```
MODULE crdb_writes

EXTENDS $TLC$, $Integers$, $FiniteSets$

CONSTANT $Servers$
CONSTANT $Values$
CONSTANT $Clients$
CONSTANT $ClockTicker$
CONSTANT $NULL$

$MsgTypes \triangleq \{$
    “write”,
    “push”,
    “commit”
$\}$

$Range(f) \triangleq \{f[x] : x \in \text{DOMAIN } f\}$
$Timestamps \triangleq 0 \,..\, 2$
$MaxTimestamp \triangleq \text{CHOOSE } ts \in Timestamps :$
    $\forall\, other \in Timestamps :$
        $other \leq ts$

ASSUME $Cardinality(Servers) > 0$
ASSUME $Cardinality(Clients) = Cardinality(Servers)$

$ClientToServer \triangleq \text{CHOOSE } f \in [Clients \rightarrow Servers] : \text{TRUE}$

$PrioritizePushes(set) \triangleq$
    LET
        $pushes \triangleq \{r \in set : r.type = \text{“push”}\}$
    IN
        IF $pushes = \{\}$
         THEN $set$
         ELSE $pushes$

$IntentExists(store) \triangleq$
    $\exists\, k \in \text{DOMAIN } store : \neg store[k][2]$

$GetIntent(store) \triangleq$
    CHOOSE $k \in \text{DOMAIN } store : \neg store[k][2]$

  --algorithm crdb

variables
    $lease = \text{CHOOSE } s \in Servers : \text{TRUE},$
    $storage = \langle\rangle,$
    $requests = \{\},$
    $responses = \{\},$
    $clock = [s \in Servers \mapsto 1]$,
    $tsCache = [s \in Servers \mapsto 0]$,
     Set of timestamps corresponding to committed transactions
    $committed = \{\}$ ;

**process** $server \in Servers$
**variables**
    $msg = NULL$ ;
**begin**
$Run$:
**while** $\exists c \in Clients : pc[c] \neq$ "Done" **do**
$Receive$:
    **await** $\land lease = self$ ;
    **with**
        $req \in PrioritizePushes(requests)$
     **do**
        $msg := req$ ||
        $requests := requests \setminus \{req\}$ ;
    **end with** ;
$PushClock$:
     Push the clock to request header $ts$
    **if** $clock[self] < msg.txn.ts$ **then**
        $clock[self] := msg.txn.ts$ ;
    **end if** ;
$EvalRequest$:
    **if** $msg.type =$ "write" **then**
        $HandleWrite$:
            **if** $IntentExists(storage)$
             **then**
                $requests := requests \cup \{[$
                    $type \mapsto$ "push",
                    $txn \mapsto msg.txn$,
                    $from \mapsto msg.from$,
                    $intent \mapsto GetIntent(storage)$
                $]\}$ ;
             **else**
                $storage := msg.txn.ts :> \langle msg.txn.value, \text{FALSE} \rangle \, @@ \, storage$ ;
                $responses := responses \cup \{[$
                    $to \mapsto msg.from$
                $]\}$
            **end if** ;
     **elsif** $msg.type =$ "push" **then**
        $HandlePush$:
            $tsCache[self] := msg.intent$ ;
            $requests := requests \cup \{[$
```

```
                    $type \mapsto$ "write",
                    $from \mapsto msg.from$,
                    $txn \mapsto msg.txn$
                ]};
                $storage := [ts \in (\text{DOMAIN } storage \setminus \{msg.intent\}) \mapsto storage[ts]]$;
    else
            $HandleCommit$:
                $storage := msg.txn.ts :> \langle msg.txn.value, \text{TRUE} \rangle @@ storage$
    end if ;
end while ;
end process ;

process $client \in Clients$
variables
    $client\_txn \in [ts : \{0\}, value : Values]$
begin
$Begin$:
    with
        $now = clock[ClientToServer[self]]$
    do
        $client\_txn.ts := now$;
    end with ;
$SendWrite$:
    $requests := requests \cup \{[$
        $type \mapsto$ "write",
        $from \mapsto self$,
        $txn \mapsto client\_txn$
    ]};
$WaitForResponse$:
    await $\exists\, resp \in responses : resp.to = self$;
    with
        $resp \in \{r \in responses : r.to = self\}$
    do
        print $resp$;
        $responses := responses \setminus \{resp\}$;
    end with ;
$SendCommit$:
    $requests := requests \cup \{[$
        $type \mapsto$ "commit",
        $to \mapsto lease$,
        $from \mapsto self$,
        $txn \mapsto client\_txn$
    ]};
end process ;
```

```
**process** $clock\_ticker = ClockTicker$
**begin**
$TickClocks$:
    **while** $(\exists ts \in Range(clock) : ts < MaxTimestamp)$ **do**
        **with**
            $s \in \{s \in Servers : clock[s] < MaxTimestamp\}$
        **do**
            $clock[s] := clock[s] + 1$ **;**
        **end with ;**
    **end while ;**
**end process ;**
**end algorithm** ;
```

BEGIN TRANSLATION

VARIABLES $lease, storage, requests, responses, clock, tsCache, committed, pc, msg, client\_txn$

$vars \triangleq \langle lease, storage, requests, responses, clock, tsCache, committed, pc, msg, client\_txn\rangle$

$ProcSet \triangleq (Servers) \cup (Clients) \cup \{ClockTicker\}$

$Init \triangleq$ Global variables

$$\begin{aligned}
&\land lease = (\text{CHOOSE } s \in Servers : \text{TRUE})\\
&\land storage = \langle\rangle\\
&\land requests = \{\}\\
&\land responses = \{\}\\
&\land clock = [s \in Servers \mapsto 1]\\
&\land tsCache = [s \in Servers \mapsto 0]\\
&\land committed = \{\}
\end{aligned}$$

Process server

$$\land msg = [self \in Servers \mapsto NULL]$$

Process client

$$\begin{aligned}
&\land client\_txn \in [Clients \to [ts : \{0\}, value : Values]]\\
&\land pc = [self \in ProcSet \mapsto \text{CASE } self \in Servers \to \text{“Run”}\\
&\qquad\qquad \Box\ self \in Clients \to \text{“Begin”}\\
&\qquad\qquad \Box\ self = ClockTicker \to \text{“TickClocks”}]
\end{aligned}$$

$$\begin{aligned}
Run(self) \triangleq\ &\land pc[self] = \text{“Run”}\\
&\land \text{IF } \exists c \in Clients : pc[c] \neq \text{“Done”}\\
&\qquad \text{THEN } \land pc' = [pc \text{ EXCEPT } ![self] = \text{“Receive”}]\\
&\qquad \text{ELSE } \land pc' = [pc \text{ EXCEPT } ![self] = \text{“Done”}]\\
&\land \text{UNCHANGED } \langle lease, storage, requests, responses, clock,\\
&\qquad tsCache, committed, msg, client\_txn\rangle
\end{aligned}$$

$$\begin{aligned}
Receive(self) \triangleq\ &\land pc[self] = \text{“Receive”}\\
&\land \land lease = self
\end{aligned}$$

```
$\land \exists\, req \in PrioritizePushes(requests) :$
      $\land msg' = [msg \text{ EXCEPT } ![self] = req]$
      $\land requests' = requests \setminus \{req\}$
$\land pc' = [pc \text{ EXCEPT } ![self] = \text{“PushClock”}]$
$\land \text{UNCHANGED } \langle lease, storage, responses, clock, tsCache,$
                        $committed, client\_txn \rangle$

$PushClock(self) \triangleq$ $\land pc[self] = \text{“PushClock”}$
      $\land \text{IF } clock[self] < msg[self].txn.ts$
            $\text{THEN } \land clock' = [clock \text{ EXCEPT } ![self] = msg[self].txn.ts]$
            $\text{ELSE } \land \text{TRUE}$
                  $\land clock' = clock$
      $\land pc' = [pc \text{ EXCEPT } ![self] = \text{“EvalRequest”}]$
      $\land \text{UNCHANGED } \langle lease, storage, requests, responses,$
                        $tsCache, committed, msg, client\_txn \rangle$

$EvalRequest(self) \triangleq$ $\land pc[self] = \text{“EvalRequest”}$
      $\land \text{IF } msg[self].type = \text{“write”}$
            $\text{THEN } \land pc' = [pc \text{ EXCEPT } ![self] = \text{“HandleWrite”}]$
            $\text{ELSE } \land \text{IF } msg[self].type = \text{“push”}$
                        $\text{THEN } \land pc' = [pc \text{ EXCEPT } ![self] = \text{“HandlePush”}]$
                        $\text{ELSE } \land pc' = [pc \text{ EXCEPT } ![self] = \text{“HandleCommit”}]$
      $\land \text{UNCHANGED } \langle lease, storage, requests, responses,$
                        $clock, tsCache, committed, msg,$
                        $client\_txn \rangle$

$HandleWrite(self) \triangleq$ $\land pc[self] = \text{“HandleWrite”}$
      $\land \text{IF } IntentExists(storage)$
            $\text{THEN } \land requests' = ($          $requests \cup \{[$
                                    $type \mapsto \text{“push”},$
                                    $txn \mapsto msg[self].txn,$
                                    $from \mapsto msg[self].from,$
                                    $intent \mapsto GetIntent(storage)$
                              $]\})$
                  $\land \text{UNCHANGED } \langle storage, responses \rangle$
            $\text{ELSE } \land storage' = (msg[self].txn.ts :> \langle msg[self].txn.value, \text{FALSE} \rangle \mathbin{@@} storage)$
                  $\land responses' = ($          $responses \cup \{[$
                                    $to \mapsto msg[self].from$
                              $]\})$
                  $\land \text{UNCHANGED } requests$
      $\land pc' = [pc \text{ EXCEPT } ![self] = \text{“Run”}]$
      $\land \text{UNCHANGED } \langle lease, clock, tsCache, committed, msg,$
                        $client\_txn \rangle$

$HandlePush(self) \triangleq$ $\land pc[self] = \text{“HandlePush”}$
      $\land tsCache' = [tsCache \text{ EXCEPT } ![self] = msg[self].intent]$
```

```
          $\land$ $requests' = ($          $requests \cup \{[$
                      $type \mapsto$ "write",
                      $from \mapsto msg[self].from,$
                      $txn \mapsto msg[self].txn$
                   $]\})$
          $\land$ $storage' = [ts \in (\textsc{domain}\ storage \setminus \{msg[self].intent\}) \mapsto storage[ts]]$
          $\land$ $pc' = [pc\ \textsc{except}\ ![self] =$ "Run"$]$
          $\land$ $\textsc{unchanged}\ \langle lease, responses, clock, committed, msg,$
                       $client\_txn \rangle$

$HandleCommit(self) \triangleq$ $\land$ $pc[self] =$ "HandleCommit"
                     $\land$ $storage' = (msg[self].txn.ts :> \langle msg[self].txn.value, \textsc{true} \rangle \mathbin{@@} storage)$
                     $\land$ $pc' = [pc\ \textsc{except}\ ![self] =$ "Run"$]$
                     $\land$ $\textsc{unchanged}\ \langle lease, requests, responses, clock,$
                                  $tsCache, committed, msg, client\_txn \rangle$

$server(self) \triangleq$ $Run(self) \lor Receive(self) \lor PushClock(self)$
               $\lor EvalRequest(self) \lor HandleWrite(self)$
               $\lor HandlePush(self) \lor HandleCommit(self)$

$Begin(self) \triangleq$ $\land$ $pc[self] =$ "Begin"
              $\land$ $\textsc{let}\ now \triangleq clock[ClientToServer[self]]\ \textsc{in}$
                   $client\_txn' = [client\_txn\ \textsc{except}\ ![self].ts = now]$
              $\land$ $pc' = [pc\ \textsc{except}\ ![self] =$ "SendWrite"$]$
              $\land$ $\textsc{unchanged}\ \langle lease, storage, requests, responses, clock,$
                           $tsCache, committed, msg \rangle$

$SendWrite(self) \triangleq$ $\land$ $pc[self] =$ "SendWrite"
                  $\land$ $requests' = ($          $requests \cup \{[$
                              $type \mapsto$ "write",
                              $from \mapsto self,$
                              $txn \mapsto client\_txn[self]$
                           $]\})$
                  $\land$ $pc' = [pc\ \textsc{except}\ ![self] =$ "WaitForResponse"$]$
                  $\land$ $\textsc{unchanged}\ \langle lease, storage, responses, clock, tsCache,$
                               $committed, msg, client\_txn \rangle$

$WaitForResponse(self) \triangleq$ $\land$ $pc[self] =$ "WaitForResponse"
                        $\land$ $\exists\, resp \in responses : resp.to = self$
                        $\land$ $\exists\, resp \in \{r \in responses : r.to = self\} :$
                             $\land$ $PrintT(resp)$
                             $\land$ $responses' = responses \setminus \{resp\}$
                        $\land$ $pc' = [pc\ \textsc{except}\ ![self] =$ "SendCommit"$]$
                        $\land$ $\textsc{unchanged}\ \langle lease, storage, requests, clock,$
                                     $tsCache, committed, msg, client\_txn \rangle$

$SendCommit(self) \triangleq$ $\land$ $pc[self] =$ "SendCommit"
```

```
                        ∧ requests′ = (           requests ∪ {[
                                           type ↦ "commit",
                                           to ↦ lease,
                                           from ↦ self,
                                           txn ↦ client_txn[self]
                                      ]})
                        ∧ pc′ = [pc EXCEPT ![self] = "Done"]
                        ∧ UNCHANGED ⟨lease, storage, responses, clock, tsCache,
                                     committed, msg, client_txn⟩

client(self) ≜ Begin(self) ∨ SendWrite(self) ∨ WaitForResponse(self)
                  ∨ SendCommit(self)

TickClocks ≜ ∧ pc[ClockTicker] = "TickClocks"
             ∧ IF (∃ ts ∈ Range(clock) : ts < MaxTimestamp)
                  THEN ∧ ∃ s ∈ {s ∈ Servers : clock[s] < MaxTimestamp} :
                             clock′ = [clock EXCEPT ![s] = clock[s] + 1]
                       ∧ pc′ = [pc EXCEPT ![ClockTicker] = "TickClocks"]
                  ELSE ∧ pc′ = [pc EXCEPT ![ClockTicker] = "Done"]
                       ∧ clock′ = clock
             ∧ UNCHANGED ⟨lease, storage, requests, responses, tsCache,
                          committed, msg, client_txn⟩

clock_ticker ≜ TickClocks

Next ≜ clock_ticker
          ∨ (∃ self ∈ Servers : server(self))
          ∨ (∃ self ∈ Clients : client(self))
          ∨  Disjunct to prevent deadlock on termination
            ((∀ self ∈ ProcSet : pc[self] = "Done") ∧ UNCHANGED vars)

Spec ≜ Init ∧ □[Next]_vars

Termination ≜ ◇(∀ self ∈ ProcSet : pc[self] = "Done")

 END TRANSLATION


IsTxn(txn) ≜
    ∧ txn.ts ∈ Timestamps
    ∧ txn.value ∈ Values

IsRequest(req) ≜
    ∧ req.type ∈ MsgTypes
    ∧ IsTxn(req.txn)

IsResponse(req) ≜
    ∧ req.to ∈ Clients
```

```
$ServerOk \triangleq$
    $\forall m \in Range(msg) :$
        $\lor m = NULL$
        $\lor IsRequest(m)$

$RequestsOk \triangleq \forall req \in requests : IsRequest(req)$

$ResponsesOk \triangleq \forall resp \in responses : IsResponse(resp)$

$StorageOk \triangleq$
     $\lor storage = \langle\rangle$
     $\lor \forall ts \in \text{DOMAIN } storage :$
         $\land ts \in Timestamps$
         $\land storage[ts] \in (Values \times \text{BOOLEAN})$

$NothingIsCommitted \triangleq \forall record \in Range(storage) : \neg record[2]$

$StaysCommitted \triangleq$
    $\Box[\forall x \in \text{DOMAIN } storage :$
        $storage[x][2] \Rightarrow \land x \in \text{DOMAIN } storage'$
                               $\land storage[x] = (storage')[x]$
     $]_{vars}$

$CommittedOk \triangleq committed \subseteq Timestamps$

$OnlyOneIntent \triangleq$
    $\forall a, b \in \text{DOMAIN } storage :$
        $\neg storage[a][2] \land \neg storage[b][2] \Rightarrow a = b$

$NoPartialCommit \triangleq$
    $\forall ts \in committed :$
        $\land ts \in \text{DOMAIN } storage$
        $\land storage[ts][2]$
```

\* Modification History
\* Last modified *Wed* May 15 17:15:55 *EDT* 2019 by *ajwerner*
\* Created *Wed* May 15 13:18:23 *EDT* 2019 by *ajwerner*