Kadecot: Android Web API Server for Home Appliances and Sensors

Shigeru Owada, Sony Computer Science Labs, Inc.
Kazuhito Nakamura, Sony corp.
Masahiro Karaki, Crestec Inc.
Talk Summary

• We develop a Web API server ‘Kadecot’ runs on Android for home appliances/sensors

• It internally adopts WAMP-based framework for managing the whole system.

• We develop a 3D agent-based user interface on top of them.
WAMP : The Web Application Messaging Protocol

WAMP is an open WebSocket subprotocol that provides two application messaging patterns in one unified protocol: Remote Procedure Calls + Publish & Subscribe.

• Actually, WAMP does not require using WebSocket. Other transports are possible if some requirements are met.
  – We use both WebSocket connection and function call as the transport

Copied from WAMP HP http://wamp.ws/
System architecture

Applications
- Web API Users
  - WAMP API
  - Protocols manager

Server
- Android OS

Home Appliances and Sensors
System architecture

Applications

Web API Users

Server

WAMP API

Protocols manager

Home Appliances and Sensors

Android OS
Goal of Server Design

• Device communication protocol independent

• Flexible, Easy-to-use WebAPI

• Use public messaging protocol
Definition

- Resource: a piece of information or a controlled property in a device
  - Eg. Power, Temperature setting, Brightness sensor value
WAMP messaging patterns

**RPC pattern**
- Active resource request from consumer (‘Caller’ in WAMP terminology) to provider (Callee)

**PubSub pattern**
- Asynchronous resource delivery from provider (Publisher) to consumer (Subscriber)
WAMP in-depth

• WAMP is NOT a one-to-one messaging architecture
• There is another entity, called a ‘Router’, deals with message deliveries.
• All other entities are called ‘Client’
• ‘Caller’, ‘Callee’, ‘Publisher’, ‘Subscriber’ are compatible Roles of each client.
  – They switch roles on types of messages

![Diagram with roles of Clients and Router](image)
Our Home Server Architecture

- **Web App #1**
  - Caller & Subscriber

- **Web App #2**
  - Caller & Subscriber

- **Web App #3**

- **Device Manager**
  - Callee

- **Polling Manager**
  - Publisher

- **Router**

- **Protocol #1**
  - (ECHONET Lite)
  - Callee & Publisher & Subscriber

- **Protocol #2**
  - (Sony protocol)
  - Callee & Publisher & Subscriber

- **Protocol #3**

- **White Goods / Sensors**

- **Black Goods**

Confidential – need to know basis only / do not distribute
Our Home Server Architecture

Web App #1
Caller&Subscriber

Web App #2
Caller&Subscriber

Web App #3

...  

Router

Device Manager
Callee

Polling Manager
Publisher

Protocol #1
(ECHONET Lite)
Callee&Publisher&Subscriber

Protocol #2
(Sony protocol)
Callee&Publisher&Subscriber

Protocol #3

...

White Goods / Sensors

Black Goods

Confidential – need to know basis only / do not distribute
Device Manager

• Receives devices list information from protocols and assigns unique ID to each device

• Keeps all recognized devices list and their available resources as a cache
  – Provides fast replies on..
    • Devices ability query from the apps
    • Information recovery on reboot
Our Home Server Architecture

Web App #1
Caller&Subscriber

Web App #2
Caller&Subscriber

Web App #3

Router

Device Manager
Callee

Polling Manager
Publisher

Protocol #1
(ECHONET Lite)
Callee&Publisher&Subscriber

Protocol #2
(Sony protocol)
Callee&Publisher&Subscriber

Protocol #3

White Goods / Sensors

Black Goods

Confidential – need to know basis only / do not distribute
Polling requirement

• Polling is necessary to let passive sensors act as a publisher

• However, in IoT environment, it is not desirable to poll all available resources

• Even if a resource is subscribed by multiple apps, the polling should not be multiplied.

![Diagram showing the polling requirement and its implications for multiple apps accessing a resource through a polling manager.](image-url)
Polling Manager

• Our solution
  – Polling manager publishes the counter how many apps are interested in each resource.
  – Each protocol subscribes the related counter to determine which resource value should be polled.
  – Polling interval is internally determined within protocol client.

• This violates WAMP framework, since information about subscription exists in the router (router cannot be a callee nor a publisher)
Our Home Server Architecture

Web App #1
Caller & Subscriber

Web App #2
Caller & Subscriber

Web App #3

Protocol #1
(ECHONET Lite)
Callee & Publisher & Subscriber

Protocol #2
(Sony protocol)
Callee & Publisher & Subscriber

Protocol #3

Device Manager
Callee

Polling Manager
Publisher

Router

WebAPI

White Goods / Sensors

Black Goods
WebAPI

• Web (and other) application participate in the system through standard WAMP protocol
  – Such as ‘HELLO’, ‘SUBSCRIBE’, etc..

• Tips:
  – Resource types are represented as topics/procedures string
  – Device ID (assigned by Device Manager client,) is specified as ‘Options’ field in WAMP

Eg)
[CALL,1,{"deviceID”:1},”com.sonycsl.kadecot.provider.procedure.getDeviceList”]

, in the format of RPC messsaging : [CALL, Request|id, Options|dict, Procedure|uri]
Our Home Server Architecture

Web App #1
Caller&Subscriber

Web App #2
Caller&Subscriber

Web App #3

Protocol #1
(ECHONET Lite)
Callee&Publisher&Subscriber

Protocol #2
(Sony protocol)
Callee&Publisher&Subscriber

Protocol #3

Device Manager
Callee

Polling Manager
Publisher

Router

DB

White Goods / Sensors

Black Goods

Confidential – need to know basis only / do not distribute
APPLICATION
Moekaden Project

• We are currently running the “Moekaden” project which tries to combine IoT service and personification / concierge characters

Personified Blu-ray recorder acts as a concierge
Demos

RPC demo

• The remote controller function uses RPC messaging

PubSub Demo

• Error notification is subscribed by the application. If an error is notified, related manuals are shown
Conclusion

• We develop a home WebAPI server with WAMP

• Efficient polling requires non-WAMP message sharing between router and polling manager

• We run Moekaden

• Announcements
  – Moekaden HP (<http://moekaden.com>) is currently under maintenance, will be open on July 5th
  – New version of Moekaden will be released on September.
  – Pure Java implementation of WAMP (not available on WAMP HP) will be soon released by us