Link failure and two-phase commit

```
CHAN(From=0, To =1)
   =(chan[From][To].send[m:Msg] -> CHAN[m]
    | step1 -> CHAN
    | step2 -> CHAN['null]
    ),
CHAN[m:Msg]
   =(chan[From][To].recv[m] -> CHAN
    | step1 -> CHAN
    | step2 -> CHAN[m]
    | step2 ->linkfail[From][To]->CHAN['null]
    ).

||NETWORK = (forall[i:0..N-1][j:0..N-1]
               if (i!=j) then CHAN(i,j)
            ||CLOCK
            ).
```


Atomic Commitment Properties - safety

```
fluent
  LINKFAIL[i:ID] = <linkfail[i].{{[ID]}\{[i]}}, never>
```

Validity – 2:
If *all* processes vote *yes*, and there are **no failures**, then *yes* is the only possible decision value.

```
assert VALID_2
   = [](forall[i:ID]
        (VOTE[i]['yes] && !CRASHED[i] && !LINKFAIL[i])
                    -> !ABORT[ID])
```


Atomic Commitment Properties - liveness

•**Weak Termination**: If there are **no failures**, then all processes eventually decide.

```
assert
WEAKTERM = ([]forall[i:ID] (!CRASHED[i] && !LINKFAIL[i])
                -> <>forall[i:ID] DECIDED[i])
```

This reads: if its always the case that all processes do not crash and there are no link failures then eventually all processes reach the decided state.

Distributed two-phase commit

```
PARTICIPANT(Id=0)
  = (init->vote[Id][v:{yes,no}]->step1->SEND_ALL(Id,v); ROUND1[v]),
ROUND1[v:{yes,no,null}]
  = (chan.{{[ID]}\{[Id]}}.[Id].recv[m:Msg]
     -> if (v =='no) then ROUND1['no]
        else if (m == 'no) then ROUND1['no]
        else if (m == 'yes && v == 'yes) then ROUND1['yes]
        else if (m == 'yes && v == 'null) then ROUND1['null]
        else if (m == 'null && (v == 'yes || v == 'null )) then
ROUND1['null]
     |step1 -> ROUND2[v]
     ),
ROUND2[v:{yes,no,null}]
   = DECIDE(Id,v);ENDED,
ENDED
   = ({step1,step2}->ENDED)
     +{chan[ID][Id].recv[Msg],chan[Id][ID].send[Msg]}.
```