An Open Internet of Things System Architecture based on Software-defined Device

Pengfei Hu, *Student Member, IEEE,* Huansheng Ning, *Senior Member, IEEE,* and Liming Chen, *Senior Member, IEEE*

***Abstract*—The Internet of Things connects more and more devices and supports an ever-growing diversity of applications. The heterogeneity of the cross-industry and cross-platform device resources is one of the main challenges to realize the unified management and information sharing, ultimately the large-scale uptake of the Internet of Things. Inspired by software-defined networking, we propose the concept of software-defined device and further elaborate its definition and operational mechanism from the perspective of cyber-physical mapping. Based on the device-as-a-software concept, we develop an open Internet of Things system architecture which decouples upper-level applications from the underlying physical devices through the software-defined device mechanism. A logically centralized controller is designed to conveniently manage physical devices and flexibly provide the device discovery service and the device control interfaces for various application requests. We also describe an application use scenario which illustrates that the software-defined device based system architecture can implement the unified management, sharing, reusing, recombining and modular customization of device resources in multiple applications, and the ubiquitous Internet of Things applications can be interconnected and intercommunicated on the shared physical devices.**

***Index Terms*—Internet of Things, software-defined device, cyber-physical mapping, IoT system architecture, device discovery, devices sharing.**

I. Introduction

With the development of Internet of Things (IoT), Cyber-Physical Systems (CPS), and Mobile Internet, more and more devices are connected in the cyber space (also known as information space) [1]. The efficient management and scheduling of the large number of devices is critical for improving the overall capability and efficiency of IoT services and the energy efficiency of IoT infrastructures [2].

In current IoT applications, due to the different application areas, the different application target, and the different developer, the development of IoT is industry-specific, domain-dependent, and fragmented in technology, standards and applications [3]. Therefore, the piecemeal "information island" and "application island" are formed. Under these circumstances, the interoperability and cross-domain composability of device services become one of the major challenges.

In addition, devices used in different industries, and even the same type of devices produced by different manufacturers usually have different technical standards, application standards and service specifications in different application domains. Because the description, data formats and control instructions of underlying devices are different, the upper-level applications cannot directly access and process the data produced by the underlying cross-platform devices. It is also difficult for such IoT applications to uniformly manage and schedule these device resources. This leads to the heterogeneity between the multiple applications, and hinders the sharing and reusing of both information and resources among IoT applications.

In order to address the sharing and interoperability issues of heterogeneous devices, the traditional way is to define some industry specifications and international standards, for example, Bluetooth, Universal Plug and Play (UPnP), and Zero configuration networking (Zeroconf). However, most standards are incompatible with each other so that the interoperability among standards still cannot be fulfilled [4]. Some researchers have adopted middleware technologies which standardize a set of protocols and formats to interact with other devices, for example, Common Object Request Broker Architecture (CORBA) and Java-Remote Method Invocation (Java-RMI). Though these methods can meet the interoperability, they are not flexible for device resources discovery and service composition [5]. Faraci et al. [6] proposed a Network Functions Virtualization (NFV) approach to share home multimedia devices. This method instantiated a virtual network function called virtual presence to export a real hardware or software resource physically or virtually located in the same personal network of a given user towards another personal network.

Inspired by software-defined networking (SDN) [7], this article proposes the concept of software-defined device (SDD) and an open IoT system architecture based on SDD in order to manage heterogeneous device resources dynamically available in an open IoT deployment scenario. The SDD abstracts and describes the underlying physical devices in their virtualized cyber models by ontology and knowledge engineering technologies to implement the mapping from physical space to cyber space, and decouple the hardware and software. The control and management functions are separated from physical hardware devices, and running it as software instead. The hardware devices become programmable, which enables the flexibly customized control, and simplifies the development and deployment of new device service. The SDD has two main functions. On the one hand, it can control, manage and schedule the global device resources through the virtualization of hardware devices, and on the other hand, it provides IoT applications with various device resources and public services through programmable software definitions.

The open IoT system architecture based on SDD paradigm is designed in this article through adding the SDD layer into the traditional IoT architecture. By adopting the bottom-oriented device description modelling and the upper-level

application requirements oriented device discovery methods, the software definition mechanism of this architecture can make the upper-level applications not worry about the details of underlying physical devices, and completely separates the IoT applications from the physical devices. The scheme provides a framework for the unified management, sharing, reusing, on-demand segmentation, recombination and modular customization of physical device resources in IoT, which will greatly improve the utilization efficiency of device resources. In addition, the system architecture implements interconnection and intercommunication of the cross-industry and cross-platform IoT applications.

The main contributions of this article are shown as follows:

- Based on the perspective of cyber-physical mapping, we propose the concept of SDD to realize the software virtualization of physical hardware device resources and separate the control and management functions as the software instead.
- We design an open IoT system architecture based on SDD to provide a unified framework and infrastructure for the interconnection and intercommunication of the heterogeneous device resources and the industry-specific, domain-dependent, and fragmented IoT applications.

II. The Principle of SDD

A. The SDD Concept

There exist various wide software-defined objects in different technical fields, including SDN, software defined systems (SDSystem), software defined storage (SDStorage), software defined security (SDSecurity), and software defined data center (SDDC) [8]. In these paradigms, the main core concept and essence of software definition are accordant and uniform, which separates the software and hardware and realizes the modularized segmentation and on-demand recombining of hardware resources [9], [10]. It manages and schedules these virtualized hardware resources through controlling software, so that the whole system function can be flexibly customized and expanded [11].

The concept of software definition is similar to cyber-physical mapping. In IoT and CPS, the physical object, which is the existence form of things in physical space, is linked into cyber space through various sensors. A corresponding digital counterpart or clone of a physical object can be generated in cyber space by comprehensively describing properties and abstracting knowledge and rules of the physical object, called cyber object [12]. The cyber object is the abstract existence form of physical object in cyber space. In the whole lifetime of a physical object, the corresponding cyber object will be generated, and evolved, and terminated accordingly. For example, Ma et al. [13] proposed a cyber-physical mapping model to human beings, called cyber-individual (Cyber-I) model. It is a unique and comprehensive counterpart or special digital clone in cyber space of a real-individual (Real-I) in physical space. It describes the mapping relationships between Cyber-I and Real-I.

The processes of cyber-physical mapping mainly include two aspects: the mapping from physical space to cyber space and the mapping from cyber space to physical space. In the former case, the natural attribute data of physical object are obtained through a variety of sensors. By processing and analyzing these attributes and their related history data, further in-depth information and knowledge can be abstracted and extracted. Based on the knowledge, the corresponding cyber object is generated by cyber-physical mapping model in cyber space. In the latter case, the corresponding cyber objects are searched and discovered according to the application requirements. By computing and analyzing the cyber objects, the applications can manipulate the corresponding physical objects to act on physical space.

For the device resources in IoT, the definition of SDD is proposed by integrating the concept of software-definition and the perspective of cyber-physical mapping. The definition is as following:

SDD is the software virtualization for hardware device, which maps a physical device (Physical-D) in physical space into a virtual cyber device (Cyber-D) model in cyber space by device properties description modelling and service capability abstraction methods. A Physical-D is virtualized, abstracted and pooled to generate a Cyber-D model and provided to IoT applications in the form of services. Meanwhile, applications can automate the use of device resources on-demand by controller. In this paradigm, the separation of hardware device and software is implemented, in which hardware device is responsible for sensing, computing and execution, and software is responsible for management, controlling and scheduling.

In this definition, there are two main meanings. On the one hand, a Physical-D is modeled into a digital counterpart, namely Cyber-D, to realize the virtualized and digital software definition and description of physical device resources. The relationship between a Cyber-D and a Physical-D is an one-to-one correspondence. On the other hand, controller computes and analyzes Cyber-Ds so that cyber space can control and act on Physical-Ds and physical space to develop various IoT applications. This has also an implication of making the programmer not worry about the details of underlying physical devices, and makes full use of the programmability of hardware.

B. The Cyber-Physical Mapping Processes of SDD

According to the previous introduction and definition, the cyber-physical mapping process of SDD includes two steps, namely a Physical-D is modeled into a Cyber-D, and a Cyber-D can control the corresponding Physical-D based on application requirements. The processes are shown in Figure 1.

In the cyber-physical mapping processes of SDD, as the counterpart or clone of a Physical-D in physical space, the Cyber-D is a unique, digital, comprehensive description model in cyber space for every real device. It is generated along with the Physical-D accesses to cyber space for the first time. Moreover, the Cyber-D will evolve (or update) along with the change of the Physical-D, and terminate once the Physical-D disconnects with cyber space [14]. The dynamic life cycle of the Cyber-D, that is creation, evolution and termination phases, is introduced as follows:

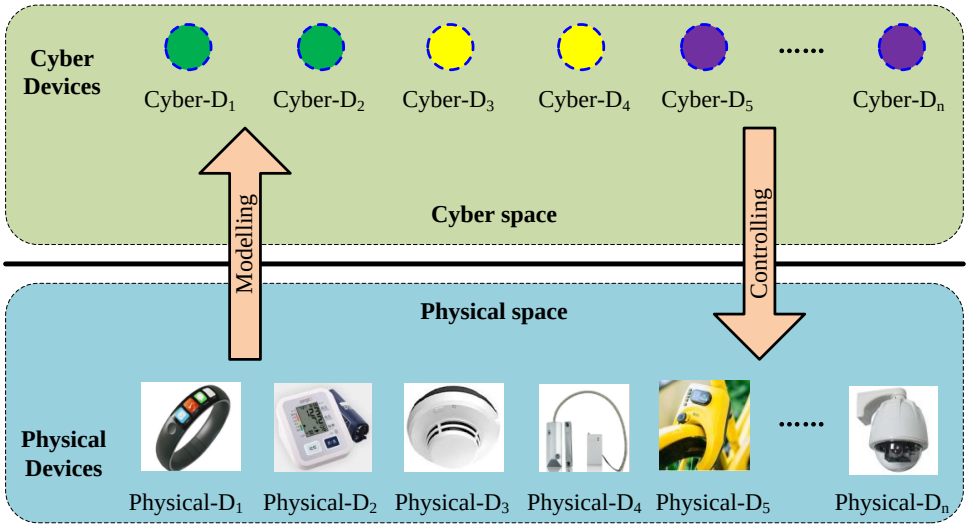

Fig. 1. The cyber-physical mapping processes of SDD.

The creation of Cyber-D: When a new Physical-D accesses to system for the first time, a corresponding Cyber-D will be generated in cyber space. The creation phase initiates a new created Cyber-D instance. In this process, the Physical-D's related information, including identity, location, ownership, functionality, data format, and service capacity, the Cyber-D's related information, including identifier, creation time, and log, and the related context information, including environments, and resource status, are all described into the Cyber-D model. Moreover, domain knowledge and rules related to the Physical-D are also abstracted and extracted as the part of Cyber-D model by processing and analyzing these attributes and context data. In order to formally represent these information, some modelling technologies, for example, semantic ontology and graph theory, need to be adopted.

The evolution of Cyber-D: After a Cyber-D being generated, it will always exist in pairs with its Physical-D. In order to keep consistency, the Cyber-D model needs to update data along with the change of Physical-D, for example, location change and performance upgrades. This also reflects the evolution process of a Cyber-D. The evolution of Cyber-D is mainly embodied in two aspects, which are the changed attributes of Physical-D, and the gradually added domain knowledge and rules about the Physical-D. Some related update methods need to be adopted, including local dynamic update and global dynamic periodic update mechanisms of attributes, real-time increment update and periodic inventory update of knowledge and rules.

The termination of Cyber-D: Resulting from some factors, for example, battery depletion, aging of device units, termination of function, functional displacement, and so on, a Physical-D will gradually terminate and disconnect with the cyber space. Correspondingly, the Cyber-D will also terminate together with the Physical-D. This means the disappearance of device functionality and service capabilities. In addition, because there may be collaborative relationships among some Physical-Ds, the termination of a Cyber-D may influence on other Cyber-Ds so that they need to update their models.

III. A SDD-based Open IoT System Architecture

A. The architecture of the SDD-based IoT paradigm

In order to actualize the unified management, sharing and reusing of underlying device resources, an open IoT system architecture with cross-domain interoperability need to be designed. Figure 2 shows the hierarchical architecture of the SDD-based IoT paradigm. It is composed of the following four layers, namely the device layer, the network layer, the SDD layer, and the application layer. Compared with a traditional IoT architecture, the SDD-based architecture adds the SDD layer to achieve interconnection and intercommunication of the cross-industry and cross-platform IoT applications [15].

The SDD-based IoT system architecture contains multiple application domains platforms and software definition mechanism. The functions of every layer are as following:

The device layer: This layer consists of various IoT devices, which can be broadly divided into sensing devices and actuating devices. The sensing devices have data uplink function. They are mainly responsible for sensing and collecting the data of physical space, and uploading them to the corresponding application platform. These devices include temperature sensors, fingerprint scanners, blood pressure sensors, and wearable devices, to name but a few. The actuating devices have data downlink functions. They support application platform to control the device operation. These devices include entrance guard switch, some smart domestic appliances, and so on. In particular there are some access devices along with data uplink and downlink functions, such as controllable surveillance camera devices. Access devices have the characteristics of both sensing devices and actuating devices. In practice, they are still mapped into one Cyber-D, but can be abstracted into two kinds of service ability.

The network layer: This layer is the entrance of a Physical-D accessing into cyber space. It is divided into two sub layers: the access network and the core network. The access network is oriented to underlying devices, so there are all kinds of communication protocols of underlying devices, including WiFi,

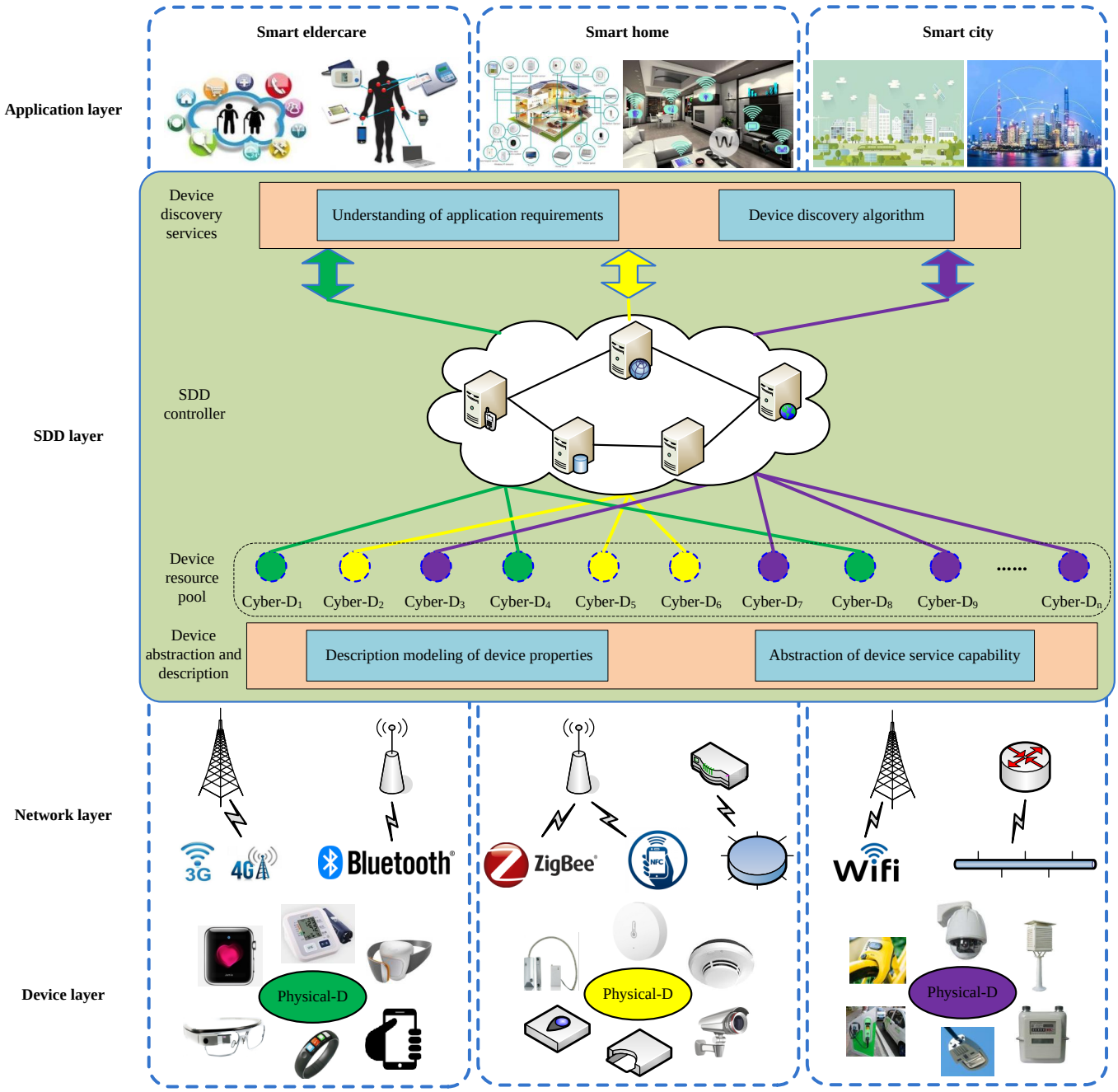

Fig. 2. The hierarchical architecture of the SDD-based IoT paradigm.

BlueTooth, ZigBee, 3G, 4G, NFC, and so on. In general, some smart network devices are adopted as network middleware to implement the compatibility for different communication protocols. The core network is oriented to the upper-level service abstraction and application, so it needs to shield the differences of underlying protocol. This can be achieved by the protocol conversion middleware, so that the upper-level services and applications are independent of the underlying network. Meanwhile, considering the requirements of cross-domain interoperability for different application scenarios, the core network needs to support both IPv4 and IPv6 networks, and their interoperability.

The SDD layer: This layer is the core for the unified management, sharing and reusing of device resources, and realizing interconnection and intercommunication of the cross-industry and cross-platform. It is responsible for abstracting and modelling underlying device resources, and can match appropriate device resources for various application requirements. Furthermore, this layer can modularize the underlying device resources and the upper-level application requirements to reconstitute and customize service mode. As shown in Figure 2, the SDD layer contains four components, including: the bottom-oriented description modelling of device properties and abstraction of service capability; the device resource pool which generates, updates, manages and stores the Cyber-D models; the SDD controller which is responsible for control-

ling and scheduling the global device resources; the upper-level oriented application requirements understanding and device discovery service. Their working mechanisms will be introduced in the following sections in detail.

The application layer: This layer is directly oriented to the IoT users, and epitomizes the SDD function and service. It can support to multiple application domains, including smart eldercare, smart home, smart city and smart healthcare. The application layer mainly analyzes the demands of IoT application scenarios, and converts the physical space demands into the expressions of cyber space demands. When the application is connected into the SDD platform, the corresponding device resources and service capabilities will be directly matched to implement the application demands.

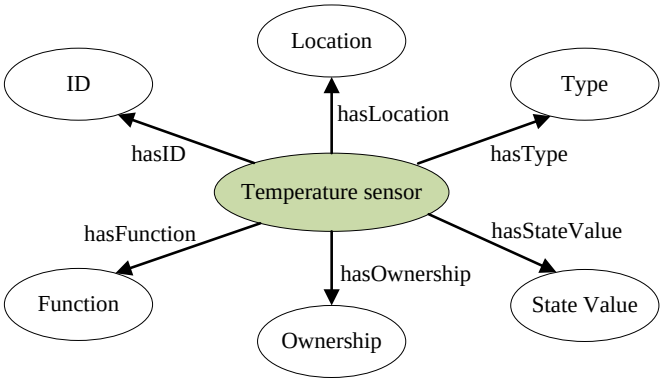

Fig. 3. The ontology description model of a temperature sensor.

B. The Software Definition Mechanism for Describing IoT Devices

In SDD paradigm, the software definition mechanism consists of four parts: device abstraction and description, device resource pool, SDD controller, and device discovery services. Their working mechanisms are as follows.

Device abstraction and description: The effective realization of the SDD paradigm requires not only an explicitly device description so that they can be understood, discovered, shared and scheduled by multiple IoT applications, but also the domain specific knowledge about how these device resources should be combined and cooperate with each other to solve some practical problems. The device abstraction and description module is responsible for abstracting and describing the properties, data and knowledge of underlying devices into service capability in the standardized way. It creates the abstracted device model, namely Cyber-D, and implements the mapping from device resource in physical space to a semantic description model in cyber space.

In order to generate the Cyber-D model, we adopt the ontology and knowledge engineering technologies to model device properties and abstract device service capabilities. The semantic modelling allows us to provide a unified and semantic information abstraction and description method to eliminate the heterogeneity of underlying device resources. The ontology and rule description languages, such as Web Ontology Language(OWL) and Semantic Web Rule Language(SWRL), can be used to describe device properties and represent domain knowledge about device sources. The device has various related properties information, for example, there are ID, location, type, function, ownership and state value for a temperature sensor. Figure 3 shows the ontology description model of a temperature sensor. In addition, the rule reasoning mechanism can be adopted into device ontologies to infer some implied information.

The essence of service capability abstraction is that it transforms the device resources into service capability resources, namely the device resources are described as the service components that can be invoked. The basic device service capability is built to be discovered and accessed by upper-level applications. The unified encapsulation of access and control protocols for different types of devices is made to form the unified device call interface. In order to facilitate the search and addressing of device service capabilities, the unique identifier based on Uniform Resource Identifier (URI) form are distributed for every device resources. For example, the URI identifier of a temperature sensor:

temperaturesensorID.location.deviceProviderID.platformID.

Device resource pool: The device resource pool is responsible for managing the Cyber-Ds, including their creation, evolution and termination. The programmable interfaces, including CREAT interface, DELETE interface, UPDATE interface, and QUERY interface, are defined and provided for developers to add, delete, update and query device model data. The Cyber-D models can be organized and managed according to the different industry or management domain in which they belong. When a new device is added into the system, the corresponding Cyber-D model data will be added into device resource pool. When the status of physical device change, the corresponding Cyber-D model paired with the Physical-D will be updated synchronously. When a device is deleted, the corresponding Cyber-D model data which include semantic description of its properties and related knowledge will be also deleted from device resource pool.

SDD controller: In the SDD architecture, there is a logically centralized controller which controls and manages the global device resources and meets on-demand segmentation and recombination. The controller is the core of SDD and enables intelligent device management and control. It lies between underlying devices and upper-level applications. Any interconnections and intercommunications between applications and devices have to go through the controller. The SDD controller also uses device discovery algorithms to search and discover the device resources that can meet application needs for various IoT applications.

Different from traditional middlewares, the SDD controller serves as a sort of operating system (OS) for underlying devices. By taking the control and management function off the hardware device and running it as software instead, the SDD controller facilitates automated device management and makes it easier to integrate and administer cross-industry and cross-platform IoT applications.

The SDD controller executes the device control by various programmable interfaces, mainly including general interfaces

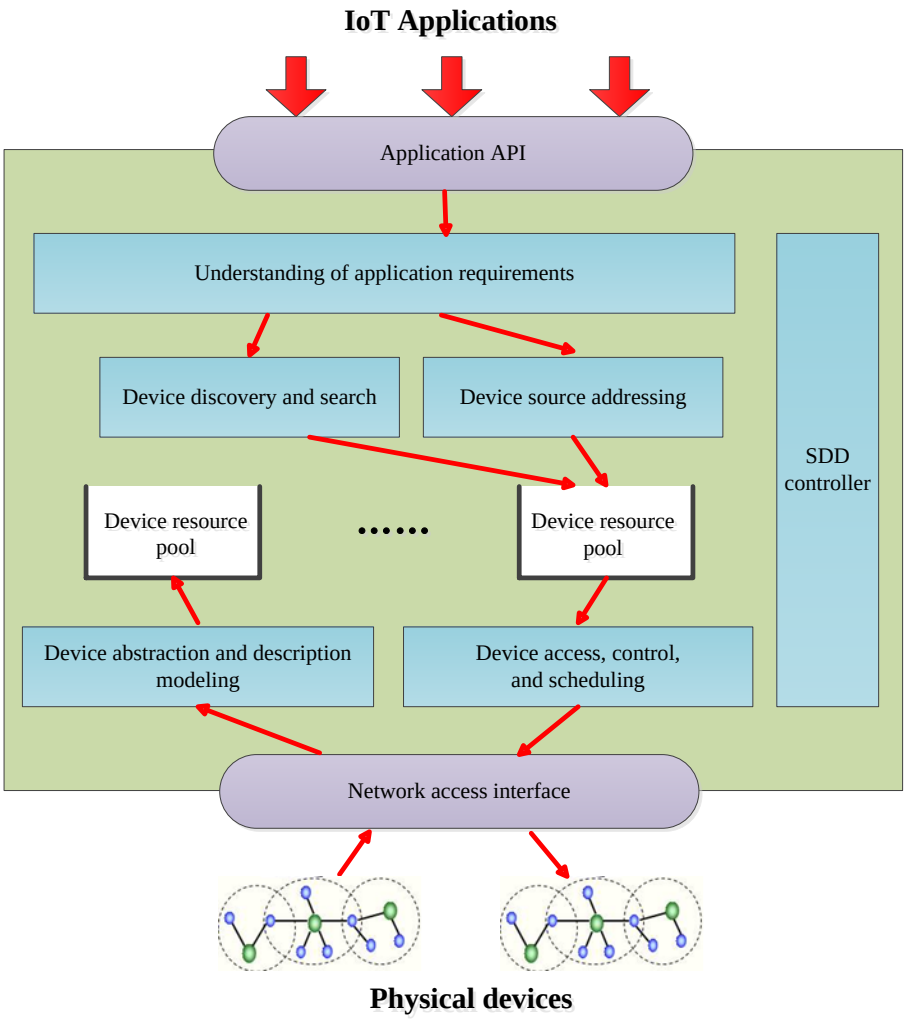

Fig. 4. The workflow of software definition mechanism.

and specific interfaces. And the methods are defined in the interfaces to implement various functions. The general interfaces can be used to implement the general functions for all devices. For example, we can define the two methods in *Device_General_Interface*:

- *turn_on()*: turning on the device.
- *turn_off()*: turning off the device.

The special interfaces are used to implement the special functions of a device and their implementation methods are also defined for a device. For example, in *Camera_Interface* for a camera, we can define these methods:

- *connect_internet()*: controlling the Internet connection of the camera.
- *up_down_rotation ()*: controlling the up and down rotation of the camera.
- *left_right_rotation()*: controlling the left and right rotation of the camera.
- *photograph_distance()*: controlling the photograph distance of the camera.

Device discovery services: The device discovery service module is responsible for matching and discovering appropriate device resources and service components for IoT applications requirements. It analyzes the requirements of application layer firstly, and maps the requirements of physical space into the standardized query requests expression of information space by query expansion and semantic understanding technologies. Then the corresponding Cyber-Ds and service components in device resource pool are matched with application requirements by the device discovery algorithm based on ontology-driven reasoning. Finally, application requirements can be implemented by controlling and scheduling the Physical-Ds which is corresponding to the discovered Cyber-Ds. Generally, a single device is difficult to meet complex application requirements, and they need the services combination of multiple devices.

Overall, the above four parts work together to implement the software definition mechanism of SDD. Figure 4 shows the workflow of the software definition mechanism, which includes the following five steps:

- The SDD platform abstracts physical device resources into the semantic description models by using the methods of device abstraction and semantic description.
- The virtualized resource description models are stored in-

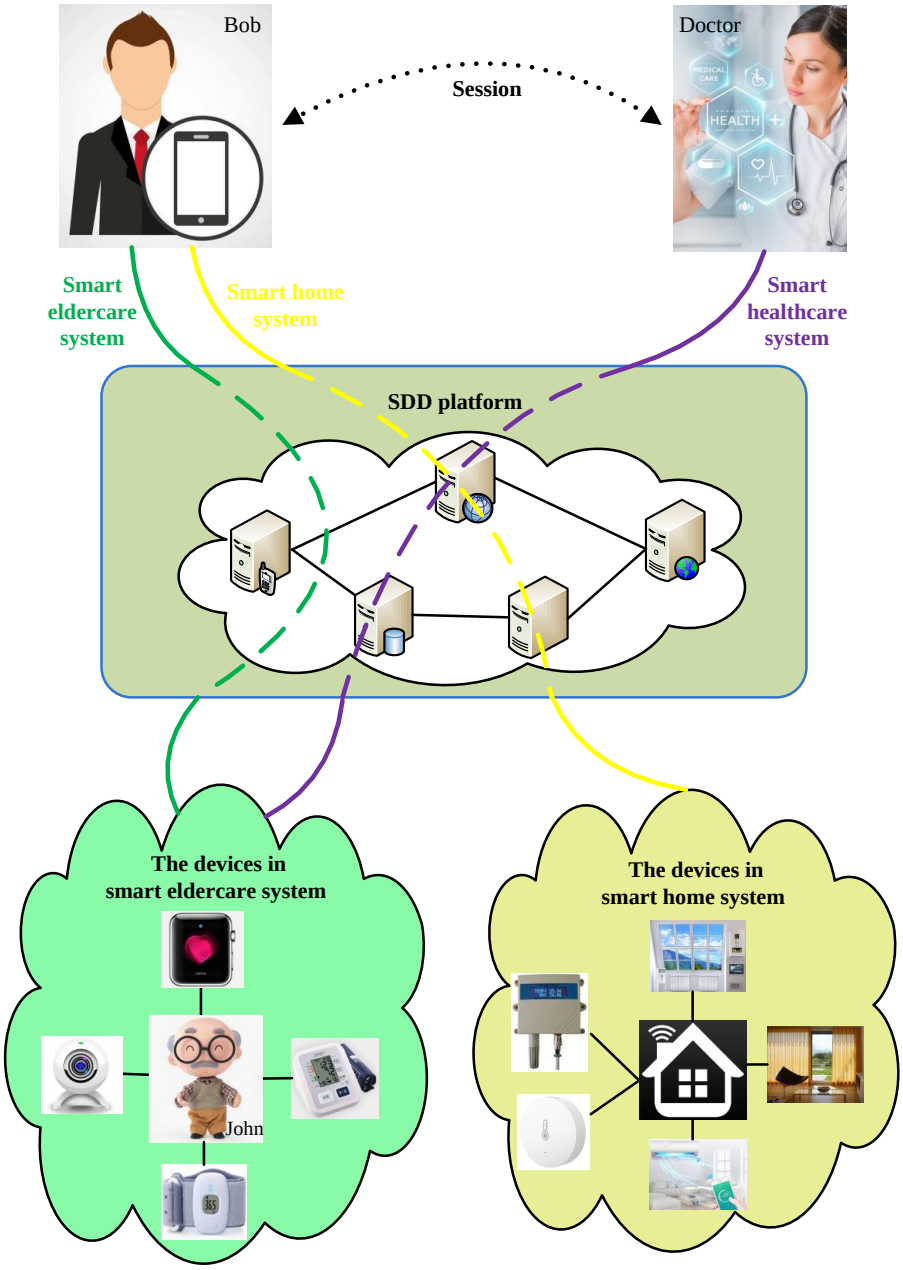

Fig. 5. The illustrative application use scenario.

to the device resource pool, and managed and maintained by device resource pool.

- The multiple IoT applications are developed and connected into the SDD platform by Application Programming Interfaces (APIs). The application requirements are transformed and understood into the standardized query requests expression of information space by semantic understanding technologies.
- The corresponding devices which can meet the application requirements are addressed and discovered from the device resource pool by the algorithms of discovery, search and addressing.
- The SDD controller accesses, controls and schedules the discovered underlying physical devices by programmable interfaces to meet the application requests.

IV. An Illustrative Application Use Scenario

In this section, we present a brief illustrative application use scenario to further explain the benefits of SDD in IoT applications. We take the combination of smart eldercare, smart home and smart healthcare application scenarios as an example to present how a sick older to obtain the intelligent life services. The illustrative application use scenario is shown in Figure 5.

The details of application scenario are as follows. John is an old man with inconvenient action. In order to assist his daily living, his son, Bob, installs a smart eldercare system and a smart home system based on the SDD platform in

John's home. The devices in smart eldercare system and smart home system are connected into SDD platform according to the scheme proposed above. These include some sensors which are camera, body temperature sensor, heart rate sensor, blood pressure sensor, room temperature sensor, and humidity sensor. Some actuators are also included, for example, curtain controller, window controller, and air conditioner controller. The control interfaces and corresponding methods of these devices are defined in SDD platform.

John had a fever three days ago and suffered from hypertension. Because Bob is on a business trip in another city, he gets John's physical status remotely through smart phone which connects with the SDD platform based smart eldercare system. Today, Bob asks John's personal doctor for a remote healthy reexamination by the SDD platform based smart healthcare system deployed in the hospital. The smart healthcare system can cross-domain request and call the sensing devices in smart eldercare system through the SDD platform, rather than redeploy these devices. Through calling the body temperature sensor, blood pressure sensor, and heart rate sensor, the doctor remotely acquires John's physiological indexes of body temperature, blood pressure and heart rate. And the mental state is observed by the camera. After finishing reexamination, the doctor tells Bob that his father has recovered and blood pressure is also normal, and suggests that the room should keep ventilation and the ambient temperature should not be too low. Then Bob uses smart phone to access to the SDD platform based smart home system which is deployed in John's house. He remotely checks the room's environmental status with the temperature sensor and humidity sensor. Since the temperature and humidity are low, through the corresponding control interfaces in SDD platform, Bob remotely handles the curtain controller, window controller, and air conditioner controller with his smart phone to open the curtain and window for ventilating and turn the room temperature up. By adopting these smart systems integrated into the SDD platform, the older can live more conveniently.

In this application scenario, all the operations can be implemented by the SDD platform, and all the underlying sensors and actuators can be unitedly shared, reused and scheduled in cross-domain scenarios. The involved smart eldercare system, smart healthcare system and smart home system can be interconnected and intercommunicated.

V. Conclusion and Future Work

The concept of SDD offers a new IoT computing paradigm which can hide the management complexity of hardware devices by abstracting the control and management functions of a underling device as a software in an independent SDD layer. In this article, we have elaborated the SDD concept and described its operation-level rationale, namely the cyber-physical SDD mapping processes. Based on the SDD paradigm, we have designed an open IoT system architecture, and introduced its software definition mechanism in detail. We have also introduced an illustrative application use scenario to present the practical effectiveness of proposed SDD paradigm.

The SDD-based open IoT system architecture implement the unified management and scheduling, sharing, reusing, recombining, and modular customization of underlying IoT device resources. This helps solve the problem of interconnection and intercommunication of heterogeneous resources, and improve the utilization efficiency of device resources. The proposed scheme has the following advantages. (1) The SDD platform can decouple upper-level applications from the underlying physical devices, so that the programmers need not worry about the details of underlying physical devices, and makes full use of the programmability of hardware; (2) The architecture allows rapid development of customizable and personalized IoT applications which involve multiple physical IoT resources; (3) With the help of software definition mechanism, users can combine these virtualized devices to create their own virtualized devices with richer functions and/or features, providing a flexible scheme that can adapt to many different users' needs.

In the future, we will enter an era of software definition. The basic characteristics are that all things can be interconnected, and all can be programmed. The SDD is a significant and indispensable part of it. There are still open issues which need to be explored and further in-depth research. Specifically, (1) device abstraction and description modelling based on the semantic ontology and knowledge engineering technologies will be emphatically considered to downward model device properties and abstract device service capability; (2) device discovery and search algorithms based on semantic matching methods with ontology-driven reasoning mechanism will be researched to upward provide corresponding device resources and service components for various IoT applications requirements; (3) privacy protection and security schemes, for example, identity verification, access control, device model data encryption, authentication, authorization, and accounting (AAA), for underlying device resource and SDD platform will also be designed to provide secure and reliable SDD-based IoT services.

Acknowledgment

This work is supported by the National Natural Science Foundation of China under Grant 61471035.

References

[1] A. Al-Fuqaha, M. Guizani, M. Mohammadi, M. Aledhari, and M. Ayyash, "Internet of things: A survey on enabling technologies, protocols, and applications," *IEEE Communications Surveys & Tutorials*, vol. 17, no. 4, pp. 2347–2376, 2015.

[2] T. Baker, M. Asim, H. Tawfik, B. Aldawsari, and R. Buyya, "An energy-aware service composition algorithm for multiple cloud-based iot applications," *Journal of Network and Computer Applications*, vol. 89, pp. 96–108, 2017.

[3] I. Yaqoob, E. Ahmed, I. A. T. Hashem, A. I. A. Ahmed, A. Gani, M. Imran, and M. Guizani, "Internet of things architecture: Recent advances, taxonomy, requirements, and open challenges," *IEEE Wireless Communications*, vol. 24, no. 3, pp. 10–16, 2017.

[4] Z. Song, A. A. Cardenas, and R. Masuoka, "Semantic middleware for the internet of things," in *2010 Internet of Things (IOT)*, 2014, pp. 1–8.

[5] M. A. Razzaque, M. Milojevic-Jevric, A. Palade, and S. Clarke, "Middleware for internet of things: A survey," *IEEE Internet of Things Journal*, vol. 3, no. 1, pp. 70–95, 2016.

[6] G. Faraci and A. Lombardo, "An nfv approach to share home multimedia devices," in *2017 IEEE Conference on Network Softwarization (NetSoft)*, 2017, pp. 1–6.

[7] H. Kim and N. Feamster, "Improving network management with software defined networking," *IEEE Communications Magazine*, vol. 51, no. 2, pp. 114–119, 2013.
[8] Y. Jararweh, M. Al-Ayyoub, A. Darabseh, E. Benkhelifa, M. Vouk, and A. Rindos, "Sdiot: a software defined based internet of things framework," *Journal of Ambient Intelligence & Humanized Computing*, vol. 6, no. 4, pp. 453–461, 2015.
[9] D. Kreutz, F. M. V. Ramos, P. E. Verłssimo, C. E. Rothenberg, S. Azodolmolky, and S. Uhlig, "Software-defined networking: A comprehensive survey," *Proceedings of the IEEE*, vol. 103, no. 1, pp. 14–76, 2015.
[10] J. Liu, Y. Li, M. Chen, W. Dong, and D. Jin, "Software-defined internet of things for smart urban sensing," *IEEE Communications Magazine*, vol. 53, no. 9, pp. 55–63, 2015.
[11] F. Hu, Q. Hao, and K. Bao, "A survey on software-defined network and openflow: From concept to implementation," *IEEE Communications Surveys & Tutorials*, vol. 16, no. 4, pp. 2181–2206, 2014.
[12] H. Ning, H. Liu, J. Ma, L. T. Yang, and R. Huang, "Cybermatics: Cyber-physical-social-thinking hyperspace based science and technology," *Future Generation Computer Systems*, vol. 56, pp. 504–522, 2016.
[13] J. Ma, J. Wen, R. Huang, and B. Huang, "Cyber-individual meets brain informatics," *IEEE Intelligent Systems*, vol. 26, no. 5, pp. 30–37, 2011.
[14] J. Wen, K. Ming, F. Wang, B. Huang, and J. Ma, "Cyber-i: Vision of the individual's counterpart on cyberspace," in *Eighth IEEE International Conference on Dependable, Autonomic and Secure Computing*, 2010, pp. 295–302.
[15] C. J. Bernardos, A. d. l. Oliva, P. Serrano, A. Banchs, L. M. Contreras, H. Jin, and J. C. Zuniga, "An architecture for software defined wireless networking," *IEEE Wireless Communications*, vol. 21, no. 3, pp. 52–61, 2014.

Pengfei Hu [S'16] (hupf7721@126.com) received the B.E. degree from the Zhengzhou University of Aeronautics, China. He is currently working toward the Ph.D. degree from the School of Computer and Communication Engineering, University of Science and Technology Beijing, China. He focuses on the objects modelling in cyber-physical space convergence and Internet of Things. His research interests include device virtualization and modelling, cyber-physical modelling, identification and resolution of physical objects, and Internet of Things.

Huansheng Ning [M'10-SM'13] (ninghuansheng@ustb.edu.cn) received the B.S. degree in radio technology from Anhui University, Hefei, China, in 1996 and the Ph.D. degree in communication engineering from Beihang University, Beijing, China, in 2001. He is currently a professor and vice dean of the School of Computer and Communication Engineering, University of Science and Technology Beijing, Beijing, China. His current research interests include the Internet of Things and cyber-physical modelling. He is the founder of Cybermatics and Cyberspace International Science and Technology Cooperation Base.

Liming Chen [M'10-SM'17] (liming.chen@dmu.ac.uk) received the B.E. and M.E. degrees from the Beijing Institute of Technology, Beijing, China, and the Ph.D. degree in artificial intelligence from De Montfort University, Leicester, U.K. He is currently a professor of computer science and the head of the Context, Intelligence and Interaction Research Group with the School of Computer Science and Informatics, De Montfort University. His research interests include pattern recognition, intelligent systems, smart environment, and assisted living.