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

```
49        ∧ DoSerial(c)

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

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

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

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

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

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

83  ClientConstraint ≜  Each client generates at most 2 operations.
84     ∀ c ∈ Client : cseq[c] ≤ 3
85 └──────────────────────────────────────────────────────────────────────┘
    \* Modification History
    \* Last modified Tue Jan 29 09:36:46 CST 2019 by hengxin
    \* Created Wed Dec 05 19:55:52 CST 2018 by hengxin
```