

Article

Integrated Management Strategy with Feasible Smartness over Heterogeneous IoT Environments

Taehun Yang ¹  and Jinsoo Han ^{2,*} ¹ Department of Computer Science and Engineering, Chungnam National University, Daejeon 34134, Korea; thyang@cclab.cnu.ac.kr² Electronics and Telecommunications Research Institute (ETRI), Daejeon 34129, Korea

* Correspondence: hanjinsoo@etri.re.kr

Abstract: Recently, Internet of Things (IoT) applications have been increasingly deployed in smart domains, such as homes, buildings, and so on. A wide variety of smart devices and solutions bring improved lifestyles. However, current provider-oriented and individual application deployment leads to the separation of a smart domain into respective regions by providers and applications. Such heterogeneous environments hinder unified operation and the utilization of smart IoT applications. Therefore, this Article firstly addresses analyses on conventional smart domain technologies—smart home, smart building, etc.— and deployment in the real world with heterogeneous IoT technologies; then, a novel smart domain strategy for inter-cloud and inter-service operability and mobile-user-attached interactivity is proposed. Performance is compared in terms of user experience and service availability. Finally, numeric analyses are provided to prove the proposed strategy, and the proof-of-concept is presented to show feasibility and performances.

Keywords: Internet of Things (IoT); smart domain; interoperability; inter-service provisioning; user experience



Citation: Yang, T.; Han, J. Integrated Management Strategy with Feasible Smartness over Heterogeneous IoT Environments. *Electronics* **2021**, *10*, 149. <https://doi.org/10.3390/electronics10020149>

Received: 30 November 2020

Accepted: 3 January 2021

Published: 12 January 2021

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.



Copyright: © 2021 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

During the past decade, Internet of Things (IoT) has gained increasing attention and momentum. In 2013, global standard initiative on IoT (GSI-IoT) defined the IoT as “the infrastructure of information society [1]”. The IoT provides objects with connectivity through existing network infrastructure and enables them to be monitored and controlled remotely to create more benefits of integration. The IoT is expected to offer not only simple connectivity of devices but also advanced connectivity of systems and services that goes beyond machine-to-machine (M2M) communications [2]. This advanced connectivity of IoT devices enables various applications, such as smart homes, smart grids, and smart transportation [3,4]. These IoT environments pervasively help individuals, businesses, and society on a daily basis.

As the state-of-the-art strategy, many studies regarding interoperation of IoT devices have been performed to provide more enhanced IoT services [5–7]. Social IoT (SIoT) was introduced a paradigm of a social network of intelligent devices [5]. It is analogous with social network services for human beings to provide global connection. Autonomous systems of things (ASoT) was introduced to address future-driven routing architecture putting large-scale IoT domains into the legacy one [6]. ASoT considers layer-three connectivity. Dynamic social structure of things (DSSoT) was proposed to boost sociality and narrow down computational and networking contextual complexity based on situational awareness [7]. While SIoT and ASoT take into account global connectivity, they focus on configuring temporary social relations among co-located smart devices. However, those previous studies required high intelligence and a new protocol stack in devices, gateways, and cloud servers; or powerful computing elements for centralized control.

On the other hand, recently, commercial IoT applications have already been launched and installed to enhance smart domains, such as smart homes, smart buildings, and so on. Each IoT application consists of smart domain appliances, an IoT gateway, and a cloud server over its own platform; it mainly provides its own IoT services. Such services are controlled by mobile applications or voice interface devices, e.g., Amazon Echo, Apple Homepod, or Google Home, based on a cloud server. Different IoT services cause provider-dependent heterogeneous IoT environments. Such heterogeneous IoT environments pull down a conventional smart domain paradigm where a powerful gateway can wrap up all kinds of smart devices [8,9]. However, the heterogeneous IoT environments still have problems regarding inter-operability and connectivity of IoT services in terms of real-world feasibility. In addition, a stationary service interface of each platform restricts service boundary and inter-service operability; the current smart domain technologies over the heterogeneous environment are practically not so smart.

This Article proposes a novel smart domain strategy for user-friendly service provisioning over heterogeneous IoT environments and inter-service operation for integrated management of various devices. Thus, the novel smart domain strategy can provide inter-connectivity among such heterogeneous IoT platforms. This novel practical inter-connectivity and service strategy is called mobile-driven inter-operation and user-attached interactivity (MIUI) and object-oriented management of things (OOMoT). The MIUI and OOMoT framework enables smartness in a smart domain and enhance user experience (UX). User convenience and service availability are evaluated to show the MIUI's integrated interface and OOMoT's unified management.

A proof-of-concept was developed to not only provide interactivity between a smart phone and IoT devices, but also connect legacy devices. In addition, a mobile application was developed for integrated control of diverse application services and a user-attached mobile interface. Experimental results prove that the MIUI and OOMoT can improve UX and achieve feasibility of synthesizing heterogeneous IoT platforms.

Following the introduction, Section 2 investigates the recent smart domain technologies. Section 3 describes the current IoT environment and IoT services architectures. In Section 4, MIUI is explained regarding the architecture for the integrated interface. Section 5 describes OOMoT architectures and operations for unified management. Sections 6 and 7 explain numerical analyses and experimental results for MIUI and OOMoT. Finally, Section 8 concludes this Article.

2. Survey on Smart Domain Technologies

In this section, we explore four smart domain technologies: smart home, smart building, smart factory, and smart city. Table 1 shows a comparison of these smart domains.

Table 1. Comparison of four domain examples.

	Smart Home	Smart Building	Smart Factory	Smart City
Location (Size)	House (Small)	Building (Medium)	Factory (Large)	City (Very Large)
Infrastructure Element	Sensors, Consumer Electronics, Home Appliances	Sensors, HVAC, Lights	Sensors, Machineries, Lights	Sensors, Transportation, Facilities
Server Type	Cloud	Cloud/Local	Cloud/Local	Cloud
Smart Services	Home Automation, Home Security	Building Automation, Energy Efficiency, Tenants Comfort	Energy Efficiency, Production Automation, Worker Safety	Public Safety, Better Mobility, Energy Efficiency, Health-care

Table 1. Cont.

	Smart Home	Smart Building	Smart Factory	Smart City
Network Connection	Network-enabled Home Devices	Network-enabled Equipment	Network-enabled Equipment	Network-enabled Sensors and Assets
Benefit	Energy Reduction, User Comfort	Energy Reduction, Tenant Comfort	Energy Reduction, Worker Safety, High Productivity	Energy Reduction, Safety & Health, Sustainability

2.1. Smart Home

As technologies have been developing, more and more home devices have been produced for user convenience. There are home appliances such as refrigerators, air conditioners, washing machines, robot cleaners, TVs, and so on. The home automation technology has been emerged to control these home devices automatically and remotely. The home automation networks comprise wired and wireless communication networks that enable monitoring and control applications toward home devices. A home automation system typically connects home devices to a gateway. Wireless network infrastructure such as Wi-Fi stimulates home automation enabling wireless home devices to easily connect to the home gateway. Accordingly, the system can be remotely accessed via an Internet-connected gateway using a computer outside home. As smart phones have been widely adopted, network-enabled home devices are monitored and controlled by mobile applications. Unfortunately, most of these home devices have their own application and do not collaborate with others. Typical home automation has been focused on connection only. This kind of home is called a “connected home”.

Meanwhile, a new type of home automation is configured as follows: home devices—a home gateway—a cloud server—a smartphone (applications). The home devices continuously communicate with the cloud server to update new data. The cloud server provides new data to the smartphone, when requested by a user. The user can browse their home status and control home devices via the cloud server through applications. This kind of home is called a “smart home”. Various smart home applications are provided based on home devices with web services and cloud computing [10].

2.2. Smart Building

As the size of buildings is increasing and more equipment is being installed, efficient building management is required. In the buildings, there is a lot of mechanical and electrical equipment, such as ventilation, lighting, power systems, fire systems, and security systems. That equipment is necessary for indoor environment quality and occupant satisfaction. Building operators need to manage the equipment properly to maintain a satisfactory quality of environment while minimizing the energy consumption.

Recently, the building automation system (BAS) or building management system (BMS) has emerged. A BAS or BMS is a computer-based control system installed in buildings that controls and monitors the building’s mechanical and electrical equipment [11]. The BAS/BMS achieves centralized automatic control of equipment to improve efficient operation of building systems, reduce energy consumption and operation costs, and increase the lifetimes of utilities.

All sensors and controllers are connected to a central computer through various communication methods, wired or wireless. Building operators can monitor and control all equipment in the building for proper and efficient operation. In an operating room, they easily and quickly manage the whole building to maintain occupant comfort and minimize energy consumption. However, there are no optimization or analytic insights since the BAS/BMS focuses on automation of equipment.

2.3. Smart Factory

Recently, the rise of the Fourth Industrial Revolution, Industry 4.0, has made the transformation of factory automation. The National Institute of Standards and Technology (NIST) defines “Smart Manufacturing” as systems that are fully-integrated; collaborative manufacturing systems that respond in real-time to meet changing demands and conditions in the factory, in the supply network, and in customer needs. Additionally, the smart manufacturing leadership coalition (SMLC) says that, “Smart Manufacturing’ is the ability to solve existing and future problems via an open infrastructure that allows solutions to be implemented at the speed of business while creating advantaged value” [12].

The smart factory is defined as numerous sensors, IoT devices, various production machines, and integrated machine controllers being fully connected to various control systems. The smart factory is enabled by IoT-connected devices, big data, data analytics, robotics, machine learning, sensor technologies, artificial intelligence, and integrated machine control [13]. Among these, IoT-connected devices and sensor technologies contribute to gaining tremendous data for analytics to optimize production. The smart factory transforms the manufacturing process from a complex of isolated silos into a seamless production environment.

Major features of the smart factory are as follows: connectivity, optimization, transparency, proactivity, and agility [14]. The smart factory requires components and processes to be fully connected to create valuable data for real-time decisions. Connectivity is the most important feature of the smart factory. Optimization based on analytics and artificial intelligence enables highly automated production and material handling with minimal human intervention. The transparency of the captured data provides real-time data visualization that enables actionable insights for either human or autonomous decision making. Proactivity enables employees and systems to act in advance before issues arise, rather than simply reacting to arisen issues after they occur. Agility enables flexible and adaptable scheduling and configurable factory layouts and equipment. These features provide greater operating efficiency, minimal machine downtime, increased worker safety, optimized inventorying, and supply chain management [13].

2.4. Smart City

The smart city can be defined as the full connection of all citizens, all devices, all assets, and all intelligence based on analysis of tremendous amounts of data. As a result, the smart city concept integrates various devices and assets connected to the network so that it optimizes the efficiency of city operations and services and provides various services to citizens [15]. Smart city technology allows governments to interact directly with both citizens and city infrastructure and to monitor what is happening in the city and how the city is evolving.

For this, the smart city uses ICTs so that its critical infrastructure, and its components and public services, are more interactive and efficient. ICT is a key element of the smart city; an emerging IoT technology helps to enhance the smart city. The ICT includes an intelligent network of connected objects and machines that transmit data using wireless technology to the cloud. The ICT also uses different types of electronic data collection sensors to supply information which is used to manage assets and resources efficiently [15]. In addition, IoT, the cloud, and big data based on ICT help municipalities, enterprises, and citizens make better decisions for a better quality of life.

3. Landscape about Internet of Things Deployment

3.1. Heterogeneous Environments

The representative one of a smart domain with a heterogeneous environment is the smart home. As shown in the left in Figure 1, the legacy smart home consists of a single almighty home gateway and various kinds of wired and/or wireless devices. The single almighty home gateway plays the role of a bridge between a user and home devices. A

user can access the home devices through the Internet. However, this legacy smart home has not proliferated in spite of numerous studies and developments [8,9].

Currently, IoT is popular in smart homes. Various IoT devices have been deployed in smart homes for everyday life. Users have an easy way to manage the IoT devices using smartphones as shown in the center in Figure 1. Numerous IoT providers make their own platforms, such as cloud servers to connect IoT devices to Internet. Users can access the IoT devices by using a provider-dependant mobile application interacting with cloud servers. Users can deploy various IoT devices from different providers that provide their own IoT gateways, cloud services, and mobile applications. Several different mobile applications are installed in a smartphone. The current smart home with IoT is heterogeneous. The more different IoT platforms are deployed, the more heterogeneous the smart home becomes.

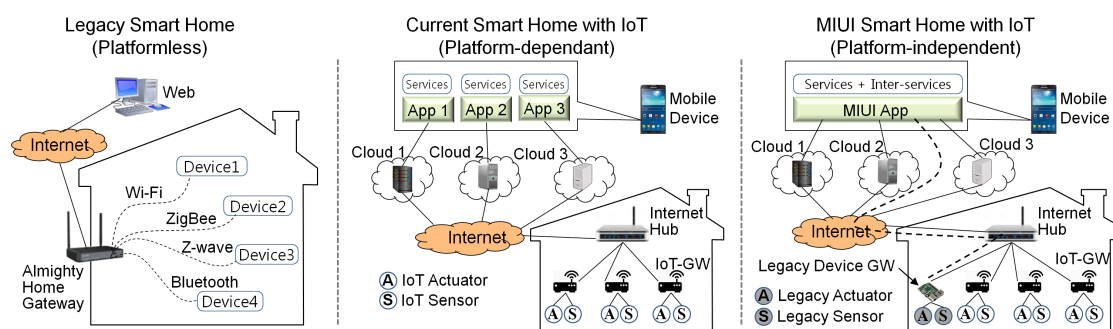


Figure 1. Evolution of smart homes in three phases. System architecture for the mobile-driven interoperation and user-attached interactivity (MIUI).

3.2. IoT Service Architectures

IoT service architectures can be classified into three phases: legacy, current, and MIUI. The characteristics can be analyzed in terms of various aspects: physical structure, service channel, service controller, and service integration. Table 2 shows comparison of three IoT service architectures. The physical structure means components of smart home. The current and MIUI architectures have multiple IoT gateways, whereas the legacy architecture has a single almighty gateway. The service channel means a method of using services. The legacy architecture has a single service channel via the almighty gateway. The current architecture has multiple service channels, such as various mobile applications. The MIUI architecture has a single integrated service channel; users can access and control the IoT devices using a single mobile application. The service controller means a physical user interface to enable IoT services. The legacy architecture uses a single web-based interface, such as desktop or laptop computer. The current architecture uses various controllers, such as smart phone, Amazon Echo, Google Home, etc., based on current IoT environments. Users can use a smart phone outside home and/or voice assistants, such as Echo and Home inside home. The MIUI architecture has an integrated controller that enables interoperation of IoT services compared to the current architecture. The service integration means whether inter-service operability or inter-service provisioning is available. The legacy and current architectures have no service integration, whereas the MIUI architecture enables inter-service provisioning and generates new IoT services.

Table 2. Comparison of three IoT service architectures.

	Platformless (Legacy)	Platform-Dependant (Current)	Platform-Independent (MIUI)
Physical Structure	Single Almighty Gateway + Various Devices	Multiple IoT Gateways + IoT Devices	Multiple IoT Gateways + IoT Devices
Service Channel	Single	Multiple	Single
Service Controller	Single	Independent Multiple	Independent Single
Service Integration	No	No	Yes

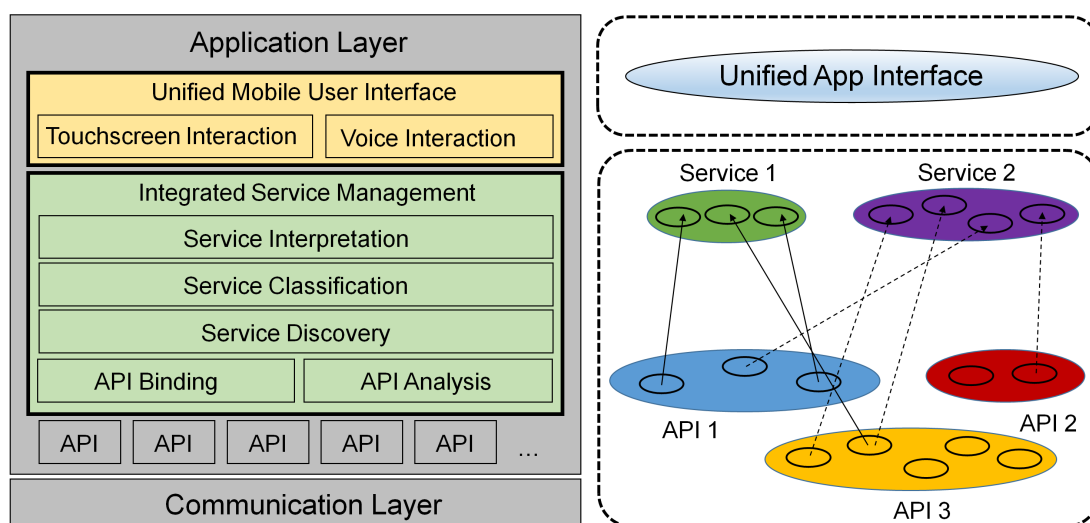
4. Mobile-Driven Interoperation and User-Attached Interactivity (MIUI)

4.1. MIUI Overview

As shown in the right part of Figure 1, a new architecture named MIUI is proposed to provide IoT users with interoperability and user-customized interactivity for high-quality UX over current IoT environments. The MIUI is realized as a mobile application over existing applications and cloud servers. Each cloud server contains the list and data of all IoT devices in its own group because users register at the cloud server when they deploy the IoT devices. Actually, as a registered user, users can access the registered cloud servers via authentication process using identity and password. The MIUI architecture is based on user's accessibility to the registered cloud servers. A MIUI mobile application accesses the registered cloud servers, gets the data of IoT devices, and controls the IoT devices. While provider-dependent mobile applications access their own cloud server and manage their own IoT devices, the MIUI mobile application can access and manage all IoT devices regardless of providers. As a result, it can mash IoT devices up and create new synergetic services to give users advanced comfortableness and convenience.

4.2. MIUI Architecture

Figure 2 shows the architecture of the MIUI. The left part describes the MIUI software stack. The communication layer enables a user's smartphone and the cloud server to share, modify, and manipulate data. Various APIs serve as a basic function for services. To obtain these APIs, users directly should acquire the authentication/authorization for control of smart devices provided on their own platform. That is, it requires progress which registers each device of different platforms to the MIUI architecture by the user. The integrated service management (ISM) binds numerous APIs and creates new IoT services. The integrated service management (ISM) binds numerous APIs and creates new IoT services.

**Figure 2.** MIUI architecture: software stack (left) and service mashup (right).

API binding binds several APIs from different cloud servers to create a new service. API analysis checks and analyzes the functionality of APIs in cloud servers. Service discovery (SD) searches the available services from registered IoT platforms and lists up platform-dependent services. The discovered services from different platforms are used for service mashup. Service classification (SC) categorizes discovered available services according to their characteristics such as monitoring (sensor) and control (actuator). The services in the monitoring class are used for inputs to a new service; the ones in the control class are used for outputs of a new service. Service interpretation (SI) binds the discovered services in both monitoring and control classes. A new IoT service is produced by binding several services in both classes. The unified mobile user interface (UMUI) is composed of typical touchscreen interaction and voice interaction. Voice recognition is increasingly being adopted in smartphones.

The right part of Figure 2 illustrates new service mashup. The API 1, API 2, and API 3 mean APIs from different platforms. The APIs of one platform can be simultaneously used by different services; new services can adopt APIs from several platforms. For example, service 1 adopts two APIs from platform 1 and one API from platform 3. One API from platform 3 is adopted by both service 1 and service 2.

4.3. Interaction between Users and Smart Devices

Figure 3 shows the interaction process where users access and control smart devices using a smart phone on the user side. Layered functional blocks are shown on each side. There are two kinds of interactions. First, users can access smart devices via IoT gateways. In this case, IoT devices are registered using device registry (DR) and managed using device management (DM) in the cloud. Data management in the cloud manages the status and events of IoT devices using IoT protocols.

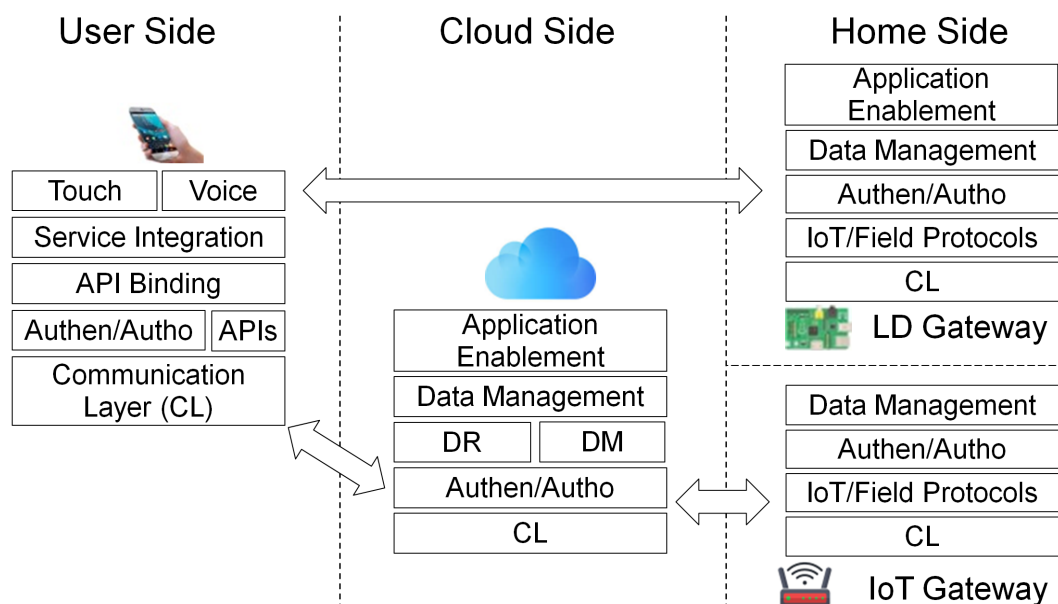


Figure 3. Interactions relevant for the smart home: user side (left), cloud side (center), and home side (right). Authen/Autho: authentication/authorization, DR: device registry, DM: device management, LD: legacy device.

Application enablement helps users use IoT services in the cloud as if they directly access IoT devices. The IoT gateway plays a role of a bridge to connect IoT devices to the cloud. The IoT gateway uses IoT protocols to communicate with the cloud; it uses either IoT or field protocols to communicate with smart devices. Authentication and authorization are used for security. Data management in the IoT gateway manages the connected IoT devices.

Second, legacy devices are connected to a legacy device (LD) gateway. Raspberry Pi, a light-weight embedded system, can be a candidate for the LD gateway. Legacy devices are connected to the LD gateway via various interfaces, such as universal serial bus (USB), Wi-Fi, Bluetooth, and several serial communication means. Users can directly access the LD gateway without the cloud. The LD gateway has application enablement as in the cloud to enable users to access and manage legacy devices. Users can connect legacy devices to the LD gateway and access them using a smartphone. Authentication and authorization are also used for security.

Users can create a new service by binding APIs from different cloud servers and LD gateways. They can mash up smart devices with a specific purpose and create a new user-friendly service. The MIUI architecture enables users to manage both IoT and non-IoT devices in the unified interface. The MIUI provides real smartness to the smart domains.

5. Object-Oriented Management of Things (OOMoT)

In this section, the scalable and interoperable management of objects is presented. A hierarchical, decentralized, smart domain managing system, denoted by object-oriented management of things (OOMoT), has been built on the core design principles, and it implements all the principles as system components. The system is set up above the control server, named the agent. Then, the system becomes its hierarchy to embrace all of the smart service areas where smart IoT devices are embedded.

5.1. System Architecture

As shown in Figure 4, the system architecture of OOMoT is located within the entire architecture of IoT services including whole IoT infrastructures and all cloud systems to provide big data analytics. Additionally, applications which rely on such IoT infrastructures and cloud systems are established over the OOMoT layer. Among infrastructures, cloud systems, applications, and the OOMoT layer, interconnection is fulfilled by cross-layered communication.

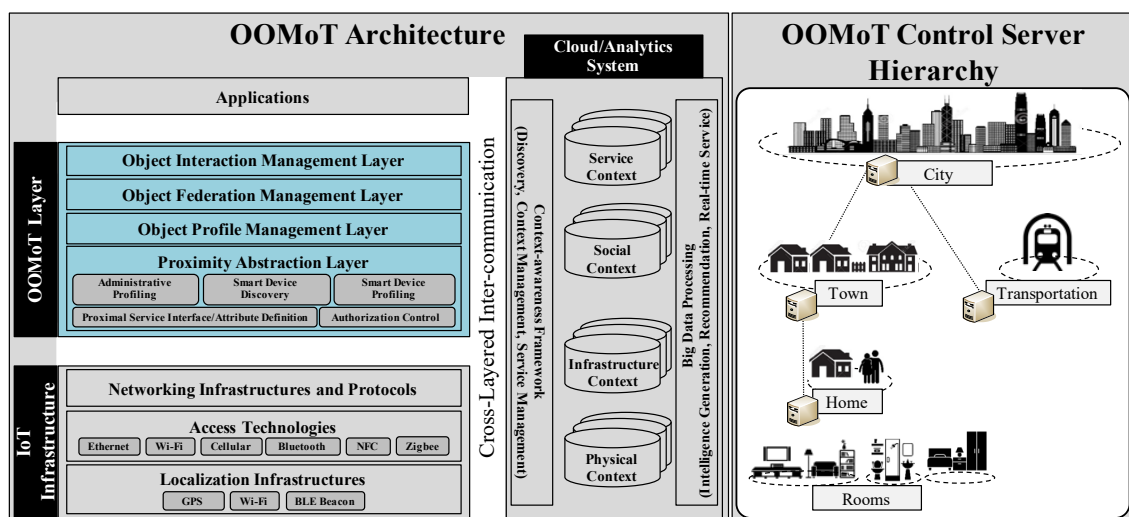


Figure 4. Object-oriented management of things (OOMoT) architecture and control server hierarchy example.

The OOMoT consists of four sub-layers:

1. Proximity abstraction layer: This sub-layer mainly fulfills the core function “proximity abstraction” of this proposed system. It means that a physical space where smart devices are deployed is embodied into the cyber world to be dealt with in online. For this, administrative profiling to get administrator profile information of an object and users who use the object is carried out. Then, the methods and profiles of smart device discovery, discovered devices, and the connection to their cloud systems are figured out. In addition, within the modulated and encapsulated object, possible

function interfaces and attributes are defined. Finally, to manage access control to interfaces and attributes, authorization and authentication mechanisms are involved in this sub-layer. All the information of users, interfaces, and attributes are called as the object profile.

2. Object profile management layer: It manages profiles of all objects including base objects and all federations of them an agent (a control server) copes with. Proximity-based smart services are triggered with this information.
3. Object federation management layer: This sub-layer provides super-categorization of base objects, which are derived from the proximity abstraction layer to embody a physical smart domain as a logical one, or sub-objects logically composing the super-object. This carries out formation of all the federations, called super-objects, of the base objects or sub-objects.
4. Object interaction management layer: All the situations of interaction among objects is controlled by this sub-layer. The main function of this sub-layer is supporting and managing comprehensive and compositive operations and functions provided by interoperation among objects. The interfaces of an object should be recursively called for the comprehensive and compositive operations.

5.2. OOMoT Operations

In deployment processes of an object, each smart device is registered to an agent (a control server) of the object. Smart devices are aware of address(es) of the agent, and the agent maintains their profiles and specifications including addresses, data, functions, access technologies, and states.

1. User management: When a user arrives at a smart domain, the user can explicitly or implicitly have a connection to an agent that the smart device belongs to. The user's profile information (user identifier, information for authorization, available access technologies, addresses, etc.) is disseminated to the agent for authorization and authentication of him/her. The agent confirms authority and sends an acknowledgment message including object type, available access technologies, and their addresses to a service manager, such as user's smartphone or home automation server via available access technologies. The service manager responds to the agent and the connection is established. The agent manages users by maintaining these connections.
2. Service request: In case of pull services, the user sends service request messages to the agent by his/her smart phones, such as a smartphone, via suitable access technologies directly. The request message follows interfaces of the objects. The user may be aware of standardized interfaces or be notified from the agent. According to the requested service, the agent simply replies to the user, requests data to smart devices, and replies to the user, giving commands actuators to relevant actions. In case of push services, the agent could provide proper information to the user or activate actuators triggering from the service manager.
3. Different-scale service request: When an agent receives requests for profiles of objects, the agent sends all the objects' profile information, i.e., super-objects and sub-objects' profiles. Then, a smart service manager for push services or the user's mobile applications for pull services, calls functions to use the objects, and the objects which receive the requests return the current values or do the requested actuations. If a task of a super-object driven from a function call is the compositive function relying on interaction among some sub-objects, the interaction is fully controlled by the agent.
4. User mobility: If a user moves away from a previous object and enters a new object, a new connection is explicitly or implicitly made with a new agent. For seamless and continuous smart service provisioning, concatenated tunneling might be taken into account to relay user profiles, controls, and service profiles to the new agent along with user movement trajectories.

6. Numeric Analyses

6.1. User Experience

To compare the user's experience, we adopted three metrics: traffic volume, touch number, and operation time. It is assumed that one task is available in one mobile application. First, in traffic volume, only application layer traffic is considered: request and reply. In either case, the numbers of service requests and replies are the same to execute tasks. They are proportionate to the number of tasks. Second, in touch number, it is assumed that mobile application execution requires three touches—home button, group application icon, and the corresponding application—and that one more touch is required to execute a task. In the heterogeneous IoT environment, every task needs four touches because a user has to move to another application for a new task, whereas the MIUI can execute all tasks in the same application after initial execution. The touch numbers of the heterogeneous IoT environment and the MIUI are $4 \times N_{task}$ and $3 + N_{task}$, respectively. The MIUI reduces touch number down to around one-third. Third, in operation time, it is assumed that task execution time (T_{task}) is constant, that it takes user latency (T_{user}) to decide to execute another task after the end of one task, and that application switching time ($T_{appswitching}$) is required. Heuristically, we assume that T_{task} , T_{user} , and $T_{appswitching}$ are 1.5, 0.5, and 3.0 s, respectively. The MIUI reduces the operation time by half. As shown in Figure 5, the MIUI is superior to the normal heterogeneous IoT environment in terms of user convenience.

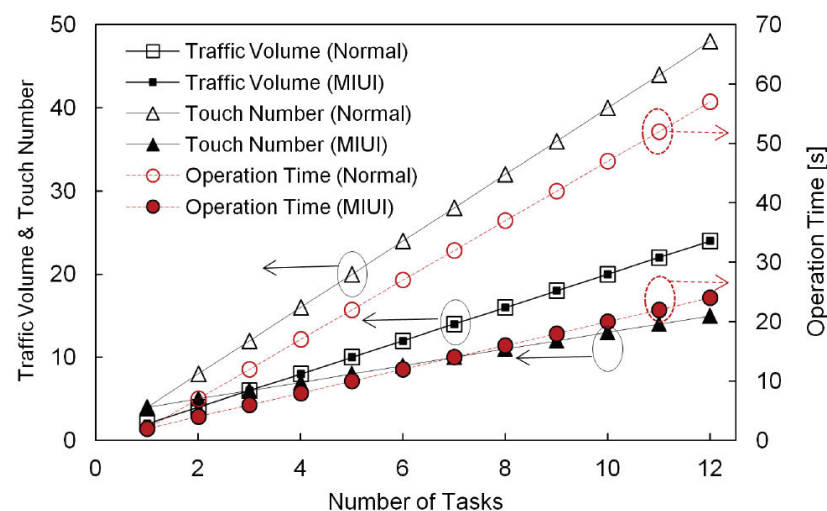


Figure 5. User convenience: Comparison of traffic volume, touch number, and operation time versus the number of tasks.

6.2. Service Availability

Figure 6 shows comparison of the number of tasks and services in the heterogeneous IoT environment and in the MIUI. Domain means a provider-dependent application like a mobile application. Domains 1, 2, and 3 have 2, 4, and 3 available tasks, respectively. Each domain has the same number of services with that of tasks. In addition, each domain can create a new service by binding two tasks. Three or above three tasks are not considered. Domains 1, 2, and 3 create 3, 10, and 6 new services, respectively; the heterogeneous IoT environment creates 19 new services. The MIUI creates 45 new services from 9 available tasks using a unified interface. The MIUI can create more number of services than the heterogeneous IoT environment. The service availability helps enhance UX.

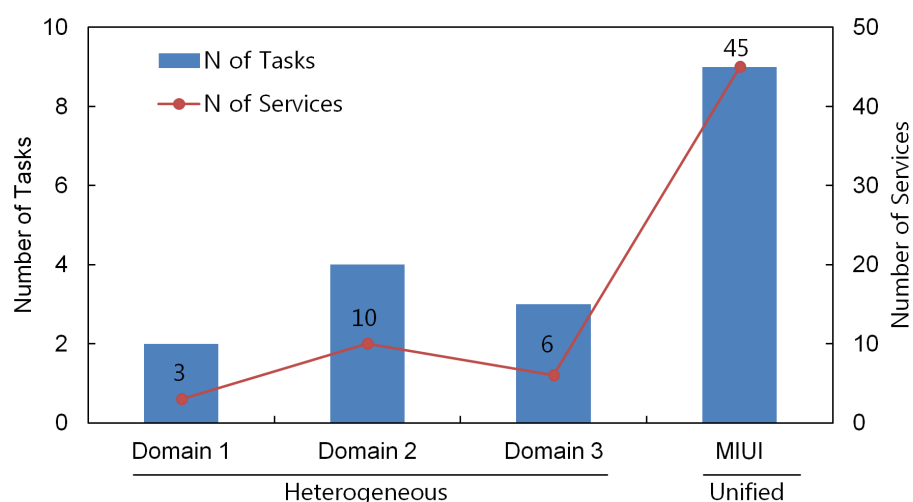


Figure 6. Service availability: Comparison of the number of tasks and services in heterogeneous and MIUI environments.

7. Performance Evaluation

7.1. Proof-of-Concept Prototype

This section explains performance evaluation via the proof of concept. To implement the proof-of-concept prototype, this Article takes into account a smart office scenario with sensor nodes (TinyOS, ATmega128L, CC2240), a networked printer (HP Laserjet P4515x), light switches, a smart phone (iOS version 14), a laptop (Windows 10 OS), and a desktop (Windows 10 OS), as shown in Figure 7. Sensor nodes, which are non-IP devices, adopted the IEEE 802.15.4 and had an LED lighting actuator, an illumination sensor, and a motion sensor. A printer was located in the seminar room and could be accessed by IPv4 over Wi-Fi. Light switches were connected by Bluetooth. The sensor nodes were deployed into two rooms: one was the desk room; the other was the seminar room. Then, two rooms were considered an office belonging to a company. For this, an agent (a control server) was implemented in a desktop to perform abstraction of two rooms and super-categorization of two objects as one office. The personal profiles of employees, attributes, and interfaces of objects were managed in the agent. A laptop and a smartphone were used as the smart service manager and the user's smart phone, respectively.



Figure 7. Smart office scenario with proximity abstraction and OOMoT.

7.2. Experimental Results

Figure 8 shows experimental results for each method in two service scenarios, S1 and S2. One service scenario S1 is that the smart service manager turns on the light if people exist in one room and turn off the light if no one exists. Another service scenario S2 is that the user's smart phone in the desk room tries to print out a paper via the networked printer located in the seminar room. The experiments are evaluated in terms of communication cost and control complexity.

Figure 8a describes a communication cost for each service scenario in legacy method and OOMoT. The communication cost means the cost for exchanging messages between relevant devices to successfully accomplish a service. Thus, the communication cost is represented by the number of messages delivered. In this experiment, the messages are classified into four types: response, request, cloud discovery (CD), and local discovery (LD). In the case of legacy method in S1 (L-S1), the smart service manager discovers the light switches and a motion sensor in local area. Then, the service manager sends a request message to discovered motion sensor, and then receives a response message from the motion sensor. According to the result of the response message, the service manager exchanges a request/response message with light switches to control them. Thus, the communication cost of L-S1 is 9. Meanwhile, in the case of legacy method in S2 (L-S2), we could see the use of CD messages since the user's smart phone asks the cloud to discover a printer in another room. Thus, the communication cost of L-S2 is higher than L-S1. On the other hand, in the case of OOMoT in S1 (OOMoT-S1), the service manager sends one request message and one response message. Unlike the legacy method, OOMoT does not require a discovery process since the service manager only communicates with the agent, which has information for relevant devices and could automatically control them. Likewise, OOMoT shows the same results as S1 in S2. Consequently, OOMoT performs about 5 times better than the legacy method in respect of the communication cost.

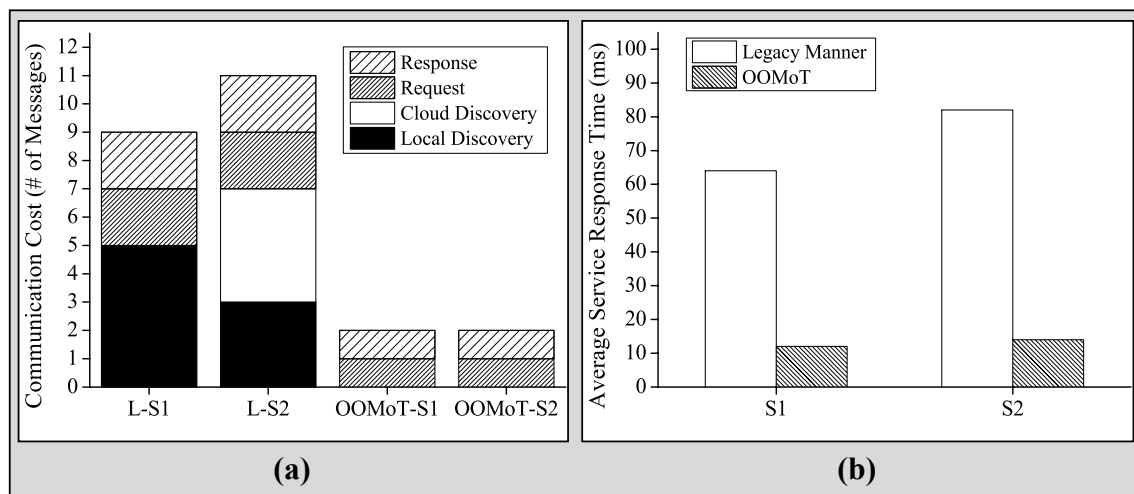


Figure 8. Experimental results of legacy method and OOMoT for two service scenarios S1 and S2 (L-S1, L-S2, OOMoT-S1, and OOMoT-S2): (a) communication cost (b) control complexity.

Figure 8b illustrates the average service response time for each method in terms of a control complexity. In cases of S1 and S2, the legacy method spends more response time since legacy method without the unified management should communicate with not only all relevant devices related to service but also could require additional cloud discovery process. Especially, in S2, the response time is higher than S1 since the smart phone cannot discover the printer in local discovery and then finds it via cloud services provided in different domains. On the contrary, the OOMoT requires the lower average service response time compared to the legacy method. The unified management of OOMoT brings the effects that unnecessary communication is reduced and it makes control relevant

devices to easy and simple. As a result, the OOMoT shows 5 to 6 times lower response time than legacy method.

8. Conclusions

In this Article, we addressed practical limitations of the current heterogeneous IoT environment of smart domains. We compared and analyzed three IoT service architectures. Then, we proposed the MIUI architecture to overcome the practical limitations and give users more integrated smart services and high-quality UX. Additionally, this Article proposes a novel smart device management paradigm considering end user's perspective. In this new strategy, proximity abstraction is used to embody a physical domain with smart devices, where an end user currently stays; a logical object is defined; and object-oriented management is used to not only provide easy discovery and utilization of available smart devices in the physical domain to the user, but also achieve flexible and scalable smart service provisioning. The performance was evaluated in terms of user convenience—for instance, regarding touch number and operation time, and service availability. The MIUI shows higher user convenience by achieving less operation time and lesser touch numbers. It also shows higher service availability. The proof-of-concept showed an implementation of this strategic system and provided experimental results to prove the system can improve the performance of IoT-based smart service operations.

Author Contributions: Data curation, J.H.; Software, T.Y. and J.H.; Supervision, T.Y.; Visualization, T.Y.; Writing—original draft preparation, T.Y.; Writing—review and editing, J.H. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by the Korea Institute of Energy Technology Evaluation and Planning (KETEP) and the Ministry of Trade, Industry & Energy (MOTIE) of the Republic of Korea (No. 20191210301820).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. International Telecommunication Union (ITU). *Internet of Things Global Standards Initiative*; International Telecommunication Union (ITU): Geneva, Switzerland, 2015.
2. Höller, J.; Tsiatsis, V.; Mulligan, C.; Karnouskos, S.; Avesand, S.; Boyle, D. *From Machine-to-Machine to the Internet of Things: Introduction to a New Age of Intelligence*; Academic Press: Cambridge, MA, USA, 2014.
3. Karnouskos, S. The cooperative Internet of Things enabled Smart Grid. In Proceedings of the 14th IEEE International Symposium on Consumer Electronics (ISCE2010), Braunschweig, Germany, 7–10 June 2010.
4. Zanella, A.; Bui, N.; Castellani, A.; Vangelista, L.; Zorzi, M. Internet of things for smart cities. *IEEE Internet Things J.* **2014**, *1*, 22–32. [[CrossRef](#)]
5. Atzori, L.; Iera, A.; Morabito, G. SIoT: Giving a Social Structure to the Internet of Things. *IEEE Commun. Lett.* **2011**, *15*, 1193–1195. [[CrossRef](#)]
6. Park, S.; Crespi, N.; Park, H.; Kim, S.-H. IoT routing architecture with autonomous systems of things. In Proceedings of the 2014 IEEE World Forum on Internet of Things (WF-IoT), Seoul, Korea, 6–8 March 2014.
7. Hussein, D.; Park, S.; Han, S.N.; Crespi, N. Dynamic social structure of things: A contextual approach in CPSS. *IEEE Internet Comput.* **2015**, *19*, 12–20. [[CrossRef](#)]
8. Chung, T.-Y.; Mashal, I.; Alsaryrah, O.; Hsu, T.-H.; Chang, C.-H.; Kuo, W.-H. Design and implementation of light-weight smart home gateway for Social Web of Things. In Proceedings of the 2014 Sixth International Conference on Ubiquitous and Future Networks (ICUFN), Shanghai, China, 8–11 July 2014.
9. Kim, S.-M.; Choi, H.-S.; Rhee, W.-S. IoT home gateway for auto-configuration and management of MQTT devices. In Proceedings of the 2015 IEEE Conference on Wireless Sensors (ICWiSe), Melaka, Malaysia, 24–26 August 2015.
10. Soliman, M.; Abiodun, T.; Hamouda, T.; Zhou, J.; Lung, C.H. Smart Home: Integrating Internet of Things with Web Services and Cloud Computing. In Proceedings of the 2013 IEEE 5th International Conference on Cloud Computing Technology and Science, Bristol, UK, 2–5 December 2013; pp. 317–320.

11. Manic, M.; Wijayasekara, D.; Amarasinghe, K.; J. Rodriguez-Andina, J. Building Energy Management Systems: The Age of Intelligent and Adaptive Buildings. *IEEE Ind. Electron. Mag.* **2016**, *10*, 25–39. [[CrossRef](#)]
12. Davis, J.; Swink, D.; Tran, J.; Wetzel, J.; Profozich, G.; McKewen, E.; Thys, R. *CMTC's Guide to Smart Manufacturing*; Technical report; CMTC: Torrance, CA, USA, 2015.
13. Mabkhot, M.M.; Al-Ahmari, A.M.; Salah, B.; Alkhalefah, H. Requirements of the Smart Factory System: A Survey and Perspective. *Machines* **2018**, *6*, 23. [[CrossRef](#)]
14. Burke, R.; Mussomeli, A.; Laaper, S.; Hartigan, M.; Sniderman, B. *The Smart Factory: Responsive, Adaptive, Connected Manufacturing*; Deloitte University Press: Westlake, TX, USA, 2017.
15. Du, R.; Santi, P.; Xiao, M.; V. Vasilakos, A.; Fischione, C. The Sensable City: A Survey on the Deployment and Management for Smart City Monitoring. *IEEE Commun. Surv. Tutor.* **2019**, *21*, 2, 1533–1560. [[CrossRef](#)]