```
┌──────────────── MODULE XJupiterImplCJupiter ────────────────┐
In this module, we show that XJupiter implements CJupiter.  To this end, we first extends
XJupiter by replacing its Cop with that used in CJupiter.
EXTENDS XJupiterExtended

VARIABLES
    cincomingCJ,    cincoming for CJupiter which contains original operations
                    instead of transformed ones in XJupiter
    op2ss,          a function from an operation (represented by its Oid)
                    to the part of 2D state space produced while the operation is transformed
    c2ssX           c2ssX[c]: redundant (eXtra) 2D state space maintained for client c ∈ Client

varsImpl ≜ ⟨vars, cincomingCJ, op2ss, c2ssX⟩

TypeOKImpl ≜
    ∧ TypeOK
    ∧ cincomingCJ ∈ [Client → Seq(Cop)]
    ∧ ∀ oid ∈ DOMAIN op2ss : oid ∈ Oid ∧ IsSS(op2ss[oid])
    ∧ ∀ c ∈ Client : IsSS(c2ssX[c])

The Init predicate.
InitImpl ≜
    ∧ Init
    ∧ cincomingCJ = [c ∈ Client ↦ ⟨⟩]
    ∧ op2ss  = ⟨⟩
    ∧ c2ssX = [c ∈ Client ↦ [node ↦ {{}}, edge ↦ {}]]

Client c ∈ Client generates an operation and performs it locally.
DoImpl(c) ≜
    ∧ Do(c)
    ∧ UNCHANGED ⟨cincomingCJ, op2ss, c2ssX⟩

ss1 ⊕ ss2 ≜
   [ss1 EXCEPT !.node = @ ∪ ss2.node,
               !.edge  = @ ∪ ss2.edge]
Client c ∈ Client receives a message and processes it.
RevImpl(c) ≜
    ∧   Rev(c)
    ∧   cincomingCJ[c] ≠ ⟨⟩   there are (original) operations to handle with
    ∧   cincomingCJ′ = [cincomingCJ EXCEPT ![c] = Tail(@)]  also consume a message
    ∧   LET cop ≜ Head(cincoming[c])
          IN   c2ssX′ = [c2ssX EXCEPT ![c] = @ ⊕ op2ss[cop.oid]]
    ∧   UNCHANGED ⟨op2ss⟩

Also broadcast the original operation to clients (using the cincomingCJ channels)
```

```
SRevImpl ≜
    ∧ SRev
    ∧ LET cop ≜ [Head(sincoming) EXCEPT !.sctx = soids]
            c ≜ cop.oid.c
           ss ≜ xForm(cop, s2ss[c], scur[c], Remote)
       IN   ∧ cincomingCJ′ = [cl ∈ Client ↦
                                  IF cl = c
                                   THEN cincomingCJ[cl]
                                   ELSE Append(cincomingCJ[cl], cop)]
            ∧ op2ss′ = op2ss @@ (cop.oid :> [node ↦ Range(ss.node), edge ↦ Range(ss.edge)])
    ∧ UNCHANGED ⟨c2ssX⟩
```

The next-state relation.

```
NextImpl ≜
    ∨ ∃ c ∈ Client : DoImpl(c) ∨ RevImpl(c)
    ∨ SRevImpl
```

The specification.

$SpecImpl \triangleq InitImpl \land \Box[NextImpl]_{varsImpl} \land \mathrm{WF}_{varsImpl}(SRevImpl \lor \exists\, c \in Client : RevImpl(c))$

Ignore the lr field in edges of $2D$ state space.

```
IgnoreDir(ss) ≜
    [ss EXCEPT !.edge =
        {[field ∈ (DOMAIN e \ {"lr"}) ↦ e.field] : e ∈ @}]
      {[from ↦ e.from, to ↦ e.to, cop ↦ e.cop] : e ∈ @}]

CJ ≜ INSTANCE CJupiter
        WITH cincoming ← cincomingCJ,
              css ← [r ∈ Replica ↦
                        IF r = Server
                         THEN IgnoreDir(SetReduce(⊕, Range(s2ss), [node ↦ {{}}, edge ↦ {}]))
                         ELSE IgnoreDir(c2ss[r] ⊕ c2ssX[r])],
              cur ← [r ∈ Replica ↦
                        IF r = Server
                         It SHOULD be that Cardinality(Range(scur)) = 1
                         THEN CHOOSE n ∈ Range(scur) : TRUE
                         ELSE ccur[r]]

THEOREM SpecImpl ⇒ CJ!Spec
```

* Modification History
* Last modified *Thu Nov* 01 13:51:35 *CST* 2018 by *hengxin*
* Created *Fri Oct* 26 15:00:19 *CST* 2018 by *hengxin*