Panasonic

Treatment of Implementation

As described in scope of TF-AP, implementation is treated as “reference” for mapping WoT servient to physical device.

But, during discussion, everyone talks based on each own implementation experience, then the discussion sometimes seems to be confused.

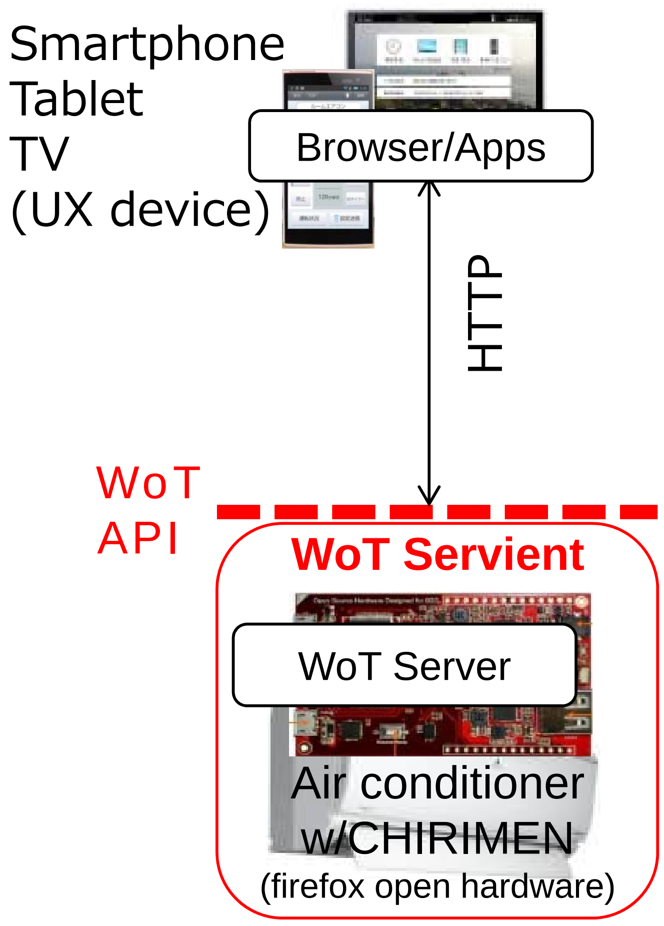
I’d like to propose to try to make it clear that
- listing up implementation models and WoT servient mapping
- confirmation on what kind of I/F our TF-AP should define as API

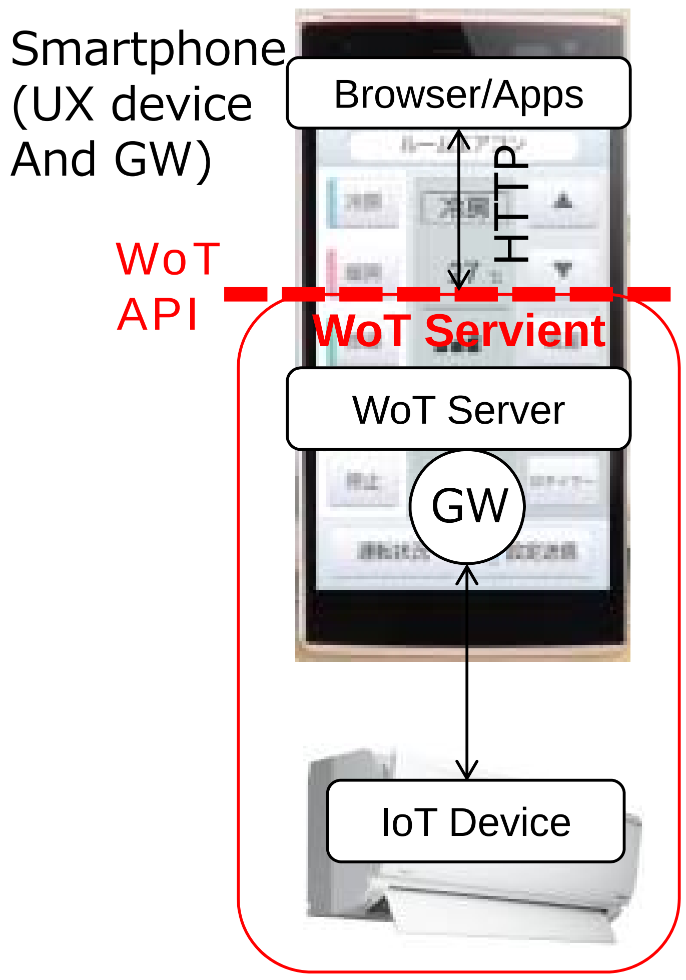
In next 2 slides, I try to figure out 4 different type of implementation of WoT servient.
(a) Web centric implementation, that is, a physical device itself is WoT Servient.
(b) Smartphone centric implementation, that is, smartphone connects to a physical device and smartphone includes both UX apps and GW.
(c) Hub centric implementation, that is, hub connects to a physical device and hub includes GW and provides REST API to other UX devices.
(d) Cloud centric implementation, that is, hub connects to a physical device and includes GW, cloud binds hub and UX devices and provides REST API.

**GW don’t have to be physical box. GW is the logical module providing following functions.*
- media conversion (ethernet / WiFi / 802.14.4x / BTLE /…)
- protocol conversion (HTTP / CoAP / IEEE1888 / Alljoyn / Echonet Lite / ZigBEE /…)

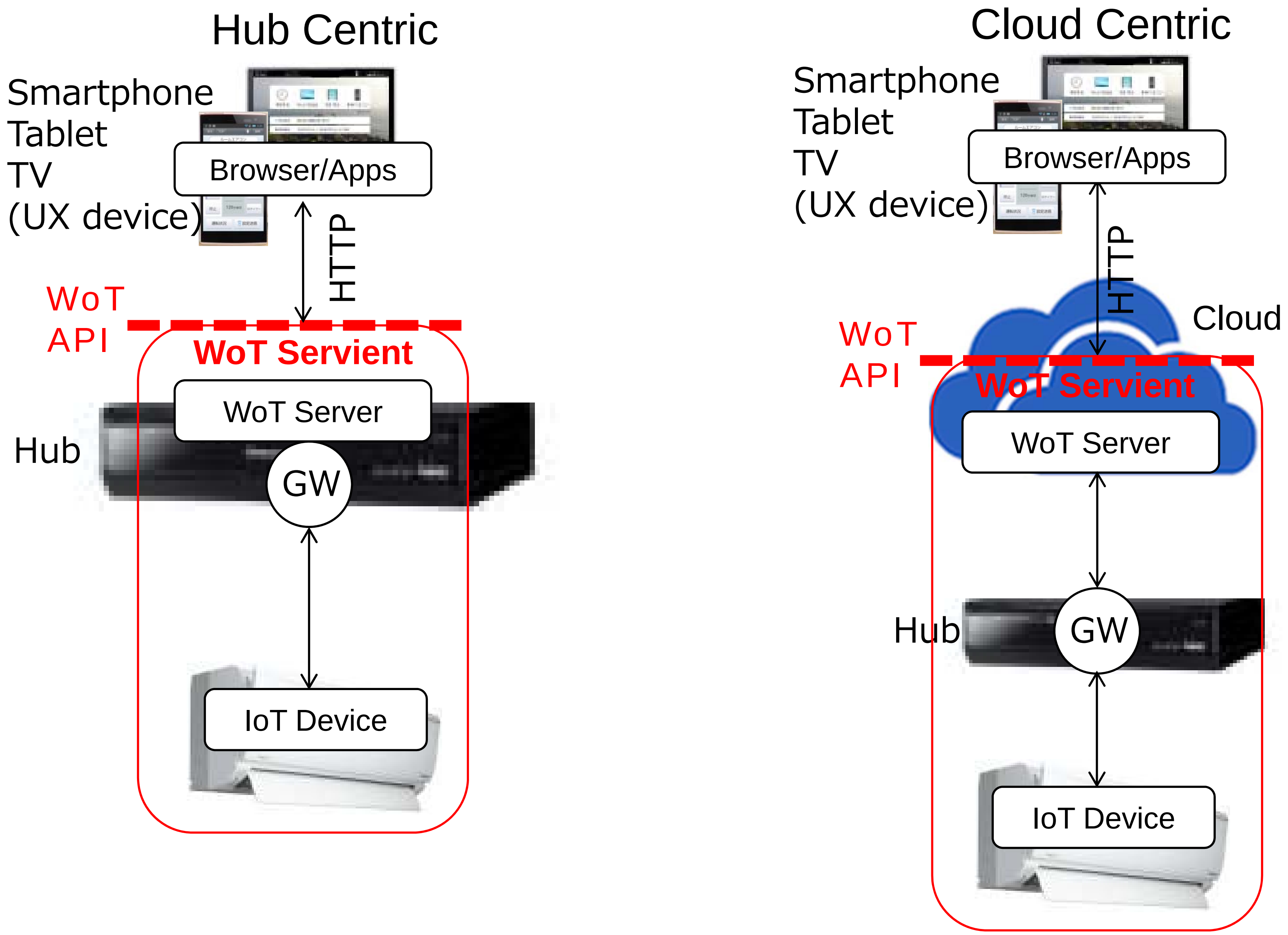
Panasonic

Implementation Model (1)



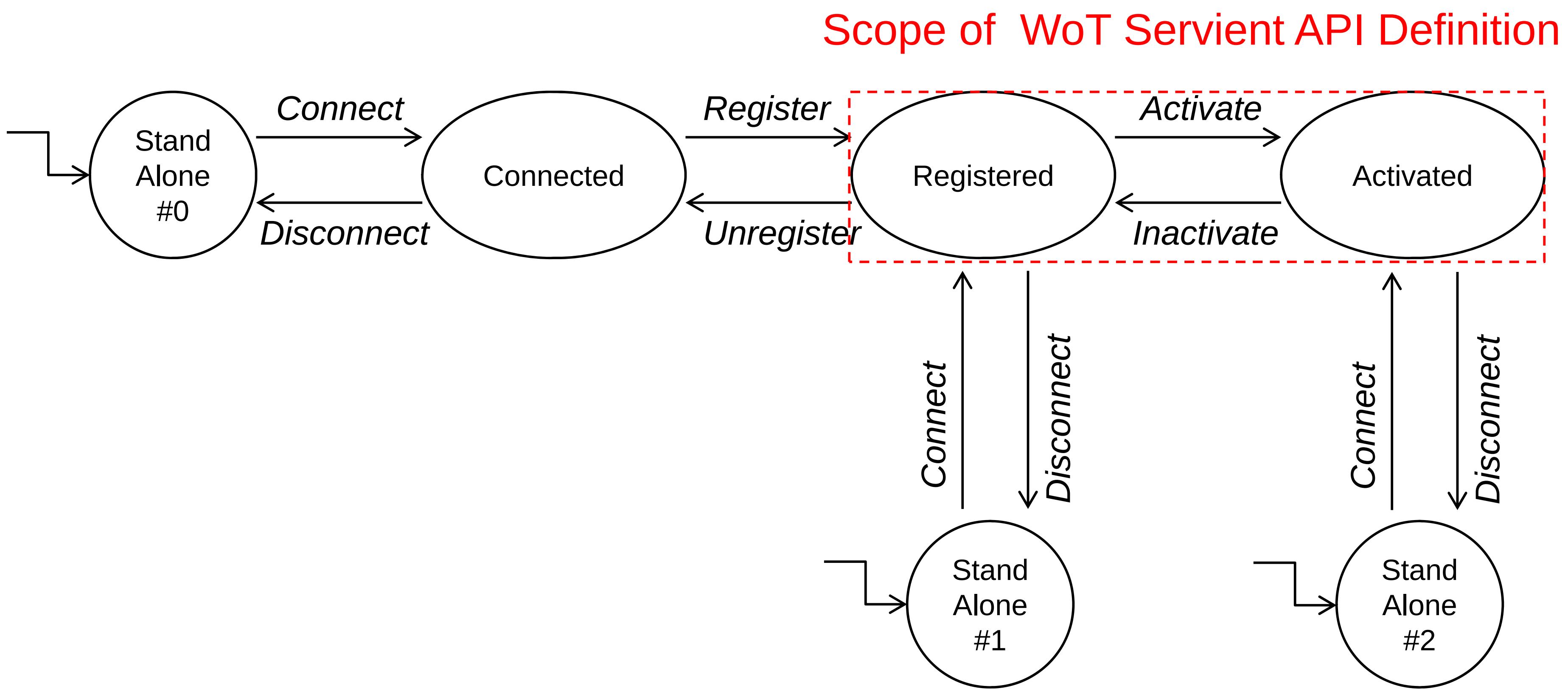


Implementation Model (2)



Implementation models are different, but WoT API can be defined as same API.

WoT Servient Lifecycle (State Diagram)



“Stand Alone”:	not connected status
“Connected”:	just connected to IP reachable network, but NOT worked as WoT servient
“Registered”:	URI-like address is registered and worked as WoT servient
“Activated”:	powered on the device and support full service of “things” as WoT servient

Panasonic

WoT Servient API classification

“Activate/Inactivate”:	Power On/Off concept
“Get”:	Get Things Description Get Real Sensor Values Get Name tags list (like dynamic IOCTL concept)
“Put”:	Update Things Description Set Actor Parameters and Control Actor itself
“Notify”:	Bind callback function (if any) embed callback condition description thru Put API