EdgeOS$_H$: A Home Operating System for Internet of Everything

Jie Cao, Lanyu Xu, Raef Abdallah, and Weisong Shi
Department of Computer Science
Wayne State University
Detroit, MI, USA
{jiecao, xu.lanyu, raef.abdallah, weisong}@wayne.edu

***Abstract*—The proliferation of Internet of Everything (IoE) and the success of rich Cloud services have pushed the horizon of a new computing paradigm, Edge Computing, which calls for processing the data at the edge of the network. Smart home as a typical IoE application is being widely adapted into people's lives. Edge Computing has the potential to empower the smart home, but it needs more contribution from the community before it truly benefits our lives. In this paper, we present the vision of EdgeOS$_H$, a home operating system for Internet of Everything. We further discuss functional challenges, namely programming interface, self-management, data management, security & privacy, and naming, as well as non-functional challenges, such as user experience, system cost, delay, and the lack of availability of open testbed. Within each challenge we also discuss the potential directions that are worth further investigation.**

Keywords : Edge Computing, Internet of Everything, Smart home, Operating System

I. Introduction

The era of smart home is coming. According to the report from Berg Insight [1], the number of smart homes in Europe and North America will reach 68 million in 2019. The market for the smart home in 2012 was $4.8 billion, and it will grow up to $35.3 billion by 2020, predicted by a report from Allied Market Research [2]. However, the current smart home system is far from people's expectations of being "smart". We think that a real smart home should be able to have self-awareness, self-management, and self-learning. We will explain these abilities in the following sections. Occupants should be part of the human-home interaction, rather than the administrators of all the connected devices.

Thanks to the flourishing of the Internet of Everything and low-cost, small but powerful sensors and chips are available to be deployed everywhere in the domestic environment [3], [4], [5], [6], [7], [8]. From a light to a thermostat to a camera, every aspect of a home could be sensed and controlled in a connected manner. The wireless network is also mature although the protocols are not unified yet. Ultra low power Wi-Fi, Bluetooth low energy, Zigbee, Z-wave, and NFC all look promising and strong to support domestic data transmission. In this case, we think the operating system might be the last missing piece of the smart home puzzle.

Edge Computing [9] could be a good solution for the smart home operating system by allowing computation to be performed at the edge of the network, on downstream data on behalf of Cloud services and upstream data on behalf of IoE devices. We observed that with more and more connected things becoming available at home, many people are doing DIY style smart home design and installation. The lack of a home operating system makes it very difficult to manage devices, data, and services. This is due to most of the systems nowadays work in a silo-based manner and can not be connected or communicate with other systems, as shown in Figure 1. To solve this problem using Edge Computing, we introduce the concept of EdgeOS$_H$[1], which is a home operating system for the Internet of Everything. With Edge Computing, the devices and services in the home could be connected to a central EdgeOS. This paradigm manages the devices and services more efficiently and easily.

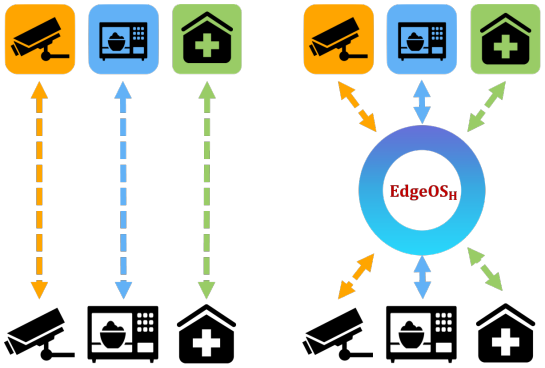

Fig. 1. A comparison of silo-based vs. EdgeOS-based smart home.

We organize the remainder of this article as follows. In Section II, we introduce the background and related work and give our understanding of smart home. During the revisit of the previous work, we found the operating system for smart home is currently missing. We introduce our solution as EdgeOS$_H$, and present the overview and structure of EdgeOS$_H$ in Section III. From Section IV to Section VIII we present the current functional challenges in smart home and discuss the potential solutions that EdgeOS$_H$ offers. In Section IX we discuss several non-functional issues for smart home systems. Finally, the paper concludes in Section X.

[1]EdgeOS stands for the operating system for Edge Computing. For different environments, EdgeOS could have multiple variations. For example, EdgeOS$_H$ for smart home, EdgeOS$_V$ for vehicle, EdgeOS$_B$ for smart building, and so on.

II. Related work

Smart home has drawn a lot of attention from the community since ubiquitous computing became popular. Researchers and practitioners spare tremendous effort in applying novel ideas as well as technologies in the domestic environment. In 2002 [10] the research team from MIT has developed a living laboratory using context-aware sensing to empower the home. In their understanding, a smart home will be a place where people can live a healthier and longer life via the help of digital and robotic agents. Resource consumption will be reduced in the smart home, and the home will be fully automated where occupants do not need to think about daily tasks at all. Kientz, *et al.* from Georgia Tech developed Aware Home [11] in 1998, which is viewed as the "glimpse of the most advanced domestic technologies for the potential future home."

With the proliferation of high-speed Internet and the Internet of Everything, more and more products for the smart home are also available on the market. A smart device such as iRobot, Philips Hue, and Nest learning thermostat, etc. shows that homeowners are ready to embrace smart devices in their daily lives. Amazon Echo, Samsung SmartThings, and Google Home provide a hub and user interface for occupants to interact with connected devices. HomeOS from Microsoft and HomeKit from Apple enable a framework for communicating with and controlling connected accessories in a smart home.

Our understanding of smart home is that it should be an *automated* and *energy efficient* domestic place where occupants could enjoy *a healthier meanwhile more comfortable life*. A smart home should come with *self-awareness, self-management, and self-learning* ability to satisfy and improve occupants' lifestyles.

Self-awareness means the home should be available to sense the occupants' status and domestic data automatically. How many people are in the home? Where are they? Are they sleeping? Status information of occupants like these should be derived from the domestic data such as room temperature, motion sensor, camera, etc. Self-awareness is a necessary capability for self-management and self-learning, and there is some ongoing research in this field [12], [13], [14], [15], [16], [17].

As a smart home is supposed to arrange everything automatically and free occupants' hands, self-management plays a crucial role in the whole system [18]. We will discuss self-management in details in Section V.

Self-learning refers to the ability to profile the occupant's personal behavior based on historical data to make personalized configuration of the home. Researchers and practitioners have contributed some work to provide this ability to the home environment [19], [20], [21], [22].

Despite the research and smart home products being readily available, some challenges always exist before we have a real smart home. In [23], Edwards and Grinter listed seven challenges that should be addressed for the smart home, including the accidentally domestic, impromptu interoperability, the lack of system administrator, adoption of domestic technologies, social implications, the reliability of the smart home, and ambiguity in ubiquitous sensing and computing. 16 years later, these challenges remain unsolved in today's home environment. When Mennicken, *et al.* examined the recent research with the observation of industry in [24], they provided three challenges in the current smart home such as meaningful technologies, complex domestic spaces, and human-home collaboration.

Although challenges and potential solutions have been examined in previous research, we think that there are still some issues that are worth raising about smart home. In the next section, we will introduce the overview and structure of EdgeOS_H, and our hope is to contribute to the consensus among various disciplines that make up smart home.

III. EdgeOS_H: overview and design

In a smart home, data is produced by various sensors and devices in the domestic place. Moreover, the data is also consumed at home to serve the occupants. With the rational *"Computing should happen at the proximity of data sources"* [9], we think that the idea of Edge Computing fits perfectly and should be deployed as the computing paradigm for the smart home. In order to apply Edge Computing to smart home, we propose EdgeOS_H, which is a smart home operating system for the Internet of Everything, as shown in Figure 2. EdgeOS_H is the bridge to connect the devices at home with the Cloud, home occupants, and developers. For the Cloud, EdgeOS_H can upstream/downstream data and computing requests on behalf of the devices. For home occupants, EdgeOS_H provides collaboration between humans and home. For service practitioners, EdgeOS_H is capable of reducing the complexity of development by offering an unified programming interface. For the smart home, EdgeOS_H is the brain that manages the data, devices, and services while guaranteeing the security and privacy of the data.

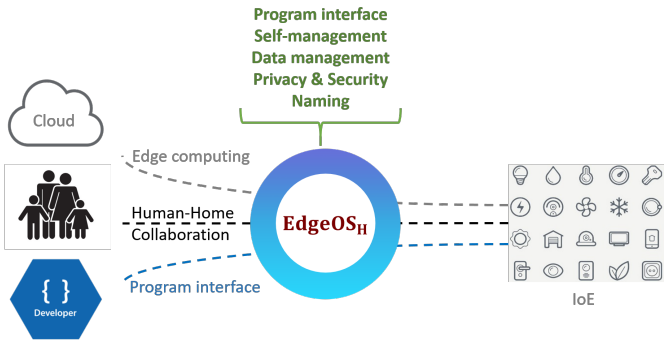

Fig. 2. The overview of EdgeOS_H.

Applying Edge Computing in the smart home will bring several benefits. First, network load could be reduced if the data is processed at home rather than uploaded to the Cloud. This is important for the domestic environment considering the bandwidth is usually limited. Second, service response time could be decreased since the computing takes place closer to

both data producer and consumer. Third, the data could be better protected from an outside attacker since most of the raw data will never go out of the home.

Compared to conventional computing platforms such as PCes, smart phones, and cloud, smart home has its specific characteristics.

First, for PCs or smart phones, the operating system can easily manage all the hardware resources since they are limited by the manufacturer fixed design. However, the home environment is very dynamic, which means the home operating system will face heterogeneous hardware provided by different manufacturers. The dynamic environment brings about new challenges in communication and management. Moreover, the current smart home applications usually work in a silo-based manner rather than attached to a specific operating system. Therefore, how EdgeOS$_H$ can manage various combinations of devices and services is still a huge challenge.

Second, traditional operating systems are usually resource-oriented. In the conventional computing platform, e.g., laptops and mobile devices, the most important responsibility of the operating system is resource management. However, the smart home is a data-oriented environment, which means services in a smart home should interact directly with the data collected by the devices, rather than the resources, or in another word, the specific devices.

A. Overview

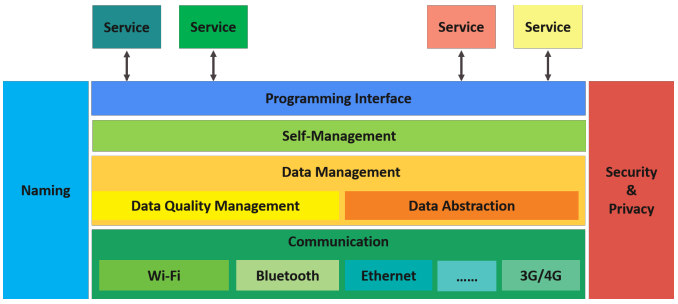

Fig. 3. The logical view of EdgeOS$_H$.

In Figure 3, we present the logical view of EdgeOS$_H$, which consists of four vertical layers: *Communication, Data Management, Self-Management,* and *Programming Interface*, as well as two extra components across all four layers, *i.e., Naming* and *Security & Privacy*. In the *Communication* layer, EdgeOS$_H$ needs to collect data from mobile devices and all kinds of things through multiple communication methods such as Wi-Fi, Bluetooth, ZigBee or a cellular network. Data from different sources needs to be fused and massaged in the *Data Management* layer. In this layer, the data abstraction model will fuse and massage the data into one database, and the data quality model will manage the quality of the data. On top of the *Data Management* layer is the *Self-Management* layer. Services such as device registration, maintenance, and replacement will be provided here. Moreover, *Self-Management* layer should also be able to detect the conflict among services and optimize the service quality. A unified programming interface should be supported to provide satisfactory performance for user applications with minimum development effort, which is the *Programming Interface* layer. The *Naming* mechanism is required for all layers with different requirements. At last, data security and privacy should be protected in *Security & Privacy*.

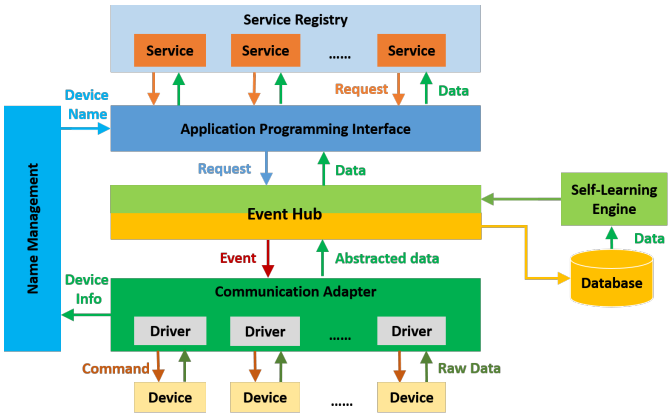

Fig. 4. The design of EdgeOS$_H$.

B. Design

The design of EdgeOS$_H$ is shown in Figure 4, including seven components: *Communication Adapter*, *Event Hub*, *Database*, *Self-Learning Engine*, *Application Programming Interface*, *Service Registry*, as well as *Name Management*, which stretches across other components. To integrate the device into EdgeOS$_H$, *Communication Adapter* gets access to devices by the embedded drivers. These drivers are responsible for sending commands to devices and collecting state data (raw data) from them. Sitting between devices and the *Event Hub*, *Communication Adapter* maps to the Communication layer in the logical view. It packages different communication methods that come from various kind of devices, while providing a uniform interface for upper layers' invocation. In this way, developers and users do not need to deal with multiple kinds of communication methods when manipulating the system. Moreover, it only provides abstracted data to upper layer components, reducing privacy risk to some extent. As the core of the architecture, the *Event Hub* maps to two layers in the logical view: the *Data Management* and *Self-Management* layers. The *Event Hub* is responsible for capturing system events and sending instructions to lower levels. Those instructions are smart commands based on machine learning developed through communication with the Self-Learning Engine. It collects requests from services and sends them to the *Communication Adapter*, and in turn, collects abstracted data from the *Communication Adapter* and sends them to upper layers. The *Database* is another component in the *Data Management* layer. As a data-oriented system, EdgeOS$_H$ generates large amount of data every day, which contains valuable information related to user preferences and settings. The *Event Hub* stores data in the *Database*. The data

stored in the *Database* is utilized by the *Self-Learning Engine* that belongs to the *Self-Management* layer. The *Self-Learning Engine* creates a learning model. This learning model called the Self-Learning Model acts as an input to the *Event Hub* to provide decision-making capability. To provide better user experience, the *Self-Learning Engine* is developed to analyze user behavior, generate the personal model for the user, and help improve the system. *Application Programming Interface* (API) and *Service Registry* are located in the upper layers of the system, and are utilized for third-party services. Developers are encouraged to use EdgeOS_H APIs to communicate with the *Event Hub*, and register their services with the system. Required by all layers, *Name Management* helps the system keep devices organized. When a new device is registered with the system, *Name Management* allocates a name for it using the following rule: location (where), role (who), and data description (what). This rule is complied by all layers.

In this section, we introduced Edge Computing to the domestic environment and addressed some of the challenges, and we briefly presented the logical view of EdgeOS_H and explained how it maps to its design. The successful operation of EdgeOS_H in a smart home relies heavily on the successful implementation of the functional and non-functional requirements. Since EdgeOS_H consists of several layers, We will briefly describe some of those requirements for EdgeOS_H as related to a smart home environment. In the following sections, we will reflect on solving some of the challenges presented in previous work as well as discussing new challenges we have found by deploying Edge Computing in a smart home. We will discuss the functional challenges in the different layers of EdgeOS_H, including programming interface, self-management, data management, security & privacy, and naming.

IV. Programming Interface

In most of the current smart home systems, devices work on a silo-based policy. Take as an example, a light with a motion sensor. When a motion is detected, the light turns on. This new light status information is not shared with other devices on the network. In this section, we will discuss the challenge for creating a programming interface in EdgeOS_H.

In order to achieve self-management, all the devices and sensors in the domestic environment must be connected and controlled by the EdgeOS_H mechanisms. However, the various existing programming interfaces require extensive effort from users and developers to add, replace, and configure smart devices.

Without a flexible interface, developers would have to put a lot of effort into implementing applications for the each different device in a domestic environment.

EdgeOS_H Programming Interface is a flexible interface, through which devices and services can communicate with each other. This interface will make the task of a developer much easier since it reduces multiple interfaces into one.

In this interface, EdgeOS_H will manage the collection of data form all devices. The collected data will be stored in

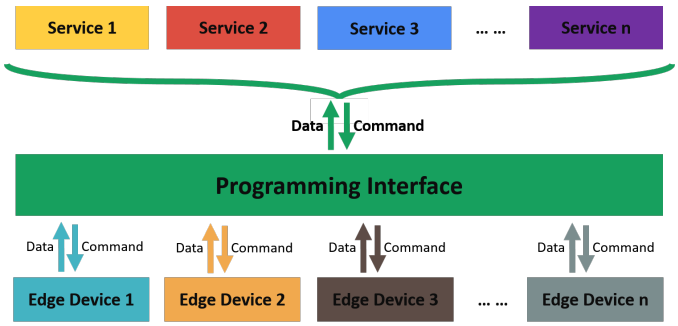

Fig. 5. The programming interface in EdgeOS_H.

tables (Figure 5). A user can then utilize the unified interface to get data and send commands from EdgeOS_H.

Although the idea of this programming interface is straightforward, its implementation in a smart home can be very challenging. Smart device manufacturers might not provide API's for their products alleging security issues. Although open source API's are abundant, the infrequent software updates and the limited documentation make it very difficult to manage API's.

To solve this issue, API management tools such as Apigee [25] could be considered as a potential solution. Online service such as IFTTT [26] could also manage devices if the API driver is available. Open source systems such as Home Assistant [27] and openHAB [28] can be utilized as a learning tool for API management. We do not believe there is a magic solution to this API issue but it is a good opportunity to propose new standards for home device API's.

In this section, we presented the Programming Interface challenges and introduced the high-level design of a potential solution.

V. Self-Management

Service quality is one of the key issues that should be addressed in EdgeOS_H. Four fundamental features (DEIR) of service quality are argued in [9] for Edge Computing, and we think they should be inherited in the smart home system.

- *Differentiation:* In the domestic place, there could be multiple services deployed. Moreover, services will have different priorities from the occupants' point of view. In a smart home, service should be prioritized. A service with a higher priority could interrupt other service and be executed first. For example, when the user wants to watch a movie online, can another device such as a security camera stop the data uploading/downloading to save Internet bandwidth?
- *Extensibility:* In the self-management section, we mentioned the idea of extensibility. When a system is evaluated, researchers and practitioners want to ask the following questions: Can the new device and service be installed in the system easily? If a device wears out, can it be replaced and can the previous service adopt the replacement easily?

- *Isolation:* Isolation should be evaluated in two dimensions. In the vertical dimension, practitioners should test if the service could be isolated from the device. For example, if one service crashed, can it free the device it is using so that other service can still access that device? In the horizontal dimension, can one service be isolated from other services so that the private data is not accessible by other services?
- *Reliability:* Reliability is a fundamental requirement for every computing system, and a smart home is also not an exception. From the system point of view, can the conflict between various services be detected? From the service point of view, can the service maintain a reliable connection to the device? How much reliable the device is, and can the device nofy the system n a battery needs to be replaced?

In order to support the DEIR service quality requirements, we think the self-management layer should take five parts into consideration: device registration, device maintenance, device replacement, conflict mediation, and self-involving optimization. It highly involves not only the system itself but also the connected devices.

A. Device registration

When a new device is added to the home, it calls EdgeOS_H for registration. In the registration part, EdgeOS_H searches available services for the added device. These available services could be adopted by the same kind of devices or manually arranged by home occupants. If the same kind of device was never implemented before, EdgeOS_H will check configuration file for predefined services. This part is open to occupant through a programming interface, allowing the occupant to configure newly added device according to his/her favor. The occupant can also determine which service should be applied to this device, and which service can be removed from it. Of course, the occupant can let EdgeOS_H decide everything according to the existing profile automatically, and only receive the notification of registration status during the process. A plain example is the light. When the occupant installing a light, EdgeOS_H first pops out notification of several available services for occupant to choose from, or if the occupant is not interested in intervention, EdgeOS_H can configure the light automatically according to home's profile (brighter or darker).

B. Device maintenance

To guarantee that the system works properly, it is important for devices to maintain a healthy status. Given the complexity and structural specificity of EdgeOS_H, it is not feasible for occupants to take care of all the devices by themselves. In this case, EdgeOS_H is responsible for all the devices, including monitoring their health status and sending warning notification to occupants promptly.

Device maintenance should have two phases: survival check and status check. In the survival check phase, devices are required to send heartbeats to EdgeOS_H in a fixed frequency. The heartbeats notify EdgeOS_H that device is still alive. If no heartbeat is received from a certain device, EdgeOS_H will report the dead device and ask for a replacement. Status check is designed for those live devices, and in this phase EdgeOS_H focus on their life status. If a device is in *bad* status, it sends heartbeats periodically, while not being able to perform its task normally. For instance, a smart light keeps sending heartbeat but doesn't light, or a security camera keeps recording extremely blurred video or image. When the user is far away from home and remotely open the stove to heat a slow cook, he/she can check whether the stove is working through the indoor camera.

C. Device replacement

When a device died or fails, device replacement is aroused. EdgeOS_H reports the failure to the occupant and reminds the occupant to do the replacement. EdgeOS_H will suspend all the services adopted by the malfunctioning device to avoid any disorder or damage to other devices. After the replacement device is installed, original configuration and services are restored. Device replacement is challenging due to the complex nature of network structure, communication protocols, and services. To replace a device, a new network address is assigned, and associated services should resume to restore the settings of the old device. For example, when a smart surveillance camera malfunctions, EdgeOS_H will halt the services running associated with the device. EdgeOS_H will send a replacement request notification to the occupant. During the new device replacement process, EdgeOS_H will notify the user that a new device has been detected and provides recommendations for the next steps. If the user accepts the recommendation, EdgeOS_H will be able to add the replacement device without the user having to manually configure the device. Referring back to our camera example, EdgeOS_H will associate the new camera IP address with every service that was running before the malfunctioning occurred.

D. Conflict mediation

In any network, service conflicts are likely to occur and a domestic environment is no different. Services are supposed to be isolated from each other. However, this service isolation does not always apply; thus, causing conflicts to arise.

An example is the binding services running on a smart light bulb. Suppose there are two services, one is *"turn on the light at sunset"*, and another is *"keep the light turned off until the user comes back home"*. What will happen if the user comes back before sunset? This is a scenario where service conflicts are possible.

with a single device that has several services running, EdgeOS_H can provide the user with the option to set priority for each service. In the case where conflicts arise, the higher priority service takes precedence.

E. Self-Learning

To provide the user with a good smart home experience, the Self-Management layer is an integral part of EdgeOS_H.

Initially, the proposed operating system ($EdgeOS_H$) will utilize the first few smart devices to learn more about the user. The more devices added to the smart home network, the more the operating system learns about the user. The Self-Management layer will assist in creating a user profile that ($EdgeOS_H$) will utilize to establish new services associated with new devices.

By introducing self-management layer to $EdgeOS_H$, the user does not need to reconfigure devices and related services. $EdgeOS_H$ will be able to make the addition or replacement process fast and smooth.

VI. Data Management

In this section, we will examine the challenge in data management for $EdgeOS_H$ from two aspects: data quality and data abstraction.

A. Data Quality

Data quality is an integral part of $EdgeOS_H$ Data Management Layer. Data driven applications provide information to users in order to make decisions. It is important that those decisions are based on data that is accurate, complete, and in or close to real-time. Data accuracy, completeness, and delay play a critical role, particularly in a smart home environment. In order to detect sensing error, we think in $EdgeOS_H$ data quality could be evaluated by two aspects: history pattern and reference data.

In a smart home, data could easily fall into a certain pattern due to the periodical user behavior. To provide better service to home applications and devices, current data mining and machine learning algorithm [29] could be leveraged to train a model for data quality detection model in $EdgeOS_H$, as shown in Figure 6. This model could automatically detect abnormal data pattern from the historical data record, and further analyze the reason for the abnormal pattern, which could be user behavior changing, device failure, communication interfacing, or attack from outside.

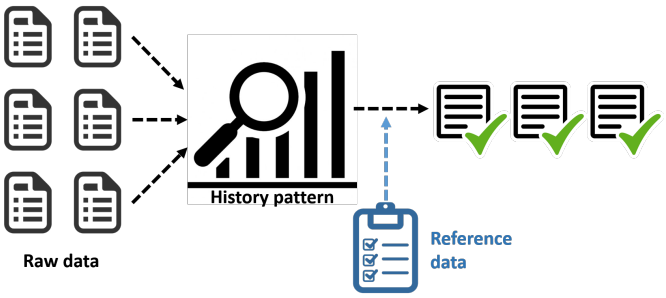

Fig. 6. The data quality management model of $EdgeOS_H$.

B. Data Abstraction

As we have discussed in Section V, services running on $EdgeOS_H$ should be isolated from the devices, which means raw data should be abstracted from the home devices, and only abstracted data should be presented to the services. There will be several challenges in the abstraction.

First, data reported from different things comes with various formats. For the concern of privacy and security, applications running on $EdgeOS_H$ should be blinded from raw data. Moreover, they should extract the knowledge they are interested in from an integrated data table. We can easily define the table with id, time, name, data (*e.g.*,{0000, 12:34:56PM 01/01/2016, kitchen.oven2.temperature3, 78}) such that data from various edge devices can be fitted in. However, the details of sensed data have been hidden, which may affect the data usability.

Second, it is sometimes difficult to decide the degree of data abstraction. If too much raw data is filtered out, some applications or services could not learn enough knowledge. However, if we want to keep a large quantity of raw data, there would be a challenge for data storage.

Last, data reported by home devices could be unreliable sometime due to the low precision sensor, hazard environment, and unreliable wireless connection. In this case, how to abstract useful information from the unreliable data source is still a challenge for smart home application and system developers.

In this section, we discussed the data management layer for $EdgeOS_H$ with two components: data quality management and data abstraction. We will examine the challenge for security and privacy in the next section.

VII. Security & Privacy

Most things in the smart home are resource constrained for the consideration of limited battery life and data storage size, and things are connected to $EdgeOS_H$ through diverse wireless connection methods. These two features make it very difficult to protect a smart home from attackers. In this section, we will discuss the challenges for security and privacy in a smart home.

Although the smart home system has experienced rapid development in recent decades, and early complicated device deployment systems had been improved into recently more friendly ones, security and privacy issues have not been taken great care of [30], [31], [32], [33]. Several researchers and news reports illustrate that smart home system is vulnerable and easy to disclose users' privacy when facing attacks. For example, Forbes.com reported that hackers broke into a baby monitor and yelled at the baby [34]. Oluwafemi *et al.* induced seizures in epileptic users by causing compact florescent lights to rapidly power cycle [35]. Fernandes *et al.* found that Samsung's SmartThings framework has several design flaws that make door lock and alarm system exposed to attackers [36]. CNN Money also reported that smart home devices were exposed security flaws to hackers [37]. A smart home is more difficult to protect when compared to a traditional smart phone or personal computer because many device producers may have little expertise in security, unintentionally using the operating system and protocols only half patched or even maliciously tampered. Normally, if a malicious program infects a laptop, the user can run a malware detector and reboot the machine. But if the same thing happens in a smart home where various kinds of devices dividing tasks and cooperating

with each other, it is impossible to just simply turn everything off and locate the problem among devices. Consequences of insecure smart home are much more severe and pose a heavier threat to users than PC or smart phone. Victims could suffer from property or psychological damages, and even life threatening occurrences. Since neither users nor device manufacturers can guarantee the safety, the operating system in the smart home is duty-bound to take care of potential security and privacy issues for users.

Like Cloud computing raises community's attention on privacy, smart home in edge computing also has to face the same problem. Deployed with IoE, a smart home can generate a lot of privacy information along with sensed usage data. For a smart home, there are three main problems:

a) The incomplete community awareness of privacy protection: According to a community research, among the 439 million households using wireless connections, 49% of Wi-Fi networks are unsecured, and 80% of households even still use default passwords on their routers [38]. It is urgent for users to realize that their privacy would be harmed if they keep ignoring the security of the smart home.

b) The ownership of collected home data: For smartphone applications, it is regular to store and analysis user data at service provider side. However, considering the privacy of the data generated at home environment, it will be better to keep this data at home and let the user have full authority to it. Furthermore, the user should have the privilege to 1) decide what kind of data could be provided to service providers, and 2) remove highly private data before they are uploaded for further process.

c) The missing tool to protect user privacy at the smart home: In EdgeOS_H, some of the privacy information could be erased temporarily before processing. For example, IP camera can record video and report abnormal snapshots regularly, masking all the faces in the video. However, not all devices in a smart home have such powerful computational capabilities. Some of the devices and sensors are highly constrained by resource, therefore the privacy-preserving algorithms can only run on EdgeOS_H.

In this section, we discussed the issues of data security and privacy in EdgeOS_H. We will examine the challenge for naming in the next section.

VIII. NAMING

In this section, we will discuss the naming issue in the smart home. The more the devices are in the domestic place, the more naming becomes a critical feature of the system [39], [40]. In order to communicate and locate a device on the network, a service need to know the specific network address and communication protocol. However, a standardized mechanism that can name the device efficiently is still missing.

EdgeOS_H can assign each device a human friendly name that describes the following information: location (where), role (who), and data description (what). For example, kitchen.oven2.temperature3. Then EdgeOS_H will assign identifier and network address to this thing. The human friendly name is unique for each thing and it will be used for service management, device diagnosis, and replacement. This naming mechanism makes the management of devices and data much easier. For example, the user will receive a message from EdgeOS_H such as Bulb 3 (what) of the celling light (who) in living room (where) failed, and then the user can replace the failed bulb without having to reconfigure the new bulb manually. Also, this naming mechanism supports better programming interface to service providers while protecting the the privacy of hardware specifications. This in turn will better protect data privacy and security. In this manner, a network address (IP address or MAC address) will be used to support various communication protocols such as BlueTooth, ZigBee or WiFi while mapping network addresses to human friendly names in EdgeOS_H. The human friently names will then be used to manage the services, data, and devices in EdgeOS_H.

In this section, we examined the challenges of naming in EdgeOS_H., and we think an efficient naming mechanism is still an open problem and worth further investigation by the community. We will discuss several issues in the next section that were not covered in this section.

IX. OPEN ISSUES

In addition to the technical challenges, we think there are several open non-functional issues such as open testbed, user experience, system cost and delay. Here we list them and hope practitioners could keep them in mind when designing and developing smart home systems.

A. Open testbed

Measuring the performance of a smart home system or application is a really challenging problem [41], [42]. Unlike other traditional computing systems, it is difficult to describe the performance of a home system using quantifiable metrics such as accuracy, latency, or power consumption. Also, there is not an open testbed specifically designed to evaluate smart home performance [43], as called in the NSF report [44]. In this paper, we call for the development of a few open testbeds for smart home environments that can be shared with the research community.

B. User experience

As a place where human activities are heavily involved, a smart home should consider user experience as an important evaluation metric. A lot of stories or even jokes have been posted about how the failure of smart home systems exacerbated the living conditions of the occupants [45], [46]. In [47] Woo *et al.* analyzed the user experience in the DIY style smart home. Based on previous work and our observation, we think the simplicity of installation, configuration, and user interface are the most important part of a good user experience. When the user wants to turn on the light, he/she should be able to do that with minimal effort (just one operation or one command), rather than "unlock the phone $\rightarrow$ find the app $\rightarrow$ locate the light $\rightarrow$ turn on." Also, when the user wants to turn on the light, the light should turn on without noticeable delay.

Maintenance is another user experience requirement that affects the health of the system. Maintenance of smart devices can prevent smart device service interruptions and fix potential problems. A device failure will lead to data loss [48]. Therefore, it is critical that there is a simple and straightforward procedure that the user can follow to maintain and backup smart home devices. This will help improve the overall performance of the smart devices, increase EdgeOS_H efficiency, and consequently make smart homes more reliable.

Another aspect of user experience requirement that an average smart home user might consider before starting the smart home project is portability. People often move form one place to another, and therefore they would also like to move the smart home functionality wherever the new destination is. Currently, some home security alarm systems do not require professional installation. Instead, the preconfigured system is shipped ready for the user to activate. In the same manner, our implementation of EdgeOS_H should allow the smart home system to be portable. In order to provide the user with a good experience, he or she should not need to reconfigure the system. Instead, the system should be able to function at the new location with minimal effort.

C. Cost

Building a smart home requires hardware and software that the average homeowner may find expensive. The more equipped a smart home is, the more cost the homeowner will incur. By the same token, the more equipped a smart home is, the more data it collects. The more data is collected, the faster and better EdgeOS_H will perform self-learning and self-management. Smart devices vary in price. Usually, a more expensive device provides faster data transmission rate than less expensive one. Private budgets are usually limited. This limitation might drive the successful implementation of a smart home. This is another example of a non-functional requirement that has a major effect on the overall system. Two aspects of the cost should be taken into consideration when we evaluate a smart home system.

First is the financial cost of system installation in the domestic place. According to a report by HomeAdvisor [49], the average cost to install a home automation system is $1,268. The cost is even higher if the homeowner hires professionals to install a hard-wired system. Therefore, it is important to ensure that the total cost of smart home system installation is within an affordable range.

Second is the resource consumption of the system. One reason of having a smart home is to make a domestic environment more energy efficient. Therefore, it is necessary to evaluate how much utility resource such as water, electricity, gas, and Internet bandwidth could be saved by the smart home.

D. Delay

EdgeOS_H has to make decisions and take actions based on data collected. Utilizing machine learning algorithms, the Data Management Layer will act as a checkpoint for ensuring the correctness of the captured data from the different types of sensors. The operating system will be able to learn and detect sudden data abnormalities. This layer will also be able to sense gaps in the data stream and report such occurrences. In some cases, this gap might be due to delay in data transmission, which leads us to tackle the last data quality characteristic: delay. Particularly in a smart home environment, if data is not processed in real or close to real-time, the operating system will make decisions that have diverse negative effects on the whole system. For example, capturing the correct outside temperature in real-time will ensure that the right inside home temperature is set at the right time.

X. Conclusions

In this paper, we presented our vision of a home operating system and introduced EdgeOS_H to the domestic environment. We also listed several functional as well as non-functional challenges that should be addressed before our vision could be fulfilled.

We discussed the different components of EdgeOS_H and introduced how EdgeOS_H should be designed to have a potent programming interface and a self-management capability. We also discussed how the data should be abstracted, stored, and evaluated in EdgeOS_H. Security and privacy protection was also examined in this paper as a home might be the most private environment for human beings. At last, we raised the lack of naming mechanism and discussed open issues such as testbed for evaluating the performance of a smart home. We also discussed the user experience, and the cost associated with a smart home.

We hope that EdgeOS_H can be used as a guidance for prototyping on smart home systems. We also hope that this paper can provide helpful information to researchers and practitioners from various disciplines when designing new technologies for smart homes.

References

[1] "Smart homes and home automation," [Online; accessed 02-22-2017]. [Online]. Available: http://www.berginsight.com/reportpdf/productsheet/bi-sh1-ps.pdf

[2] "Smart homes and buildings market by 2020," [Online; accessed 02-22-2017]. [Online]. Available: https://www.alliedmarketresearch.com/press-release/smart-homes-and-buildings-market-to-reach-35-3-billion-by-2020.html

[3] N. Jiang, C. Schmidt, V. Matossian, and M. Parashar, "Enabling applications in sensor-based pervasive environments," in *Proceedings of the 1st Workshop on Broadband Advanced Sensor Networks (BaseNets 2004)*, 2004, p. 48.

[4] M. Gowda, A. Dhekne, S. Shen, R. R. Choudhury, X. Yang, Y. Lei, S. Golwalkar, and A. Essanian, "Bringing iot to sports analytics." NSDI, 2017.

[5] R. Chandra, "Farmbeats: Iot for agriculture," 2017.

[6] H. S. Ferdous, B. Ploderer, H. Davis, F. Vetere, K. O'Hara, G. Farr-Wharton, and R. Comber, "Tabletalk: integrating personal devices and content for commensal experiences at the family dinner table," in *Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing*. ACM, 2016, pp. 132–143.

[7] B. Fang, Q. Xu, T. Park, and M. Zhang, "Airsense: an intelligent home-based sensing system for indoor air quality analytics," in *Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing*. ACM, 2016, pp. 109–119.

[8] Q. Kong, T. Maekawa, T. Miyanishi, and T. Suyama, "Selecting home appliances with smart glass based on contextual information," in *Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing*. ACM, 2016, pp. 97–108.

[9] W. Shi, J. Cao, Q. Zhang, Y. Li, and L. Xu, "Edge computing: Vision and challenges," *IEEE Internet of Things Journal*, vol. 3, no. 5, pp. 637–646, 2016.

[10] S. S. Intille, "Designing a home of the future," *IEEE pervasive computing*, vol. 1, no. 2, pp. 76–82, 2002.

[11] J. A. Kientz, S. N. Patel, B. Jones, E. Price, E. D. Mynatt, and G. D. Abowd, "The georgia tech aware home," in *CHI'08 extended abstracts on Human factors in computing systems*. ACM, 2008, pp. 3675–3680.

[12] E. Soltanaghaei and K. Whitehouse, "Walksense: Classifying home occupancy states using walkway sensing," in *Proceedings of the 3rd ACM International Conference on Systems for Energy-Efficient Built Environments*. ACM, 2016, pp. 167–176.

[13] Y. Agarwal, B. Balaji, R. Gupta, J. Lyles, M. Wei, and T. Weng, "Occupancy-driven energy management for smart building automation," in *Proceedings of the 2nd ACM Workshop on Embedded Sensing Systems for Energy-Efficiency in Building*. ACM, 2010, pp. 1–6.

[14] D. Austin, Z. T. Beattie, T. Riley, A. M. Adami, C. C. Hagen, and T. L. Hayes, "Unobtrusive classification of sleep and wakefulness using load cells under the bed," in *Engineering in Medicine and Biology Society (EMBC), 2012 Annual International Conference of the IEEE*. IEEE, 2012, pp. 5254–5257.

[15] G. Gao and K. Whitehouse, "The self-programming thermostat: optimizing setback schedules based on home occupancy patterns," in *Proceedings of the First ACM Workshop on Embedded Sensing Systems for Energy-Efficiency in Buildings*. ACM, 2009, pp. 67–72.

[16] L. Shu, Y. Zhang, Z. Yu, L. T. Yang, M. Hauswirth, and N. Xiong, "Context-aware cross-layer optimized video streaming in wireless multimedia sensor networks," *The Journal of Supercomputing*, vol. 54, no. 1, pp. 94–121, 2010.

[17] G. Zhang and M. Parashar, "Context-aware dynamic access control for pervasive applications," in *Proceedings of the Communication Networks and Distributed Systems Modeling and Simulation Conference*, 2004, pp. 21–30.

[18] P. Liu, D. Willis, and S. Banerjee, "Paradrop: Enabling lightweight multi-tenancy at the networks extreme edge," in *Edge Computing (SEC), IEEE/ACM Symposium on*. IEEE, 2016, pp. 1–13.

[19] H. Zheng, H. Wang, and N. Black, "Human activity detection in smart home environment with self-adaptive neural networks," in *Networking, Sensing and Control, 2008. ICNSC 2008. IEEE International Conference on*. IEEE, 2008, pp. 1505–1510.

[20] J. Byun, B. Jeon, J. Noh, Y. Kim, and S. Park, "An intelligent self-adjusting sensor for smart home services based on zigbee communications," *IEEE Transactions on Consumer Electronics*, vol. 58, no. 3, 2012.

[21] P. Rashidi and D. J. Cook, "Keeping the resident in the loop: Adapting the smart home to the user," *IEEE Transactions on systems, man, and cybernetics-part A: systems and humans*, vol. 39, no. 5, pp. 949–959, 2009.

[22] C. Reinisch, M. J. Kofler, and W. Kastner, "Thinkhome: A smart home as digital ecosystem," in *Digital Ecosystems and Technologies (DEST), 2010 4th IEEE International Conference on*. IEEE, 2010, pp. 256–261.

[23] W. K. Edwards and R. E. Grinter, "At home with ubiquitous computing: Seven challenges," in *International Conference on Ubiquitous Computing*. Springer, 2001, pp. 256–272.

[24] S. Mennicken, J. Vermeulen, and E. M. Huang, "From today's augmented houses to tomorrow's smart homes: new directions for home automation research," in *Proceedings of the 2014 ACM International Joint Conference on Pervasive and Ubiquitous Computing*. ACM, 2014, pp. 105–115.

[25] "apigee," https://apigee.com/, 2017, [Online; accessed 02-22-2017].

[26] "Ifttt," https://ifttt.com/, 2017, [Online; accessed 02-22-2017].

[27] "Home assistant," https://home-assistant.io/, 2017, [Online; accessed 02-22-2017].

[28] "openhab," http://www.openhab.org/, 2017, [Online; accessed 02-22-2017].

[29] Q. Zhang, M. Qiao, R. R. Routray, and W. Shi, "H 2 o: A hybrid and hierarchical outlier detection method for large scale data protection," in *Big Data (Big Data), 2016 IEEE International Conference on*. IEEE, 2016, pp. 1120–1129.

[30] J. Singh, T. Pasquier, J. Bacon, H. Ko, and D. Eyers, "Twenty security considerations for cloud-supported internet of things," *IEEE Internet of Things Journal*, vol. 3, no. 3, pp. 269–284, 2016.

[31] F. Wang, C. Yun, S. Goldwasser, V. Vaikuntanathan, and M. Zaharia, "Splinter: Practical private queries on public data," 2017.

[32] N. Saleheen, S. Chakraborty, N. Ali, M. M. Rahman, S. M. Hossain, R. Bari, E. Buder, M. Srivastava, and S. Kumar, "msieve: differential behavioral privacy in time series of mobile sensor data," in *Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing*. ACM, 2016, pp. 706–717.

[33] P. Huang, T. Xu, X. Jin, and Y. Zhou, "Defdroid: Towards a more defensive mobile os against disruptive app behavior," in *Proceedings of the 14th Annual International Conference on Mobile Systems, Applications, and Services*. ACM, 2016, pp. 221–234.

[34] Baby monitor hacker still terrorizing babies and their parents. [Online]. Available: http://www.forbes.com/sites/kashmirhill/2014/04/29/baby-monitor-hacker-still- terrorizing-babies-and-their-parents/5b91ff7717e2

[35] T. Oluwafemi, T. Kohno, S. Gupta, and S. Patel, "Experimental security analyses of non-networked compact fluorescent lamps: A case study of home automation security," in *Proceedings of the LASER 2013 (LASER 2013)*, 2013, pp. 13–24.

[36] E. Fernandes, J. Jung, and A. Prakash, "Security analysis of emerging smart home applications," 2016.

[37] "Your hackable house." [Online]. Available: http://money.cnn.com/interactive/technology/hackable- house/

[38] "Wi-fi networks security." [Online]. Available: http://graphs.net/wifi-stats.html

[39] W. Shang, A. Bannis, T. Liang, Z. Wang, Y. Yu, A. Afanasyev, J. Thompson, J. Burke, B. Zhang, and L. Zhang, "Named data networking of things," in *Internet-of-Things Design and Implementation (IoTDI), 2016 IEEE First International Conference on*. IEEE, 2016, pp. 117–128.

[40] T. Wolf and A. Nagurney, "A layered protocol architecture for scalable innovation and identification of network economic synergies in the internet of things," in *Internet-of-Things Design and Implementation (IoTDI), 2016 IEEE First International Conference on*. IEEE, 2016, pp. 141–151.

[41] L. T. T. Phuong, N. T. Hieu, J. Wang, S. Lee, and Y.-K. Lee, "Energy efficiency based on quality of data for cyber physical systems," in *Internet of Things (iThings/CPSCom), 2011 International Conference on and 4th International Conference on Cyber, Physical and Social Computing*. IEEE, 2011, pp. 232–241.

[42] P. Patel, T. Luo, and U. Bellur, "Evaluating a development framework for engineering internet of things applications," *arXiv preprint arXiv:1606.02119*, 2016.

[43] K. Jayarajah, R. K. Balan, M. Radhakrishnan, A. Misra, and Y. Lee, "Livelabs: Building in-situ mobile sensing & behavioural experimentation testbeds," in *Proceedings of the 14th Annual International Conference on Mobile Systems, Applications, and Services*. ACM, 2016, pp. 1–15.

[44] M. Chiang and W. Shi, "Nsf workshop report on grand challenges in edge computing," Tech. Rep., 2016.

[45] H. Tsukayama, "Why smart homes are still so dumb," June 2016, [Online; posted June-2016]. [Online]. Available: https://www.washingtonpost.com/news/the-switch/wp/2016/06/06/why-smart-homes-are-still-so-dumb/

[46] S. Higginbotham, "5 reasons why the smart home is still stupid," August 2015, [Online; posted 19-August-2015]. [Online]. Available: http://fortune.com/2015/08/19/smart-home-stupid/

[47] J.-b. Woo and Y.-k. Lim, "User experience in do-it-yourself-style smart homes," in *Proceedings of the 2015 ACM international joint conference on pervasive and ubiquitous computing*. ACM, 2015, pp. 779–790.

[48] P. A. Kodeswaran, R. Kokku, S. Sen, and M. Srivatsa, "Idea: A system for efficient failure management in smart iot environments," in *Proceedings of the 14th Annual International Conference on Mobile Systems, Applications, and Services*. ACM, 2016, pp. 43–56.

[49] "How much does it cost to install a home automation system?" [Online; accessed 02-22-2017]. [Online]. Available: http://fortune.com/2015/08/19/smart-home-stupid/