Vision: Towards an Extensible App Ecosystem for Home Automation through Cloud-Offload

Yuichi Igarashi
Hitachi Yokohama Research Laboratory
yuichi.igarashi.hb@hitachi.com

Kaustubh Joshi, Matti Hiltunen, and Richard Schlichting
AT&T Shannon Labs
{krj,hiltunen,rick}@research.att.com

ABSTRACT

Home security and automation—temperature, lighting and energy management, access control, and alarming—is an area of growth in residential and office settings. These systems typically include a number of sensors and actuators connected to a controller that runs the automation software. For cost reasons, these home controllers are often resource constrained devices that are not easy to upgrade or replace at scale. But with the emergence of more capable sensors, there is a need for applications that require significant amounts of computing resources, e.g., video feeds from cameras being used to identify people and their activities. The limited resources at the home controller makes it hard to deploy such applications, especially when numerous ones are being used concurrently. This problem is reminiscent of applications on mobile phones that necessitate cloud off-load. However, we posit that home control applications pose a new set of requirements unique to this domain. In this paper, we motivate a few such requirements including the need for disconnected operation and an offload decision engine with system-wide visibility, and propose an architecture to address them.

Categories and Subject Descriptors

D.4.7 [**Software**]: Organization and Design—*Distributed Systems*

General Terms

Design,Experiment,Reliability

Keywords

Programable, Cloud, Off-loading, Home Automation

1. INTRODUCTION

The emergence of a mobile app ecosystem built on touch-enabled smartphones has transformed the way billions of people interact with the digital world. Can a similar ecosystem of home and building automation applications transform the way we live in the physical world? And is there a device analogous to the smartphone, flexible yet unifying, around which such an ecosystem could be anchored? While home automation and smart spaces have had a long history with enthusiasts and in the academic literature since the late 90s [3, 10, 15, 9], it is only recently that the area has seen commercial offerings at scale, mainly from telecom providers and cable operators, e.g., [2]. The "device" such systems provide — the home controller — is a small physical box that is meant to be the central hub for home automation. It provides the platform that various sensors and actuators in the home connect to, and on which home automation applications run. On such a platform, it is easy to imagine a rich array of sensors, actuators, and applications, ranging from home security and energy management to assistance for aging in place.

However, as a unifying device around which a flexible ecosystem could be built, we postulate that the home controller's potential is yet to be unleashed. A critical aspect is the flexibility of the home controller itself. Unlike mobile phones, which are intensely personal devices that users are incentivized to upgrade frequently, home controllers are more intimately connected to homes rather than people. Once installed, they are difficult and expensive to upgrade at scale, often requiring on-site technical support. Therefore, service providers are left with a difficult choice - how to balance the computing power of a controller (and thus its cost) with the wide array of current and future uses that may be vastly different for different homes?

To be sure, requirements for various uses in the home are vastly different. A simple passcode based door lock application requires very few resources to manage, while the same door lock application, but equipped with a front-door camera and face recognition system requires significantly more (we provide some numbers in §4). Provisioning all home controllers for the latter use case makes the service more expensive for everyone, while provisioning for the former locks out the more advanced applications.

The problem becomes worse when all combinations of applications that the controller might be called on to support at a time are considered. It is little wonder that in today's home automation marketplace, service providers have to make decisions on what services can or cannot be supported apriori, based on worst case scenarios. The result is less opportunity for everyone — service providers who have to pick winners and losers with little apriori informa-

MCS'14, June 16, 2014, Bretton Woods, New Hampshire, USA.

http://dx.doi.org/10.1145/2609908.2609949.

tion about which applications will be popular, users that are limited to service provider choices, and third party developers who are locked out altogether.

We contend that this area is ripe for "cloudification". We propose making the home controller a truly flexible platform device by allowing its application hosting capabilities to be extended by cloud resources. Thus, applications might run either on the controller box when they can and partially or fully migrating to a cloud VM as and when needed. This would allow service providers to provision home controllers without fear that they would be unable to cope with new applications as they emerge.

The first contribution of this paper is to propose exactly such a programmable cloud-enabled home controller architecture. While at first glance it might appear that systems proposed in the vast literature on mobile offloading might work in the home automation space, our second contribution in §2 is to demonstrate that there are crucial differences between the two in terms of the user interaction model, the timesharing model, and in reliability requirements. The final contribution of this paper in §3 is to identify the mechanisms that will be needed to address the challenges and opportunities resulting from these differences. Finally, we conclude and discuss future work in §5.

2. CLOUD OFFLOADING FOR THE HOME

Motivating use-case: Home automation, such as home energy management and home monitoring, are gaining increasing interest both in terms of actual deployments in homes and standardization organizations [13][8]. The suite of remote controllable network devices for the home, e.g., temperature sensors, remote switches, door locks, lights, and cameras, is expanding rapidly. Home controllers typically communicate with these devices through a USB interface, e.g., a Z-Wave USB dongle, a ZigBee USB dongle, and a USB camera. There has also been work on designing operating systems specifically for home automation, such as the HomeOS [7]. HomeOS handles network devices as peripherals with abstract interfaces, simplifies the development of applications by providing higher-level abstractions, and gives users a management interface designed for the home environment. The aim of these technologies is to extend the home easily by adding new devices and applications.

With the increased range of possible home devices and applications that these devices enable (e.g., recognition of residents based on camera image and automatic unlocking of doors), the home automation system has to manage multiple applications at the same time, for example, energy management, monitoring inside and outside of the home (motion sensors, cameras), and automatic locking and unlocking of doors. Therefore, the home controller hardware quickly becomes a bottleneck for deploying a richer home automation experience and the option of deploying home automation applications in the "cloud" becomes an appealing option.

Figure 1 illustrates the components and their interactions in a cloud-enhanced home automation system where the standalone home controller has been replaced with a hybrid home controller consisting of home and cloud elements [13]. The home devices (sensors and actuators) and home element reside at the home while the cloud element is located at some Internet data center.

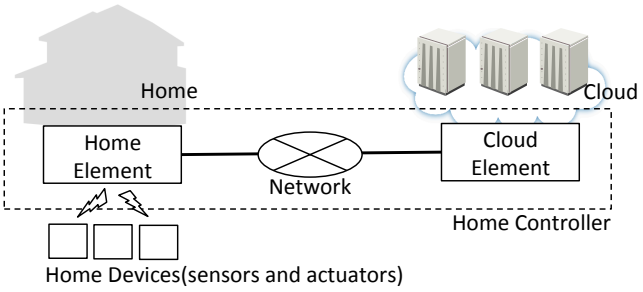

Figure 1: Cloud-enhanced home automation system.

Cloud offloading for mobile devices: The idea of offloading computation to the cloud has been utilized in the context of mobile computing. Mobile applications such as gaming, video, navigation, and speech recognition requires significant amount of computing resources and a number of research projects, e.g., MAUI[5] and CloneCloud[4], have explored the idea of cloud offloading. MAUI provides dynamic energy-aware offloading of mobile code to the cloud. MAUI decides which methods should be remotely executed in the cloud based on expected energy consumption of local processing versus remote execution (including data transfer). CloneCloud dynamically offloads part of the Android Dalvik from the mobile device to the cloud and uses this cloned VM image as a powerful virtual device. The main purpose of CloneCloud is to speed-up execution time and decrease energy consumption on the mobile device.

What is needed for home automation?: As indicated above, many emerging home applications can easily overwhelm the resources available on a home element. For example, our preliminary experiments (see Section 4) show that an application that uses a web camera to monitor a home or its surroundings can consume 20-40% of CPU resources of a home element. Therefore, the motivations to offload computation to the cloud are similar to mobile applications and existing cloud offloading techniques used in mobile computing become appealing. However, our domain of home automation has many unique characteristics that differ from the mobile computing scenario. In this paper, we focus on these differences between mobile applications scenarios and our home automation use case. First, home automation applications are in a sense easier to offload since they are not as interactive as the mobile applications typically used to motivate code offloading. Home applications normally gather information periodically from sensors, actuators, and cameras and analyze and react to the data automatically, and thus, require *less human interaction* than gaming on mobile phone. Therefore, it is often possible to execute the whole application in the cloud versus the home element. This is also a major difference to code offloading in mobile computing since in the mobile computing use cases, the control of the application remains on the mobile device and the cloud is simply used as an extra computational resource. Second, many home automation applications are *safety critical* (e.g., smoke alarms, burglar alarms, automatic door locks) and must continue to provide their service (potentially in a degraded mode) even if the network connectivity to the cloud server is lost accidentally or intentionally (e.g., burglars cutting the network cable at the home or launching

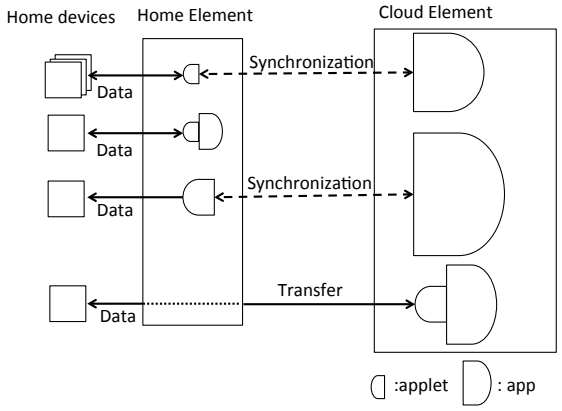

Figure 2: Application deployment in cloud-offloaded home controller for extensible home automation.

a DoS attack against the cloud server). Thus, moving the application completely to the cloud causes severe problems when the network connectivity is lost. Specifically, the application has to be able to resume execution at the home element in the case of network disconnection without losing its state, that is, it must not forget if the doors are unlocked, irrigation system is on, or if people are at the home at the time of the network disconnection. Therefore, simply restarting the applications on the home element may not be acceptable. Finally, a home automation system often runs *multiple applications for different home functions and multiple users* at the same time and therefore the runtime system has to determine which applications should run on the home element and which ones on the cloud element. The mobile code offloading solutions are not designed for multiple applications, potentially belonging to multiple users, running continuously without losing application state even in the case of network or cloud outages.

3. ARCHITECTURE

In this section, we provide an overview of our *Programmable Cloud-Enabled Home Controller* (PCEHC) architecture. Let us consider first the alternatives. If all home automation applications run on the home element, we can call the system a *Local Home Controller* system. This is the architecture used by most existing home automation systems. This solution is typically cost efficient when there are not many resource intensive home automation applications but runs into resource constraints when the set of applications and their complexity grows. An alternative would be to deploy the home automation applications in the cloud on a cloud element —we call this the *Cloud Home Controller* system—and all events and data from home devices are simply transferred to the cloud via the home element. In this case, the home element simply acts as a router and the processing load at the home element is low. However, a fully cloud-based home controller will be disabled when there is a network or cloud outage and may also not be cost efficient for all usage scenarios. Therefore, we propose a hybrid architecture, *the Programmable Cloud-Enabled Home Controller* where the home automation applications are executed both on the home and cloud elements.

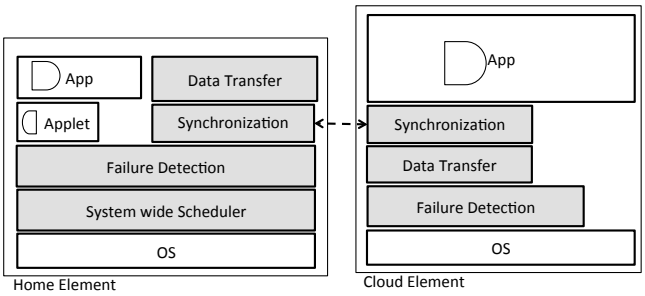

Figure 3: Functionality overview of cloud-offloaded home controller for extensible home automation

Application decomposition: Simply providing the choice of execution locations (home vs. cloud element) is not sufficient to solve the problems for our domain. Specifically, some applications are too large to run on the home element but they must operate on the home element to provide services when the network connectivity is down. Therefore, to satisfy all the combined requirements, we propose an application design paradigm where each home automation application is designed as a combination of two components, an *applet* and an *app*. The applet is a small, often gracefully degraded, version of the application and it is small enough to run on the home element. The app is the fully featured implementation of the application and needs to be able to run on the cloud element in case the home element does not have enough resources available for this application. For example, while a home security application may use video processing to recognize residents and automatically lock and unlock doors, the applet for the application may require users to authenticate using a key pad. The applet and app may be deployed together at the home element, together at the cloud element, or separately where the applet run on the home element and the app runs on the cloud element as illustrated in Figure 2. The applet and app may operate concurrently with each processing some subset of the events independently, they may operate in a pipeline where the applet provides preprocessing of data, or as alternatives, that is, the applet is only activated when the app is not available due to network outage.

A number of system capabilities are required to support this programming paradigm and to manage the resulting system. The main functions are illustrated in Figure 3. Specifically, we propose a home automation system with functions of System-Wide Scheduler, Synchronization, Failure Detection, and Data Transfer.

System-Wide Scheduler (SWS): The SWS allocates home automation applications over the distributed platform consisting of the home and cloud elements. For each home control application, it determines where its applet and app should be located given the currently available resources at the home element, available network bandwidth and latency to the cloud element, (expected) required resources by the new application, and requirements of the application. For example, any home control application that requires operation during network outage requires that the applet always run on the home element. Also, several home control applications may operate on the same event streams (e.g., video feed) and if the feed is already transmitted to the cloud

element, the deployment of an app that relies on the feed will not introduce any additional strain on the network link between the elements. The deployment of a new home control application may require the previously allocated applications to be reallocated. Finally, the introduction of new devices in the home may change the application allocation. We are working on formulating this multi-resource, multi-application scheduling problem and will develop efficient solutions.

Synchronization: For many home automation applications, it is important to provide synchronized state between the app and applet. For example, an app of an application may use video processing for face recognition and the application state then consists of the identities of the people currently at home. If the network connectivity is lost, the applet of this application must now take over running (a potentially degraded mode) of the application functionality. If the applet does not know the application state (i.e., who are the people at the home now), it may not be able to take the correct actions (e.g., locking or unlocking doors or activating motion sensors, etc). Therefore, we envision our system providing a synchronized shared state abstraction that allows an application state update from app to be propagated synchronously to the applet, or the other way around. This capability makes it possible for an applet to take over in the case of network outage and the app to resume control after network is restored. We are working on designing a convenient and efficient synchronized shared state abstraction for the home automation domain.

Failure Detection: For the fail-over capability, it is crucial that the home controller platform monitors for connectivity between the home and cloud elements. When the connectivity is lost, it notifies the applications that are registered for the failure event. The application may either activate an applet that has been waiting in an inactive state or notify a running applet that it needs to switch to an active control mode. Applications are similarly notified when a network connectivity is restored. The Failure Detection functionality ensures the application notifications at the home and cloud elements are synchronized with regard to event processing so that no application state inconsistency can occur due to both applet and app assuming they are in charge of processing and acting on a specific event.

Data Transfer: The Data Transfer functionality provides general data transfer capabilities such as buffering during connection failures and general purpose data compression facilities. Any application specific data compression or filtering would typically be implemented in an applet.

4. PRELIMINARY EXPERIMENTAL RESULTS

We have performed a number of preliminary experiments to demonstrate the feasibility and need for the cloud-offloaded home controller. The experiments were performed on the following hardware. As the home element, we used a Raspberry Pi with a 700MHz ARM core and 512MB of RAM running 3.6.11+ kernel Raspbian OS and a Z-wave daughter board connected to the Raspberry Pi via a serial port as shown in Figure. 4. Our cloud element is a PC with a 3.10GHz Intel dual core system and 4GB of RAM, running Ubuntu 12.04.

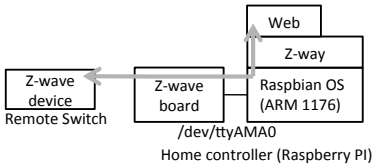

Figure 4: Local home controller.

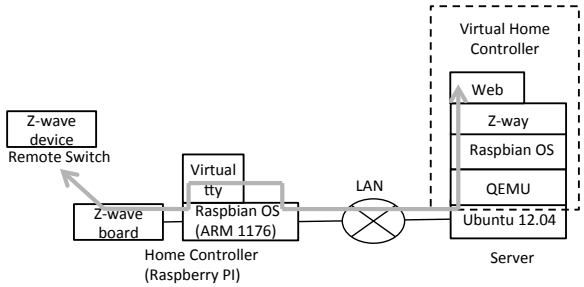

Figure 5: Cloud home controller.

First, we evaluated the feasibility of the Cloud Home Controller architecture. We tested offloading GPIO (General Purpose I/O) procedures using Socat[6] and USB/IP[11].USB /IP provides a virtual peripheral bus extension over IP Network to tunnel home device USB events from the home element to the cloud element. We used Socat to establish a bidirectional byte stream between the home and cloud elements and encapsulate all ioctl commands for the Z-Wave daughter board using these Socat streams. We evaluated the response time of remotely controlling a Z-Wave device from the cloud element in multiple emulated network settings shown in Table 1. We chose bandwidth and latency based on published measurements[1, 12, 14]. As shown in Table 1, the Z-Way application that manages Z-Wave devices and provides sensor data to any web server via Web API can operate on the cloud element in normal mobile networks and home networks.

Second, we tried to offload the capturing and viewing of live streaming video from a web camera connected to the home element. In this experiment, we used USB/IP to tunnel from the cloud element to the USB devices. The network in this experiment was a 100Mbps LAN. The experiment showed that the server can remotely control light weight devices such as a mouse and a keyboard, but can not capture the video stream on a server due to frame loss. The application for streaming video requires advanced timing control to capture whole video frames. In this case, we need a better transfer mechanism than USB/IP.

Finally, we used a Local Home Controller to measure the CPU Load of the Z-Way application and video viewing applications. The Z-way application consumed on the average about 15% of CPU resources and 2% of memory. The video viewing applications, like guvcview and freekinect, required on the average about 20% of CPU resources and 6% of memory for 320×240 resolution at 20 frame/s rate, and on the average about 40% of CPU resources (up to about 50%) and 7% of memory for 800×600 resolution at 20 frame/s rate. Also, the experiment showed a resource constrained device like our home element could not support two web cameras at the same time.

Table 1: Evaluation of network environments

Home controller Type	Network Type	Bandwidth or RTT	Evaluation Environment	Response time[ms]	
				turn on	turn off
local (Fig.4)	–	–	real	227	214
virtual (Fig.5)	LAN	100Mbps	real	460	447
	3G	80ms ± 30%	emulated	1,063	1,094
	4G/LTE	65ms ± 50%	emulated	1,004	1,040
	Cable	14.4/7.2Mbps	emulated	1,989	1,807

Implications: Even these very preliminary experiments show both the need for, and potential of, our cloud-offloaded home controller architecture. Our experiments showed clearly that it is easy to overwhelm a low-powered home element device such as our Raspberry Pi (which is comparable to some real home gateway processors) with multiple video processing applications (e.g., feed from 2 or 3 cameras). We also showed that the delay caused by offloading application logic to a cloud does not necessarily introduce unreasonable latency given typical network connectivity to homes including 3G or LTE networks. Even for applications such as automatic unlocking of doors, a one second delay is reasonable. However, we also determined that it is not possible simply to tunnel video to the cloud element. Rather, application decomposition into apps and applets and platform services such as data transfer are needed to enable applications to run on this hybrid platform.

5. CONCLUSION AND FUTURE WORK

In this paper, we make the case that cloud-offloading can be instrumental in enabling a new flexible ecosystem of rich applications in the rapidly growing home and building automation space. Although the problem of cloud-offload has received a great deal of attention in the literature, most of this work has been in the context of offloading computation from mobile phones and tablets. We have demonstrated through examples and preliminary experimental results that home automation presents the following characteristics not typically found in mobile offload settings: a) a more relaxed user interaction model that can tolerate large network latencies, b) a different timesharing model consisting of multiple (potentially compute intensive) applications that need to be continuously operational, and c) a critical need for disconnected operation. We proposed an architecture for a cloud-enabled home controller that addresses these differences through a combination of application-level decomposition, system-wide offload decisions, and shared objects between the home and the cloud that support state synchronization and reintegration. In future work, we intend to address the remaining challenges in realizing our vision, including new offload decision making algorithms and efficient support for state synchronization.

6. REFERENCES

[1] Y. Abe, R.Geambasu, K.Joshi, H. A.Lagar-Cavilla, and M.Satyanarayanan. vtube: efficient streaming of virtual appliances over last-mile networks. *In Proceedings of the 4th annual Symposium on Cloud Computing*, 2013.

[2] AT&T. *AT&T Digital Life*. https://my-digitallife.att.com/learn/.

[3] M. Chan, C. Hariton, P. Ringeard, and E. Campo. Smart house automation system for the elderly and the disabled. In *Systems, Man and Cybernetics, 1995. Intelligent Systems for the 21st Century., IEEE International Conference on*, volume 2, pages 1586–1589 vol.2, Oct 1995.

[4] B.-G. Chun, S. Ihm, P. Maniatis, M. Naik, and A. Patti. Clonecloud: Elastic execution between mobile device and cloud. *EuroSys*, 2011.

[5] E. Cuervo, A. Balasubramanian, A. Wolman, S. Saroiu, R. Chandra, and P. Bahl. Maui: making smartphones last longer with code offload. *In Proceedings of the 8th international conference on Mobile Systems, applications, and services*, pages 49–62, 2010.

[6] dest unreach.org. *socat - Multipurpose relay*. http://www.dest-unreach.org/socat/doc/README.

[7] C. Dixon, R. Mahajan, S.Agarwal, J. A.Brush, B.Lee, S.Saroiu, and P.Bahl. An operating system for the home. *In Proceedings of the 9th USENIX conference on Networked Systems Design and Implementation*, 2012.

[8] ETSI. *Machine to Machine Communications*. http://www.etsi.org/technologies-clusters/technologies/m2m.

[9] G. Evans. Solving home automation problems using artificial intelligence techniques. *IEEE Trans. on Consum. Electron.*, 37(3):395–400, Aug. 1991.

[10] J. Gerhart. *Home Automation and Wiring*. McGraw-Hill Professional, 1999.

[11] T. Hirofuchi, E. Kawai, K. Fujikawa, and H. Sunahara. Usb/ip – a peripheral bus extension for device sharing over ip network. *Proceedings of USENIX Annual Technical Conference*, FREENIX Track:47–60, 2005.

[12] J. Huang, F. Qian, A. Gerber, Z. M. Mao, S. Sen, and O. Spatscheck. A close examination of perfor- mance and power characteristics of 4g lte net- works. *In Proceedings of the 10th international conference on Mobile systems, applications, and services, MobiSys '12*, pages 225–238, 2012.

[13] oneM2M. *oneM2M Use Cases Collection*. http://www.onem2m.org/library/index.cfm, 2013.

[14] PCMag. *Fastest mobile networks 2013*. http://www.pcmag.com/article2/0,2817,2420333,00.asp, 2013.

[15] A. Ranganathan and R. H. Campbell. Supporting tasks in a programmable smart home. *IIn ICOST 2005 : 3rd International Conference On Smart Homes and Health Telematic – From Smart Homes to Smart Care*, 2005.