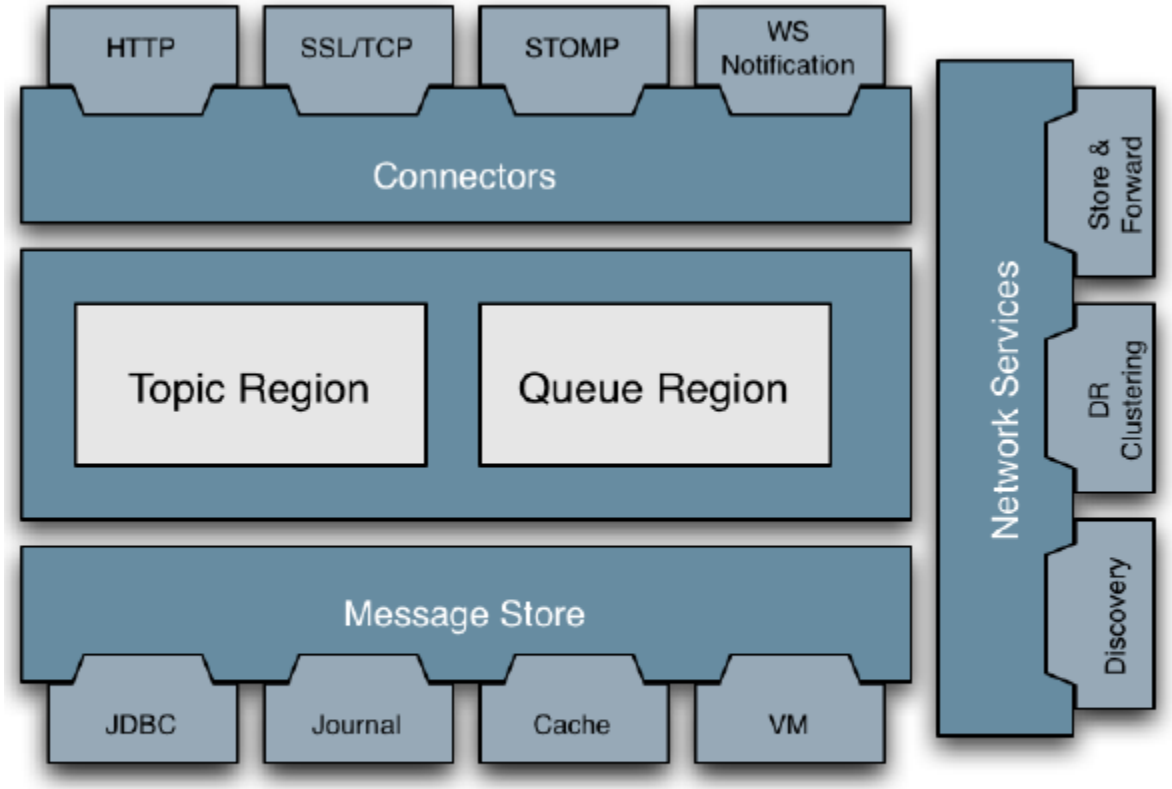
WP SmartHome
Einführung in die Living Place Architektur Kommunikation
Erste Schnittstellen

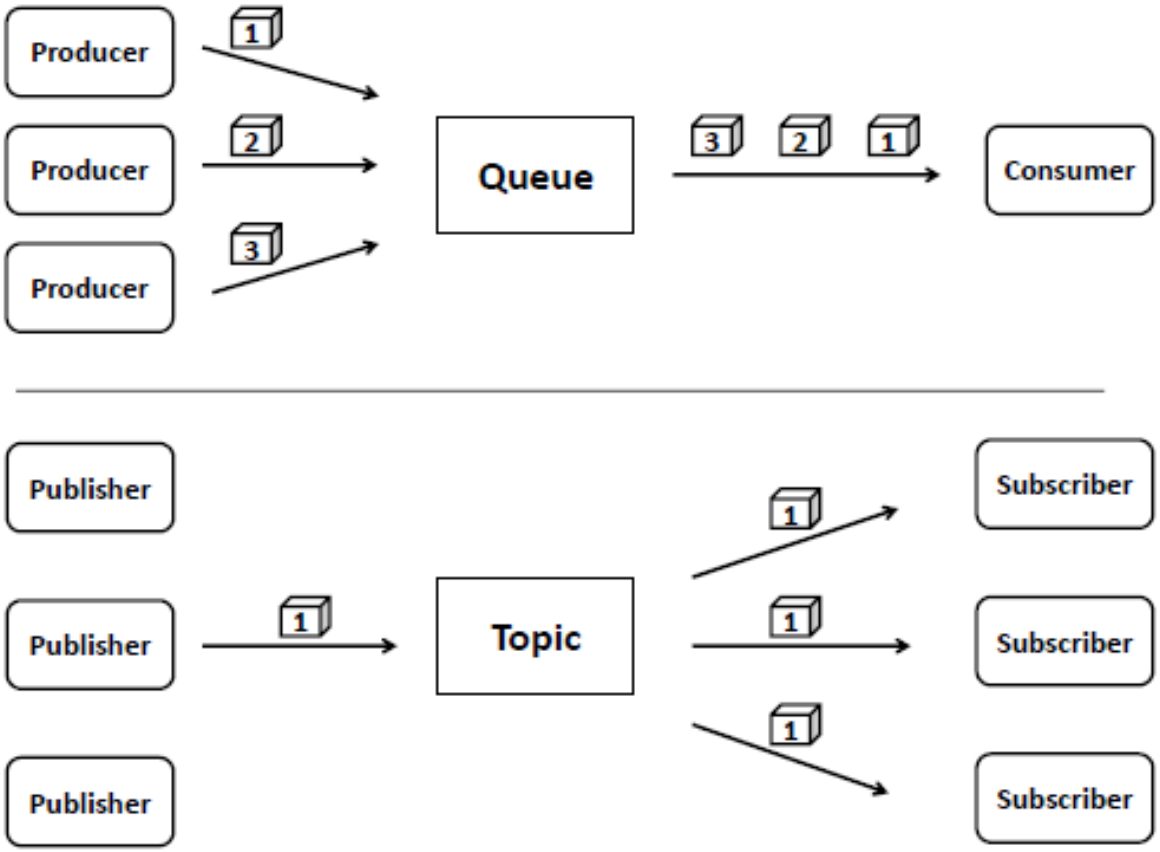
Prof. Dr. Ing. Birgit Wendholt

Blackboard Architektur Active MQ



Architektur des ActiveMQ (The Apache Software Foundation (2011))

Queues und Topics



Queues und Topics

Queues

- Producer schreiben Daten in eine Warteschlange
- Consumern nehmen Daten entgegen
- „n-zu-1“-Beziehung zwischen Producern und Consumern
- sind mehrere Verbraucher an einer Queue angemeldet, wird die Nachricht lediglich ein Mal ausgeliefert
- angemeldete Consumer bei der Erzeugung eines neuen Datums nicht erreichbar → Informationen werden so lange gespeichert, bis der Konsument erreichbar ist

Topics

- realisieren „n-zu-m“-Beziehung zwischen Publishern und Subscribern
- jeder Subscriber erhält jede erzeugte Nachricht, unabhängig von der Anzahl der Verbraucher auf dem Topic
- **durable Subscriber**: erhalten nachträglich sämtliche Nachrichten des Topics auch bei Inaktivität
- **non-Durable Subscriber**: verlieren Nachrichten während der Inaktivität

Living Place Objekt IDs (Lamps only)

Kommunikation mit Active MQ über den LivingPlace Proxy

- keine Implementierung eine JMS Clients auf Android → Verwendung eines Proxies, der die Daten für den ActiveMQ empfängt und weiterleitet.
- erreichbar über den Port 12349
- installiert auf dem Server mit IP: 172.16.0.200
- Proxy muss auf dem selben Server laufen wie der ActiveMQ, da er die Nachrichten an localhost weiterleitet
- Starten über Konsole: java -jar AndroidProxy ActivMQ_IP Proxy_Port (ist für Server 172.16.0.200 auf Port 12349 gestartet)
- Kommunikation mit dem Proxy über den Android Publisher
  - Download: http://livingplace.informatik.haw-hamburg.de/svn/android_publisher

Aufrufmethode für den Publishers


```
private class sendMessageToProxy extends AsyncTask<String, Void, String> {
// params [0] Server; [1] Port; [2] TopicName; [3] topic/queue;
// [4] message;
@Override
protected String doInBackground(String... params) {
    try {
          AndroidPublisher publisher = new AndroidPublisher(params[0],
                    Integer.valueOf(params[1]), params[2]);
          publisher.setMessage(params[4]);
          if (params[3].equals("topic")) {
                    publisher.publishToTopic();
          } else { publisher.publishToQueue(); }
    } catch (IOException e) {
          Log.e("Publisher", "Can't publish the message");
    }
      return null;
    }
}
```

Erzeugen und Aufruf des AndroidPublishers


```
@Override

public void statusChanged(Message msg) {
    String[] params = new String[5];
    params[0] = selectedTab.getServer(); // z.B. 172.16.0.200
    params[1] = Integer.toString(selectedTab.getPort()); // z.B. 12349
    params[2] = msg.getTopic(); // z.B. LP.LIGHTCONTROL
    params[3] = msg.getTyp();   // Topic oder Queue
    params[4] = msg.getMessage().toString();

    sendMessageToProxy smtp = new sendMessageToProxy();
    smtp.execute(params);

}
```

Steuerung des Lichtes


```
LP.LIGHTCONTROL  // Topic
{
"action":"Eine definierte Aktion",
"values":{"Key":"Value","Key:"Value"},
"Id":"irgendeine",
"Version":null
}
```

- Aktionen
  - Level1 Aktionen: direkter Wrapper auf die Skript-API des Pharos-Controllers
  - Level2 Aktionen: vorgefertigte Actions-Sets und benutzen die Level1 Aktionen
  - Level3 Aktionen: Meta-Actions und erlauben jede Art von Manipulation über das Übermitteln von Lua-Code

- Topic `LP.LIGHTCONTROL`

Level 1 Aktionen

- start_szene,
- stop_szene,
- stop_all_szene,
- halt_szene,
- resume_szene,
- write_to_log,
- set_light_intensity,
- set_light_color,
- clear_light_props,
- clear_all_light_props

Level 2 Aktionen

- lounge_light_on / lounge_light_off / lounge_light_color
- kitchen_cooking_light_on / kitchen_cooking_light_off / kitchen_cooking_color
- kitchen_main_light_on / kitchen_main_light_off / kitchen_main_light_color
- dining_light_on / dining_light_off / dining_light_color
- corridor_light_on / corridor_light_off / corridor_light_color
- sleeping_light_on / sleeping_light_off / sleeping_light_color
- bathroom_light_on / bathroom_light_off / bathroom_light_color
- start_szene_wakeUp / stop_szene_wakeUp / resume_szene_wakeUp / halt_szene_wakeUp

Level 2 Aktionen

- start_szene_relax / stop_szene_relax / resume_szene_relax / halt_szene_relax
- start_szene_comingHome / stop_szene_comingHome / resume_szene_comingHome / halt_szene_comingHome
- start_szene_cooking / stop_szene_cooking / resume_szene_cooking / halt_szene_cooking
- start_szene_alert / stop_szene_alert / resume_szene_alert / halt_szene_alert
- start_szene_romanticDinner / stop_szene_romanticDinner / resume_szene_romanticDinner / halt_szene_romanticDinner
- start_szene_sunset / stop_szene_sunset / resume_szene_sunset / halt_szene_sunset

Keys und Values der Lichtsteuerung

Key	Value	Default
fixture	Integer von 1 bis x, Pharos interne ID für jede einzeln adressierbare Lampe	1
szene	Integer von 1 bis x, ID für hinterlegte Szenen, die verwendet werden können	1
fadeTime	Fade für die Aktion – in Sekunden	0
intensity	Intensität des Fixtures, Integer 0 – 255	255
red / green / blue	Bei Farbaktionen Rot/Grün/Blauwert – Integer 0-255	0
msg	Message für Log etc. String	No Message given
code	Individueller Lua-Code ,String	No Code given

Aktionen und unterstütze Werte

(weitere Beispiele: http://livingplace.informatik.haw-hamburg.de/svn/LP-LightControl/doc/API%20Dokumentation.pdf)

Level 1

- set_light_color
  - fixture
  - fadeTime
  - red
  - green
  - blue
- set_light_intensity
  - fixture
  - intensity
  - fadeTime

Level 2

- kitchen_cooking_light_off
  - fadeTime
- kitchen_cooking_color
  - fadeTime
  - red
  - green
  - blue
- kitchen_main_light_on
  - intensity
  - fadeTime

Beispielnachricht Lichtsteuerung

Ziel: Farbe im Esszimmer ändern

```
{
   "action":"dining_light_color",
   "values":
   {
   "red":"255",
   "green":"0",
   "blue":"128",
   "fadeTime":"2"
   },
   "Id":"client_1578909153",
   "Version":null
}
```

Steuerung der Gardinen und Rolos

Aktionen

- `lounge_curtain_open`
- `lounge_curtain_close`
- `sleeping_hall_curtain_open`
- `sleeping_hall_curtain_close`
- `sleeping_window_curtain_open`
- `sleeping_window_curtain_close`
- `blinds_open`
- `blinds_close`

Beispiel Nachricht

- **Ziel**: Gardine zwischen Flur und Schlafzimmer schließen

```
{
   "action":"sleeping_hall_curtain
   _close",

   "values": { },

   "Id":"client_1578909153",

   "Version":null
}
```

Steuerung der Heizung und Fenster

ActiveMQ Anbindung

Es wird über folgende Topics kommuniziert:

Topic	Beschreibung
WINDOW.CONTROL	Senden von Steuerinformationen an die Fenster
WINDOW.INFO	Empfangen von Statusinformationen der Fenster
HEATING.CONTROL	Senden von Steuerinformationen an die Heizungen
HEATING.INFO	Empfangen von Statusinformationen der Heizungen

Steuerung der Heizung und Fenster

Bezeichner

Tabelle 1: Liste der Fenster

Bezeichner	Beschreibung	Name im Plan	Stromverteiler
winDining0	Sitzecke Links	F1	S1
winDining1	Sitzecke Rechts	F2	S1
winKitchen0	Küche	F3	S2
winKitchen1	Küche rechts	F4	S2
winLounge0	Lounge ganz links	F5	S3
winLounge1	Lounge links	F6	S3
winLounge2	Lounge mitte	F7	S3
winLounge3	Lounge rechts	F8	S3
winLounge4	Lounge ganz rechts	F9	S3
winBathroom	Badezimmer	F10	S2

Tabelle 2: Gruppen

Bezeichner	Beschreibung
ALL	Alle :)
DINING	F1 & F2
KITCHEN	F3 & F4
LOUNGE	F5,F6,F7,F8 & F9

Tabelle3: Heizungsventile

Bezeichner	Beschreibung	Name im Plan	Heizungsmodul
heatDining	Sitzecke (2 Ventile)	H1 & H2	heatModul0 (Port1)
heatLounge0	Lounge links (2 Ventile)	H3 & H4	heatModul1 (Port1)
heatLounge1	Lounge rechts (2 Ventile)	H5 & H6	heatModul1 (Port2)
heatBathroom	Badezimmer	H7	heatModul1 (Port3)
heatSleep	Schlafbereich (2 Ventile)	H8 & H9	heatModul0 (Port2)

Befehle

- **Fenster**
  - Steuerbefehl (senden an WINDOW.CONTROL)

{"FensterID":["zielposition","geschwindigkeit"]}

- FensterID: Bezeichner aus Tabelle1 und Tabelle2
- Zielposition: Öffnungsweite in cm. 0-20cm. 0 ist geschlossen.
- Geschwindigkeit:

SLOW	Langsame Fahrt, für automatischen Betrieb
FAST	Schnelle Fahrt, laut, für manuellen Betrieb
STOP	Hält den Motor an
NSTP	Notstop

- Statusinformationen anfordern (WINDOW.CONTROL)

```
{"STATUS":"REQUEST"}
```

- Antwort auf Statusrequest (WINDOW.INFO)

```
{"STATUS":["winID1","ziel","geschwindigkeit","position",
           "winID2","ziel","geschwindigkeit","position",...]}
```

Befehle

- **Fenster**
  - Steuerbefehl (senden an WINDOW.CONTROL)

{"FensterID":["zielposition","geschwindigkeit"]}

- - FensterID: Bezeichner aus Tabelle1 und Tabelle2
  - Zielposition: Öffnungsweite in cm. 0-20cm. 0 ist geschlossen.
  - Geschwindigkeit:

SLOW	Langsame Fahrt, für automatischen Betrieb
FAST	Schnelle Fahrt, laut, für manuellen Betrieb
STOP	Hält den Motor an
NSTP	Notstop

- - Statusinformationen anfordern (WINDOW.CONTROL)

```
{"STATUS":"REQUEST"}
```

- - Antwort auf Statusrequest (WINDOW.INFO)

```
{"STATUS":["winID1","ziel","geschwindigkeit","position",
           "winID2","ziel","geschwindigkeit","position",...]}
```

Nachrichtenformat

- **Heizungen**
  - Steuerbefehl (senden an HEATING.CONTROL)

```
{"Heizungsmodul":"XX"}
```

XX enspricht einem Hex Wert. Die unteren 5 Bit repräsentieren je einen Ausgang des Heizungsmoduls.

Eine "1" schaltet den Port ein (und damit das Ventil aus).

Bezeichner	Modul	Wert
heatDining	heatModul0	0x01
heatSleep	heatModul0	0x02
heatLounge0	heatModul1	0x01
heatLounge1	heatModul1	0x02
heatBathroom	heatModul1	0x04

Sollen mehrere Heizungen an einem modul angesteuert werden, werden die Werte per AND kombiniert.

Es muss immer das komplette Byte übertragen werden!

- Statusinformationen (HEATING.INFO)

```
{"Heizugnsmodul":"XX"}
```

JSON Nachrichten

- Einfaches navigierbare XML Format, bestehend aus
  - JSON Objekten mit key, value Paare
  - JSON Arrays
- Erzeugen aus Maps möglich
- Arrays müssen zuvor in JSONArray gewandelt werden

JSON Nachrichten erzeugen am Beispiel


```
private void jsonDemo() {
    Map<String,String> valueMap = new HashMap<String, String>();
    valueMap.put("red","255");
    valueMap.put("green","0");
    valueMap.put("blue","128");
    valueMap.put("fadeTime","0");

    Map<String,Object> aendernMap = new HashMap<String, Object>();

    aendernMap.put("values", valueMap);
    aendernMap.put("Id", "client_1578909153");
    aendernMap.put("Version", null);
    aendernMap.put("action", "dining_light_color");
    JSONArray jsoa = new JSONArray(Arrays.asList(new Integer[]{1,2,3}));
    aendernMap.put("anAry", jsoa );
    JSONObject msgFarbeEsszimmerAendern = new JSONObject(aendernMap);
    Log.d("JSONDemo", msgFarbeEsszimmerAendern.toString());
}
```