```
1  ---------- MODULE AbsJupiter ----------
   Abstract Jupiter, inspired by the COT algorithm proposed by Sun and Sun; see Sun@TPDS'2009.
6  EXTENDS JupiterSerial, SetStateSpace
7  ----------
8  VARIABLES
9      copss    copss[r]: the state space (i.e., a set) of Cop maintained at replia r ∈ Replica
11 vars ≜ ⟨intVars, ctxVars, serialVars, copss⟩
12 ----------
13 TypeOK ≜
14     ∧  TypeOKInt
15     ∧  TypeOKCtx
16     ∧  TypeOKSerial
17     ∧  copss ∈ [Replica → SUBSET Cop]
18 ----------
19 Init ≜
20     ∧ InitInt
21     ∧ InitCtx
22     ∧ InitSerial
23     ∧ copss = [r ∈ Replica ↦ {}]
24 ----------
25 NextCop(r, cop, ss, ctx) ≜  Return the next fcop ∈ Cop against which cop is to be transformed.
26     LET foid ≜ CHOOSE oid ∈ ctx :  the first oid in ctx according to serial[r]
27                    ∀ id ∈ ctx \ {oid} : tb(oid, id, serial[r])
28     IN  CHOOSE fcop ∈ ss :  THEOREM : Existence of fcop
29             fcop.oid = foid ∧ fcop.ctx = cop.ctx
31 Perform(r, cop) ≜
32     LET xform ≜ xForm(NextCop, r, cop, copss[r])   [xcop, xss]
33     IN  ∧ copss' = [copss EXCEPT ![r] = xform.xss]
34         ∧ SetNewAop(r, xform.xcop.op)
36 ClientPerform(c, cop) ≜ Perform(c, cop)
38 ServerPerform(cop) ≜
39     ∧ Perform(Server, cop)
40     ∧ Comm!SSendSame(ClientOf(cop), cop)
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     ∧ DoCtx(c)
```

```
50          ∧ DoSerial(c)

52  Rev(c) ≜
53          ∧ RevInt(ClientPerform, c)
54          ∧ RevCtx(c)
55          ∧ RevSerial(c)

57  SRev ≜
58          ∧ SRevInt(ServerPerform)
59          ∧ SRevCtx
60          ∧ SRevSerial
61  ----------------------------------------------------------------
62  Next ≜
63          ∨ ∃ c ∈ Client : Do(c) ∨ Rev(c)
64          ∨ SRev

66  Fairness ≜
67          WF_vars(SRev ∨ ∃ c ∈ Client : Rev(c))

69  Spec ≜ Init ∧ □[Next]_vars  ∧ Fairness
70  ----------------------------------------------------------------
71  QC ≜   Quiescent Consistency
72          Comm!EmptyChannel ⇒ Cardinality(Range(state)) = 1
73  THEOREM Spec ⇒ □QC

75  SEC ≜   Strong Eventual Consistency
76          ∀ r1, r2 ∈ Replica :
77              ds[r1] = ds[r2] ⇒ state[r1] = state[r2]
78  THEOREM Spec ⇒ □SEC

80  Compactness ≜   Compactness of state space
81          Comm!EmptyChannel ⇒ Cardinality(Range(copss)) = 1
82  THEOREM Spec ⇒ □Compactness
83  ================================================================

    \* Modification History
    \* Last modified Tue Feb 05 11:05:02 CST 2019 by hengxin
    \* Created Wed Dec 05 19:55:52 CST 2018 by hengxin
```