```
MODULE AbsJupiter
Abstract Jupiter, inspired by the COT algorithm proposed by Sun and Sun; see TPDS'2009.
EXTENDS JupiterSerial
VARIABLES
    copss    copss[r]: the state space (i.e., a set) of Cops maintained at replia r ∈ Replica
vars ≜ ⟨intVars, ctxVars, serialVars, copss⟩
TypeOK ≜
    ∧  TypeOKInt
    ∧  TypeOKCtx
    ∧  TypeOKSerial
    ∧  Comm(Cop)!TypeOK
    ∧  copss ∈ [Replica → SUBSET Cop]
Init ≜
    ∧ InitInt
    ∧ InitCtx
    ∧ InitSerial
    ∧ Comm(Cop)!Init
    ∧ copss = [r ∈ Replica ↦ {}]
RECURSIVE xForm(_, _)
xForm(cop, r) ≜
  LET ctxDiff ≜ ds[r] \ cop.ctx   THEOREM : cop.ctx ⊆ ds[r]
      RECURSIVE xFormHelper(_, _, _)
       xFormHelper(coph, ctxDiffh, copssr) ≜   copssr: state space generated during transformation
         IF ctxDiffh = {} THEN [xcop ↦ coph, xcopss ↦ copssr]
          ELSE LET foph ≜ CHOOSE op ∈ ctxDiffh :   the first op in serial
                                ∀ opprime ∈ ctxDiffh \ {op} : tb(op, opprime, serial[r])
                   fcophDict ≜ {op ∈ copssr : op.oid = foph ∧ op.ctx = coph.ctx}
                   fcoph ≜ CHOOSE op ∈ fcophDict : TRUE   THEOREM : Cardinality(fophDict) = 1
                   xcoph ≜ COT(coph, fcoph)
                    xfcoph ≜ COT(fcoph, coph)
               IN  xFormHelper(xcoph, ctxDiffh \ {foph}, copssr ∪ {xcoph, xfcoph})
   IN  xFormHelper(cop, ctxDiff, copss[r])
Perform(cop, r) ≜
  LET xform ≜ xForm(cop, r)   [xcop, xcopss]
   IN  ∧ state' = [state EXCEPT ![r] = Apply(xform.xcop.op, @)]
       ∧ copss' = [copss EXCEPT ![r] = xform.xcopss ∪ {cop}]
DoOp(c, op) ≜   Client c ∈ Client processes a locally generated operation op.
    LET cop ≜ [op ↦ op, oid ↦ [c ↦ c, seq ↦ cseq'[c]], ctx ↦ ds[c]]
```

```
48          IN    ∧ Perform(cop, c)
49                ∧ Comm(Cop)!CSend(cop)

51  Do(c) ≜
52        ∧ DoCtx(c)
53        ∧ DoSerial(c)
54        ∧ DoInt(DoOp, c)

56  Rev(c) ≜
57        ∧ Comm(Cop)!CRev(c)
58        ∧ Perform(Head(cincoming[c]), c)
59        ∧ RevSerial(c)
60        ∧ RevCtx(c)
61        ∧ RevInt(c)

63  SRev ≜
64      ∧ Comm(Cop)!SRev
65      ∧ LET cop ≜ Head(sincoming)
66          IN    ∧ Perform(cop, Server)
67                ∧ Comm(Cop)!SSendSame(cop.oid.c, cop)
68      ∧ SRevSerial
69      ∧ SRevCtx
70      ∧ SRevInt
71  ├──────────────────────────────────────────────────────────────────────┤
72  Next ≜
73      ∨ ∃ c ∈ Client : Do(c) ∨ Rev(c)
74      ∨ SRev

76  Fairness ≜
77      WF_vars(SRev ∨ ∃ c ∈ Client : Rev(c))

79  Spec ≜ Init ∧ □[Next]_vars  ∧ Fairness
80  ├──────────────────────────────────────────────────────────────────────┤
81  Compactness ≜
82      Comm(Cop)!EmptyChannel ⇒ Cardinality(Range(copss)) = 1

84  THEOREM Spec ⇒ Compactness
85  └──────────────────────────────────────────────────────────────────────┘
    \* Modification History
    \* Last modified Mon Dec 31 20:27:49 CST 2018 by hengxin
    \* Created Wed Dec 05 19:55:52 CST 2018 by hengxin
```