|  |
| --- |
| Software Engineering of Distributed Systems, KTH |
| Distributed Systems Advanced Homework 4 |
| Read-Impose Write-Consult and Read-Impose Write-Consult-Majority |

|  |
| --- |
| Shanbo Li and Sike Huang  2/24/2008 |

# Exercise 1. Read-Impose Write-Consult

# Exercise 2: RIWCM in Fail-Recovery Model

We launce process 0 and process 1 at the same time, with ops D500:W5:R:D5000:R:D30000 and D500:W6:R:D5000:R:D30000 respectively, When process 0 and 1 begin to wait for 30 seconds, we start process 2 with ops D500:R:D500:R:D10000.

We obtain the following results:

|  |
| --- |
| Node 0 |
| D500  wake up  W5  28684: Node 0: broabcast AtomicRegisterWriteEvent -> r:0, val:5  28724: Node 0: receive ReadMessage -> from:0, r:0  28727: Node 0: receive ReadValueMessage -> from:0, r:0, v:0  29704: Node 0: receive ReadMessage -> from:1, r:0  30741: Node 0: receive ReadValueMessage -> from:1, r:0, v:0  30743: Node 0: receive WriteMessage -> from:0, r:0, val:5  30744: ------ Write to local! r:0, val:5  30745: Node 0: receive AckMessage -> from:0, r:0  31712: Node 0: receive WriteMessage -> from:1, r:0, val:6  31713: ------ Write to local! r:0, val:6  32766: Node 0: receive AckMessage -> from:1, r:0  \*\*\*\*\*\*\* AtomicRegisterWriteReturnEvent-> r:0, val:5  R  32777: Node 0: AtomicRegisterReadEvent -> r:0  32841: Node 0: receive ReadMessage -> from:0, r:0  32909: Node 0: receive ReadValueMessage -> from:0, r:0, v:6  33757: Node 0: receive ReadMessage -> from:1, r:0  34809: Node 0: receive ReadValueMessage -> from:1, r:0, v:6  34833: Node 0: receive WriteMessage -> from:0, r:0, val:6  34899: Node 0: receive AckMessage -> from:0, r:0  35788: Node 0: receive WriteMessage -> from:1, r:0, val:6  36816: Node 0: receive AckMessage -> from:1, r:0  \*\*\*\*\*\*\*\* AtomicRegisterReadReturnEvent-> r:0, val:6  D5000  wake up  R  41827: Node 0: AtomicRegisterReadEvent -> r:0  41855: Node 0: receive ReadMessage -> from:0, r:0  41889: Node 0: receive ReadValueMessage -> from:0, r:0, v:6  42796: Node 0: receive ReadMessage -> from:1, r:0  43859: Node 0: receive ReadValueMessage -> from:1, r:0, v:6  43861: Node 0: receive WriteMessage -> from:0, r:0, val:6  43916: Node 0: receive AckMessage -> from:0, r:0  44801: Node 0: receive WriteMessage -> from:1, r:0, val:6  45863: Node 0: receive AckMessage -> from:1, r:0  \*\*\*\*\*\*\*\* AtomicRegisterReadReturnEvent-> r:0, val:6  D30000  49439: Node 0: receive ReadMessage -> from:2, r:0  51456: Node 0: receive WriteMessage -> from:2, r:0, val:6  53970: Node 0: receive ReadMessage -> from:2, r:0  55975: Node 0: receive WriteMessage -> from:2, r:0, val:6  wake up |

|  |
| --- |
| Node 1 |
| D500  wake up  W6  28611: Node 1: broabcast AtomicRegisterWriteEvent -> r:0, val:6  28619: Node 1: receive ReadMessage -> from:1, r:0  28620: Node 1: receive ReadValueMessage -> from:1, r:0, v:0  29739: Node 1: receive ReadMessage -> from:0, r:0  30708: Node 1: receive ReadValueMessage -> from:0, r:0, v:0  30710: Node 1: receive WriteMessage -> from:1, r:0, val:6  30710: ------ Write to local! r:0, val:6  30711: Node 1: receive AckMessage -> from:1, r:0  31745: Node 1: receive WriteMessage -> from:0, r:0, val:5  32719: Node 1: receive AckMessage -> from:0, r:0  \*\*\*\*\*\*\* AtomicRegisterWriteReturnEvent-> r:0, val:6  R  32721: Node 1: AtomicRegisterReadEvent -> r:0  32725: Node 1: receive ReadMessage -> from:1, r:0  32726: Node 1: receive ReadValueMessage -> from:1, r:0, v:6  33791: Node 1: receive ReadMessage -> from:0, r:0  34758: Node 1: receive ReadValueMessage -> from:0, r:0, v:6  34760: Node 1: receive WriteMessage -> from:1, r:0, val:6  34762: Node 1: receive AckMessage -> from:1, r:0  35812: Node 1: receive WriteMessage -> from:0, r:0, val:6  36791: Node 1: receive AckMessage -> from:0, r:0  \*\*\*\*\*\*\*\* AtomicRegisterReadReturnEvent-> r:0, val:6  D5000  wake up  R  41793: Node 1: AtomicRegisterReadEvent -> r:0  41795: Node 1: receive ReadMessage -> from:1, r:0  41796: Node 1: receive ReadValueMessage -> from:1, r:0, v:6  42831: Node 1: receive ReadMessage -> from:0, r:0  43798: Node 1: receive ReadValueMessage -> from:0, r:0, v:6  43799: Node 1: receive WriteMessage -> from:1, r:0, val:6  43800: Node 1: receive AckMessage -> from:1, r:0  44861: Node 1: receive WriteMessage -> from:0, r:0, val:6  45802: Node 1: receive AckMessage -> from:0, r:0  \*\*\*\*\*\*\*\* AtomicRegisterReadReturnEvent-> r:0, val:6  D30000  49444: Node 1: receive ReadMessage -> from:2, r:0  51457: Node 1: receive WriteMessage -> from:2, r:0, val:6  53969: Node 1: receive ReadMessage -> from:2, r:0  55979: Node 1: receive WriteMessage -> from:2, r:0, val:6  wake up |

|  |
| --- |
| Node 2 |
| D500  wake up  R  48397: Node 2: AtomicRegisterReadEvent -> r:0  48405: Node 2: receive ReadMessage -> from:2, r:0  48407: Node 2: receive ReadValueMessage -> from:2, r:0, v:0  50452: Node 2: receive ReadValueMessage -> from:0, r:0, v:6  50453: Node 2: receive ReadValueMessage -> from:1, r:0, v:6  50454: Node 2: receive WriteMessage -> from:2, r:0, val:6  50455: ------ Write to local! r:0, val:6  50456: Node 2: receive AckMessage -> from:2, r:0  52460: Node 2: receive AckMessage -> from:1, r:0  \*\*\*\*\*\*\*\* AtomicRegisterReadReturnEvent-> r:0, val:6  D500  52461: Node 2: receive AckMessage -> from:0, r:0  wake up  R  52963: Node 2: AtomicRegisterReadEvent -> r:0  52967: Node 2: receive ReadMessage -> from:2, r:0  52968: Node 2: receive ReadValueMessage -> from:2, r:0, v:6  54971: Node 2: receive ReadValueMessage -> from:1, r:0, v:6  54973: Node 2: receive WriteMessage -> from:2, r:0, val:6  54975: Node 2: receive ReadValueMessage -> from:0, r:0, v:6  54976: Node 2: receive AckMessage -> from:2, r:0  56977: Node 2: receive AckMessage -> from:0, r:0  \*\*\*\*\*\*\*\* AtomicRegisterReadReturnEvent-> r:0, val:6  D10000  56981: Node 2: receive AckMessage -> from:1, r:0 |

The value read by process 2 is “6” which is written by process 1. Though process 0 and process 1 try to write different value at the same time, which means two writes have the same timestamp, however, process 1 has higher rank than process 0, so finally “6” is written into the register rather than “5”.

The initial deed process 2 still can participate into the shared memory access, or to some extent, we can say that the read-impose write-consult-majority algorithm can be used in a fail-recovery model. Because to read a value from r, process pj­ reads all registers from writer[r, 1] to writer[r, N], and returns the value with the largest timestamp, whereas to write a value v in r, process pj reads all (1, N) registers and selects the largest timestamp, which it increments and associates with the value v to be written. In a word, as long as a process can obtain the largest timestamp, it can move on, no matter when it reads or writes.