```
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
    ∧  copss ∈ [Replica → SUBSET Cop]
Init ≜
    ∧ InitInt
    ∧ InitCtx
    ∧ InitSerial
    ∧ copss = [r ∈ Replica ↦ {}]
RECURSIVE xForm(_, _)
xForm(r, cop) ≜
    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(r, cop) ≜
    LET xform ≜ xForm(r, cop)   [xcop, xcopss]
    IN  ∧ copss' = [copss EXCEPT ![r] = xform.xcopss ∪ {cop}]
        ∧ SetNewAop(r, xform.xcop.op)
ServerPerform(cop) ≜
    ∧ Perform(Server, cop)
    ∧ Comm!SSendSame(ClientOf(cop), cop)
```

```
48  DoOp(c, op)  ≜
49        LET cop ≜ [op ↦ op, oid ↦ [c ↦ c, seq ↦ cseq[c]], ctx ↦ ds[c]]
50         IN   ∧ Perform(c, cop)
51              ∧ Comm!CSend(cop)

53  Do(c) ≜
54        ∧ DoInt(DoOp, c)
55        ∧ DoCtx(c)
56        ∧ DoSerial(c)

58  Rev(c) ≜
59        ∧ RevInt(Perform, c)
60        ∧ RevCtx(c)
61        ∧ RevSerial(c)

63  SRev ≜
64      ∧ SRevInt(ServerPerform)
65      ∧ SRevCtx
66      ∧ SRevSerial
67  ├──────────────────────────────────────────────────────────────────────┤
68  Next ≜
69      ∨ ∃ c ∈ Client : Do(c) ∨ Rev(c)
70      ∨ SRev

72  Fairness ≜
73     WF_vars(SRev ∨ ∃ c ∈ Client : Rev(c))

75  Spec ≜ Init ∧ □[Next]_vars  ∧ Fairness
76  ├──────────────────────────────────────────────────────────────────────┤
77  Compactness ≜
78      Comm!EmptyChannel ⇒ Cardinality(Range(copss)) = 1

80  THEOREM Spec ⇒ Compactness
81  └──────────────────────────────────────────────────────────────────────┘
    \* Modification History
    \* Last modified Wed Jan 02 20:47:48 CST 2019 by hengxin
    \* Created Wed Dec 05 19:55:52 CST 2018 by hengxin
```