Master 2 MoSIG

Advanced Software Modelling & Engineering

Lab: UML-RT

Authors:
Quentin Nivon
William Templier

Lecturer:
Nicolas Hili

November 30, 2021

Exercise 2

From the code of the `traffic-light.rt` file, we drew the following schemes, where Figure 1 represents the state machine and Figure 2 the capsule diagram.

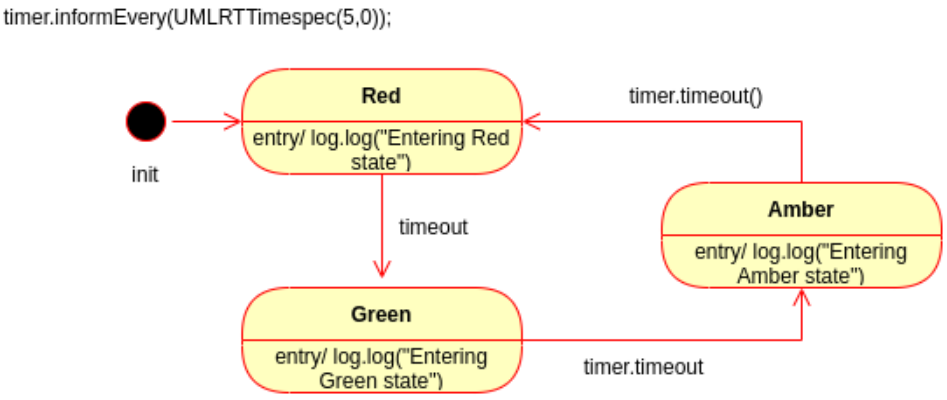

Figure 1: State machine

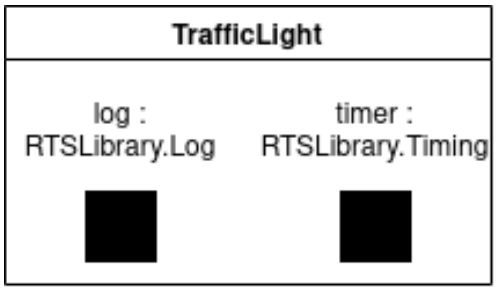

Figure 2: Capsule diagram

Regarding the timer part, we made the following changes:

```
state Red {
    entry `log.log("Entering Red state");
        timer.informIn(UMLRTTimespec(5, 0));`
}
state Green {
    entry `log.log("Entering Green state");
        timer.informIn(UMLRTTimespec(5, 0));`
}
state Amber {
    entry `log.log("Entering Amber state");
```

```
        timer.informIn(UMLRTTimespec(2, 0));
}
```

Exercise 3

From the code of the `ping-pong.rt` file, we drew the following schemes, where Figure 3 represents the state machine and Figure 4 the capsule diagram.

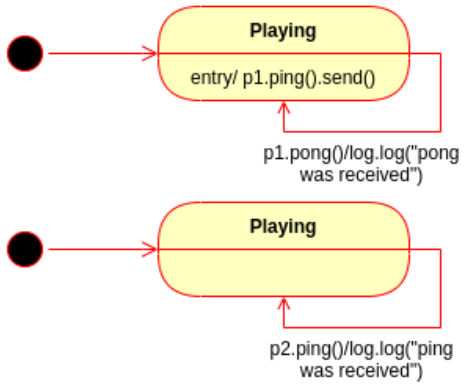

Figure 3: State machine

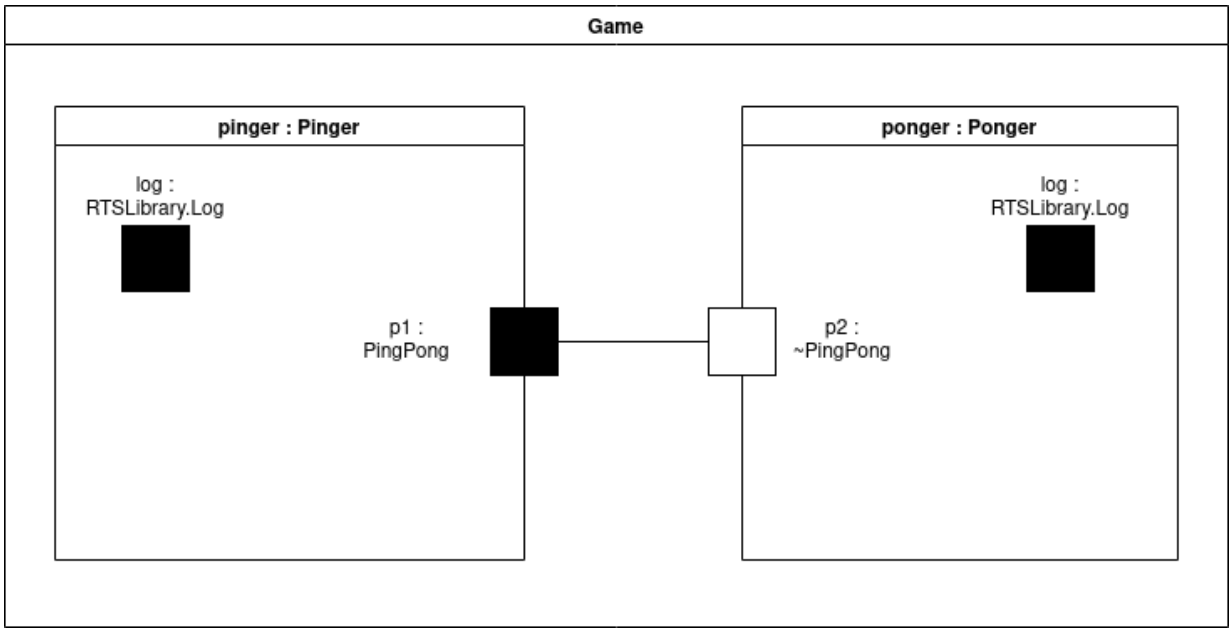

Figure 4: Capsule diagram

To produce an infinite trace of ping and pong messages, we did the following in the `Ponger` capsule:

```
transition {
    from Playing to Playing
    triggers from p2 on ping
    action `log.log("ping was received");
            p2.pong().send();`
}
```

Exercise 4

From the code of the `traffic-light-with-replication.rt` & `led.rt` files, we drew the following schemes, where Figure 6 represents the state machine and Figure **??** the capsule diagram.

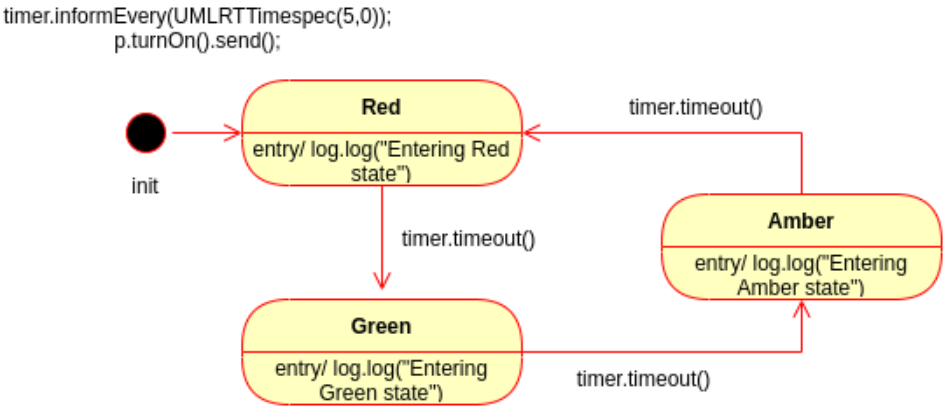

Figure 5: State machine

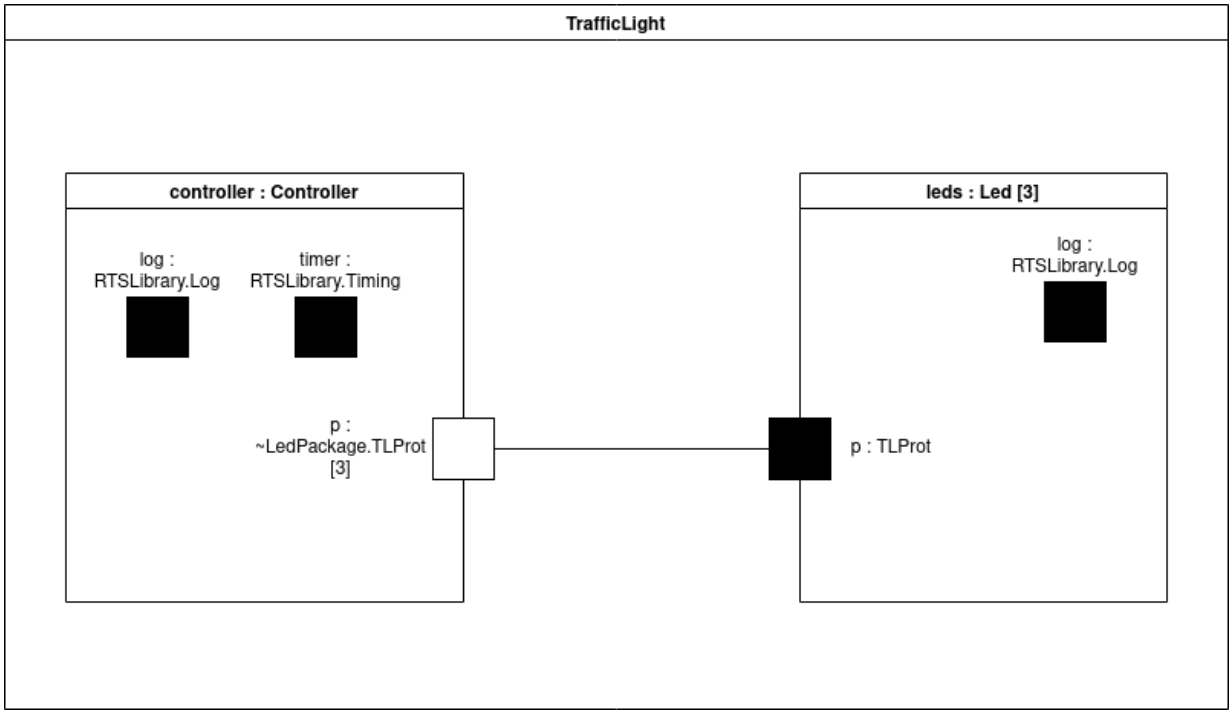

Figure 6: Capsule diagram

Exercise 6

To make use of two traffic lights at the same time, we simply added a new `AssignementProtocol` port and a new `TrafficLight` instance to the `TestCapsule`, where the connections are identical to the first one. Then we just assigned the GPIOs 12, 13 & 14 to the new port, and it worked.

For the synchronization part, we simply made sure that the two lights always start in different states (i.e. one initial $\rightarrow$ red & one initial $\rightarrow$ green) by creating two capsules `TrLight1` and `TrLight2`. We set the durations to $\text{duration}_{\text{red}} = \text{duration}_{\text{amber}} + \text{duration}_{\text{green}}$, which ensures that the lights are synchronized.

All of these things are visible in the code files attached (`Crossroad/cpp/cr.rt`).