# Helena 2.3
# Example 1 - The distributed database system

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

March 28, 2013

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, $N.(N-1)$ 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.

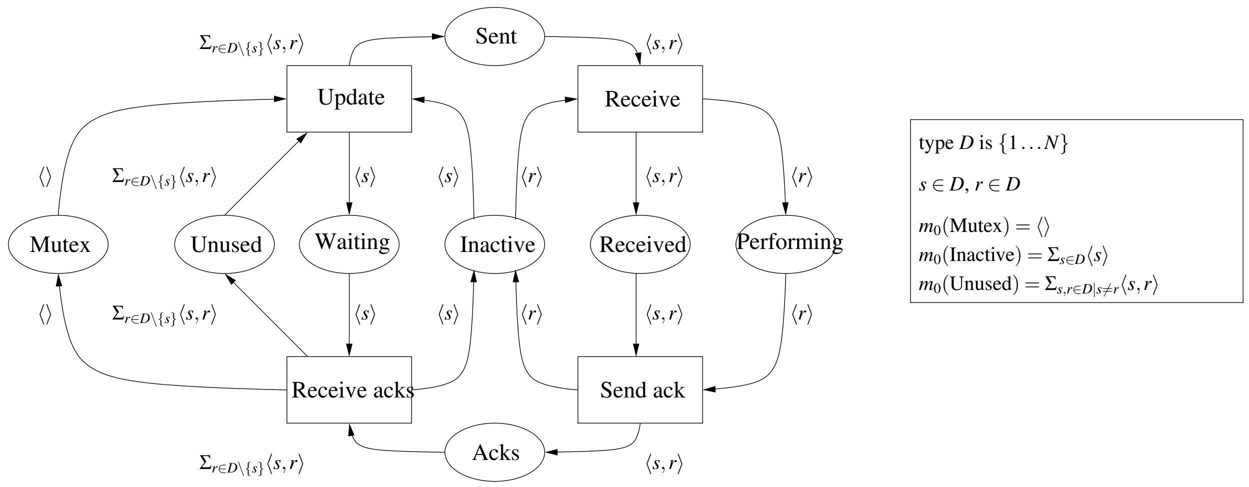
Sent
Update
Receive
Mutex
Unused
Waiting
Inactive
Received
Performing
Receive acks
Send ack
Acks
$\Sigma_{r\in D\setminus\{s\}}\langle s,r\rangle$
$\langle s,r\rangle$
$\langle\rangle$
$\langle s\rangle$
$\langle r\rangle$
type $D$ is $\{1\ldots N\}$
$s\in D, r\in D$
$m_0(\text{Mutex}) = \langle\rangle$
$m_0(\text{Inactive}) = \Sigma_{s\in D}\langle s\rangle$
$m_0(\text{Unused}) = \Sigma_{s,r\in D|s\neq r}\langle s,r\rangle$

Figure 1: The distributed database system

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

```
/*****************************************************************************
*
*  Example file of the Helena distribution
*
*  File   : dbm.lna
*  Author: Sami Evangelista
*  Date   : 27 oct. 2004
*  Source:
*     Coloured Petri Nets: A high level language for system design and analysis
*     In Application and Theory of Petri Nets, p.342--416, Springer, 1989
*     Kurt Jensen
*
*  If symbol UNUSED is defined, the model includes the place unused.
*
*****************************************************************************/
```

```

dbm (N := 10) {   /*  N = number of sites   */

   type site_id : mod N;

   /*
    *  process places modelling the control flow of processes
    */
   place inactive {
      dom : site_id;
      init : for(s in site_id) <( s )>;
      capacity : 1;
      type: process;
   }
   place waiting {
      dom : site_id;
      capacity : 1;
      type: process;
   }
   place performing {
      dom : site_id;
      capacity : 1;
      type: process;
   }

   /*
    *  places modelling communication channels
    */
   place sent {
      dom : site_id * site_id;
      capacity : 1;
      type: buffer;
   }
   place received {
      dom : site_id * site_id;
      capacity : 1;
      type: buffer;
   }
   place acks {
      dom : site_id * site_id;
      capacity : 1;
      type: ack;
   }
#ifdef UNUSED
   place unused {
      dom : site_id * site_id;
      init : for(s in site_id, r in site_id) if(s != r) <( s, r )>;
      capacity : 1;
      type: buffer;
   }
#endif
   place mutex {
      dom : epsilon;
      init : epsilon;
      capacity : 1;
      type: shared;
   }

   transition update_and_send {
      in {
         inactive : <( s )>;
         mutex    : epsilon;
#ifdef UNUSED
         unused   : for(r in site_id) if(s != r) <( s, r )>;
#endif
      }
      out {
         waiting : <( s )>;
         sent    : for(r in site_id) if(s != r) <( s, r )>;
      }
   }
   transition receive_acks {
      in {
         waiting : <( s )>;
         acks    : for(r in site_id) if(s != r) <( s, r )>;
      }
      out {
         inactive : <( s )>;
         mutex    : epsilon;
#ifdef UNUSED
         unused   : for(r in site_id) if(s != r) <( s, r )>;
#endif
      }
   }
   transition receive_message {
      in {
         inactive : <( r )>;
         sent     : <( s, r )>;
      }
      out {
         performing : <( r )>;
         received   : <( s, r )>;
      }
   }
   transition send_ack {
      in {
         performing : <( r )>;
         received   : <( s, r )>;
      }
      out {
         inactive : <( r )>;
         acks     : <( s, r )>;
      }
   }

   /*
    *  state propositions
    */
   proposition site_waiting: waiting'card > 0;
}
```

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

```
/*
 *  a site waiting for answer will eventually leave this state
 */
ltl property bounded_wait:
   ([] (site_waiting => <> (not site_waiting)));
```