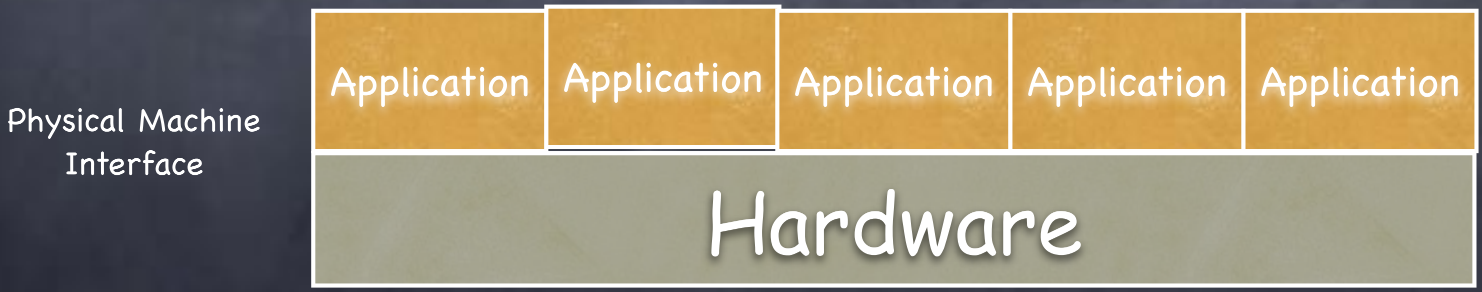
Cornell University

Introduction

CS 4410

What is an OS?

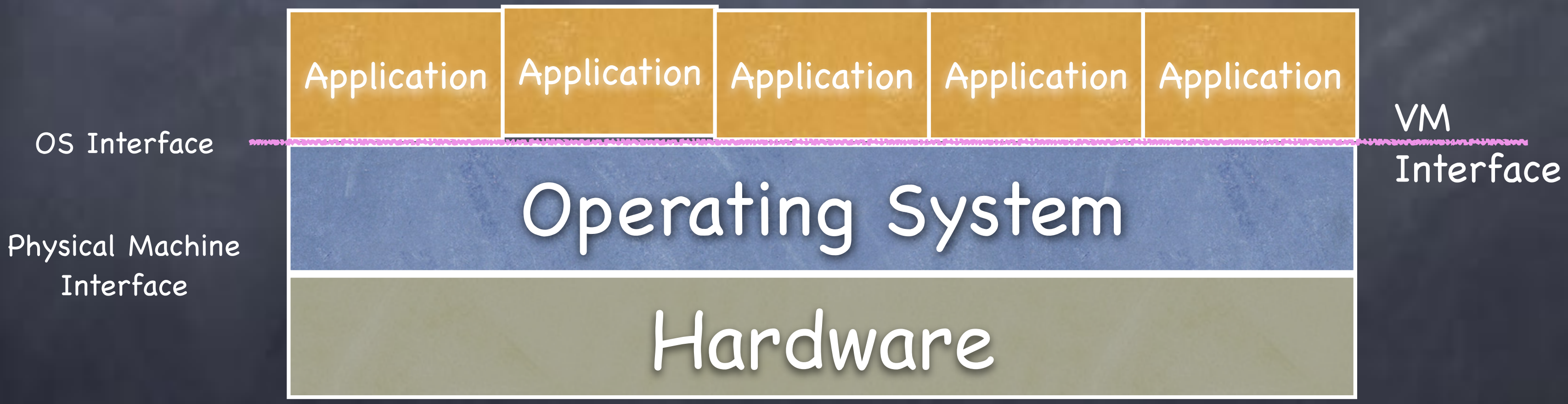
- An Operating System implements a virtual machine whose interface is more convenient* that the raw hardware interface



* easier to use, simpler to code, more reliable, more secure…

What is an OS?

- An Operating System implements a virtual machine whose interface is more convenient* that the raw hardware interface



* easier to use, simpler to code, more reliable, more secure…

Operating System

What is a system?

"A complex unit formed of many often diverse parts subject to a common plan or serving a common purpose"

Webster Third New International Dictionary

What is a system?

interconnections

components

"A complex unit formed of ~~many often~~
~~diverse parts~~ subject to a common plan
or serving a common purpose"

Webster Third New International Dictionary

a certain behavior
at the interface
with an environment

"A set of interconnected components
with an expected behavior observed at
the interface with its environment"

Common systems challenges

- Emergent properties
- Propagation of effects
- Incommensurate scaling
- Trade-offs

Emergent Properties

Evident only when components are combined

Emergent Properties

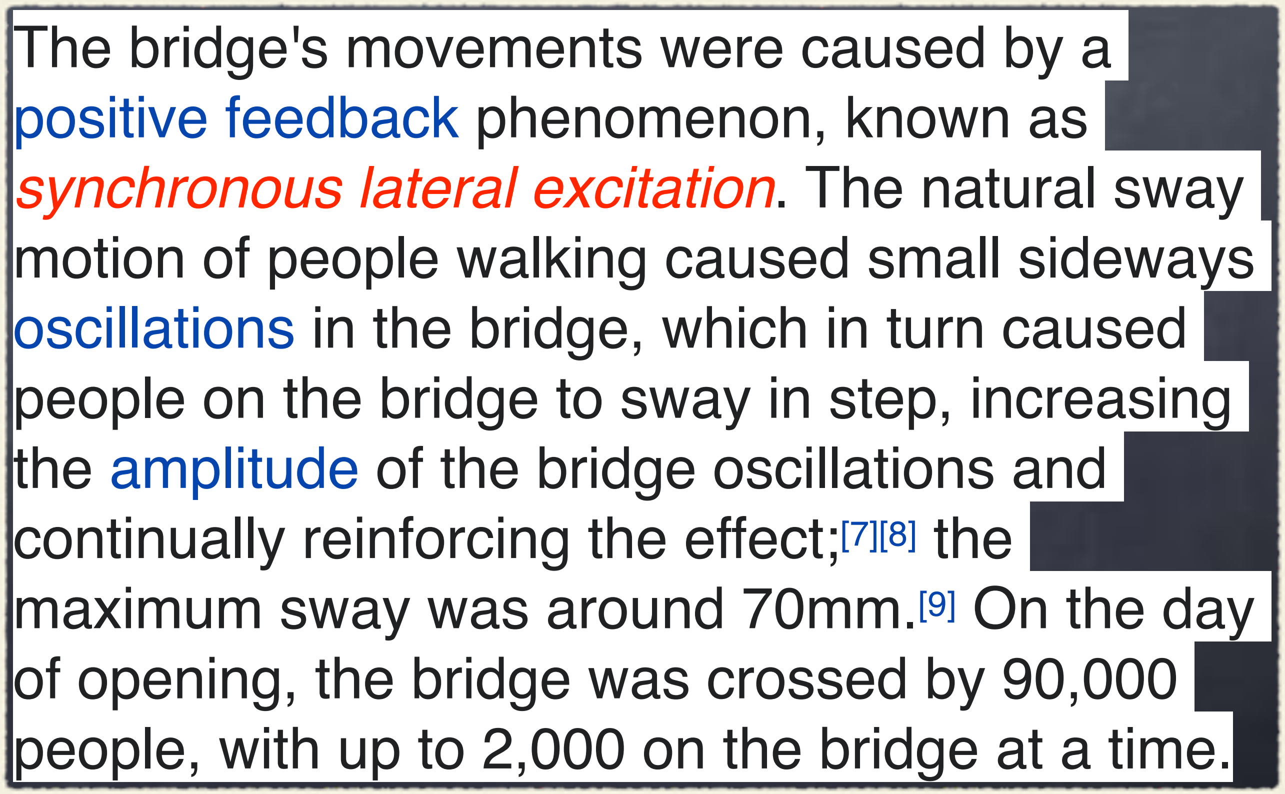
Millenium Bridge London

Emergent Properties

Millenium Bridge London

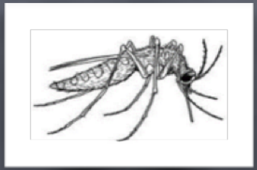
Emergent Properties

Millenium Bridge London



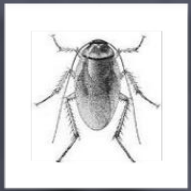
The bridge's movements were caused by a positive feedback phenomenon, known as *synchronous lateral excitation*. The natural sway motion of people walking caused small sideways oscillations in the bridge, which in turn caused people on the bridge to sway in step, increasing the amplitude of the bridge oscillations and continually reinforcing the effect;[7][8] the maximum sway was around 70mm.[9] On the day of opening, the bridge was crossed by 90,000 people, with up to 2,000 on the bridge at a time.

Propagation of effects: fighting malaria



- WHO sprayed villages in N. Borneo with DDT

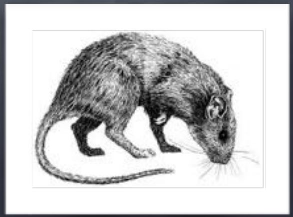
- Wiped out mosquitos, but…
- Roaches collected DDT in tissue



- Lizard ate roaches, and became slower

- Easy target for cats

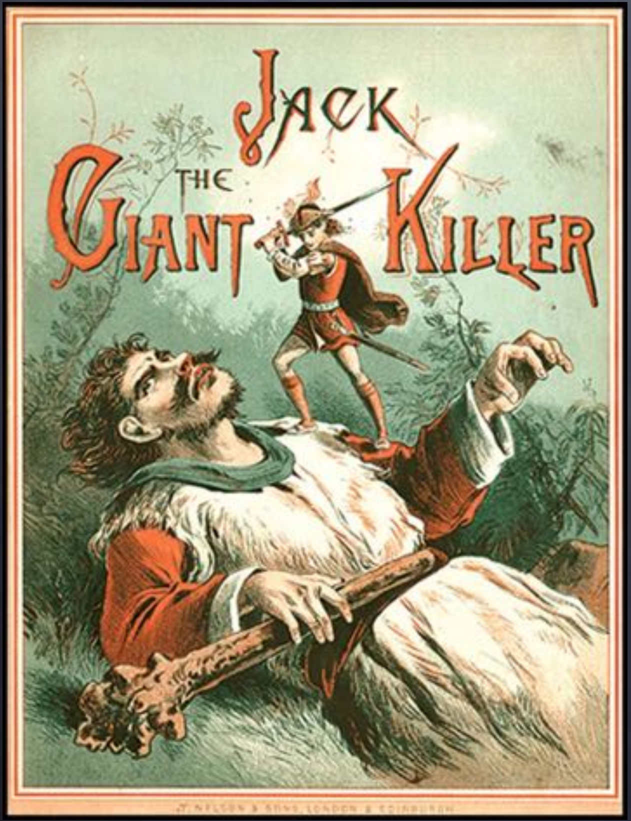
- DDT cause cats to die
- Forest rats moved into villages



- Rats carried the bacillus of the plague!

Incommensurate scaling

As the system increases in size or speed, not all components can manage the scale, and things break down



10x higher than Jack!

but also 10x wider and thicker:
About 1000x Jack’s weight!

The cross section of the Giant’s bones, however, is only 100x Jack’s

A human thigh bone breaks at about 10x human weight

The giant would have broken his thighs every time he was taking a step!

On being the right size
J.B.S. Haldane

Inevitable Trade-offs

Speed vs power in processors

Bandwidth vs computation in compression

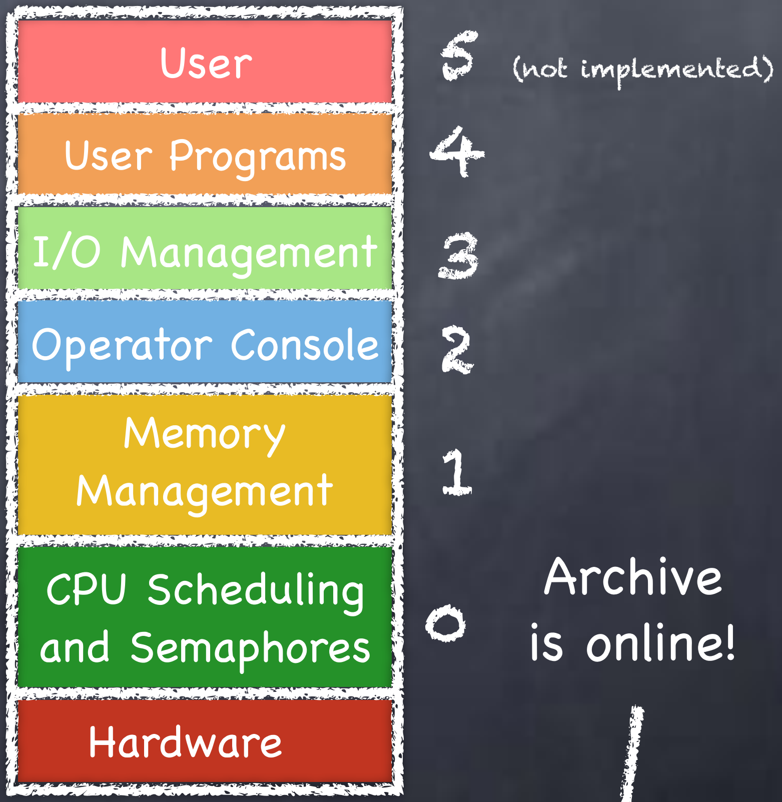
Space vs time almost everywhere

A pawn vs better position in chess

...

How to Manage Complexity

- Modularity
  - □ Good modularity minimizes connections between components
- Abstraction
  - □ Separate interface from internals; separate specification from implementation
- Hierarchy/Layering
  - □ Constrain interactions so they are easier to understand



User 5 (not implemented)

User Programs 4

I/O Management 3

Operator Console 2

Memory Management 1

CPU Scheduling and Semaphores 0

Hardware

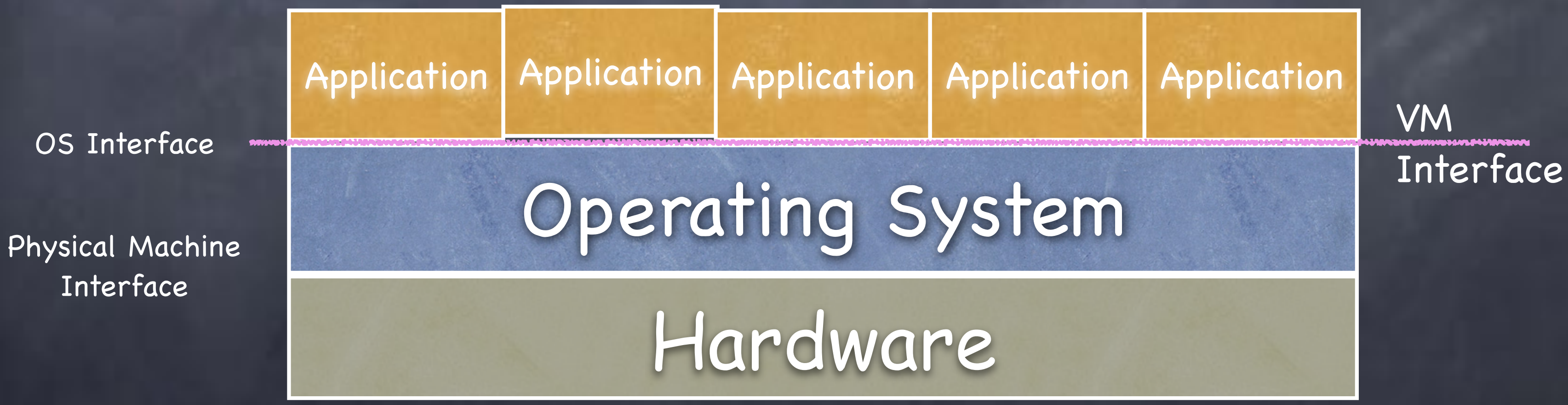
THE Operating system

Archive is online!

EWD 196, 1965

What is an OS?

- An Operating System implements a virtual machine whose interface is more convenient* that the raw hardware interface



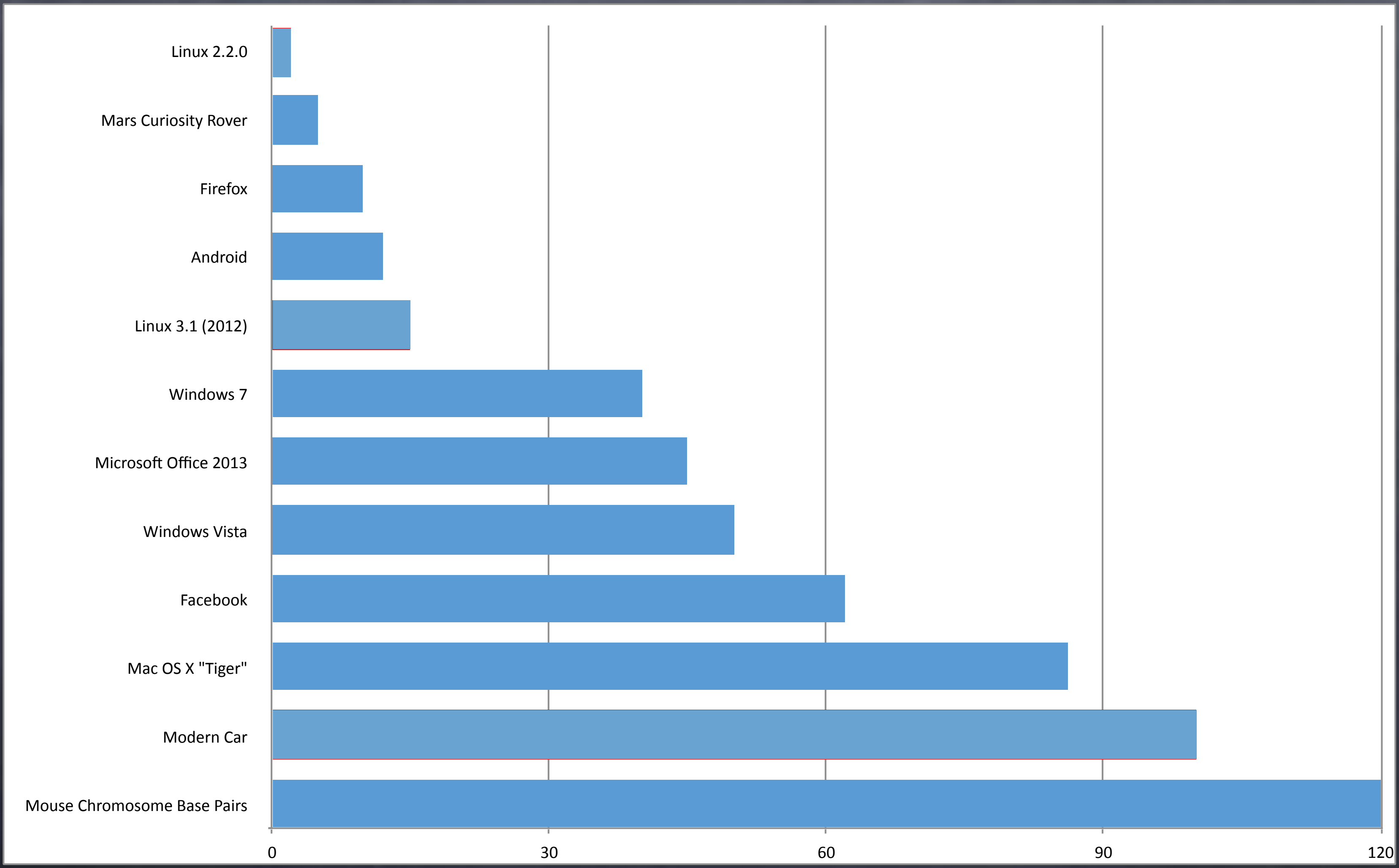
* easier to use, simpler to code, more reliable, more secure…

OS wears many hats

- Referee
  - Manages shared resources: CPU, memory, disks, networks, displays, cameras…
- Illusionist
  - Clean, easy-to-use abstractions of physical resources
  - Look! Infinite memory! Your own private processor!
- Glue

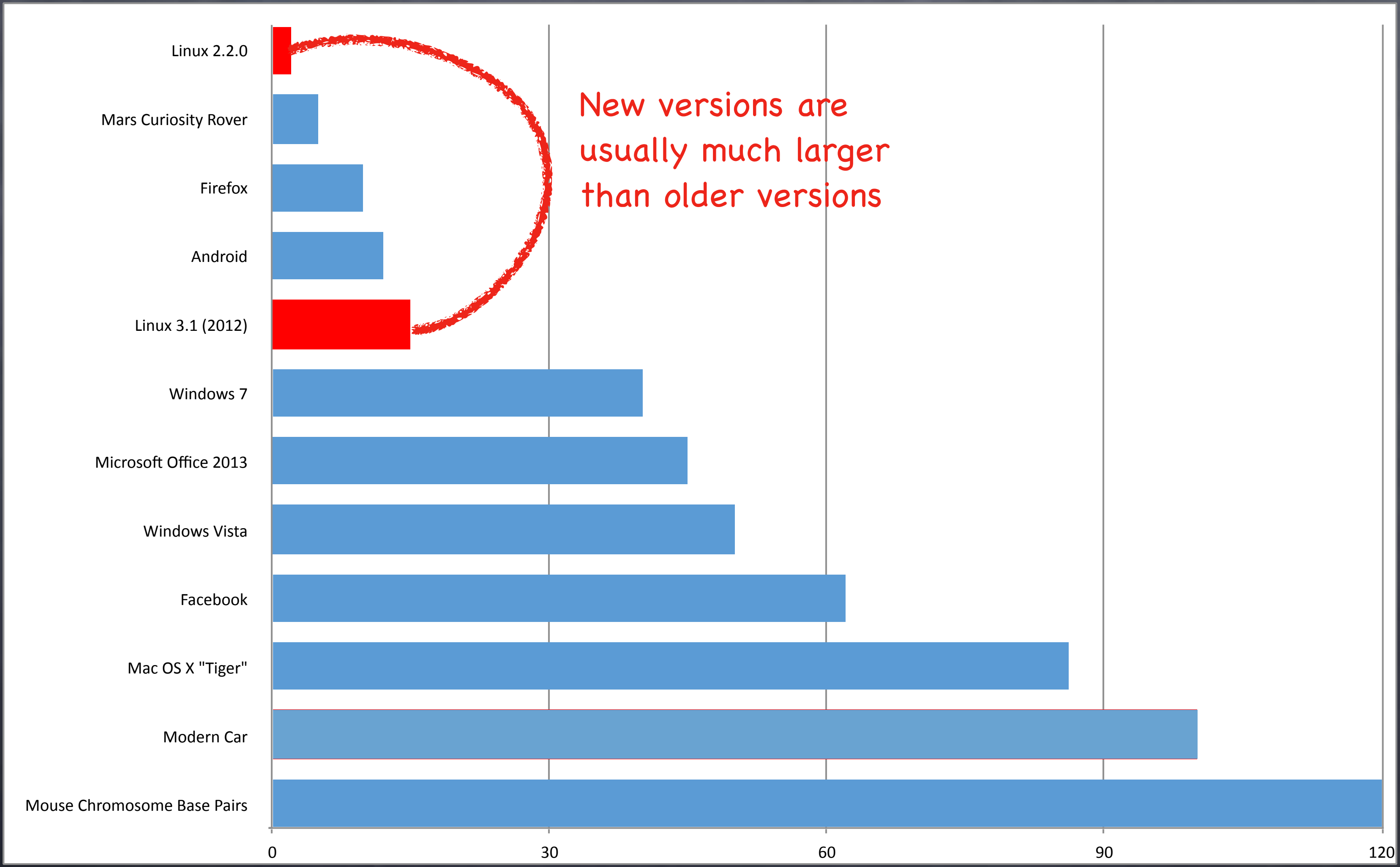
  - Offers a set of common services (e.g., UI routines)

LOC of different OSs



Millions of lines of code

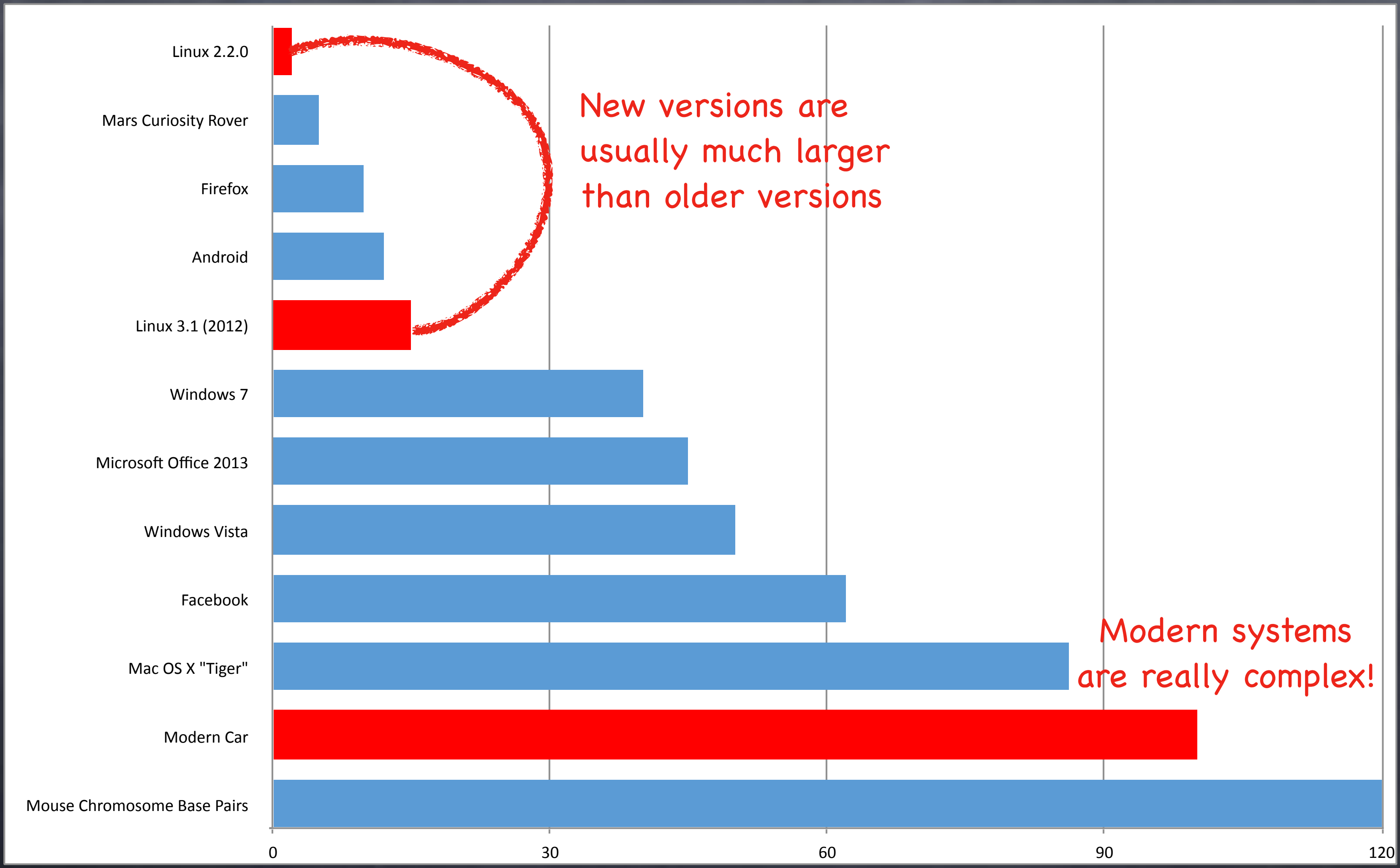
LOC of different OSs



New versions are usually much larger than older versions

Millions of lines of code

LOC of different OSs



New versions are usually much larger than older versions

Modern systems are really complex!

Millions of lines of code

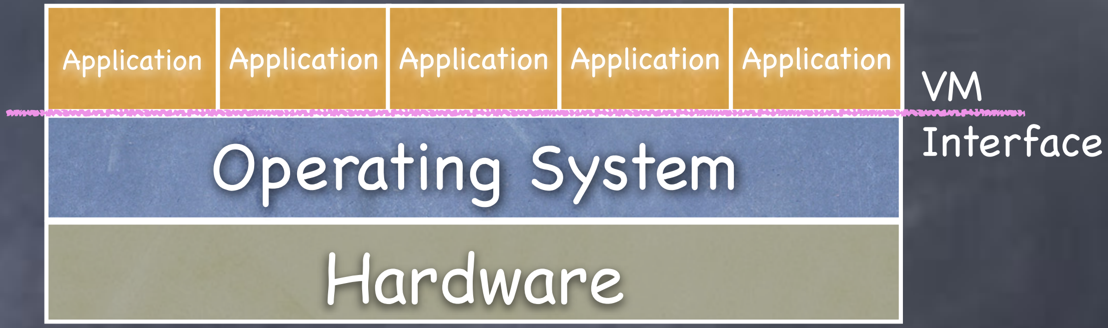
OS as Referee

- Resource allocation
  - Multiple concurrent tasks… who gets how much?
- Isolation
  - A faulty app should not disrupt other apps or OS
- Communication/Coordination
  - Apps need to coordinate and share state

OS as Illusionist

Appearance of resources not physically present

- Virtualization
  - □ Processor, memory, screen space, disk, network



Application

Application

Application

Application

Application

VM Interface

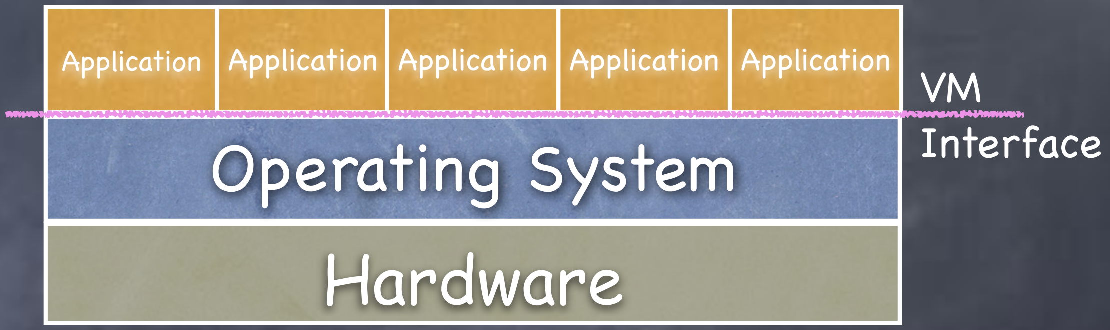
Operating System

Hardware

OS as Illusionist

Appearance of resources not physically present

- Virtualization
  - □ Processor, memory, screen space, disk, network
  - □ The entire computer
    - Fooling the OS itself!
    - Eases debugging, portability, isolation

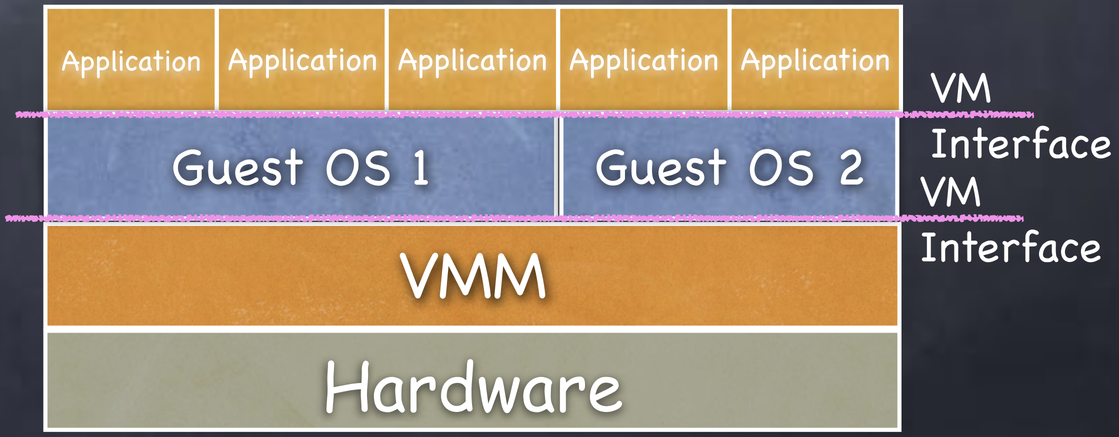


Application | Application | Application | Application | Application

VM Interface

Operating System

Hardware



Application | Application | Application | Application | Application

VM Interface

Guest OS 1 | Guest OS 2

VM Interface

VMM

Hardware

OS as Illusionist

Appearance of resources not physically present

- Atomic operations
  - HW guarantees atomicity at the word level…
    - What happens during concurrent updates to complex data structures?
    - What is a computer crashes while writing a file block?
  - At the hardware level, packets are lost
    - Reliable communication channels

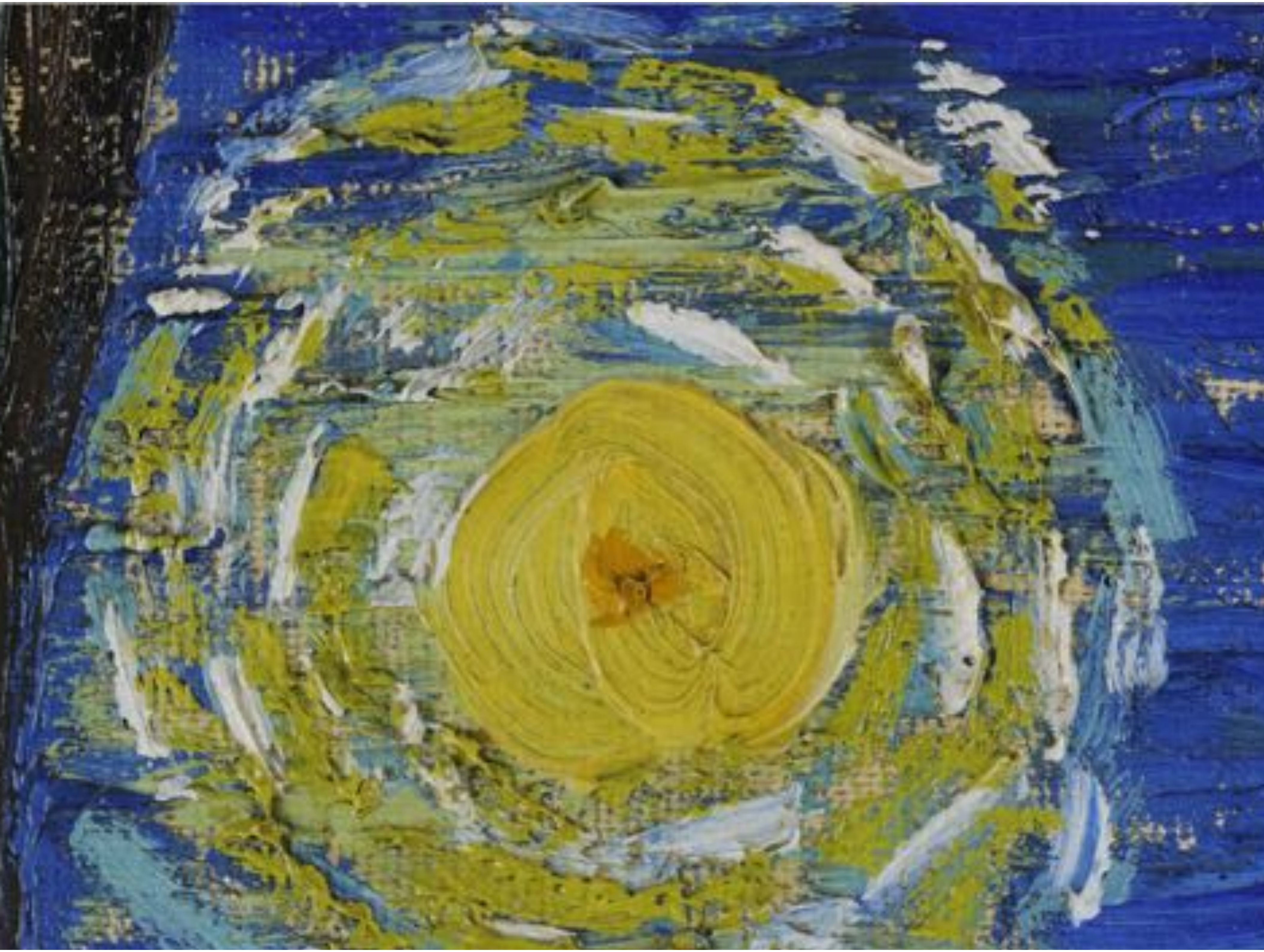
OS as Glue

- Offers standard services to simplify app design and facilitate sharing
    - Send/Receive byte streams
    - Read/Write files
    - Pass messages
    - Share memory
    - UI
- Decouples HW and app development

The Road Ahead

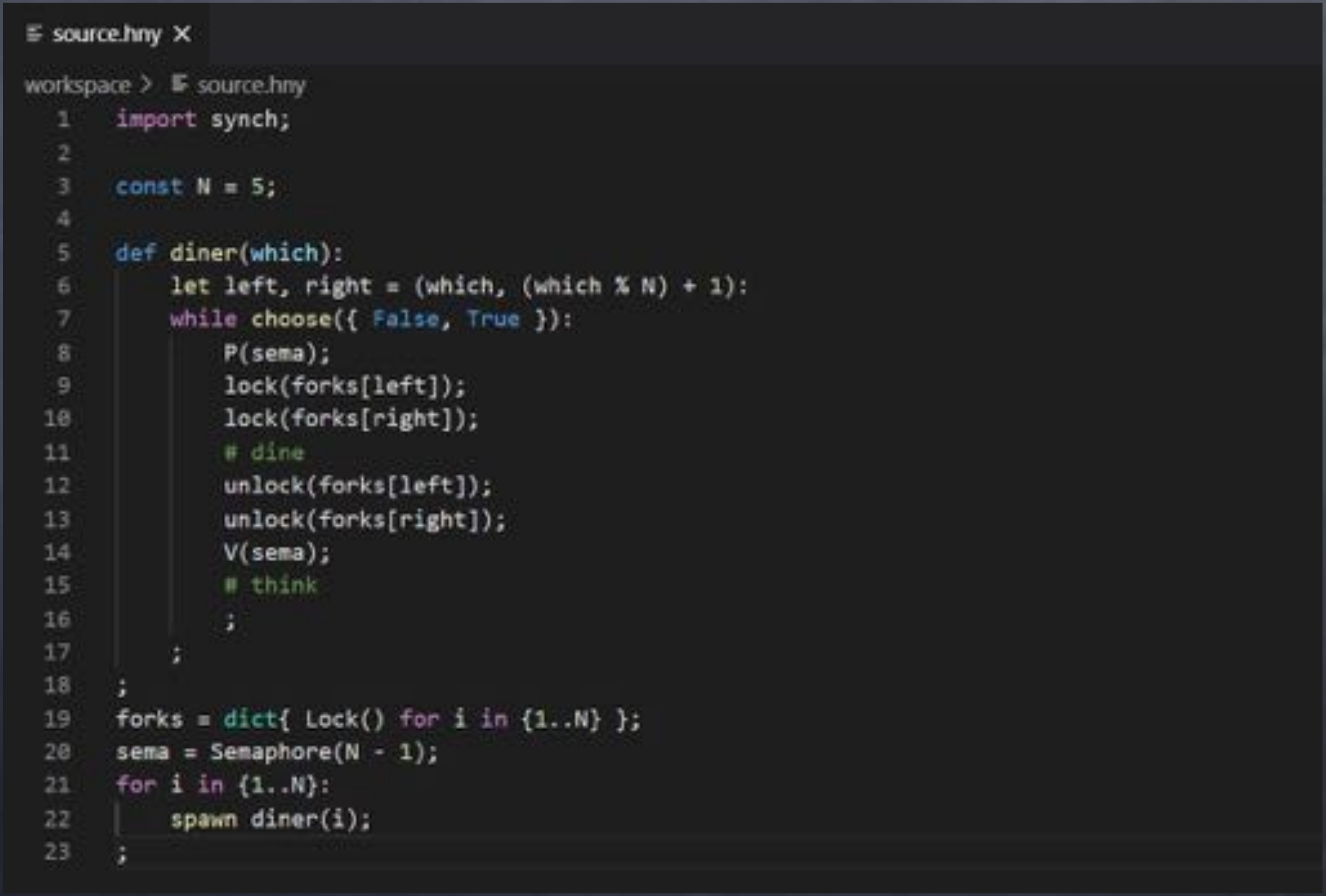
- Virtualizing the CPU
  - Process Abstraction and API
  - Threads and Concurrency
  - Scheduling
- Virtualizing Memory
  - Virtual Memory
  - Paging
- Persistence
  - I/O Devices
  - File Systems

What will the course be like?



Harmony

Your Automated Concurrency Tutor



```
import synch;

const N = 5;

def diner(which):
    let left, right = (which, (which % N) + 1):
    while choose({ False, True }):
        P(sema);
        lock(forks[left]);
        lock(forks[right]);
        # dine
        unlock(forks[left]);
        unlock(forks[right]);
        V(sema);
        # think
        ;
    ;
;
forks = dict{ Lock() for i in {1..N} };
sema = Semaphore(N - 1);
for i in {1..N}:
    spawn diner(i);
;
```

Issues in OS Design

- Structure: how is the OS organized?
- Concurrency: how are parallel activities created and controlled?
- Sharing: how are resources shared?
- Protection: how are distrusting parties protected from each other?
- Naming: how are resources named by users?
- Security: how to authenticate, authorize, and ensure privacy?
- Performance: how to make it fast?

More Issues in OS Design

- Reliability: how do we deal with failures??
- Portability: how to write once, run anywhere?
- Extensibility: how do we add new features?
- Communication: how do we exchange information?
- Scale: what happens as demands increase?
- Persistence: how do we make information outlast the processes that created it?
- Accounting: who pays the bill and how do we control resource usage?