```
┌──────────────────────────── MODULE AbsJupiter ────────────────────────────┐
 Abstract Jupiter, inspired by the COT algorithm proposed by Sun and Sun; see TPDS′2009.
EXTENDS JupiterSerial, SetStateSpace
├────────────────────────────────────────────────────────────────────────────┤
VARIABLES
    copss    copss[r]: the state space (i.e., a set) of Cop 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 ↦ {}]
├────────────────────────────────────────────────────────────────────────────┤
NextCop(r, cop, ss, ctx) ≜   Return the next fcop ∈ Cop against which cop is to be transformed.
    LET foid ≜ CHOOSE oid ∈ ctx :  the first oid in ctx according to serial[r]
                     ∀ id ∈ ctx \ {oid} : tb(oid, id, serial[r])
    IN  CHOOSE fcop ∈ ss :  THEOREM : Existence of fcop
              fcop.oid = foid ∧ fcop.ctx = cop.ctx

Perform(r, cop) ≜
    LET xform ≜ xForm(NextCop, r, cop, copss[r])   [xcop, xss]
    IN  ∧ copss′ = [copss EXCEPT ![r] = xform.xss]
        ∧ SetNewAop(r, xform.xcop.op)

ClientPerform(c, cop) ≜ Perform(c, cop)

ServerPerform(cop) ≜
    ∧ Perform(Server, cop)
    ∧ Comm!SSendSame(ClientOf(cop), cop)
├────────────────────────────────────────────────────────────────────────────┤
DoOp(c, op) ≜
     LET cop ≜ [op ↦ op, oid ↦ [c ↦ c, seq ↦ cseq[c]], ctx ↦ ds[c]]
     IN  ∧ ClientPerform(c, cop)
         ∧ Comm!CSend(cop)

Do(c) ≜
     ∧ DoInt(DoOp, c)
     ∧ DoCtx(c)
```

```
        $\land DoSerial(c)$

$Rev(c) \triangleq$
        $\land RevInt(ClientPerform,\ c)$
        $\land RevCtx(c)$
        $\land RevSerial(c)$

$SRev \triangleq$
      $\land SRevInt(ServerPerform)$
      $\land SRevCtx$
      $\land SRevSerial$
--------------------------------------------------------------------------------
$Next \triangleq$
      $\lor \exists\, c \in Client : Do(c) \lor Rev(c)$
      $\lor SRev$

$Fairness \triangleq$
      $\mathrm{WF}_{vars}(SRev \lor \exists\, c \in Client : Rev(c))$

$Spec \triangleq Init \land \Box[Next]_{vars}$   $\land Fairness$
--------------------------------------------------------------------------------
$QC \triangleq$   Quiescent Consistency
        $Comm!EmptyChannel \Rightarrow Cardinality(Range(state)) = 1$
THEOREM $Spec \Rightarrow \Box QC$

$SEC \triangleq$   Strong Eventual Consistency
        $\forall\, r1,\ r2 \in Replica :$
            $ds[r1] = ds[r2] \Rightarrow state[r1] = state[r2]$
THEOREM $Spec \Rightarrow \Box SEC$

$Compactness \triangleq$   Compactness of state space
      $Comm!EmptyChannel \Rightarrow Cardinality(Range(copss)) = 1$
THEOREM $Spec \Rightarrow \Box Compactness$
================================================================================
\* Modification History
\* Last modified Tue Jan 29 10:11:54 CST 2019 by hengxin
\* Created Wed Dec 05 19:55:52 CST 2018 by hengxin
```