```
──────────────── MODULE JupiterInterface ────────────────
This module declares the parameters and defines the operators that describe the interface of a
family of Jupiter specs.
EXTENDS SequenceUtils, OT
─────────────────────────────────────────────────────────
CONSTANTS
    Client,       the set of client replicas
    Server,       the (unique) server replica
    Msg,          the set of messages
    Char,         the set of characters
    InitState     the initial state of each replica

ASSUME  We assume that all inserted elements are unique.
    ∧  Range(InitState) ∩ Char = {}   due to the uniqueness requirement
─────────────────────────────────────────────────────────
VARIABLES
    aop,        op[r]: the actual operation applied at replica r ∈ Replica
    state,      state[r]: state (the list content) of replica r ∈ Replica
    cincoming,   cincoming[c]: incoming channel at the client c ∈ Client
    sincoming,   incoming channel at the Server
    chins       a set of chars allowed to insert; this is for model checking

intVars ≜ ⟨aop, state, cincoming, sincoming, chins⟩
─────────────────────────────────────────────────────────
Comm ≜ INSTANCE CSComm

Replica ≜ Client ∪ {Server}

List ≜ Seq(Char ∪ Range(InitState))                all possible lists
MaxLen ≜ Cardinality(Char) + Len(InitState)  the max length of lists in any state

ClientNum ≜ Cardinality(Client)
Priority ≜ CHOOSE f ∈ [Client → 1 . . ClientNum] : Injective(f)
─────────────────────────────────────────────────────────
The set of all operations. Note: The positions are indexed from 1.
Rd ≜ [type : {"Rd"}]
Del ≜ [type : {"Del"}, pos : 1 . . MaxLen]
Ins ≜ [type : {"Ins"}, pos : 1 . . (MaxLen + 1), ch : Char, pr : 1 . . ClientNum]  pr: priority

Op ≜ Ins ∪ Del  Now we don't consider Rd operations

SetNewAop(r, aopr) ≜
    aop′ = [aop EXCEPT ![r] = aopr]

ApplyNewAop(r) ≜
    state′ = [state EXCEPT ![r] = Apply(aop′[r], @)]
─────────────────────────────────────────────────────────
```

```
TypeOKInt ≜
      ∧ aop ∈ [Replica → Op ∪ {Nop}]
      ∧ state ∈ [Replica → List]
      ∧ Comm!TypeOK
      ∧ chins ⊆ Char

InitInt ≜
      ∧ aop = [r ∈ Replica ↦ Nop]
      ∧ state = [r ∈ Replica ↦ InitState]
      ∧ Comm!Init
      ∧ chins = Char

DoIns(DoOp(_, _), c) ≜    Client c ∈ Client generates an "Ins" operation.
     ∃ ins ∈ Ins :
          ∧ ins.pos ∈ 1 . . (Len(state[c]) + 1)
          ∧ ins.ch ∈ chins
          ∧ ins.pr = Priority[c]
          ∧ DoOp(c, ins)
          ∧ chins′ = chins \ {ins.ch}  We assume that all inserted elements are unique.

DoDel(DoOp(_, _), c) ≜    Client c ∈ Client generates a "Del" operation.
     ∃ del ∈ Del :
          ∧ del.pos ∈ 1 . . Len(state[c])
          ∧ DoOp(c, del)
          ∧ UNCHANGED chins

DoInt(DoOp(_, _), c) ≜    Client c ∈ Client issues an operation.
      ∧ ∨ DoIns(DoOp, c)
        ∨ DoDel(DoOp, c)
      ∧ ApplyNewAop(c)

RevInt(ClientPerformInt(_, _), c) ≜    Client c ∈ Client receives and processes a message.
      ∧ Comm!CRev(c)
      ∧ ClientPerformInt(c, Head(cincoming[c]))
      ∧ ApplyNewAop(c)
      ∧ UNCHANGED chins

SRevInt(ServerPerformInt(_)) ≜    The Server receives and processes a message.
      ∧   Comm!SRev
      ∧   ServerPerformInt(Head(sincoming))
      ∧   ApplyNewAop(Server)
      ∧   UNCHANGED chins
```

* Modification History
* Last modified *Wed Jan* 02 19:01:18 *CST* 2019 by *hengxin*
* Created *Tue Dec* 04 19:01:01 *CST* 2018 by *hengxin*