Poster Abstract: Enabling Plug-n-Play for the Internet of Things with Self Describing Devices

Vasanth Rajaraman, Prasant Misra, Kumaresh Dhotrad, Jay Warrior
Robert Bosch Centre for Cyber Physical Systems, Indian Institute of Science, Bangalore, India
{vasanth.rajaraman, prasant.misra, kumaresh.dhotrad, jay.warrior}@rbccps.org

ABSTRACT

The primal problem with the Internet of Things is the lack of interoperability at various levels, and more predominately at the device level. While there exists multitude of platforms from multiple manufacturers, the existing ecosystem still remains highly closed. In this work, we propose SNaaS or Sensor/Network as a Service: a service layer that enables the creation of the plug-n-play infrastructure, across platforms from multiple vendors, necessary for interoperability and successful deployment of large-scale systems. We present the design and implementation of SNaaS, along with preliminary microbenchmarks.

Categories and Subject Descriptors

C.2.4 [**Distributed Systems**]: Distributed applications

General Terms

Design, Algorithms, Experimentation, Measurement

Keywords

Internet of Things, Plug and Play, TEDS, IEEE 1451

1. INTRODUCTION

The Internet of Things, or the IoT is a vision for a ubiquitous society wherein people and "Things" are connected in an immersively networked computing environment. Estimates on the spread of IoT over the next few years is in the neighborhood of 50 billion or more connected "Things". "Things" or physical modules/platforms contain the interface to the physical world. There exists a large number of sensor products, from multiple manufacturers, to measure various physical parameters such as temperature, pressure, humidity, illumination, acoustics, motion, location, touch, etc.,. These devices range in heterogeneity and complexity and interface with different proprietary communication technologies over a wide variety of protocols. IoT, in principle, should be an open ecosystem with high levels of interoperability from the lowest (i.e., device) level to the highest (i.e., application) level. However, the current look of IoT is extremely *closed* with an ecosystem that is highly *locked-in* by vendors, especially at the device level.

In order to overcome this fundamental limitation, we propose SNaaS. Expanded as Sensor/Network as a Service, it enables the creation of the plug-n-play infrastructure (across platforms from multiple vendors) necessary for interoperability and successful deployment of large-scale systems. The enabling feature for plug-n-play is a transducer electronic data sheet (TEDS) that is used to describe the physical elements such as transducer identification, calibration, correction data, measurement range, and manufacturer related information, etc.,. In this work, we discuss the overall architecture and implementation details of SNaaS, and present preliminary microbenchmarks.

2. THE DESIGN OF SNaaS

2.1 Self describing devices with TEDS

TEDS is the electronic version of the data sheet that is used to configure a sensor. TEDS bring forward the concept that if the data sheet is electronic and can be readily accessed upon sensor discovery, it would be possible to configure the sensor automatically. This is analogous to the operation of plugging a mouse, keyboard, or monitor in the computer and using them without any kind of manual configuration. Thus, in principle, TEDS enables self-configuration of the system by self-identification and self-description of sensors and actuators (i.e., plug-and-play).

The IEEE 1451 [1] standard defines TEDS as a collection of multiple sections (such as sensor identification, calibration, correction data and manufacturer-related information) to form a generic electronic data sheet. IEEE 1451 divides the system into two general categories of devices: (1) transducer interface module (TIM), which provide the necessary interfacing mechanism/functional units (such as the physical interface, signal conditioners, ADC/DAC convertors) for sensors and actuators; and (2) network capable application processor (NCAP), which acts as the gateway between the TIM and the external network. TEDS can reside in TIM's memory, or at a central repository that is accessible by the NCAP from the network (referred to as Virtual TEDS). The standard mandates the following four TEDS sections.

- **Meta TEDS.** It contains referencing codes (for unique identification), different timing parameters (to detect the responsiveness of the TIM), and other information pertaining to the available transducer channels.
- **TransducerChannel TEDS.** To enable proper function-

IPSN '15 Apr 14-16, 2015, Seattle, WA, USA

ACM 978-1-4503-3475-4/15/04
http://dx.doi.org/10.1145/2737095.2742927

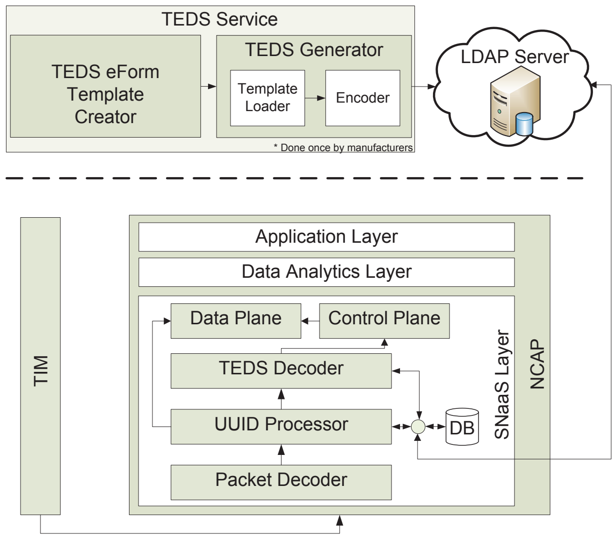

Figure 1: **SNaaS: plug-n-play system architecture.**

ing of the addressed transducer channel, various operational parameters (such as type, physical units of measurements, operating ranges, measure of various delays, etc.,) are specified in this TED.

• **User's Transducer Name TEDS.** It contains the system referenced name (such as the model number) of the transducer.

• **PHY TEDS.** It contains information about the physical connection between the TIM and the NCAP.

The other TEDS section such as the calibration TEDS, Transfer Function TEDS, Frequency Response TEDs, etc., are optional; however, they can be used to increase the richness of the self-descriptive features of the transducer. In the following section, we present the architecture of SNaaS, and use IEEE 1451 with certain design changes for IoT systems.

2.2 System architecture

The plug-n-play system architecture consist of four components: (1) TEDS Service, (2) Lightweight Directory Access Protocol (LDAP) Server, (3) NCAP, and (4) TIM.

TEDS service and LDAP server. The online TEDS service is a web application package that is used to generate TEDS for TIMs, and comprises of two modules: (1) Template Creator (TC) and (2) TEDS Generator (TEDS-G). This information needs to be entered by the manufacturer *once*. Manufacturers can use the TC to generate a TEDS e-form template with fields related to the configuration parameters of TIMs. TEDS-G loads the e-form template, and allows the manufacturers to include details about the TIM. On submission, a XML file with the TIM description is generated and provided to the TEDS Encoder (TEDS-E) that encodes the TIM parameters into Type-Length-Value (TLV) based binaries (as per the IEEE 1451 standard TEDS data block representation). The encoded TEDS binary is stored in the LDAP server (directory service) with the TIM UUID as the key.

TIM and NCAP. TIMs with larger memory, communication bandwidth and continuous power source can have TEDS within its memory, and it can be received by the NCAP on request. For resource constrained TIMs, the idea of Virtual TEDS can be used for enabling self-describing capability. SNaaS, a service layer that offers a conduit for streaming data from distributed physical sensors, is implemented in the NCAP for plug-n-play functionality. NCAP listens on the specific medium of communication for TIM association. The Packet Decoder (PD) parses the communicated information packet, following which the UUID processor checks with the local cache layer for TEDS history about the corresponding TIM. If the information is not available locally, the LDAP server is queried with TEDS UUID as the key to obtain the binary TEDS followed by an update to the local cache. The decoded TEDS binary is passed to the control plane. It is responsible for managing the sensor/network and providing a standardized life-cycle for the discovery, configuration and use of the channel models (e.g., event driven, sample and hold, etc.,). Henceforth, the data plane provides the necessary abstraction for streaming the sensed information.

3. MICROBENCHMARKS

We have developed the SNaaS service layer for Android and Linux based NCAPs, and it was experimented with two types of TIMs: (a) custom designed CC2540*Cheep* and (b) Arduino. CC2540*Cheep* uses Bluetooth Low Energy (BLE) as the communication primitive with the Android NCAP, while Arduino uses USB Type B wired communication mode with the Linux NCAP. Packets (containing the UUID of the TIM) were sent every 100 msec.

Our preliminary microbenchmarks on latency measurements (for 50 iterations) for the linux NCAP are presented in Table 1. Two specific cases were considered for this analysis: (a) the binary TEDS is available in local cache, and (b) the local cache is empty. For Case A, since the binary TEDS resides in local cache, the total latency is only 4600 μs. For Case B, the time taken for fetching the binary TEDS from the LDAP server, storing it in the local cache and decoding takes $900,000$ μs. It is to be noted that the LDAP is queried only *once* for a particular TIM, after which the local cache of the NCAP is updated.

Table 1: Modulewise latency for Linux based NCAP

Module	**Case A (μs)**	**Case B (μs)**
Packet Decoder	1300	1300
UUID Processor	2	2
Local Cache	3300	1500
LDAP Query	0	897000
TEDS Decoder	3	3
Total	**4600**	**900000**

4. FUTURE WORK

In further enhancements to this work, we are working towards combining semantic capability and data push/pull (with a publish/subscribe model) to allow "Things" to be assembled into a semantically linked data flow graphs. Thus, it would allow the SNaaS to act as a registry for semantic discovery and linkage of "Things", and for virtualizing the real world.

5. REFERENCES

[1] IEEE Standard for a Smart Transducer Interface for Sensors and Actuators - Common Functions, Communication Protocols, and Transducer Electronic Data Sheet (TEDS) Formats. *IEEE Std 1451.0-2007*, pages 1–335, Sept 2007.