MODULE $U2PC\_MC$

EXTENDS $Apalache$, $U2PC$, TLC

```
@type: (a, b) => <a, b>;
Pair(A, B) == <A, B>

T1 == SetAsFun({Pair("T1", {"X"})})
T1_2 == SetAsFun({Pair("T1", {"X"}), Pair("T2", {"X"})})
S1 == SetAsFun({Pair("X", {"X1", "X2"})})

T3 == SetAsFun({
  Pair("T1", {"X", "Y"}),
  Pair("T2", {"Y", "Z"}),
  Pair("T3", {"Z", "X"})})
S3 == SetAsFun({
  Pair("X", {"X1", "X2"}),
  Pair("Y", {"Y1", "Y2"}),
  Pair("Z", {"Z1", "Z2"})})


CInit ==
  /\ Txns := T3
  /\ Shards := S3

TransitiveClosure(R) ==
  LET S == {r[1] : r \in R} \cup {r[2] : r \in R}
      RECURSIVE TCR(_)
      TCR(T) == IF T = {}
                THEN R
                ELSE LET r == CHOOSE s \in T : TRUE
                         RR == TCR(T \ {r})
                     IN  RR \cup {<<s, t>> \in S \X S :
                                  <<s, r>> \in RR /\ <<r, t>> \in RR}
  IN  TCR(S)

TransactionOrdering == LET
  F(acc, tid) == acc \cup (Range(Coordinator_txn_state[tid]) \X {tid})
  Base == ApaFoldSet(F, {}, TIDs)
  IN  TransitiveClosure(Base)

RecoveryCommitted(S) ==
  {t \in TIDs :
    \A r \in S :
    KeyLookup[r] \in Txns[t]
     => \/ Replicas[r].locked /\ Replicas[r].logged = t
        \/ Replicas[r].version = t
        \/ <<t, Replicas[r].version>> \in TransactionOrdering
```

```
  }

Safety_recovery ≜
  ∀ S ∈ SUBSET RIDs :
   Valid recovery
  (∀ k ∈ DOMAIN Shards : ∃ r ∈ S : r ∈ Shards[k])
  ⇒
  IF Serialisability(CommittedTIDs ∪ RecoveryCommitted(S))
   THEN TRUE
   ELSE Print([rec ↦ RecoveryCommitted(S), com ↦ CommittedTIDs], FALSE)

RecoveryAborted(S) ≜
  {t ∈ TIDs :
    ∃ r ∈ S :
     ∧ KeyLookup[r] ∈ Txns[t]
     ∧ ∨ ¬Replicas[r].locked
       ∨ Replicas[r].locked ∧ Replicas[r].logged ≠ t}

Durability ≜
  ∀ S ∈ SUBSET RIDs :
   Valid recovery
  (∀ k ∈ DOMAIN Shards : ∃ r ∈ S : r ∈ Shards[k])
  ⇒
  ∀ t ∈ TIDs :
  ∧ t ∈ CommittedTIDs ⇒ t ∈ RecoveryCommitted(S)
  ∧ t ∈ AbortedTIDs ⇒ t ∈ RecoveryAborted(S)

Invs ≜
  ∧ Safety_recovery
  ∧ Durability
```