Efficient and scalable IoT service delivery on Cloud

Fei Li, Michael Vögler, Markus Claeßens, and Schahram Dustdar
Distributed Systems Group
Vienna University of Technology
1040 Vienna, Austria
{*lastname*}*@dsg.tuwien.ac.at, markus.claessens@gmail.com*

***Abstract*—Nowadays, IoT services are typically delivered as physically isolated vertical solutions, in which all system components ranging from sensory devices to applications are customized and tightly coupled for the requirements of each specific project. The efficiency and scalability of such service delivery model are intrinsically limited, posing significant challenges to IoT solution providers. Therefore, we propose a novel PaaS framework that provides essential platform services for IoT solution providers to efficiently deliver and continuously extend their services. This paper first introduces the IoT PaaS architecture, on which IoT solutions can be delivered as *virtual verticals* by leveraging computing resources and middleware services on cloud. Then we present the detailed mechanism and implementation of *domain mediation*, which helps solution providers to efficiently provide domain-specific control applications. The proposed approaches are demonstrated through the implementation of a domain mediator for building management and two use cases using the mediator.**

***Keywords*-Cloud computing, PaaS, Internet of Things, Service delivery**

I. Introduction

Current Internet of Things solutions are typically provided in single domains [1][2], for example building management, smart logistics and so on. In such applications, domain-specific or project-specific requirements drive the design of all system components and determine most technological elements ranging from sensors and smart devices to middleware components and application logics. The service delivery process is orchestrated by IoT solution providers, who survey target application environments, analyze application requirements, select hardware devices, integrate subsystems provided by various vendors, develop applications, provide computing infrastructure and maintain services throughout the lifetime of the system.

Although this service delivery model has propelled the fast growth of IoT businesses in the last couple of years, it leads to many physically isolated vertical systems, in which hardware, networks, middleware and application logics are tightly coupled. As IoT continues to be adopted in more and more businesses and to weave into our daily life through movements like smart cities[3][4], the intrinsic limitations of such vertical systems have started to emerge. IoT solution providers are burdened with the maintenance of existing systems, which run separate software instances on top of different IoT infrastructures and computing resources in diverse physical environments. It is difficult to extend deployed services because such activities often require changes in all system layers. More importantly, delivery of services to new customers is inefficient because each time the same process has to be repeated to develop and deploy a new physically isolated vertical solution.

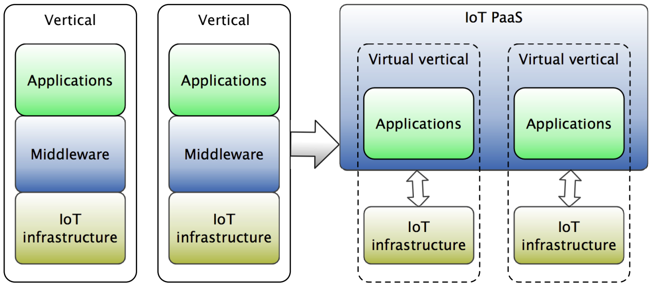

Figure 1. From physically isolated verticals to virtual verticals

To this end, this work aims at leveraging cloud service delivery models to enable efficient and scalable IoT service delivery. The core idea is to realize a domain-independent PaaS framework that provides essential platform services on cloud for IoT solution providers to efficiently deliver and continuously extend their services. The contribution of this paper is two-fold. First is the design and implementation of an IoT PaaS architecture. It inherits the multi-tenant character of cloud to enable a concept of *virtual verticals*, as opposed to physically isolated vertical solutions. In virtual verticals, each IoT solution customer owns a virtually isolated solution that they can customize to their physical environments and devices while sharing the underlying computing resources and middleware services with other customers. The concept is illustrated in Figure 1. Since IoT solutions are highly domain-specific, the second contribution of this paper is an approach to extending the generic IoT PaaS framework to different domains. The approach is based on extensible *domain mediators* that handle domain-specific device and data models. Multi-tenant, customizable provisioning of domain-specific control applications is supported by the domain mediators and the IoT PaaS architecture.

The remaining parts of the paper are structured as follows:

Section II presents an industrial case study in a typical application domain of IoT—building management, and analyzes the limitations of current service delivery model. Section III introduces the core IoT PaaS architecture and its implementation. Section IV details the domain mediation mechanism and demonstrates it through two use cases in building management domain. The related work is compared in V. And finally the paper concludes in Section VI.

II. Industrial case—building management systems

Building management systems(BMS) are one of the typical usage domains of Internet of Things. BMS generally aims at efficiently managing building facilities (HVAC, light controls, power systems, security monitoring, life safety systems and so on) in order to conserve energy, save operation costs and improve safety and security. A BMS project starts with a solution provider collecting information about the target building, which can either be a building already in use or a new building in design. The former would often require retrofitting devices and building facilities, where as the latter is synchronised with the development progress of the building. The scale of a project can range from a single building to a large compound of various building types[1], such as airports, business districts and university campuses. Therefore, the number of devices, the volume of data to be processed and the complexity of applications could vary significantly. After surveying and design for the specific building, the solution provider will acquire suitable hardware devices from OEMs, integrate them into an infrastructure solution, develop analytical and control applications and deploy the applications on dedicated server resources.

This process produces many vertically isolated BMS[5] (often referred to as ”silos”), which lead to two acute problems for solution providers. First is maintainability. The more silos are provisioned, the more system instances the solution provider needs to maintain. System maintenance becomes a particularly painful process — hardware devices are monitored in separate systems, and software instances need to be updated separately and tested on-site with the specific hardware configurations. Second is extensibility. Many campuses and building compounds expand continuously to accommodate new users. This is particularly common in large-scale projects which are usually planned for multiple progressive phases. BMS ought to scale up with the projects accordingly. In the current silo-based service delivery model, such expansion may require a bottom-up re-configuration of the whole system and provisioning of new computing resources because of the tight coupling of devices, middleware and applications.

Furthermore, isolated BMS also limit the potential of them to provide further services beyond controlling buildings. The data collected from individual buildings are largely underutilized because of the isolation of data storage and processing modules. Nowadays, the painstaking process of data cleaning and integration is a prerequisite to conduct fine-grained data analysis in large-scale[5]. BMS solutions are also outlets of civilian safety and security services. Novel civil service systems can take direct alarms from sensors in buildings and redirect necessary contextual information of accidents to appropriate emergency services such as fire stations and police stations[2]. To apply such civil service to isolated buildings, changes have to be made in each BMS solution.

It is worth noting that the limitations demonstrated in delivering building management solutions are commonly observed in many other IoT applications domains, such as smart homes, healthcare, fleet management and so on. Taking fleet management as an example, multiple fleet management solutions are used to manage different fleets. Each fleet may have different number of vehicles, which are of different types and serve different purpose (e.g., transportation of goods, emergency service). The silo-based delivery model is commonly applied in the state-of-the-art fleet management solutions[3].

III. IoT PaaS

A. Architecture

To address the demonstrated limitations and enable efficient and scalable delivery of IoT services, we propose the IoT PaaS architecture, illustrated in Figure 2.

The IoT infrastructure consists of networked tags, sensors, actuators, smart devices and so on. Gateways[6][7][8] are commonly applied in many IoT solutions to connect heterogeneous, resource-constraint devices. The gateways provide device drivers and protocol stacks for various lower-level communication protocols such as IPv6 over Low power Wireless Personal Area Networks (6LoWPAN) and WLAN. Web-based RESTful interfaces, e.g., Open Building Information Exchange (oBIX)[9] and Constrained Application Protocol (CoAP)[10], are growingly being supported by many gateways to ease the integration of IoT infrastructure with enterprise applications. The mechanisms for providing service interfaces for devices are generally referred to as device virtualization[11], since they effectively translate device and network interfaces to software interfaces. *IoT resource management* provides a registration point for virtualized devices, gateways and control applications. The component monitors the resource status and enforce the access policies through gateways. Although most existing gateway solutions are intended to mitigate lower-level hardware and communication heterogeneity to a certain extent, the diversity of exiting domain-specific data models has introduced another

[1] http://www.pacificcontrols.net/projects/ict-project.html

[2] http://www.pacificcontrols.net/projects/national-security-life-safety.html

[3] http://www.gt.honeywell.com/en-us/Industrysolutions/Landmobile/

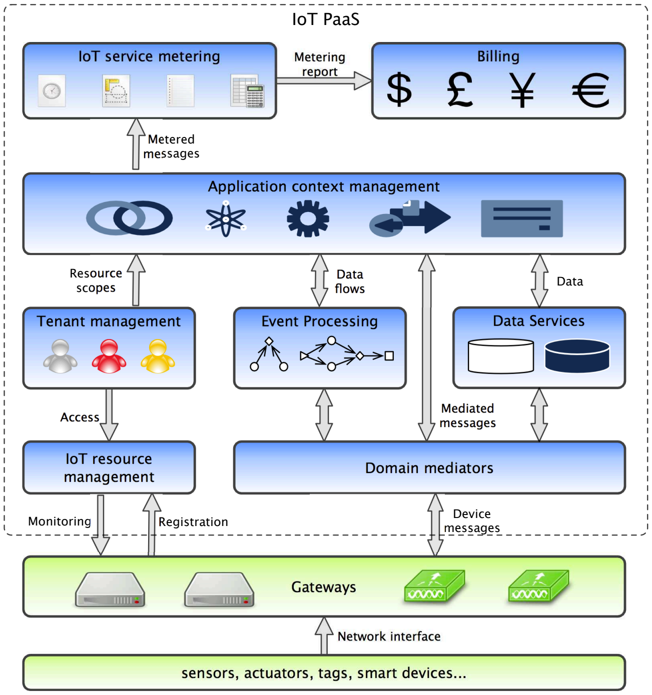

Figure 2. The IoT PaaS platform

layer of heterogeneity. Therefore, we propose *domain mediators* to mediate the interfaces between different gateways in the same application domain. This mechanisms allows IoT solutions to conform to the standardization efforts in various domains, such as oBIX for building management or Continua Health Alliance (CHA)[12] for healthcare. The cloud-based mediation mechanisms will be detailed in the next section.

IoT solutions usually generate considerable amount of data. On IoT PaaS, two types of services related to data are provided to handle real-time events and persisted data respectively. *Event processing* is to process and analyze real-time events generated by sensory devices. The service is able to produce data flows and detect interested patterns or events according to data users' specifications. In addition, it ensures that the usage of events comply to the access policies of the event provider. Alternatively, *Data service*, as a standard service of PaaS, facilitates storing, retrieving and manipulating persisted data while hiding the specifics of the underlying database systems.

Tenant management provides a consolidated view of the resources that are accessible by each tenant. In the IoT PaaS architecture, the resources include not only cloud resources such as virtual machines and software instances in traditional cloud offerings, but also IoT resources. Device capabilities and control applications can be provided to multiple tenants through virtualization. For instance, fire alarms can be shared between building management and emergency service of a city. Two distinct sharing modes are provided in our architecture. First is the sharing of real-time information. It resembles non-exclusive read access to database, and is supported by both domain mediators and event processing. Second is the sharing of control applications across multiple solutions. The mechanism to provide multi-tenant control applications is detailed in the next section.

In the convergence of IoT and cloud, each application is running in a complex and dynamic context, which may encompass available IoT and cloud resources as well as software configurations. Thus, *Application context management* is focused on maintaining the optimal runtime resources and software configurations for applications. Based on the resources acquired through tenant management, application context management helps applications to select the necessary resources at runtime to fulfill the functional requirements and meet the SLA and cost targets. The tenant management and application context management together give each IoT solution a virtually isolated operational environment, enacting the concept of virtual verticals.

Quantitatively measuring the delivery of IoT services helps service providers to understand how services are consumed. Furthermore, it provides the usage information for stakeholders involved in service delivery, for example telecom service provider and cloud infrastructure provider, to decide how to distribute cost and share revenue. *IoT service metering* measures the usage of various services that could be involved in the delivery of an application, mainly by monitoring service messages and invocations that are concerned by the platform and stakeholders. We devised three metering models, which are correspondent to three common service usage patterns: 1) time-based, which measures a service usage by the start and stop of a service instance; 2) invocation-based, which monitors the number of specific service invocations; 3) volume-based, which measures the amount of data used by a service. It is worth noting that this metering mechanisms are complementary to the metering of computing resources on cloud. The metered information of both IoT and cloud resources is composed to provide a comprehensive view of service usage. Eventually, in the whole lifecycle of service delivery, *billing* bounds services with various business models at runtime. It generates bills for stakeholders by analyzing the metered information according to charging schemes configured by stakeholders.

B. Implementation

The implementation of IoT PaaS architecture is based on an open-source PaaS solution—WSO2 Stratos[4]. It is a fully-fledged and infrastructure-agnostic PaaS solution that can be extended and customized for our purpose. Stratos comes with built-in multi-tenancy, which provides tenant-aware load-balancing, identity management for tenants and multi-tenant platform services (data, logging, monitoring).

[4] http://wso2.com/cloud/stratos/

The WSO2 services directly used in our implementation are ESB, Data Services, Governance Registry and Identity server. The implementation of event processing and metering are presented in the following, and domain mediators are detailed in the next section.

The event processing component is composed of basic and complex event processing. We use JMS as our underlying message oriented middleware API, supported by WSO2 Message Broker. Basic event processing is focused on handling events that get published to queues and/or topics. It implements pipe and filter chains, so that several event-processors can be hooked up to implement more complex routines. Complex event processing (CEP) combines data from multiple event streams to infer patterns or run more complex analysis. The goal of this process is to identify important events and react to them promptly and properly. The implementation of CEP component utilizes WSO2 CEP Server with Esper[5] and Siddhi[6] CEP engines.

Stratos allows us to measure the consumption of resources like bandwidth, invocations, storage, etc. The aforementioned IoT service metering models are extended from the basic Stratos meters, which are designed for enterprise services rather than IoT services. The metered information is collected by a Business Activity Monitor (BAM) and summarized periodically. Based on these metering informations we can run Billing to generate bills according to a given metric e.g.: consumed bandwidth or invocations of a certain service. Furthermore a Throttling component is configured and scheduled to run throttling rules periodically and update permission of tenants for accessing or consuming various resources and services.

IV. Domain mediation

In the IoT PaaS architecture, one of the key problems is to deliver control applications that rely on device capabilities in various physical environments. The control applications are at the core of many IoT services, e.g. temperature control in buildings. Traditionally, they are deployed on gateways, which provide application runtime environments with limited capabilities. When the solutions are provided at small scale, it is feasible to manually configure and manage the control applications on the gateways. However, when the applications are provided at the scale of a city, further challenges arise in the scalability of such delivery model. In our experience of developing and deploying control applications in smart building domains and other industries, we found that the following features of control applications should be considered when providing them in large scale.

- The applications reflect industrial practices on common control logics and are often similar across solutions in different environments. For example, the Office of Scientific & Technical Information at U.S. Department of Energy[7] has published a set of methods for continuous commissioning of building systems[13]. These methods are vendor-agnostic and widely adopted in the industry.
- They need to be customized for each solution. Even though the control logics can be applied to many similar solutions, each of them need to be parameterized for each set of devices and their physical environments. Further more, IoT applications are often required to respond to context changes of users. For example, presence-based lighting is commonly applied in many buildings. Such applications, though of relatively simple logic, should be configured for different user preferences and management policies.
- The gateway environments are often vendor-specific and incompatible with each other. It is very common that various gateway models are used in different solutions due to the existence of legacy systems, technical specifics and cost constraints. Thus to provide IoT services in large-scale, it is infeasible to maintain a large number of such applications on heterogeneous gateways.

In this section we present an approach to efficiently providing such control applications through IoT PaaS to domain-specific solutions, and detail the design and implementation of a domain mediator for delivering facility control applications to a large amount of buildings.

A. The process of providing control applications

The process of providing control applications on the IoT PaaS architecture is illustrated in Figure 3.

First of all, the control applications are regarded as part of the IoT resources for IoT services to employ. Thus they are registered to IoT resource management after they are developed. The development can be done by either third-party developers or solution providers. The provider of each virtual vertical solution needs to subscribe to the applications that will be used in the solution. The subscription is under the agreed tenancy between the solution provider and IoT PaaS platform provider. After subscription, the solution provider can configure each control application with the proper parameters (e.g. goal of control, device IDs) through application context management. At the deployment phase, the solution provider will deploy the solution with subscribed control applications to the configured application context. The availability of necessary IoT resources is monitored by IoT resource management and provisioned during deployment. Finally, the applications are executed in their own contexts and devices are invoked through mediators. Each virtual vertical solution can decide on its own when and how to use the applications, which can be invoked

[5] http://esper.codehaus.org

[6] http://siddhi.sourceforge.net

[7] http://www.osti.gov/

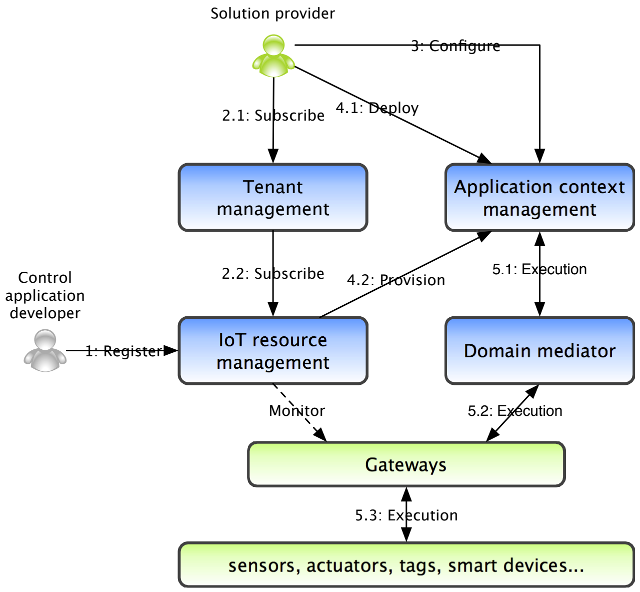

Figure 3. Providing control applications on IoT PaaS

periodically or irregularly according to the monitored system conditions and user context.

Compared to the traditional physically-isolated IoT service delivery model, by which the control applications have to be developed for each gateway model and configured on-site because of their isolated runtime environment, the proposed approach to providing control applications offers the following benefits.

1) Highly reusable, multi-tenant control applications. Same implementation can be shared between different solutions through multi-tenancy and application context management.
2) Control applications are managed within the overall service framework, in contrast to the vertical solutions in which applications are managed separately within each solution.
3) Applications instances are gateway agnostic. The control applications are implemented as services and devices are invoked through mediators. This approach ensures that the same control logic can be used on different gateway models and physical devices.

B. oBix

The key component to enable the proposed delivery model is the domain mediator. Before exemplifying the implementation of a domain mediator for building management, we first briefly introduce the oBix standard, which is the basis for the design of building management mediator.

”The purpose of the OASIS Open Building Information Exchange (oBIX) TC is to define a standard web services protocol to enable communications between building mechanical and electrical systems, and enterprise applications.

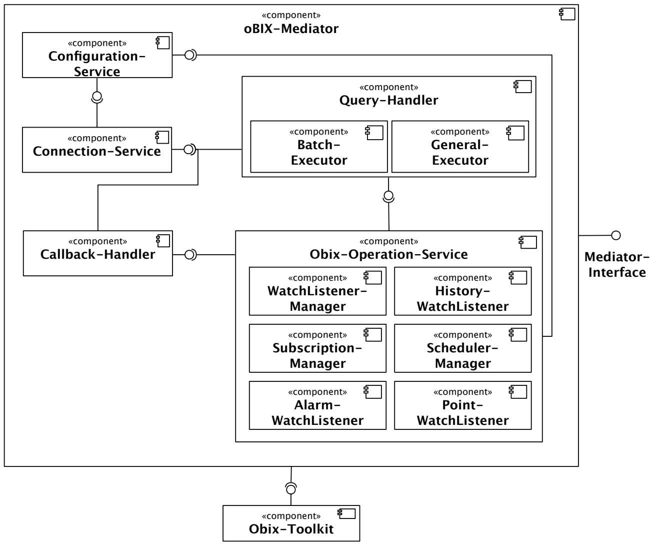

Figure 4. Domain mediator for building management

This protocol will enable facilities and their operations to be managed as full participants in knowledge-based businesses.”[8]. The oBIX architecture consists of *Object Model*, *XML Encoding*, *Binary Encoding* (for constrained devices and networks like 6LoWPAN), *URIs*, *REST interfaces*, *Contracts* (for defining new oBIX types), and *Extendibility*. For our cloud based service delivery platform, the three web service interfaces of our concern are as follows[9].

- ”Points: representing a single scalar value and its status — typically these map to sensors, actuators, or configuration variables like a setpoint.
- Alarming: modeling, routing, and acknowledgment of alarms. Alarms indicate a condition which requires notification of either a user or another application.
- Histories: modeling and querying of time sampled point data. Typically edge devices collect a time stamped history of point values which can be fed into higher level applications for analysis.”

C. A mediator for building management

Figure 4 illustrates the architecture of the building management domain mediator. The implementation uses open-source oBIX Tookit[10] for basic oBIX data and object models. And the Niagara Framework[11] is used for defining the device interfaces used in the use cases.

The oBIX-Operation-Service handles callbacks from gateways and queries from building management solutions.The Callback-Handler is the generic interface for getting various types of updates, which include value changes of control

[8] http://obix.org/what.htm
[9] http://en.wikipedia.org/wiki/OBIX
[10] http://sourceforge.net/projects/obix/
[11] http://www.niagaraax.com/cs/products/niagara_framework

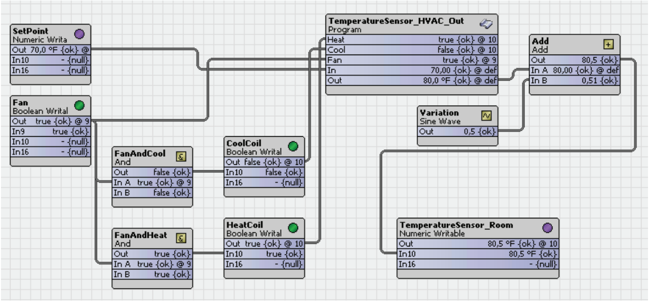

Figure 5. Temperature control use case

points, alarms, and histories. They are implemented respectively as the WatchListners. Queries are used to perform retrieval of current status and change the values of control points. Both single query and batch query are supported. Single query is applied to specific device type, e.g. ”AirHandler”, whereas batch query can be used to operate on a combination of control points and output as a list. Queries and callbacks form the basic operations on control points, alarms and histories. On top of the basic operations, Subscriptions are provided to continuously monitor the states of facilities, and Schedules to periodically apply certain operations. The Connection-Service offers interface for login into Niagara Framework.

D. Use cases

Two frequently used control logics in building management—Temperature control and Presence-based light control—have been implemented based on the presented domain mediator.

1) Temperature control: This use case implements a room with a HVAC system that keeps the room temperature around a preset value. The physical devices (Fan, CoolCoil, HeatCoil, TemperatureSensor_Room and so on) of the HVAC system are modeled in the Niagara Workbench, illustrated in Figure 5.

The control logic is implemented as a service on IoT PaaS platform and uses the domain mediator to access these devices. The application reads the current room temperature from the TemperatureSensor_Room (Out) and the desired temperature defined by the SetPoint (Out). By comparing them, either the combination of CoolCoil and Fan or the HeatCoil and Fan is activated. This is accomplished by writing to the ”In9” control of the Fan and to the ”In B” of FanAndCool/Heat, which are directly wired to their Coil. The FanAndCool/Heat module is used to assure that the Fan is running. The program module named TemperatureSensor_HVAC_Out simulates the output air temperature of a HVAC facility. The Variation module adds further randomness and assures that the room temperature keeps changing.

The oBIX operations used in this use case through the

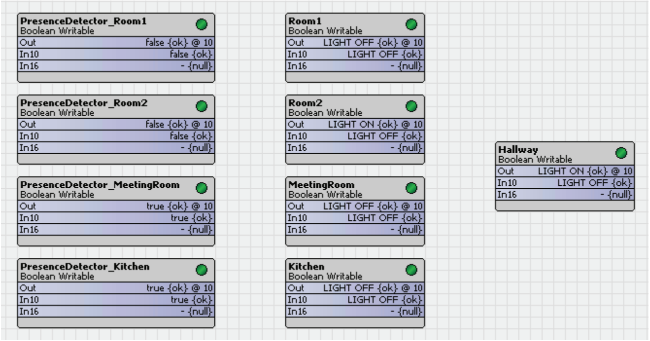

Figure 6. Light control use case

mediator are Subscription, Point-WatchListner and BatchQueries. The SetPoints output value can be set manually by invoking the ”Set” action which sets the default fallback value of the SetPoint, then applies to its output. As the temperature control service subscribes to WatchListeners of the SetPoints output and TemperatureSensor_Room’s output, it reacts to the changes by controlling the fans and coils. To perform the cooling or heating action all the necessary operations are added to a BatchQuery, which then sends them as one message to the mediator.

2) Presence-based light control: This use case implements several rooms of an office which have a presence detector built in to control each rooms light. The presence detectors are simulated by randomly changing their boolean values. If the light of all rooms is turned off, the light of the Hallway is turned off too.

The control application on IoT PaaS subscribes to the output of PresenceDetector of each room through respective WatchListeners. The control on each room light and the hallway light is implemented as single Queries on respective control points.

V. Related work

Cloud and Internet of Things are emerging computing paradigms featuring distinctly different computing resources and system architecture. As both paradigms are growingly being adopted in more and more application areas, researchers and practitioners have started to investigate the convergence of cloud and IoT in order to exploit their intrinsic complementarities.

The basis for the convergence of cloud and IoT was established in the work of Web of Things[11], which has proposed a set of methods to access devices through web-based technologies such as web services and RESTful interfaces. It has solved the problem of managing and using IoT resources in a service-oriented framework. The early work on the convergence of cloud and IoT are mostly direct applications of WoT architecture on cloud. Hassan et al. [14] integrated cloud and wireless sensor networks (WSN) by developing several key functional components of WSN on cloud, namely pub/sub broker and resource registry. [15] virtualizes physical sensors as software entities on cloud, which provides users with sensory service provisioning, resource management, and monitoring. Semantic technologies have long been adopted in modeling sensory information. Alam et al. [16], [17] enhanced sensor virtualization through semantic abstraction for sensor capabilities. Generally, the focus of the early work is on IoT resource management rather than service delivery. Cloud is viewed as computing infrastructure to facilitate the management of large amounts of IoT resources. Following these early results, domain-specific systems have been proposed on ambient living[18], healthcare[19] agriculture[20] and so on.

Since recently, new research initiatives have started to emerge on exploiting the service delivery models of cloud to accommodate the growing scale and diversity of IoT services. Soldatos et al.[21] has presented the idea of converging IoT and utility computing on cloud as the core concept of the OpenIoT project[12]. The proposed architecture is based on CoAP[10] and linked data. The work uses the cloud concept at infrastructure level, in the way that the utility of services provided by inter-connected objects is measured. However, the concept does not address the problem of efficient service delivery and multi-tenancy in a PaaS model. Cloud of Things [22] is intended to establish a conceptual architecture by mapping various elements in both clouds and IoT to the three layers of cloud architecture (IaaS, PaaS and SaaS). It is based on the assumption that IoT resources are voluntarily provided by their owners. IoT management functions, such as node management and policy enforcement are viewed as peer functions of cloud infrastructure management. At PaaS level, IoT resources and cloud computing infrastructures are mashed up for applications, which is then delivered through SaaS. We argue that the very different natures in computing power, mobility, reliability and usage patterns of cloud and IoT resources would make the mash-up approach difficult for application providers to accommodate the needs for multi-tenancy, scalability and SLA management. Both OpenIoT and Cloud of Things are still in the conceptual stage.

One notable recent work is from Hummen et al. [23]. They focus particularly on privacy and security concerns as a consequence of applying multi-tenancy on SensorCloud architecture. Although our work is on service delivery methods rather than privacy and security, the two researches are complementary in the respect that both are promoting multi-tenant architecture for IoT services.

To summarize, although the benefits of leveraging cloud service delivery models in IoT have been recognized, the research on how to exploit the benefits and the actual system architecture is still in its infancy. Some early models on cloud-based IoT service delivery have been proposed, but they are yet to be realized and validated in real-world applications. In contract, this paper is based on our experience in continuously delivering real-world IoT services at the scale of a large city. The proposed IoT PaaS architecture enables a common IoT service delivery platform that supports multiple virtual vertical solutions. The domain mediator provides an extensible and efficient way to provide control applications to multiple tenants in multiple domains.

VI. Conclusion and future work

This paper proposed IoT PaaS — a novel cloud platform that supports efficient and scalable IoT service delivery. On the platform IoT solution providers are able to efficiently deliver new solutions by leveraging computing resources and platform services such as domain mediation, application context management and metering on cloud. The multi-tenant nature of the architecture helps to isolate the operation environments of different solutions, enabling virtual vertical service delivery that are more extensible and scalable compared to the mainstream physically isolated vertical solutions. The domain mediators provide an extensible mechanism for IoT PaaS to engage with various domain-specific data models and provide control applications that rely on physical devices. A domain mediator for building management and two control applications are implemented to demonstrate the mechanism.

The proposed architecture and reference implementation is being further developed into an industrial-grade IoT cloud offering. At the same time, the future research work on the IoT PaaS will be conducted in two directions. First is to evaluate and model the resource consumption of IoT applications in order to effectively allocate computing resources on the multi-tenant IoT service platform. The application-oriented resource model will consider device behavior, physical context of applications, data processing requirements and usage patterns. Second is to investigate a comprehensive QoS model across IoT and cloud environments, in order to provide virtual verticals with end-to-end QoS assurance.

Acknowledgment

This work is sponsored by Pacific Controls Cloud Computing Lab (PC^{3}L)[13], a joint lab between Pacific Controls L.L.C., Dubai and the Distributed Systems Group of the Vienna University of Technology.

References

[1] D. Miorandi, S. Sicari, F. De Pellegrini, and I. Chlamtac, “Internet of things: Vision, applications and research challenges,” *Ad Hoc Networks*, vol. 10, no. 7, pp. 1497–1516, Sep. 2012.

[2] L. Atzori, A. Iera, and G. Morabito, “The Internet of Things: A survey,” *Computer Networks*, vol. 54, no. 15, pp. 2787–2805, Oct. 2010.

[12] http://vmusm03.deri.ie

[13] http://pc3l.infosys.tuwien.ac.at/

[3] IBM, “TheSmartCity.” [Online]. Available: http://www-03.ibm.com/innovation/us/thesmartercity/index_flash.html

[4] MIT, “MIT Cities.” [Online]. Available: http://cities.media.mit.edu/

[5] J. Cook, D. Smith, and A. Meier, “Coordinating Fault Detection, Alarm Management, and Energy Efficiency in a Large Corporate Campus,” in *2012 ACEEE Summer Study on Energy Efficiency in Buildings*, 2012, pp. 83–93.

[6] Q. Zhu, R. Wang, Q. Chen, Y. Liu, and W. Qin, “IOT Gateway: BridgingWireless Sensor Networks into Internet of Things,” *2010 IEEEIFIP International Conference on Embedded and Ubiquitous Computing*, pp. 347–352, 2010.

[7] Tridium, “JACE Controller.” [Online]. Available: http://www.tridium.com/cs/products_/_services/jace

[8] ThereCorporation, “ThereGate.” [Online]. Available: http://therecorporation.com/en/platform

[9] OASIS, “Open Building Information Exchange (oBIX).” [Online]. Available: https://www.oasis-open.org/committees/tc_home.php?wg_abbrev=obix

[10] IETF, “Constrained Application Protocol (CoAP).” [Online]. Available: http://tools.ietf.org/html/draft-ietf-core-coap-08

[11] D. Guinard, V. Trifa, S. Karnouskos, P. Spiess, and D. Savio, “Interacting with the SOA-Based Internet of Things: Discovery, Query, Selection, and On-Demand Provisioning of Web Services,” *IEEE Transactions on Services Computing*, vol. 3, no. 3, pp. 223–235, Jul. 2010.

[12] “Continua Health Alliance (CHA).” [Online]. Available: http://www.continuaalliance.org/

[13] L. Luskay, M. Brambley, and S. Katipamula, “Methods for Automated and Continuous Commissioning of Building Systems,” PORTLAND ENERGY CONSERVATION, INC. and BATTELLE NORTHWEST DIVISION, Tech. Rep., Apr. 2003.

[14] M. M. Hassan, B. Song, and E.-N. Huh, “A framework of sensor-cloud integration opportunities and challenges,” in *Proceedings of the 3rd International Conference on Ubiquitous Information Management and Communication - ICUIMC ’09*. New York, New York, USA: ACM Press, Feb. 2009, pp. 618–626.

[15] M. Yuriyama and T. Kushida, “Sensor-Cloud Infrastructure - Physical Sensor Management with Virtualized Sensors on Cloud Computing,” in *2010 13th International Conference on Network-Based Information Systems*. IEEE, Sep. 2010, pp. 1–8.

[16] S. Alam and J. Noll, “A Semantic Enhanced Service Proxy Framework for Internet of Things,” in *2010 IEEE/ACM Int’l Conference on Green Computing and Communications & Int’l Conference on Cyber, Physical and Social Computing*. IEEE, Dec. 2010, pp. 488–495.

[17] S. Alam, M. M. R. Chowdhury, and J. Noll, “SenaaS: An event-driven sensor virtualization approach for Internet of Things cloud,” in *2010 IEEE International Conference on Networked Embedded Systems for Enterprise Applications*. IEEE, Nov. 2010, pp. 1–6.

[18] X. M. Zhang and N. Zhang, “An Open, Secure and Flexible Platform Based on Internet of Things and Cloud Computing for Ambient Aiding Living and Telemedicine,” in *2011 International Conference on Computer and Management (CAMAN)*. IEEE, May 2011, pp. 1–4.

[19] D. Gachet, M. de Buenaga, F. Aparicio, and V. Padron, “Integrating Internet of Things and Cloud Computing for Health Services Provisioning: The Virtual Cloud Carer Project,” in *2012 Sixth International Conference on Innovative Mobile and Internet Services in Ubiquitous Computing*. IEEE, Jul. 2012, pp. 918–921.

[20] Y. Bo and H. Wang, “The Application of Cloud Computing and the Internet of Things in Agriculture and Forestry,” in *2011 International Joint Conference on Service Sciences*. IEEE, May 2011, pp. 168–172.

[21] J. Soldatos, M. Serrano, and M. Hauswirth, “Convergence of Utility Computing with the Internet-of-Things,” in *2012 Sixth International Conference on Innovative Mobile and Internet Services in Ubiquitous Computing*. IEEE, Jul. 2012, pp. 874–879.

[22] S. Distefano, G. Merlino, and A. Puliafito, “Enabling the Cloud of Things,” in *2012 Sixth International Conference on Innovative Mobile and Internet Services in Ubiquitous Computing*. IEEE, Jul. 2012, pp. 858–863.

[23] R. Hummen, M. Henze, D. Catrein, and K. Wehrle, “A Cloud design for user-controlled storage and processing of sensor data,” in *4th IEEE International Conference on Cloud Computing Technology and Science Proceedings*. IEEE, Dec. 2012, pp. 232–240.