```
1  ---------- MODULE AbsJupiter ----------
   Abstract Jupiter, inspired by the COT algorithm proposed by Sun and Sun; see TPDS'2009.
5  EXTENDS JupiterSerial
6  ----------
7  VARIABLES
8      copss    copss[r]: the state space (i.e., a set) of Cops maintained at replia r ∈ Replica
10 vars ≜ ⟨intVars, ctxVars, serialVars, copss⟩
11 ----------
12 TypeOK ≜
13     ∧  TypeOKInt
14     ∧  TypeOKCtx
15     ∧  TypeOKSerial
16     ∧  Comm(Cop)!TypeOK
17     ∧  copss ∈ [Replica → SUBSET Cop]
18 ----------
19 Init ≜
20     ∧ InitInt
21     ∧ InitCtx
22     ∧ InitSerial
23     ∧ Comm(Cop)!Init
24     ∧ copss = [r ∈ Replica ↦ {}]
25 ----------
26 RECURSIVE xForm(_, _)
27 xForm(cop, r) ≜
28     LET ctxDiff ≜ ds[r] \ cop.ctx   THEOREM : cop.ctx ⊆ ds[r]
29         RECURSIVE xFormHelper(_, _, _)
30          xFormHelper(coph, ctxDiffh, copssr) ≜   copssr: state space generated during transformation
31              IF ctxDiffh = {} THEN [xcop ↦ coph, xcopss ↦ copssr]
32               ELSE LET foph ≜ CHOOSE op ∈ ctxDiffh :   the first op in serial
33                                    ∀ opprime ∈ ctxDiffh \ {op} : tb(op, opprime, serial[r])
34                         fcophDict ≜ {op ∈ copssr : op.oid = foph ∧ op.ctx = coph.ctx}
35                         fcoph ≜ CHOOSE op ∈ fcophDict : TRUE   THEOREM : Cardinality(fophDict) = 1
36                         xcoph ≜ COT(coph, fcoph)
37                          xfcoph ≜ COT(fcoph, coph)
38                    IN   xFormHelper(xcoph, ctxDiffh \ {foph}, copssr ∪ {xcoph, xfcoph})
39      IN   xFormHelper(cop, ctxDiff, copss[r])
41 Perform(cop, r) ≜
42     LET xform ≜ xForm(cop, r)   [xcop, xcopss]
43      IN   ∧ state' = [state EXCEPT ![r] = Apply(xform.xcop.op, @)]
44           ∧ copss' = [copss EXCEPT ![r] = xform.xcopss ∪ {cop}]
45 ----------
   Client c ∈ Client issues an operation op.
```

```
49  DoOp(c, op) ≜   op: the raw operation generated by the client c ∈ Client
50        ∧ LET cop ≜ [op ↦ op, oid ↦ [c ↦ c, seq ↦ cseq'[c]], ctx ↦ ds[c]]
51          IN   ∧ Perform(cop, c)
52               ∧ Comm(Cop)!CSend(cop)

54  DoIns(c) ≜
55     ∃ ins ∈ {op ∈ Ins : op.pos ∈ 1 .. (Len(state[c]) + 1) ∧ op.ch ∈ chins ∧ op.pr = Priority[c]} :
56        ∧ DoOp(c, ins)
57        ∧ chins' = chins \ {ins.ch}   We assume that all inserted elements are unique.

59  DoDel(c) ≜
60     ∃ del ∈ {op ∈ Del : op.pos ∈ 1 .. Len(state[c])} :
61        ∧ DoOp(c, del)
62        ∧ UNCHANGED chins

64  Do(c) ≜
65        ∧ DoCtx(c)
66        ∧ DoSerial(c)
67        ∧ ∨ DoIns(c)
68          ∨ DoDel(c)
69  ├──────────────────────────────────────────────────────────────────────────┤
70  Rev(c) ≜
71        ∧ Comm(Cop)!CRev(c)
72        ∧ Perform(Head(cincoming[c]), c)
73        ∧ RevSerial(c)
74        ∧ RevCtx(c)
75        ∧ UNCHANGED chins
76  ├──────────────────────────────────────────────────────────────────────────┤
77  SRev ≜
78     ∧ Comm(Cop)!SRev
79     ∧ LET cop ≜ Head(sincoming)
80         IN   ∧ Perform(cop, Server)
81              ∧ Comm(Cop)!SSendSame(cop.oid.c, cop)
82     ∧ SRevSerial
83     ∧ SRevCtx
84     ∧ UNCHANGED chins
85  ├──────────────────────────────────────────────────────────────────────────┤
86  Next ≜
87     ∨ ∃ c ∈ Client : Do(c) ∨ Rev(c)
88     ∨ SRev

90  Fairness ≜
91     WF_vars(SRev ∨ ∃ c ∈ Client : Rev(c))

93  Spec ≜ Init ∧ □[Next]_vars   ∧ Fairness
94  ├──────────────────────────────────────────────────────────────────────────┤
```

$$Compactness \triangleq$$
$$\quad Comm(Cop)!EmptyChannel \Rightarrow Cardinality(Range(copss)) = 1$$

THEOREM $Spec \Rightarrow Compactness$

* Modification History
* Last modified *Mon Dec* 31 10:50:36 *CST* 2018 by *hengxin*
* Created *Wed Dec* 05 19:55:52 *CST* 2018 by *hengxin*