Software Defined Unified Device Management for Smart Environments

Varun M Tayur
Jain University
Dept. of Computer Science, Bangalore

Suchithra R
Jain University
Dept. of Computer Science, Bangalore

ABSTRACT

Applications such as street lights, parking meters, traffic light and congestion sensors, safety cameras, air quality sensors, weather sensors, remote personal healthcare monitors, and acoustic detection define the new world of “Smart Environments” that we live in. The technologies driving these applications are customized to the environment in which they run. The backhauling infrastructure to support each application is as diverse as the application itself. An abstraction which hides the complexities of such disparate systems presenting a unified view is desired. Integrated management platform aids in comprehensive management of the infrastructure. “Smartness” also means it is ready to use by just plug and play. A plug-in based implementation supports multiple vendor devices while a unified northbound controller ensures minimal vendor lock for the applications running on the top.

General Terms

Internet of Things

Keywords

Smart Environment unified device access; Smart Environment device management, IoT vendor locks

1. INTRODUCTION

A smarter place to live in is a reality now due to the intelligence being embedded into the environment around us. These tiny devices have changed the world around us in many ways – however, behind the nice environment they provide to us there is a massive inter-operability challenge masked within it. These devices need to communicate with humans and amongst themselves in order to provide the expected service. A new technology paradigm called the Internet of Things (IoT) has enabled a suitable workable framework for enabling such communication. All this is made possible when the architecture can enable heterogeneous devices to interoperate and yet be scalable to address the huge number of such devices in the environment.

A Smart Environment is comprised of many types of embedded sensing and actuation devices; they are customized to run on specific protocols and technologies suited to their need. More importantly the smart environments can be composed of multiple vendors providing the technologies and protocols in the system. Interoperability, management and programming such diverse systems need an effective management platform which can hide the complexities and present a unified view to the user.

Communication between such disparate set of devices is enabled by conversion gateways or adapters sitting in between the devices and the internet. The gateways are limited in the scope as they are tied to the hardware and a particular protocol in question. They are very rigid in the sense that they cannot be re-programmed or re-used. Reprogrammable software controlled switching fabric customized to handle the needs of the IoT which can treat the devices as first class citizens in the network can ease the management of the network and provide a homogenous operating environment.

Hardware is prone to failure, hence it is important to be able to manage the lifecycle of all the devices. It includes constant monitoring, fault detection and isolation, ability to provision new devices, remove faulty ones in the network and update the configurations on the devices; all these non-intrusively with minimum overhead as most of all these devices are low powered and are resource constrained.

Smartness is more about automation than anything else – how to react to a particular context and provide the necessary service. Automation can be enabled by providing the necessary programming infrastructure. An end-to-end solution for programming and provisioning the IoT applications by providing suitable abstractions is very much desired as it enables developer communities to progress further on the vision of the Smart Environment.

A new computing paradigm called the Cloud provides distributed and scalable computing infrastructure on demand. The Cloud can be exploited to meet need of IoT, where it needs massive computing power to manage the billions of devices. The distributed nature of the devices is another challenge where the Cloud can support the IoT to aggregate information across various sites and unify them to present the notion of a single system. So, it is very crucial for IoT to be Cloud aware as it provides the necessary infrastructure and can scale to the sheer amount of devices we have to manage which may not be possible with the existing infrastructure.

Mobile computing and platforms can enable resource control and management on the go. The mobility of this sort is supported by the vision of the Cloud where resource aggregation is possible. Hence, it is very important for IoT environments to be cloud aware so that integration is easy

2. RELATED WORK

IoT includes a broad range of technologies, most of them legacy technologies. The Integration of the non IP based devices into the Internet is challenging considering the custom protocols and resource constraints. Adaptation and extension of the existing IoT building blocks (such as solutions from IEEE 802.15.4, BT-LE, RFID) while maintaining backwards compatibility with legacy networked embedded systems is discussed in [1]. An extended Internet stack with a set of adaptation layers from non-IP towards the IPv6-based network layer in order to enable homogeneous access for applications and services using conversion gateways implemented on a card.

A framework for managing the devices and configuring the network dynamically based on SDN System Architecture is presented in [2]. The architecture consists of the control plane, having the controller and application platform, the other is data plane, which makes up devices and networks. An agent resides in M2M device or M2M gateway which corresponds to the Execution Framework (EF). The EF receives the commands from the controller and programs the network. Due to the heterogeneity of devices and access protocols, IoT networks are becoming enormous and complex, the SDN allows devices to be treated as objects decoupling the control plane from the data plane and hiding the complexity.

An OpenFlow implementation adapted for the wireless sensor networks is presented in [3]. It proposes two new abstract layers, a Common platform layer over the IoT devices and virtualization layer which be added at the top and bottom of a present Infrastructure. It proposes to utilize the openflow protocol for providing a common management protocol and it concept borrowed from software defined networks to establish connectivity to the devices. Simulation results have shown that this architecture to scale very well for large network sizes and achieve upto 39% points more traditional sensor networks.

Higher level connection-technology-independent protocols are needed to shield different connection technologies in the integration requirements for things'. A management protocol which can be used to exchange information end-to-end is necessary. A SOAP-based [4] Things Management Protocol (TMP) is proposed which operates in the Application layer of the Internet stack. Operations like get/set similar to SNMP operations enables uniform interface for communication between the things with things and things with the applications. TMP is a key technology for information integration and application based on connection-technology-independent protocols, SOAP-based TMP can take full advantage of HTTP, XML, SOA and other widely used technology with broad application prospects.

IoT's integration into IPv6 and its related protocols has been a major challenge considering the constrained capabilities offered by Wireless Sensor Networks, building automation, and home appliances. Integration of the existing management protocols in IPv6 into the emerging IoT networks based on protocols such as 6LoWPAN is discussed in [5][6]. The COnstrained networks and devices MAnagement (COMAN) Group from the IETF proposes solutions such as simplified MIB, new SNMP consideration, and CoAP-based management which could be the protocol to use for network management in IoT.

LoWPAN Network Management Protocol (LNMP) is management architecture suited for the 6LoWPAN networks [7]. LNMP architecture focuses on reducing the cost of communication and hence increases the lifetime of the network. The main objective of LNMP is interoperability with SNMP, but, SNMP is considered large both in terms of communication and complexity for devices that have limited resources. The devices deployed within the Internet of Things are resource constrained with respect to memory and processing capabilities and the low-power radio standards. An investigation on using existing management protocols like SNMP, NETCONF over IPv6 to manage low powered devices is done in [8]. A lighter version of SNMP and NETCONF protocol implemented on the contiki OS showed that SNMP was better suited over NETCONF with respect to resource utilization.

LNMP's operational architecture provides a distributed network discovery support with the help of coordinators which non-intrusively monitor and manage the devices. The informational architecture enables usage of SNMP based on the traditional IPv6 LoWPAN stack enabled by the adaptation layer for 6LoWPAN. SNMP being an Application layer protocol can be adapted to run over IPv6 with some modifications [8], a popular implementation NET-SNMP exists for use on both IPv4 and IPv6, the suite includes a full implementation of SNMP adapted for IPv6.

Open source cloud based platform [9] provides a generic platform that enables devices, RFID, NFC, M2M and sensor technologies and systems to be hosted in a decentralized architecture enabling interoperability amongst the many complementary technologies.

CloudThings [10] is an online platform that allows system integrators and solution providers to use the application infrastructure for developing, deploying, operating, and composing Things applications and services. The CloudThings is an IaaS solution that lets users run their applications easily using the platform as the base. The PaaS solution provides a developer suite to developers for development and deployment. The SaaS solution complements the platform by providing necessary services like device management and deployment. IoT and cloud integration in was based on Arduino as hardware and Paraimpu for sensor management with the software hosted on the cloud.

The lack of reasoning and intelligence in the loT is augmented by Agent of Things (AoT) [11]. AoT proposes that every "thing" will be augmented with internal reasoning and intelligence capability enabling the things to interact directly with other things of the same or different type. Such Intelligence is enabled by the software agents sitting on them but the 'things' need enough resources to run the software agent program. The AoT does not propose to address agent based implementation by upgrading the hardware to run the agent but does so by address the upper part of the stack which can be controlled more by software. The difference between AoT and loT is that the AoT, uses augmented software agents to provide the things the ability of reasoning, negotiation and delegation, in the loT concept, the things are not intelligent by themselves, but collectively.

IoT Mash-up as a Service (IoTMaaS) [12] proposes to address the problem connecting heterogeneous devices by following the model driven architecture principles and computational scalability based on cloud computing paradigm. New services are composed from existing services, using a mash-up. The mash-up is made possible with existing web mash-up technology provided each thing exposes its functionalities as a web service.

In [13] an intercepting intermediary intercepts all the requests coming into and out of the device it does all the work of transforming to and from the Constrained Application Protocol (CoAP). Matching adapters can be sequenced to handle the CoAP interactions non-intrusively providing security and other services which otherwise can strain the devices. An embedded OData implementation on top of CoAP without requiring an intermediary gateway device is presented in [14]. Additional resources required for an OData/JSON implementation are justified considering the issues in interoperability in enterprise networks.

A scalable and automated deployment of things is proposed in [15] which eliminate human intervention for configuration and maintenance. A Peer-to-Peer (P2P) architecture provides automated local and global service and resource discovery mechanisms. The P2P overlays namely the Distributed Location Service (DLS) and Distributed Geographic Table (DGT) provides the name lookup service and location information of a resource respectively. The main interface to the P2P overlay environment is via an intermediary IoT gateway which provides connectivity to the P2P infrastructure.

Large amount of work has happened in solving the challenges that has plagued IoT adoption. Scaling the existing internet and associated infrastructure has been a more immediate problem, where lot of proposals suggests using IPv6 with cloud computing paradigms. Heterogeneous nature of the devices is an important characteristic of IoT, which could be addressed to some extent by using the asynchronous messaging platforms and platform independent models. The current communication infrastructure does not suit well for the Low powered constrained devices as they will be strained – which can be addressed by CoAP type protocols. Management and monitoring is another active area of research considering all the challenges that IoT presents to us. All these developments have led us to a more automated, device centric world where a device can talk to another device without an intermediary.

3. SMART DEVICE MANAGEMENT PLATFORM

3.1 Architecture

Figure 1 shows the IoT Controller driving the Things Attach Virtual Control Unit (TAVCU). The IoT controller provides a unified common management user-interface based on the Northbound REST API. The architecture is majorly software driven and it can be run on any Operating System as long as it supports Java. It can also be hosted in a data-center providing the platform as a service (PaaS).

The Controller exposes open bi-directional Northbound APIs which are used by Applications. The business logic and algorithms reside in the Applications. The Applications use the Controller to gather network intelligence, runs its algorithm to do analytics and then use the Controller to orchestrate the new rules throughout the network. The User Interface is implemented as an application using the same Northbound API as would be available for any other user application.

The Things Attach Fabric (TAF) is a programmable virtual mesh that has multiple TAVCU. The TAVCU connects to the IoT controller. The controller itself is modular in design; the Topology Manager maintains the devices, their capabilities, reachability, etc. The Device Manager helps in generating the topology database for the Topology Manager. The statistics manager is responsible for maintaining the statistics and counters related to usage. Trust manager handles the security keys and other infrastructure related to security and trust.

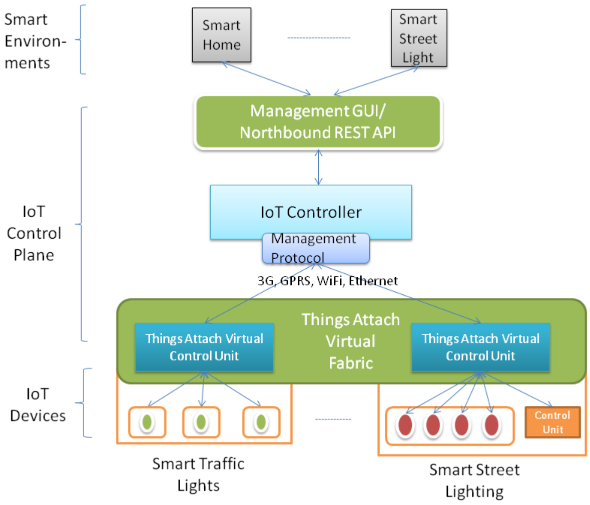

Figure 1: IoT Controller

3.1.1 High Availability

The IoT Controller supports a Cluster based High Availability. Several instances of the Controller can logically act as one controller. This gives high redundancy but also allows a scale-out model for linear scalability. To make the Controller highly available at the controller level 1 or more controller instances in clustered fashion have to be added.

3.1.2 Security

The application interfacing the controller's REST API's is secured with TLS. The multi-tenancy semantics are ensured with the concept of container abstraction that is issued with each REST request. The communication between the TAVCU and the controller can be secured with TLS. All communication on native IPv6 Low power Wireless Personal Area Network (6LoWPAN) device run the Constrained Application Protocol (CoAP) with security mode turned on. Other resources like messaging systems Message Queue Telemetry Transport (MQTT) can secure the communication with TLS or any such protocol.

3.1.3 Cloud awareness and Multi Tenancy

The controller can be hosted within a data-center, fronted with a load-balancer. The controller can be accessed from the Internet. Since the controller has complete knowledge of the topology, sharing of devices can be enabled. Access Control Lists can ensure secure access to the devices.

Multi-tenancy is one of the founding principles in the design of the controller as it is critical for providing the controller as platform as a service. Separation of the platform for access by multiple tenants is enabled by abstracting access in software. Since, devices themselves may not support sharing; sharing is limited to logical separation enforced in software; so in most cases it may be limited to sequential access.

3.2 Things Attach Virtual Control Unit

The TAVCU is a Control Unit fully implemented in software and can be ported to run on any hardware. It can be run on the router in a Smart Home. TAVCU ensures fair treatment for all types of devices; be it a Personal Computer (PC) or Smart Things (IoT); which means that the traditional networking infrastructure and protocols also need to co-exist with the IoT specific adaptations. The uplink (U) connects to the IoT controller, while the other "ports" provide connectivity to the devices. The TAVCU can be managed with a management protocol (ex. OpenFlow adaptation) or based on simple messages between the controller and the control unit. The TAVCU instantiates the plugin dynamically based on the registered device type.

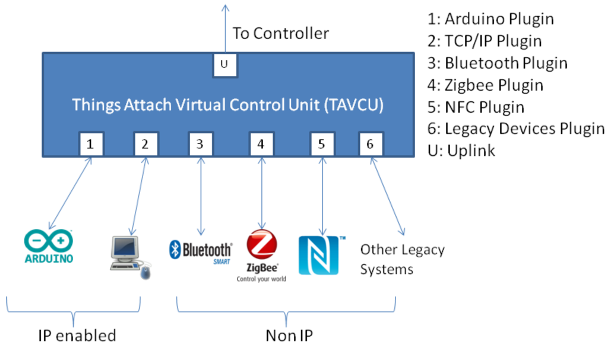

Figure 2: Things Attach Virtual Control Unit (TAVCU)

3.2.1 *Control Unit Interfaces*

There are two ways of connecting the device to the Control Unit – which can be wired or wireless. Upon connecting the device to the CU, the appropriate driver for the type of device is activated. For example, connecting a PC to TAVS will instantiate the TCP/IP stack, similarly connecting a smart device like Bluetooth LE will instantiate the Bluetooth driver which will enable the CU port to act as a slave in the Bluetooth network.

3.2.2 *Control Unit Functions*

Upon connecting/registering the device to a particular port in the Control Unit, an IP address gets bound to that port for a non-IP capable device. This IP address is only used for packets exiting out of this port to the controller. The ‘Encapsulate Packet’ rule will provide this functionality.

The Control Unit is mostly a forwarding agent, all decisions are taken in the controller and these decisions manifest as rules which are then programmed into it. So, any packet exiting/entering a port needs a rule/set of rules to be executed, in order for these rules to be computed, the first packet needs to be forwarded to the controller. The ‘Forward to Controller’ rule will provide this functionality.

In cases where there are multiple resources to be monitored or multiple ports which are sources of information, in order to solicit requests/responses to all of them ‘Flood to Ports’ can be used. ‘Send to Port’ is used when the specific port to where the communication is to be directed is known.

Table 1 presents the generic rules implemented on the Control Unit. More specifically, these are decisions that need to be taken at the port level based on the rule configured from the controller.

Table 1: TAVCU Rules

Rule	**Function**
Forward to Controller	Default action to perform if decision cannot be made on the switch. The raw packet is encapsulated with a IP header
Flood to Ports	Broadcast the packet to all the ports
Send to Port	Uni-cast the packet to a particular port only
Encapsulate Packet	Wrap the data packet (raw packet) within an IP packet

3.3 Things Attach Virtual Fabric

The virtual fabric is responsible for hosting the multiple Control Units. It also provides a unified view of the network logically. Multiple Control Units can be programmed to allow collaboration between themselves and amongst the devices. It provides an abstraction layer that enables hosting multi vendor devices in the network.

Table 2 shows some of the REST API available for the basic management functions. Each of these is available on the Control Unit. The REST API is implemented by the protocol specific plugin. Ex. Intel Edison IoT board will implement the specific REST API that will allow it to be registered, discovered and be monitored. The REST API’s also allow extensions to support features specific to a particular platform. The extensions tag specified in the payload is interpreted by the plugin appropriately.

Table 2: Management Northbound API

Feature	**API**	**HTTP Mode**	**Description**
Registration	/register	POST	{payload:data}
Discovery	/discover	POST	{operation:’start’},{operation:’stop’}
Inventory	/inventory	GET	
Configuration	/configure	POST	{payload:data}
Monitoring	/monitor	POST	{payload:data}
	/status	GET	

4. USE CASE/DISCUSSION

A Smart City is comprised of multiple applications like street lights, traffic lights, parking, ambient air quality etc. Each of these applications independently working may no doubt serve a given purpose but all these diverse set of applications, must be integrated together to give a notion of a smart city. The integration is loaded with challenges because of the absence of any standards and lack of a common extensible, open management platform.

In the context of Smart Street Lighting System, if we consider that the Street Lights can be managed remotely and some of its functions can be automated, we can see that there will be huge financial savings for the civic authorities; add to it the reduction in the power consumed which has direct ecological implication. The entire life-cycle of the Street-Light becomes managed, thus its maintenance can have quicker turnaround times – which increases the citizens quality of life. Considering the size of the deployment, a typical city will have thousands of Street Lights to be managed – a robust management platform is required. The platform is to be shared amongst all civic bodies in the city for effective management purposes.

Following is the description of a typical workflow:

An automated ambient light monitoring system attached to each street light will autonomously control the on/off state based on the lighting condition – this behavior is largely autonomous except that the ambient light value can be set by the IoT controller based on the season and time of the day.

Each of street lights belong to a logical cluster, one node amongst them has a control unit instance. A logical cluster is formed between the street lights for enabling decentralized management. The street lights are analogous to the network nodes like in a traditional network and they connect to the Things Attach Control Unit. The Control Unit communicates with the IoT Controller via any of the standard transports like 3G, GPRS, WiFi, Ethernet etc.

The control unit connects to the IoT controller to receive instructions for management and control. The rules can be programmed based on some business decisions from the top by the end-user. The status of the street lights is communicated to the IoT Controller periodically via the control unit. The communication from the Control Unit to the IoT Controller is made possible by way of rule called 'Forward to Controller' where information can be sent to the controller directly. The rule can specify additional selection criteria to forward the packet when the packet type is a periodic information packet.

5. CONCLUSION

The inclusion of multitude of existing and new technologies under the umbrella of Internet of Things has broadened the scope and it applications tremendously. A Smart City is made up of multitude of applications that are diverse in their composition and infrastructure. Diverse systems as these need a robust, open and extensible platform for management. With a sample use-case on the Smart Lighting System, the simplicity of the Software Defined Platform with a centrally hosted Controller for management and configuration was discussed. The rule based orchestration enables programmability of diverse set of technologies in a protocol and vendor agnostic way. The work can be extended to integrate the IoT Controller to work with any existing Software Defined Networking (SDN) controllers. This enables a unified management solution for the Smart Environments and existing Networks.

6. REFERENCES

[1] Antonio J. Jara, Socrates Varakliotis, Antonio F. Skarmeta, Peter Kirstein.: "Extending the Internet of Things to the Future Internet through IPv6 support", Mobile Information Systems, 2014, DOI:10.3233/MIS-130169, IOS Press

[2] Hai Huang, Jiping Zhu, Lei Zhang: "An SDN_based Management Framework for IoT Devices", ISSC 2014 / CIICT 2014, Limerick, June 26-27

[3] Arif Mahmud, Rahim Rahmani and Theo Kanter: "Deployment of flow-sensors in Internet of Things' virtualization via OpenFlow", 2012 Third FTRA International Conference on Mobile, Ubiquitous, and Intelligent Computing, 978-0-7695-4727-5, 2012 IEEE, DOI 10.1109/MUSIC.2012.41, Pg: 195-200

[4] Guiping Da: "Design and Implementation on SOAP-Based Things Management Protocol for Internet of Things", 978-1-4673-1398-8/12, 2012 IEEE, Pg: 4305-4308

[5] Hanane Lamaazi, Nabil Benamar, Antonio J. Jara, Latif Ladid, Driss El Ouadghiri: "Challenges of the Internet of Things: IPv6 and Network Management", 2014 Eighth International Conference on Innovative Mobile and Internet Services in Ubiquitous Computing, 978-1-4799-4331-9/14, 2014 IEEE, DOI 10.1109/IMIS.2014.43, Pg: 328-333

[6] Management of Networks with Constrained Devices: Problem Statement, Use Cases and Requirements, https://tools.ietf.org/html/draft-ersue-constrained-mgmt-02, 2013

[7] Anuj Sehgal, Vladislav Perelman, Siarhei Kuryla, and Jürgen Schönwälder: "Management of Resource Constrained Devices in the Internet of Things", IEEE Communications Magazine, December 2012, PG: 144-149

[8] Hanane Lamaazi, Nabil Benamar, Antonio J. Jara, Latif Ladid, Driss El Ouadghiri: "Challenges of the Internet of Things: IPv6 and Network Management", 2014 Eighth International Conference on Innovative Mobile and Internet Services in Ubiquitous Computing, 978-1-4799-4331-9/14, 2014 IEEE, DOI 10.1109/IMIS.2014.43, Pg: 328-333

[9] George Suciu et el., "Generic platform for IoT and cloud computing interoperability study," in Signals, Circuits and Systems (ISSCS), 2013 International Symposium, vol., no., pp.1,4, 11-12 July 2013 doi: 10.1109/ISSCS.2013.6651222

[10] Jiehan Zhou, Teemu Leppänen, Erkki Harjula, Mika Ylianttila, Timo Ojala, Chen Yu, Hai Jin, Laurence Tianruo Yang, CloudThings: a Common Architecture for Integrating the Internet of Things with Cloud Computing, Proceedings of the 2013 IEEE 17th International Conference on Computer Supported Cooperative Work in Design, 978-1-4673-6085-2/13 IEEE, Pg : 651-657

[11] Mzahm, Anas M; Ahmad, Mohd Sharifuddin; Tang, Alicia Y. C.; Agents of Things (AoT): An Intelligent Operational Concept of the Internet of Things (loT), IEEE 2013 13th International Conference on Intelligent Systems Design and Applications (ISDA) , 978-1-4799-3516-13, 2013, Pg: 159-164

[12] Janggwan Im et el., "IoT Mashup as a Service: Cloud-Based Mashup Service for the Internet of Things," in Services Computing (SCC), 2013 IEEE International Conference, vol., no., pp.462,469, June 28 2013-July 3 2013 doi: 10.1109/SCC.2013.68

[13] Floris Van den Abeele et el., "Fine-grained management of CoAP interactions with constrained IoT devices," in Network Operations and Management Symposium (NOMS), 2014 IEEE , vol., no., pp.1,5, 5-9 May 2014 doi: 10.1109/NOMS.2014.6838368

[14] Matthias Thoma el el., "Rest-based sensor networks with OData," in Wireless On-demand Network Systems and Services (WONS), 2014 11th Annual Conference, vol., no., pp.33,40, 2-4 April 2014 doi: 10.1109/WONS.2014.6814719

[15] Simone Cirani et el., "A Scalable and Self-Configuring Architecture for Service Discovery in the Internet of Things," in Internet of Things Journal, IEEE, vol.1, no.5, pp.508,521, Oct. 2014 doi: 10.1109/JIOT.2014.2358296