

HES-SO MSE

Hes·SO

Haute Ecole Spécialisée
de Suisse occidentale

DESIGN OF COMMUNICATIVE EMBEDDED SYSTEMS
S1-2021

DeSEm project
10/01/2021

Fabio Baldo

Contents

1	Diagrams	2
1.1	DeseNET protocol documentation	2
1.2	Joystick application documentation	6
2	Tests	6
2.1	DeseNET protocol test	6
2.2	Joystick application test	6
2.3	Results	7
2.4	Joystick application documentation	8

1 Diagrams

1.1 DeseNET protocol documentation

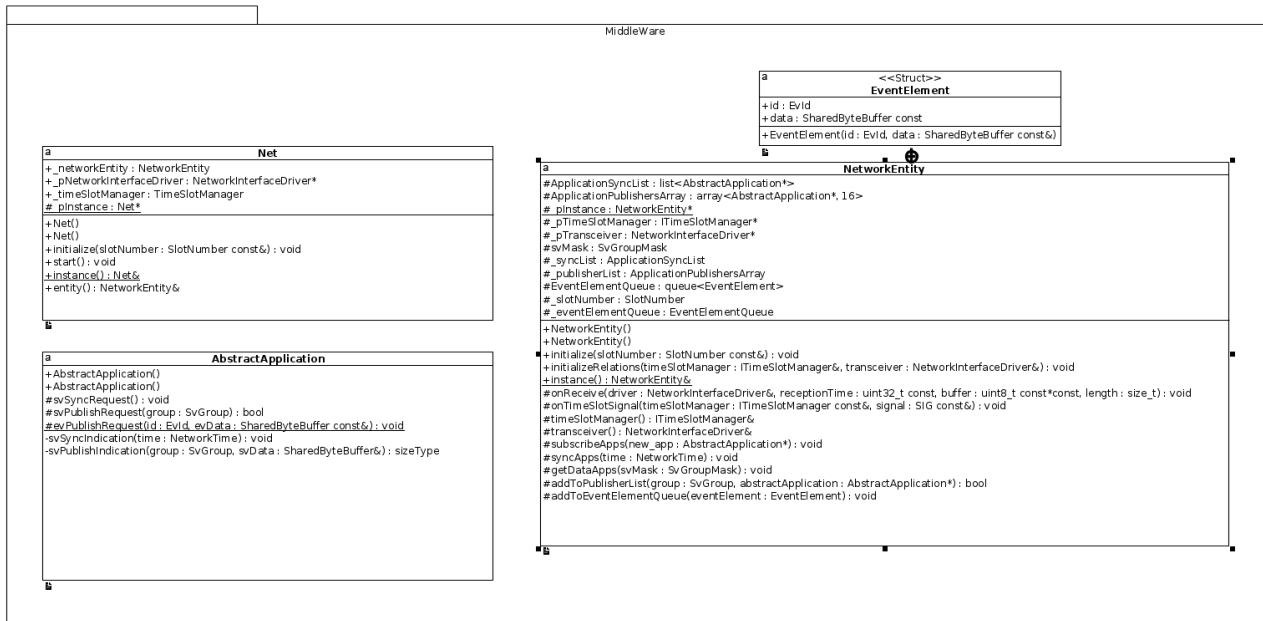


Figure 1: Class diagram of the sensor middleware

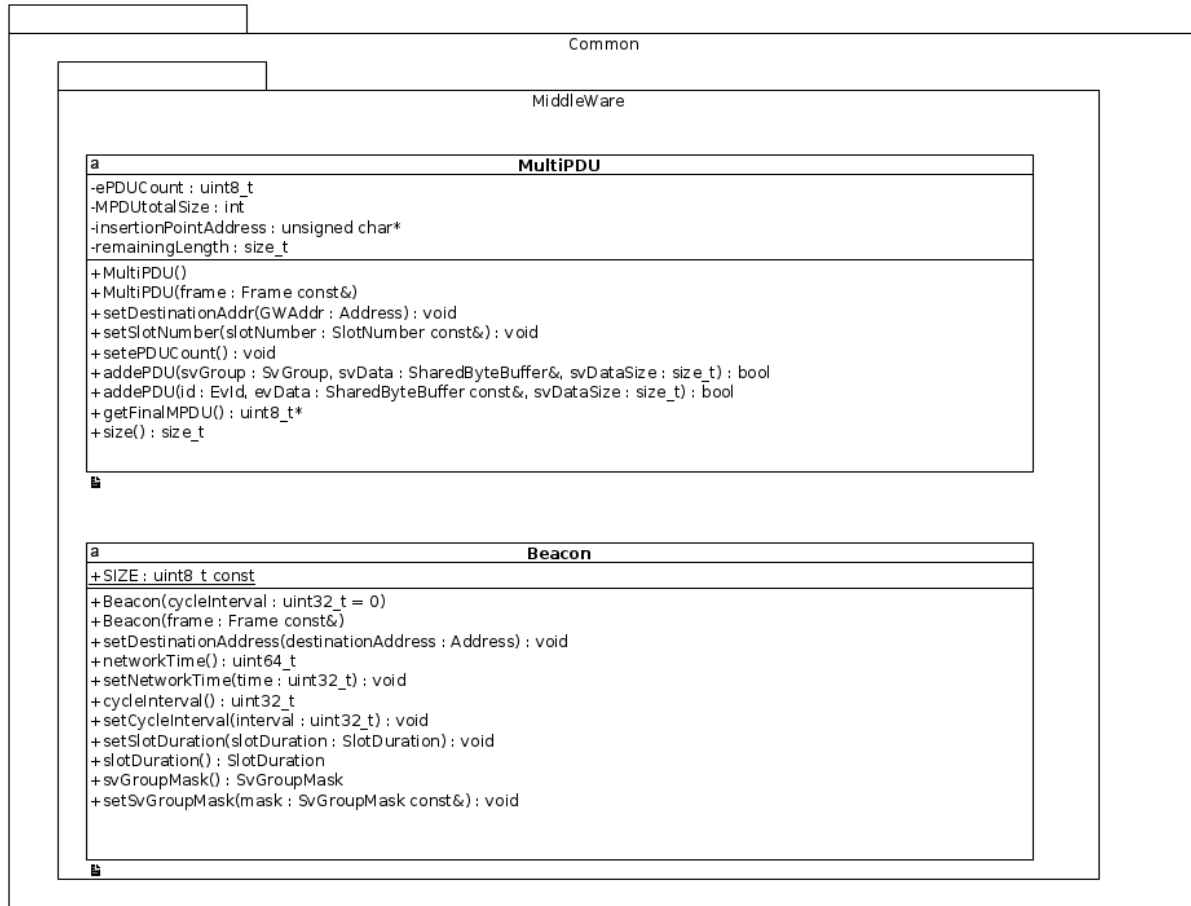


Figure 2: Class diagram of the two main classes of the Common middleware

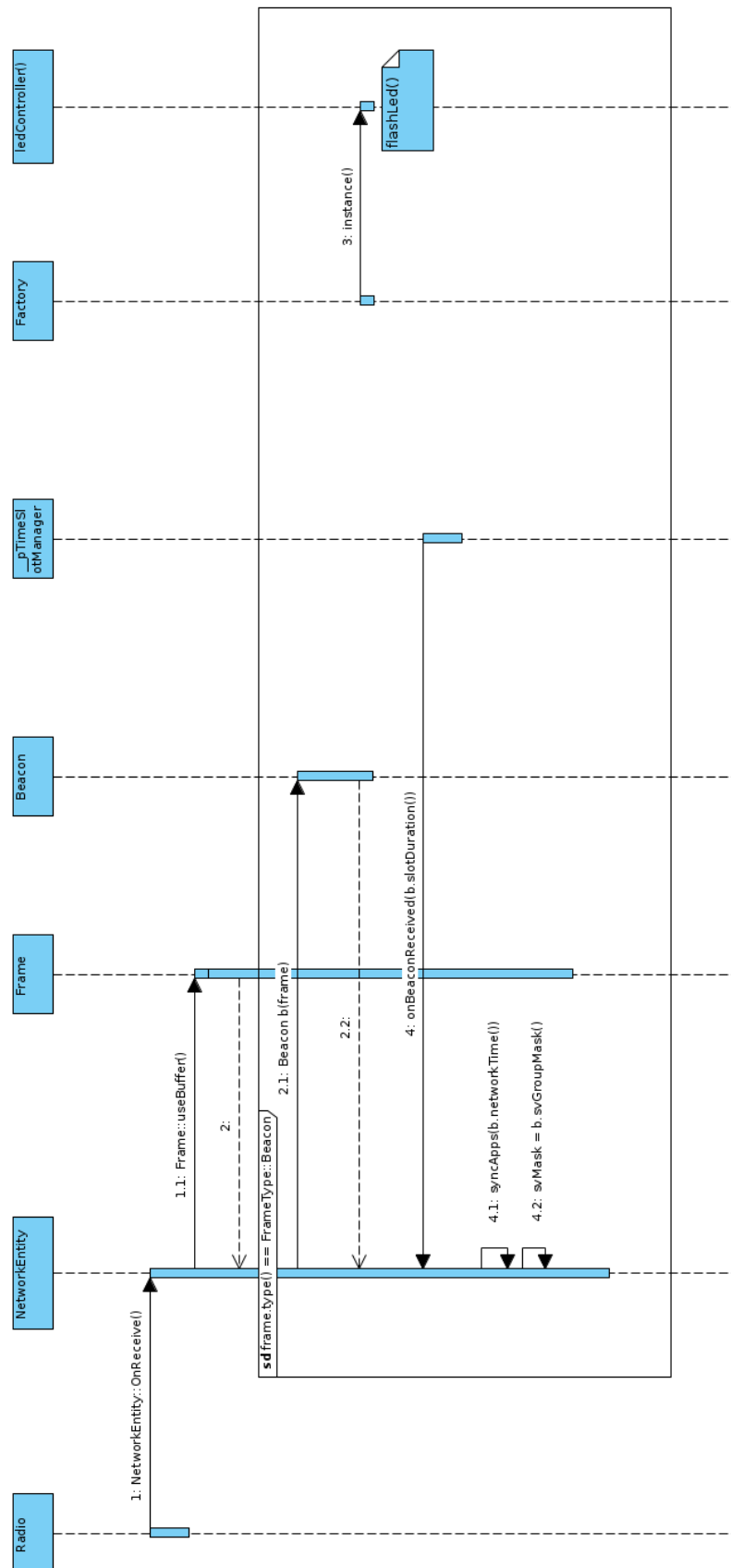


Figure 3: Action diagram of the function onReceive called when a Beacon is received

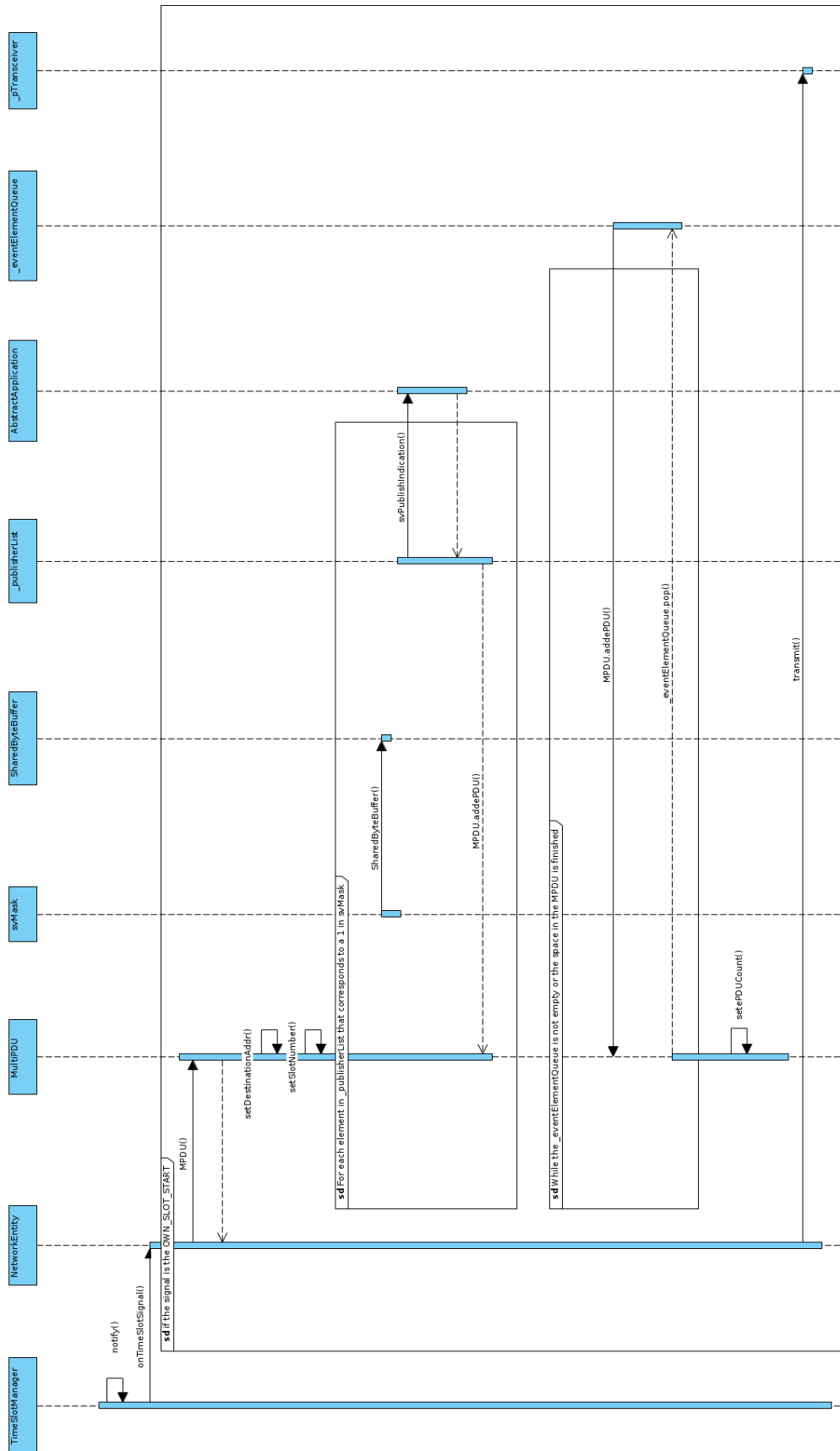


Figure 4: Action diagram of the function `onTimeSlotSignal` called when the timer for the slot runs out

1.2 Joystick application documentation

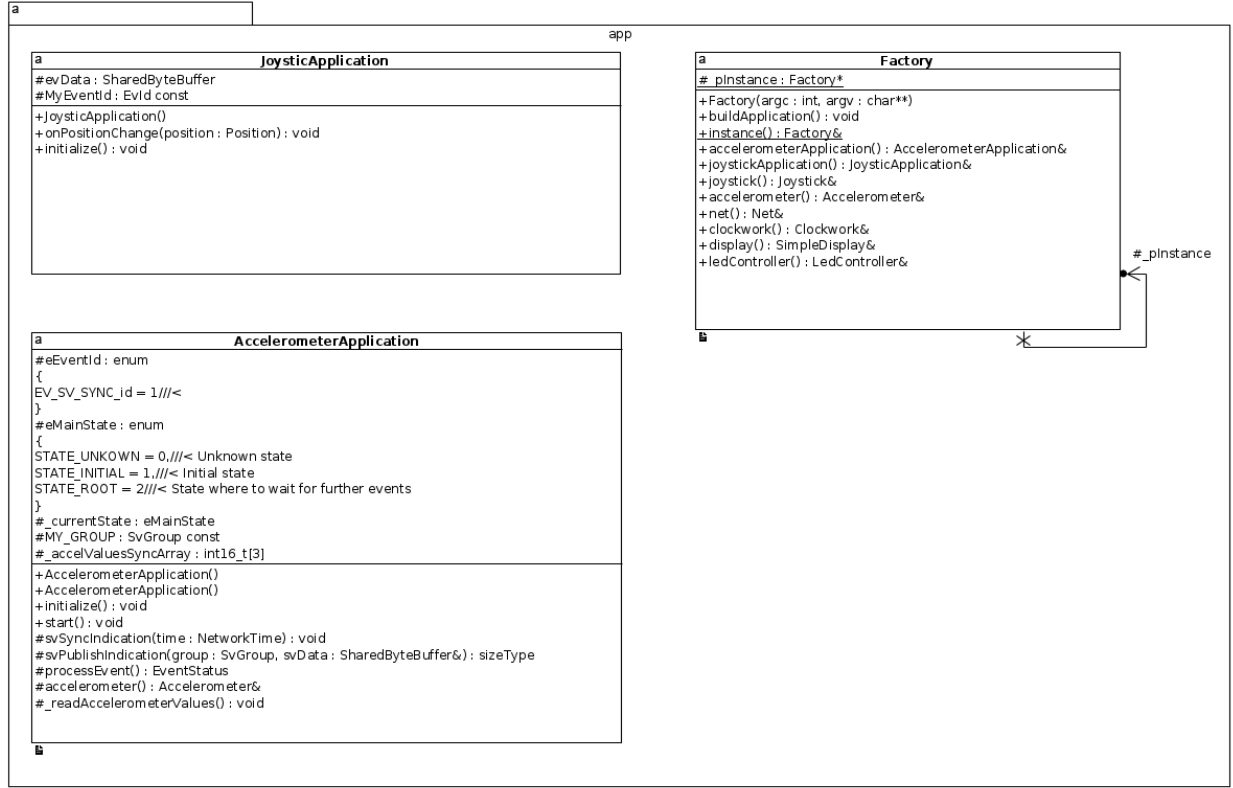


Figure 5: Class diagram of the two main classes of the Common middleware

2 Tests

2.1 DeseNET protocol test

In order to test the good functioning of the entire deseNET protocol a combination of Tracing and repeated test has been used. For controlling the correctness of the send data, using the Trace class and its method `outln` the content of the variables has been written to the console both when the data is sampled and send back to the Abstract application and when the MPDU is formed. The whole buffer hosting the MPDU data has been printed byte per byte in order to make it more readable for understanding its content. This MPDU eventually has been tested against the value shown in the mesh simulator for a final check. In the following listings and images is reported a sample of the testing results.

2.2 Joystick application test

Like in the previous section the correctness of the data of the events has been performed by a combination of tracing and repeated test with the double check of the result shown in the mesh simulator. The sample reported in the following images and listing represents a case in which the left button of the joystick has been pressed 3 times. Unfortunately the function, where

the evData are copied into the queue used for building the MPDU, is called two times. Many test have been performed in order to understand from where the error comes. Testing also the official demo it seems that when only one push is done to the button then two events are send to the GW. In order to try to avoid this problem during the tests of the rest of the application, the values of the push button have been set manually to 0xAB. With this simple modification and a small one in the code where only one event over a couple is added to the the MPDU the result has been confirmed to be correct. In the reported example the official demo joystick has been pushed only once.

2.3 Results

Highlighted are the lines of the resulting MPDU containing the values.

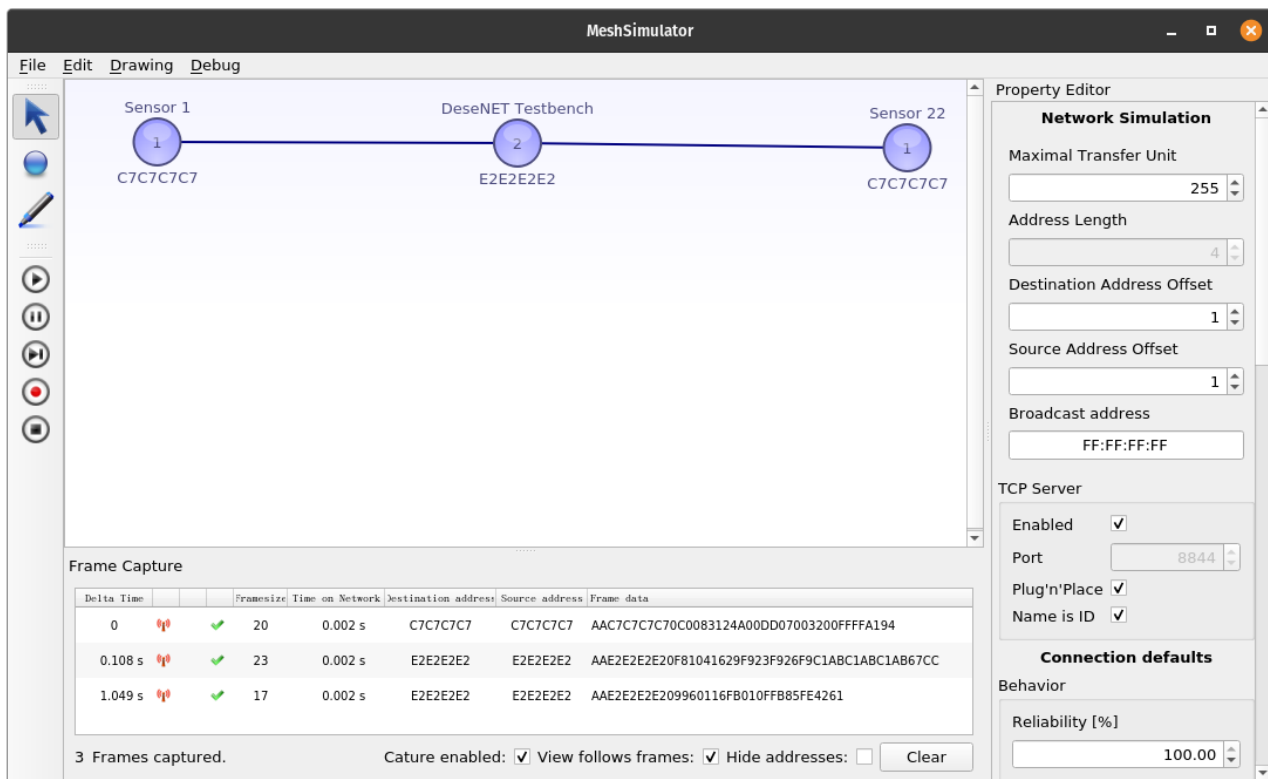


Figure 6: Screenshot of the simulator window after the MPDU was received from the Sensor back to the GW

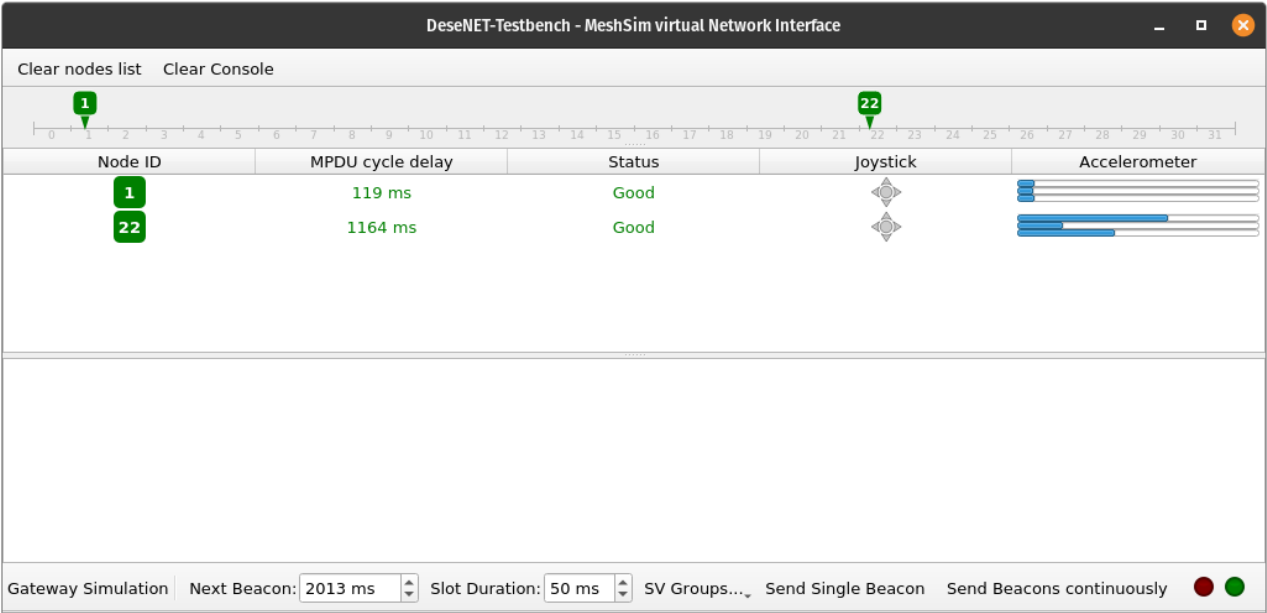


Figure 7: Screenshot of the testbench window after the MPDU was received from the Sensor back to the GW

```
1  AcceApplication data send: 29F923F926F9
2  NE_AcceApplication data send: 29F923F926F9
3  MPDU before sv-start
4  E2
5  E2
6  E2
7  E2
8  20
9  81
10 0
11 0
12 0
13 0
14 0
15 0
16 0
17 0
18 0
19 0
20 18
21 0
22 0
23 0
24 0
25 0
26 0
27 0
28 0
29 0
```

```
30 0
31 0
32 27
33 1
34 B5
35 1
36 E8
37 3
38 0
39 0
40 0
41 MPDU before sv-end
42 MPDU: svData 29F923F926F9
43 MPDU: mySVePDU 29F923F926F9
44 MPDU after sv-start
45 E2
46 E2
47 E2
48 E2
49 9
50 81
51 0
52 16
53 29
54 F9
55 23
56 F9
57 26
58 F9
59 MPDU after sv-end
60 NE_JoysticApplication: Data send: AB
61 MPDU before event-start
62 E2
63 E2
64 E2
65 E2
66 9
67 81
68 0
69 16
70 29
71 F9
72 23
73 F9
74 26
75 F9
76 MPDU before event-end
77 MPDU after event-start
78 E2
79 E2
80 E2
81 E2
82 B
83 81
```

```
84  0
85  16
86  29
87  F9
88  23
89  F9
90  26
91  F9
92  C1
93  AB
94  MPDU after event-end
95  NE_JoysticApplication: Data send: AB
96  MPDU before event-start
97  E2
98  E2
99  E2
100 E2
101 B
102 81
103 0
104 16
105 29
106 F9
107 23
108 F9
109 26
110 F9
111 C1
112 AB
113 MPDU before event-end
114 MPDU after event-start
115 E2
116 E2
117 E2
118 E2
119 D
120 81
121 0
122 16
123 29
124 F9
125 23
126 F9
127 26
128 F9
129 C1
130 AB
131 C1
132 AB
133 MPDU after event-end
134 NE_JoysticApplication: Data send: AB
135 MPDU before event-start
136 E2
137 E2
```

```
138 E2
139 E2
140 D
141 81
142 0
143 16
144 29
145 F9
146 23
147 F9
148 26
149 F9
150 C1
151 AB
152 C1
153 AB
154 MPDU before event-end
155 MPDU after event-start
156 E2
157 E2
158 E2
159 E2
160 F
161 81
162 0
163 16
164 29
165 F9
166 23
167 F9
168 26
169 F9
170 C1
171 AB
172 C1
173 AB
174 C1
175 AB
176 MPDU after event-end
177 MPDU-start
178 E2
179 E2
180 E2
181 E2
182 F
183 81
184 4
185 16
186 29
187 F9
188 23
189 F9
190 26
191 F9
```

192	C1
193	AB
194	C1
195	AB
196	C1
197	AB
198	MPDU-end
199	