- **The structure of the “THE” multiprogramming system – Edsger W. Dijkstra**
  - Presented at the 1st ACM SOSP, Gatlinburg, Tennessee, October 1-4, 1967
  - Awarded SOSP Hall of Fame award in 2005
  - Paper discusses the reasoning behind techniques that might have been taught as facts in your ugrad course
    - Note that they don’t know the “correct terminology” for what they invented (e.g., deadlock)
  - Style is so different. On the other hand, it might be embarrassing to see you rant in 40 years

What it talks about?

- Dijkstra describes his invention called semaphores
- They describe page relocation (so pages back from drum/disk can go anywhere), virtual memory (segmented memory is larger than physical memory)
- They describe the need for clean interfaces and a layered approach so that we can verify the logical soundness of their design
- They argue that with proper synchronization, logical clocks can be used – the physical clock speeds are not relevant

Context

- Exciting times: multiple processors, devices (tape, tty) are becoming available.
- a desire for multiprogramming though multiuser systems are still far off.
- a desire to get more out of the device so that we can allow more users (perhaps commoditize computing though the authors do not explicitly say so)
- a desire to bring order to systems software development to manage complexity, bugs, correctness etc.

How were their ideas relevant back then?

- Semaphores – they over estimate the complexity of managing semaphores. They assume that semaphores encourage clean and provable systems. They mention “an unstable situation” which we now refer as unsafe and deadlocked
- Layered approach: they argue the some commercial entities might not appreciate the simplicity because their university system was just too simple
- Page relocation –hard to imagine because machines had so little resources (32K main, 512K disk)
- How many processors did they have?

How are the ideas relevant now

- Layers approach - later became the monolithic kernel vs microkernel/nano kernel argument. There has been much work on debugging/managing OS development. We are still not there yet
- Semaphores – undergrad OS students cannot escape semaphores
- Other systems ideas: VM, relocation etc. appropriate

- What do we mean by now: main frame, desktops, laptops, PDAs, phones?

Discussion

- This paper is more than 40 years old. What do you think is the state of the OS now?