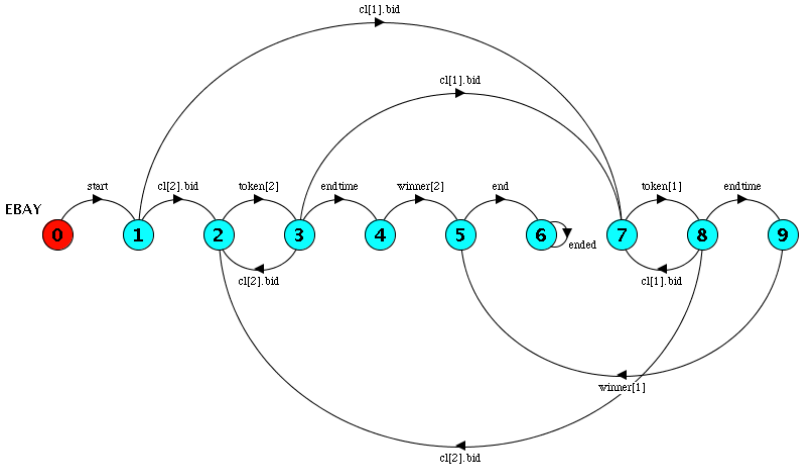
homework 3
Roberto E. Vargas
Diego Garcia-Olano
Rajagopal Hariharan

```
//1.22
const False=0
const True=1
range Bool=False..True
range P=0..1    // P runways
range V=1..3 // V planes

FLIGHT = (request_runway->WAIT),
WAIT = (free[i:P]->take.off[i]->FLIGHT).

//Control[Path0][Path1] , Path0 is occupied, Path1 is occuppied
//1
CONTROL = CONTROL[False][False],
CONTROL[p0:Bool][p1:Bool] = ( when(p0 == False) free[0] -> CONTROL[True][p1]
                            |when(p0 == True ) take.off[0] -> CONTROL[False][p1]
                            |when(p1 == False) free[1] -> CONTROL[p0][True]
                            |when(p1 == True ) take.off[1] -> CONTROL[p0][False]
                            ).

//2
property SAFE = SAFE[0][0],
SAFE[i:0..3][j:0..3] = ( when(i == 0 ) v[x:V].free[0] -> SAFE[x][j]
                       | when(i != 0 ) v[i].take.off[0] -> SAFE[0][j]
                       | when(j == 0 ) v[y:V].free[1] -> SAFE[i][y]
                       | when(j != 0 ) v[j].take.off[1] -> SAFE[i][0] ).


||AIRPORT = (v[V]:FLIGHT || v[V]::CONTROL || SAFE).

//3
||AIRPORT_STRESS = AIRPORT << {v[n:V].request_runway}.

//The resulting stressed system, AIRPORT_STRESS contains no progress violations
//because all flights have equal priority in the manner in which they are assigned a free runway.
// With this type of stress, all flights will do the request_runway action before a take.off action is performed.



//1.23  (1,2,3,4)
const N=2

CLIENT = (bid -> CLIENT).

ENDAUCTION = (end -> ENDED),
ENDED = (ended -> ENDED).

AUCTION = (start->cl[j:1..N].bid->token[j]->AUCT),
AUCT = (endtime->end->STOP
      |cl[j:1..N].bid->token[j]->AUCT
      ).

TOKEN = (token[j:1..N]->TOKEN[j]),
TOKEN[j:1..N] = (token[i:1..N]->TOKEN[i]
              |endtime->winner[j]->end->STOP
             ).

||EBAY = (AUCTION || TOKEN || cl[1..N]:CLIENT || ENDAUCTION || UNIQUE || WINNER).

//1Draw the EBAY LTS
```




```
//2 There exists a progress property of EBAY on State 6 with the action "ended".
progress ENDP = {ended}

// 3: write a safety property that shows the winner is unique
property UNIQUE = (winner[i:1..N]->STOP).

//4: write a safety property that shows the privilege (token) always
//   belong to the client who made the last bid.

property WINNER = (cl[i:1..N].bid ->LEADER[i]),
LEADER[i:1..N] = (token[i] -> WINNER[i]),
WINNER[i:1..N] = (winner[i]->STOP | cl[j:1..N].bid ->LEADER[j]).
```