SmartThings: Overview

Table of Contents

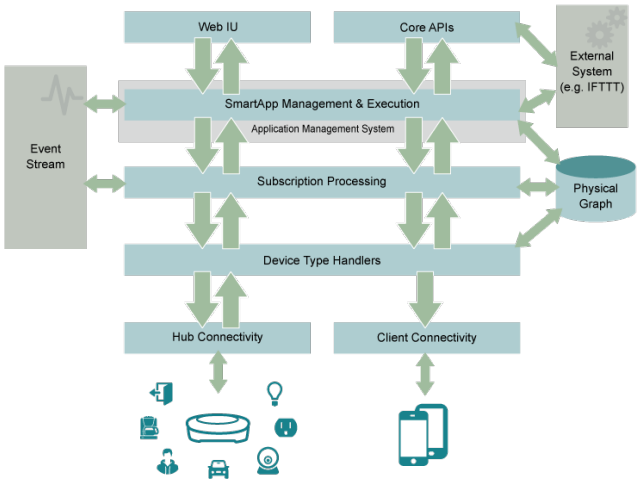
1 Overview

- **Target domain**: Home automation
- **devices** provide native capability
- **composition of native device services** happen at **CLOUD**, FOR NOW!
  - they are thinking of moving some capability (i.e intelligent logic or choreography code to Hubs)
  - c.f. Meadow is flexible enough to put "composition logic" into anywhere (cloud, hub, or device itself).

1.1 What is it?

- SmartThings: **platform for "Open Physical Graph"**
- **physical graph**: virtual, online representation of physical world
- people interact with physical graph
- graphs are dynamic – nodes come and go, users can perceive dynamicity of graphs
- graph is hierarchical – graph of graphs, etc.

2 Architecture



2.1 Components

- **end-devices**: connected to SmartThings Hub (e.g. ZigBee devices) or directly connected to Cloud (e.g. devices which directly connected to TCP/IP)
- **SmartThings Hub**: network gateway which connects ZigBee/Z-Wave network to the Internet + some event communination capability between CLOUD and devices
- **SmartThings Cloud**: centralized server
- **User Experience**: just U/I programs (e.g. iPhone, Android app) which uses some API which accesses Cloud

2.2 SmartThings Cloud Functionalities

- **connectivity**: provides connection between Hubs and Apps

  `devices - (Hubs) - CLOUD - (iPhone) - apps`

- **device management & capability abstraction**:
  - some notion we abandoned at early stage
  - should we create some "first-class" treatment of device categories? (e.g. "switches" abstract all "GE switch", "logitech switch", etc.)
  - maybe not a good idea
  - we may need to provide "MECHANISM" but let users to implement "POLICY" using the meachanism
  - Some "overlaying" tag-based mechanism (e.g. Gmail tag) could be a better alternative
- **Event processing & routing**: event pub/sub layer + some limited event processing (filtering, etc.)
  - trivial; rather bland
- **Applications (SmartApps)**: just provides some API which external programs can access cloud

  - trivial

- **Web services**: provides the web service API so Web developers can be used
  - trivial

2.3 Current status

- Supported protocols: ZigBee, Z-Wave, WiFi, etc.
- Sells two types of kits:
  - Kit #1: Only observe device status
  - Kit #2: Both observe and control devices
- When service request is sent, "fire and forget" (asynchronous call) – no service guarantee
- cloud-based: **centralized server**

2.4 Future plan

- from cloud to distributed hub-based
- support blocking service request

3 How They Support Adding Smartness

- **device-type handlers**: some sort of "device driver" which connects physical device and the "SmartThings world"
  - **maybe we might need to adopt this idea; so far, we have thought about some "import capability" to import native device capability into Meadow; however, more solid "abstraction" is needed**
- **event-handler SmartApps**: just some primitive-form of **Meadow reactors**
- **Dashboard Solutio Mobile SmartApps**: some U/I tool (U/I version of Meadow shell?)
- **Integration SmartApps**: SmartApps can **use/provide web services**, which means that they would provide some runtime which provides services but only in web services form.
  - need to compare our approach (which provides own language)
  - DISCUSS: is "web services" a technology worth pursuing? (losing to REST API)
  - need to dig futher

4 Event handlers

```
def contactHandler(evt) {
  log.debug "$evt.value"
  if (evt.value == "open") {
    switch1.on()
  } else if (evt.value == "closed") {
    switch1.off()
  }
}
```

5 Criticisms

- **Hard-coded device types**
- **Groovy for describing event handlers**
- **Supporting concurrency**:
- **Lack of blocking calls**: Supporting blocking calls requires sophisticated scheduling mechanism installed inside the "collective" runtime. Let a single event handler H be executed. They may just schedule one Java thread to handle this. When blocking calls are supported, there may be huge number of threads pending in the middle (waiting for reponses) – Java thread based scheduling cannot scale.