Smart Home Android Client

Project Documentation

Developer: Andhieka Putra
Supervisor: Dr. Teddy Mantoro
Start: June 30th, 2014
End: July 18th, 2014

Scope of Project

Smart Home Android Client is designed to work with the smart home prototype in Universitas Siswa Bangsa Internasional (USBI). It is part of the smart house system as described in the paper "Web-enabled Smart Home Using Wireless Node Infrastructure"[1] This Android client is represented as "Mobile user" in Figure 1, which is taken from the above-mentioned paper.

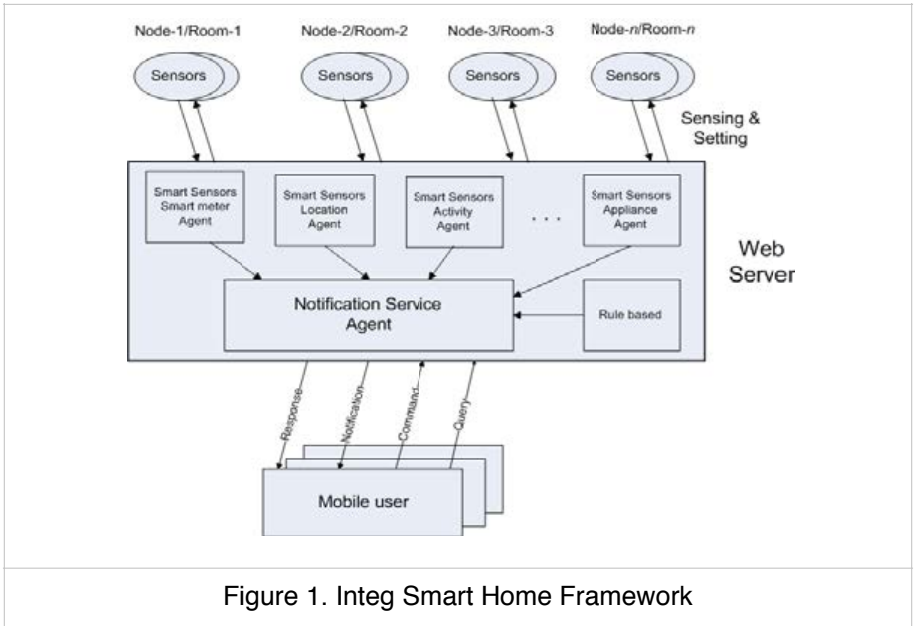

Figure 1. Integ Smart Home Framework

Previous to this project, the Web server can be controlled using a simple HTML form which is sent from the Web server every time a web browser connects. This project simply ports the same functionality into an Android application. Figure 2 shows the comparison between the HTML form and the new Android interface.

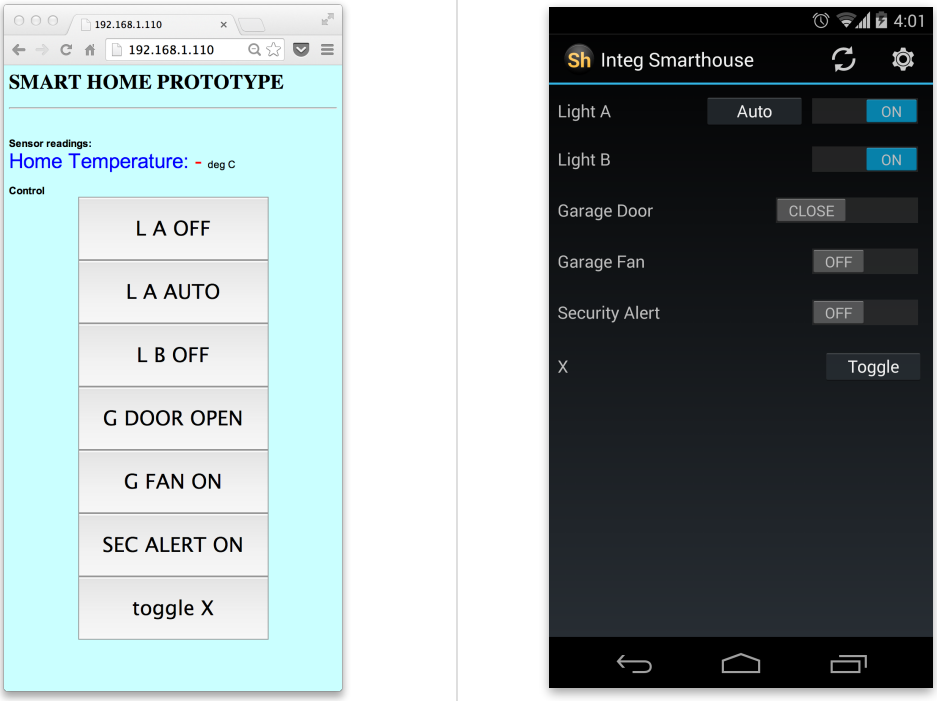

Figure 2. HTML form (left) vs. Android interface (right)

[1] Mantoro T, Ayu MA, Elnour EE. 2011. Web-enabled Smart Home Using Wireless Node Infrastructure.

Goal of Project
To create a faster and more dynamic user interface for the smart home prototype.

Project Milestones

Date	Task	Description	Progress
July 10, 2014	Back-end functions.	Study the web server code and implement the communication using socket.	100%
July 11, 2014	User Interface design	Designed the main view and settings.	70%
July 14, 2014	User Interface design	Use JSoup HTML parser to update the Android interface based on the web form sent by web server.	100%
July 15, 2014	Debug and Testing	Testing with multiple client, error handling when the connection is down, when settings are changed, etc.	100%

Smart House Android Client Tutorial

1. *Connecting to Web server*

   A. First, connect your Android device to the WLAN network provided by the smart home web server. Then, open the Integ Smarthouse application. If the connection is successful, the application will display a subtle “Refreshed” notification and you can go to step 2.

   B. If the connection to web server cannot be established, the application will display an error message “Connection error. Please correct connection settings.” Ensure that you are connected to the right network. You can tap on the Settings icon in the Android action bar to configure the server hostname and port, as shown in figure 3.

2. *Controlling the Smart Home prototype*

   Upon successful connection, you should see the list of devices, as shown in figure 2. You can refresh the view by tapping on the the Refresh icon in the Android action bar to get the latest status of the devices. This feature is useful when there are multiple mobile clients connected to the same smart home system.

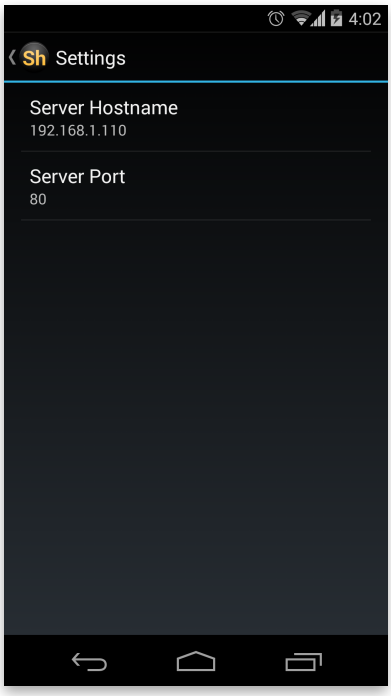

Figure 3. Settings page

Technical Details
The application is very simple, with only three main classes: DeviceListFragment, SmartHouse and WebServerCommunicator. The interaction between them is modelled in figure 4.

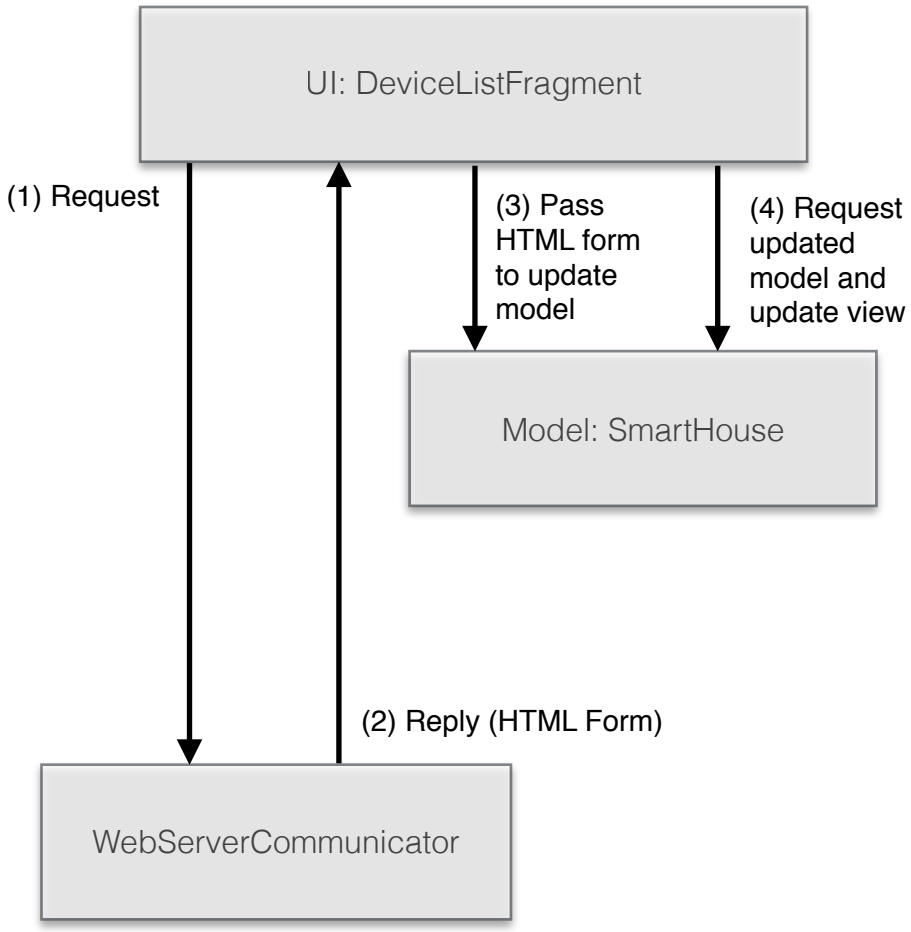

Figure 4. Interaction Diagram in Smart House Android Client

Future Improvement
The current functionalities are limited due to the restrictive syntax imposed by the Arduino-based web server. The implemented smart home states are currently binary (true-false) conditions.

A more flexible communication syntax between the web server and Android client will enable much more expressive functions. For example, the Android client can make request to the web server using SQL-like syntax and the server could transmit state information of the smart house using JSON data format, which can be easily parsed in modern programming frameworks.

I created a prototype system that demonstrates the above-mentioned improvement. The Android client is a modified version of the one presented previously in this document. The web server is programmed using Ruby, and the smart home state is simulated using an SQLite database.

Figure 6 shows the user interface of Android client built for the web server with proposed improvement.

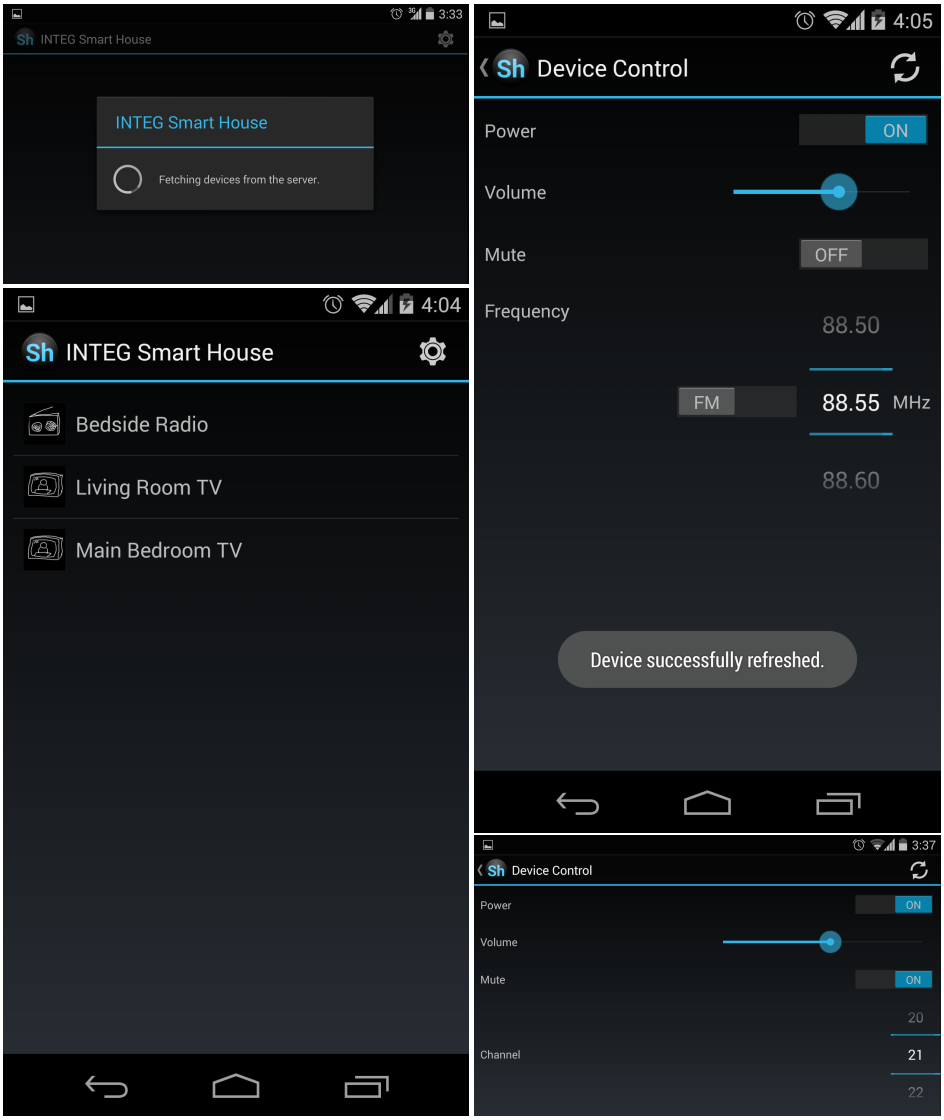

Figure 6. Clockwise, from top-left: loading page, radio controls, TV controls, device list

Technical Details for Future Improvement

The following syntax is used for interaction between client and server:

1. `GET: device_id;`
   `>> returns JSONObject containing all properties of a device.`
2. `SET: device_id; property_name; value;`
   `>> sets a property of a device. returns true if successful.`
3. `LISTDEVICES:`
   `>> returns JSONObject containing all devices with all properties.`
4. `LISTNAME:`
   `>> returns an array containing all device names.`

The smarthome is modelled in this way:

1. The class SmartHome models the smart home as a whole. More importantly, it contains a list of the devices in the smart home.
2. All devices extend DeviceBase abstract class. Moreover, they can implement one or more functionality interfaces. For example:

```
public class Radio extends DeviceBase
     implements  IPowerSwitchable,
                 IVolumeControllable,
                 IFrequencyControllable

public class WirelessLight extends DeviceBase
     implements IPowerSwitchable
```

3. Each interface specifies standard functions such as setPower, getPowerState, etc.

The user interface consist of two layouts: device list which lists all devices in the smart home, and controller list which shows all the available controls for a chosen device. The function filterControls() inside ControllerListActivity.java makes sure that only the appropriate controls are shown on the screen. This check is done using the instanceof method. For example:

```
if (!(_device instanceof IPowerSwitchable))
    findViewById(R.id.power_switch_table).setVisibility(View.GONE);
```

More details can be found in the provided Java code. The code is written using IntelliJ IDEA 13 Community Edition, Java 8 Development Kit and Android API version 19.

The web server code is also included in the submitted file. It is written in Ruby. Please read the README file for more details on how to run it.

Final remarks

If you have further question regarding this project, please contact the developer at andhieka.putra@gmail.com.