Introduction into Bluetooth Low Energy

www.Matthias Gohlke.com

Ideas beyond imagination

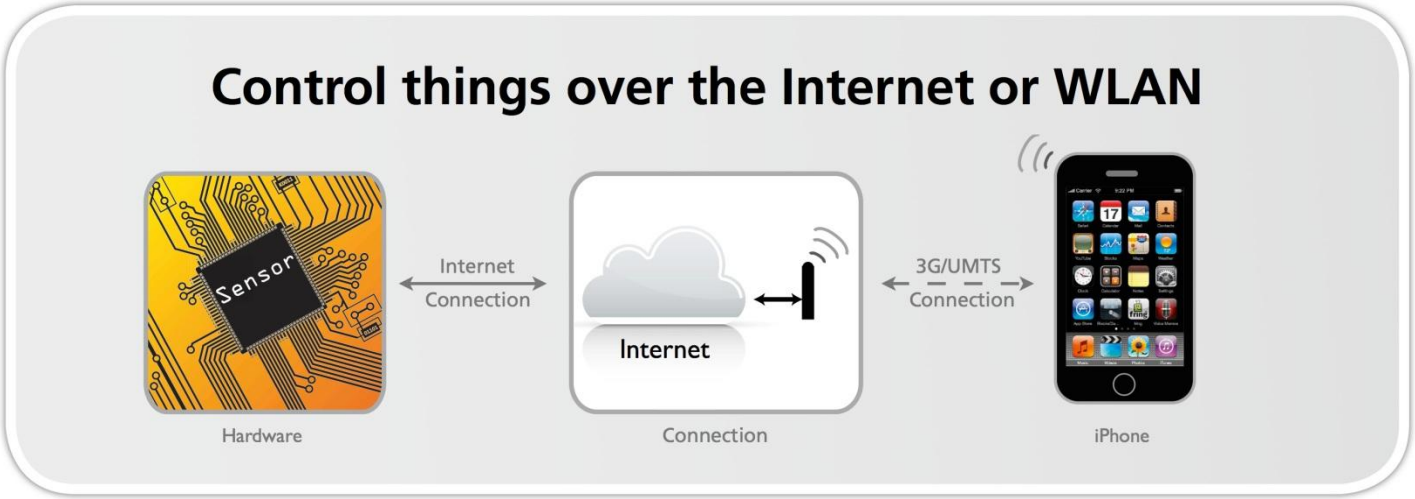
nabto

connect - simple and safe

BLE LABS®

FACHHOCHSCHULE KIEL
University of Applied Sciences


Control and monitoring solutions
for renewable technologies


Control things over the Internet or WLAN
Sensor
Internet
Connection
Internet
3G/UMTS
Connection
Hardware
Connection
iPhone

BLE

Dual mode Device

Singel mode Device

What is Blue tooth Low Energy

Accessory

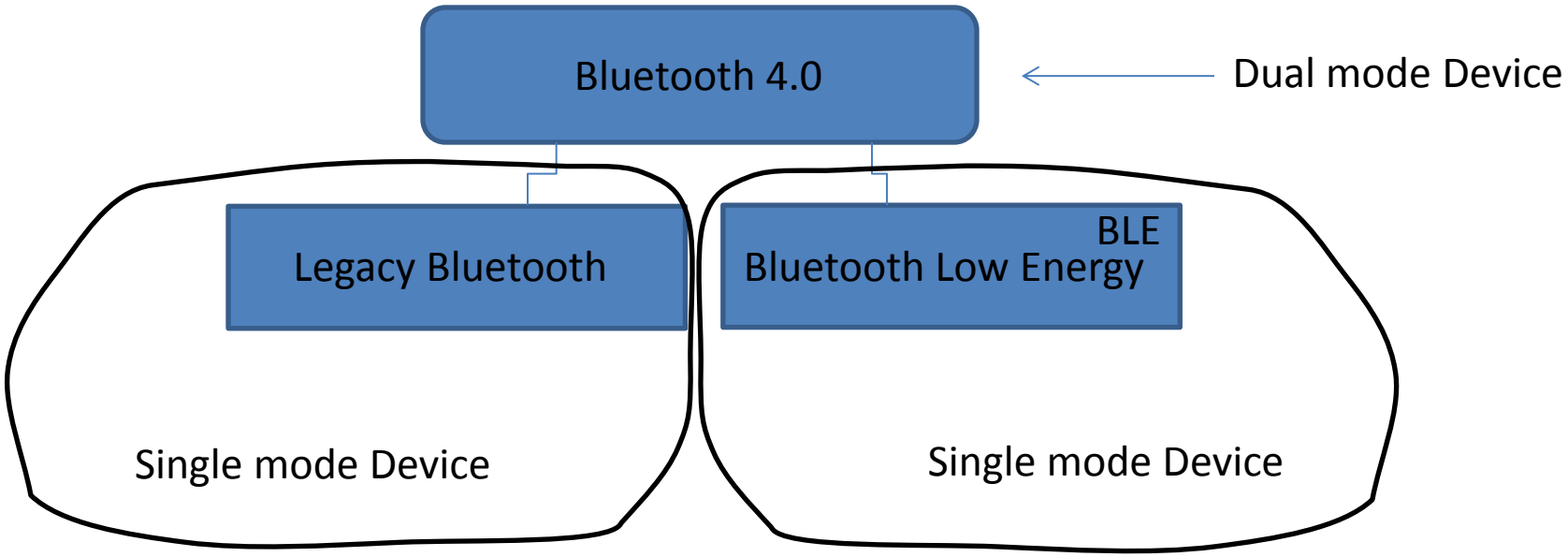
Bluetooth 4.0

Bluetooth Low Energy

Central

Legacy Bluetooth

What is Blue tooth Low Energy



Idea behind Bluetooth Low energy

x1

Send less data and spent less time on the air.

Data rate : 300 kbps

Bluetooth 3.0: 2100 kbps

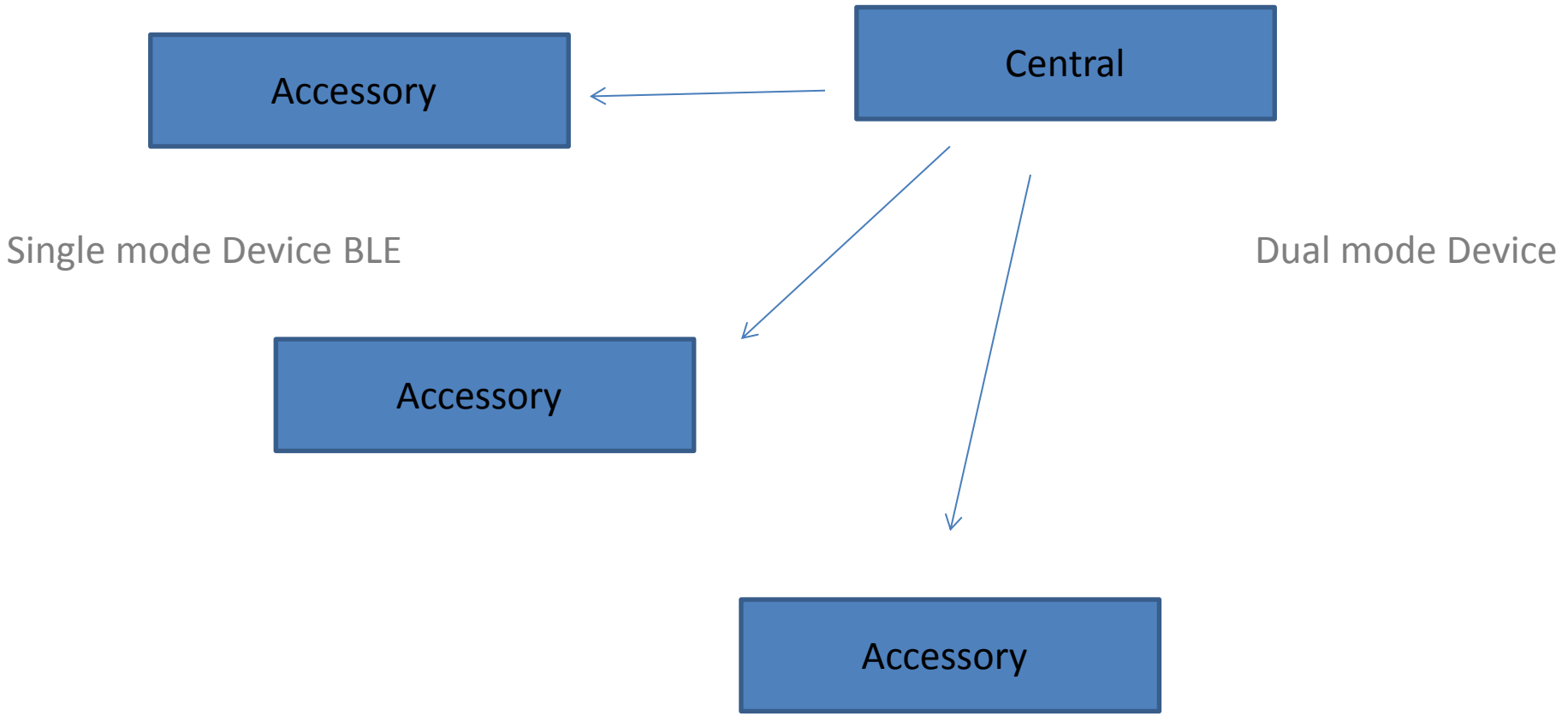
X1 source: http://en.wikipedia.org/wiki/CR2032_battery

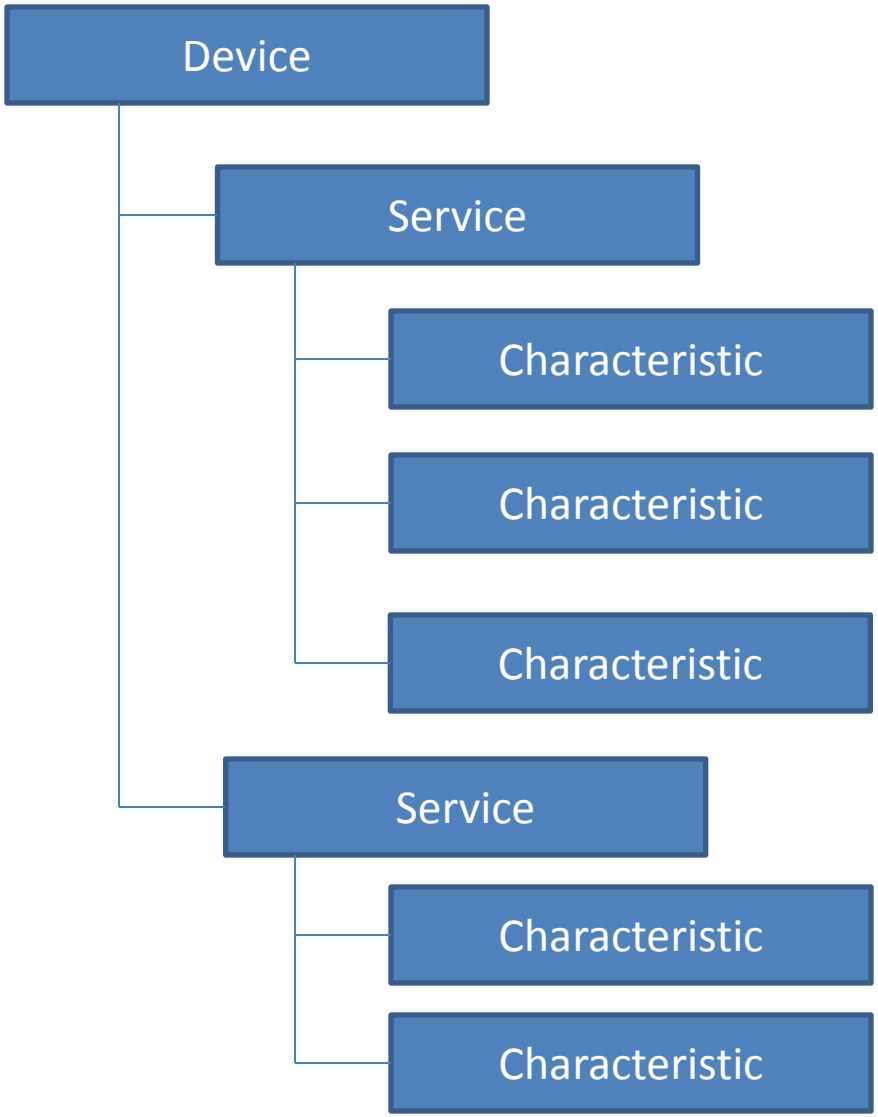
Bluetooth 3.0 / BLE comparison

Technical Specification	Classic Bluetooth technology	Bluetooth low energy technology
Distance/Range	100 m (330 ft)	50 m (160 ft)
Over the air data rate	1-3 Mbit/s	1 Mbit/s
Application throughput	0.7-2.1 Mbit/s	0.26 Mbit/s
Active slaves	7	Not defined; implementation dependent
Security	56/128-bit and application layer user defined	128-bit AES with Counter Mode CBC-MAC and application layer user defined
Robustness	Adaptive fast frequency hopping, FEC, fast ACK	Adaptive frequency hopping, Lazy Acknowledgement, 24-bit CRC, 32-bit Message Integrity Check
Latency (from a non connected state)	Typically 100 ms	6 ms
Total time to send data (det.battery life)	100 ms	6 ms[*citation needed*], <3ms[17]
Voice capable	Yes	No
Network topology	Scatternet	Star-bus
Power consumption	1 as the reference	0.01 to 0.5 (depending on use case)
Peak current consumption	<30 mA	<20 mA
Service discovery	Yes	Yes
Profile concept	Yes	Yes
Primary use cases	Mobile phones, gaming, headsets, stereo audio streaming, automotive, PCs, security, proximity, healthcare, sports & fitness, etc.	Mobile phones, gaming, PCs, watches, sports and fitness, healthcare, security & proximity, automotive, home electronics, automation, Industrial, etc.

Source: http://en.wikipedia.org/wiki/Bluetooth_low_energy

What is Blue tooth Low Energy





This could be a value like temp.

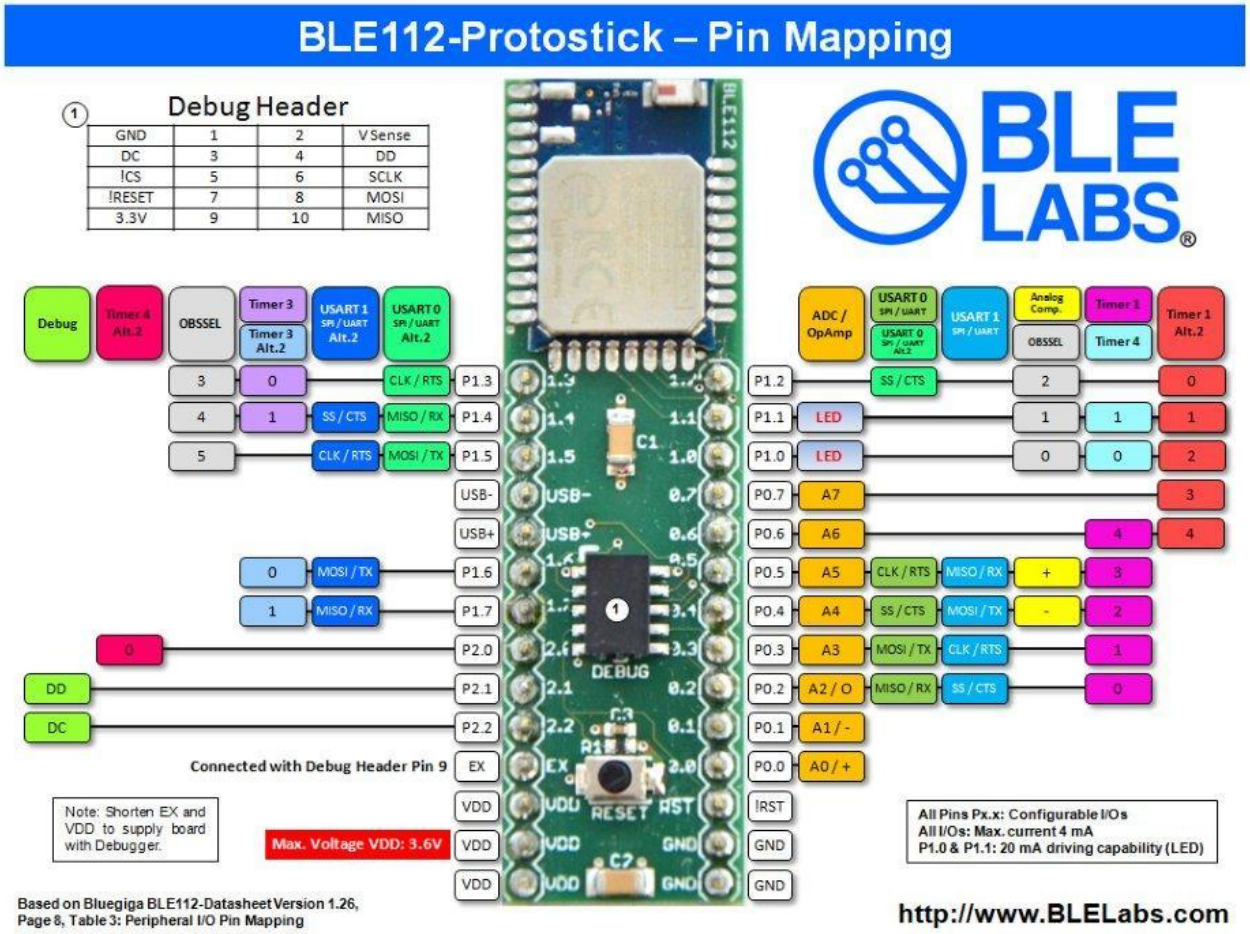
UUID with 16bit (Bluetooth SIG profiles)

UUID with 128bit customm

Chips and Development kits

Source: http://www.ti.com/graphics/tool/CC2540DK-MINI.jpg

BLELabs BLE112 Protostick



Bluetooth Low Energy for Home Automation

Tackling the limitations of Bluetooth Low Energy in complex wireless networks

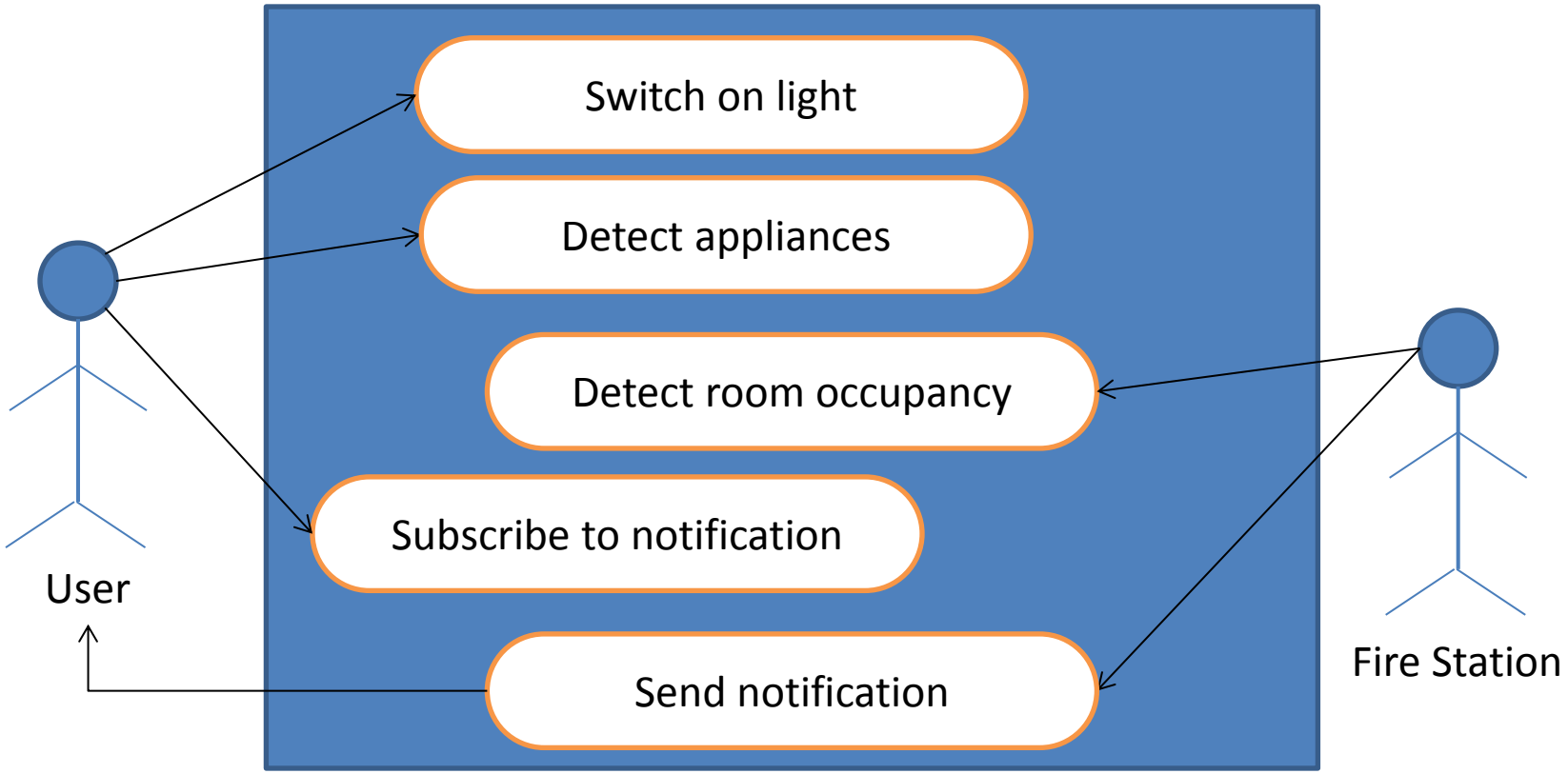
Matthias Gohlke

FACHHOCHSCHULE KIEL
University of Applied Sciences

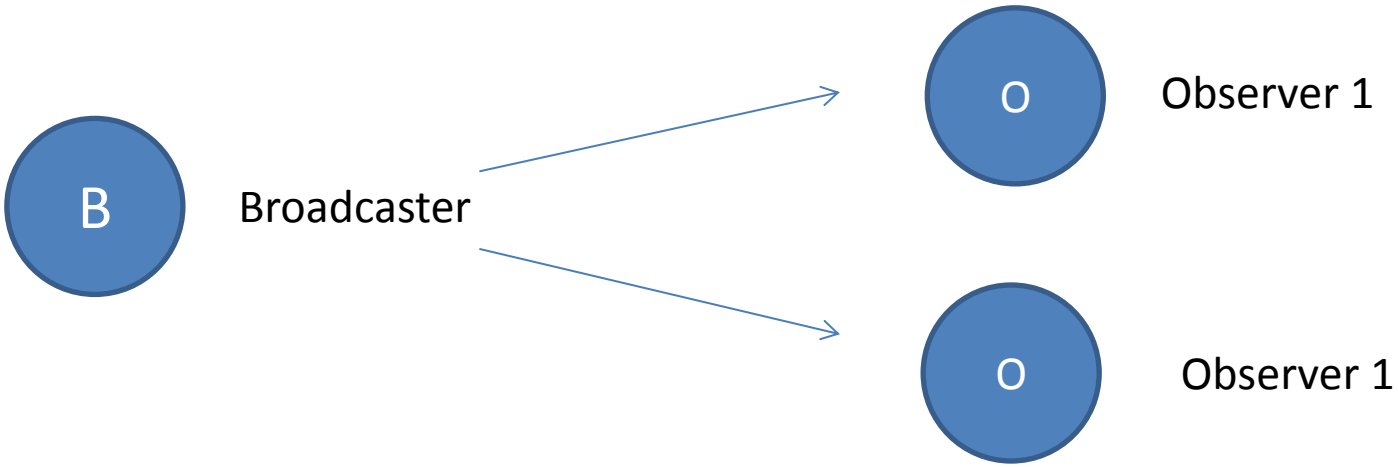
1. The Use Case
2. BLE device roles
3. Possible Gateway scenarios
4. System Diagram
5. System components
6. Used tools
7. The Low Energy aspect
8. Conclusions
9. Gateway Demo

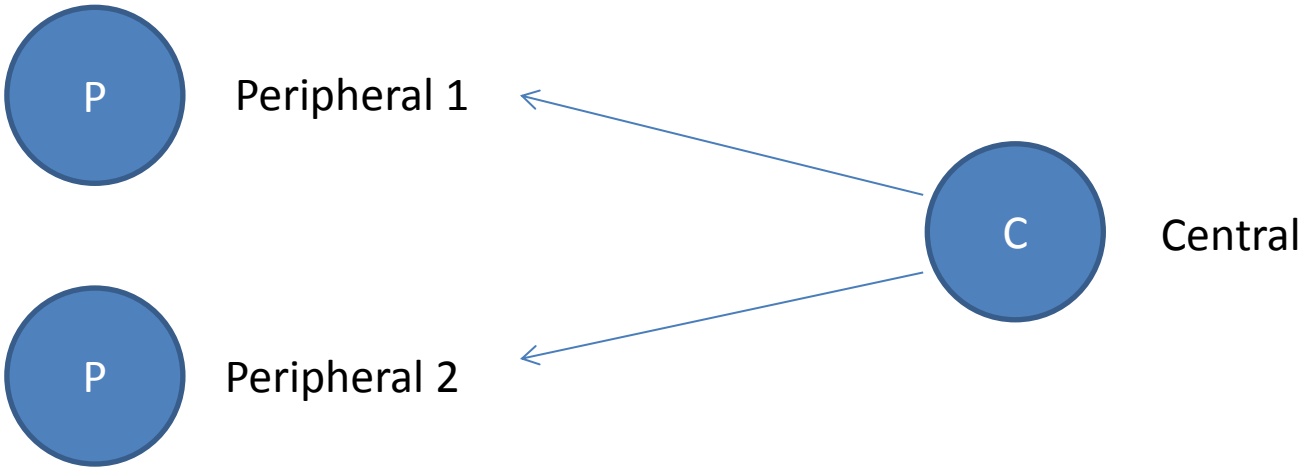
1. Use Case

- We want to detect appliances in a room an turn on the light via a Smart-phone
- Proximity switching
- Connect to a room gateway for information
- Awareness of room occupancy



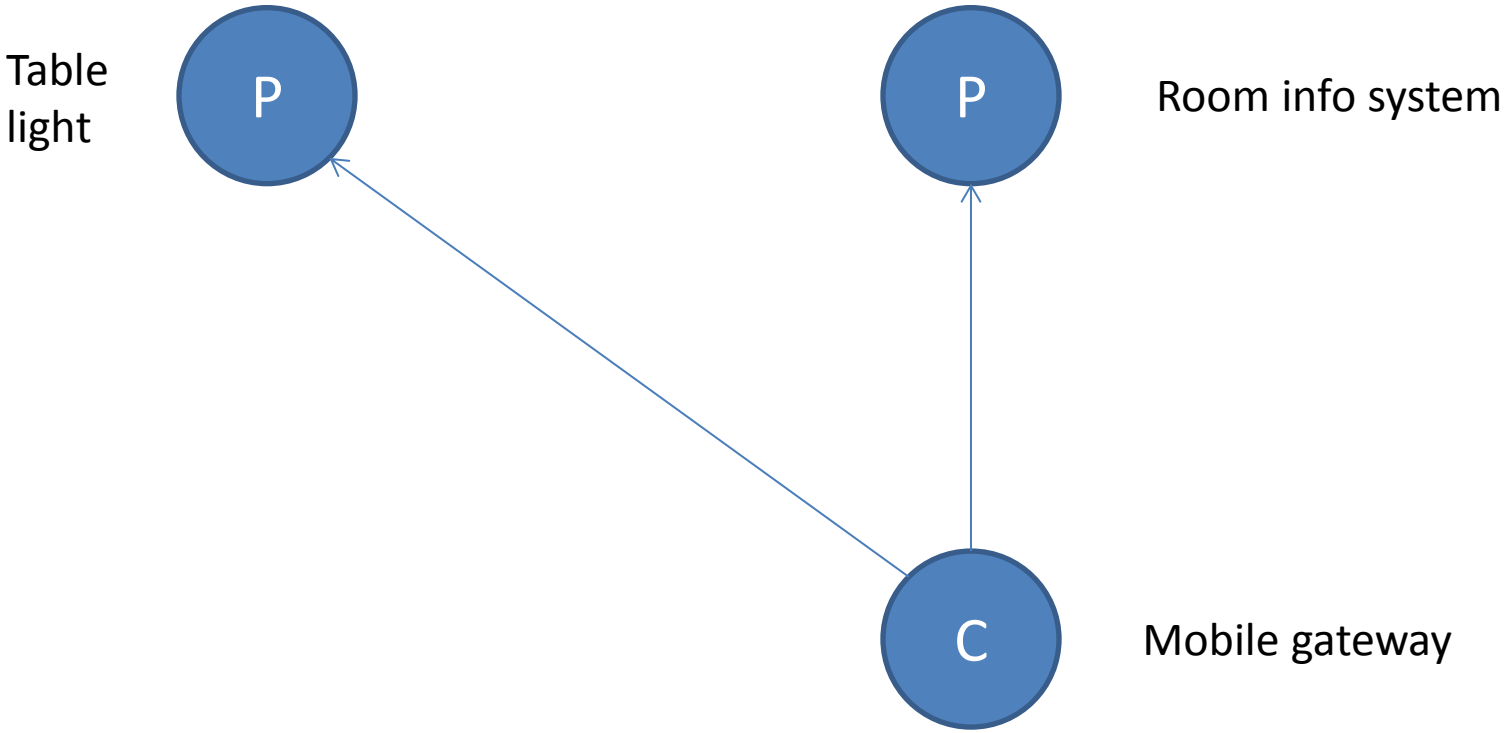
2. BLE device roles



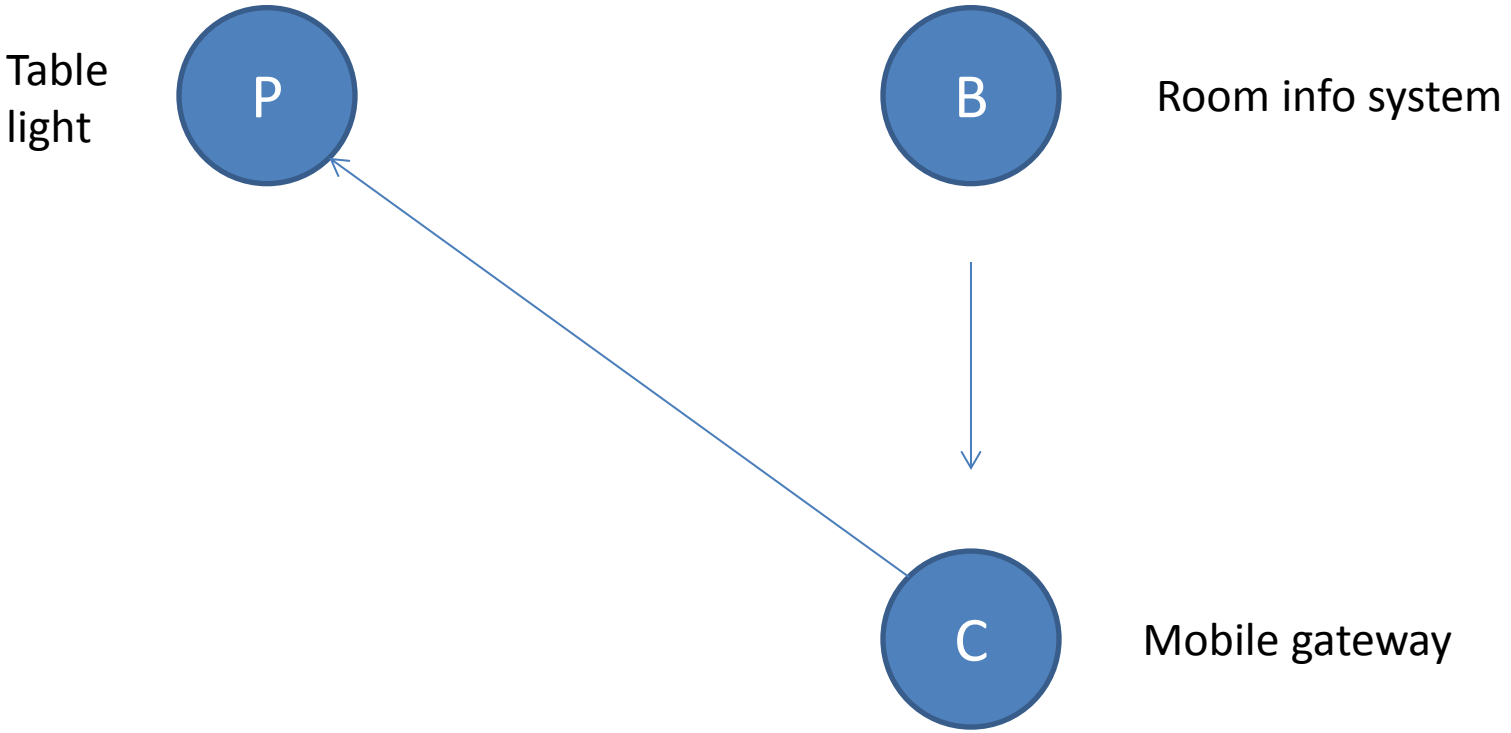


3. Possible Gateway scenarios

Scenario 1

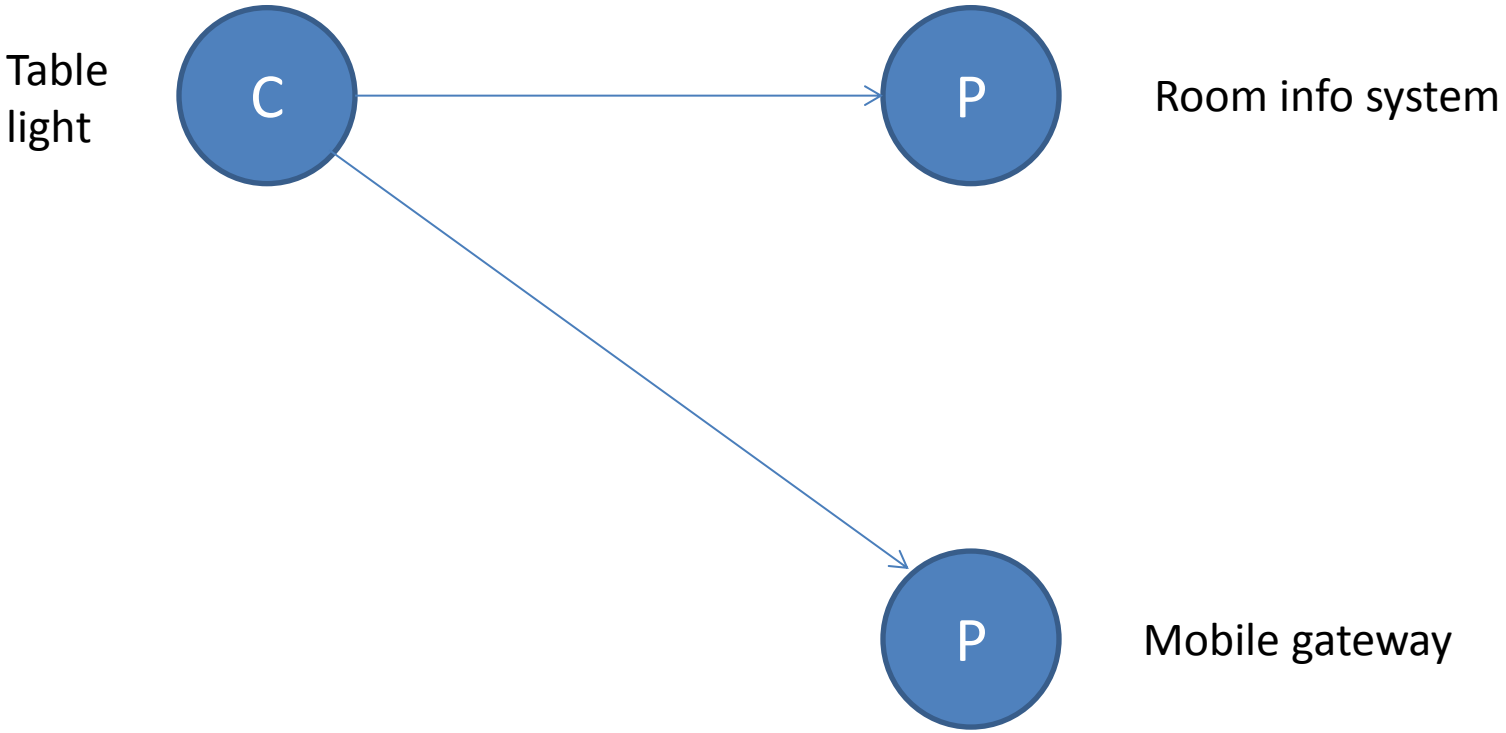


Scenario 2

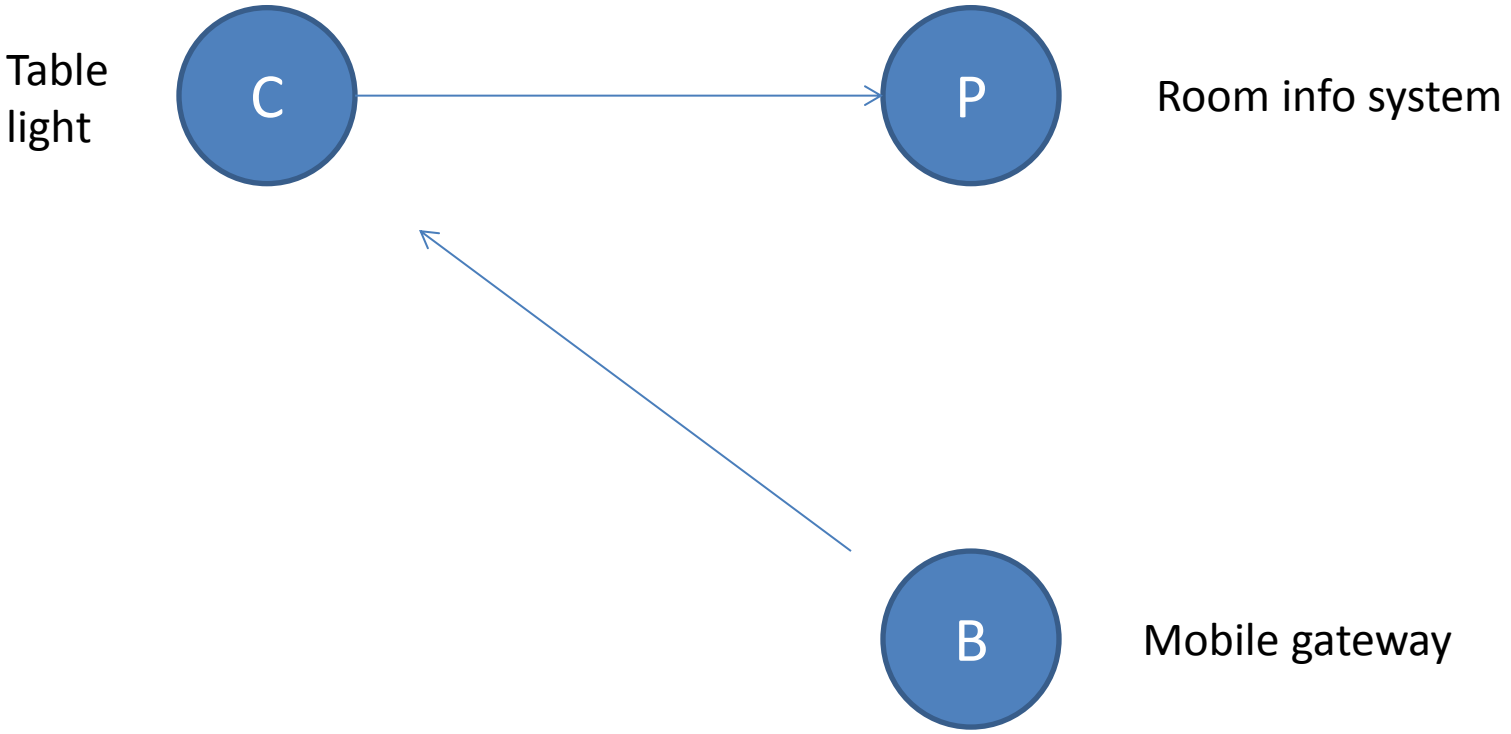


3. Possible Gateway scenarios

Scenario 3



Scenario 4



Scenario 5

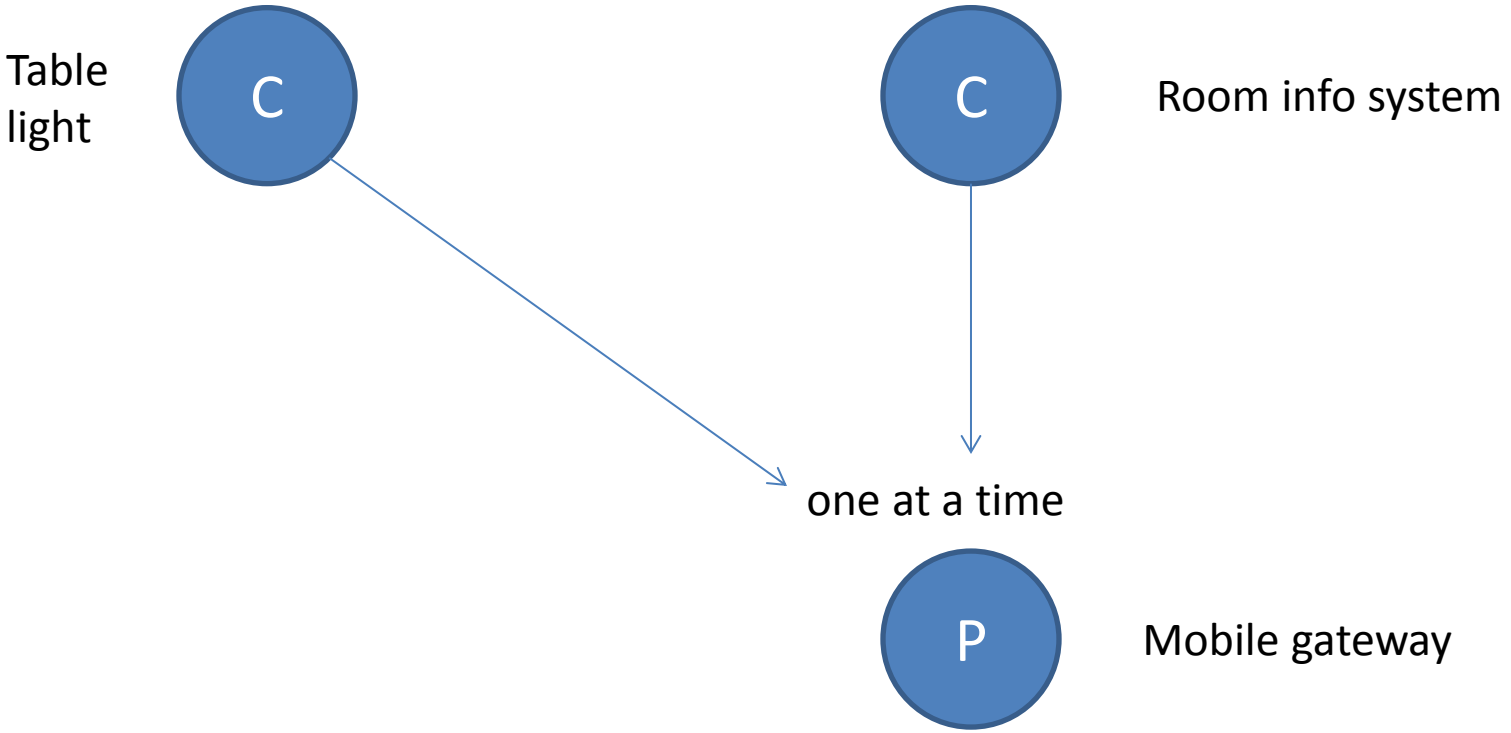
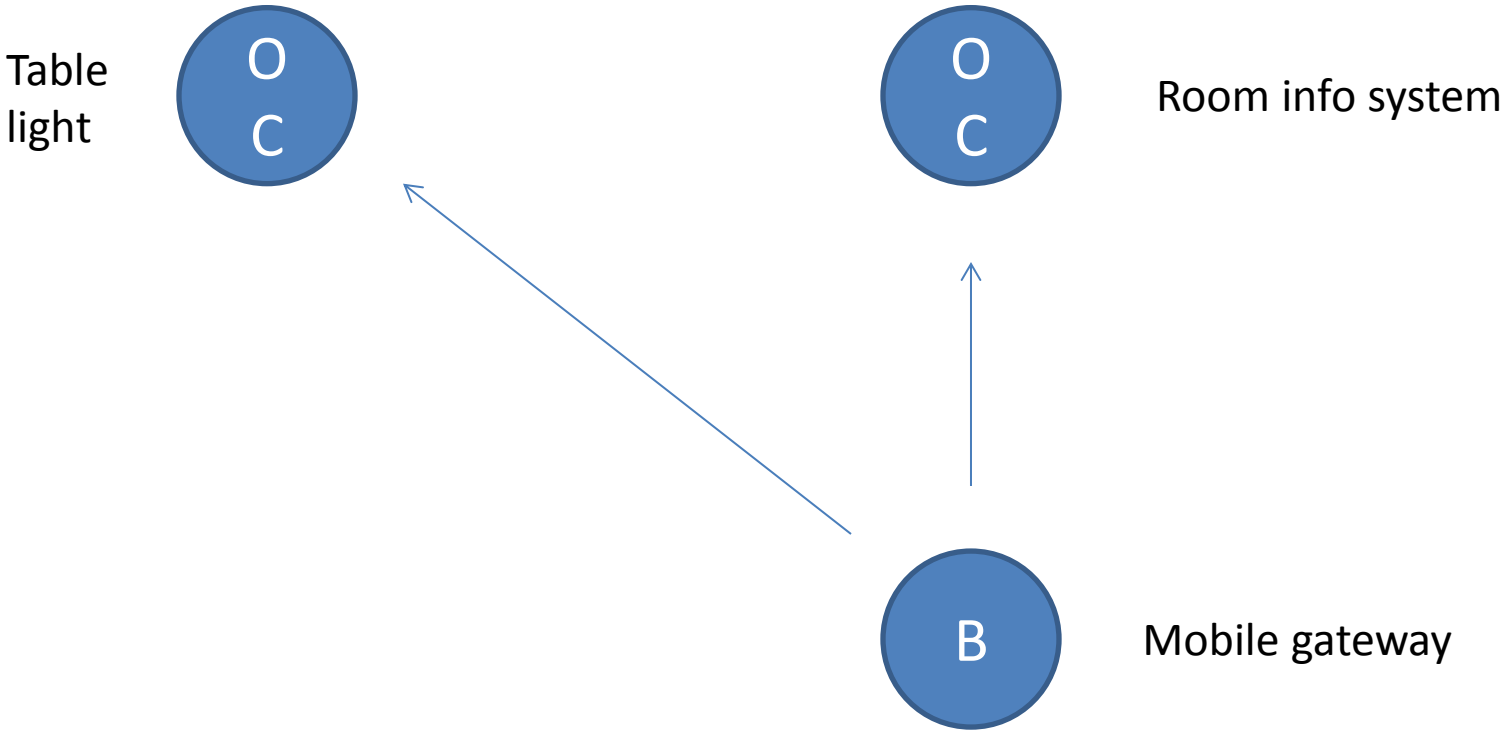
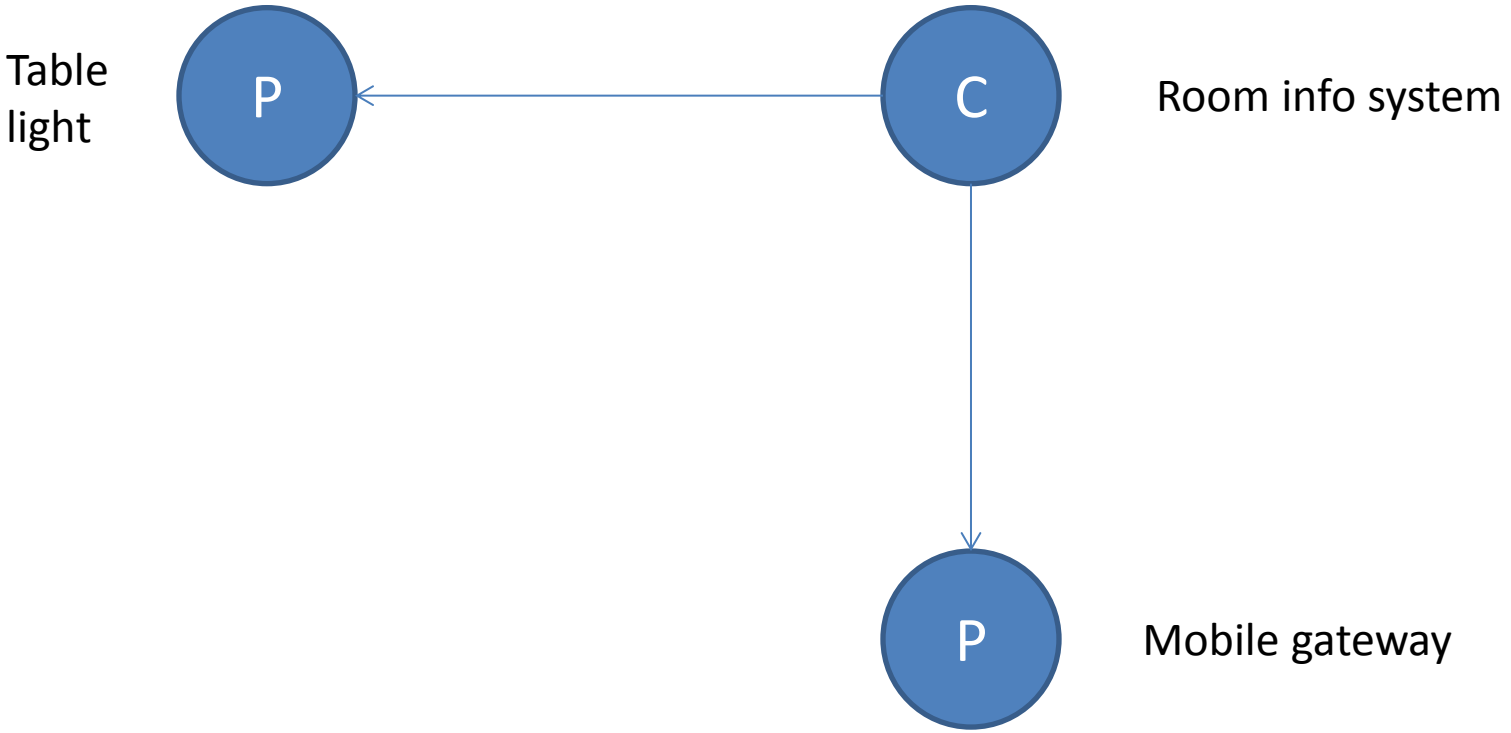


Table light
C
C
Room info system
one at a time
P
Mobile gateway

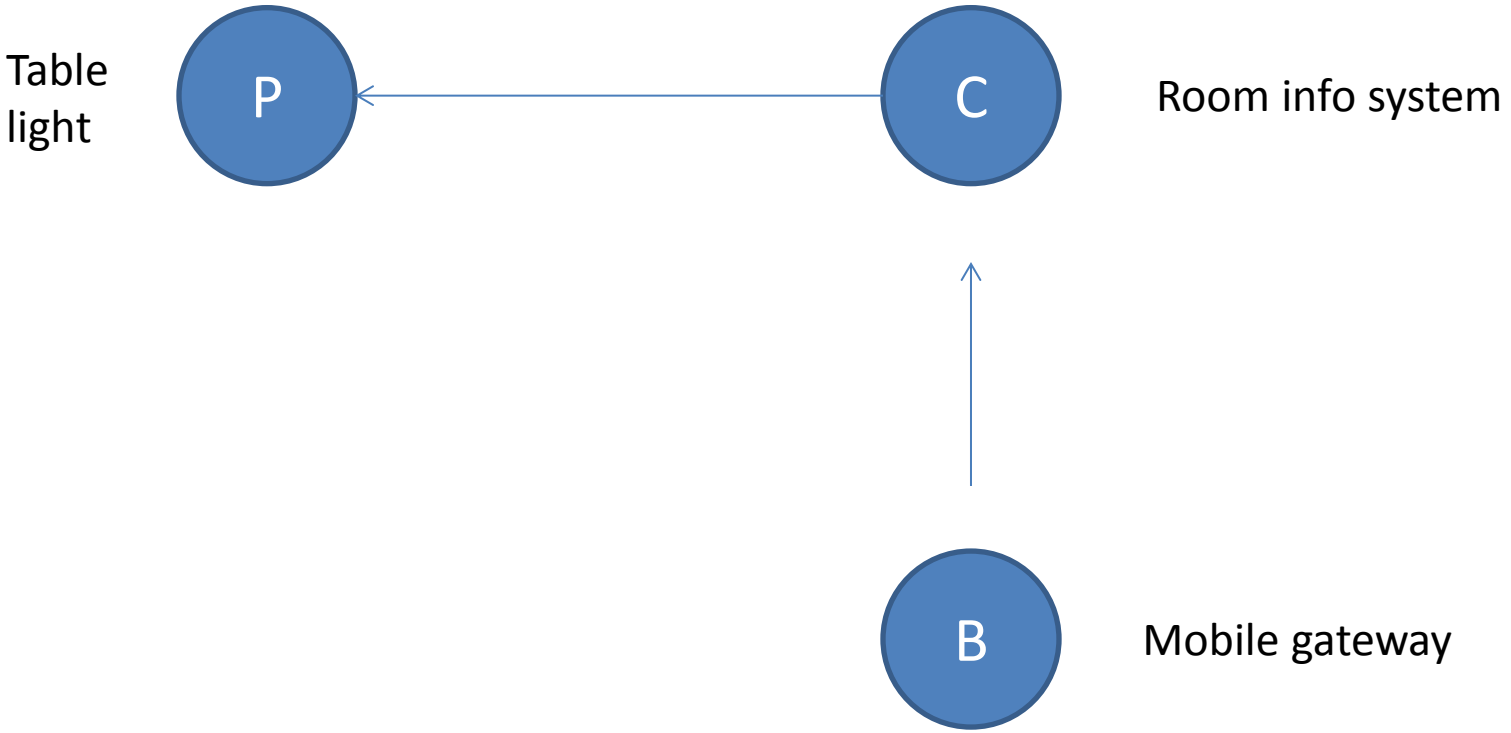
Scenario 6



Scenario 7



Scenario 8



Scenario 9

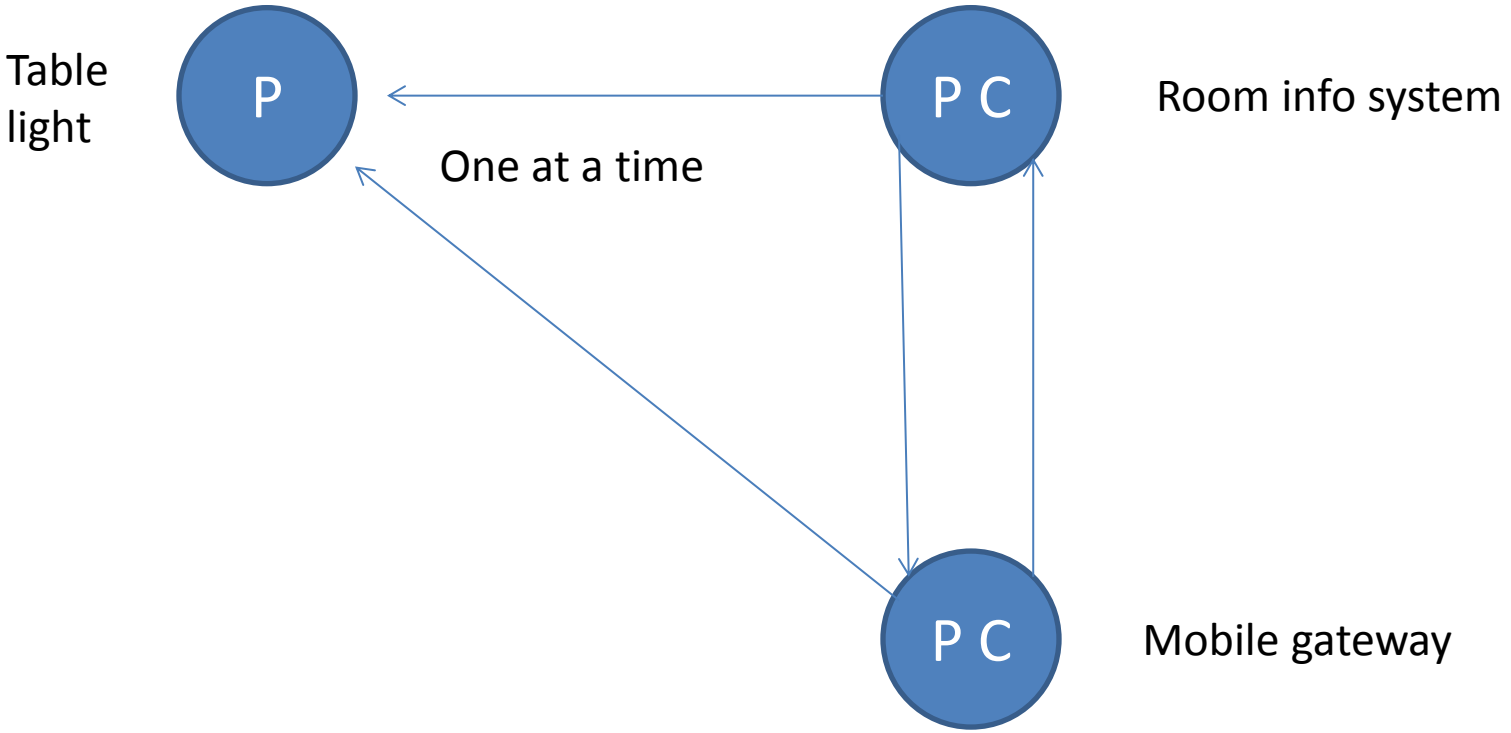
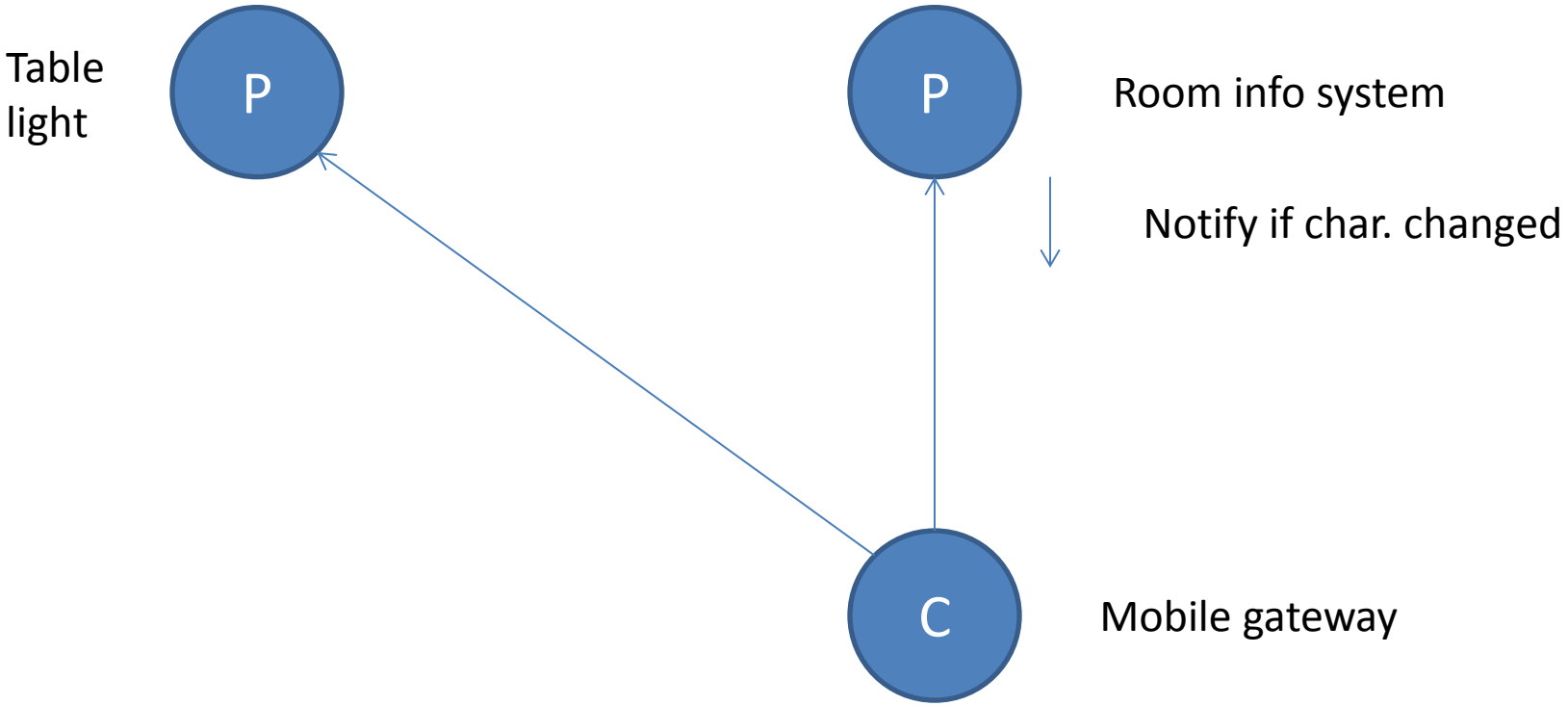
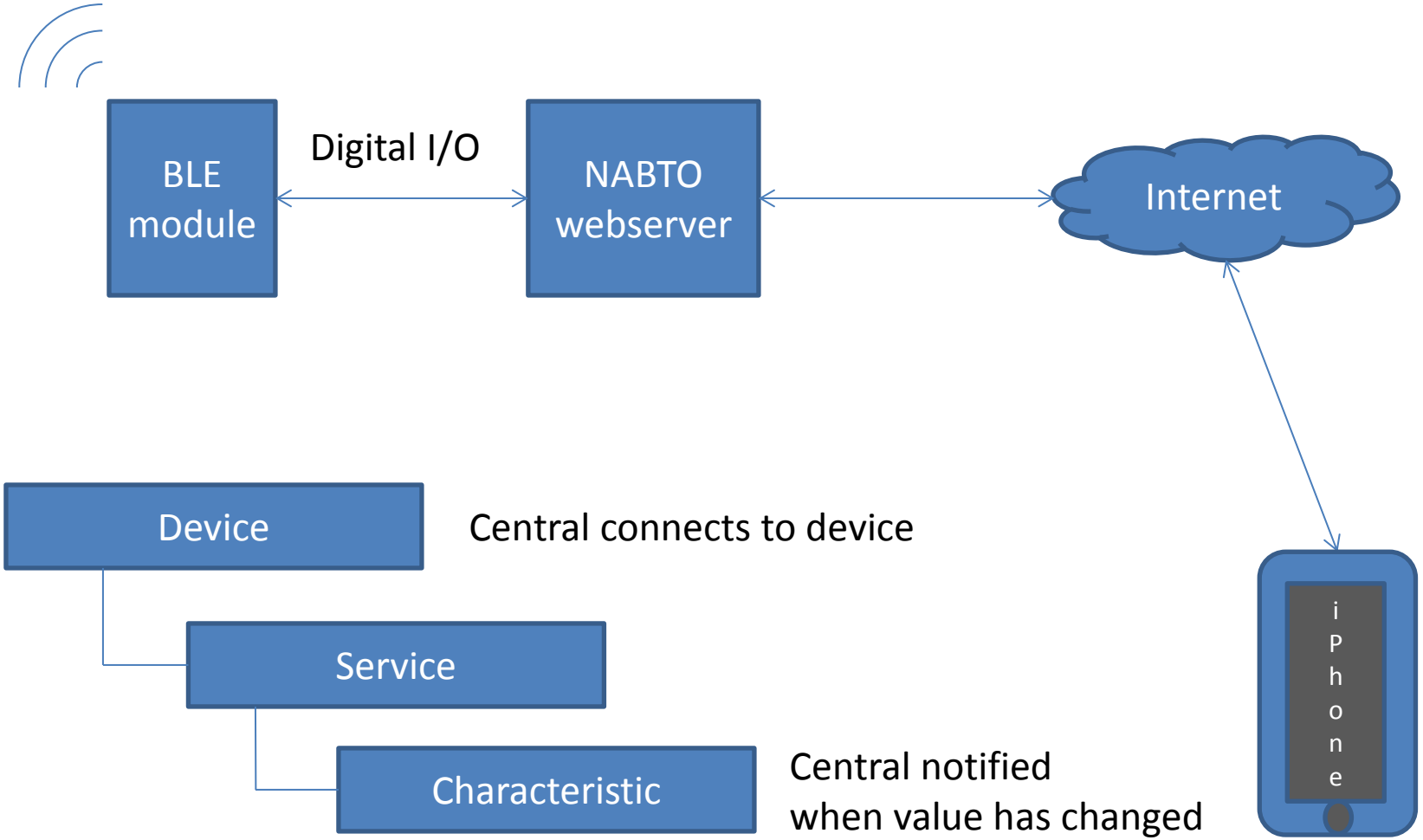


Table light
P
P C
Room info system
One at a time
P C
Mobile gateway

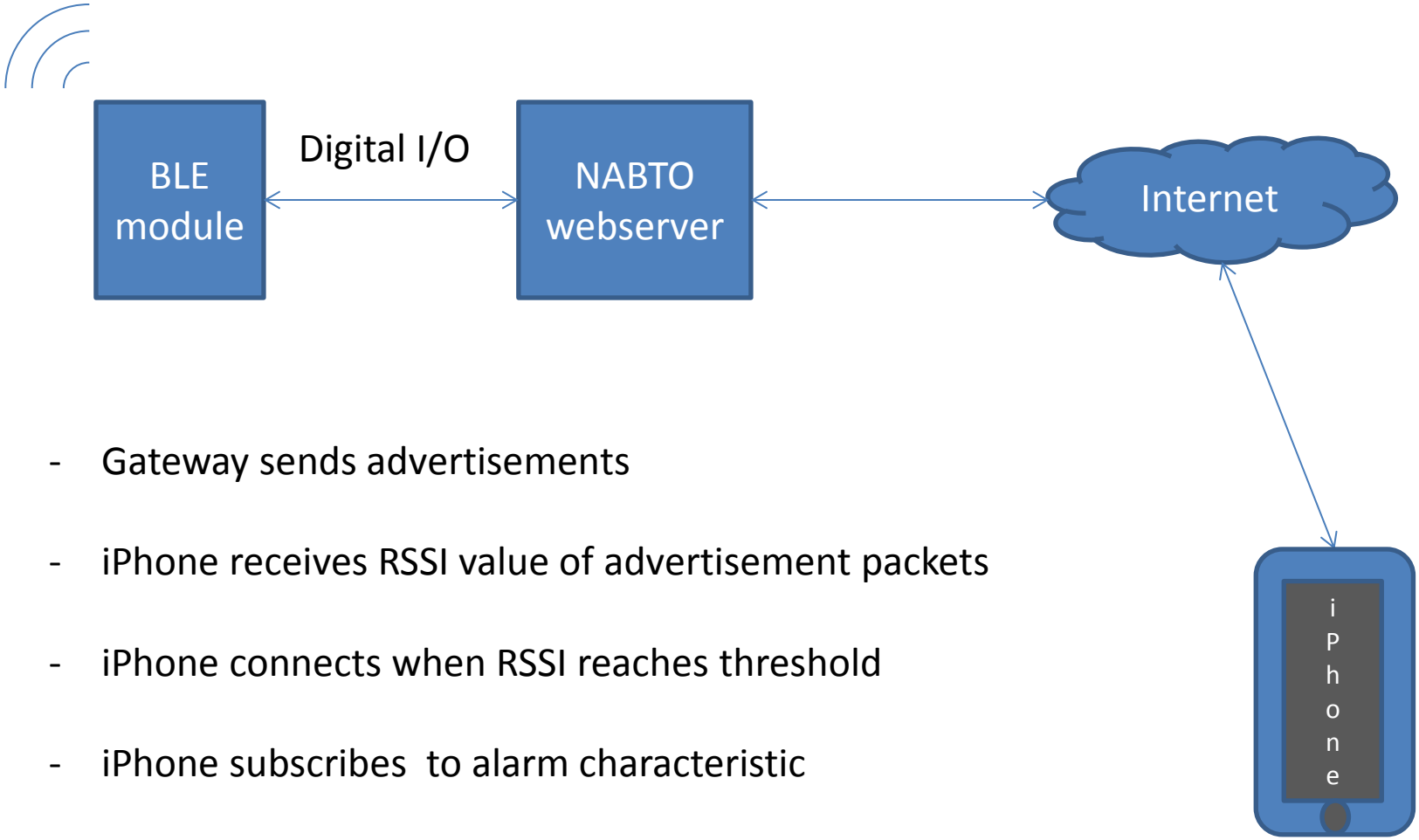
4. System Diagram



5. System components – BLE web gateway

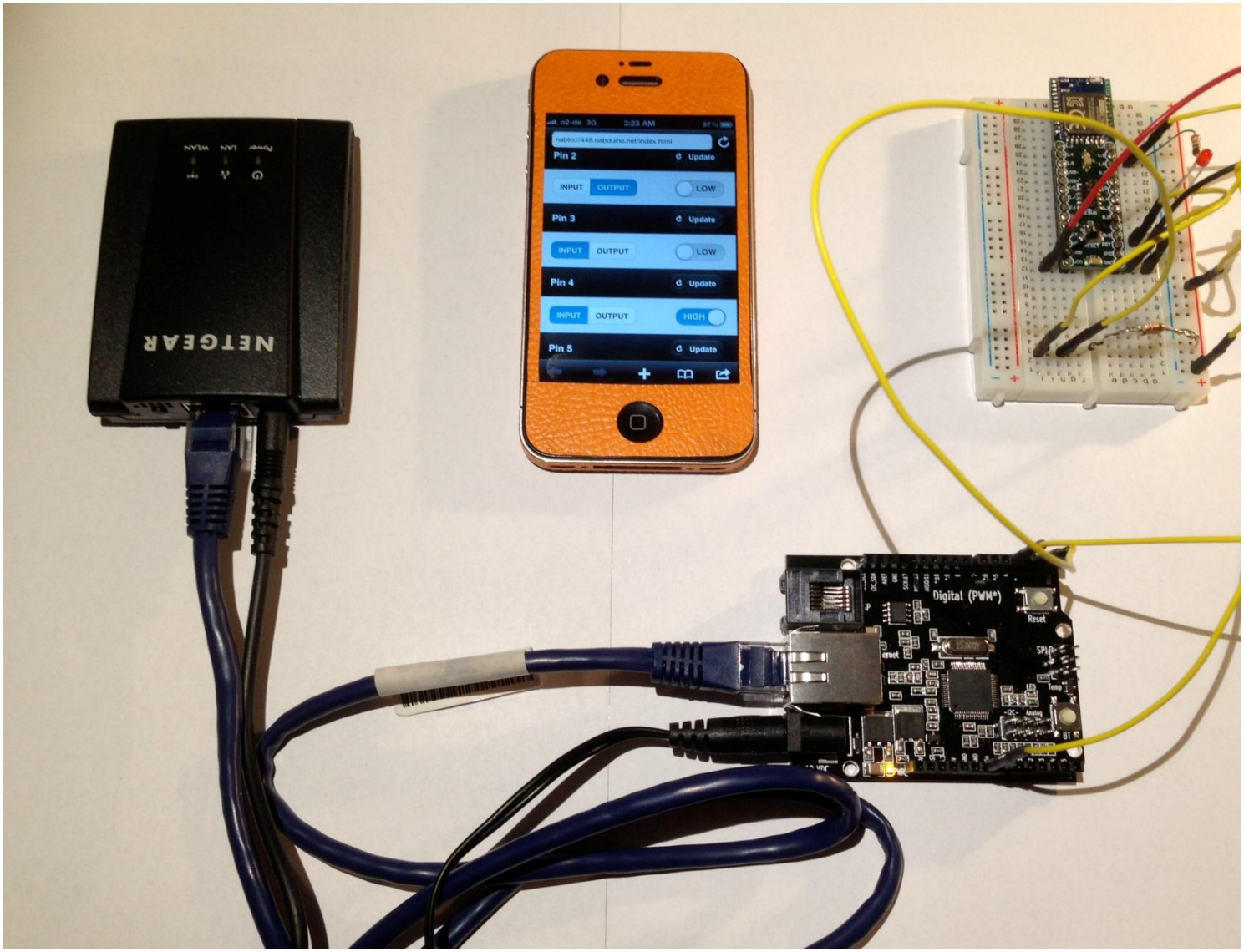


5. System components – BLE web gateway

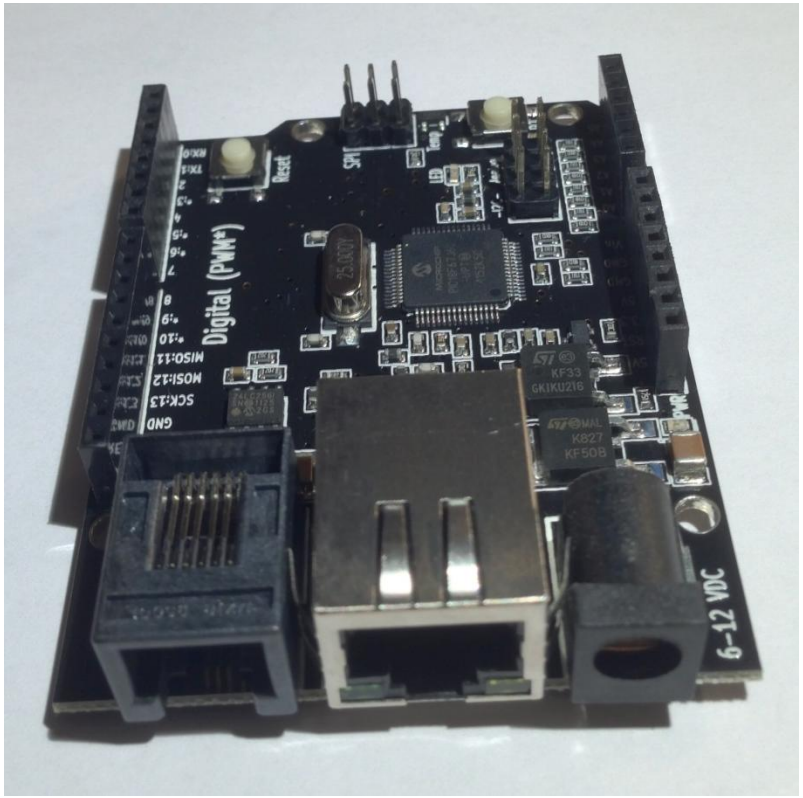


- Gateway sends advertisements
- iPhone receives RSSI value of advertisement packets
- iPhone connects when RSSI reaches threshold
- iPhone subscribes to alarm characteristic
- Alarm can be triggered over the internet
- Connected state can be read over the internet (occupancy)

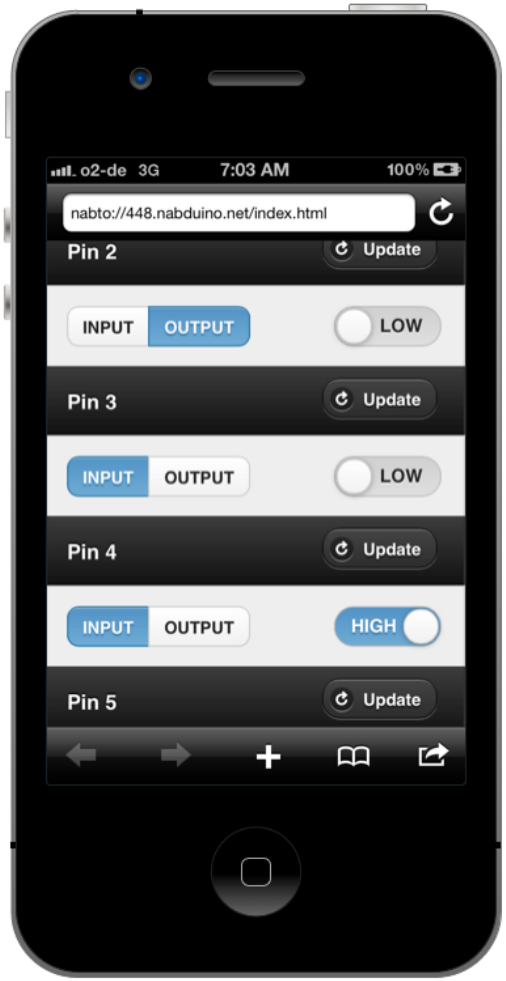
5. System components – BLE web gateway



5. System components – BLE web server

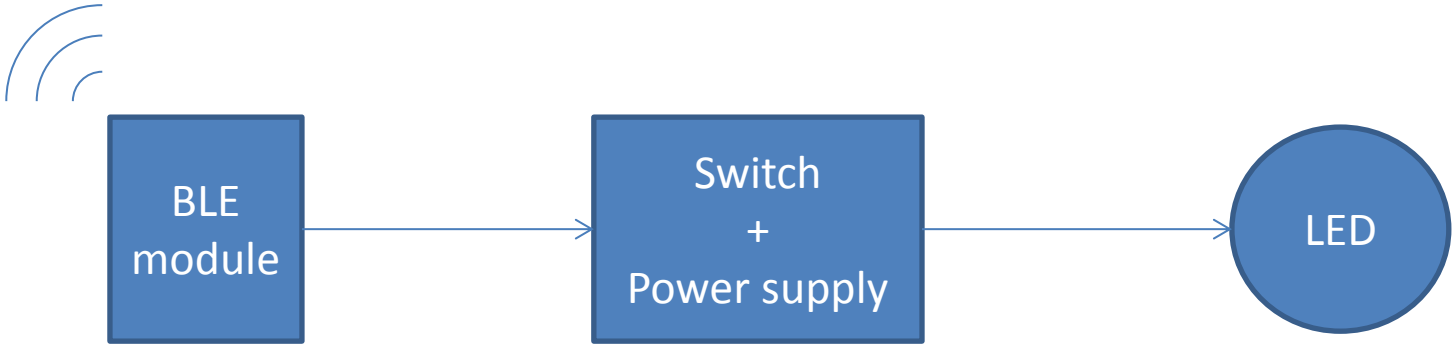


Nabduino webserver



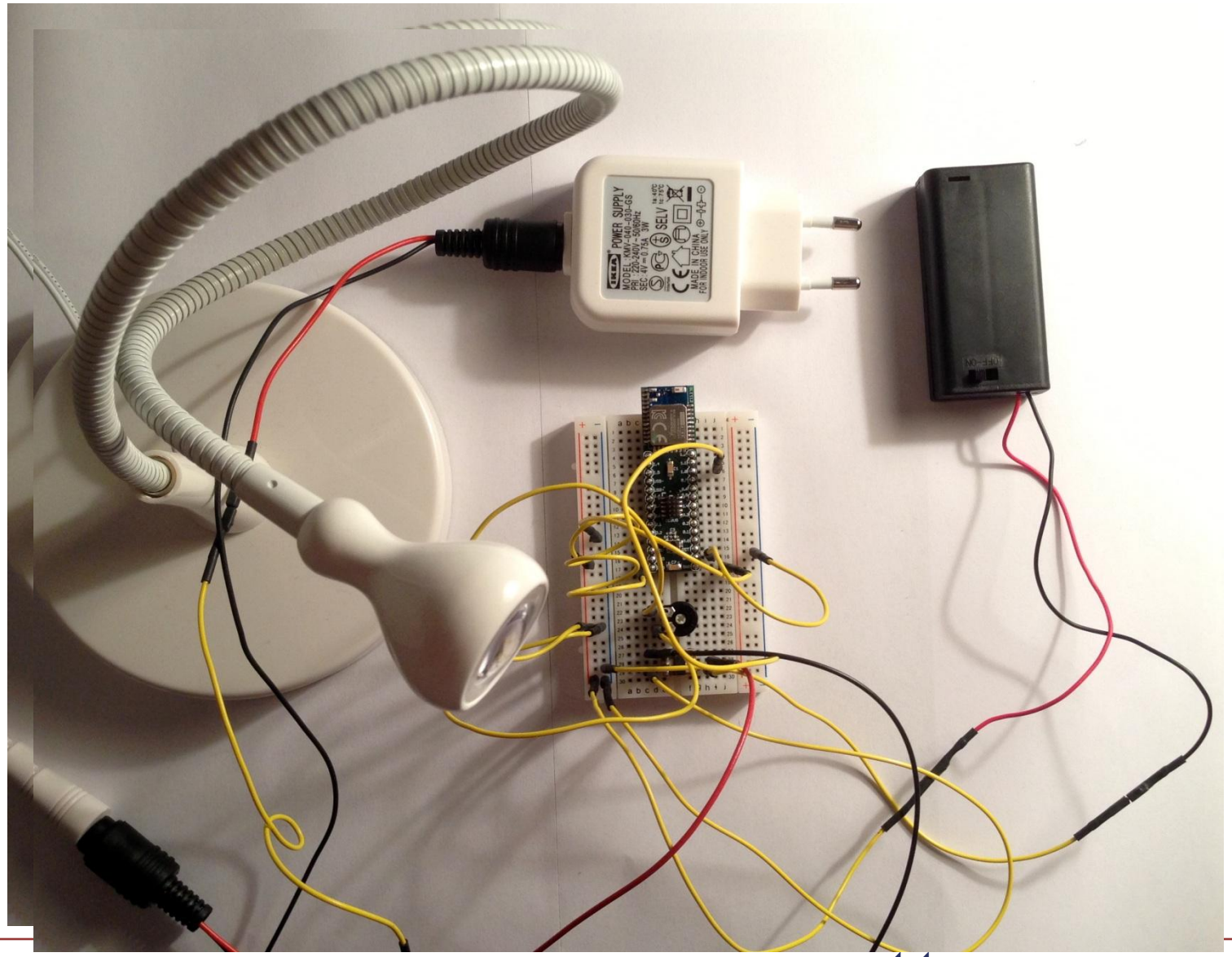
Nabto App

5. System components – BLE Lamp

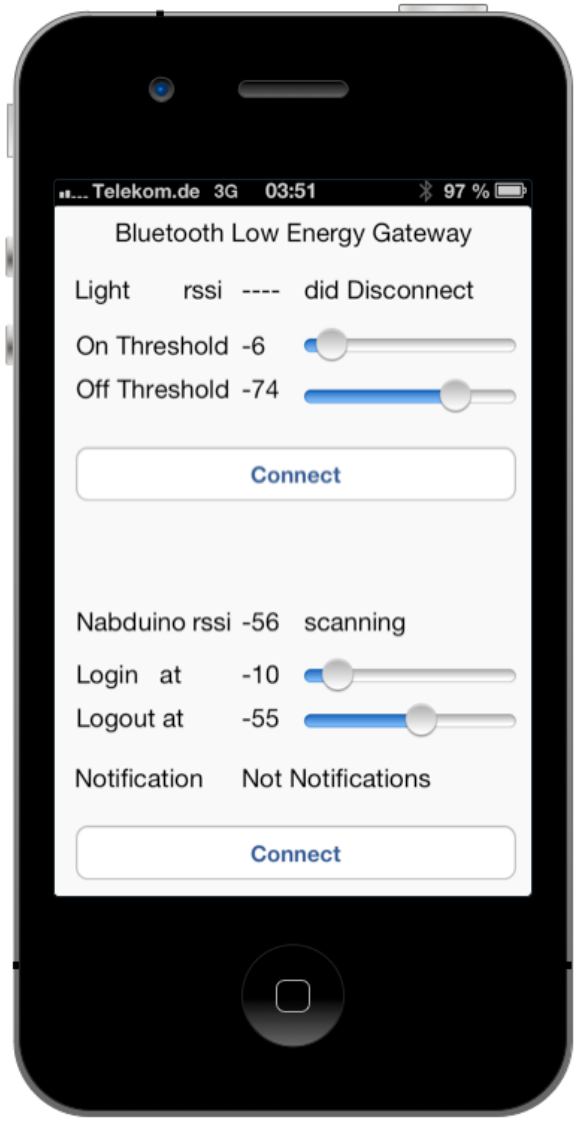


- Central connects to Peripheral
- Mobile Gateway (Central) receives RSSI value of advertisement packets
- Mobile Gateway (Central) connects when RSSI reaches threshold
- When BLE Module connected Lamp is turned on
- Central checks RSSI value form time to time
- Central disconnects if RSSI drops below threshold

4. System components – BLE Lamp



5. System components – Mobile Gateway (iPhone 5)

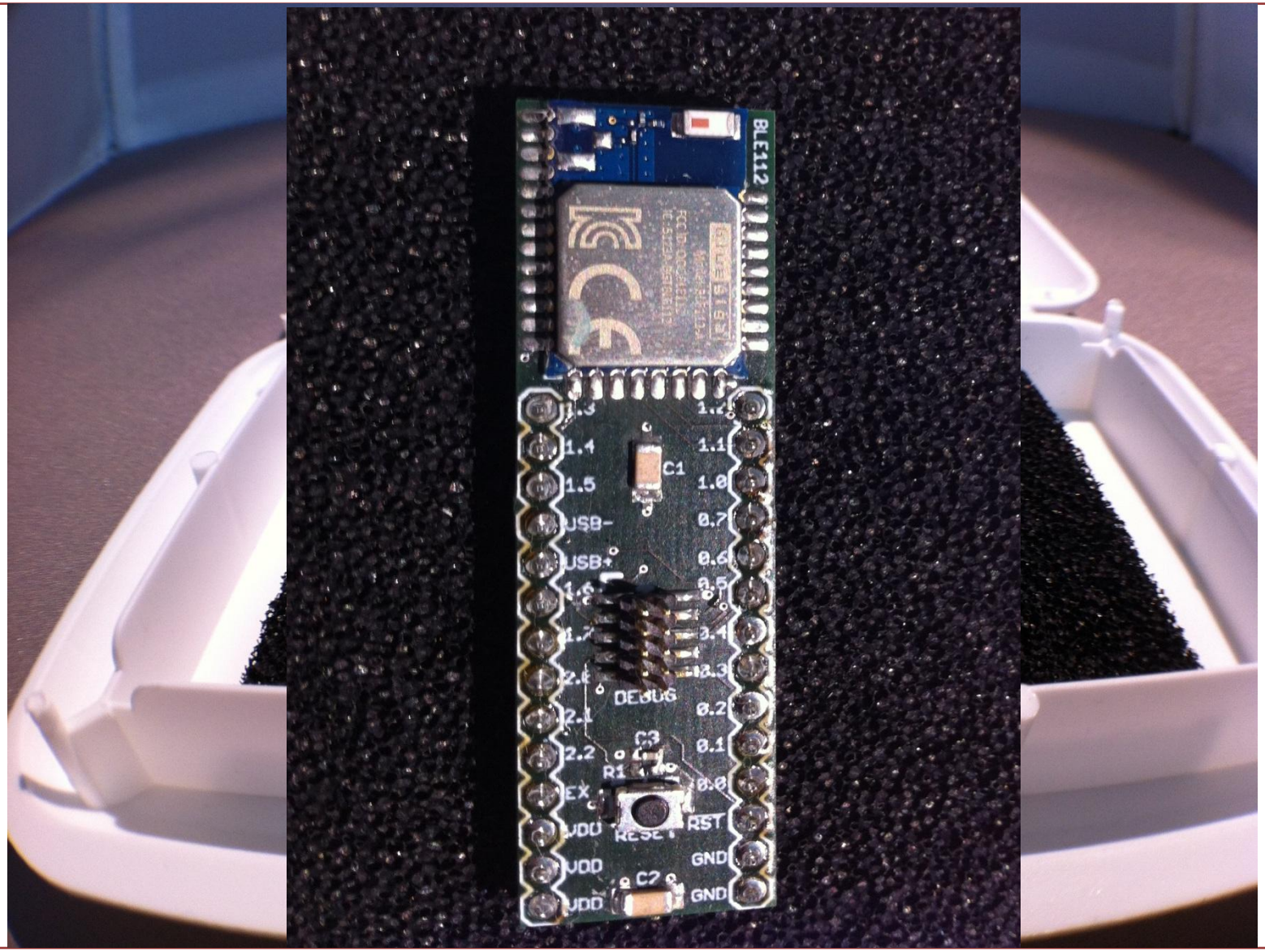


- iPhone scans for peripherals
- If detected RSSI values are shown
- iPhone receives RSSI value of advertisement packets
- iPhone connects when RSSI reaches threshold
- iPhone subscribes too web gateway Alarm notifications
- If iPhone receives notification it displays Alram
- Central checks RSSI value form time to time
- Central disconnects if RSSI drops below threshold

6. Used tools

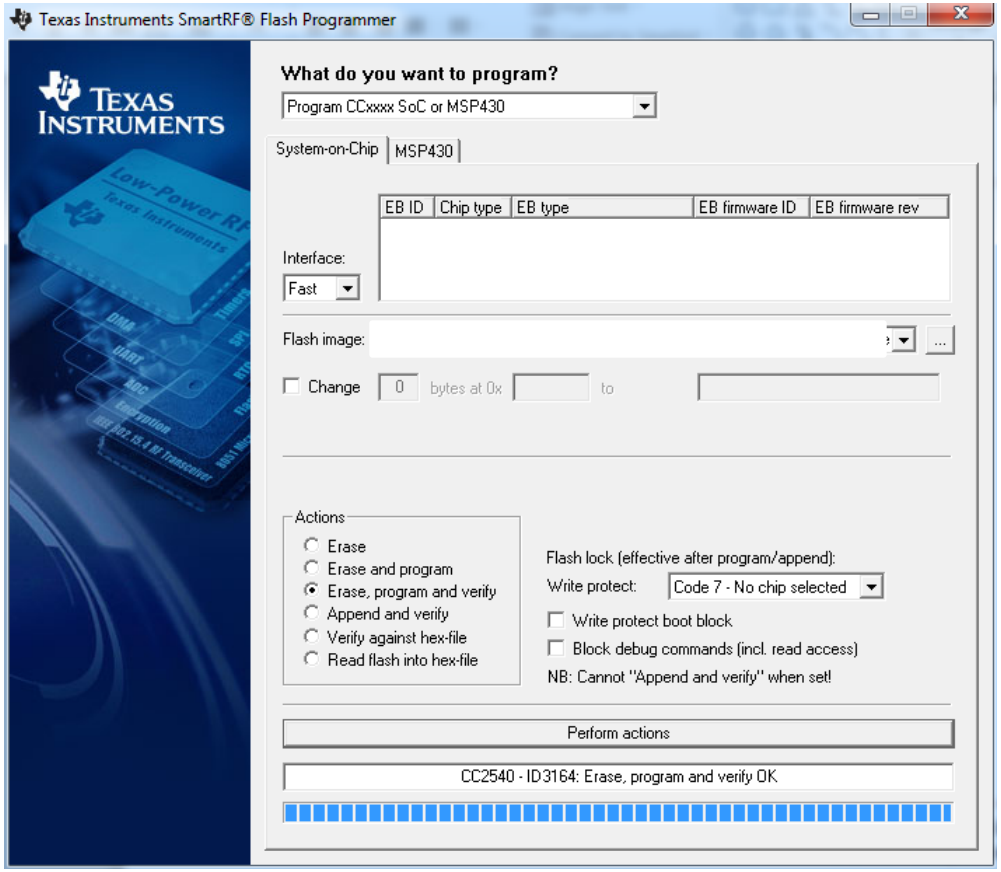
- BGSrcipt from BlueGiga
- BLE Protostick (BLE112) from BLElabs
- TI CC-Debugger and FlashSoftware
- Nabto based Nabduino web-server
-
- Nabto App for Iphone
-
- iPhone 5 as mobile gateway
- Xcode 4.5.2 for App development
- Wi-Fi modem and iPhone 4 as 3G Wi-Fi hotspot

6. Used tools


BLE112
1.4
1.5
USB-
USB+
1.1
1.0
0.7
0.6
0.5
3.4
3.3
0.2
0.1
0.0
C1
C2
C3
R1
RST
GND
GND
VDD
VDD
VDD

6. Used tools

TI CC-Debugger



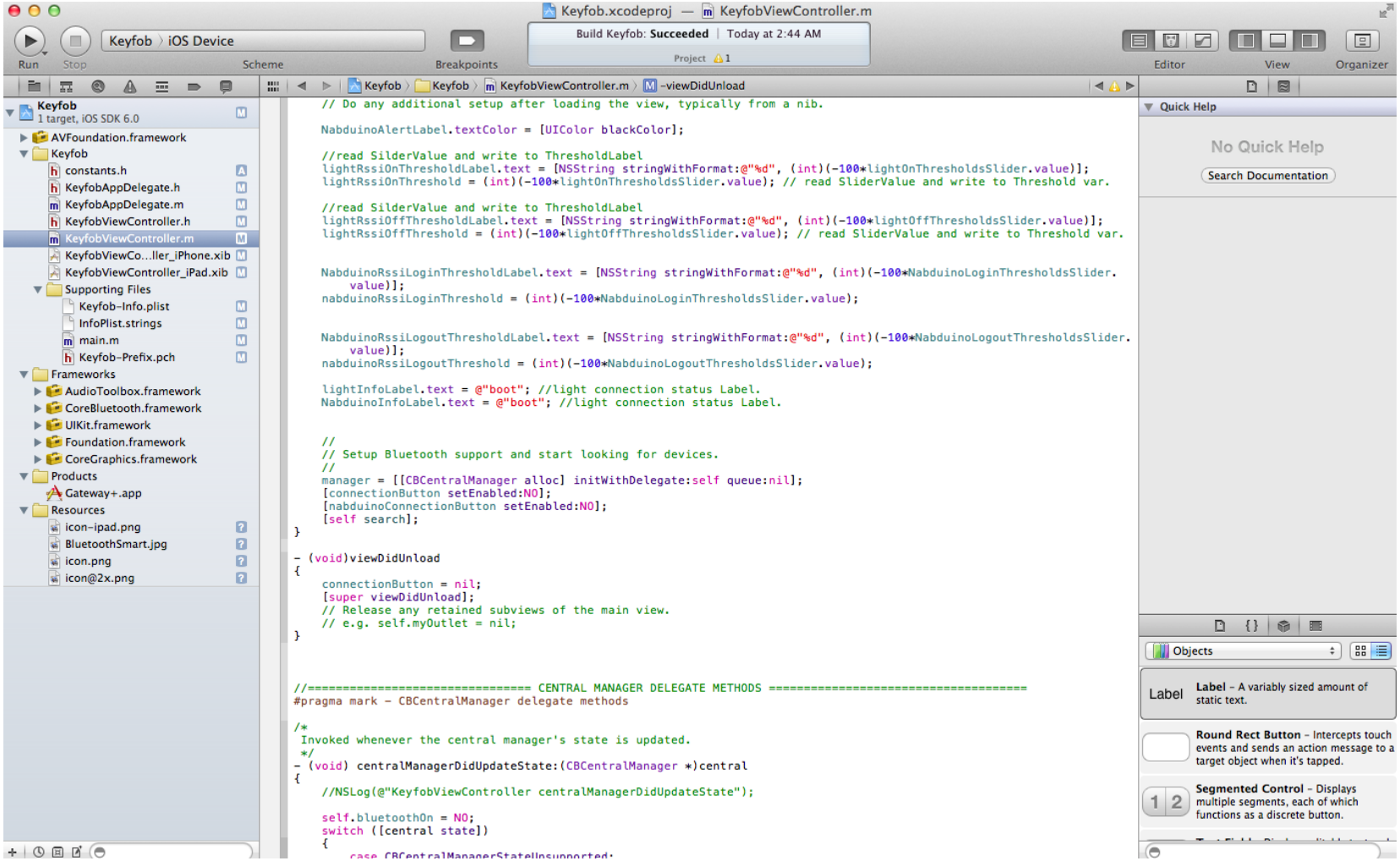
TI SmartRF TM Flash Programmer

6. Used tools

```
1
2   dim tmp(5)
3   dim celsius
4   dim offset
5
6   #init gap mod, bonding and start freerunning times on system boot
7   event system_boot(major ,minor ,patch ,build ,ll_version ,protocol_v
8       call gap_set_mode(2,2)
9       call sm_set_bondable_mode(1)
10      call hardware_set_soft_timer(32000,0,0)
11  end
12
13  #timer expired
14  event hardware_soft_timer(handle)
15      call hardware_adc_read(14,3,0)
16  end
17
18  #ADC ready
19  event hardware_adc_result(input,value)
20      offset=-1490
21
22      # ADC value is 12 MSB
23      celsius = value / 16
24      # Calculate temperature
25      # ADC*V_ref/ADC_max / T_coeff + offset
26      celsius = (10*celsius*1150/2047) * 10/45 + offset
27
28      #set flags
29      tmp(0:1)=0
30
31      #set as float
32      tmp(1:4)=float(celsius, -1)
33
34      call attributes_write(xgatt_temperature_celsius,0,5,tmp(0:5))
35  end
```

BlueGiga BGSrcipt using Notepad++

6. Used tools




```
    // Do any additional setup after loading the view, typically from a nib.

    NabduinoAlertLabel.textColor = [UIColor blackColor];

    //read SilderValue and write to ThresholdLabel
    lightRssiOnThresholdLabel.text = [NSString stringWithFormat:@"%d", (int)(-100*lightOnThresholdsSlider.value)];
    lightRssiOnThreshold = (int)(-100*lightOnThresholdsSlider.value); // read SliderValue and write to Threshold var.

    //read SilderValue and write to ThresholdLabel
    lightRssiOffThresholdLabel.text = [NSString stringWithFormat:@"%d", (int)(-100*lightOffThresholdsSlider.value)];
    lightRssiOffThreshold = (int)(-100*lightOffThresholdsSlider.value); // read SliderValue and write to Threshold var.


    NabduinoRssiLoginThresholdLabel.text = [NSString stringWithFormat:@"%d", (int)(-100*NabduinoLoginThresholdsSlider.
        value)];
    nabduinoRssiLoginThreshold = (int)(-100*NabduinoLoginThresholdsSlider.value);


    NabduinoRssiLogoutThresholdLabel.text = [NSString stringWithFormat:@"%d", (int)(-100*NabduinoLogoutThresholdsSlider.
        value)];
    nabduinoRssiLogoutThreshold = (int)(-100*NabduinoLogoutThresholdsSlider.value);

    lightInfoLabel.text = @"boot"; //light connection status Label.
    NabduinoInfoLabel.text = @"boot"; //light connection status Label.


    //
    // Setup Bluetooth support and start looking for devices.
    //
    manager = [[CBCentralManager alloc] initWithDelegate:self queue:nil];
    [connectionButton setEnabled:NO];
    [nabduinoConnectionButton setEnabled:NO];
    [self search];
}

- (void)viewDidUnload
{
    connectionButton = nil;
    [super viewDidUnload];
    // Release any retained subviews of the main view.
    // e.g. self.myOutlet = nil;
}



//============================== CENTRAL MANAGER DELEGATE METHODS ===================================
#pragma mark - CBCentralManager delegate methods

/*
 Invoked whenever the central manager's state is updated.
 */
- (void) centralManagerDidUpdateState:(CBCentralManager *)central
{
    //NSLog(@"KeyfobViewController centralManagerDidUpdateState");

    self.bluetoothOn = NO;
    switch ([central state])
    {
        case CBCentralManagerStateUnsupported:
```

Apple Xcode

- Allows to permanently scan for devices in the background on smartphones

- Makes context awareness possible

- Low data rate as drawback

- The peripherals can run on a coin cell for long time

- Gateways should be Central and Peripheral
- Low Power technology allows back grounding on cellphones
- HCI interface over the internet in future implementations
- Peripherals should timeout (cancel connection after a while)
- Multiple connections in peripheral mode
- Generic Devices

Demo

Thank
You