MicroThings: A Generic IoT Architecture for Flexible Data Aggregation and Scalable Service Cooperation

Yulong Shen, Tao Zhang, Yongzhi Wang, Hua Wang, and Xiaohong Jiang

Inspired by the Internet architecture over which divergent devices can easily be accessed and a considerable number of applications can be run, the authors propose a generic architecture for IoT. This architecture supports two DIYs: network DIY for data aggregation and application DIY for service cooperation. To connect these two DIYs, a centralized controller has been designed to provide standardized interfaces for data acquisition, organization, and storage, and to support elastic and supportive computing.

ABSTRACT

The Internet of Things has been widely deployed in various areas of daily life through heterogeneous communications protocols. Each unstandardized protocol focuses on a specific IoT communication pattern. Inspired by the Internet architecture over which divergent devices can easily be accessed and a considerable number of applications can be run, we propose a generic architecture for IoT. This architecture supports two DIY areas: network DIY for data aggregation and application DIY for service cooperation. To connect these two DIYs, a centralized controller has been designed to provide standardized interfaces for data acquisition, organization, and storage, and to support elastic and supportive computing. With these properties, divergent devices can coexist in a uniform microworld, and rich services can be developed and provided on demand to interoperate with physical devices. This article discusses the background, design principles, and advantages of the proposed architecture, as well as open problems and our initial solution, which substantiates a novel IoT architecture and new research ground.

INTRODUCTION

Currently, the Internet of Things (IoT) has been widely adopted in various crucial systems such as city sensing, highway transportation, smart communities, and green farming. IoT aims to enable data exchange and smart communication among everyday objects, from watches, cookers, and bicycles, even to humans, plants, and animals [1]. Things can see, hear, and perceive the real-world environment to achieve more comfortable and safer living conditions. For instance, London has deployed all sorts of sensors to improve urban services: traffic prediction, weather forecasts, waste management, and water quality monitoring [2]. In addition to dedicated IoT platforms, mobile devices (e.g., smartphones, wearables, and vehicles) are also utilized as sensing resources.

Sensors in different IoT systems communicate through divergent protocols and standards. These protocols include Message Queuing Telemetry Transport (MQTT), Extensible Messaging and Presence Protocol (XMPP), Constrained Application Protocol (CoAP), Data Distribution Service (DDS), Advanced Message Queuing Protocol (AMQT), and others. The lack of ubiquitous sensor access has resulted in a fragmented IoT ecosystem, which brings about the following two problems. On one hand, IoT services are built on top of application-level message protocols. The design, deployment, and adoption of vendor-dependent IoT elements (sensors, acurators, gateways, etc.) is customized with regard to specific application requirements, making them difficult for an amateur to access or operate, let alone to organize them for customer-oriented application. On the other hand, as devices are controlled by dedicated platforms, the system logics tend to be ossified and hardly meet requirement changes. The differences from one platform to another lead to data disunity and redundancy, which also make it difficult to utilize data from different platforms.

The development of the IoT system is still in its infancy. However, the Internet has exerted significant influence on our lifestyles for a long time. From the hardware perspective, devices are connected to the Internet through unified interfaces, that is, we can communicate with cyberspace via an inexpensive home router after some configurations in wizard webpages. From the software perspective, rich and varied applications flourish on the Internet to satisfy consumers' specific tastes. Service-oriented architecture (SOA) and HTML5 further enrich the web-based application resources through a standardized interface. The design of future IoT should be flexible enough to satisfy the requirements of complex networks and various applications.

There is no doubt that it is necessary to build an adaptive and scalable architecture for sustainable IoT developments, just as the Internet provides today. Some platforms, such as the Eclipse IoT project, Intel IoT framework, and Baidu IoT framework, have been proposed to enable heterogeneous integration with different communication technologies and application protocols. These solutions mainly focus on the design of enhanced protocols or smart gateways for the conversion of multiple baseline protocols. However, as they are strongly coupled with the supported protocols, different platforms may not be compatible with each other. In addition, users have to be familiar with the specifications of each platform, leading to a steep learning curve for the non-expert.

Digital Object Identifier:
10.1109/MCOM.2017.1700104

Yulong Shen, Tao Zhang and Yongzhi Wang are with Xidian University; Hua Wang is with Victoria University; Xiaohong Jiang is with Future University Hakodate.

0163-6804/17/$25.00 © 2017 IEEE

Within the contextual background, this article proposes a generic IoT architecture, called *MicroThings*, to glue all the "Things" fragments into a uniform microworld. MicroThings integrates the application environment and the information aggregation environment with a logically centralized controller. Each of the two environments supports heterogeneous cooperation with a set of uniform interfaces. The controller connects the two environments and allows interoperation among applications and sensing devices.

Figure 1 illustrates the MicroThings architecture. It implements do-it-yourself abilities (DIYs) at two levels. One is the network DIY, which is at the bottom of MicroThings, where multiple sensing devices are accessed via compatible forwarding devices. Consumers are able to customize their IoT networks for data acquisition, transmission, and interaction. The other is the application DIY, which sits on top of the MicroThings, where standardized application programming interfaces (APIs) are provided for the development, deployment, and provision of new applications, as well as the reuse and composition of existing applications. These characteristics empower MicroThings to ubiquitously access various devices and greatly enrich service resources.

In the rest of this article, we first analyze the traditional three-layer IoT architecture, along with the design directions for existing architecture. We then introduce the proposed MicroThings architecture and potential solutions. Case studies are provided to show the benefits of MicroThings. Finally, we conclude the article.

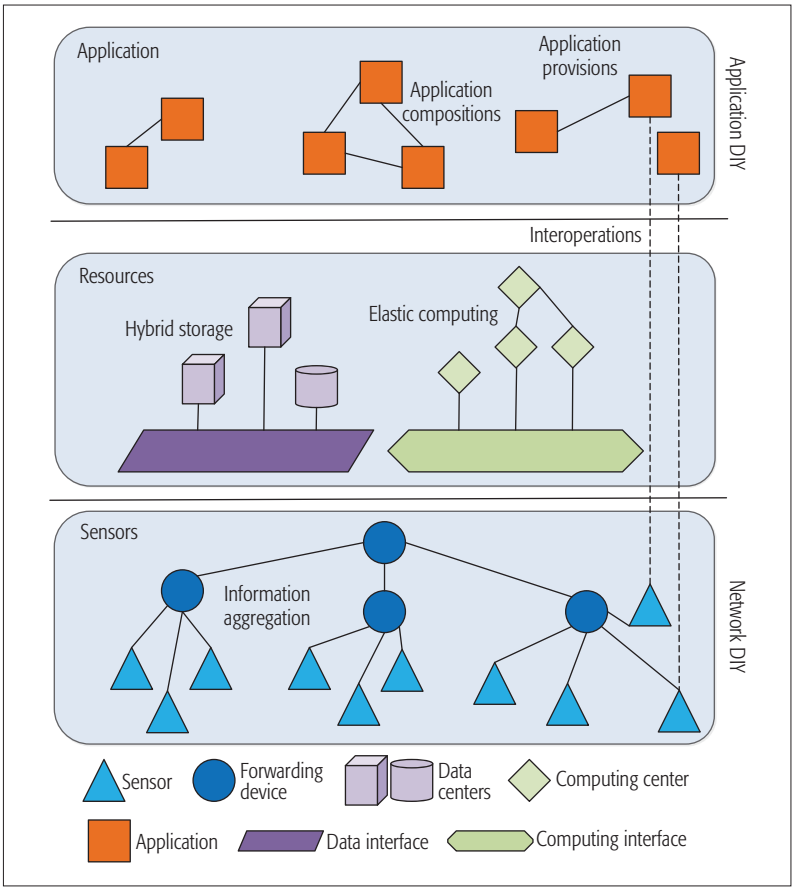

Figure 1. Illustration of the proposed architecture.

IoT: Challenges and Opportunities

State of the Art

Figure 2 illustrates the traditional IoT architecture. This architecture logically consists of three layers: the perception layer, the network layer, and the application layer. Various sensing devices are deployed in the perception layer to collect and aggregate data from the physical world (e.g., temperature and humidity). The network layer completes a wide range of information transmission and exchange on the converged network systems such as cellular networks — second/third/fourth generation (2G/3G/4G), narrowband IoT (NB-IoT), and so on — short-range communication networks (Bluetooth, Wifi, Zigbee, etc.), long-range wireless networks (LoRa, openRF, etc.), and remote communication networks (satellite). The collected data are transmitted to data centers through the network layer for further storage and processing. The application layer processes the collected data and interacts with people for further analysis and decision making. This three-layer architecture has been deployed in different areas such as intelligent homes, smart cities, public security, and green farming.

The general approach to developing IoT systems is to synthetically design these three layers with respect to specific application requirements. More specifically, engineers first select proper sensors to collect required data from the physical world, then design gateways and routers to transmit the data to local or remote data centers, and finally gather data to design various domain-specific applications.

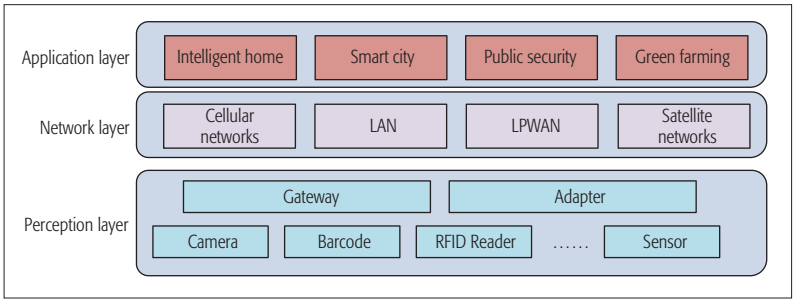

Figure 2. Three-layer Internet of Things architecture.

Problems and Directions

While the domain-specific development of the above three-layer IoT architecture seems quite straightforward, it has several problems that hinder sustainable development. We summarize these issues as follows.

Difficulty with IoT Device Networking: Through split-level developments, the perception layer manages the access and data transmission of sensing devices. However, as devices are closely relevant to the sensing data, frequent reconfigurations of the perception and network layers are needed when the collecting devices change. For example, the improvement in the accuracy of air quality monitoring needs redeployment and reconfiguration of more air sensors. In addition, with the coexistence of different physical interfaces and communication protocols, it is impractical for an ordinary consumer to organize the devices, or handle the heterogeneous security issues [3].

In addition to ubiquitous access, there is the trend toward network automation. It should provide a configurable and programmable interface with which people will need less expertise to be involved in modern intelligent IoT activities.

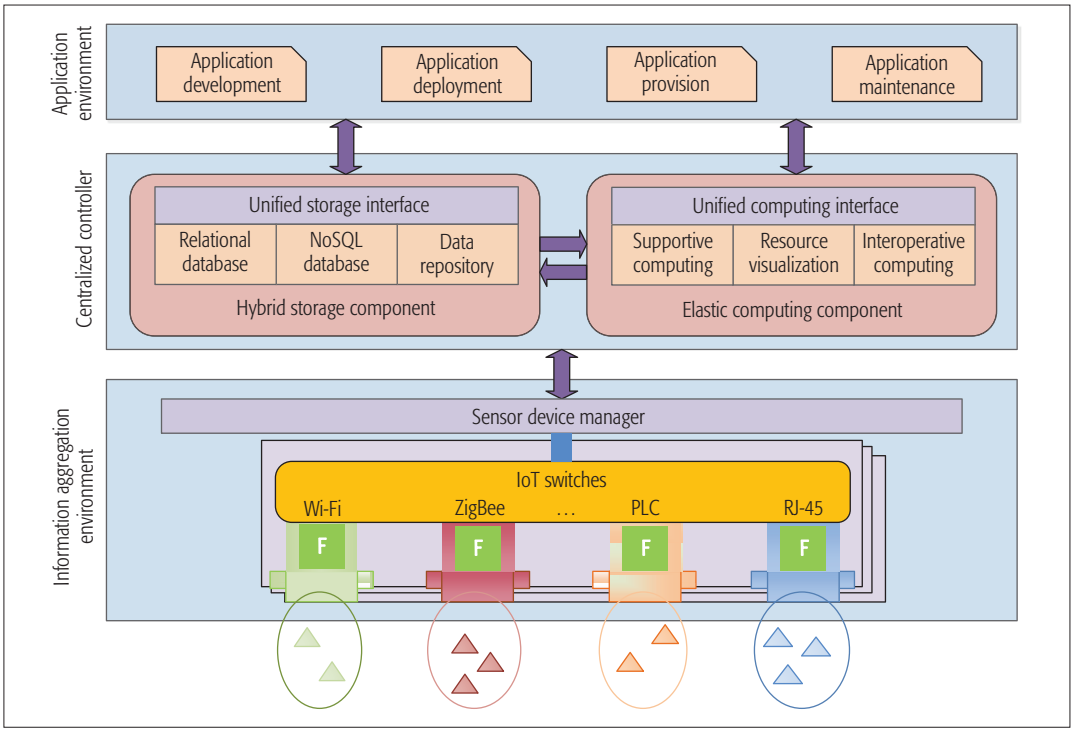

Figure 3. MicroThings architecture.

High Data Storage and Computing Cost: We should build different storage and computing frameworks for different types of data, which means high investment in the IoT application infrastructures. For instance, the system should establish file repositories and NoSQL databases to store unstructured data like real-time videos, and relational databases to store structured data. Even for applications of similar functionality from different stakeholders, the software, hardware, power, and control systems must be deployed independently, leading to a high cost for each IoT system.

Inflexible Application Provisions: The control logics of applications are closely coupled with the platform configurations such as the perceptive sensors, the network structures, and the storage/computing framework. In addition, the problem of broadly reusable applications remains unsolved. For example, existing applications cannot be directly migrated to the new environment without an application market. It requires application re-development or re-configuration for some, or even similar, IoT scenarios, which has led to overlapping investments.

With the development of network technology and the increasing number of applications, industry and academia need to establish a generic architecture that can address the main issues mentioned above. There are some trends apparent in the design of IoT architecture, which we summarize below.

Automation of the ubiquitous network: As the capillary ends, sensors are the data sources that supply blood to the whole IoT system. Since multiple sensing devices provided by different manufacturers have coexisted in the IoT system, the design of new architecture should support the ubiquitous access that is compatible with these different communication protocols and standards. For example, C. Hou *et al.* proposed profile-based access for device integration with IoT-Cloud, which groups the sensors by category and integrates the sensors by the category profile to the IoT-Cloud [4].

In addition to ubiquitous access, there is the trend toward network automation. It should provide a configurable and programmable interface with which people will need less expertise to be involved in modern intelligent IoT activities. For instance, A. Al-Fuquha *et al.* proposed a rule-based gateway for device organizations with divergent protocols which also guaranteed the quality of service (QoS) properties in the IoT systems [5].

The blend of flexible data storage and elastic computing: Flexible data storage provides the ability to properly manage the collected data from multiple data sources, and elastic computing further enhances the ability to process the massive amount data [6–8]. The blend of data storage and computing creates a uniform core that can control the IoT logics. As the linking element between the physical world and the cyber world, this core provides a resource pool with all kinds of operations that can help hide the complexity and the heterogeneity of underlying infrastructures. This trend suggests that IoT architecture should build a control core for interoperation between the southbound devices and the northbound applications.

The reuse of application programming and composition interface means that the northbound environment supports the programming, provision, and sharing of applications to different stakeholders in a unified market [9]. Also, the application environment should be a platform-independent model that can customize and consume existing services to construct composite ones that facilitate reuse, just as SOA) provides.

As the reuse of applications can significantly reduce the development and maintenance costs, it is believed that the scalable service resources should be one of the built-in characteristics of modern IoT architecture.

There are many standardization opportunities that could be implemented to make it easier for service providers and consumers to work with an IoT ecosystem. From the perspective of the IPSO Alliance, the IoT is a system-of-systems, and each system has its own standards. Therefore, supporting interoperability of different systems is still the major hurdle to achieving massive IoT adoption. Standardization organizations such as the International Telecommunication Union Telecommunication Standardization Sector (ITU-T), Third Generation Partnership Project (3GPP), and International Organization for Standardization/International Electrotechnial Commission (ISO/IEC) are leading the IoT standardization progress. As an ongoing standard specification, oneM2M (M2M: machine-to-machine) is devleloping a semantics enabler to bridge the gap between current IoT resources. As well, SmartM2M and Light-Weight M2M (LWM2M) have been developed to provide network-independent services. In addition to the aforementioned standards, some projects such as Eclipse IoT projects, the Intel IoT platform, the Baidu IoT platform, and the Tencent IoT platform are also contributing their efforts to provide uniform open APIs to simplify application development over IoT devices. The general direction is to design a cloud-based architecture to enable intelligent data acquisition and analysis through integrated protocols and standards, and bring uniform access while supporting different interactions between cloud and smart devices. However, these platforms are still excessively complicated for the non-expert, and it is necessary to build a generic architecture with lower entry barrier.

MicroThings: Architecture Overview

In this article, we propose a generic IoT architecture, referred to as MicroThings, which combines the fragmented devices, networks, and applications into a micro IoT world. As illustrated in Fig. 3, the MicroThings consists of two environments connected with a centralized controller.

Information Aggregation Environment: This environment supports the first DIY for operating physical devices. The environment aggregates data from the packages of devices, including the resource-constraint devices and the resource-rich equipment, and multiple multi-mode switches. Each switch integrates a variety of wired and wireless communication interfaces to control the transmission patterns among IoT devices. For prevalent protocols, like MQT and, AMQP, the architecture employs a conversion scheme to address the interoperation issues. In addition, to support more ubiquitous device access, this environment also establishes a JSON encoded HTTP parser to support the extensive accesses of the non-TCP/IP devices and scalable protocols. Thus, consumers can focus only on the connecting devices without worrying about the design specifications.

Moreover, the environment establishes a manager to provide a registration mechanism for the multi-domain connected devices. The manager plays the roles of both virtualized gateway and pattern recommender. The "things" can join or leave MicroThings for public or private use. Thus, MicroThings can recommend suitable applications to involved devices when they talk to each other, for example, recommending smart home applications for air quality devices. On the other hand, devices can be shared on MicroThings for public use, which greatly raises resource utilization ratio and reduces the deployment cost.

Centralized Controller Environment: This environment plays the intermediary role between the information aggregation environment and the application environment. The controller consists of two parts: the hybrid storage component and the elastic computing component. The storage component provides standardized APIs to gather the aggregated data from the southbound environment, and then stores them by category in the corresponding databases. The computing component extracts required information from the storage component, and then prepares the computing resources for further processing. The elasticity ensures that the computing resources can be dynamically allocated to coordinate with the dynamic changes across the entire IoT architecture.

Application Environment: This environment supports the second DIY for the IoT control logics, where applications can be developed with respect to the published requirements. As the underlying differences are hidden from the centralized controller, this environment creates a unified service ecosystem that supports application provision, development, deployment, and maintenance.

Moreover, this environment supports the reuse of applications, where different stakeholders can share the provided applications. Besides, for complex control logics, existing applications can be combined to rapidly create value-added functionalities.

The computing component extracts required information from the storage component, and then prepares the computing resources for further processing. The elasticity ensures that the computing resources can be dynamically allocated to coordinate with the dynamic changes across the entire IoT architecture.

Design Principles and Implementation Solutions

The previous section presented a generic architecture, MicroThings, which supports two kinds of DIYs coordinated with a centralized control. In this section, we illustrate how such architecture has been instantiated. For this purpose, a set of design principles has been proposed as the building blocks for the architecture, and the implementation solutions are given.

Information Aggregation Environment Design

Principle 1: Enabling ubiquitous and on-demand network access for IoT devices: The information aggregation environment lies in the southbound direction of the MicroThings architecture, which is designed to directly interact with the physical sensing devices. On the traditional Internet, communication devices are accessed via a set of uniform standards, such as Ethernet, to transmit information from one point to another. Compared to the traditional Internet, the design of an information aggregation environment for the IoT is much more difficult because of the heterogeneity of devices. Besides, the data aggregated in MicroThings should be identifiable, manageable, and controllable. As devices are abstracted in the

The addressable sensors can directly communicate to our MicroThings switches via the manufactured gateway. Other server-oriented communications, like satellite and wired communications, can be plugged into our architecture for data storage and process. Moreover, different IoT switches can be further placed at different locations for large-scale deployments.

data layer, end users can achieve on-demand network DIY in the MicroThings architecture.

As an initial solution to the above problems, we first designed IoT switches for data device access. An IoT switch is a multi-mode switch consisting of a set of front-end ports and a back-end port. The front-end port supports different communication protocols and standards, including Wi-Fi, ZigBee, PLC, RJ-45, and so on, while the back-end port connects to the sensor manager. To achieve the first objective, an IoT switch is deployed to access the local heterogeneous participant sensors. Therefore, the addressable sensors can directly communicate to our MicroThings switches via the manufactured gateway. Other server-oriented communications, like satellite and wired communications, can be plugged into our architecture for data storage and processing. Moreover, different IoT switches can be further placed at different locations for large-scale deployments.

For the second objective, a device manager is also used for the automatic management of IoT switches. The information that the manager receives contains the identifications and statuses of the sensors. Thus, sensors are registered automatically to provide abstractions when they are active. The centralized controller also uses this information for the abstraction of the data rather than the physical devices.

Centralized Controller Environment Design

The centralized controller is the connection between the southbound environment and the northbound environment. The design of the centralized controller should achieve two objectives: hybrid storage and elastic computing.

Principle 2: Provide a unified framework for heterogeneous data storage and management: The storage component is designed to interact with the southbound information aggregation environment. As the volume of the aggregated data increases rapidly, the data storage component should store the massive data with a high throughput capacity. Data are collected from multiple sources with different structures, and then the storage component categorizes and organizes them in a transparent way [10].

Jiang *et al.* have introduced a solution for data storage by combining multiple databases [11]. They have also built some mapping schemes for database operations. By providing standardized APIs, data can be stored by category automatically. In addition, since the information aggregation environment provides identifiable information for stored data, we propose a method to decide where the data will be stored and when the results will be computed.

Principle 3: Provide elasticity to enable supportive computing in the converged networks: The computing component is designed to interact with the northbound application environment. The design of components should support elasticity to enhance the ability for different heterogeneous data and applications [12, 13].

To achieve the objectives, MicroThings supports the following three features. First, the visualization of computing resources provides computing scalability, which in turn supports the interoperations between the northbound and southbound environments. Second, as IoT is a converged computing platform that supports the next generation communication technology, that is, the fifth generation (5G) mobile network, the edge devices with increasing computing resources can also contribute their capabilities for supportive computing. Finally, the visualization resources can be scheduled in a flexible and interoperative way, which further allows more fine-grained control between services and devices.

Application Environment Design

Principle 4: Provide a whole ecosystem for IoT application development, deployment, and provision: The application environment lies in the northbound of the MicroThings architecture, which is designed to provide applications for multiple stakeholders, such as device providers, software providers, and storage providers. As more and more devices are connected to MicroThings, the application environment should support the whole application production process, including development, deployment, provision, and maintenance [14].

SOA is the most commonly used for service provision via XML-based standards. Constructing an ecosystem for service-oriented application is a general approach to achieve these goals. On one hand, the ecosystem provides all applications with standard interfaces, with which the application can control device logics and interact with other applications. On the other hand, multiple stakeholders can share the developed applications. They can publish requirements to the ecosystem and obtain applications in a pay-for-use manner. Furthermore, the ecosystem supports value-added application provision with model-driven development (MDD) from the composition of existing applications, creating a resource pool of rich and varied applications.

Case Study and Validation

Selected Scenario

To further illustrate the advantages of our MicroThings architecture, we conduct a case study and its analysis in this section. Figure 4 illustrates the selected scenario in which four practical IoT systems are deployed in our MicroThings architecture: highway transportation, safe city, smart community, and green farming.

In the above scenario, these four systems execute the same processing flow. First, sensors are connected to the IoT switches for data collection. Then applications are deployed to process the control logics of the sensors. Finally, the central controller connects the above two parts to automatically control data collection and effectively provide computing resources.

Architecture Validation

Information Aggregation: We consider the deployment of these systems to illustrate the network DIY. The safe city system reflects the need for city sensing. It deploys sensors to observe street views, weather, noise, air condition, and so on. The aggregation data may vary with the sensors' interfaces and functionalities. Consider the following two cases for sensing data aggregation. One is the sensor deployment in a new area. For instance, a modernizing city usually deploys many

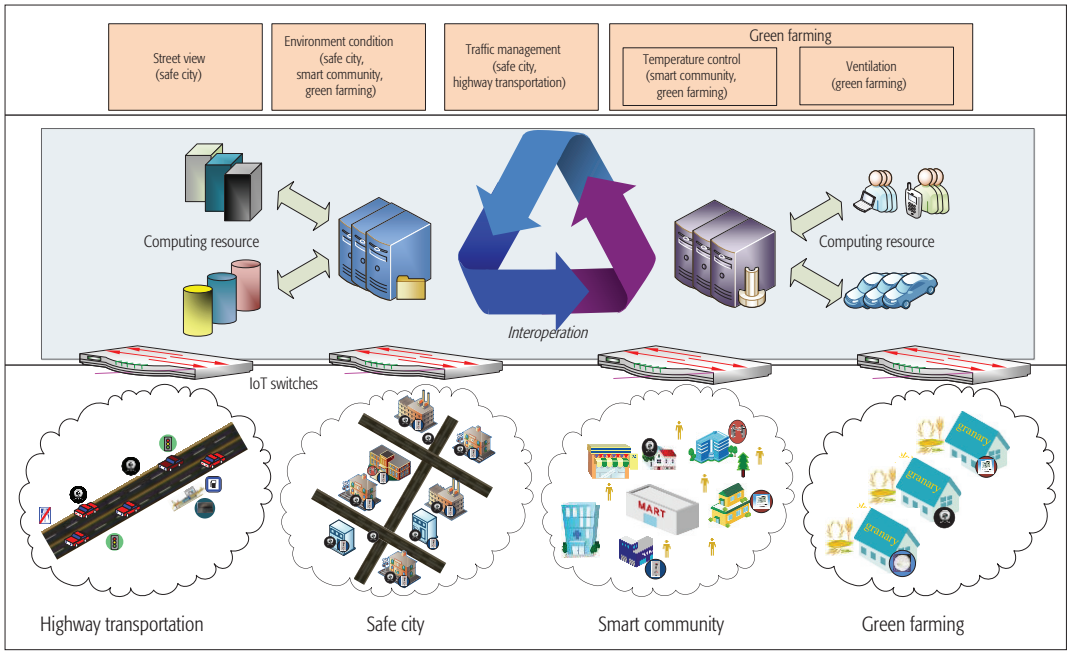

Figure 4. Implementation of multiple IoT systems with MicroThings.

The applications for environmental monitoring can be shared by multiple systems, such as safe city, smart community and green farming. In addition, the applications can be composed together using drag-and-drop way to provide value-added and various applications. MicroThings checks the input-output results of some typical patterns for each composition.

cameras for a street view. Cameras from different vendors can be connected to our IoT switches for data aggregation. Our environment provides a wizard to register the cameras, including the manufacturer, identification, sampling frequency, and so on. If the devices are supported, our MicroThings automatically translates the protocols of these devices and lists the possible communication patterns for the registered devices. The aggregated data will be further processed by the administrator for the city views. Otherwise, for unsupported devices, more configuration information to support JSON-formatted data transmission is required. The other case is sensor maintenance and replacement. For environmental condition monitoring applications, the sensors sometimes need to be replaced for accuracy improvement, and some of the sensors can easily be added or removed only if they are supported by the compatible protocols from the IoT switches.

We then consider another system for highway transportation, where some of the sensors are compatible with sensors in the safe city. The highway transportation shares the sensors to provide dynamic traffic information to enhance the safety of the neighboring city. Thus, the data can be aggregated together to share with other systems.

Hybrid Data Storage: We consider the green farming system. In this case, the environmental data (CO_2, NO_2, temperature) and video data should be stored. The environmental data are stored in structured databases as they are composed of some determined items such as time, value, and location. However, the video data are massive and unstructured. We store them in the NoSQL database. The storage provides standard interfaces to connect with the information aggregation environment and automatically categorizes the environmental and video data. It also supports locating data for further computing such as cooling the granary when the temperature is rising.

Elastic Computing: With the development of computation capability, CPU-controlled devices can contribute their power for computation. Through visualization, intelligent vehicles and handheld cell phones can support computing provision. For instance, sensors on cell phones can perceive temperature and noise data to cooperatively compute the local comfort degree. Similarly, vehicles can also compute the results of traffic flow for various IoT applications.

In addition, the computing component supports interactions between the physical and cyber worlds. Consider the smart community system, where deployed chillers can be automatically adjusted to a comfortable temperature, and the sensors send monitored data to the application for controlling the behavior of the chiller. The adjustment supports interoperation between chillers and control applications. For example, the application can activate more sensors when residents are quite sensitive to the change of the temperature, or the sensors can send control information to reduce the monitoring frequency if the temperature stays constant for a long time.

Application Life Cycle: The application environment supports the whole application life cycle including application development, deployment, and provision. It also supports some advanced features, such as application reuse and composition.

Since environmental monitoring is the most common IoT application, it can be shared by different platforms. In other words, the applications for environmental monitoring can be shared by multiple systems, such as safe city, smart community, and green farming. In addition, the applications can be composed together in a drag-and-drop way to provide value-added and various applications. MicroThings checks the input-output results of some typical patterns for each composition. For instance, a grain preservation application is provided in green farming by composing environment monitoring (for the level of O_2) service and ventilation service.

From these scenarios, we can conclude that

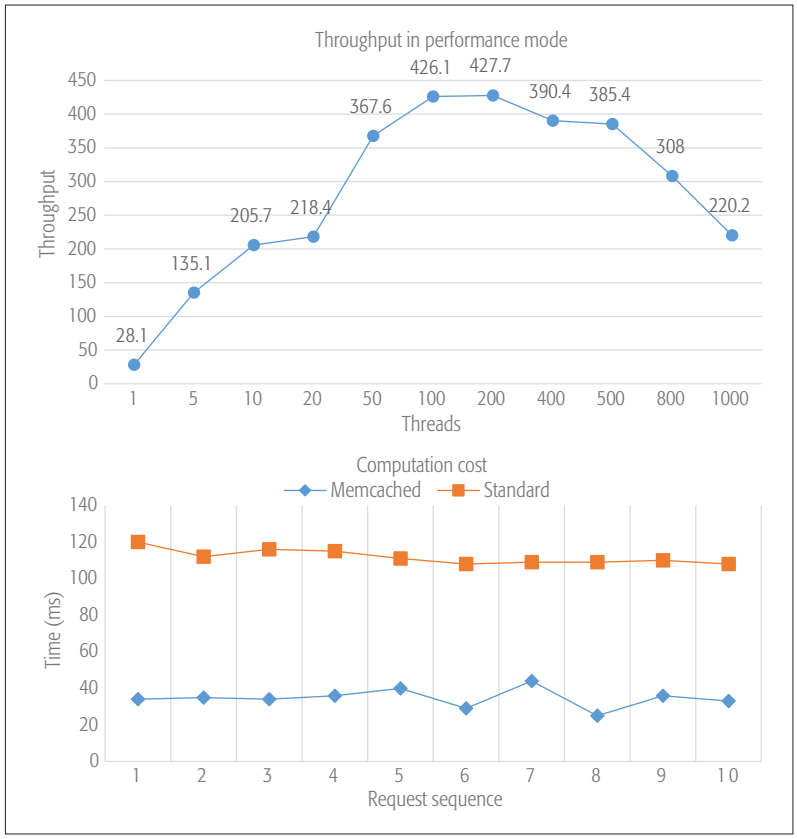

Figure 5. Performance evaluation of MicroThings: a) throughout in performance mode to support elastic computation; b) computation cost for hybrid storage.

the most intuitive advantages of our proposed architecture contain the following three aspects:

1. A unified MicroThings architecture can support multiple IoT systems.
2. Multiple novel systems can be deployed rapidly from existing systems to adapt to different environments.
3. Multiple stakeholders can share the applications run on MicroThings.

Performance Evaluation

We have performed simulation experiments to evaluate the performance of our proposed MicroThings architecture. Our MicroThings runs on a Xen-based cloud with 2-core processors and 4 GB memory. MicroThings supports two running modes, the performance mode and the balanced mode, to collect data from the physical world. The former exploits all the possible resources in the cloud-based centralized controller to handle the sensing requests, while the latter only uses the configured resources. As the latter mode can be considered as one partition of the former one, we only conducted experiments to simulate the requests from our physical devices in the performance mode.

Figure 5 illustrates the results of our evaluation on computation costs and throughput capacity. In MicroThings, our centralized core supports elastic computing using adaptive threads. Figure 5a reveals that our MicroThings adaptively creates close to 200 threads to handle the requests from physical devices under the experimental configuration. As our MicroThings supports a Memcached-enabled hybrid storage framework for ubiquitous data storage, the hot data cached in the Memcached system can be used for massive IoT requests. Compared to the non-cached hybrid framework, our MicroThings can offload 60 percent of requests to the storage system (Fig. 5b). Therefore, the core of MicroThings is an efficient architecture for interoperation between two DIY environments, even for large-scale IoT deployments.

Conclusion

This article introduces a generic Internet of Things architecture, called MicroThings, which lowers the hurdle of IoT development. Specifically, the architecture supports network and application DIYs with the help of the centralized controller. Within this architecture, crowds rather than professional designers can customize their network and application developments. MicroThings consists of four parts: data aggregation, storage, computing, and processing, and provides standardized interfaces. As a result, this architecture unifies fragmented IoT elements into a whole ecosystem, facilitates the control and management of physical devices in the physical world, enriches the application resources in the cyber world, and provides elastic computing and hybrid storage for flexible interoperations between these two worlds.

Acknowledgment

This research was supported in part by China NSFC Grants U1536202, 61373173, 61602365, and 61602364, Shaanxi Science & Technology Coordination & Innovation Project 2016KTZDGY05-07-01, and Fundamental Research Funds for the Central Universities BDY131419.

References

[1] Y. Qin *et al.*, "When Things Matter: A Survey on Data-Centric Internet of Things," *J. Net. Comp. Appl.*, vol. 64, 2016, pp. 137–53.
[2] D. Boyle *et al.*, "Urban Sensor Data Streams: London 2013," *IEEE Internet Comp.*, vol. 17, no. 6, 2013, pp. 12–20.
[3] D Abebe *et al.*, "Lightweight Cybersecurity Schemes Using Elliptic Curve Cryptography in Publish-Subscribe Fog Computing," *Mobile Net. Appl.*, vol. 22, no. 112, 2017, pp. 1–11.
[4] C. Hou *et al.*, "Middleware for IoT-Cloud Integration Across Application Domains," *IEEE Design & Test*, vol. 31, no. 3, 2014, pp. 21–31.
[5] A. Al-Fuqaha *et al.*, "Toward Better Horizontal Integration among IoT Services," *IEEE Commun. Mag.*, vol. 53, no. 9, Sept. 2015, pp. 72–79.
[6] X. Sun *et al.*, "EdgeIoT: Mobile Edge Computing for the Internet of Things," *IEEE Commun. Mag.*, vol. 54, no. 12, Dec. 2016, pp. 22–29.
[7] P. C. Brebner, "Is Your Cloud Elastic Enough? Performance Modelling the Elasticity of Infrastructure as a Service (IaaS) Cloud Applications," *Proc. ACM/SPEC Int'l. Conf. Performance Engineering*, 2012, pp. 263–66.
[8] F. Paraiso *et al.*, "Managing Elasticity across Multiple Cloud Providers," *Proc. 2013 Int'l. Wksp. Multi-Cloud Applications and Federated Clouds*, 2013, pp. 53–60.
[9] O. Krieger *et al.*, "Enabling a Marketplace of Clouds: VMware's vCloud Director," *ACM SIGOPS Operating Systems Review*, vol. 44, no. 4, 2010, pp. 103–14.
[10] A. Botta *et al.*, "On the Integration of Cloud Computing and Internet of Things," *Proc. Int'.l Conf. Future Internet of Things and Cloud*, 2014, pp. 23–30.
[11] L. Jiang *et al.*, "An IoT-Oriented Data Storage Framework in Cloud Computing Platform," *IEEE Trans. Ind. Info.*, vol. 10, no. 2, 2014, pp. 1443–51.
[12] H. L. Truong *et al.*, "Principles for Engineering IoT Cloud Systems," *IEEE Cloud Comp.*, vol. 2, no. 2, 2015, pp. 68–76.
[13] A. Gumaste *et al.*, "Network Hardware Virtualization for Application Provisioning in Core Networks," *IEEE Commun. Mag.*, vol. 55, no. 2, Feb. 2017, pp. 152–59.

[14] A. Bucchiarone *et al.*, "Incremental Composition for Adaptive By-Design Service Based Systems," *Proc. 23rd IEEE Int'l. Conf. Web Services*, 2016, pp. 236–43.

Biographies

YULONG SHEN (ylshen@mail.xidian.edu.cn) is a professor at the School of Computer Science and Technology, Xidian University, China. He is an associate director of the Shaanxi Key Laboratory of Network and System Security and a member of the State Key Laboratory of Integrated Services Networks. He has served on the Technical Program Committees of several international conferences, including ICEBE and INCoS. His research interests include Internet of Things, wireless network, and cloud computing security.

TAO ZHANG (taozhang@xidian.edu.cn) received his B.S degree in computer science from Xi'an University of Post and Telecommunications, China, in 2008. He received his M.S. and Ph.D. degrees in computer science from Xidian University in 2011 and 2015, respectively. Since August 2015, he has been an assistant professor at the School of Computer Science and Technology, Xidian University. His research interests include service-oriented computing, the Internet of Things, and cloud computing security.

YONGZHI WANG (yzwang@xidian.edu.cn) received his B.S and M.S degrees in computer science from Xidian University in 2004 and 2007, respectively. He received his Ph.D. in computer science from Florida International University in 2015. Since August 2015, he has been an assistant professor at Xidian University. His research interests include cloud computing security, network security, big data, and the Internet of Things.

HUA WANG (hua.wang@vu.edu.au) is a full professor at Victoria University. He has more than 10 years of teaching and working experience in applied informatics in both private enterprise and academia. He has expertise in electronic commerce, business process modeling, and enterprise architecture. As Chief Investigator, three Australian Research Council Discovery grants have been awarded since 2006, and 155 peer reviewed scholar papers have been published.

XIAOHONG JIANG [SM] (jiang@fun.ac.jp) is currently a full professor at Future University Hakodate, Japan. His research interests include wireless networks, optical networks, network security, and more. He has published over 280 technical papers at premium international journals and conferences. He was the winner of the Best Paper Award of IEEE HPCC 2014, IEEE WCNC 2012, and the IEEE ICC 2005 Optical Networking Symposium. He is a member of ACM and IEICE.