Distributed Computing I

CSSE6400

Brae Webb

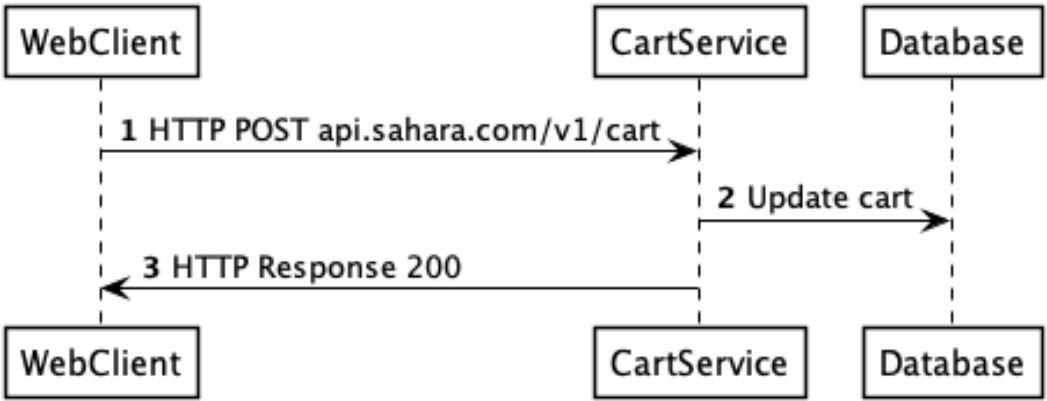
March 28, 2022

Mathias Verras
@mathiasverraes

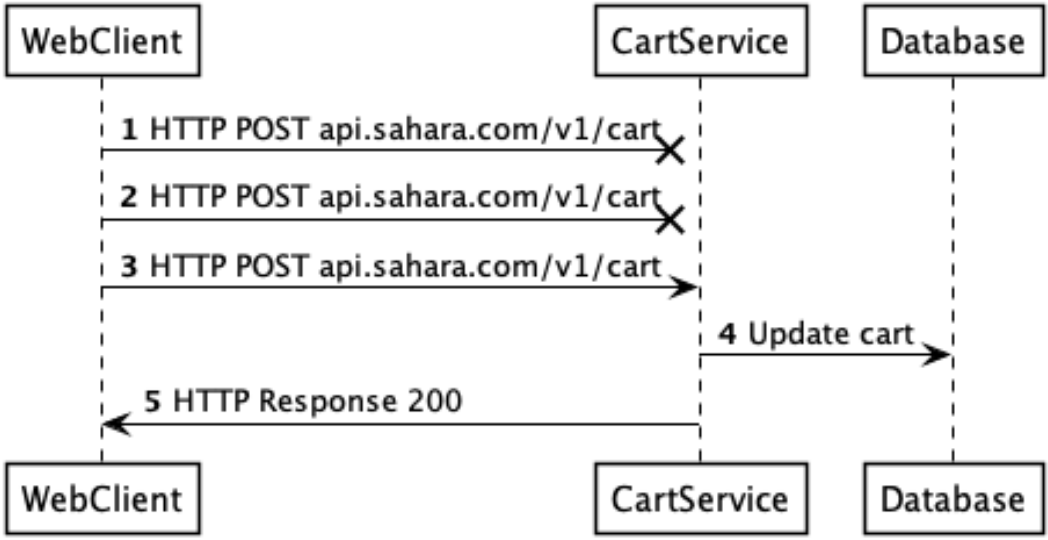
There are only two hard problems in distributed systems: 2. Exactly-once delivery 1. Guaranteed order of messages 2. Exactly-once delivery

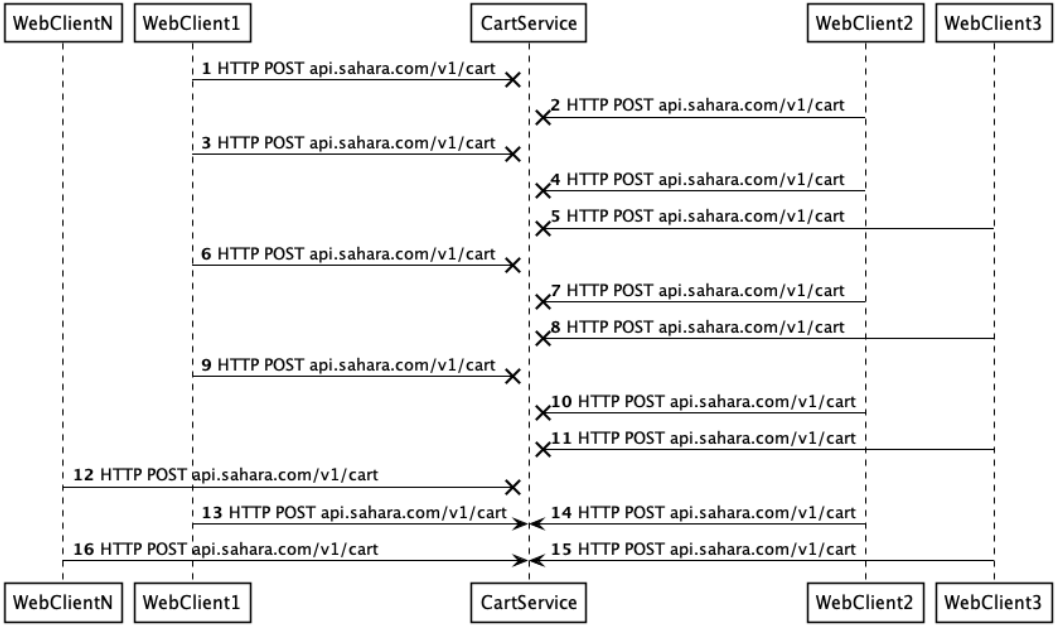
Fallacy #1

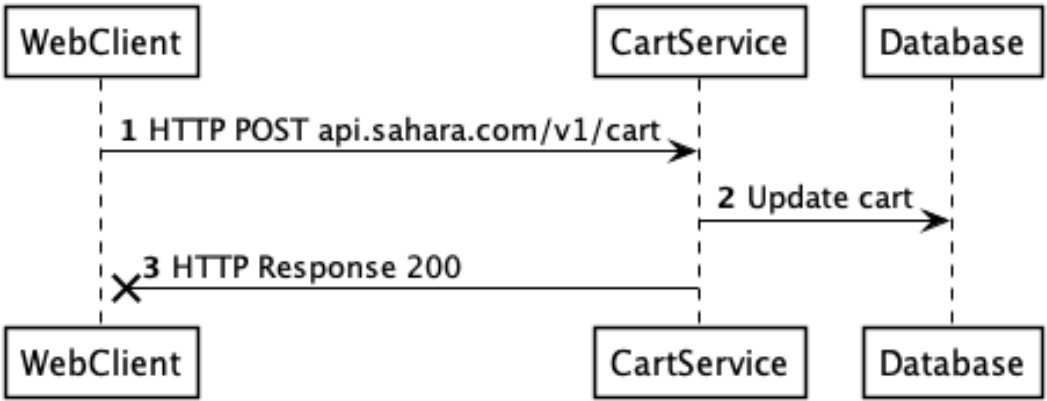
The network is reliable

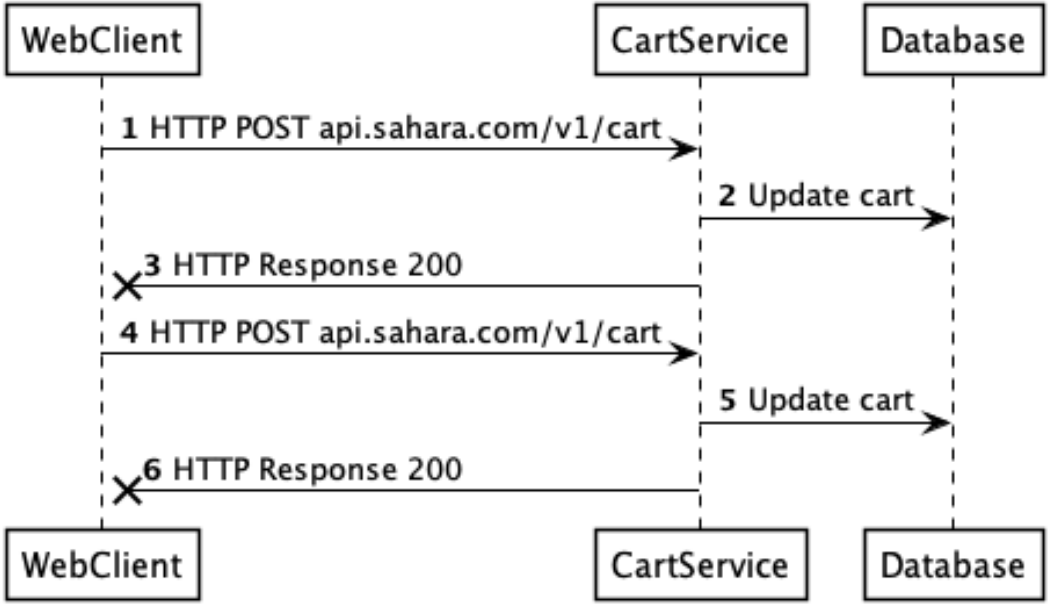

WebClient
CartService
Database
1 HTTP POST api.sahara.com/v1/cart
2 Update cart
3 HTTP Response 200
WebClient
CartService
Database

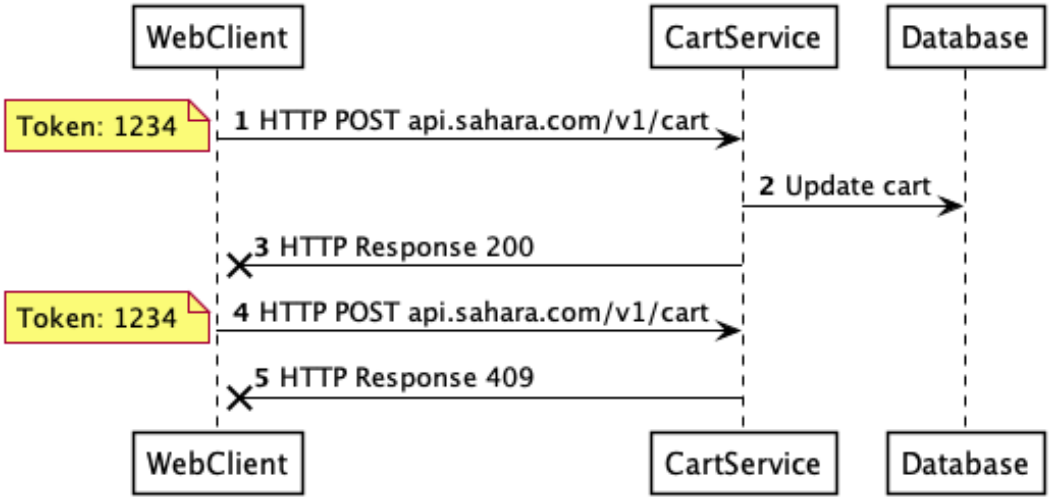

WebClient
CartService
1 HTTP POST api.sahara.com/v1/cart
WebClient
CartService


WebClient
CartService
Database
1 HTTP POST api.sahara.com/v1/cart
2 HTTP POST api.sahara.com/v1/cart
3 HTTP POST api.sahara.com/v1/cart
4 Update cart
5 HTTP Response 200
WebClient
CartService
Database


WebClientN
WebClient1
CartService
WebClient2
WebClient3
1 HTTP POST api.sahara.com/v1/cart
2 HTTP POST api.sahara.com/v1/cart
3 HTTP POST api.sahara.com/v1/cart
4 HTTP POST api.sahara.com/v1/cart
5 HTTP POST api.sahara.com/v1/cart
6 HTTP POST api.sahara.com/v1/cart
7 HTTP POST api.sahara.com/v1/cart
8 HTTP POST api.sahara.com/v1/cart
9 HTTP POST api.sahara.com/v1/cart
10 HTTP POST api.sahara.com/v1/cart
11 HTTP POST api.sahara.com/v1/cart
12 HTTP POST api.sahara.com/v1/cart
13 HTTP POST api.sahara.com/v1/cart
14 HTTP POST api.sahara.com/v1/cart
15 HTTP POST api.sahara.com/v1/cart
16 HTTP POST api.sahara.com/v1/cart
WebClientN
WebClient1
CartService
WebClient2
WebClient3


WebClient
CartService
Database
1 HTTP POST api.sahara.com/v1/cart
2 Update cart
3 HTTP Response 200
WebClient
CartService
Database


WebClient
CartService
Database
1 HTTP POST api.sahara.com/v1/cart
2 Update cart
3 HTTP Response 200
4 HTTP POST api.sahara.com/v1/cart
5 Update cart
6 HTTP Response 200
WebClient
CartService
Database


WebClient
CartService
Database
Token: 1234
1 HTTP POST api.sahara.com/v1/cart
2 Update cart
3 HTTP Response 200
Token: 1234
4 HTTP POST api.sahara.com/v1/cart
5 HTTP Response 409
WebClient
CartService
Database

Fallacy #2

Latency is zero

Fallacy #3

Bandwidth is infinite

Fallacy #4

The network is secure

Fallacy #5

The topology never changes

Fallacy #6

There is only one administrator

Fallacy #7

Transport cost is zero

Fallacy #8

The network is homogeneous

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.

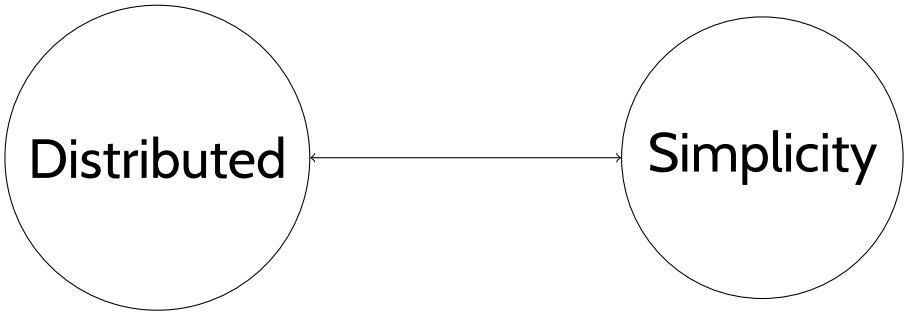
Adding *10* computers reduces the cuts the risk by *10*.

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…