```
1  ---------- MODULE Cure ----------
   See ICDCS2016: "Cure: Strong Semantics Meets High Availability and Low Latency".
5  EXTENDS Naturals, Sequences, TLC
6  ----------
7  Max(a, b) ≜ IF a < b THEN b ELSE a
8  Min(S) ≜ CHOOSE a ∈ S : ∀ b ∈ S : a ≤ b
9  Range(f) ≜ {f[x] : x ∈ DOMAIN f}
10 Last(seq) ≜ seq[Len(seq)]
11 ----------
12 CONSTANTS
13     Key,              the set of keys, ranged over by k ∈ Key
14     Value,            the set of values, ranged over by v ∈ Value
15     Client,           the set of clients, ranged over by c ∈ Client
16     Partition,        the set of partitions, ranged over by p ∈ Partition
17     Datacenter,       the set of datacenters, ranged over by d ∈ Datacenter
18     KeySharding,      the mapping from Key to Partition
19     ClientAttachment  the mapping from Client to Datacenter

21 NotVal ≜ CHOOSE v : v ∉ Value

23 ASSUME
24     ∧ KeySharding ∈ [Key → Partition]
25     ∧ ClientAttachment ∈ [Client → Datacenter]
26 ----------
27 VARIABLES
28  At the client side:
29     cvc,      cvc[c]: the vector clock of client c ∈ Client
30  At the server side (each for partition p ∈ Partition in d ∈ Datacenter):
31     clock,    clock[p][d]: the current clock
32     pvc,      pvc[p][d]: the vector clock
33     css,      css[p][d]: the stable snapshot
34     store,    store[p][d]: the kv store
35     updates,  updates[p][d]: the buffer of updates
36  Clock management
37     tick,     tick[p][d]: toggle on clock ticks
38  Client-server communication
39     msgs      the set of messages in transit

41 cVars ≜ ⟨cvc⟩
42 sVars ≜ ⟨clock, pvc, css, store, updates, tick⟩
43 mVars ≜ ⟨msgs⟩
44 vars ≜ ⟨cvc, clock, pvc, css, store, updates, tick, msgs⟩
45 ----------
46 Clock ≜ Nat
47 VC ≜ [Datacenter → Clock]   vector clock with an entry per datacenter d ∈ Datacenter
48 VCInit ≜ [d ∈ Datacenter ↦ 0]
```

$KVTuple \triangleq [key : Key,\ val : Value \cup \{NotVal\},\ vc : VC]$

$Message \triangleq$
$\quad [type : \{\text{“ReadRequest”}\},\ key : Key,\ vc : VC,\ c : Client,\ p : Partition,\ d : Datacenter]$
$\quad \cup\ [type : \{\text{“ReadReply”}\},\ val : Value \cup \{NotVal\},\ vc : VC,\ c : Client]$
$\quad \cup\ [type : \{\text{“UpdateRequest”}\},\ key : Key,\ val : Value,\ vc : VC,\ c : Client,\ p : Partition,\ d : Datacenter]$
$\quad \cup\ [type : \{\text{“UpdateReply”}\},\ ts : Clock,\ c : Client,\ d : Datacenter]$
$d \in Datacenter$: the source datacenter; $dd \in Datacenter$: the destination datacenter
$\quad \cup\ [type : \{\text{“Replicate”}\},\ d\ : Datacenter,\ kvs : Seq(KVTuple),\ p : Partition,\ dd : Datacenter]$
$\quad \cup\ [type : \{\text{“Heartbeat”}\},\ d : Datacenter,\ ts : Clock,\ p : Partition,\ dd : Datacenter]$

$TypeOK \triangleq$
$\quad \land\ \ cvc \in [Client \to VC]$
$\quad \land\ \ clock \in [Partition \to [Datacenter \to Clock]]$
$\quad \land\ \ pvc \in [Partition \to [Datacenter \to VC]]$
$\quad \land\ \ css \in [Partition \to [Datacenter \to VC]]$
$\quad \land\ \ store \in [Partition \to [Datacenter \to \text{SUBSET } KVTuple]]$
$\quad \land\ \ updates \in [Partition \to [Datacenter \to Seq(KVTuple)]]$
$\quad \land\ \ tick \in [Partition \to [Datacenter \to \text{BOOLEAN }]]$
$\quad \land\ \ msgs \subseteq Message$

$Init \triangleq$
$\quad \land\ cvc = [c \in Client \mapsto VCInit]$
$\quad \land\ clock = [p \in Partition \mapsto [d \in Datacenter \mapsto 0]]$
$\quad \land\ pvc = [p \in Partition \mapsto [d \in Datacenter \mapsto VCInit]]$
$\quad \land\ css = [p \in Partition \mapsto [d \in Datacenter \mapsto VCInit]]$
$\quad \land\ store = [p \in Partition \mapsto [d \in Datacenter \mapsto$
$\qquad\qquad [key : \{k \in Key : KeySharding[k] = p\},\ val : \{NotVal\},\ vc : \{VCInit\}]]]$
$\quad \land\ updates = [p \in Partition \mapsto [d \in Datacenter \mapsto \langle\rangle]]$
$\quad \land\ tick\ = [p \in Partition \mapsto [d \in Datacenter \mapsto \text{FALSE}]]$
$\quad \land\ msgs = \{\}$

$Send(m) \triangleq msgs' = msgs \cup \{m\}$
$SendSet(ms) \triangleq msgs' = msgs \cup ms$
$SendAndDelete(sm,\ dm) \triangleq msgs' = (msgs \cup \{sm\}) \setminus \{dm\}$

$CanIssue(c) \triangleq \forall\, m \in msgs :$
$\quad m.type \in \{\text{“ReadRequest”},\ \text{“ReadReply”},\ \text{“UpdateRequest”},\ \text{“UpdateReply”}\} \Rightarrow m.c \neq c$

Client operations at client $c \in Client$.

$Read(c,\ k) \triangleq$ $c \in Client$ reads from $k \in Key$
$\quad \land\ CanIssue(c)$
$\quad \land\ Send([type \mapsto \text{“ReadRequest”},\ key \mapsto k,\ vc \mapsto cvc[c],$
$\qquad\qquad c \mapsto c,\ p \mapsto KeySharding[k],\ d \mapsto ClientAttachment[c]])$
$\quad \land\ \text{UNCHANGED } \langle cVars,\ sVars\rangle$

```
96  ReadReply(c) ≜   c ∈ Client handles the reply to its read request
97      ∧ ∃ m ∈ msgs :
98          ∧ m.type = "ReadReply" ∧ m.c = c   such m is unique
99          ∧ cvc' = [cvc EXCEPT ![c] = [d ∈ Datacenter ↦ Max(m.vc[d], @[d])]]
100         ∧ msgs' = msgs \ {m}
101     ∧ UNCHANGED ⟨sVars⟩

103 Update(c, k, v) ≜   c ∈ Client updates k ∈ Key with v ∈ Value
104     ∧ CanIssue(c)
105     ∧ Send([type ↦ "UpdateRequest", key ↦ k, val ↦ v,
106             vc ↦ cvc[c], c ↦ c, p ↦ KeySharding[k], d ↦ ClientAttachment[c]])
107     ∧ UNCHANGED ⟨cVars, sVars⟩

109 UpdateReply(c) ≜   c ∈ Client handles the reply to its update request
110     ∧ ∃ m ∈ msgs :
111         ∧ m.type = "UpdateReply" ∧ m.c = c   such m is unique
112         ∧ cvc' = [cvc EXCEPT ![c][m.d] = m.ts]
113         ∧ msgs' = msgs \ {m}
114     ∧ UNCHANGED ⟨sVars⟩
115 ├──────────────────────────────────────────────────────────────────────────┤
116  Server operations at partition p ∈ Partition in datacenter d ∈ Datacenter.

118 ReadRequest(p, d) ≜   handle a "ReadRequest"
119     ∧ ∃ m ∈ msgs :
120         ∧ m.type = "ReadRequest" ∧ m.p = p ∧ m.d = d   such m may be not unique
121         ∧ css' = [css EXCEPT ![p][d] =
122             [dc ∈ Datacenter ↦ IF dc = d THEN @[dc] ELSE Max(m.vc[dc], @[dc])]]
123         ∧ LET kvs ≜ {kv ∈ store[p][d] :
124                         ∧ kv.key = m.key
125                         ∧ ∀ dc ∈ Datacenter \ {d} : kv.vc[dc] ≤ css'[p][d][dc]}
126               lkv ≜ CHOOSE kv ∈ kvs :   choose the latest one (Existence? Uniqueness?)
127                         ∀ akv ∈ kvs, dc ∈ Datacenter : akv.vc[dc] ≤ kv.vc[dc]
128           IN  SendAndDelete([type ↦ "ReadReply", val ↦ lkv.val, vc ↦ lkv.vc, c ↦ m.c], m)
129     ∧ UNCHANGED ⟨cVars, clock, pvc, store, updates, tick⟩

131 UpdateRequest(p, d) ≜   handle a "UpdateRequest"
132     ∧ ∃ m ∈ msgs :
133         ∧ m.type = "UpdateRequest" ∧ m.p = p ∧ m.d = d   such m may be not unique
134         ∧ m.vc[d] < clock[p][d]   waiting condition; (" ≤ " strengthed to " < ")
135         ∧ pvc' = [pvc EXCEPT ![p][d][d] = clock[p][d]]
136         ∧ css' = [css EXCEPT ![p][d] =
137             [dc ∈ Datacenter ↦ IF dc = d THEN @[dc] ELSE Max(m.vc[dc], @[dc])]]
138         ∧ LET kv ≜ [key ↦ m.key, val ↦ m.val,
139                     vc ↦ [m.vc EXCEPT ![d] = clock[p][d]]]
140           IN  ∧ store' = [store EXCEPT ![p][d] = @ ∪ {kv}]
141               ∧ updates' = [updates EXCEPT ![p][d] = @ ∘ ⟨kv⟩]
```

```
142                 ∧ SendAndDelete([type ↦ "UpdateReply", ts ↦ clock[p][d], c ↦ m.c, d ↦ d], m)
143     ∧ UNCHANGED ⟨cVars, clock, pvc, tick⟩

145 PropagateUpdates(p, d) ≜  propagate buffered updates to other datacenters
146     ∧ IF updates[p][d] ≠ ⟨⟩
147         THEN ∧ SendSet([type : {"Replicate"}, d : {d}, kvs : {updates[p][d]}, p : {p}, dd : Datacenter \ {d}])
148              ∧ updates' = [updates EXCEPT ![p][d] = ⟨⟩]
149              ∧ UNCHANGED ⟨tick⟩
150         ELSE ∧ tick[p][d]
151              ∧ SendSet([type : {"Heartbeat"}, d : {d}, ts : {pvc[p][d][d]}, p : {p}, dd : Datacenter \ {d}])
152              ∧ tick' = [tick EXCEPT ![p][d] = FALSE]
153              ∧ UNCHANGED ⟨updates⟩
154     ∧ UNCHANGED ⟨cVars, cvc, clock, pvc, css, store⟩

156 Replicate(p, d) ≜  handle a "Replicate"
157     ∧ ∃ m ∈ msgs :
158         ∧ m.type = "Replicate" ∧ m.p = p ∧ m.dd = d
159         ∧ store'  = [store EXCEPT ![p][d] = @ ∪ Range(m.kvs)]
160         ∧ pvc' = [pvc EXCEPT ![p][d][d] = Last(m.kvs).vc[m.d]]
161         ∧ msgs' = msgs \ {m}
162     ∧ UNCHANGED ⟨cVars, cvc, clock, css, updates, tick⟩

164 Heartbeat(p, d) ≜  handle a "Heartbeat"
165     ∧ ∃ m ∈ msgs :
166         ∧ m.type = "Heartbeat" ∧ m.p = p ∧ m.dd = d
167         ∧ pvc' = [pvc EXCEPT ![p][d][m.d] = m.ts]
168         ∧ msgs' = msgs \ {m}
169     ∧ UNCHANGED ⟨cVars, cvc, clock, css, store, updates, tick⟩
170 ──────────────────────────────────────────────────────────────────────
171  Clock management at partition p ∈ Partition in datacenter d ∈ Datacenter
172 Tick(p, d) ≜  clock[p][d] ticks
173       ∧ clock' = [clock EXCEPT ![p][d] = @ + 1]
174       ∧ pvc' = [pvc EXCEPT ![p][d][d] = clock'[p][d]]
175       ∧ tick' = [tick EXCEPT ![p][d] = TRUE]
176       ∧ UNCHANGED ⟨cVars, mVars, cvc, css, store, updates⟩

178 UpdateCSS(p, d) ≜  update css[p][d]
179     ∧ css' = [css EXCEPT ![p][d] =
180                 [dc ∈ Datacenter ↦ IF dc = d THEN @[dc]
181                                      ELSE Min({pvc[pp][d][dc] : pp ∈ Partition})]]
182     ∧ UNCHANGED ⟨cVars, mVars, clock, pvc, store, updates, tick⟩
183 ──────────────────────────────────────────────────────────────────────
184 Next ≜
185     ∨ ∃ c ∈ Client, k ∈ Key : Read(c, k)
186     ∨ ∃ c ∈ Client, k ∈ Key, v ∈ Value : Update(c, k, v)
187     ∨ ∃ c ∈ Client : ReadReply(c) ∨ UpdateReply(c)
```

$$
\begin{array}{l}
\quad \lor \exists\, p \in \mathit{Partition},\ d \in \mathit{Datacenter} : \\
\qquad \lor \mathit{ReadRequest}(p,\ d) \\
\qquad \lor \mathit{UpdateRequest}(p,\ d) \\
\qquad \lor \mathit{PropagateUpdates}(p,\ d) \\
\qquad \lor \mathit{Replicate}(p,\ d) \\
\qquad \lor \mathit{Heartbeat}(p,\ d) \\
\qquad \lor \mathit{Tick}(p,\ d) \\
\qquad \lor \mathit{UpdateCSS}(p,\ d) \\
\\
\mathit{Spec} \triangleq \mathit{Init} \land \Box[\mathit{Next}]_{\mathit{vars}} \\
\end{array}
$$