Case study: Online auction service

The case study concerns a typical online auction process, which consists of three participants: the online auction system and two buyers, A_1 and A_2. A seller owes a good that wants to sell to the highest possible price. Therefore, he introduces the product in an auction system for a certain time. Then, buyers (or bidders) may place bids for the product and, when time runs out, the highest bid wins. In our case, we suppose the resource is the product for auction, the value of the resource property is the current price (only the auction system can modify it), the resource subscribers will be the buyers, their subscription conditions hold when the current product value is higher than their bid, and the resource lifetime will be the time in which the auction is active. Finally, when the lifetime has expired, the auction system sends a notification to the buyers with the result of the process (the identifier of the winner, v_w) and, after that, all the processes finish. Let us consider the choreography $C = (O_{sys}, O_1, O_2)$, where $O_i = (PL_i, Var_i, A_i, A_{f_i}, \mathcal{A}_{e_i})$, i=1,2, $Var_{sys} = \{v_w, v_1, v_2, v_{EPR}, at, t\}$, $Var_1 = \{at_1, v_1, v_{w_1}\}$, $Var_2 = \{at_2, v_2, v_{w_2}\}$, $A_{f_1} = exit$, and $A_{f_2} = exit$. Variable v_{EPR} serves to temporarily store the value of the resource property before being sent; v_1, v_2, v_w, v_{w_1}, v_{w_2} are variables used for the interaction among participants, and, finally, at, at_1 and at_2 are used to control the period of time in which the auction is active. In this example, we consider a period of 10 time units. Suppose $s_{0_{sys}}, s_{0_1}$ and s_{0_2} are the initial states of O_{sys}, O_1 and O_2, respectively, and all the variables are initially 0:

$$
\begin{aligned}
A_{sys} &= assign(10, at); createResource(EPR, 25, 11, A_{not});\\
&\quad while(actualTime() <= at, A_{bid})\\
A_1 &= wait(1,1); subscribe(O_1, EPR, EPR >= 0, A_{cond_1});\\
&\quad invoke(pl1, auction\_time_1, at1); \overline{reply}(pl1, auction\_time_1, at1);\\
&\quad while(actualTime() <= at1, A_{bid_1}); receive(pl3, bid\_finish_1, vw_1, empty)\\
A_2 &= wait(1,1); subscribe(O_2, EPR, EPR >= 0, A_{cond_2});\\
&\quad invoke(pl2, auction\_time_2, at2); \overline{reply}(pl2, auction\_time_2, at2);\\
&\quad while(actualTime() <= at2, A_{bid_2}); receive(pl4, bid\_finish_2, vw_2, empty)\\
A_{not} &= ((invoke(pl_3, bid\_finish_1, v_w) || invoke(pl_4, bid\_finish_2, v_w))\\
A_{bid} &= getprop(EPR, v_{EPR}); pick(\\
&\quad (pl1, auction\_time1, t, reply(pl1, auction\_time1, at)),\\
&\quad (pl2, auction\_time2, t, reply(pl2, auction\_time2, at)),\\
&\quad (pl_1, cmp, v_1, while(v_1 > v_{EPR}, assign(v1, v_{EPR});\\
&\qquad setProp(EPR, v_{EPR}); assign(1, vw))),\\
&\quad (pl_2, cmp, v_2, while(v_2 > v_{EPR}, assign(v2, v_{EPR});\\
&\qquad setProp(EPR, v_{EPR}); assign(2, vw))), empty, 1)\\
A_{cond_1} &= getProp(EPR, v_{EPR}); invoke(pl_1, bid\_up_1, v_{EPR})\\
A_{cond_2} &= getProp(EPR, v_{EPR}); invoke(pl_2, bid\_up_2, v_{EPR})\\
A_{bid_1} &= receive(pl_1, bid\_up_1, v_1); assign(v_1 + random(), v_1);\\
&\quad invoke(pl_1, cmp, v_1); subscribe(O_1, EPR, EPR > v_1, A_{cond_1}); wait(1,1)\\
A_{bid_2} &= receive(pl_2, bid\_up_2, v_2); assign(v_2 + random(), v_2);\\
&\quad invoke(pl_2, cmp, v_2); subscribe(O_2, EPR, EPR > v_2, A_{cond_2}); wait(1,1)
\end{aligned}
$$

Regarding to the operations $auction\_time1$ and $auction\_time2$ inform buyers about the period of time in which the auction is active via variables at, $at1$ and $at2$, which are used in the while structures to control this period. The operations bid_up_1 and bid_up_2 are used to increase the current bid by adding a random amount to the corresponding variable v_i. The operation cmp is an auction system operation that receives as parameter a variable of the buyers, v_i. If the value of this variable is greater than the current value of v_{EPR}, then v_{EPR} is modified with this new value, that is, the new bid exceeds the current bid. After that, by means of the activity $setProp(EPR, v_{EPR})$, we can update the value of the resource property with the new bid. Finally, the operations bid_finish_1, bid_finish_2 update the value of v_w to inform the buyers who is the winner once the auction has expired.

In Fig. 1, we depict a simplified version of the PTCPN for the online auction system. The complete model can be accessed at the following web address: `http://www.dsi.uclm.es/retics/PetriNets2012/`. Here, we have constructed a hierarchical net relying on the notions of substitution transitions, sockets and ports offered by CPNTools.

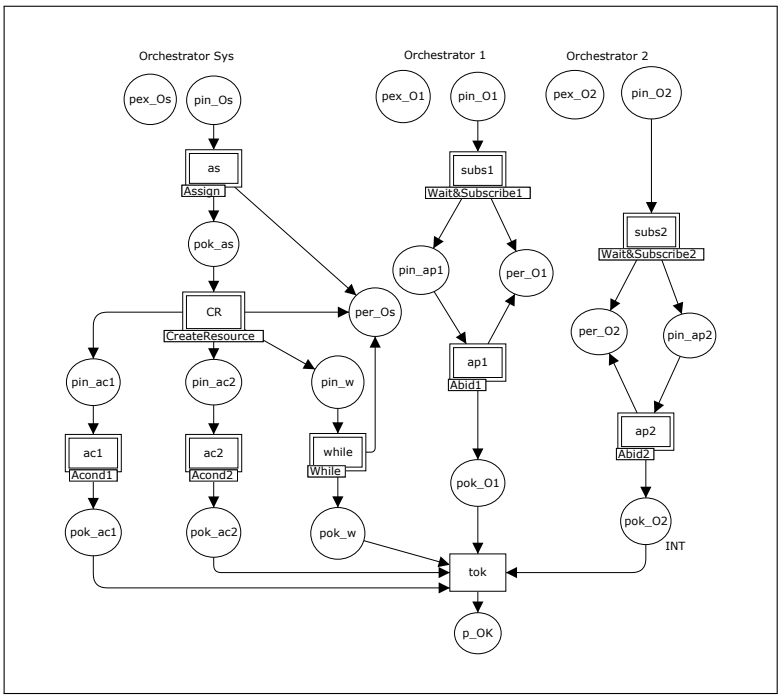

Fig. 1: A simplified PTCPN for the online auction system.