set up in the house, it uses WiFi to connect to the server. A controller that ts the device s protocol (e.g., UPnP) handles communication with the device. Based on the device s name, a driver is obtained and loaded that governs how the device is represented in the sequence editor. Sequences created by the user on the tablet are sent to the server, where they are translated into business process execution language (BPEL) code

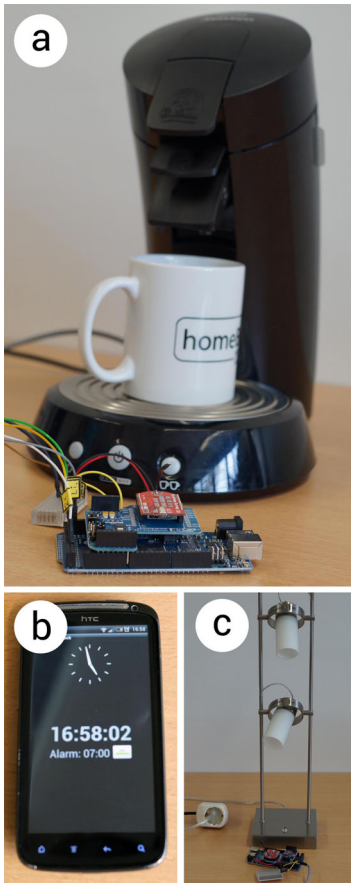

Figure 4: homeBLOX smart devices: (a) smart coffee maker, (b) UPnP alarm clock, and (c) smart lamp.

and deployed on our process engine, currently Apache ODE. Figure 3 shows how a sequence is executed. Devices send updates of their internal state to a controller (1), which maps values of state variables to specific events defined by the device driver (2). The Event Manager aggregates events and triggers the process engine (3). Some events represent a device state (e.g. *light is on*). To make such states available to the process engine, the State Manager keeps track of the device state (4) and provides the last known values to the Event Manager (5), which can then generate respective event notifications (6). The process engine evaluates processes to determine required actions. The Action Manager receives resulting action calls (7) and notifies the responsible controller (8). The controller then communicates with the appropriate device (9) that executes the action.

Smart Devices

Our homeBLOX testbed includes several types of devices, some are shown in Figure 4. We developed a UPnP controller that allows the control of any standard UPnP device, e.g. an alarm clock or media player, via homeBLOX. Online and system services, such as weather, time, or calendars, can be encapsulated as virtual devices, which use SOAP to communicate with the homeBLOX server. Furthermore, we turned household appliances, such as lamps and a coffee maker, into smart devices by equipping them with Arduino controllers. Hereby, the actual device was treated as a sensor and/or actuator, whereas the WiFi-equipped Arduino handled communication as well as event and action processing. In order to meet the limitations of the Arduino platform, we defined a lightweight control and communication protocol (*homeSPEAK*) for interaction with the server and added a respective controller.

Conclusions and Future Work

The homeBLOX architecture facilitates complex home automation scenarios with heterogeneous devices and controls, while providing a user interface that abstracts from underlying complexity without limiting expressiveness. Consistent representation of devices is based on drivers and protocol controllers. We are working on a community-based approach to build a repository of drivers, controllers, and virtual devices created by users or manufacturers. Further enhancements could be multi-user support and personal user profiles. In contrast to related work, configuration of homeBLOX is process-driven rather than rule-based. Continuous user testing in the design process indicates that our sequence editor enables users to effectively create and recognize even complex automation scenarios. We are currently conducting further user studies to evaluate the effects of our process-driven approach on home automation usability.

References

[1] Busemann, C., Gazis, V., Gold, R., Kikiras, P., Kovacevic, A., Leonardi, A., Mirkovic, J., Walther, M., and Ziekow, H. Enabling the usage of sensor networks with service-oriented architectures. In *MidSens '12 workshop*, ACM (2012).

[2] Dixon, C., Mahajan, R., Agarwal, S., Brush, A., Lee, B., Saroiu, S., and Bahl, P. An Operating System for the Home. In *NSDI '12*, USENIX Assoc. (2012).

[3] Guinard, D. Mashing up your web-enabled home. In *ICWE '10 workshops*, Springer (2010).

[4] Takayama, L., Pantofaru, C., Robson, D., Soto, B., and Barry, M. Making technology homey: finding sources of satisfaction and meaning in home automation. In *UbiComp '12* (2012).