Journal of Network and Computer Applications xx (xxxx) xxxx–xxxx

Contents lists available at ScienceDirect

Journal of Network and Computer Applications

journal homepage: www.elsevier.com/locate/jnca

Enabling synergy in IoT: Platform to service and beyond

Michael P. Andersen, Gabe Fierro*, David E. Culler

Electrical Engineering and Computer Science, UC Berkeley, United States

ARTICLE INFO

Keywords:
Internet of Things
Wireless sensor networks
Sensor motes
Middleware
Security
Trust
Embedded operating systems
Publish-subscribe

ABSTRACT

To enable a prosperous Internet of Things (IoT), devices and services must be extensible and adapt to changes in the environment or user interaction patterns. These requirements manifest as a set of design principles for each of the layers in an IoT ecosystem, from hardware to cloud services. This paper gives concrete guidelines learned from implementing and deploying a full-stack synergistic IoT platform. We address hardware design concerns and present a reference platform, Firestorm. Upon this platform, we demonstrate firmware and personal-area networking concerns and solutions. Moving out towards larger scales we address local service discovery and syndication, and show how these principles carry through to global operation where security concerns dominate.

1. Introduction

The Internet of Things that we imagine involves far more than the ability of many miniature computational devices embedded in the fabric of everyday life to communicate. We expect that these devices will be specialized in ways reflecting the "thing" they are a part of, that distinctive ensembles of connected things will provide rich functionality as natural-to-use applications and services, that space and proximity matter, as they dictate context and delineate boundaries of applicability, trust, and authority, and that all of this will leverage the deep storage and processing resources of the cloud, as well as its potentially global visibility. This is a fundamentally heterogeneous world, and yet we imagine seamless, nearly spontaneous interactions among diverse collections of things working together – in a word, *synergy*.

And yet, the prelude to the IoT we see all around us today stands in stark contrast to this conception. While the smartphone is ubiquitous and wearable devices are everywhere, almost invariably for one to work with the other an application for the particular thing must be preloaded onto the phone and the two devices must be explicitly paired using a particular, common link and protocol. The situation is no better with Zigbee or z-wave "things", essentially each requiring a product-specific gateway and unable to interact with the phones and wearables of the BLE (Bluetooth Low Energy) universe – despite immense effort to develop detailed application profiles. WiFi scarcely improves the situation, despite inheriting the ability for a device from any vendor to communicate with any other; we still need, for example, a dedicated application for the phone to interact with the thermostat – or resort to interacting with its web-accessible avatar. Certainly the phone serves as an intermediary in many ways, possessing PAN, LAN, and WAN links, but strangely this largely means isolated vendor-specific stacks pass through it.

This situation led us to develop a complete IoT system in which to explore the issues of synergy at various levels, as illustrated in Fig. 1. Simply applying the techniques associated with "the Internet" is not enough. We do need end-to-end communication amongst devices using distinct physical links, but the proximity and relationship of those devices matter, so decoupling through bridges and routers is not enough. We do need devices to access content provided by other devices in a uniform manner, but the path identifying that information is largely determined by the relationships between the providers and their context. Notification, events, and APIs are the norm, rather than `GET`-ting documents. Applications might be viewed as mash-ups of physically dependent services, but there also needs to be a principled way to assemble those ensembles from context.

This paper seeks to bring to the fore the key issues encountered in the quest to achieve synergy in the IoT. These issues arise at every level and they have some commonality. Enclosing complex, highly specialized behavior behind a simple interface is key. Service descriptions must be more than a specification to permit vendor interoperability; they should permit bootstrapping from context to avoid pre-configuration and should be accessible independent of the link between things, yet should not prohibit peer-to-peer interaction. Relationships should be extracted from metadata, and hence things should be notified as such metadata changes. "Middleboxes" have important roles to play, including bridging, adaptation and routing amongst heterogeneous

* Corresponding author.
E-mail addresses: m.andersen@cs.berkeley.edu (M.P. Andersen), gtfierro@cs.berkeley.edu (G. Fierro), culler@cs.berkeley.edu (D.E. Culler).

http://dx.doi.org/10.1016/j.jnca.2016.10.017
Received 17 June 2016; Received in revised form 17 October 2016; Accepted 24 October 2016
Available online xxxx

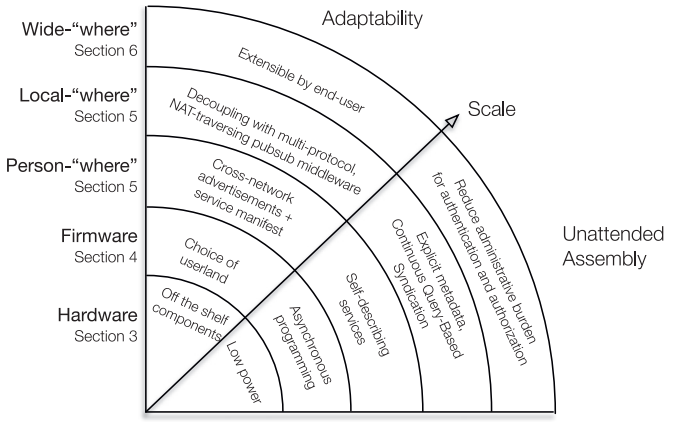

Fig. 1. An effective IoT solution requires addressing adaptability and unattended assembly and operation at every tier. *Adaptability* means the ability of a device, service or application to (a) detect and react to change in its environment and (b) support changes in its use case. *Unattended assembly* means the ability of a collection of devices, services and applications to discover each other, identify relevant resources, and operate at length without constant administration by a human supervisor.

technologies, but also establishing points of presence and boundaries of trust.

We begin with a very brief description of our exploratory system to provide a concrete framing of the study. The remainder is organized in layers: first hardware, then firmware, then three scopes of interaction which we term person-where, local-where and wide-where – expanding conventional notions of PAN, LAN, and WAN[1] to include the interactions and services that occur in those domains.

2. Synergy

The system architecture developed to explore synergy in IoT is shown schematically in Fig. 2.

Hardware: At the device level, we have introduced a new platform that brings together embedded wireless networking, wearables and "Maker" developments, typified by IEEE 802.15.4, BLE and Arduino peripherals, respectively (see Andersen et al., 2016a for a complete description). This platform is built around the *Storm* module, designed in 2013, with a Cortex-M4, 802.15.4 radio, and flash, mounted on an Arduino-compliant carrier, *Firestorm* (Fig. 3), which provides BLE communication with a Cortex-M0+ SoC,[2] and several sensors. This design point was intended to capture what embedded IoT might converge to, and, indeed, now several system-on-chip offerings provide ample flash, powerful MCUs[3] and 802.15.4 or Bluetooth radios. Storm is extremely low power (2.3μ in sleep with RTC[4] active) and supports many peripherals (63 GPIO pins). This adds to the commercial ecosystem of wearables, embedded Linux boxes (Raspberry Pi's and Beaglebones), BLE tags, and such.

Firmware: An extension of TinyOS (Levis et al., 2005) was developed for this platform that utilizes the newly available protection mechanisms (MPUs) to establish a clear user/system boundary in the domain of networked things with fewer computational resources than a Raspberry Pi or Beaglebone. A syscall interface and language runtime was developed that exposes low-power, event-driven execution to user level and two language environments were created – Lua and C++. This allows instantiation of a breadth of user-programmable "things" not represented in the commercial landscape.

To demonstrate the power of a dynamic Lua userland, we explore the creation of an active networks-inspired platform for wireless

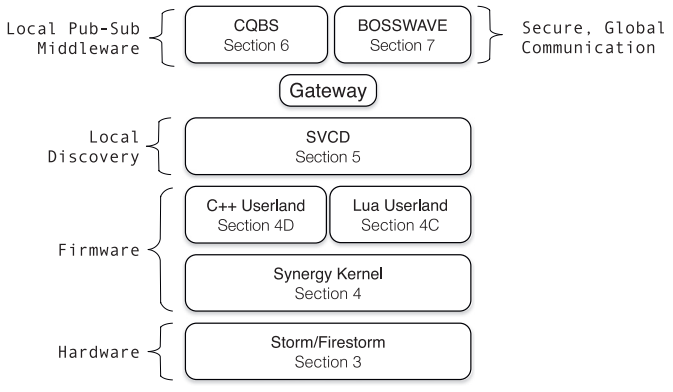

Fig. 2. An IoT system architecture.

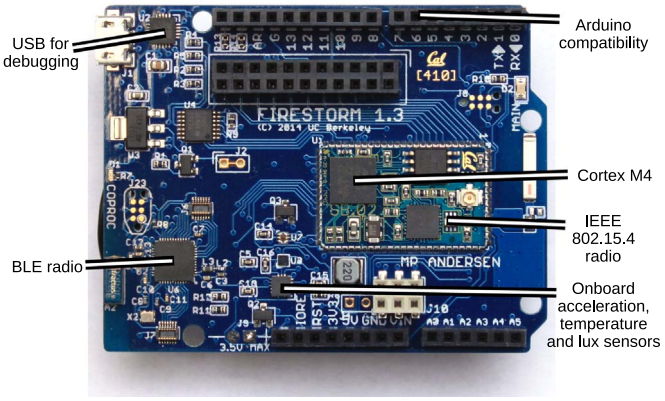

Fig. 3. The Firestorm platform.

network measurement. The platform extends the syscall interface for visibility into the networking and radio stacks, and leverages an embedded Lua interpreter to introduce new logic and replace symbols in running code without the need to re-flash. From our experiences building production-ready firmware using a dynamic userland, we develop a C++11 userland that also provides efficient event-driven execution of asynchronous code but in a static, resource-efficient language.

Person-where: In this setting, we developed a self-describing service tier to enable automated assembly of purpose-driven ensembles at the scope of the individual person, which subsumes phones interacting with things in the environment and wearables interacting with spaces and things. Such assembly must not require pre-configured applications and bindings. Things project their API and services in a manner that allows the phone to bootstrap itself into the context, using complete descriptions in the cloud. This scope retains a sense of individual management: self-assembly in this setting cannot assume the existence of coordinating infrastructure, so discovery and interaction must be able to perform in a "peer-to-peer" manner. We describe *SVCD*, a brokerless, local publish-subscribe mechanism that is symmetric over both IPv6/802.15.4 and BLE. We construct a simple, asynchronous API for SVCD and use this to construct a self-assembling "smart space heater" ensemble.

Local-where: In a similar manner, collections of things can assemble into federated ensembles to provide services and interfaces that expand that of the individual devices. This relies on discovery of context and interfaces through queries to consistent metadata. Services and enclosing applications are associated with place, rather than person, and must operate unattended, requiring automated notification of changes as represented in the metadata. Local tier routing and computing resources bridge and translate heterogeneous elements. We explain how a new "flavor" of publish-subscribe – continuous query-based syndication – can enable applications to adapt to changing

[1] Personal-area, local-area and wide-area networks, respectively.
[2] SoC: system on chip.
[3] MCU: microcontroller unit.
[4] RTC: real-time clock.