```
MODULE XJupiter
```

Specification of the *Jupiter* protocol described in *CSCW'*2014 by *Yi Xu*, *Chengzheng* Sun, and *Mo Li*. We call it *XJupiter*, with 'X' for "*Xu*".

```
EXTENDS StateSpace

VARIABLES
    c2ss,    c2ss[c]: the 2D state space (2ss, for short) at client c ∈ Client
    s2ss     s2ss[c]: the 2D state space maintained by the Server for client c ∈ Client

vars ≜ ⟨intVars, ctxVars, c2ss, s2ss⟩

TypeOK ≜
    ∧ TypeOKInt
    ∧ TypeOKCtx
    ∧ Comm(Cop)!TypeOK
    ∧ ∀ c ∈ Client : IsSS(c2ss[c]) ∧ IsSS(s2ss[c])

Init ≜
    ∧ InitInt
    ∧ InitCtx
    ∧ Comm(Cop)!Init
    ∧ c2ss = [c ∈ Client ↦ EmptySS]
    ∧ s2ss = [c ∈ Client ↦ EmptySS]
```

xForm: iteratively transform cop with a path through the 2D state space *ss* at some client.

```
xForm(cop, ss, cur) ≜
  LET u ≜ Locate(cop, ss)
      v ≜ u ∪ {cop.oid}
      RECURSIVE xFormHelper(_, _, _, _)
      xFormHelper(uh, vh, coph, xss) ≜   xss: eXtra ss created during transformation
        IF uh = cur THEN [xss ↦ xss, xcop ↦ coph]
        ELSE LET e ≜ CHOOSE e ∈ ss.edge : e.from = uh ∧ ClientOf(e.cop) ≠ ClientOf(cop)
                 copprime ≜ e.cop
                 uprime ≜ e.to
                 vprime ≜ vh ∪ {copprime.oid}
                 coph2copprime ≜ COT(coph, copprime)
                 copprime2coph ≜ COT(copprime, coph)
             IN  xFormHelper(uprime, vprime, coph2copprime,
                   xss ⊕ [node ↦ {vprime},
                          edge ↦ {[from ↦ vh, to ↦ vprime, cop ↦ copprime2coph],
                                  [from ↦ uprime, to ↦ vprime, cop ↦ coph2copprime]}])
  IN  xFormHelper(u, v, cop, [node ↦ {v}, edge ↦ {[from ↦ u, to ↦ v, cop ↦ cop]}])
```

Client $c \in$ *Client* perform operation cop.

```
53  ClientPerform(cop, c) ≜
54      LET xform ≜ xForm(cop, c2ss[c], ds[c])   xform: [xss, xcop]
55      IN   ∧ c2ss' = [c2ss EXCEPT ![c] = @ ⊕ xform.xss]
56           ∧ state' = [state EXCEPT ![c] = Apply(xform.xcop.op, @)]
    Client c ∈ Client generates an operation op.
60  DoOp(c, op)  ≜
61        LET cop ≜ [op ↦ op, oid ↦ [c ↦ c, seq ↦ cseq'[c]], ctx ↦ ds[c]]
62            IN   ∧ ClientPerform(cop, c)
63                 ∧ Comm(Cop)!CSend(cop)

65  DoIns(c) ≜
66      ∃ ins ∈ {op ∈ Ins : op.pos ∈ 1 .. (Len(state[c]) + 1) ∧ op.ch ∈ chins ∧ op.pr = Priority[c]} :
67         ∧ DoOp(c, ins)
68         ∧ chins' = chins \ {ins.ch}

70  DoDel(c) ≜
71      ∃ del ∈ {op ∈ Del : op.pos ∈ 1 .. Len(state[c])} :
72         ∧ DoOp(c, del)
73         ∧ UNCHANGED chins

75  Do(c) ≜
76        ∧ DoCtx(c)
77        ∧ ∨ DoIns(c)
78          ∨ DoDel(c)
79        ∧ UNCHANGED s2ss
    Client c ∈ Client receives a message from the Server.
83  Rev(c) ≜
84        ∧ Comm(Cop)!CRev(c)
85        ∧ LET cop ≜ Head(cincoming[c])
86            IN   ClientPerform(cop, c)
87        ∧ RevCtx(c)
88        ∧ UNCHANGED ⟨chins, s2ss⟩
89  ├──────────────────────────────────────────────────────────────────────┤
    The Server performs operation cop.
93  ServerPerform(cop) ≜
94      LET c ≜ ClientOf(cop)
95       scur ≜ ds[Server]
96      xform ≜ xForm(cop, s2ss[c], scur)   xform: [xss, xcop]
97       xcop ≜ xform.xcop
98       xcur ≜ scur ∪ {cop.oid}
99      IN   ∧ s2ss' = [cl ∈ Client ↦
100                       IF cl = c
101                        THEN s2ss[cl] ⊕ xform.xss
102                        ELSE s2ss[cl] ⊕ [node ↦ {xcur},
103                           edge ↦ {[from ↦ scur, to ↦ xcur, cop ↦ xcop]}]
```

```
                    ]
        ∧ state' = [state EXCEPT ![Server] = Apply(xcop.op, @)]
        ∧ Comm(Cop)!SSendSame(c, xcop)
```

The *Server* receives a message.

```
SRev ≜
    ∧ Comm(Cop)!SRev
    ∧ LET cop ≜ Head(sincoming)
      IN  ServerPerform(cop)
    ∧ SRevCtx
    ∧ UNCHANGED ⟨chins, c2ss⟩
```

```
Next ≜
    ∨ ∃ c ∈ Client : Do(c) ∨ Rev(c)
    ∨ SRev

Fairness ≜   There is no requirement that the clients ever generate operations.
    WF_vars(SRev ∨ ∃ c ∈ Client : Rev(c))

Spec ≜ Init ∧ □[Next]_vars   ∧ Fairness
```

```
CSSync ≜   Each client c ∈ Client is synchonized with the Server.
    ∀ c ∈ Client : (ds[c] = ds[Server]) ⇒ c2ss[c] = s2ss[c]

THEOREM Spec ⇒ □CSSync
```

```
\* Modification History
\* Last modified Sat Dec 29 18:50:10 CST 2018 by hengxin
\* Created Tue Oct 09 16:33:18 CST 2018 by hengxin
```