CLOUD COMPUTING CONCEPTS

with Indranil Gupta (Indy)

CLOUDS ARE DISTRIBUTED SYSTEMS

Lecture B

WHAT IS A DISTRIBUTED SYSTEM?

So What IS a "Distributed System"?

- Let's try to define the term!

Wait, Wait…

- Let’s first take a step back
- And define the term “Operating System”
- All machines (computers, devices, etc.) run an Operating System
  - Also called an “OS”
- If you’ve used a computer, you’ve used an Operating System

Can you name some examples of Operating Systems?

Can you name some examples of Operating Systems?

- For big devices:
  - Linux
  - Mac OS X (Yosemite, Mavericks, …)
  - Windows (10, 8, 7, Vista, XP, …)
  - Unix
  - FreeBSD …
- For small devices:
  - Android
  - iOS
  - TinyOS …

What is an Operating System?

What is an Operating System?

- User interface to hardware (device drivers)
- Provides abstractions (processes, file system)
- Resource manager (scheduler)
- Means of communication (networking)
- …

FOLDOC definition (Free Online Dictionary of Computing)

- The low-level software which handles the interface to peripheral hardware, schedules tasks, allocates storage, and presents a default interface to the user when no application program is running.
- The OS may be split into a kernel which is always present and various system programs which use facilities provided by the kernel to perform higher-level house-keeping tasks, often acting as servers in a client-server relationship.
- Some would include a graphical user interface and window system as part of the OS, others would not. The operating system loader, BIOS, or other firmware required at boot time or when installing the operating system would generally not be considered part of the operating system, though this distinction is unclear in the case of a roamable operating system such as RISC OS.
- The facilities an operating system provides and its general design philosophy exert an extremely strong influence on programming style and on the technical cultures that grow up around the machines on which it runs.

Let's Repeat the Same Exercise for the term "Distributed System"

Can you name some examples of Distributed Systems?

Can you name some examples of Distributed Systems?

- Client communicating with a server
- BitTorrent (peer to peer overlay)
- The Internet
- The Web (servers and clients)
- Hadoop
- Datacenters

NOT Distributed Systems

- The following are NOT distributed systems
  - Humans Interacting with each other
  - A standalone machine not connected to the network, and with only one process running on it

Is there a Good Definition out there?

Definition from FOLDOC (Free Online Dictionary of Computing)

A collection of (probably heterogeneous) automata whose distribution is transparent to the user so that the system appears as one local machine. This is in contrast to a network, where the user is aware that there are several machines, and their location, storage replication, load balancing and functionality is not transparent. Distributed systems usually use some kind of client-server organization.

(Definition last updated in 1994)

The FOLDOC Definition for Distributed System is Incorrect

A collection of (probably heterogeneous) automata whose distribution is transparent to the user so that the system appears as one local machine. This is in contrast to a network, where the user is aware that there are several machines, and their location, storage replication, load balancing and functionality is not transparent. Distributed systems usually use some kind of client-server organization.

- *However, it's never the case that all sites on the Web are all up or all down, e.g., site A may be up while B down; later B is up while A is down. Yet, the Web is a distributed system!*
- *Peer to peer systems like BitTorrent rely largely on only clients, no servers.*

Definitions from Textbooks

- A distributed system is a collection of independent computers that appear to the users of the system as a single computer.

  [Andrew Tanenbaum]

- A distributed system is several computers doing something together. Thus, a distributed system has three primary characteristics: multiple computers, interconnections, and shared state.

  [Michael Schroeder]

Definitions Look Unsatisfactory to Us

- Why are these definitions short?
- Why do these definitions look inadequate to us?
- Because we are interested in the insides of a distributed system
  - algorithmics
  - design and implementation
  - maintenance
  - study

Easy to Know when you See it

- The following are NOT distributed systems
  - Humans Interacting with each other
  - A standalone machine not connected to the network, and with only one process running on it

Easy to Know when you See it

I shall not today attempt further to define the kinds of material I understand to be embraced within that shorthand description; and perhaps I could never succeed in intelligibly doing so. But I know it when I see it…

[Potter Stewart, Associate Justice, US Supreme Court (talking about his interpretation of a technical term laid down in the law, case Jacobellis versus Ohio 1964)]

A Working Definition of "Distributed System"

A distributed system is a collection of entities, each of which is autonomous, programmable, asynchronous and failure-prone, and which communicate through an unreliable communication medium.

A Working Definition of "Distributed System"

A distributed system is a collection of entities, each of which is autonomous, programmable, asynchronous and failure-prone, and which communicate through an unreliable communication medium.

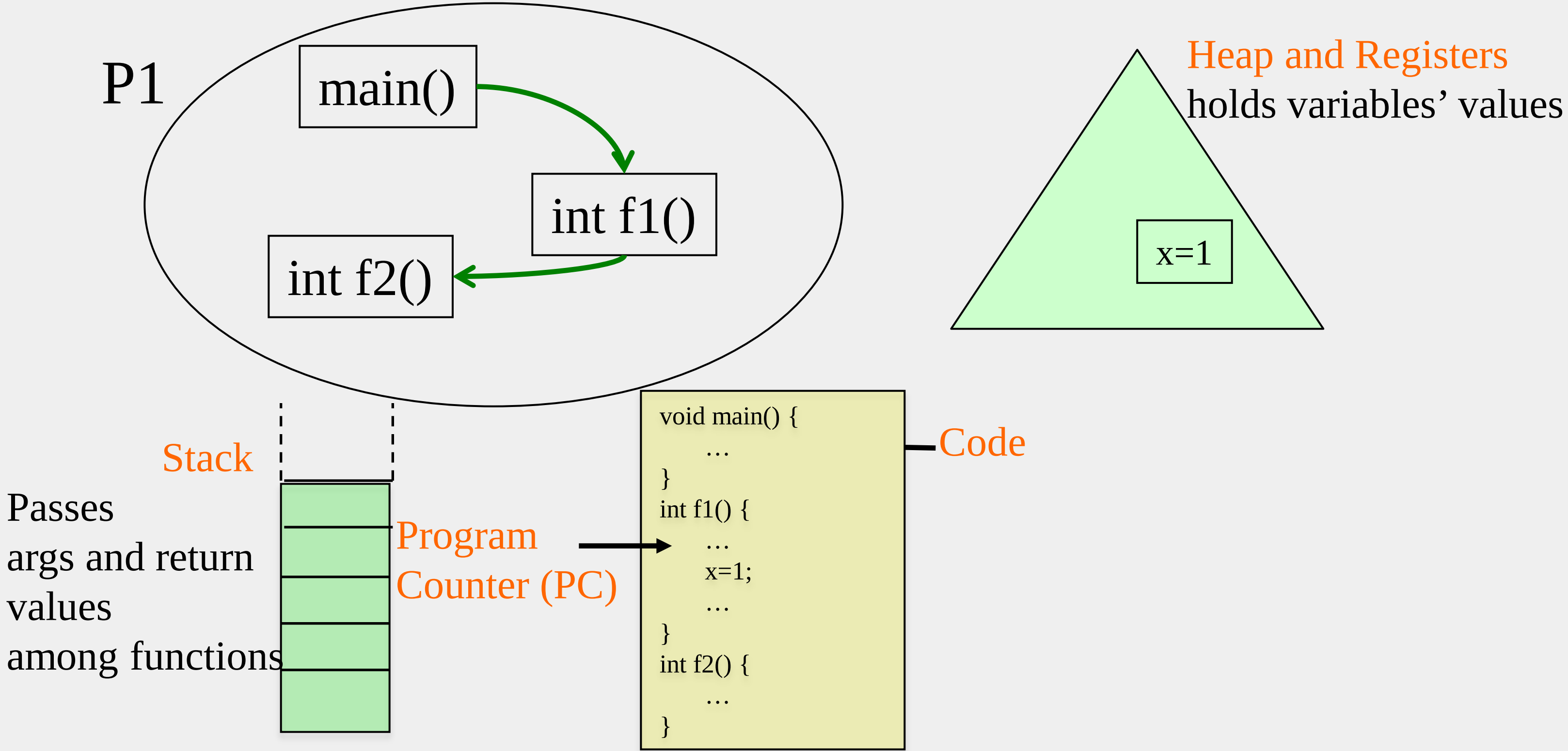
- Our interest in distributed systems involves
  - algorithmics, design and implementation, maintenance, study
- Entity=a process on a device (PC, PDA, mote)
- Communication Medium=Wired or wireless network

A Working Definition of "Distributed System"

A distributed system is a collection of entities, each of which is autonomous, programmable, asynchronous and failure-prone, and which communicate through an unreliable communication medium.

- Eliminates "Humans Interacting with each other"
- Distinguishes distributed systems from parallel systems (e.g., multiprocessor systems)

Remember a Process? (Orientation Lecture)



P1

main()

int f1()

int f2()

Heap and Registers
holds variables' values

x=1

Stack

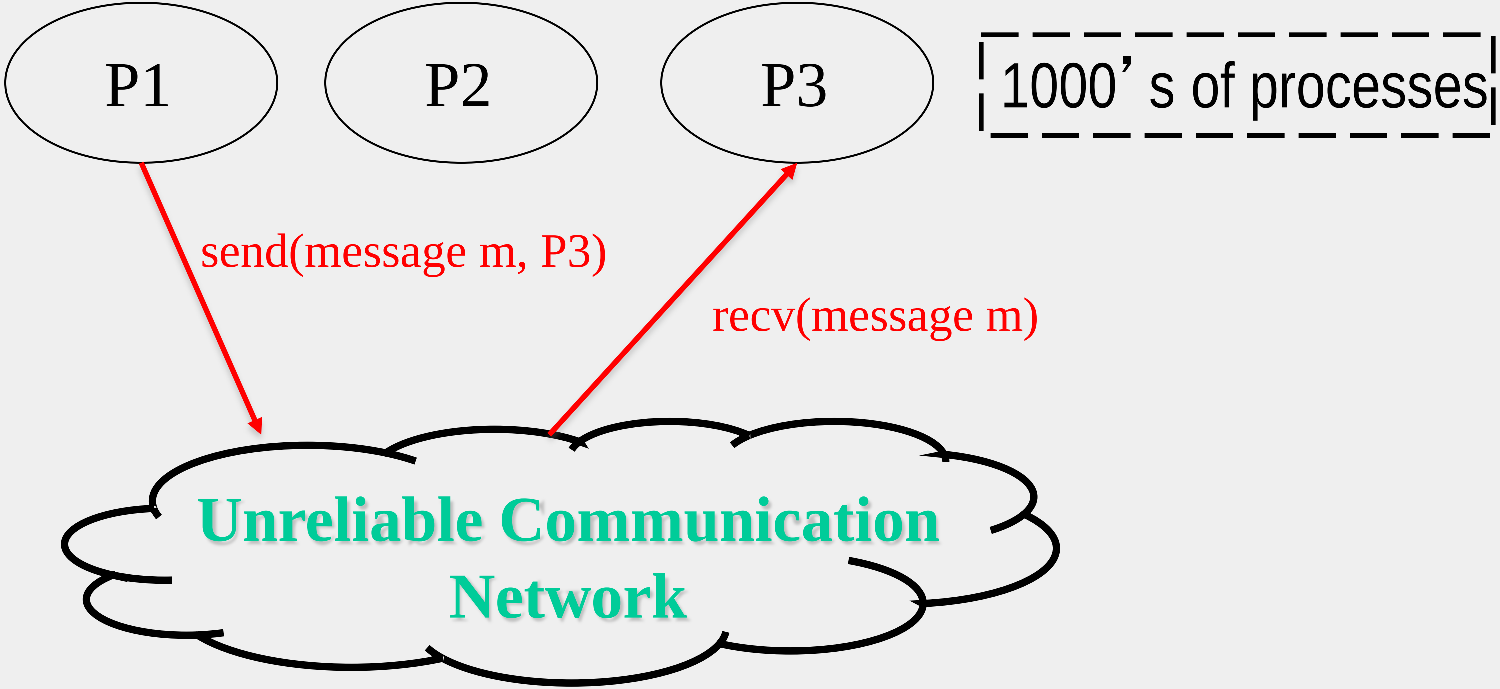
Passes
args and return
values
among functions

Program
Counter (PC)

Code

```
void main() {
      …
}
int f1() {
      …
      x=1;
      …
}
int f2() {
      …
}
```

Distributed System = Many Processes Sending and Receiving Messages



Not Meant to be a Perfect Definition

- That’s only a working definition, good for this course
- Feel free to come up with your own definition for distributed systems!
- Try the exercise after you’ve seen the many different examples of distributed systems in this course!

A range of interesting problems for Distributed System designers

-
- P2P systems [Gnutella, Kazaa, BitTorrent]
- Cloud Infrastructures [AWS, Azure, Google Cloud]
- Cloud Storage [Key-value stores, NoSQL, Cassandra]
- Cloud Programming [MapReduce, Storm, Pregel]
- Coordination [Paxos, Leader Election, Snapshots]
- Managing Many Clients and Servers Concurrently [Concurrency Control, Replication Control]
- (and many more that you’ll see in this course!)

In solving these problems, Many Challenges Abound…

- Failures: no longer the exception, but rather a norm
- Scalability: 1000s of machines, Terabytes of data
- Asynchrony: clock skew and clock drift
- Concurrency: 1000s of machines interacting with each other accessing the same data
- …

Looking Forward

- Over the next few weeks, we will see several core concepts of distributed systems
  - Gossip
  - Membership
  - Distributed Hash Tables (DHTs)
- … and alternate this with their use in real systems
  - Peer-to-peer systems (which use DHTs)
  - Key-value/NoSQL stores (which use DHTs, gossip, membership)