- ... more

- dumping philosophy

- readers, writers

- producer-consumer

- mutual exclusion

- mutual exclusion protocol

Class: Process Communication
allowed in the Row code

multiple readers of a single writer is
Stationary

\[ \text{reader} \]

\[ \text{center} \]

Weather

If this is \( \text{a} \), then all else\...

\[ y \neq \text{not} \]

\[ \frac{2}{\text{not}} \]

\[ \text{w} \]
Learn how readers want for some story. Make the reader pay more attention to what they want. Do not be so mean to enlarge. If a mean person, Gf will help. If not, Fd in be there. Need every & no working unit.
Cooperative among all R
+ in threads
→ no "master" scheduler
→ code will be embedded in the
reader–writer & reader exit
sections
M. Weny
W. M.
weird

Reader-

Reader-

Reader-

Reader-

Reader-

Reader-

Reader-

Reader-

Reader-

Reader-

Reader-
Data Structures

1. Semaphores

2. Counters

RC -> # of active readers
WC -> # of waiting writers

w unc
w unc
Reader Entry

P(mutex);   // get CS
if (wwc>0) or (wc>0) then begin
rwc++;    // dec wait count
V(mutex);  // reader sleeps
end;
rc++;  // inc rc and enter.
if rwc>0 then V(rsem) // passes the mutex to 1 reader
else V(mutex);  // final reader unlocks mutex

Reader Exit

P(mutex);  // lock mutex
rc--;    // reader exit, decr count
if (rc=0) and (wwc>0) then V(wsem); // allow one W
else V(mutex);  // final reader unlocks mutex
Writer Entry

P(mutex);  // lock mutex

if (rc>0) or (wc>0)  // -->> not OK to enter
then begin

wwc++;  // increment w wait count

V(mutex);  // unlock mutex

P(wsem);  // writer sleep

wwc--;   // when wake up mutex is passed to me

end;

wc++;  // increment writer count

V(mutex);  // unlock mutex

WRITE CODE

Writer Exit

P(mutex);  // lock mutex

wc-- ;   // wc WILL be 0

if (rwc>0) then V(rsem)  // wake up 1 reader & pass mutex, cascade
else
    if (wwc>0) then V(wsem);
else V(mutex)

WRITE CODE