

Helena 2.3

Example 1 - The distributed database system

Sami Evangelista - (Sami [dot] Evangelista [at] lipn.univ-paris13 [dot] fr)

August 29, 2017

We consider in this system a set of N database managers which communicate to maintain consistent replica of a database. It is a well-known and recurrent example of the colored Petri nets literature, initially presented by Genrich and later by Jensen.

When a manager updates its local copy of the database, he sends requests to other managers for updating their local copy (transition *Update*). As soon as a manager receives such a request (transition *Receive*) he starts the update of its copy. Its update finished, each manager acknowledges the initiating manager (transition *Send ack*). This process finishes when the initiating manager collects all the acknowledgments (transition *Receive acks*). Managers can be either *Inactive*, either *Waiting* for acknowledgments, either *Performing* an update. Places *Msgs*, *Received*, *Acks* and *Unused* model communication channels between sites. Thus, $N \cdot (N - 1)$ tokens are distributed upon these places at each marking. At last the correctness of the protocol is ensured by place *Mutex* which guarantees that two managers cannot concurrently update their local copy.

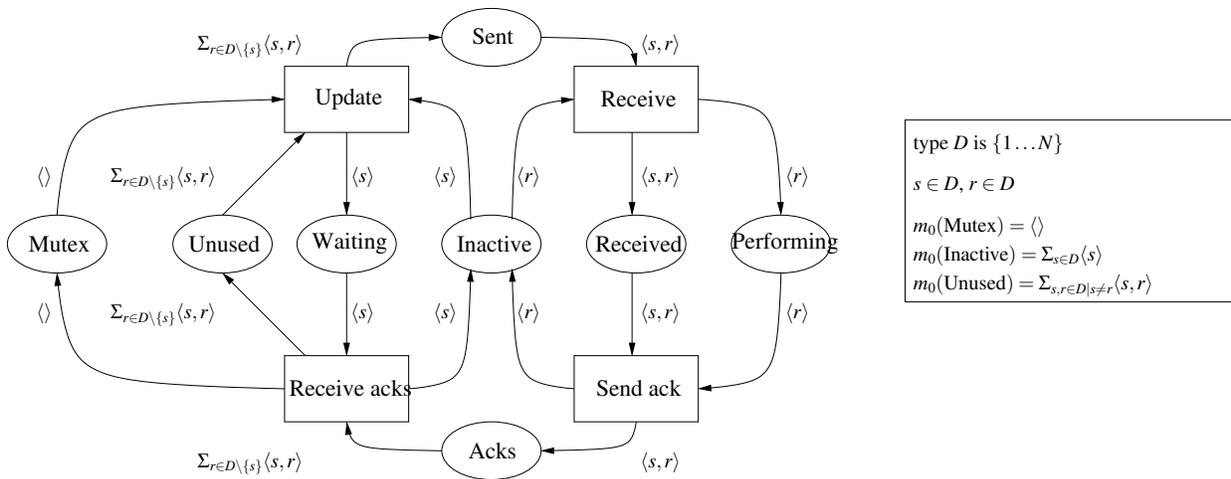


Figure 1: The distributed database system

Listing 1: Helena file of the distributed database system (file examples/dbm.lna)

```

1 /* *****
2 *
3 * Example file of the Helena distribution
4 *
5 * File : dbm.lna
6 * Author: Sami Evangelista
7 * Date : 27 oct. 2004
8 * Source:
9 * Coloured Petri Nets: A high level language for system design and analysis
10 * In Application and Theory of Petri Nets, p.342--416, Springer, 1989
11 * Kurt Jensen
12 *
13 * If symbol UNUSED is defined, the model includes the place unused.
14 *
15 ***** */

```

```

16 dbm (N := 10) { /* N = number of sites */
17
18     type site_id : mod N;
19
20     /*
21      * process places modelling the control flow of processes
22      */
23     place inactive {
24         dom : site_id;
25         init : for(s in site_id) <( s )>;
26         capacity : 1;
27         type: process;
28     }
29
30     place waiting {
31         dom : site_id;
32         capacity : 1;
33         type: process;
34     }
35
36     place performing {
37         dom : site_id;
38         capacity : 1;
39         type: process;
40     }
41
42     /*
43      * places modelling communication channels
44      */
45     place sent {
46         dom : site_id * site_id;
47         capacity : 1;
48         type: buffer;
49     }
50
51     place received {
52         dom : site_id * site_id;
53         capacity : 1;
54         type: buffer;
55     }
56
57     place acks {
58         dom : site_id * site_id;
59         capacity : 1;
60         type: ack;
61     }
62
63     place unused {
64         dom : site_id * site_id;
65         init : for(s in site_id, r in site_id) if(s != r) <( s, r )>;
66         capacity : 1;
67         type: buffer;
68     }
69
70     place mutex {
71         dom : epsilon;
72         init : epsilon;
73         capacity : 1;
74         type: shared;
75     }
76
77     transition update_and_send {
78         in {
79             inactive : <( s )>;
80             mutex : epsilon;

```

```

78 #ifdef UNUSED
79     unused    : for(r in site_id) if(s != r) <( s, r )>;
80 #endif
81     }
82     out {
83         waiting : <( s )>;
84         sent    : for(r in site_id) if(s != r) <( s, r )>;
85     }
86 }
87 transition receive_acks {
88     in {
89         waiting : <( s )>;
90         acks    : for(r in site_id) if(s != r) <( s, r )>;
91     }
92     out {
93         inactive : <( s )>;
94         mutex    : epsilon;
95 #ifdef UNUSED
96     unused    : for(r in site_id) if(s != r) <( s, r )>;
97 #endif
98     }
99 }
100 transition receive_message {
101     in {
102         inactive : <( r )>;
103         sent     : <( s, r )>;
104     }
105     out {
106         performing : <( r )>;
107         received   : <( s, r )>;
108     }
109 }
110 transition send_ack {
111     in {
112         performing : <( r )>;
113         received   : <( s, r )>;
114     }
115     out {
116         inactive : <( r )>;
117         acks     : <( s, r )>;
118     }
119 }
120
121 /*
122 * state propositions
123 */
124 proposition site_waiting : waiting card > 0;
125 }

```

Listing 2: Helena file of the distributed database system properties (file examples/dbm.prop.lna)

```

1 /*
2 * a site waiting for answer will eventually leave this state
3 */
4 ltl property bounded_wait:
5     ([ (site_waiting => <> (not site_waiting)));

```