```
1  ---------- MODULE JupiterInterface ----------
   This module declares the parameters and defines the operators that describe the interface of a
   family of Jupiter specs.
6  EXTENDS Integers, SequenceUtils, OT
7  ----------
8  CONSTANTS
9      Char,        the set of characters
10     Client,      the set of client replicas
11     Server,      the (unique) server replica
12     InitState    the initial state of each replica

14 ASSUME  We assume that all inserted elements are unique.
15     ∧   Range(InitState) ∩ Char = {}   due to the uniqueness requirement
16 ----------
17 VARIABLES
18     aop,         op[r]: the actual operation applied at replica r ∈ Replica
19     state,       state[r]: state (the list content) of replica r ∈ Replica
20     cincoming,   cincoming[c]: incoming channel at the client c ∈ Client
21     sincoming,   incoming channel at the Server
22     chins        a set of chars allowed to insert; this is for model checking

24 intVars ≜ ⟨aop, state, cincoming, sincoming, chins⟩
25 ----------
26 Comm(Msg) ≜ INSTANCE CSComm

28 Replica ≜ Client ∪ {Server}

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

33 ClientNum ≜ Cardinality(Client)
34 Priority ≜ CHOOSE f ∈ [Client → 1 .. ClientNum] : Injective(f)
35 ----------
   The set of all operations. Note: The positions are indexed from 1.
39 Rd ≜ [type : {"Rd"}]
40 Del ≜ [type : {"Del"}, pos : 1 .. MaxLen]
41 Ins ≜ [type : {"Ins"}, pos : 1 .. (MaxLen + 1), ch : Char, pr : 1 .. ClientNum]  pr: priority

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

45 SetNewAop(r, aopr) ≜
46     aop' = [aop EXCEPT ![r] = aopr]

48 ApplyNewAop(r) ≜
49     state' = [state EXCEPT ![r] = Apply(aop'[r], @)]
50 ----------
51 TypeOKInt ≜
```

```
52      ∧ aop ∈ [Replica → Op ∪ {Nop}]
53      ∧ state ∈ [Replica → List]
54      ∧ chins ⊆ Char

56  InitInt ≜
57      ∧ aop = [r ∈ Replica ↦ Nop]
58      ∧ state = [r ∈ Replica ↦ InitState]
59      ∧ chins = Char

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

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

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

80  RevInt(c) ≜   Client c ∈ Client receives a message from the Server.
81      ∧ ApplyNewAop(c)
82      ∧ UNCHANGED chins

84  SRevInt ≜   The Server receives a message.
85      ∧ ApplyNewAop(Server)
86      ∧ UNCHANGED chins
87  ====================================================================

    \* Modification History
    \* Last modified Tue Jan 01 11:38:14 CST 2019 by hengxin
    \* Created Tue Dec 04 19:01:01 CST 2018 by hengxin
```