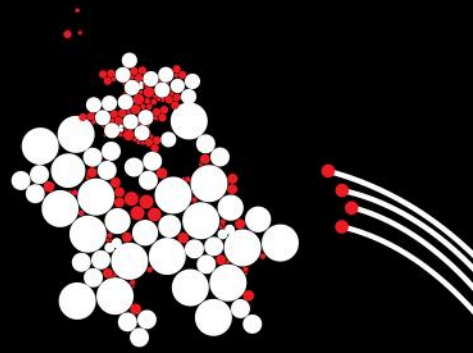
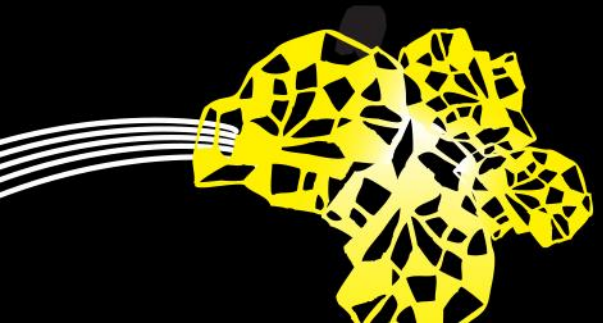
UNIVERSITY OF TWENTE.



OPERATING SYSTEMS

OVERVIEW

ERIK TEWS <e.tews@utwente.nl>



OPERATING SYSTEM

- Resource manager
  - Hardware resources
  - Resources provided by the operating system itself
- Objectives
  - Convenience
  - Efficiency
  - Evolvability

HISTORY

- Batch
- Time sharing
- Real-time
- Multi-processor
- Distributed systems
- Multi-core

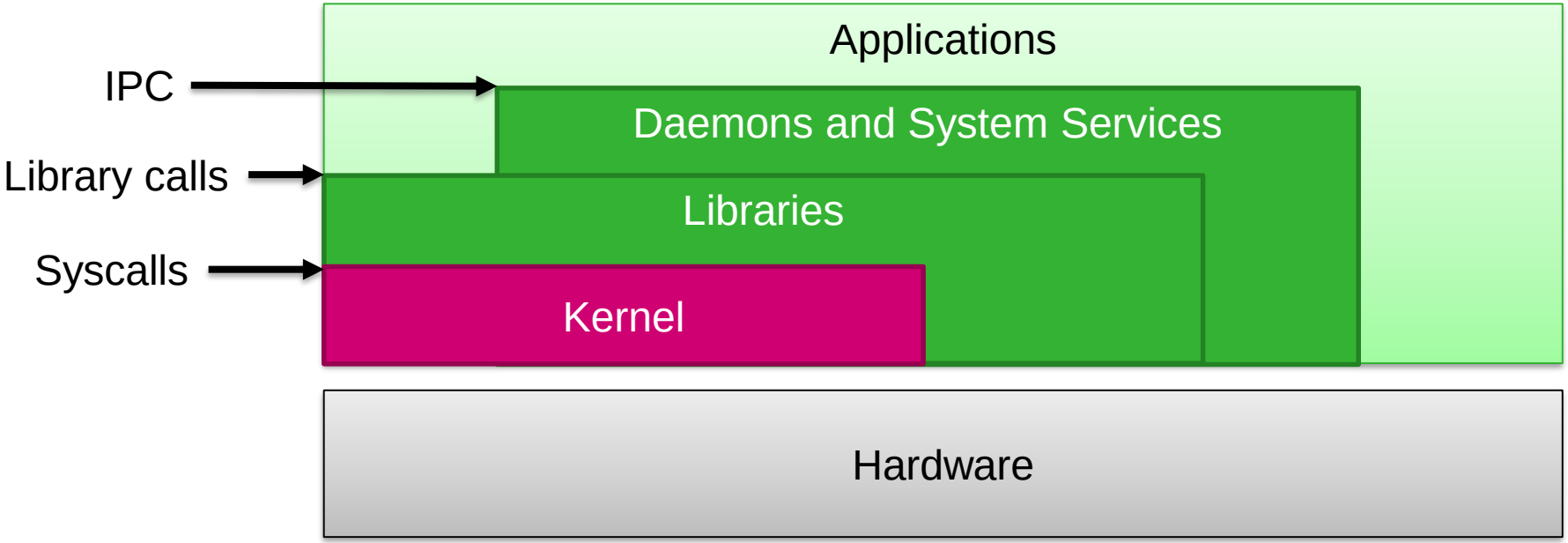
CONCEPTS AND ABSTRACTIONS

- Concept
  - Processes & Threads
  - Memory management
  - Protection and security
  - Scheduling and resource management

CONCEPTS AND ABSTRACTIONS

- Abstraction of
  - Processor
  - Physical memory & disks
  - Dedicated computer

ABSTRACTION LAYERS



DISTRIBUTIONS

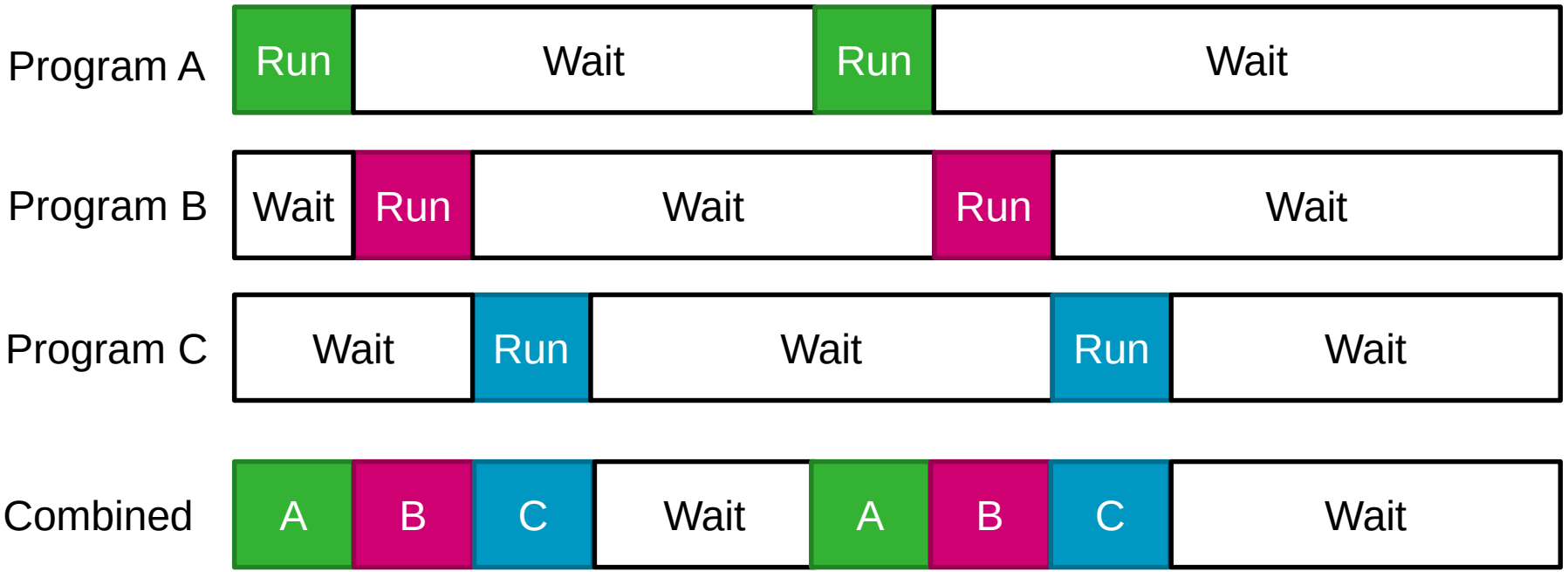
- Package a kernel with many userland applications
- Usually modify many of them
- Ship them all in an easy to install way

EXAMPLE

- Output?
- gcc Echo.c
- ./a.out Hello world
- strace ./a.out Hello World

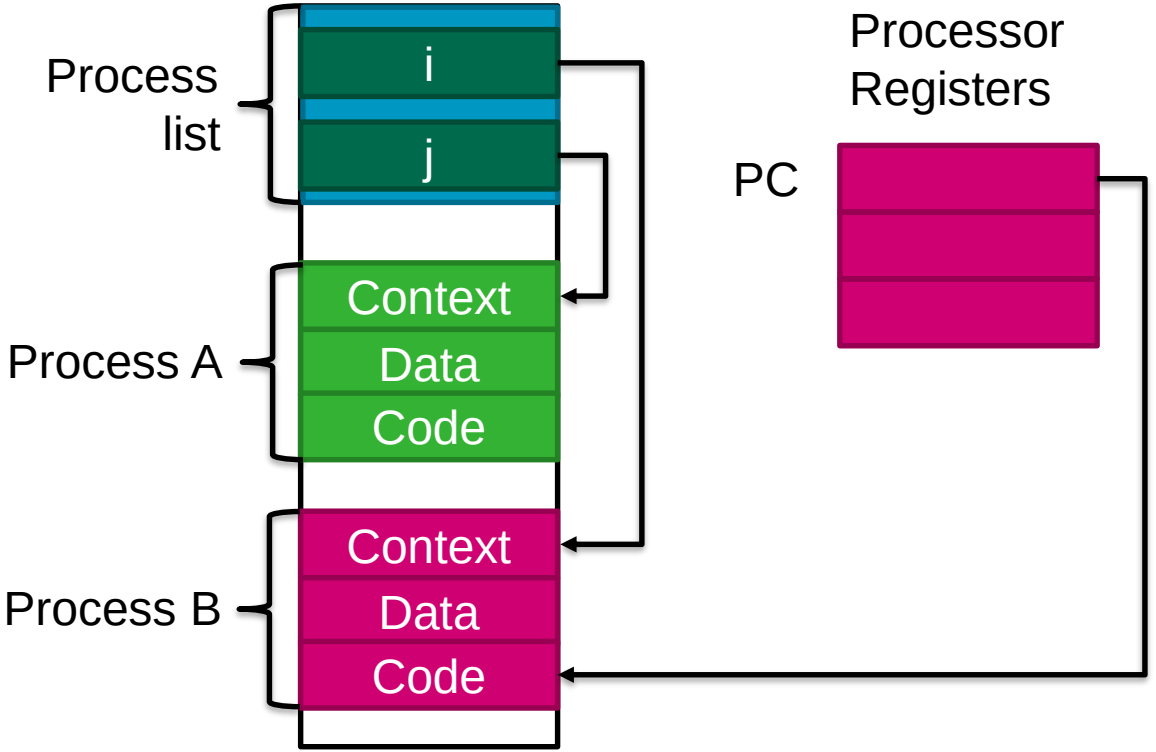
```c
#include <stdio.h>
int main(int argc, char *argv[])
{
 int i;
 for(i=0; i < argc; i++) {
  printf("%s ", argv[i]);
 }
 printf("(%d)\n", argc);
 return 0;
}
```

PROCESS ABSTRACTION

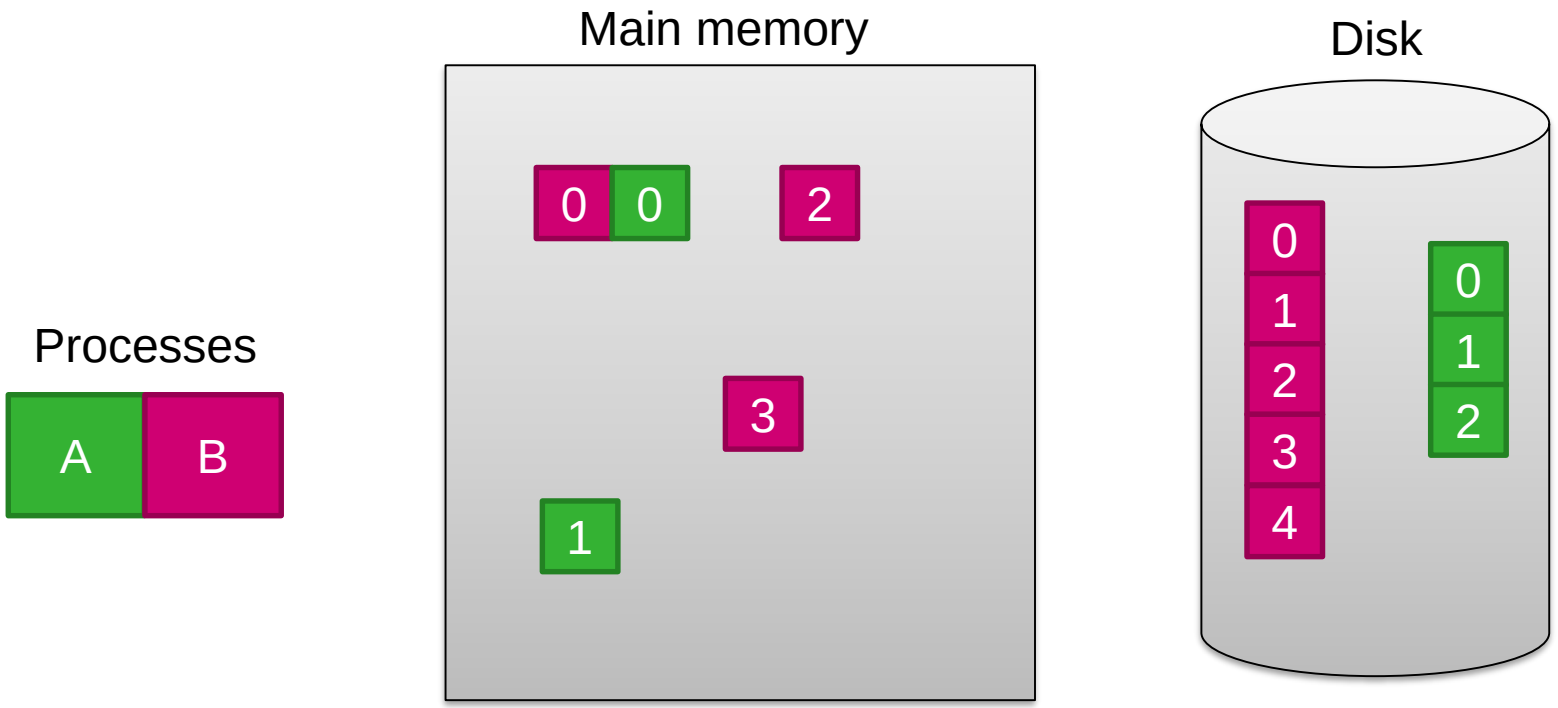


PROCESS MANAGEMENT

- Scheduling
  - Timer
- Threads (why?)



MEMORY MANAGEMENT



PROTECTION AND SECURITY

- Processes vs the OS (how?)
- Processes vs other Processes (how?)
- Privileged processes (why?)
- Access control matrix

	Files			
	1	2	3	4
User A	OWN R,W		OWN R,W	
User B	R	OWN R,W	W	R
User C	R,W	R		OWN R,W

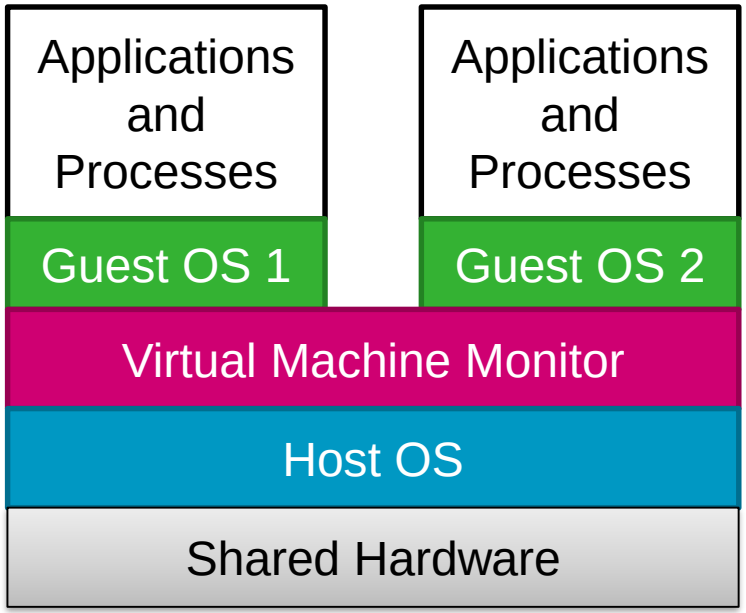
I/O MANAGEMENT

- Output?
- gcc Wrap.c
- strace ./a.out Wrap.c junk

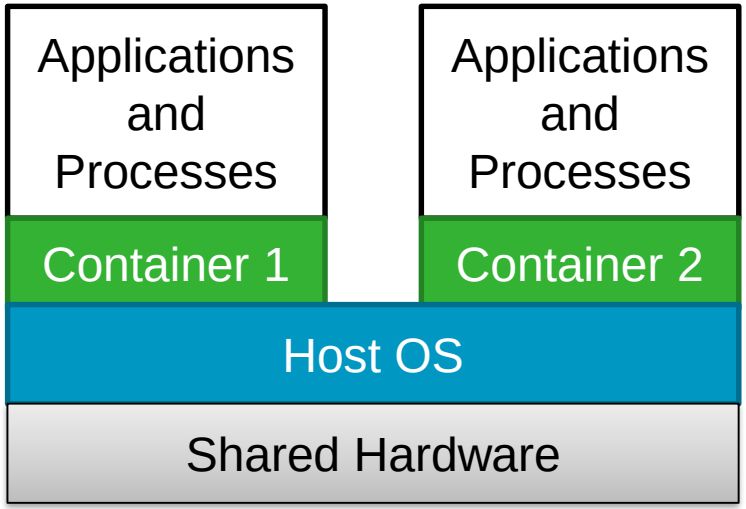
- Find the system calls for:
  - fgets()
  - fputs()

```c
#include <stdio.h>
#include <unistd.h>
#define N 41
int main(int argc, char * argv[]) {
 if(argc >= 3) {
  FILE *from = fopen(argv[1], "r");
  FILE *to = fopen(argv[2], "w");
  char buf[N];
 while (fgets(buf,N,from) != NULL) {
  fputs(buf,to);
  fputc('\n',to);
  }
  fclose(from);
  fclose(to);
  return 0;
 } else {
  printf("usage %s from to\n", argv[0]);
  return 1;
 }
}
```

VIRTUAL MACHINE MANAGEMENT



CONTAINERS


Applications and Processes
Applications and Processes
Container 1
Container 2
Host OS
Shared Hardware

SUMMARY

- Operating systems are large, hence abstraction to manage complexity:
  - Processes and threads
  - Memory
  - Files and peripherals
  - User accounts
- Management issues
  - Fairness
  - Sharing
  - Protection