Distributed Computing I

CSSE6400

Brae Webb

March 21, 2022

Mathias Verras
@mathiasverraes

There are only two hard problems in distributed systems: 2. Exactly-once delivery 1. Guaranteed order of messages 2. Exactly-once delivery

Focus

Reliability

Question

What makes software *unreliable*?

‘Working’ software

Satisfies the functional requirements

Definition 1. Reliable Software

Continues to work correctly, even when things go wrong.

Definition 2. Fault

Something goes wrong.

Death, taxes, and computer system failure are all inevitable to some degree.
Plan for the event.

- Howard and LeBlanc

Reliable software is

Fault tolerant

Problem

Individual computers fail *all the time*

Solution

Spread the risk of faults over *multiple computers*

Spreading Risk

If you have software that works with *just one* computer, spreading the software over *two* computers *halves* the risk that your software will fail.

Spreading Risk

If you have software that works with *just one* computer, spreading the software over *two* computers *halves* the risk that your software will fail.

Adding *10* computers reduces the cuts the risk by *10*.

Spreading Risk

If you have software that works with *just one* computer, spreading the software over *two* computers *halves* the risk that your software will fail.

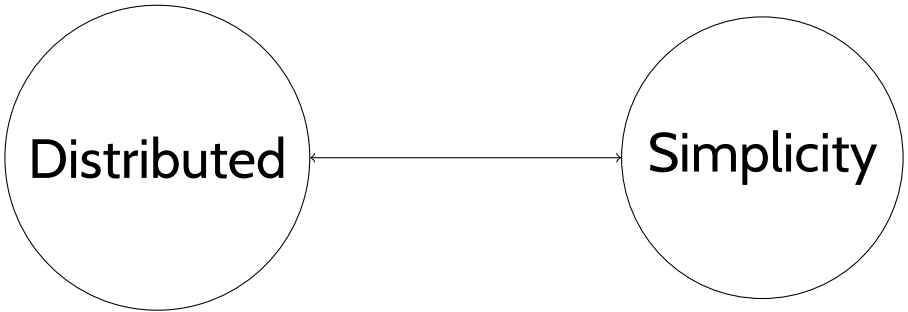
Adding *10* computers reduces the cuts the risk by *10*.

Of course, there are other reasons you might want run software on multiple computers.

Definition 3. Distributed Computing

Multiple software components that are on multiple computers, but run as a single system

The Problem



A lot of modern software development focuses on dealing with the *complexity* of distributed systems.

Question

What makes distributed computing complex?

- Faults
- Asynchronous communication
- Monitoring
- And much more…