# Integration of new devices, agents to an existing architecture

July 11, 2013

Sections 1 and 2 respectively explain how to add a sensor to a Raspberry PI that is already connected to the architecture and then, how to add a Raspberry PI to the global architecture. Section 4 then describes how to add/define a new service (i.e. subscription or other) to a QLM node such as a Raspberry PI, a computer, and so on.

## 1 Add a sensor to a Raspberry PI

Once the architecture is set up (*cf.* Figure **??** in the document entitled "*Architecture specifications*"), it is possible to add a new sensor to a Raspberry PI. To do so, perform the following steps:

- plug your sensor (noted $s_1$ in this document) on the Raspberry PI (noted $r_4$ in this document),
- connect the computer to the same network as the Raspberry PI $r_4$ (e.g. NETGEAR67 in our case),
- open a terminal and set the following command line to connect your computer to $r_4$:

```
1 > ssh pi@192.168.0.4
2 > Are you sure you want to continue connecting (yes/no)? yes
3 > password: bimqlm4
```

- in case of you have already something mounted (to know that, use the command `> cd /mnt/1wire`, if you can go inside, then it has been mounted). In such a case, go in ANOTHER directory and unmount /mnt/1wire by using the following command:

```
1 > sudo umount /mnt/1wire
```

- set the following command line to mount the sensor information to $r_4$:

```
1 > sudo /opt/owfs/bin/owfs -u /dev/ttyUSB0 --allow_other /mnt/1wire/
```

- set the following command line to go in the appropriate folder and to see the ID of the sensors connected to $r_4$ (results look something like: 26.3C5745010000; 26.A54445010000,...):

```
1 > cd /mnt/1wire/
2 > ls
```

- set the following command line (assuming that 26.A54445010000 is the ID of sensor $s_1$) to go in the appropriate folder and to see the different features/variables available on sensor $s_1$:

```
> cd 26.A54445010000
> ls
```

- the following command line would be for instance used to get the current value of the feature/variable named temperature (results look something like: 24.5312):

```
> cat temperature
```

## 2 Add a Raspberry PI to the architecture

There are two possible ways to configure the Raspberry PI in order to integrate it in the architecture:

1. the first possibility is much more complex than the second since it is necessary to follow the steps given at https://wiki.aalto.fi/display/IOT/Step+by+step+guide%3A+Dialog+in+Raspberry+Pi. This possibility is necessary when no Raspberry PI has been configured yet,
2. the second possibility consists in copying an existing image from a Raspberry PI already connected/configured, to paste it on a SD card that need to be inserted in the new Raspberry PI. After booting the Raspberry with the SD card, it should be ready to work. For more information, see https://wiki.aalto.fi/display/IOT/Step+by+step+guide%3A+Dialog+in+Raspberry+Pi.

## 3 Re-Strating the DIALOG server remotely on a Raspberry PI

Steps performed in any computer:

- connect to PuTTY with right parameters (login/password and IP address),
- check if all the sensors are mounted *cf.* section **??**,
- execute the following command:

```
$ vncserver :1
```

- close PuTTY,
- run VNC Viewer and specify the IP address followed by ":1" as follows:

```
192.168.0.4:1
```

- press connect and continue
- password: bimqlm4
- open a terminal and execute the following command line:

```
cd /home/pi/apache-tomcat-7.0.41/bin/
./startup.sh
```

- paste the following command line in a web browser to check if everything goes well (the error in the word QLMReciever is normal)

```
http://192.168.0.4:8080/qlm/QLMReciever
```

# 4 Agent development guide

The following steps describe how to code an agent (in other words, how to code a class that allows the communications between the hardware and DIALOG). In our example, we still consider Raspberry PI $r_4$ and sensor $s_1$ that has been mounted beforehand:

Steps performed from any computer:

- download the DIALOG code from... (Manik gave you it.. normally)
- with NetBean (or other), open the `eventAgent` class located in the package `fi.hut.dialog.qlm`. Change the global variable named `objname` in this class by the appropriate one[1],
- go to the method `getMessage` in the class `eventAgent`[2] and you should see somehting similar to the code given below. The following code indicates that two sensors named `CS1` and `CS2` (*cf.* rows 1 and 3) should be connected to the Rasperry PI and are expected to respectively monitor the `temperature` and the `humidity` (*cf.* rows 22 and 24).

```
1  if (formal_infoclass.equalsIgnoreCase("CS1")) {
2          message += "CS1";
3  } else if (formal_infoclass.equalsIgnoreCase("CS2")) {
4          message += "CS2";
5  } else {
6          return null;
7  }
8  message += "'><metadata>" + "<id>";
9  if (formal_infoclass.equalsIgnoreCase("CS1")) {
10         message += "CS1";
11 } else if (formal_infoclass.equalsIgnoreCase("CS2")) {
12         message += "CS2";
13 } else {
14         return null;
15 }
16 message += "</id>"
17 + "<description>Pipe Temperature</description>"
18 + "<unit>Float</unit>"
19 + "</metadata>"
20 + "<value>";
21 if (formal_infoclass.equalsIgnoreCase("CS1")) {
22         message += readFile("/mnt/1wire/12.2D1595000000/temperature");
23 } else if (formal_infoclass.equalsIgnoreCase("CS2")) {
24         message += readFile("/mnt/1wire/12.2E1685000000/humidity");
25 } else {
26         return null;
27 }
```

In our example, this part of code needs to be modify according to the devices connected to $r_4$ and also the features that these devices should monitor. In our scenario, we only have sensor $s_1$ and the feature that we want to use is `temperature`. The code below shows how the previous code is modified (note that the ID got when we mounted $s_1$, i.e. `26.A54445010000`, is added in row 18:

```
1  if (formal_infoclass.equalsIgnoreCase("s1")) {
2          message += "s1";
3  } else {
4          return null;
5  }
6  message += "'><metadata>" + "<id>";
```

[1] At this stage, this name can be anything, but keep in mind that it will be reused later (e.g. when doing subscription)
[2] Metadata can be added to the message through the option `description`, although this is not mandatory).

```
7  if (formal_infoclass.equalsIgnoreCase("s1")) {
8          message += "s1";
9  else {
10         return null;
11 }
12 message += "</id>"
13 + "<description>Pipe Temperature</description>"
14 + "<unit>Float</unit>"
15 + "</metadata>"
16 + "<value>";
17 if (formal_infoclass.equalsIgnoreCase("s1")) {
18         message += readFile("/mnt/1wire/26.A54445010000/temperature");
19 } else {
20         return null;
21 }
```

- open the `QLMAgent` class which is located in the package `fi.hut.dialog.qlm` and, as previously, change the global variable named `objname` with the same name given in the class `eventAgent`. Take the code of `getMessage` previously defined in `eventAgent` and paste it over the method `getMessage` from `QLMAgent`,
- compile the code using `using and Build Project (dialog)` (Shift+F11),
- in the build directory of the code, you should see a folder named `qlm`,
- open the WinSCP software and provide the information to get log to $r_4$ (IP add., login/password),
- delete (if exist) the `qlm` directory in the frame 2 highlighted in Figure 1, and then drag the appropriate[3] `qlm` directory as shown by arrow 1 in Figure 1 (i.e. from Frame 1 to Frame 2),
- go to the \bin folder and, if everything went well, a file named `agent.txt` should be present,
- open it and make sure that the rows below are present:

```
1  fi.hut.dialog.qlm.QLMAgent:fi.hut.dialog.qlm.QLMAgent
2  fi.hut.dialog.qlm.QLMSubscriptionAgent:QLM Subscription Agent
```

Steps performed from a computer connected to $r_4$:

- turn $r_4$ on, set the login/password and start it as below:

```
1  $ login: pi
2  $ password: bimqlm4
3  $ startx
```

- open a terminal and, first, make sure that $s_1$ has been mounted (*cf.* section 1),
- go to the bin server with the following command line:

```
1  $ cd apache-tomcat-7.0.41/bin/
```

- set the following command:

[3] make sure that you are in the correct folder in Frame 1 (*cf.* Figure 1)

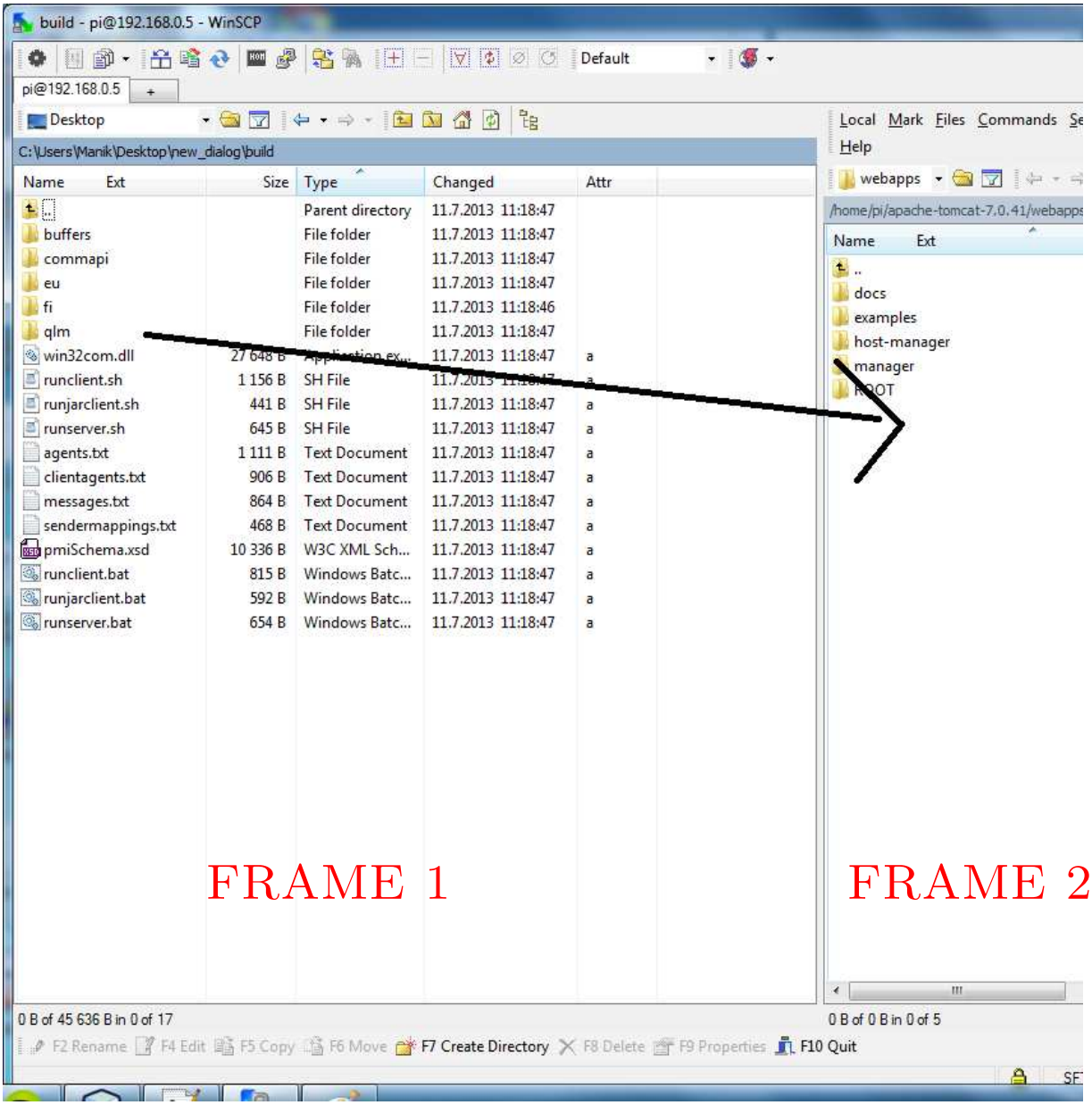


Figure 1: Screenshot of WinSCP

```
$ ./startup.sh
```

Steps performed from any computer:

- wait that Tomcat starts (can takes few minutes). To check if it is ready, try to access it from a webpage with the following command line:

```
192.168.0.4:8080
```

- Open `Pi.html` with a text editor (e.g. Notepad++) and modify the query according to your expectation: subscription, cancellation, etc. (see the QLM specifications at  or still the article at ),
- once it has been modified and save, close the text editor and open it normally (with a web browser) Then, open it normally and click `submit`. You should get, in the response (displayed on the web browser), the request ID as below (the request ID is `1373237421334@` in this example):

```
<qlmEnvelope version="1.0" ttl="0.0" xmlns="QLM_mi.xsd"><response><result><return
    returnCode="200"/><requestId>1373237421334@</requestId></result></response></
    qlmEnvelope>
```

- this ID must be used when you want to cancel the subscription. To send such a cancellation, send/submit the following message by including the ID number `1373237421334@`:

```
<qlmEnvelope version="1.0" ttl="0.0" xmlns="QLM_mi.xsd"><cancel><requestId>
    1373237421334@</requestId></cancel></qlmEnvelope>
```

- you should receive a success response (i.e. with the code `200`)