Today

Distributed Systems +
  Distributed Objects

✓ Synthesizing Network Protocol Stacks

⇒ Spring Kernel (Lesson 6)

Friday

Java RMI + EJB
  (please watch videos)

How to innovate OS?

Brand new OS?

    OR

Better implementation of known OS?

How to innovate OS?

Brand new OS?

OR

Better implementation of known OS?

Market place needs

— Large Complex Server Software

How to innovate OS?

Brand new OS?

or

Better implementation of known OS?

Market place needs

   — Large Complex Server Software
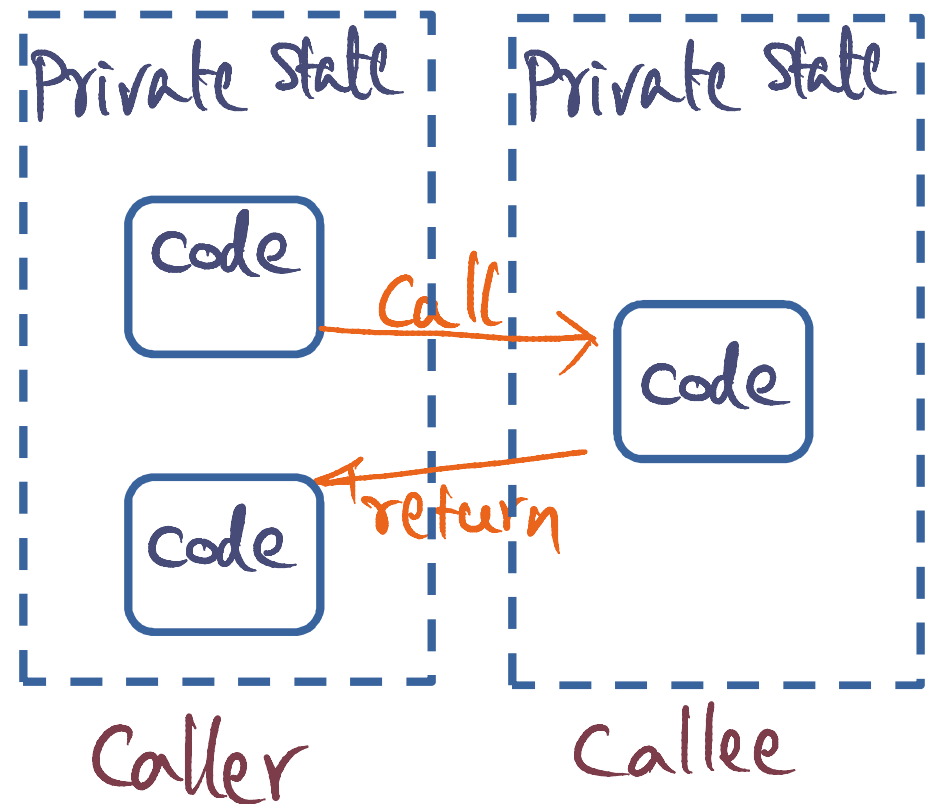
⟹ Take the Intel inside approach......
in this case Unix inside for box makers
like Sun Microsystems
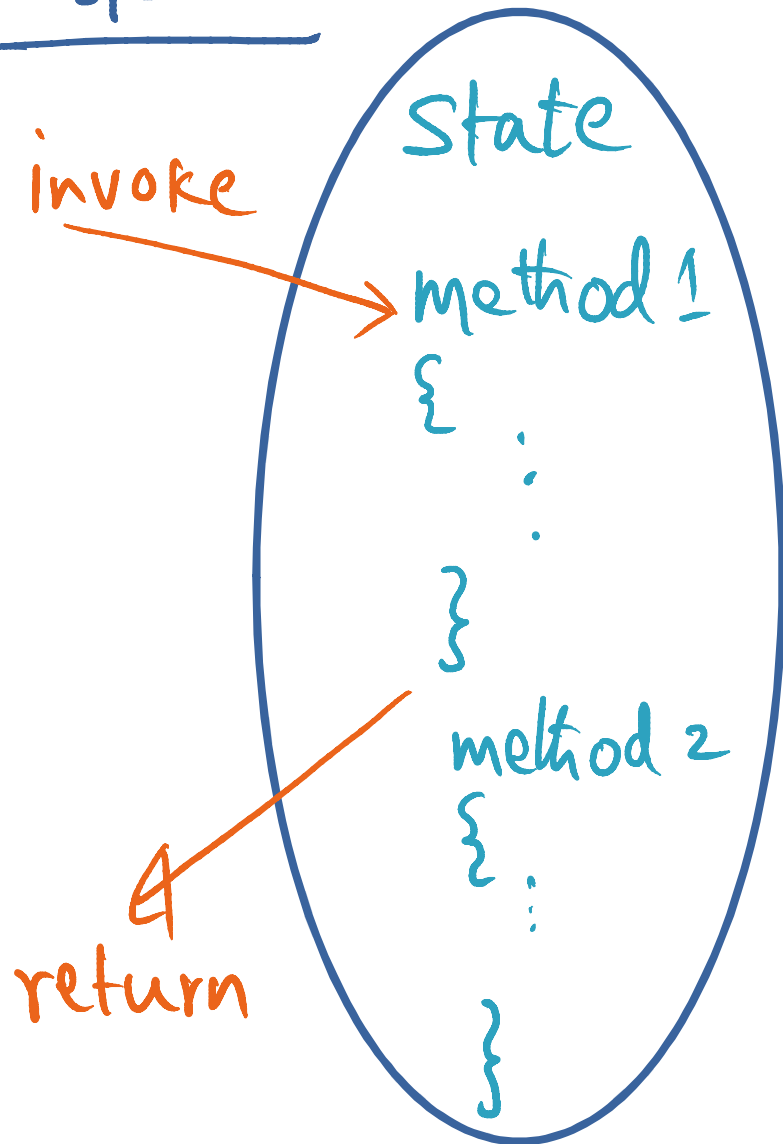
Object-based Vs. Procedural Design

Procedural

Shared state

Private State | Private State

Code

Call →

Code

Code

← *return*

Caller | Callee

Object-based Vs. Procedural Design

Objects

invoke

State

method 1
{
  .
  .
}

method 2
{
  .
  .
}

return

Procedural

Shared state

Private State

Code

Call

Code

return

Code

Caller

Private State

Callee
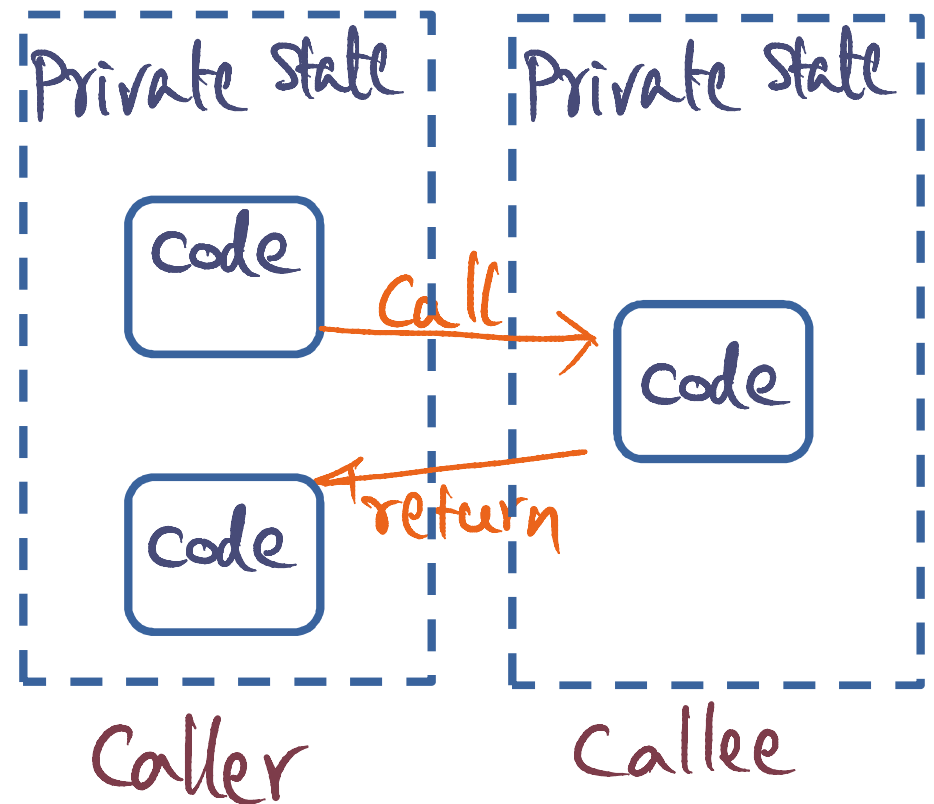
Spring Approach

Strong interfaces
open, flexible, and extensible

All outside
the kernel

		Spring Application
X11 server / libue	csh / libue	
unix process server	dynamic linker	tty server
TCP/UDP/IP	authentication manager	caching fs
network proxy	machine name server	Spring file system

Connect to other machines →

μKernel
(threads/rpc) →

nucleus   Kernel   vm manager

follows Liedtke's principle
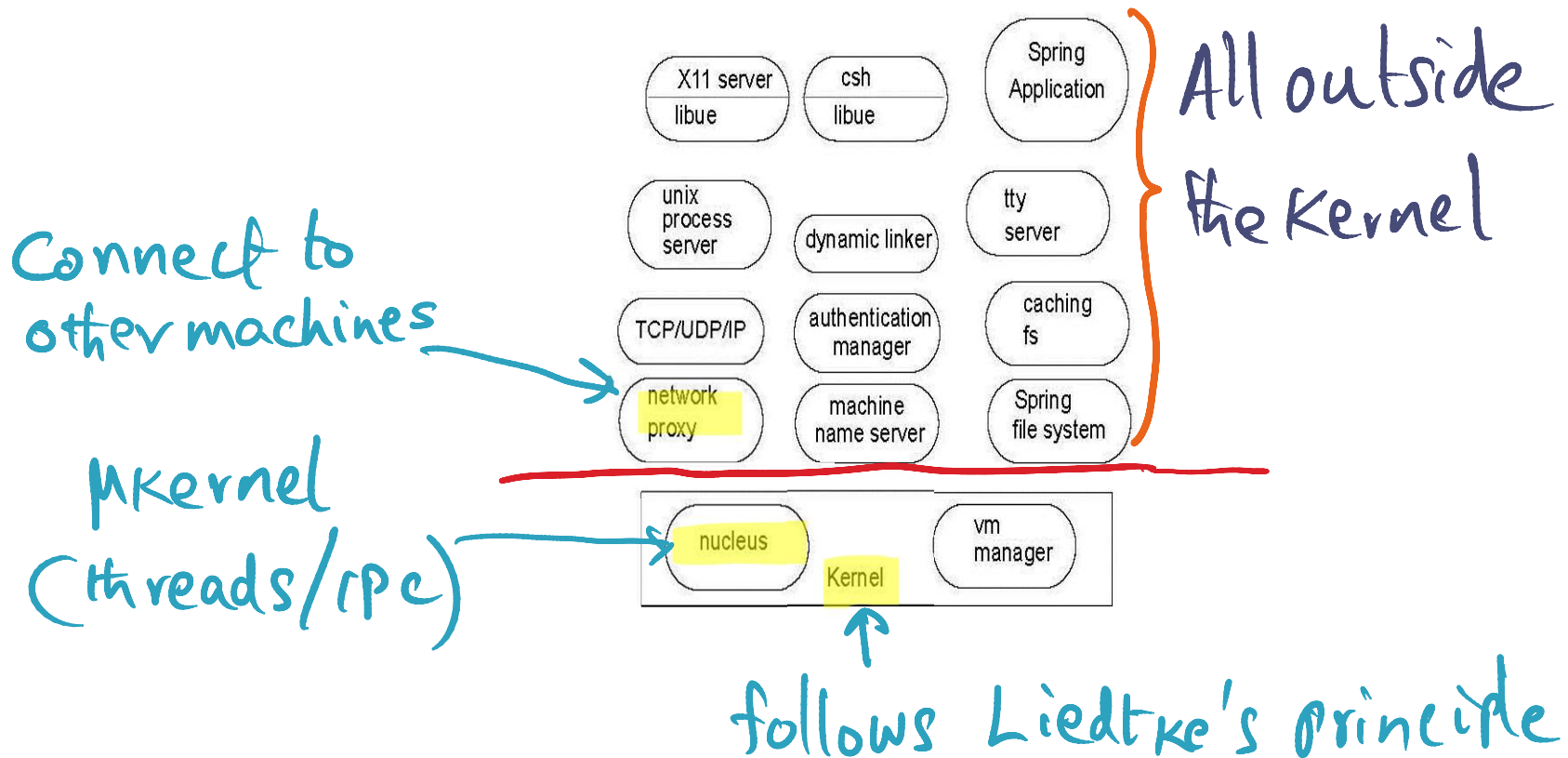
Nucleus - "micro kernel" of spring

Domain — Door Table — door IDs

Nucleus — Door — Door — Door — threads

Target Domain

Target Domain

Domain — door IDs — Door Table