

```

1  ┌────────────────────────── MODULE XJupiter ───────────────────────────┐
    Specification of the Jupiter protocol described in CSCW'2014 by Xu, Sun, and Li.
5  EXTENDS GraphStateSpace
6  ┌──────────────────────────────────────────────────────────────────────────┐
7  VARIABLES
8      c2ss,    c2ss[c]: the 2D state space (2ss, for short) at client c ∈ Client
9      s2ss     s2ss[c]: the 2D state space maintained by the Server for client c ∈ Client
11  vars ≜ ⟨intVars, ctxVars, c2ss, s2ss⟩
12  ┌──────────────────────────────────────────────────────────────────────────┐
13  TypeOK ≜
14      ∧ TypeOKInt
15      ∧ TypeOKCtx
16      ∧ ∀ c ∈ Client : IsSS(c2ss[c]) ∧ IsSS(s2ss[c])
17  ┌──────────────────────────────────────────────────────────────────────────┐
18  Init ≜
19      ∧ InitInt
20      ∧ InitCtx
21      ∧ c2ss = [c ∈ Client ↦ EmptySS]
22      ∧ s2ss = [c ∈ Client ↦ EmptySS]
23  ┌──────────────────────────────────────────────────────────────────────────┐
24  NextEdge(r, u, ss) ≜ Return the unique outgoing edge from u in 2D state space ss
25      CHOOSE e ∈ ss.edge : e.from = u before a transformation at u (r is not used).
27  ClientPerform(c, cop) ≜
28      LET xform ≜ xForm(NextEdge, c, cop, c2ss[c]) xform: [xcop, xss, lss]
29      IN  ∧ c2ss' = [c2ss EXCEPT ![c] = @ ⊕ xform.xss]
30          ∧ SetNewAop(c, xform.xcop.op)
32  ServerPerform(cop) ≜
33      LET c ≜ ClientOf(cop)
34      xform ≜ xForm(NextEdge, Server, cop, s2ss[c]) xform: [xcop, xss, lss]
35      xcop ≜ xform.xcop
36      IN  ∧ s2ss' = [cl ∈ Client ↦ IF cl = c
37          THEN s2ss[cl] ⊕ xform.xss
38          ELSE s2ss[cl] ⊕ xform.lss]
39          ∧ SetNewAop(Server, xcop.op)
40          ∧ Comm!SSendSame(c, xcop) broadcast the transformed xcop
41  ┌──────────────────────────────────────────────────────────────────────────┐
42  DoOp(c, op) ≜
43      LET cop ≜ [op ↦ op, oid ↦ [c ↦ c, seq ↦ cseq[c], ctx ↦ ds[c]]
44      IN  ∧ ClientPerform(c, cop)
45          ∧ Comm!CSend(cop)
47  Do(c) ≜
48      ∧ DoInt(DoOp, c)

```

```

49       $\wedge DoCtx(c)$ 
50       $\wedge \text{UNCHANGED } s2ss$ 

52   $Rev(c) \triangleq$ 
53       $\wedge RevInt(ClientPerform, c)$ 
54       $\wedge RevCtx(c)$ 
55       $\wedge \text{UNCHANGED } s2ss$ 

57   $SRev \triangleq$ 
58       $\wedge SRevInt(ServerPerform)$ 
59       $\wedge SRevCtx$ 
60       $\wedge \text{UNCHANGED } c2ss$ 
61  |-----|
62   $Next \triangleq$ 
63       $\vee \exists c \in Client : Do(c) \vee Rev(c)$ 
64       $\vee SRev$ 

66   $Fairness \triangleq$ 
67       $WF_{vars}(SRev \vee \exists c \in Client : Rev(c))$ 

69   $Spec \triangleq Init \wedge \Box[Next]_{vars} \wedge Fairness$ 
70  |-----|
71   $CSSync \triangleq$  Each client  $c \in Client$  is synchronized with the Server.
72       $\forall c \in Client : (ds[c] = ds[Server]) \Rightarrow c2ss[c] = s2ss[c]$ 

74  THEOREM  $Spec \Rightarrow \Box CSSync$ 
75  |-----|

  \ * Modification History
  \ * Last modified Sat Jan 12 15:57:05 CST 2019 by hengxin
  \ * Created Tue Oct 09 16:33:18 CST 2018 by hengxin

```