How to sleep at night while having a cloud service.

Common Architecture Do's and Don'ts

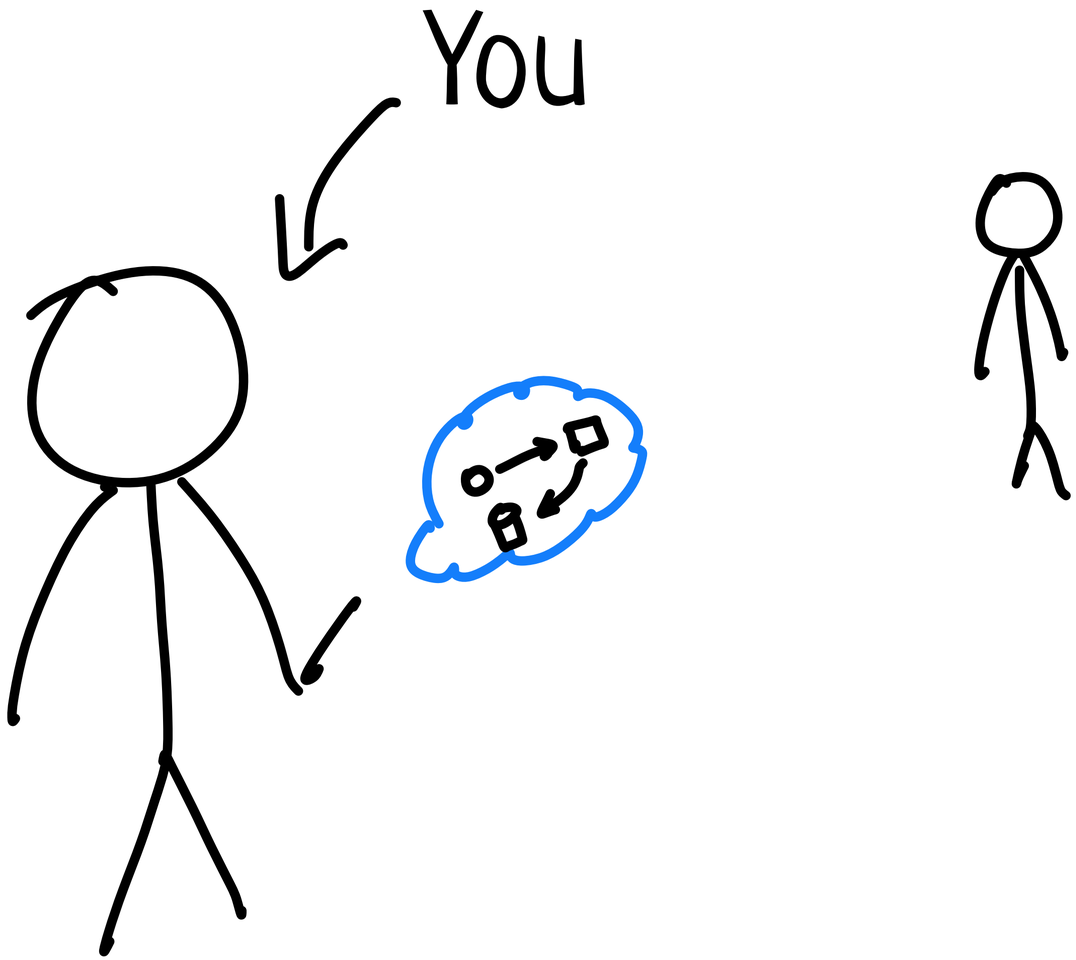
DANIEL SADA

Developer Productivity & Culture

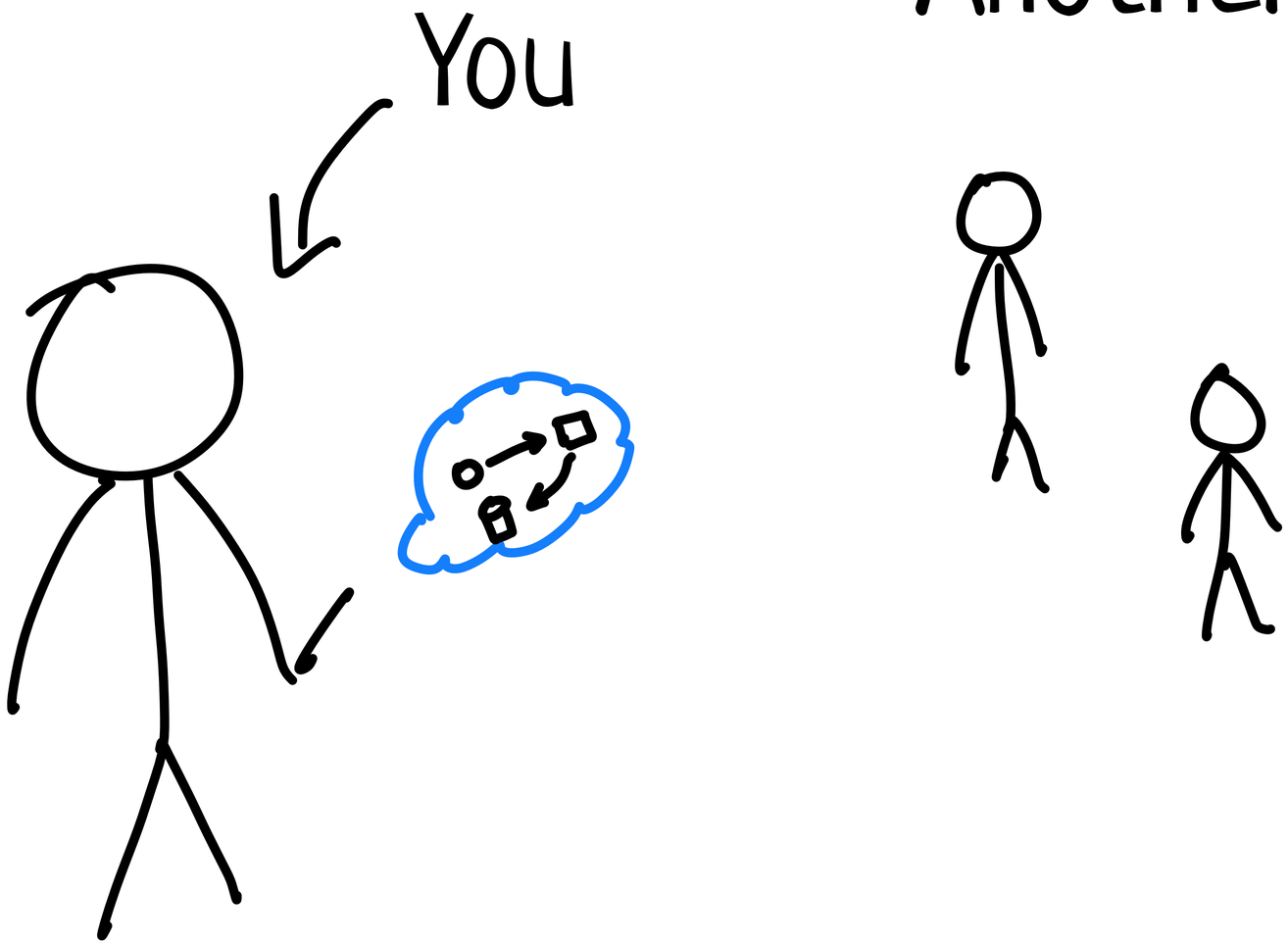
danielsada.tech /in/danielsadac @dsada@infosec.exchange

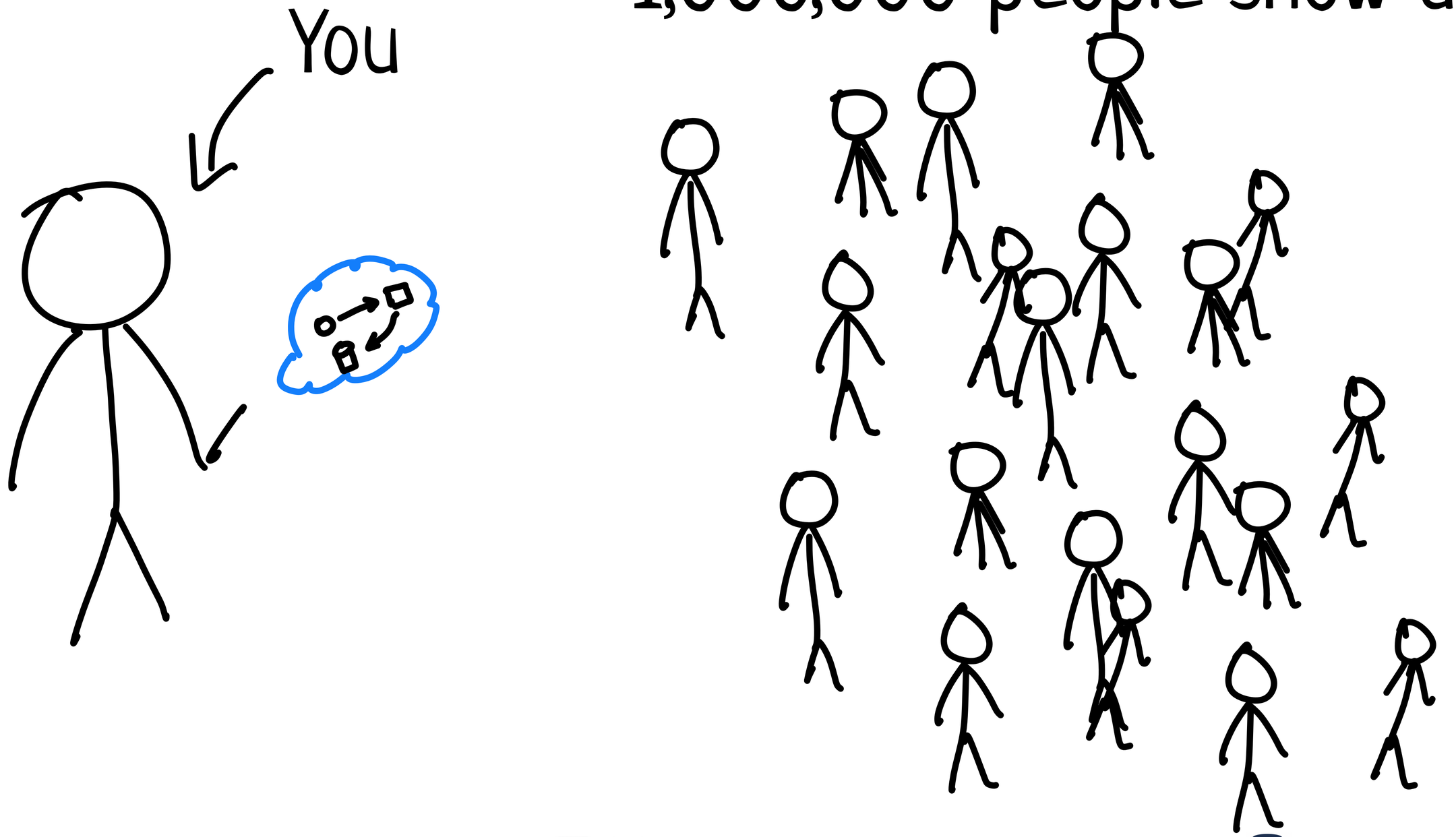
Want to deploy a service.

Someone shows up.

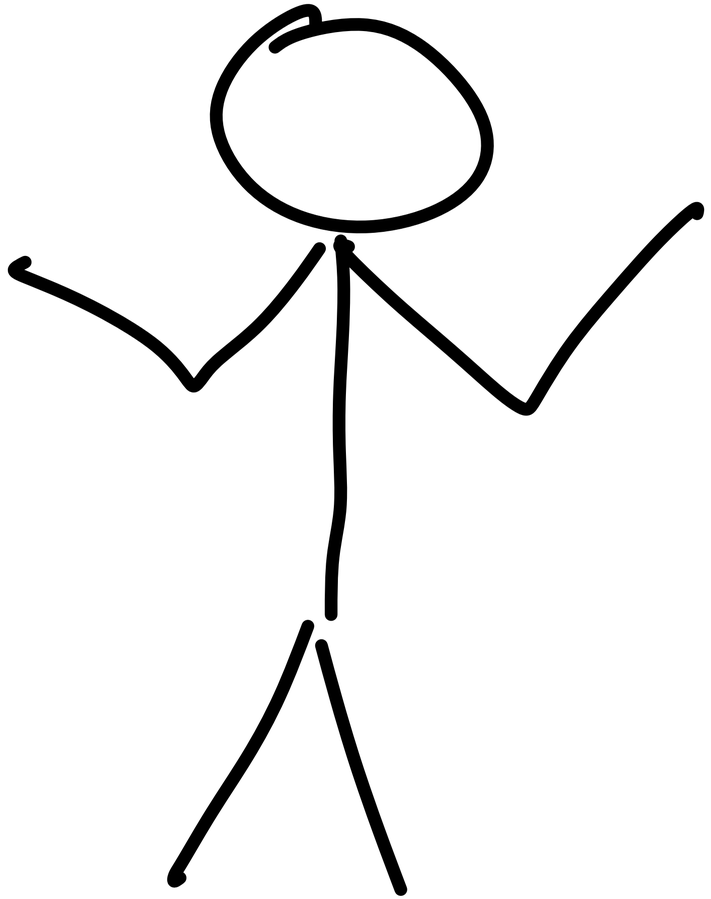


Another person shows up.

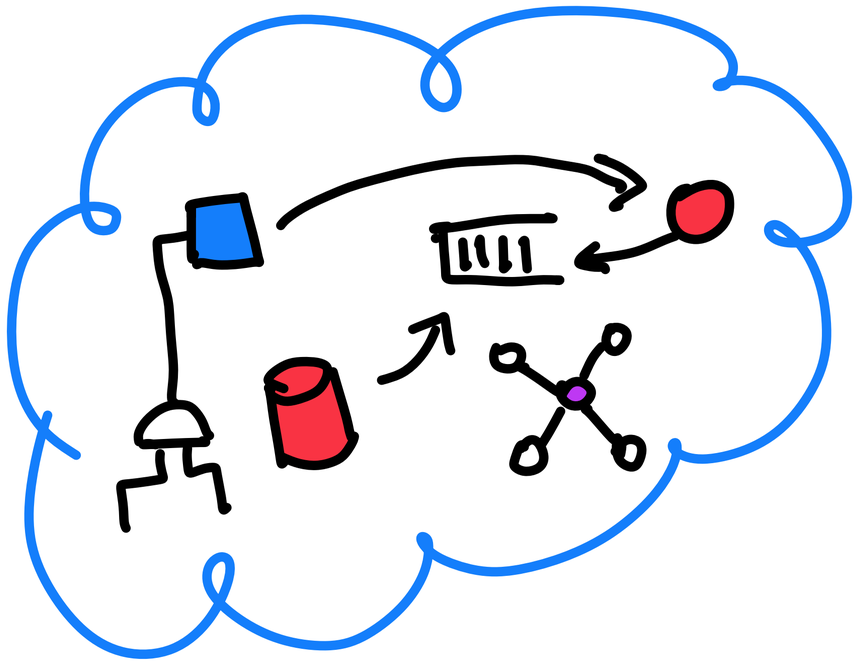



You
1,000,000 people show up

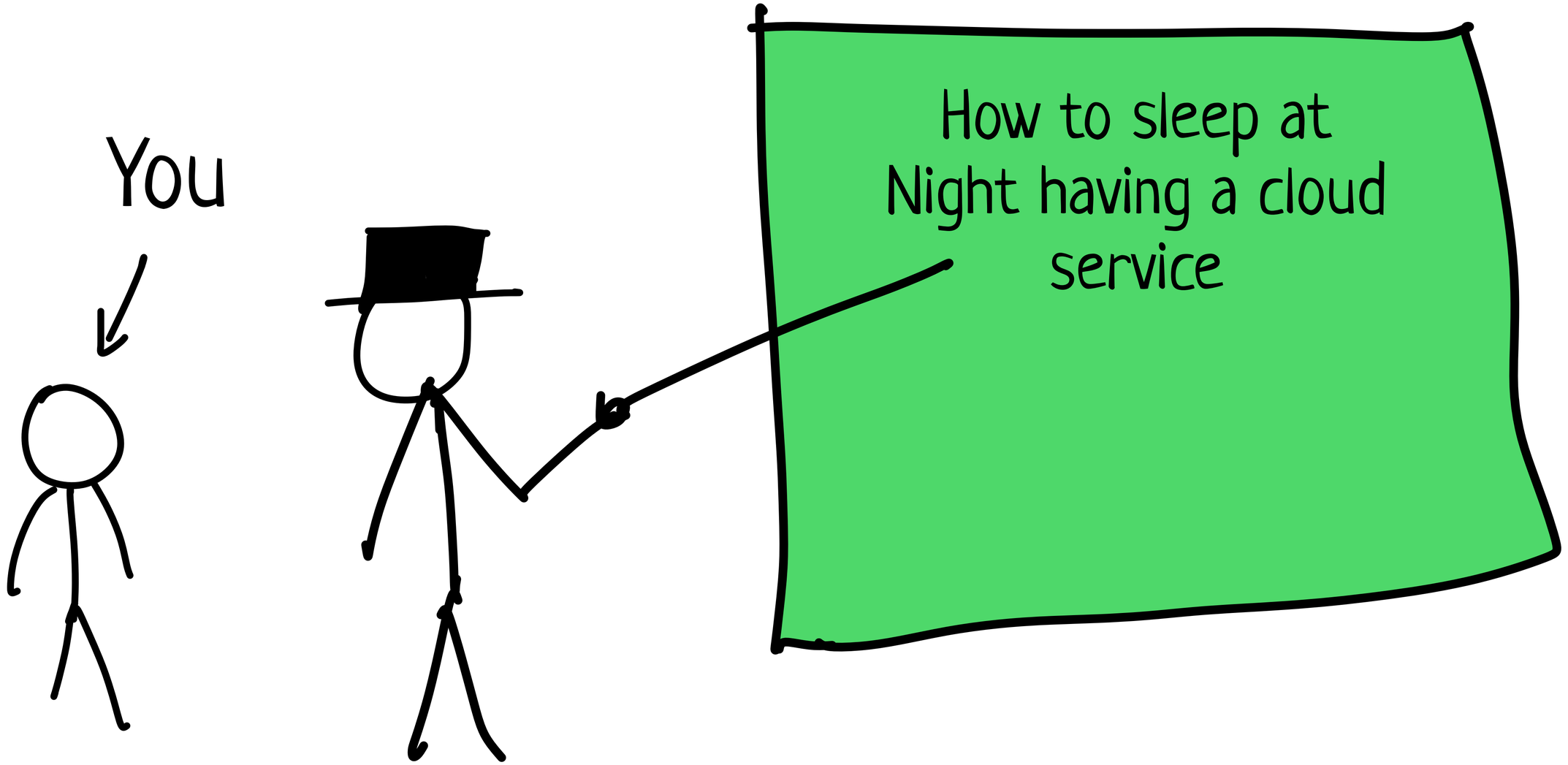
What do you do now?



I like to sleep at night



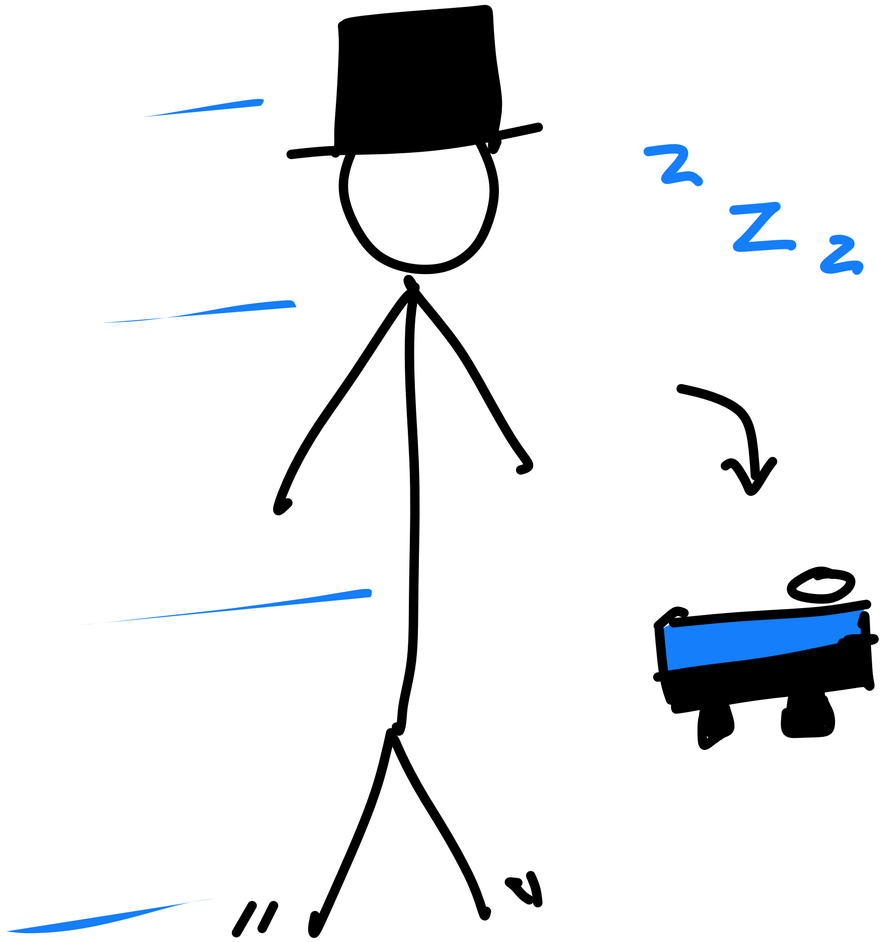
While having a cloud application


You
How to sleep at
Night having a cloud
service

One simple hack

To have a cloud service and sleep at night

Let it burn and go to sleep



Thank you for listening

Follow me on social media

DANIEL SADA

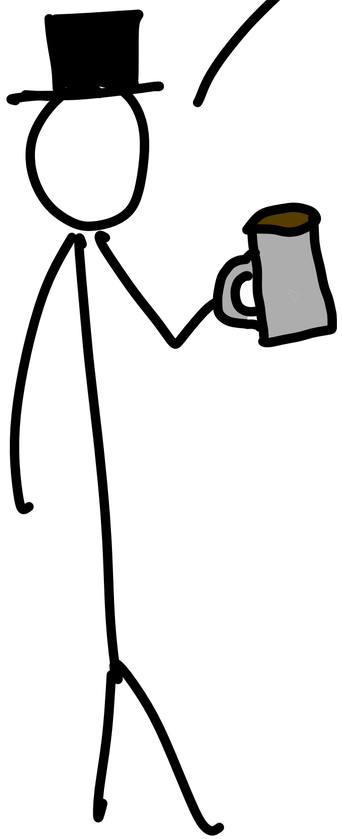
Developer Productivity & Culture

danielsada.tech

/in/danielsadac

@dsada@infosec.exchange

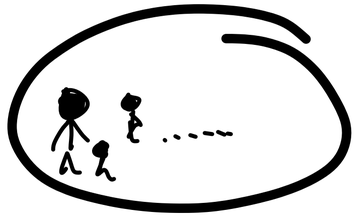
JK

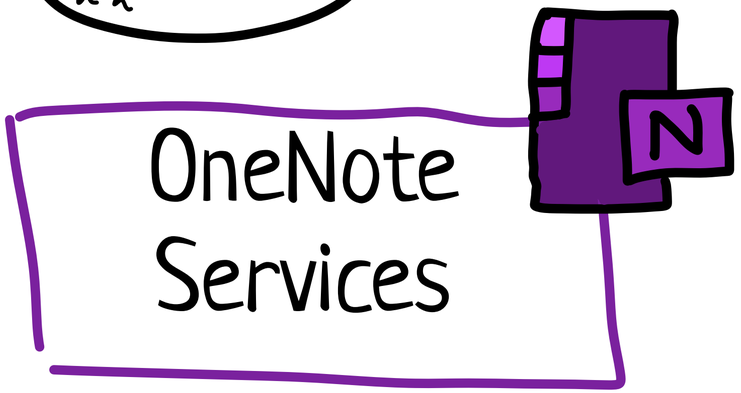


Daniel Sada

Backend services for millions

Throughout the world

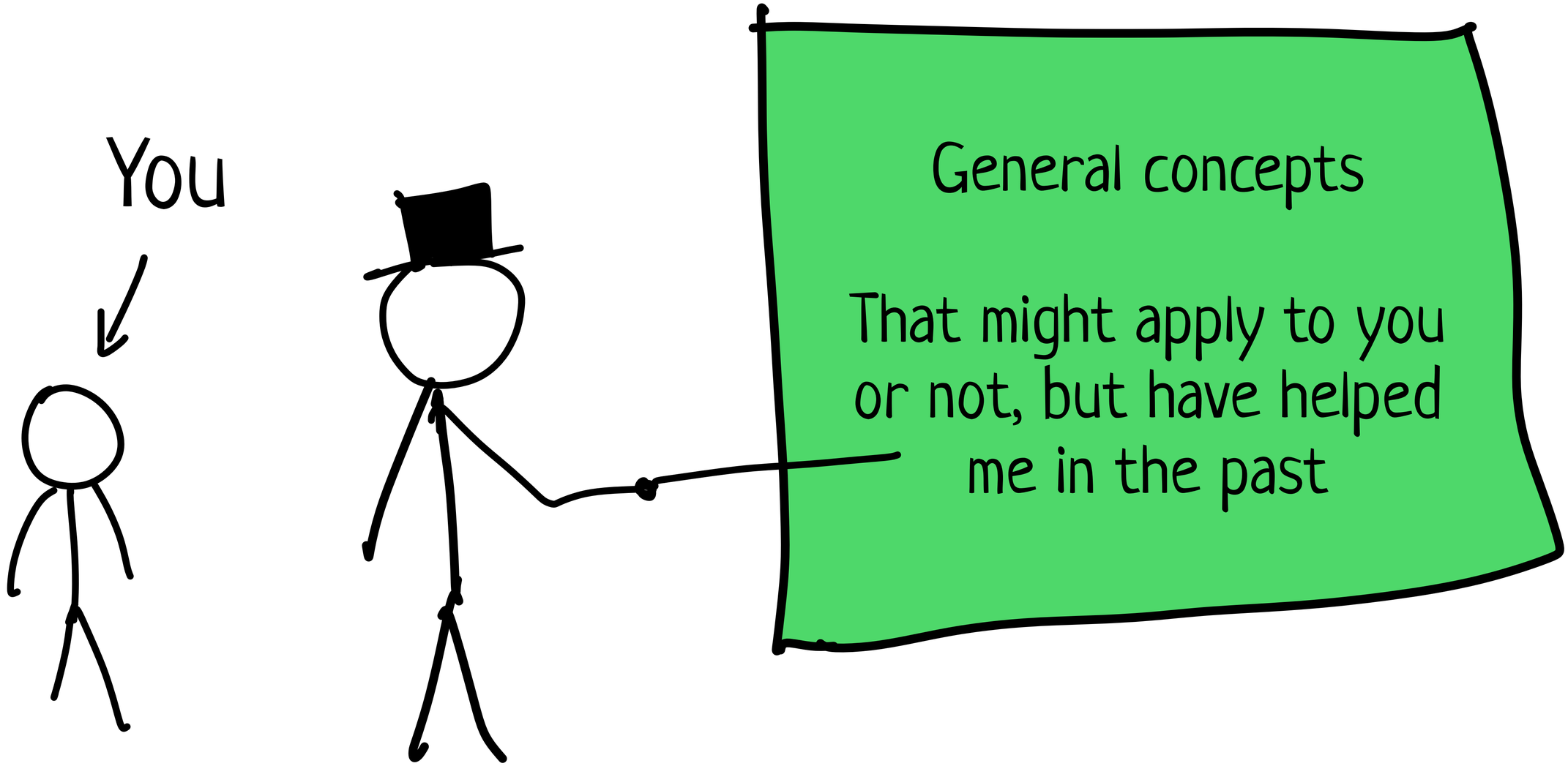




SDE & DevOps

Snowflake

Developer
Productivity

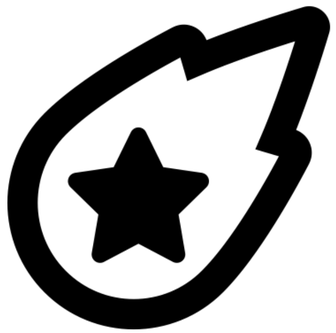

You
General concepts
That might apply to you or not, but have helped me in the past

We'll see the following levels:

Easy

Medium

Hard



EXPERT

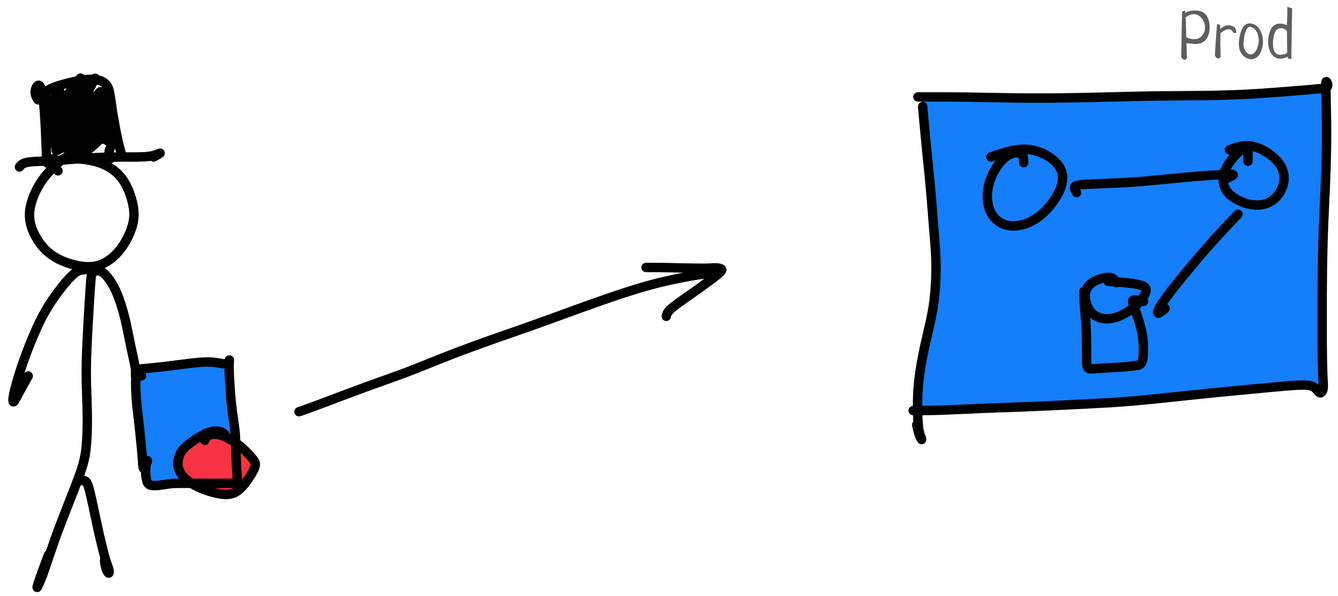
Easy *

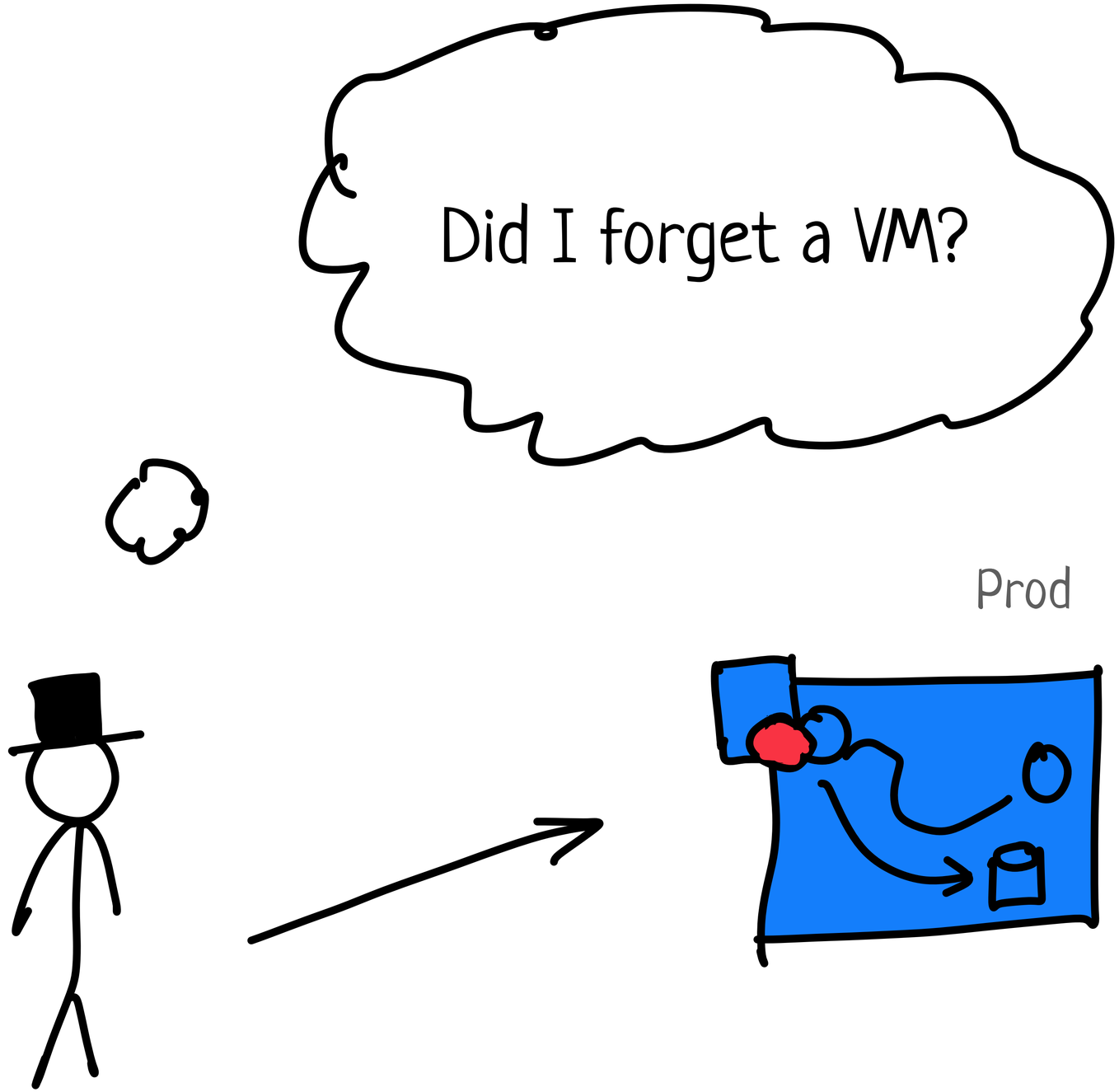
* Not really easy, but simpler than other levels.

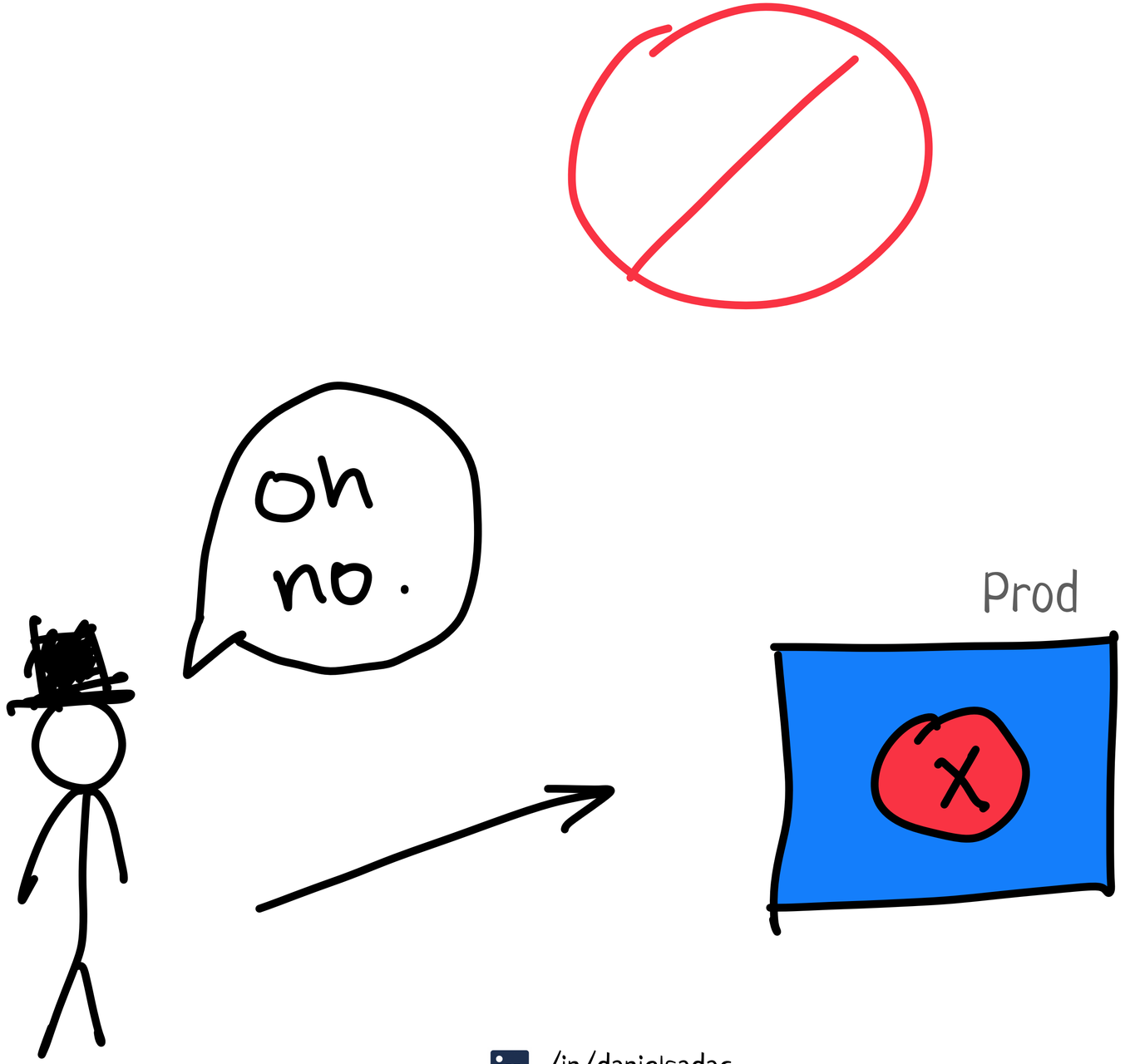
IAC

Infrastructure as Code

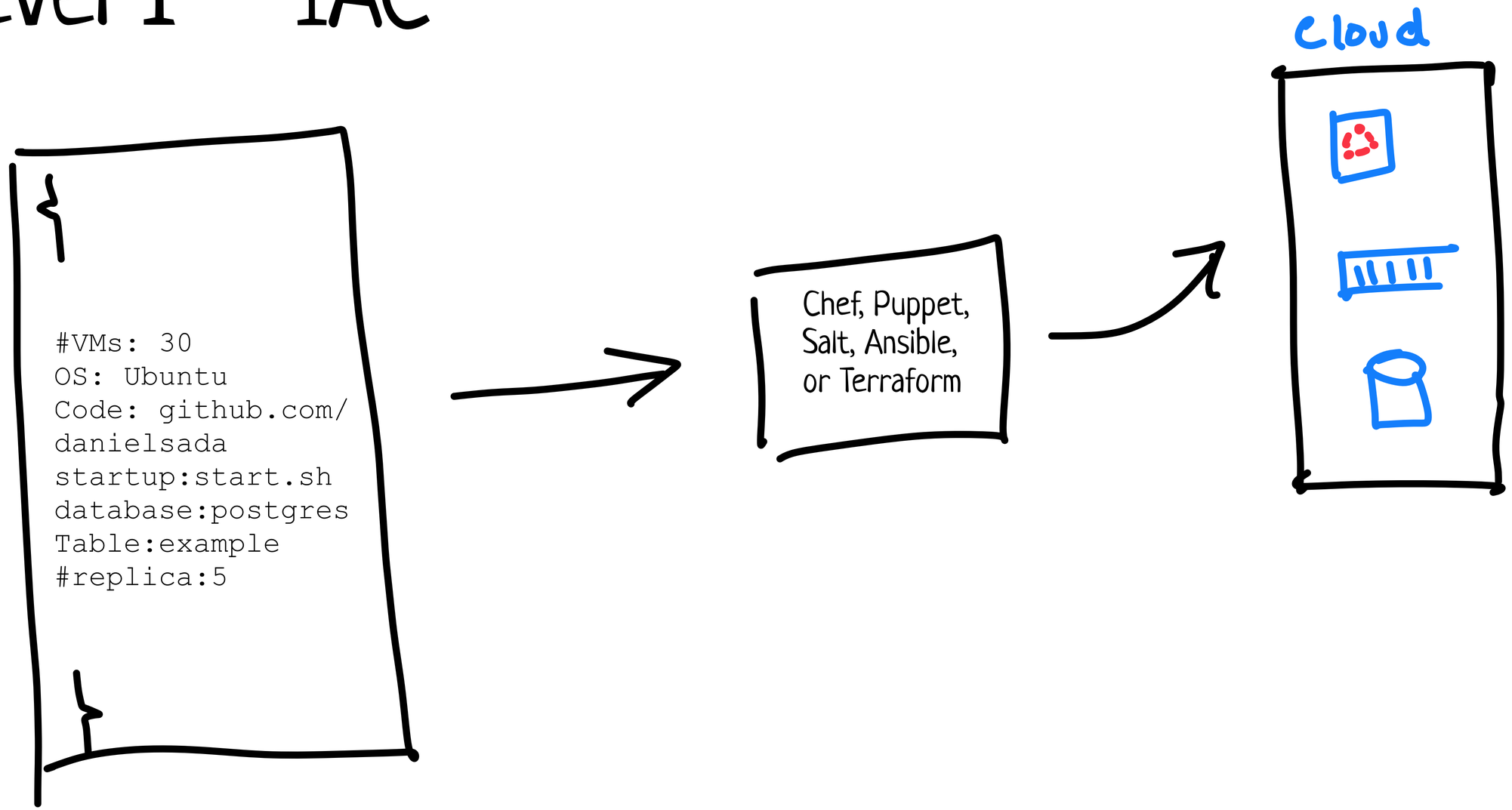
Deploying manually infrastructure in the cloud




Did I forget a VM?
Prod

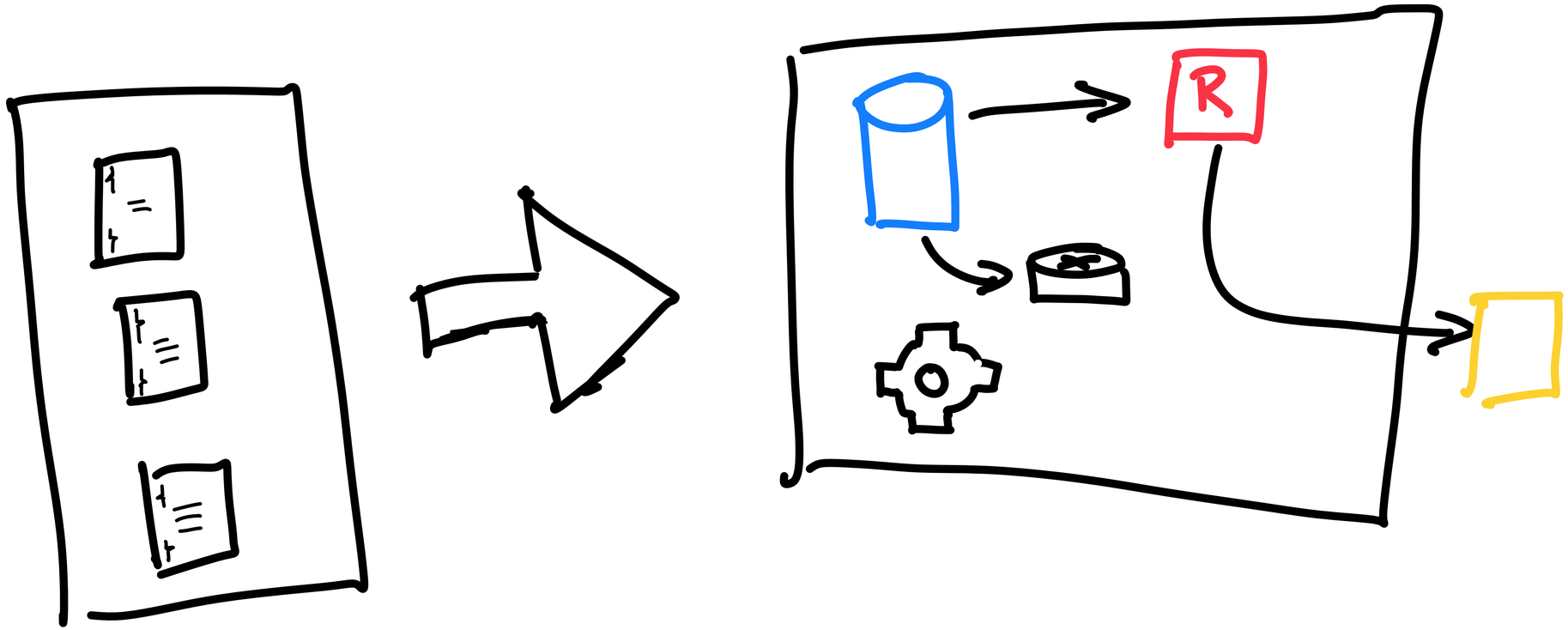

oh
no.
Prod

Level 1 – IAC



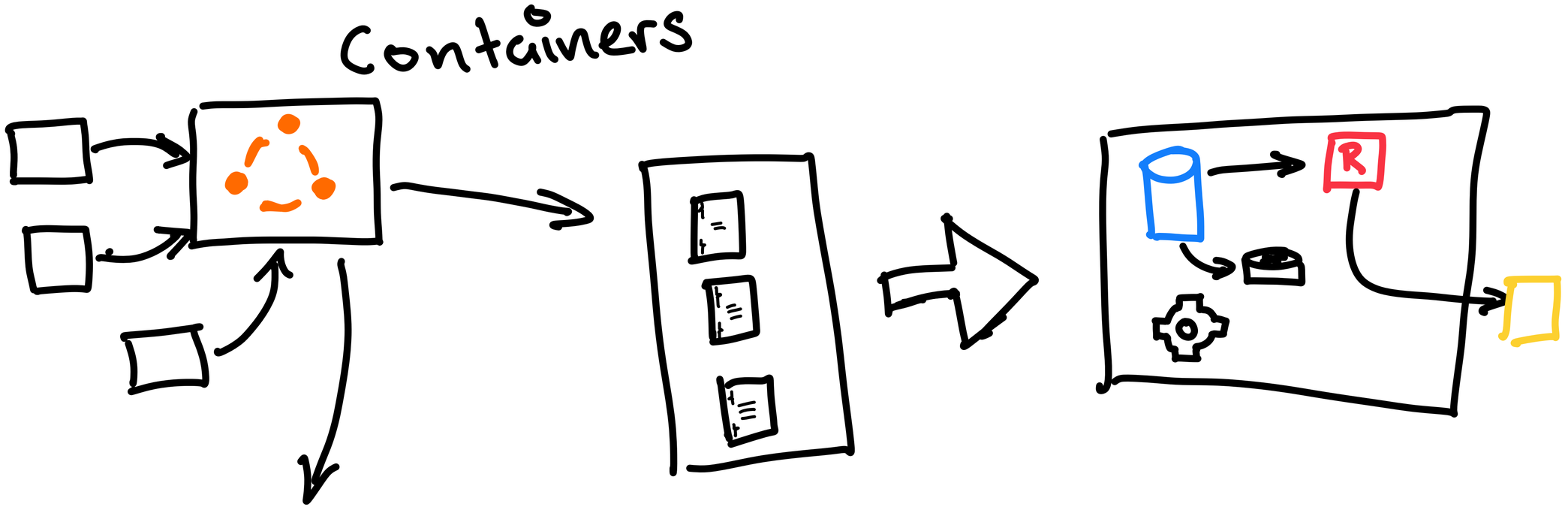
Script to Deploy

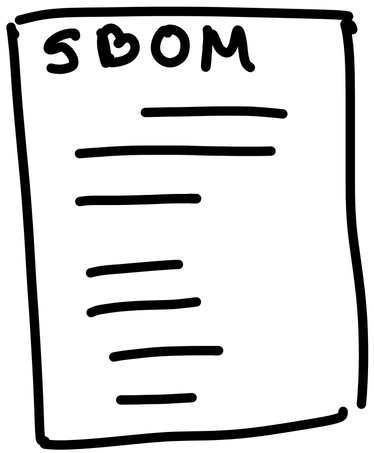
Level 2 – IAC



Set of scripts to describe a system.

Level 3 – IAC





Custom images + SBOM + Scripts

Software
Bill
Of
Materials

IAC

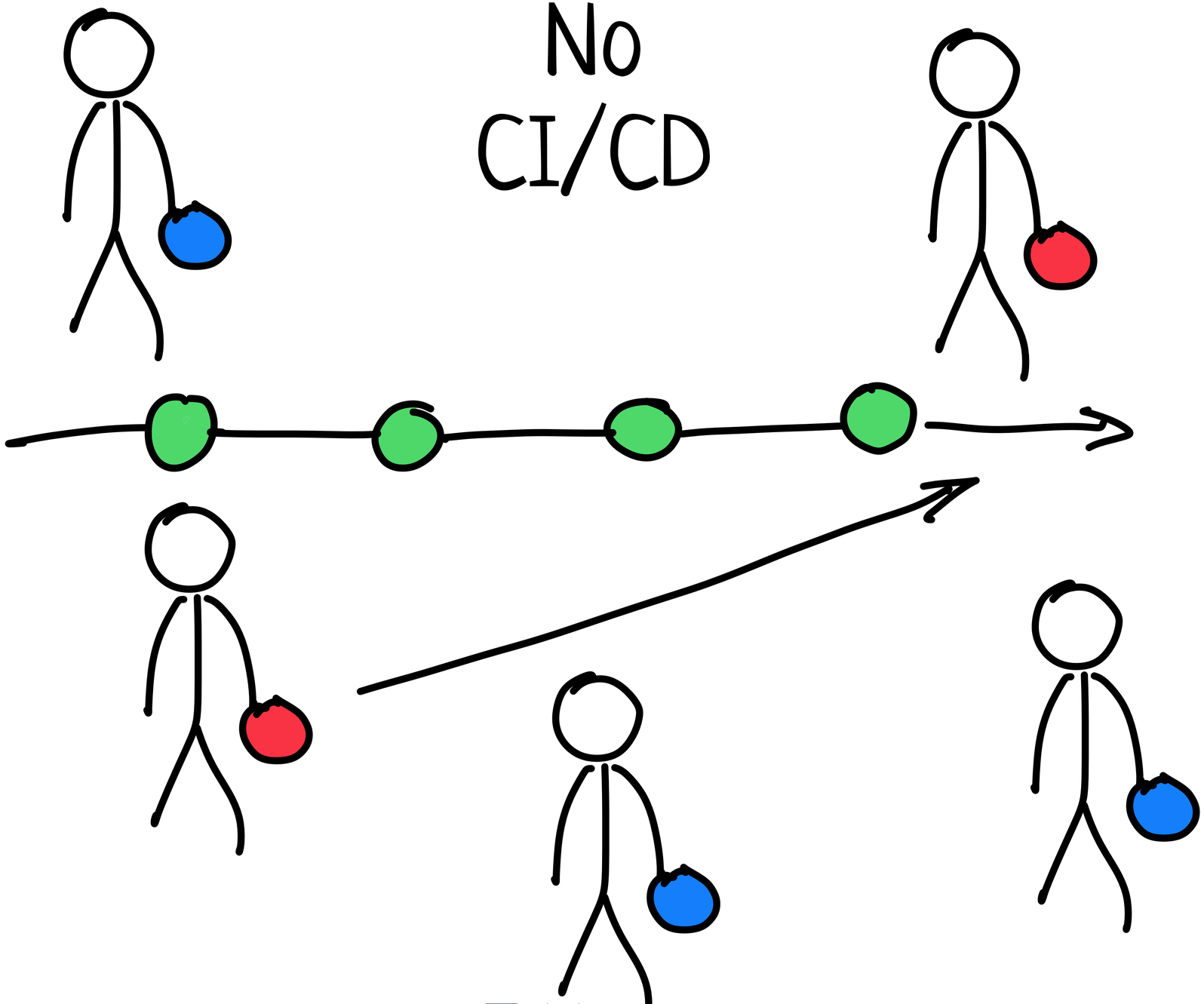
Infrastructure as Code

- Chef
- Puppet
- Ansible
- Terraform
- Cloud-specific lang

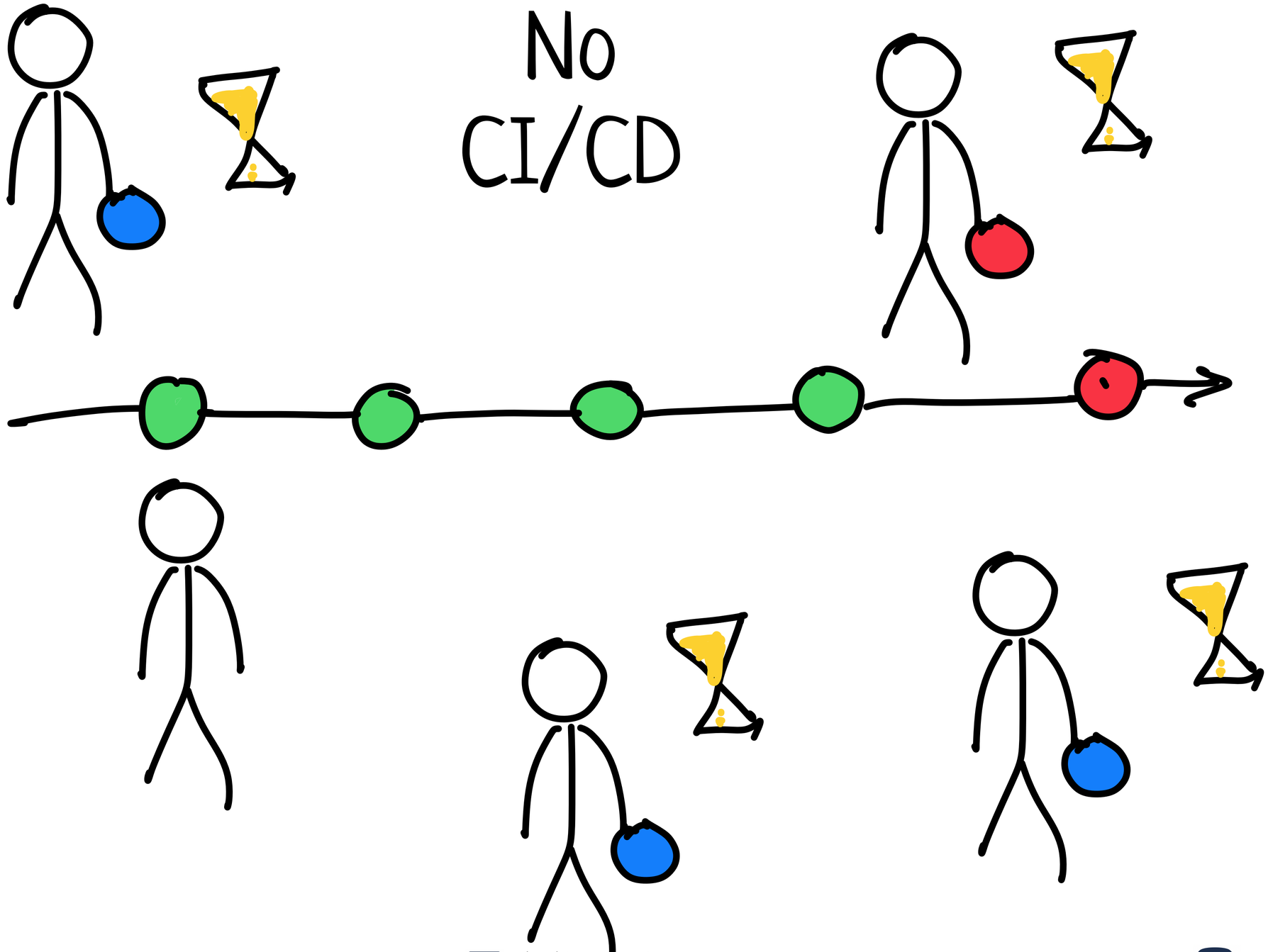
CI/CD

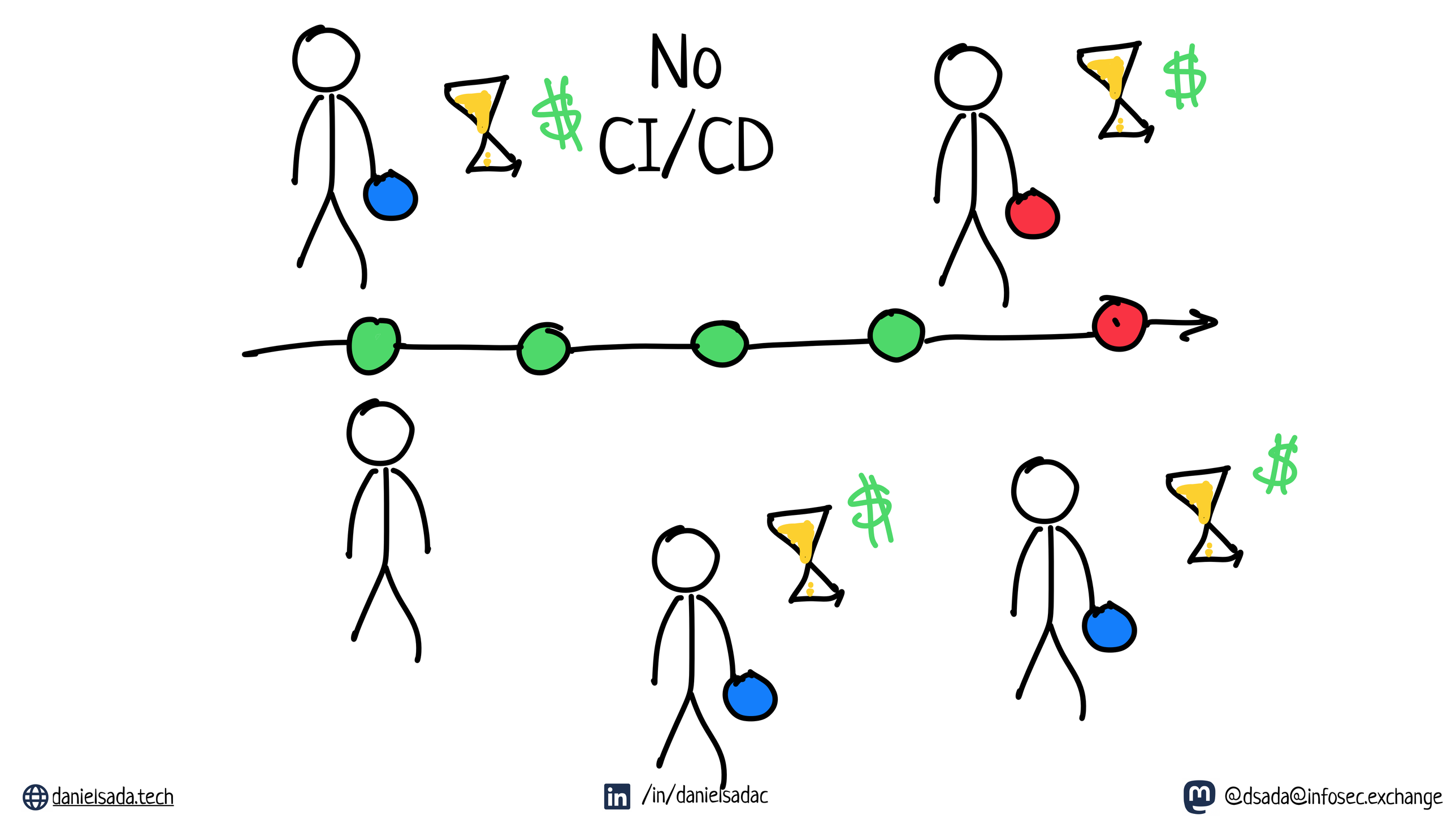
Continuous Integration / Continuous Deployment

No CI/CD

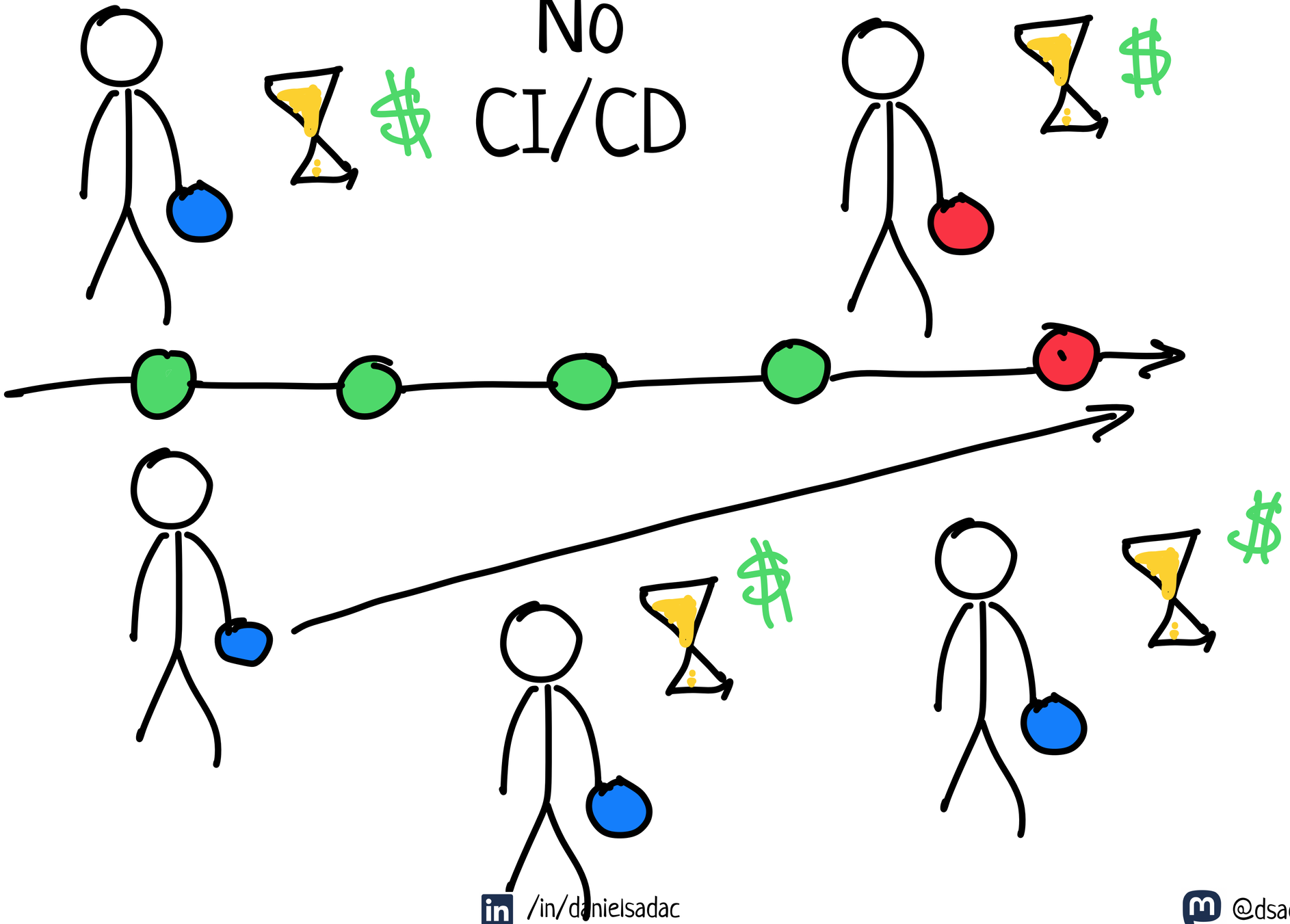


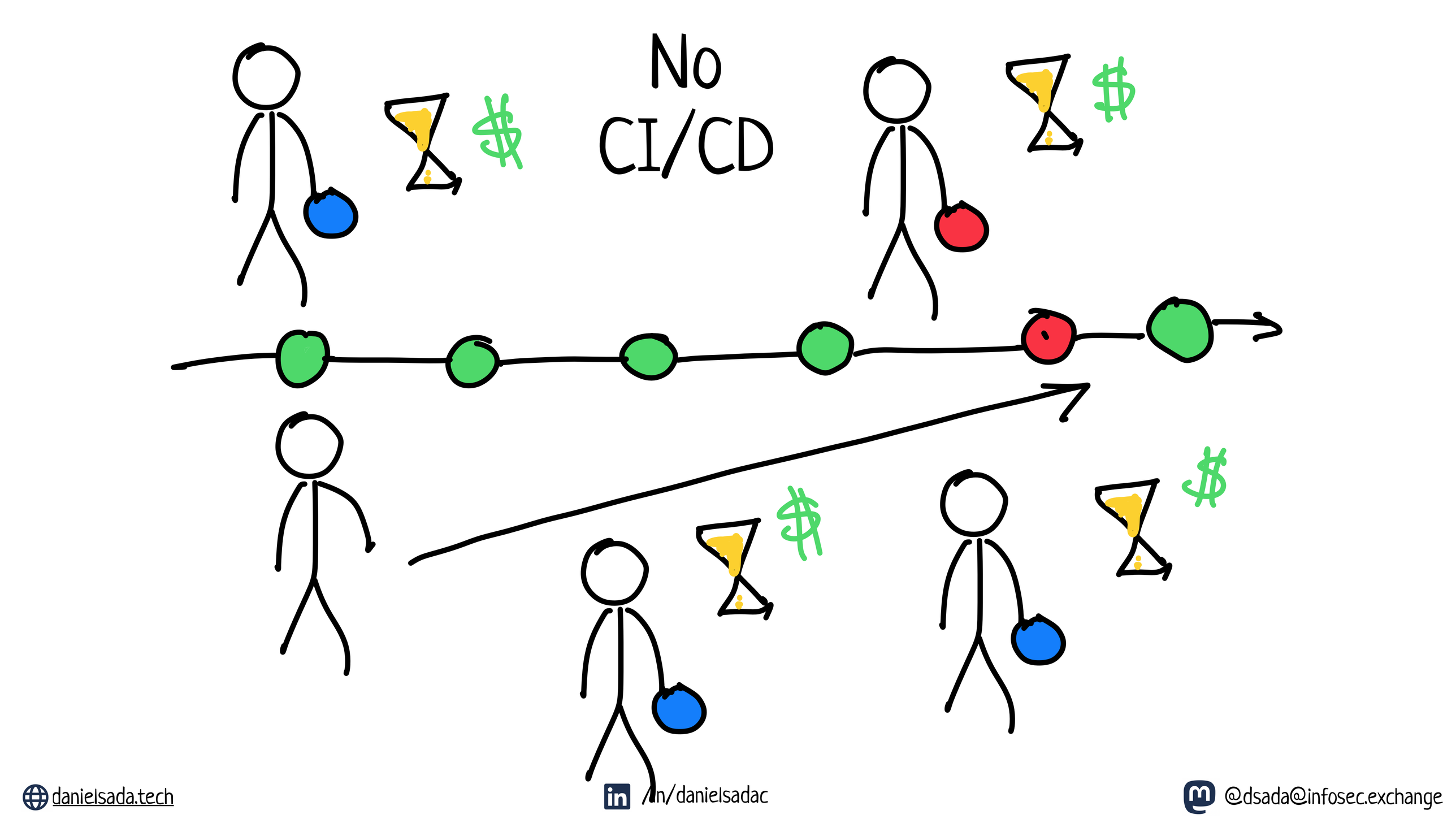
No CI/CD



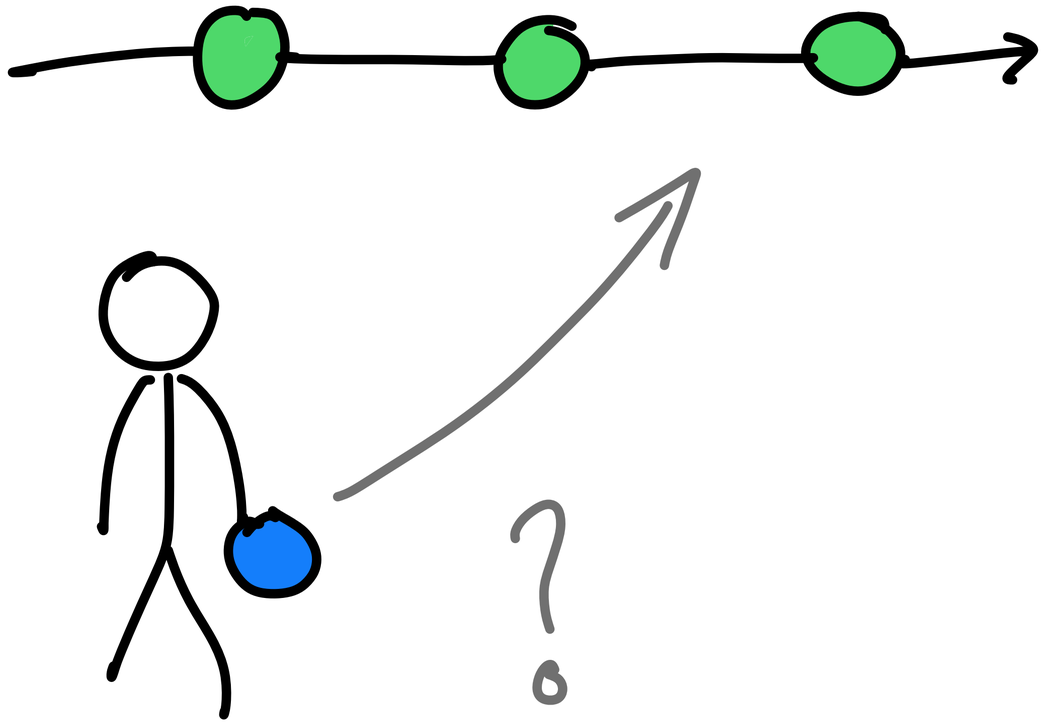

No
CI/CD

No CI/CD

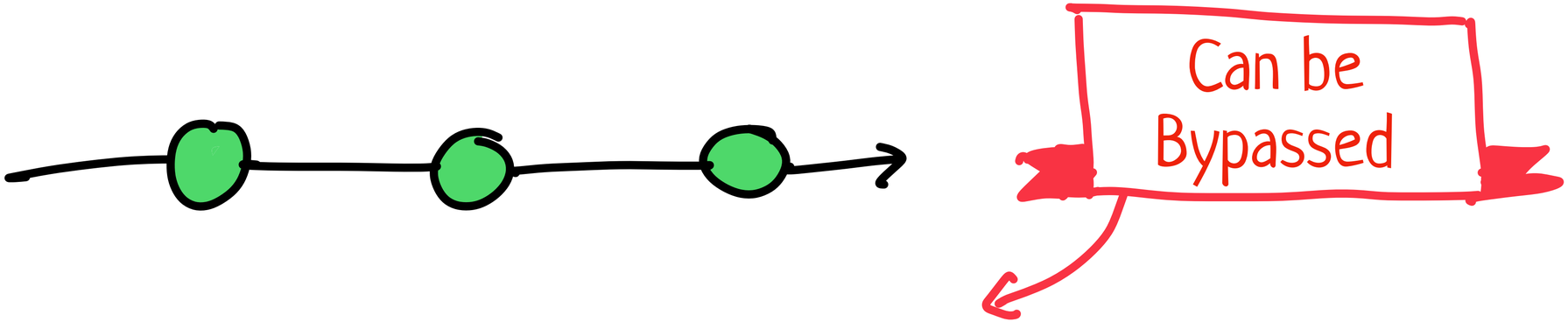


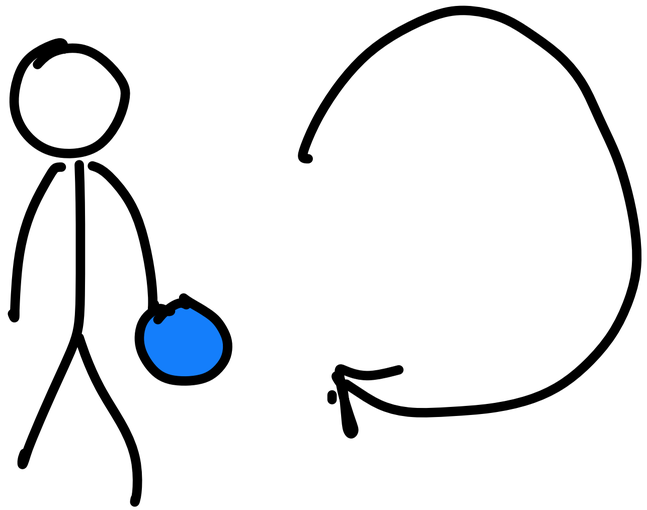

No
CI/CD

A developer wants to merge



Before commit

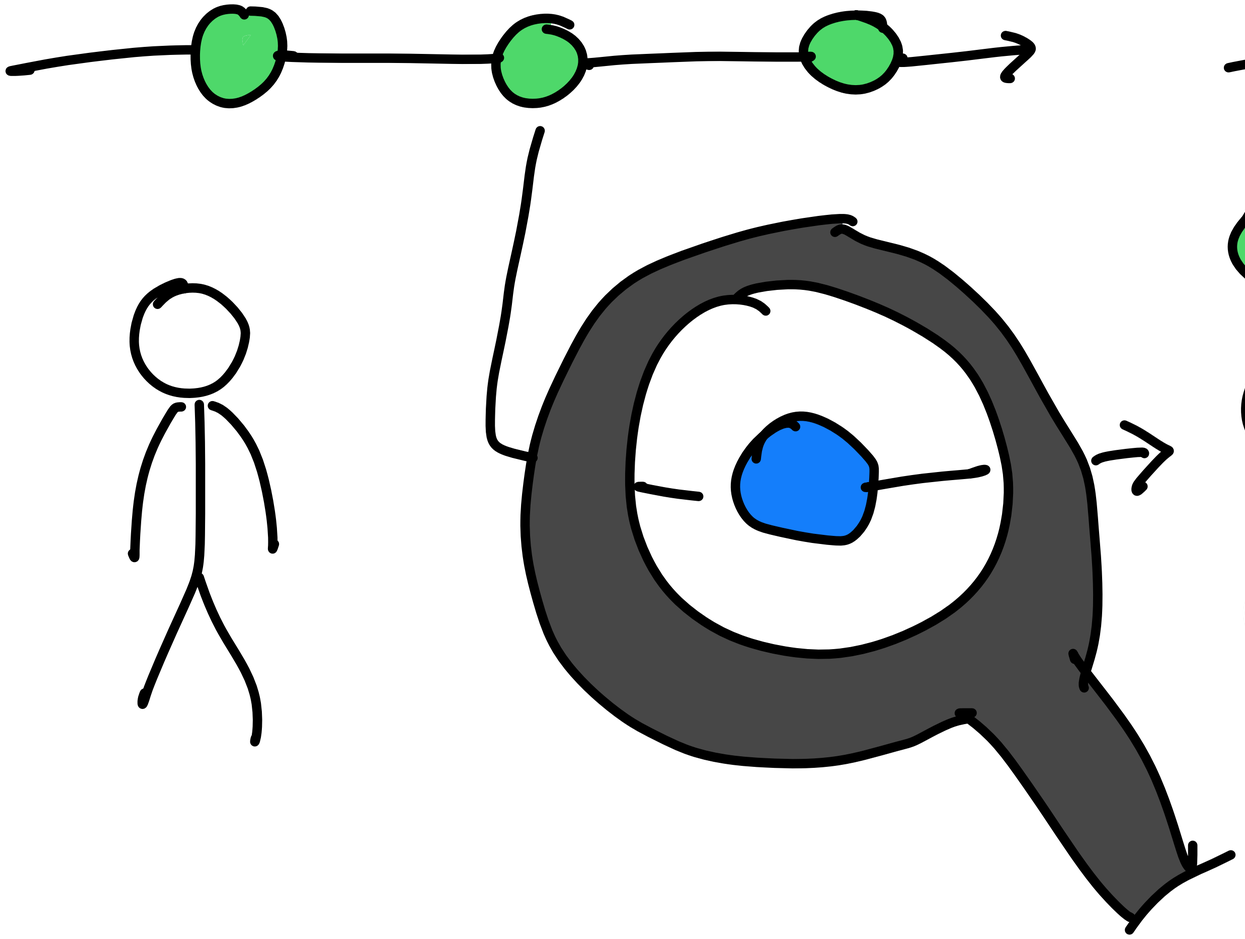




Precommit
- Linting
- Secrets
- ...

a.k.a git commit checks

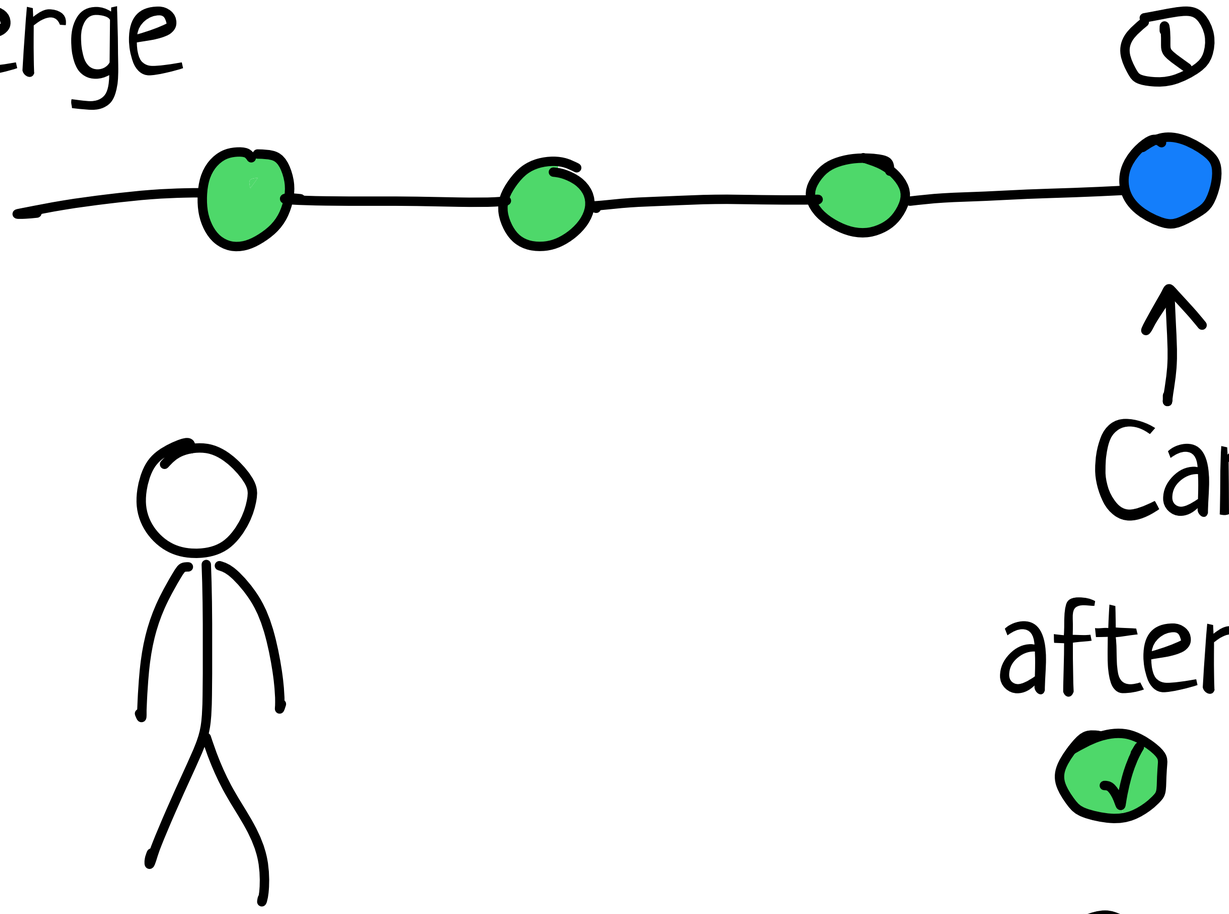
While branch is active



CI

- Linting
- Tests
- Secrets

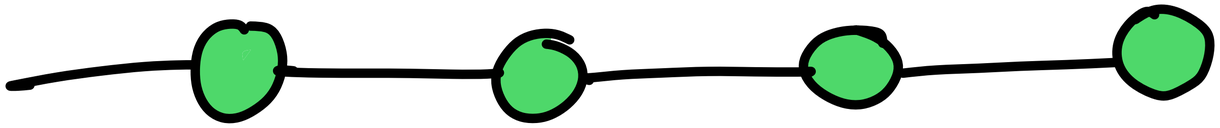
After merge

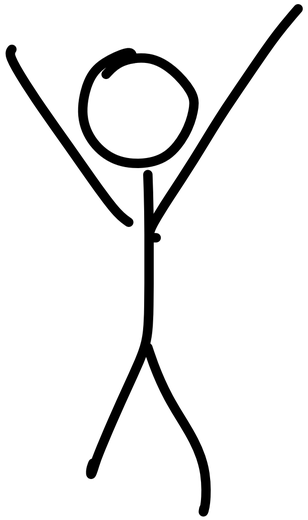


Can vary

after merge!

- Build
- Tests





Precommit

- Linting
- Secrets
- ...

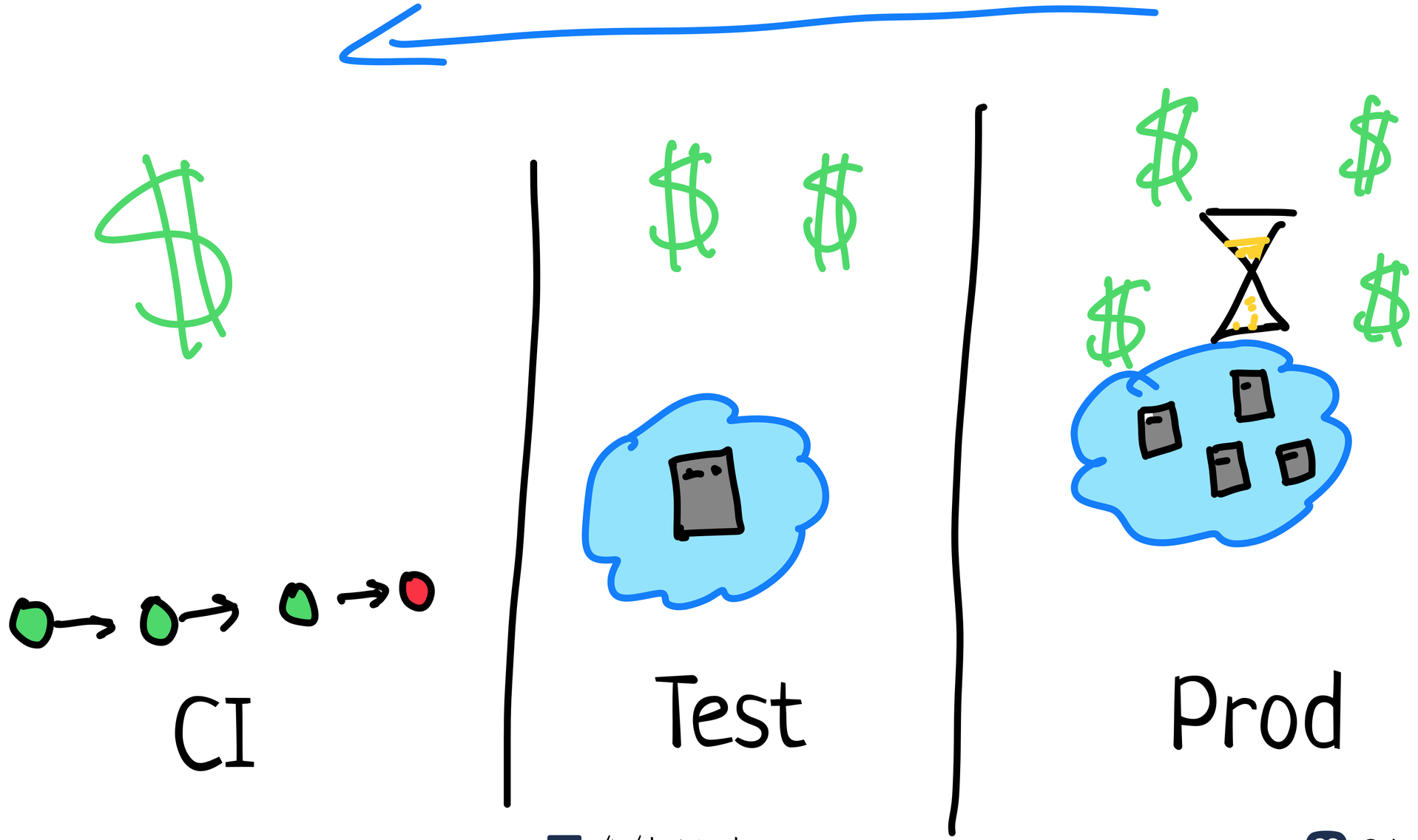
CI

- Build
- Linting
- Tests
- Secrets

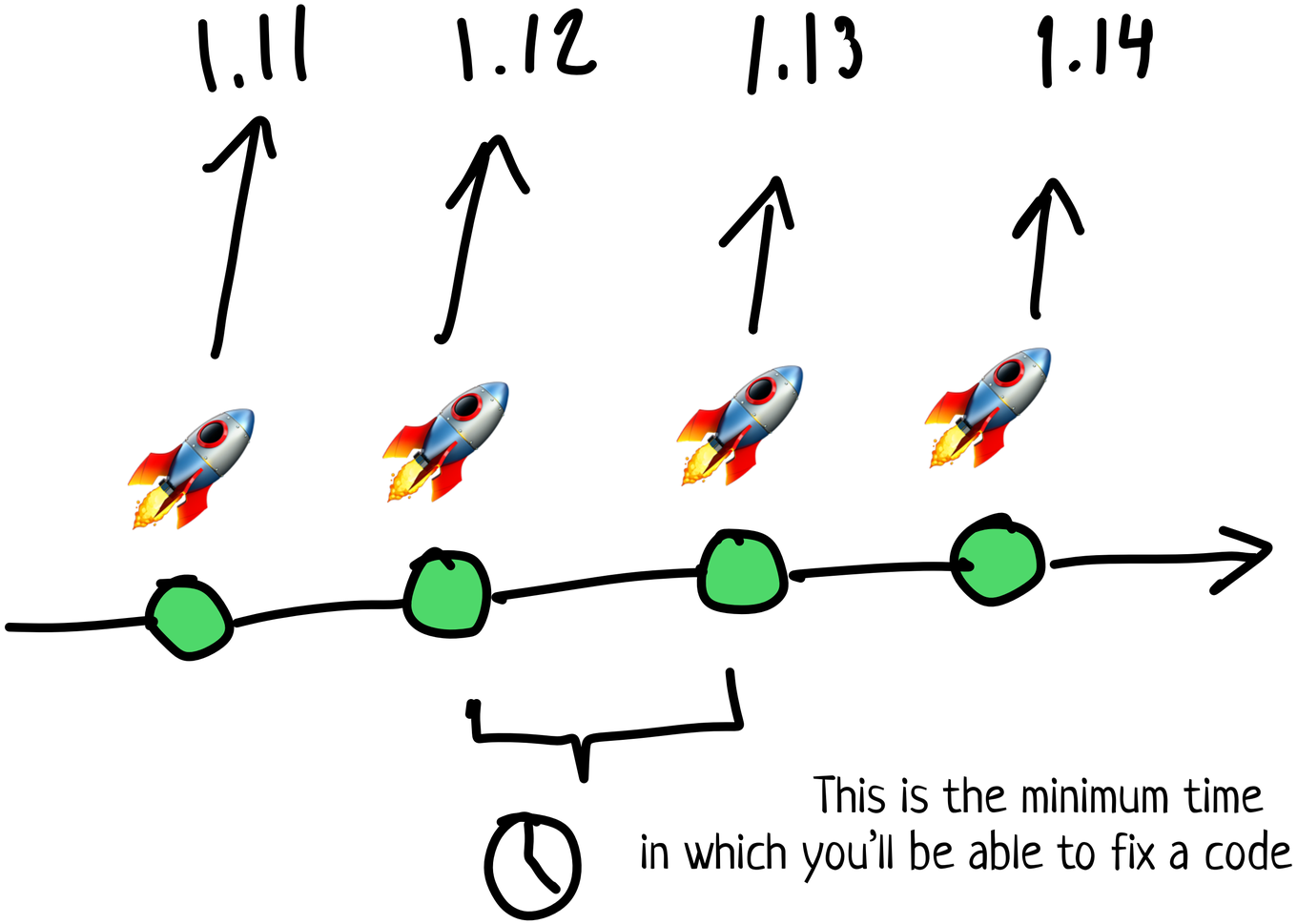
After merge

- Build
- Tests
- Secrets

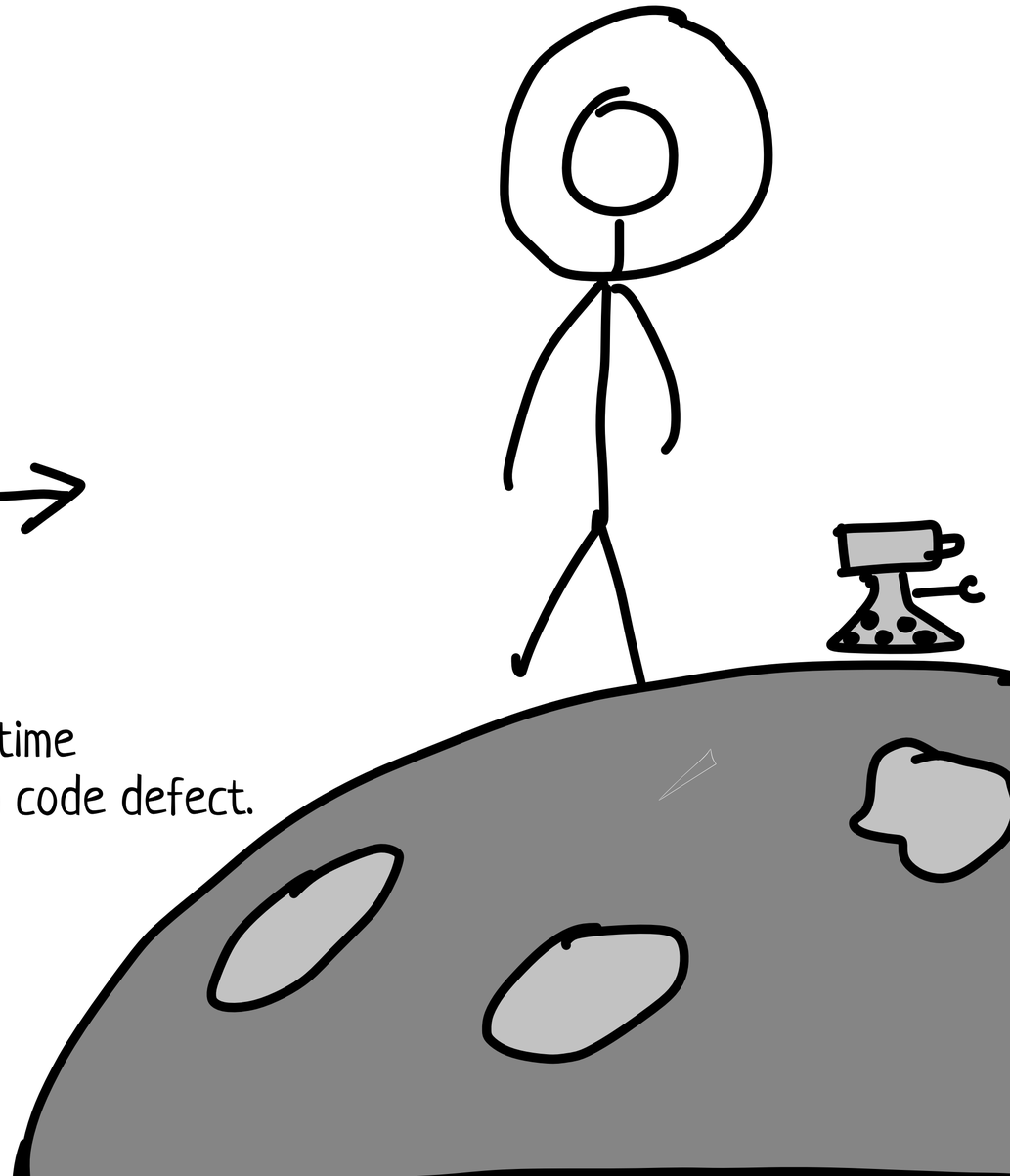
Failures are easier to catch, the earlier they come.



Continous deployment



This is the minimum time
in which you'll be able to fix a code defect.

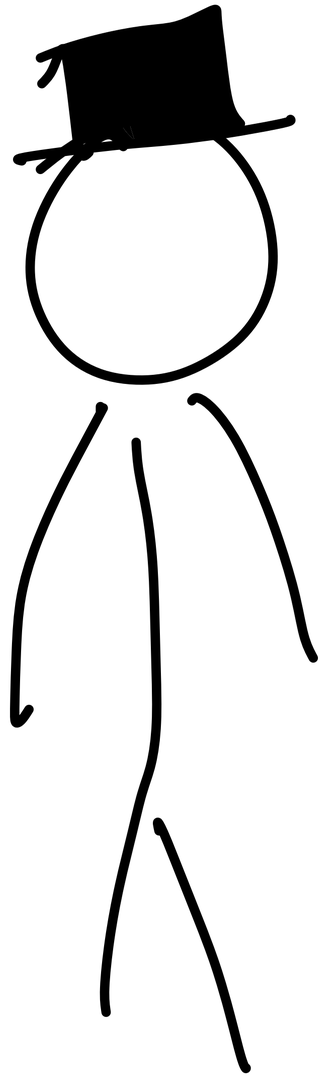


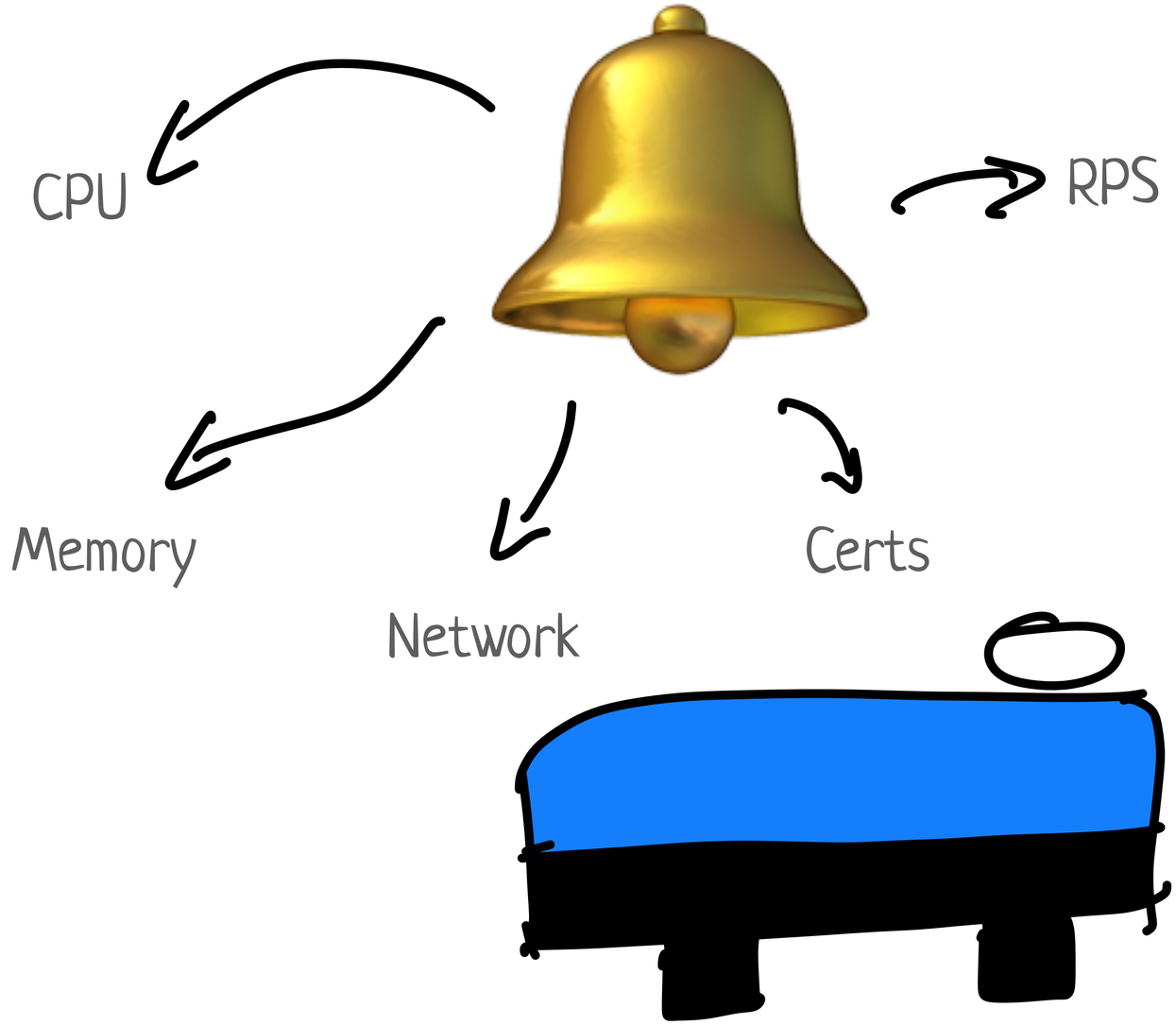
CI/CD

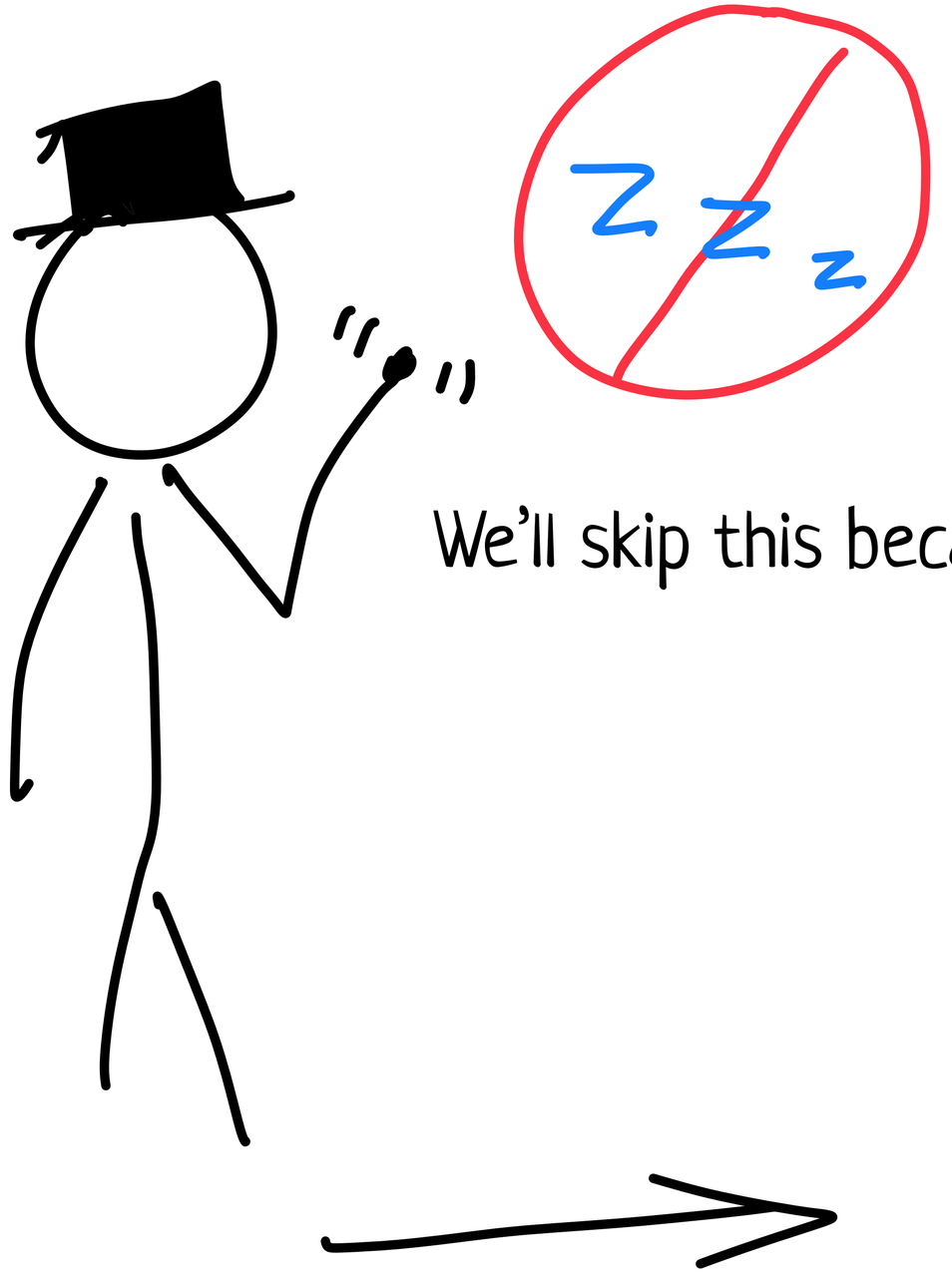
Continuous Integration
/ Continuous Deployment

- Github Actions
- Jenkins
- CircleCI
- TravisCI
- Etc...

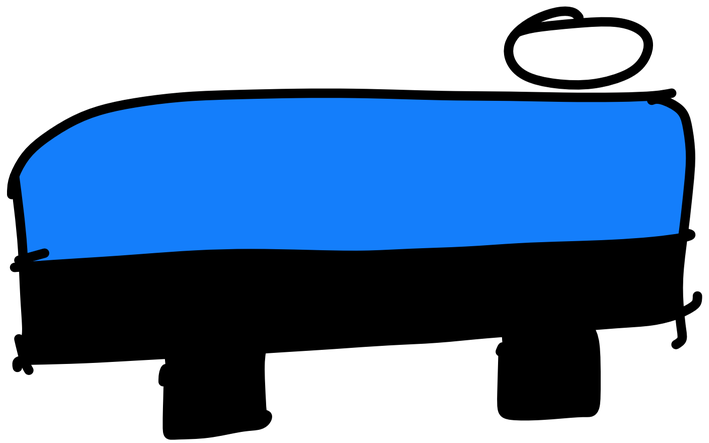
Alerts




CPU
RPS
Memory
Network
Certs

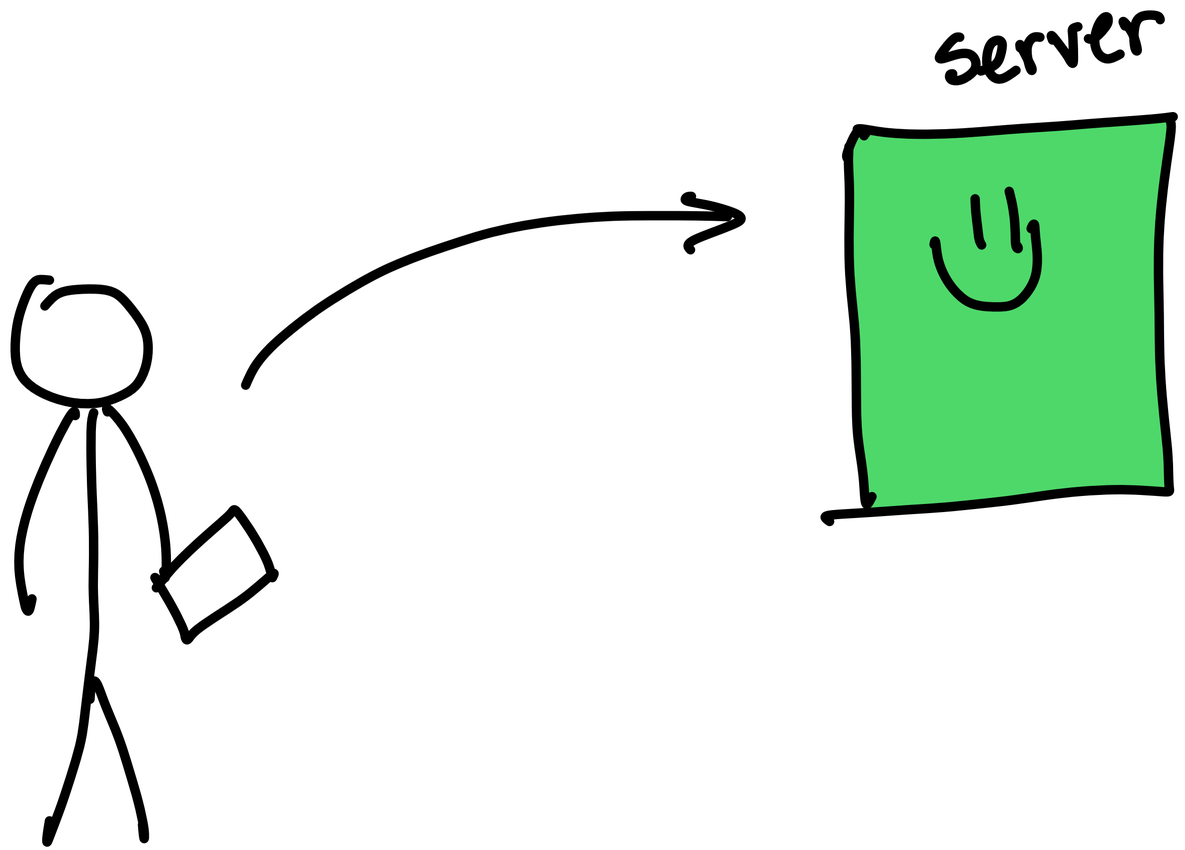


We'll skip this because we want to sleep

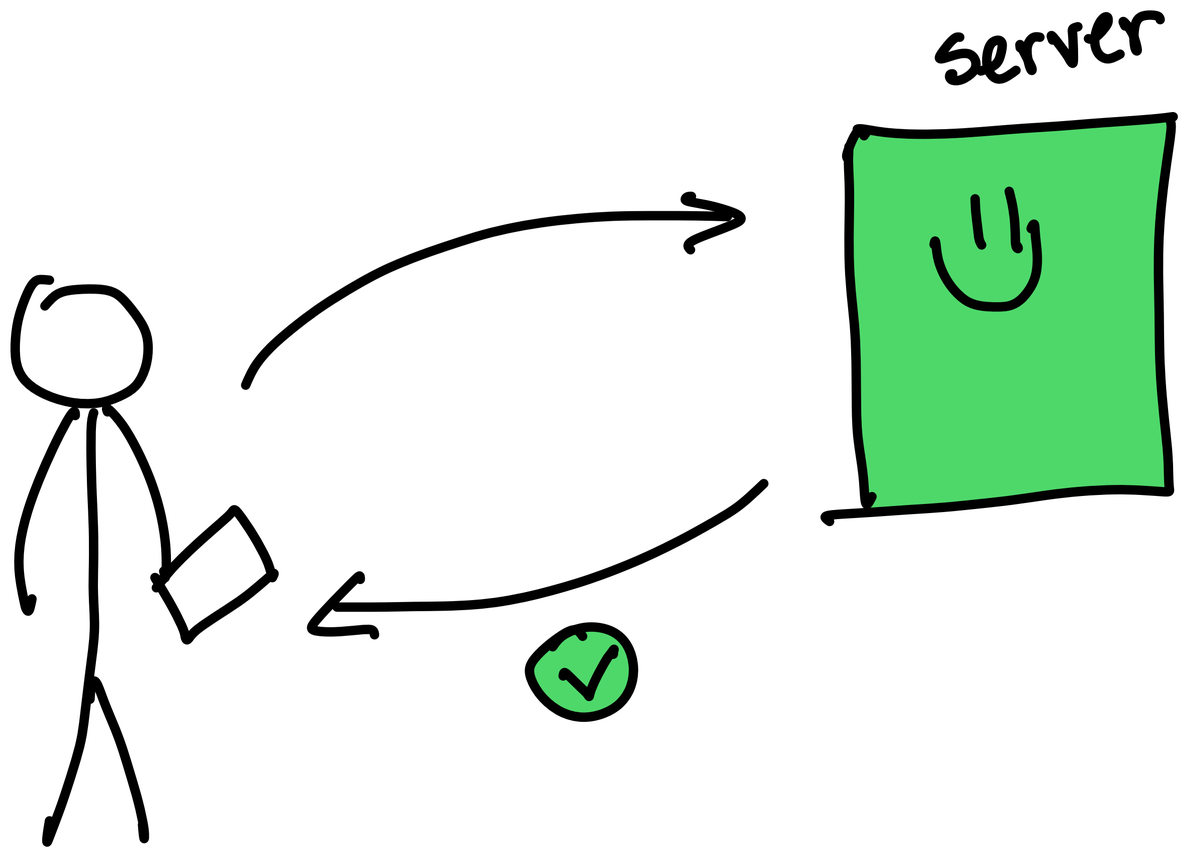


Load Balancer

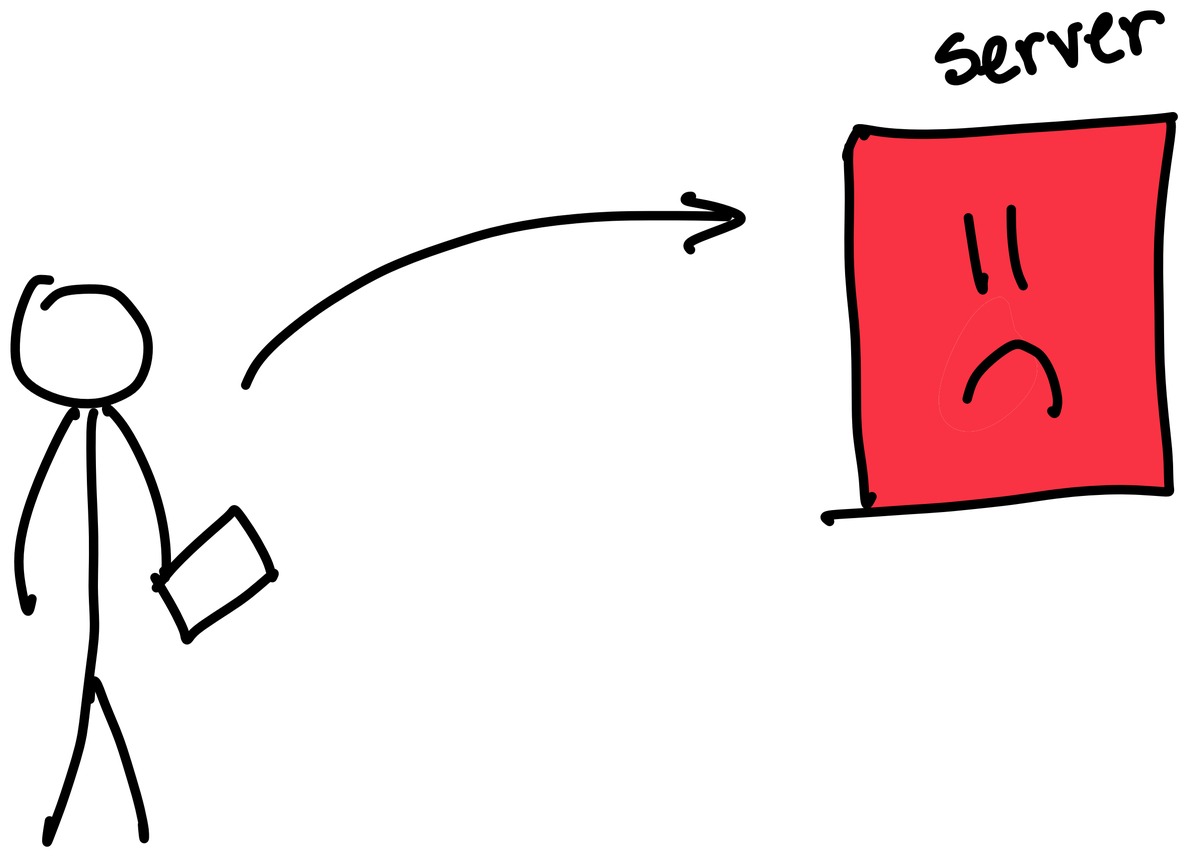
If we have only one server and no load balancing



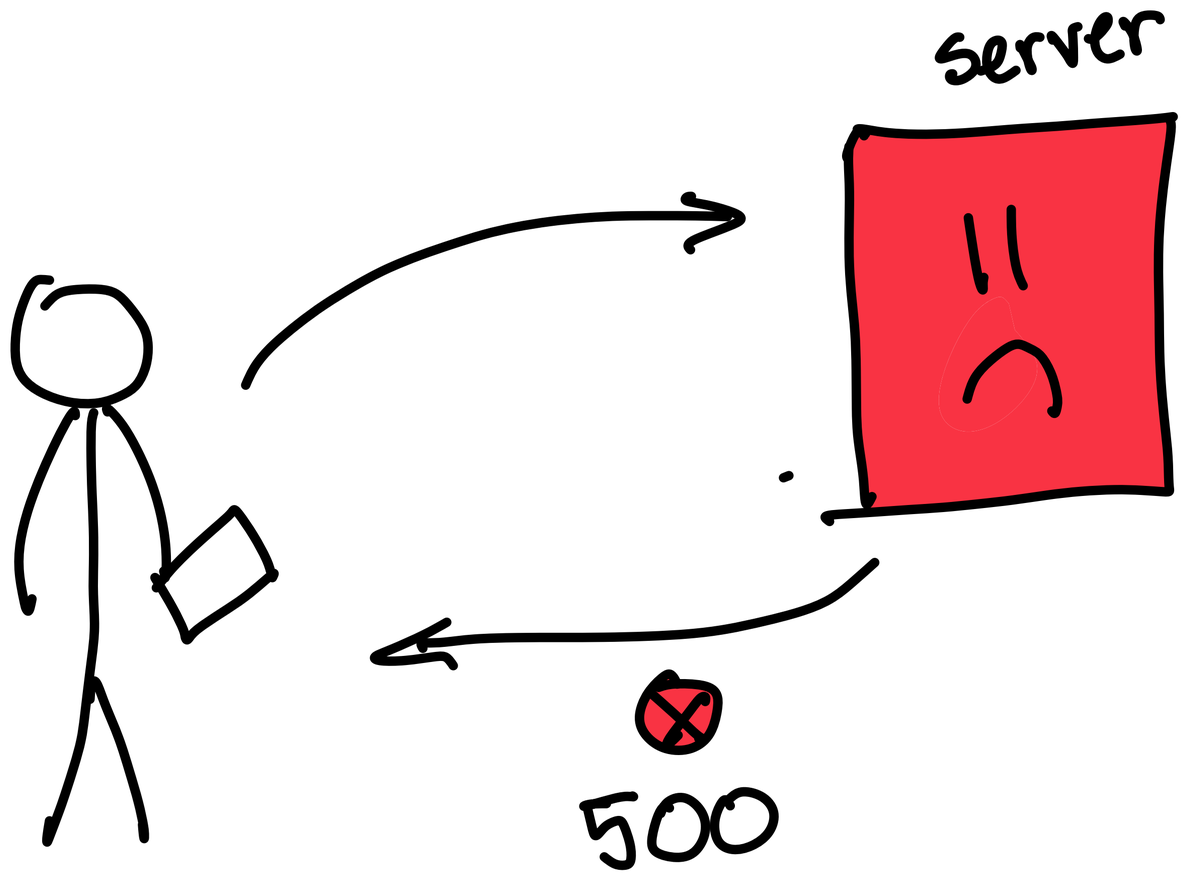
If we have only one server and no load balancing



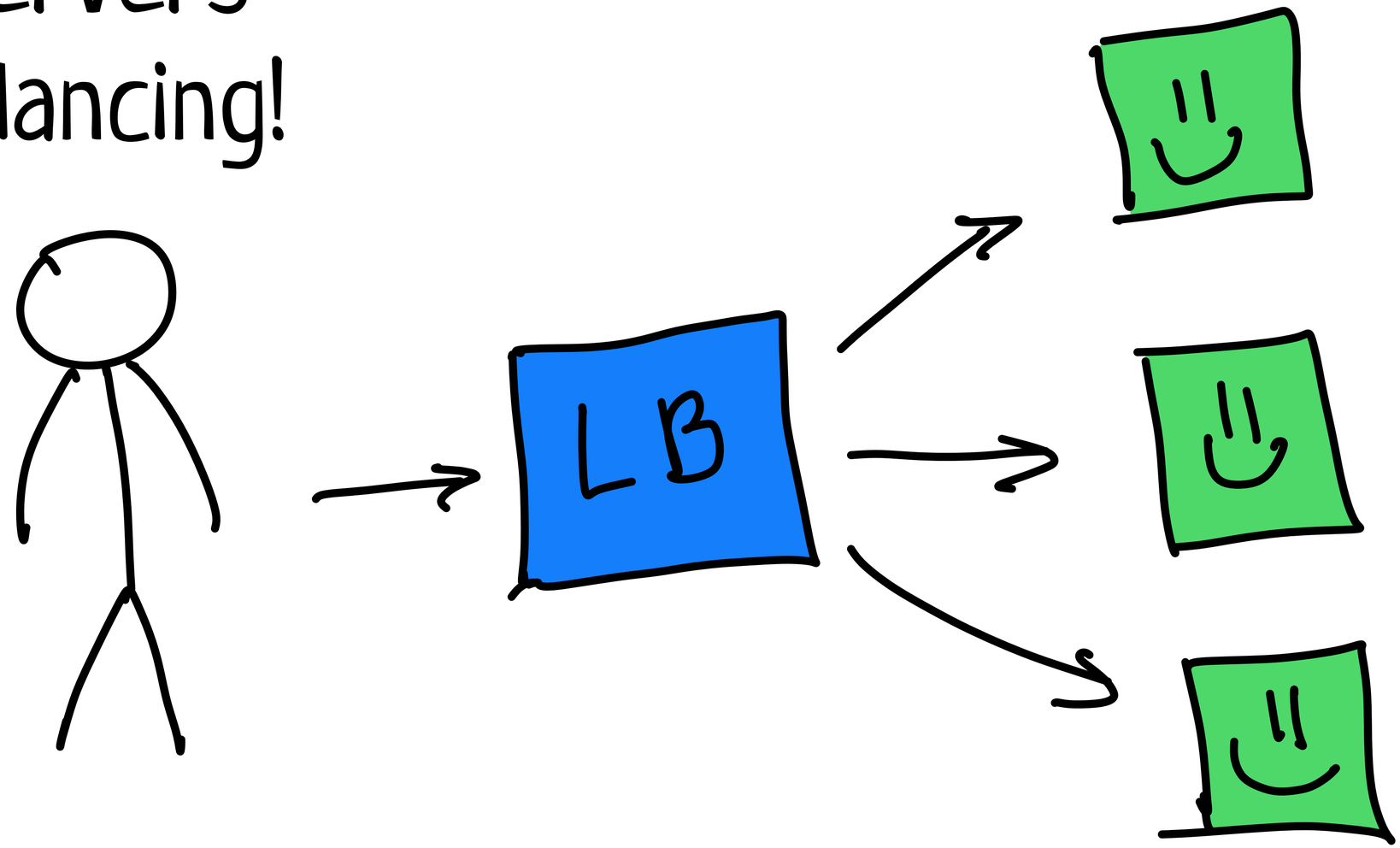
If we have only one server and no load balancing



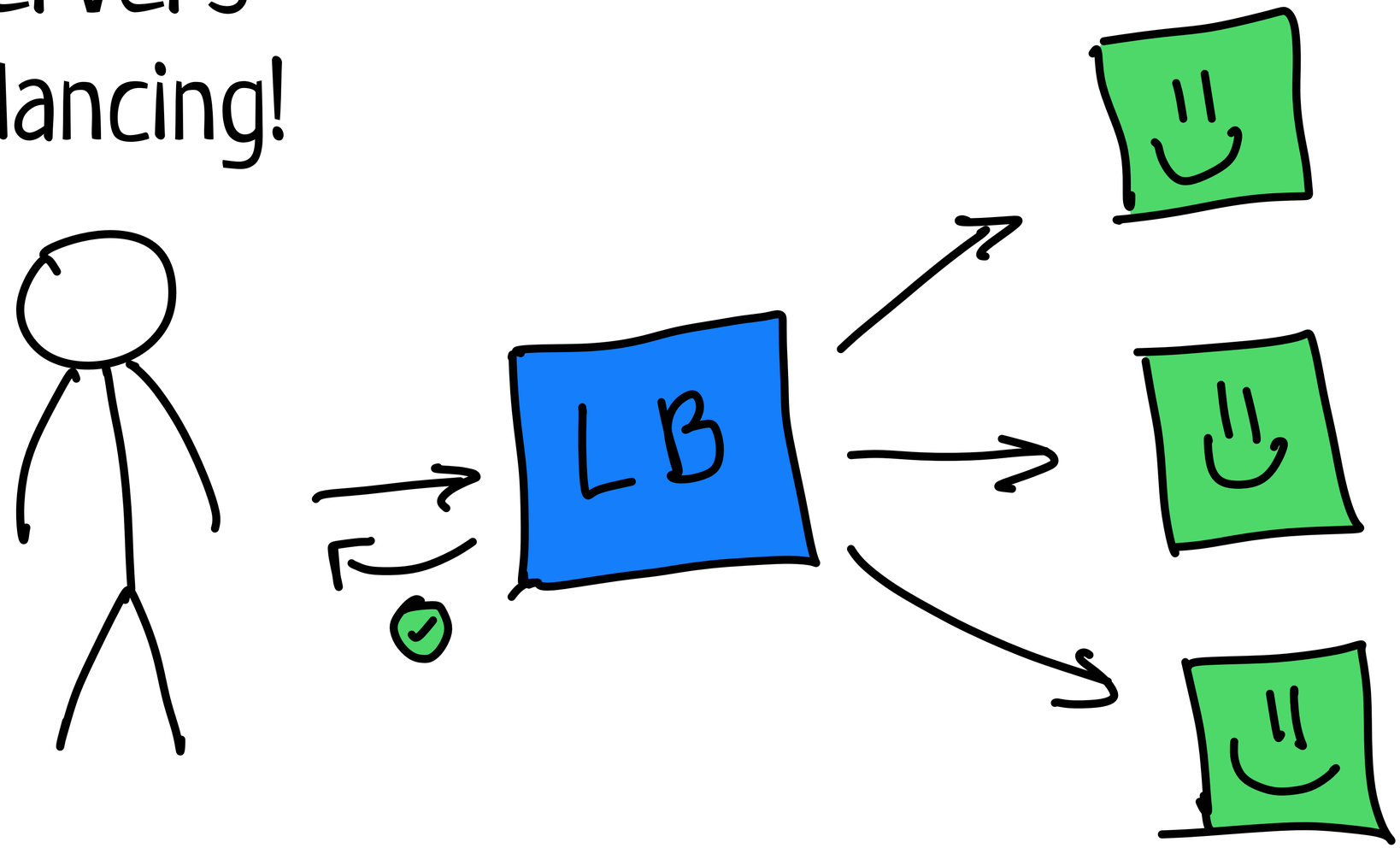
If we have only one server and no load balancing



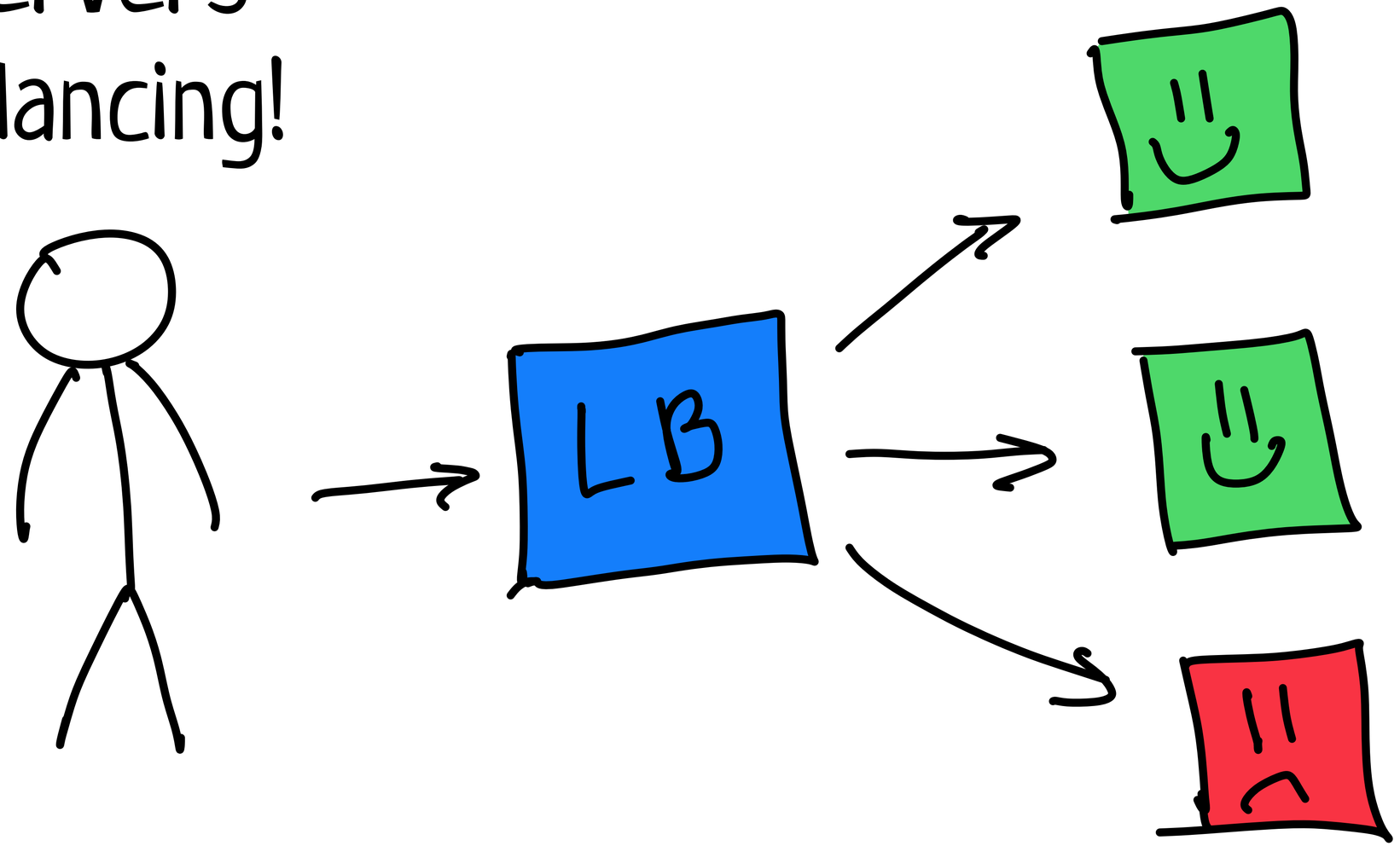
More servers
More balancing!



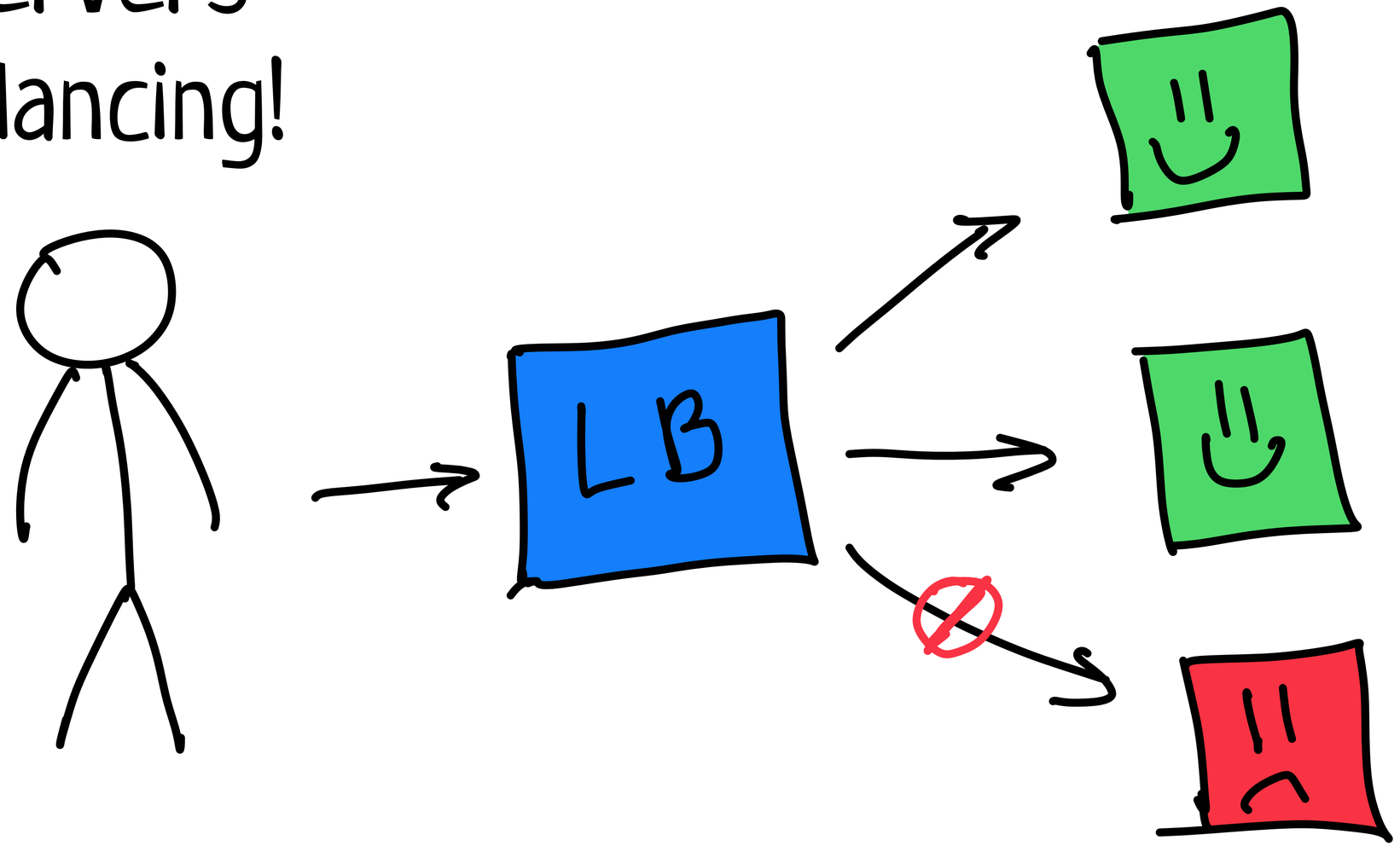
More servers
More balancing!



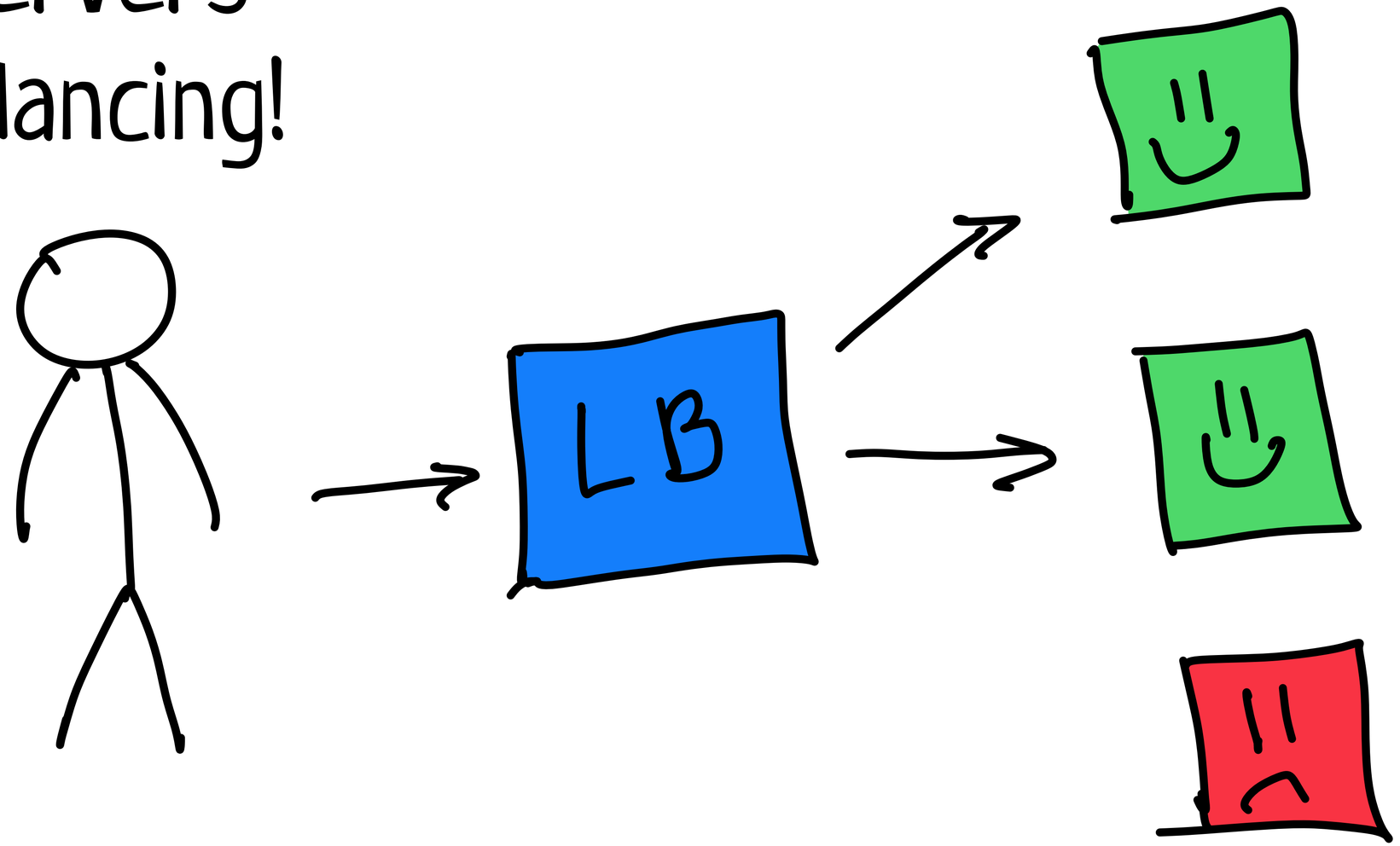
More servers
More balancing!



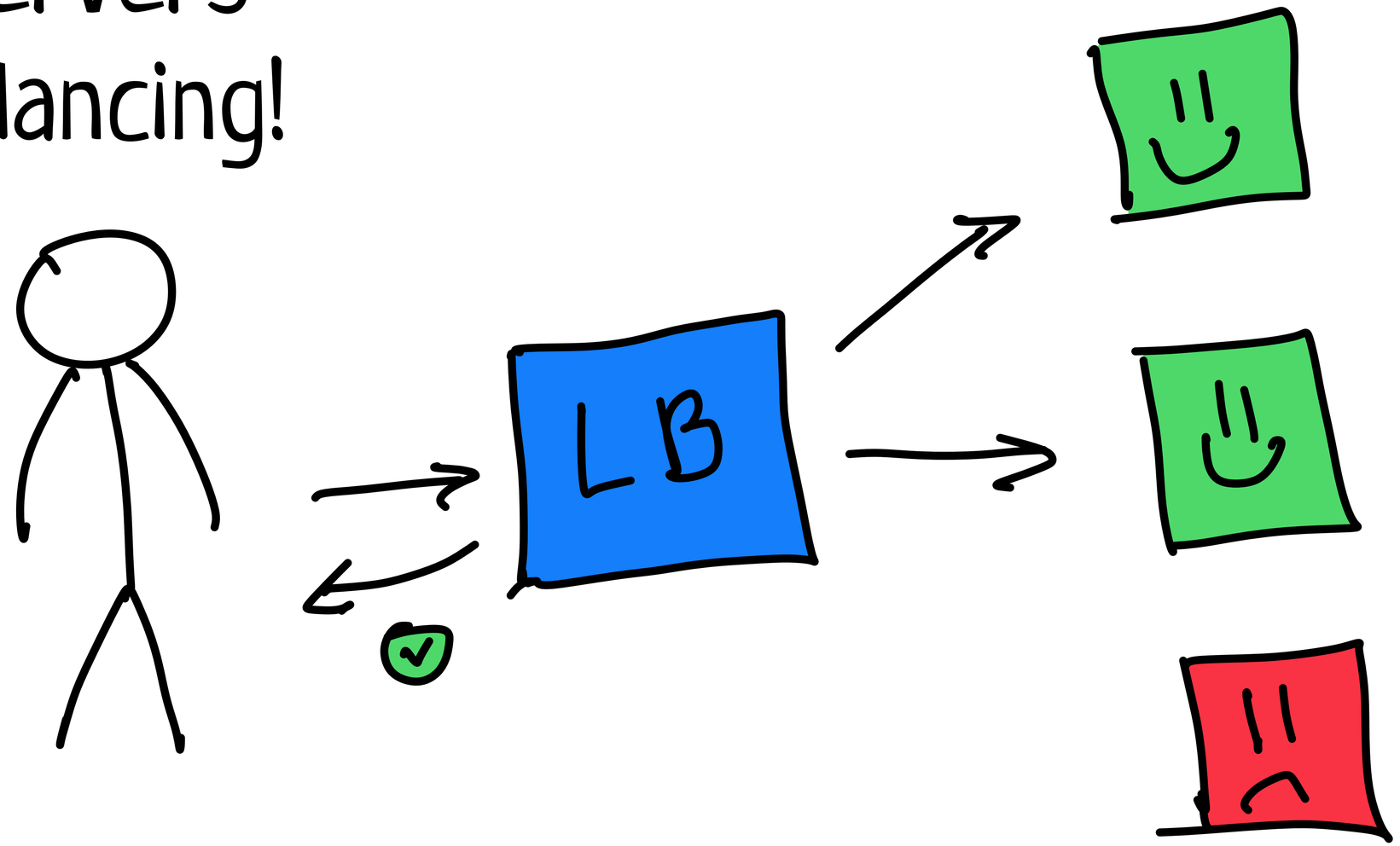
More servers
More balancing!



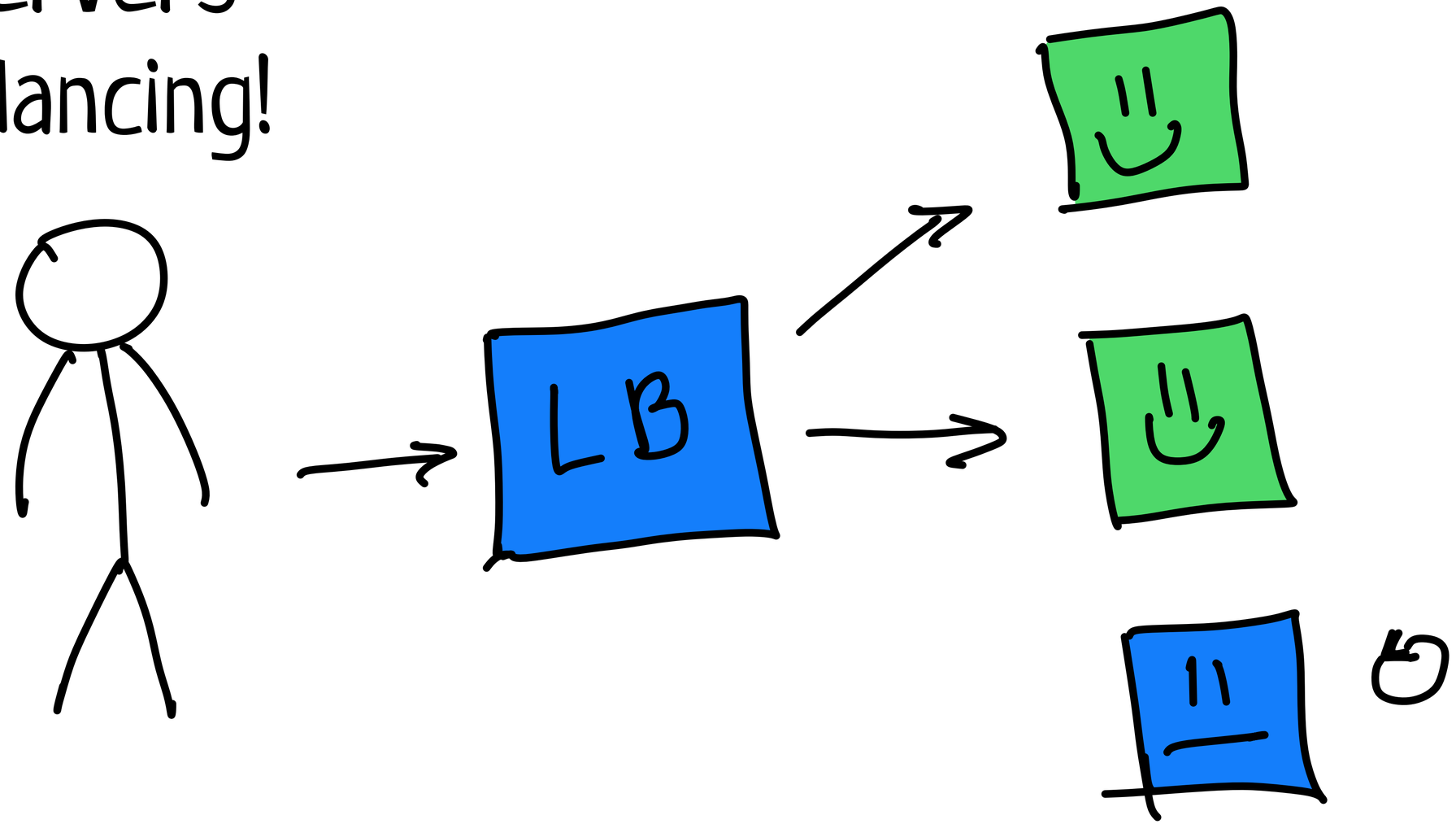
More servers
More balancing!



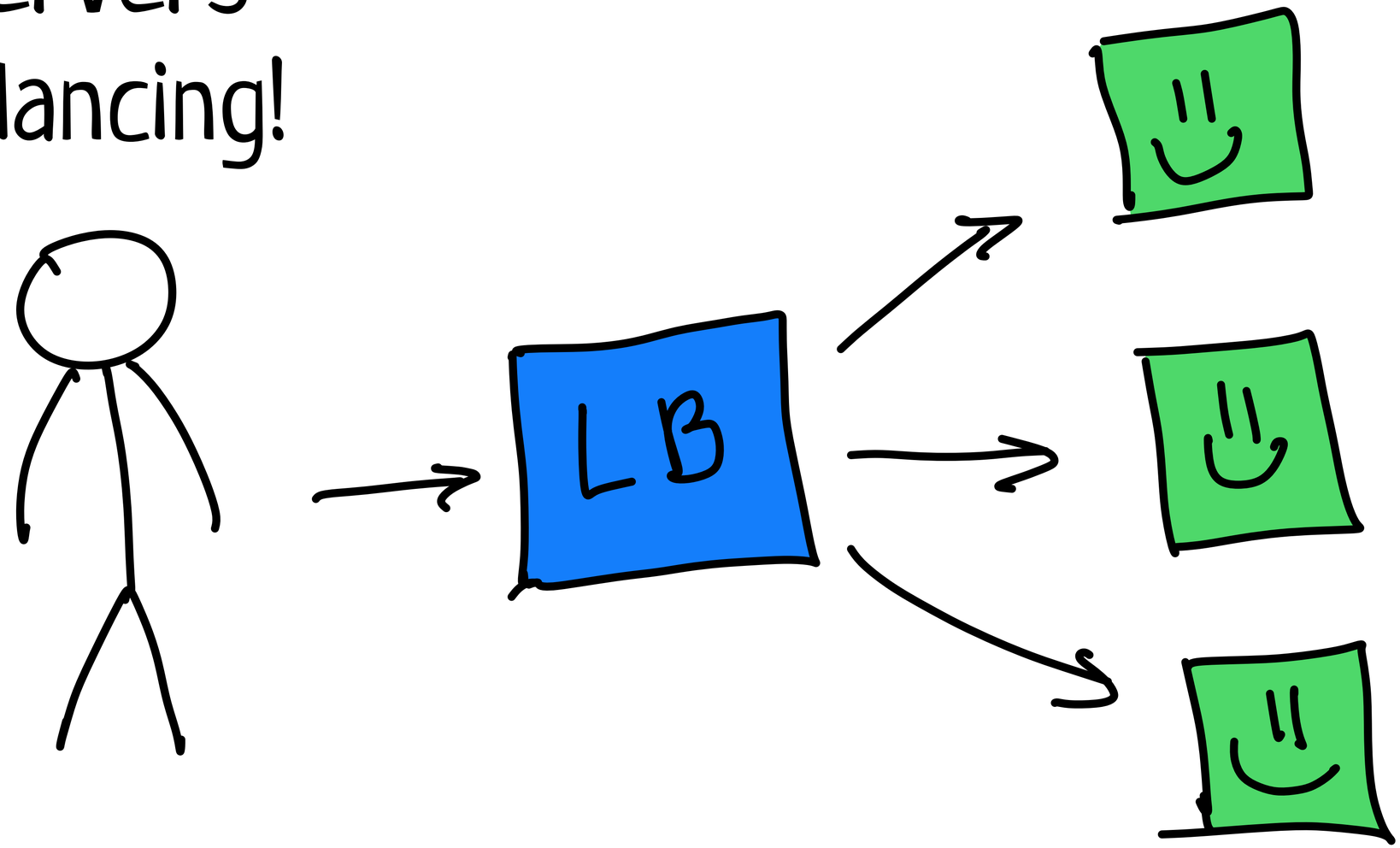
More servers
More balancing!



More servers
More balancing!



More servers
More balancing!



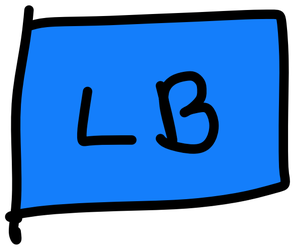
Load Balancer

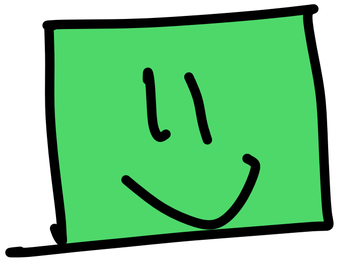
- NGINX
- Azure Traffic Manager
- AWS Elastic Load Balancing
- GC Load Balancing

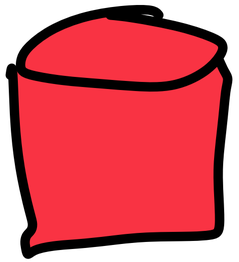
UUID

Universal Unique Identifier/Atomic Unique Identifiers

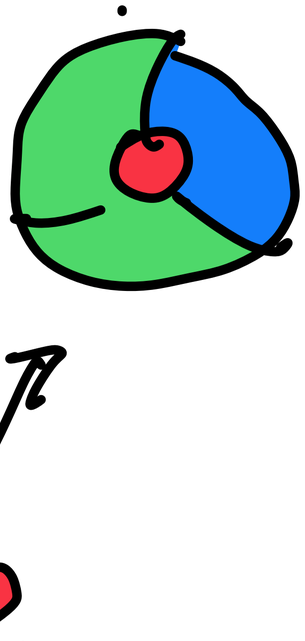
No UUIDs
Per transaction



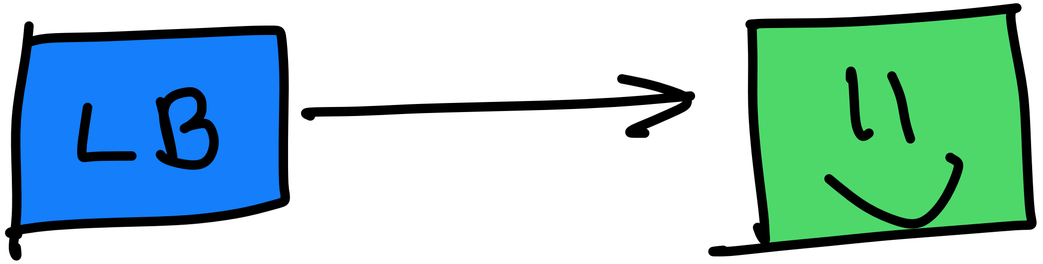


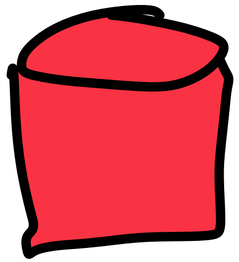


Logs

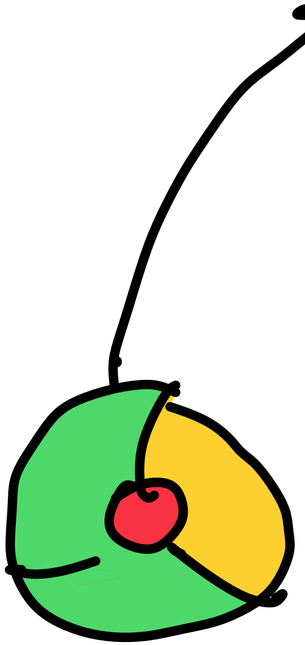


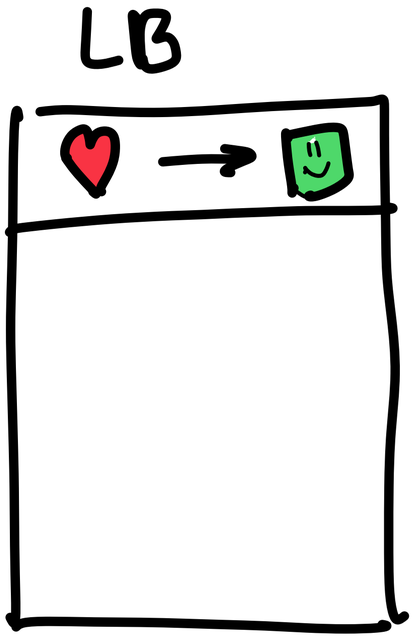
No UUIDs
Per transaction



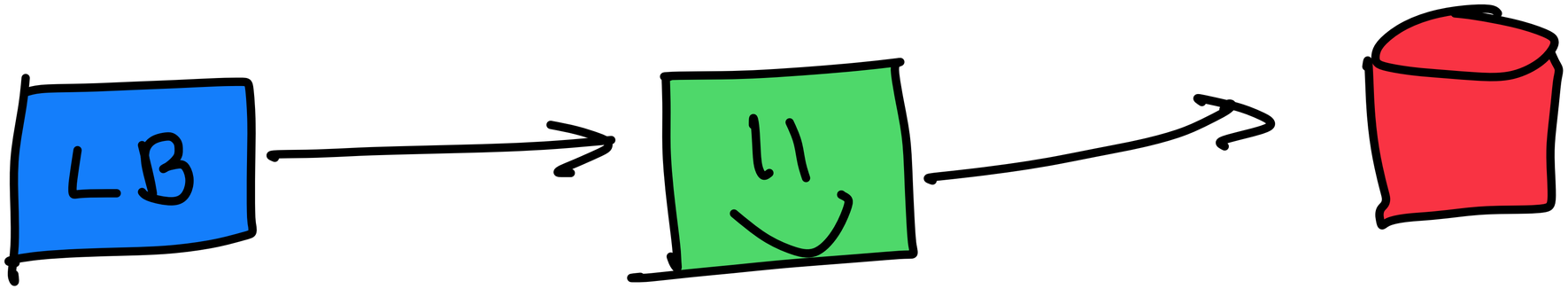


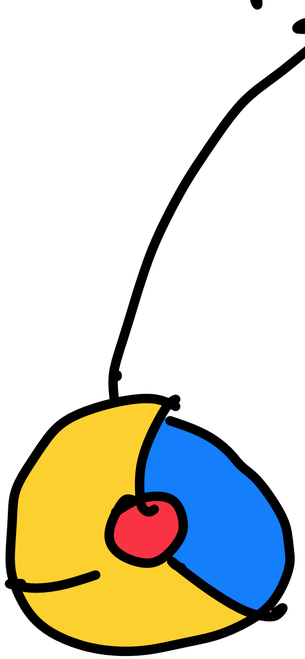
Logs



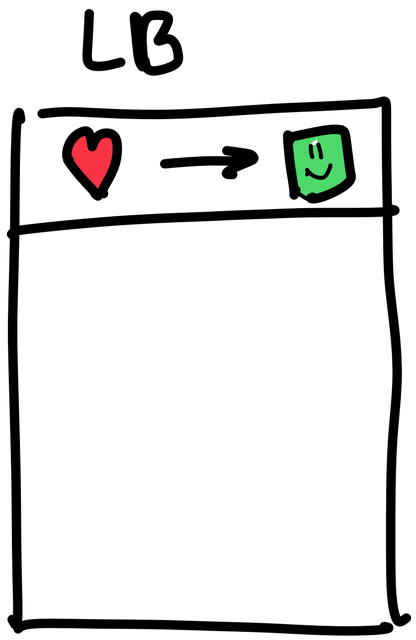


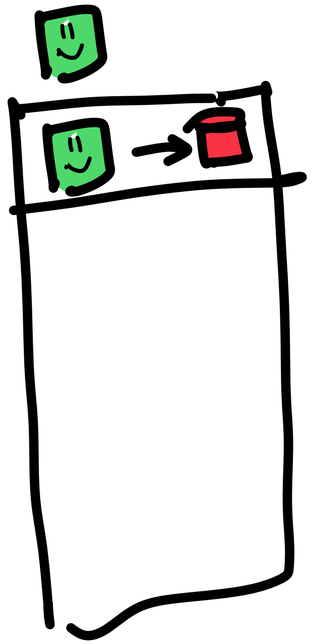
No UUIDs
Per transaction

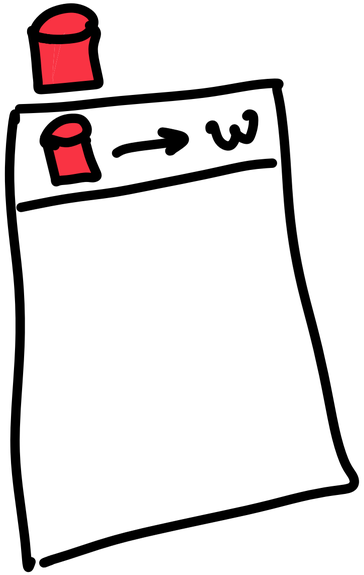




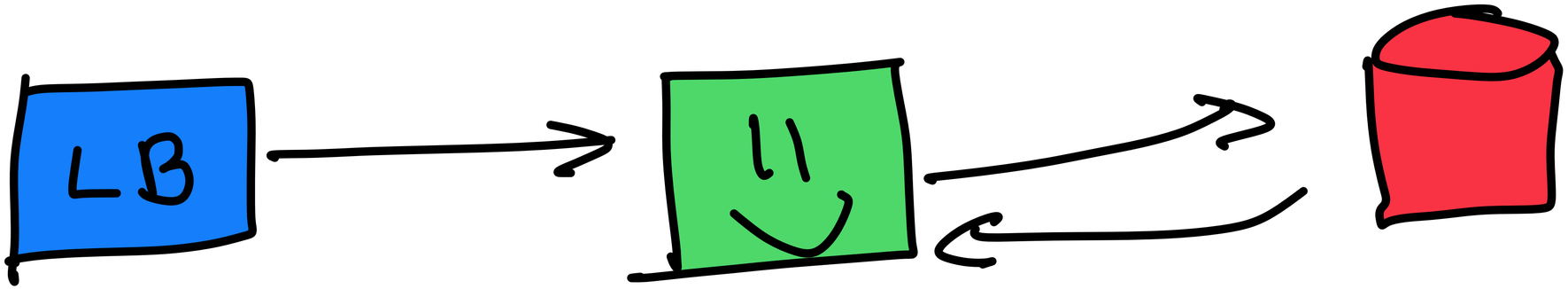
Logs

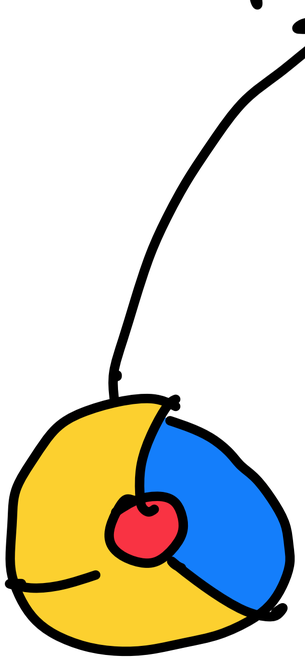




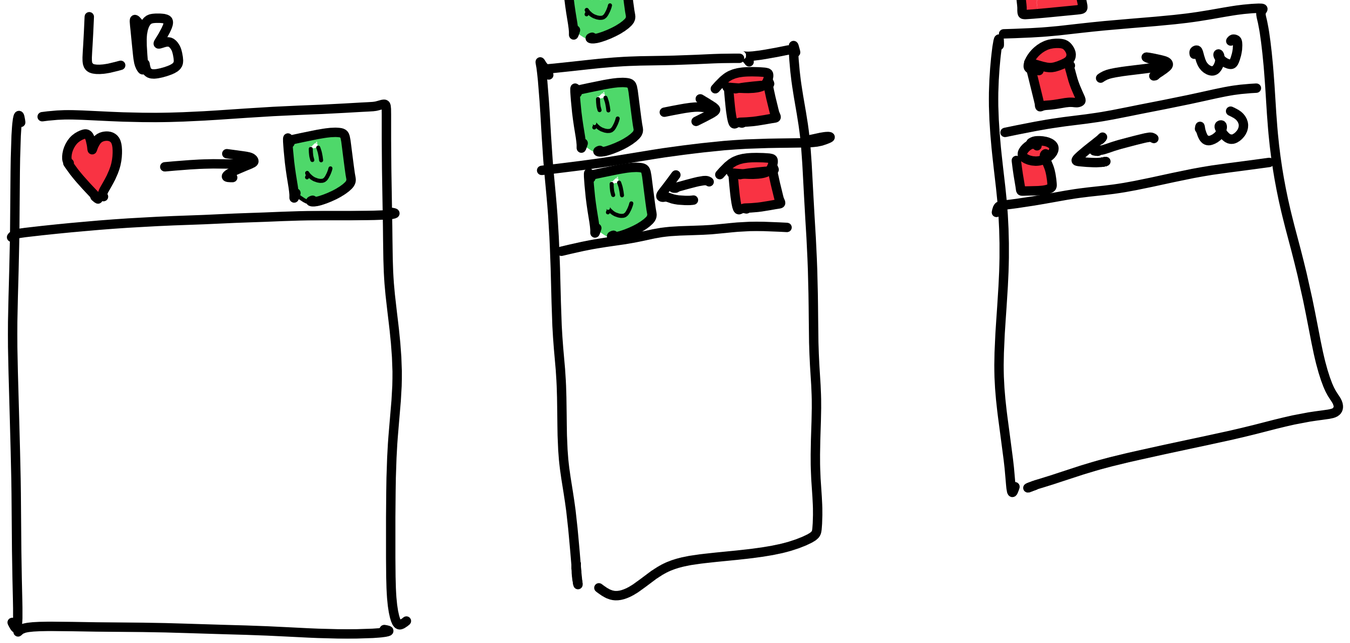


No UUIDs
Per transaction

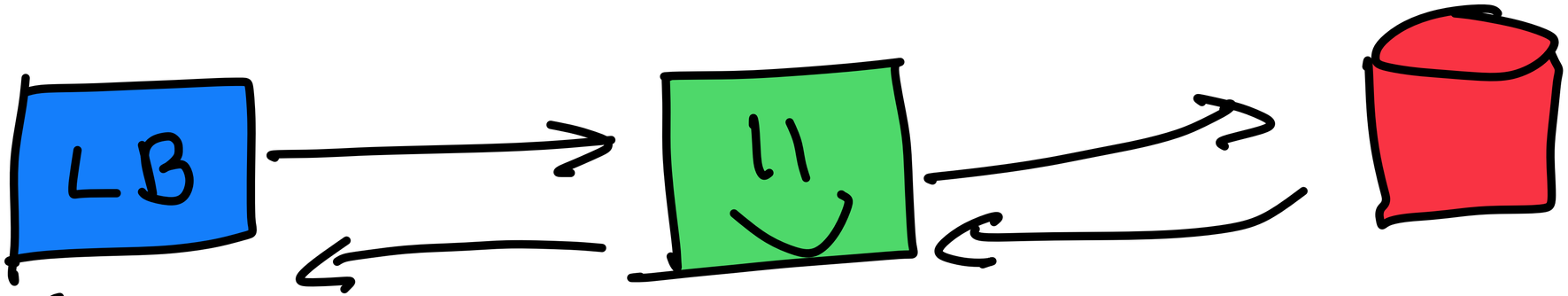


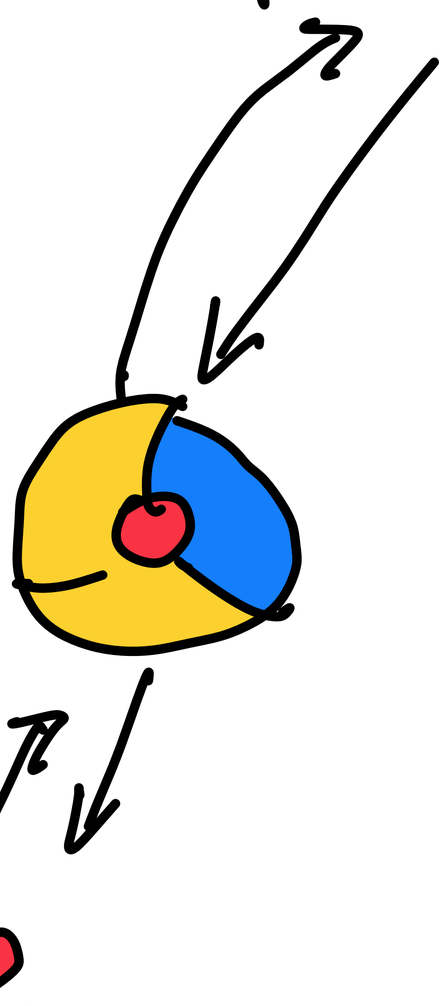


Logs

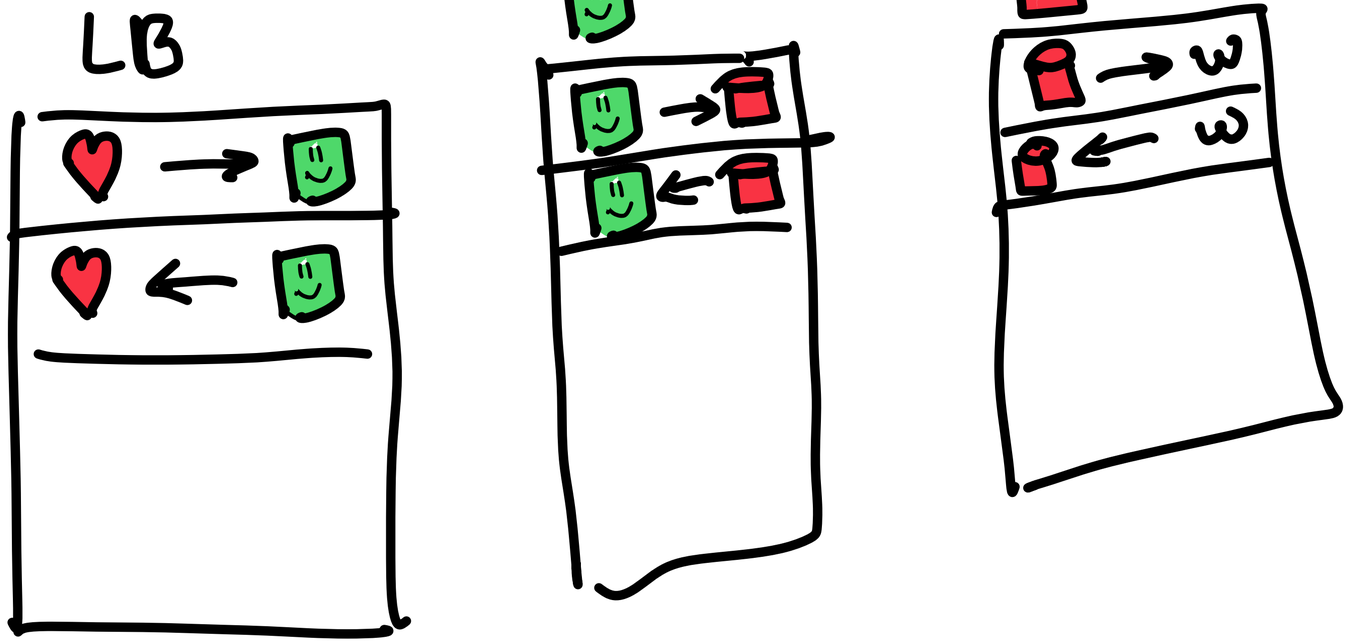


No UUIDs
Per transaction

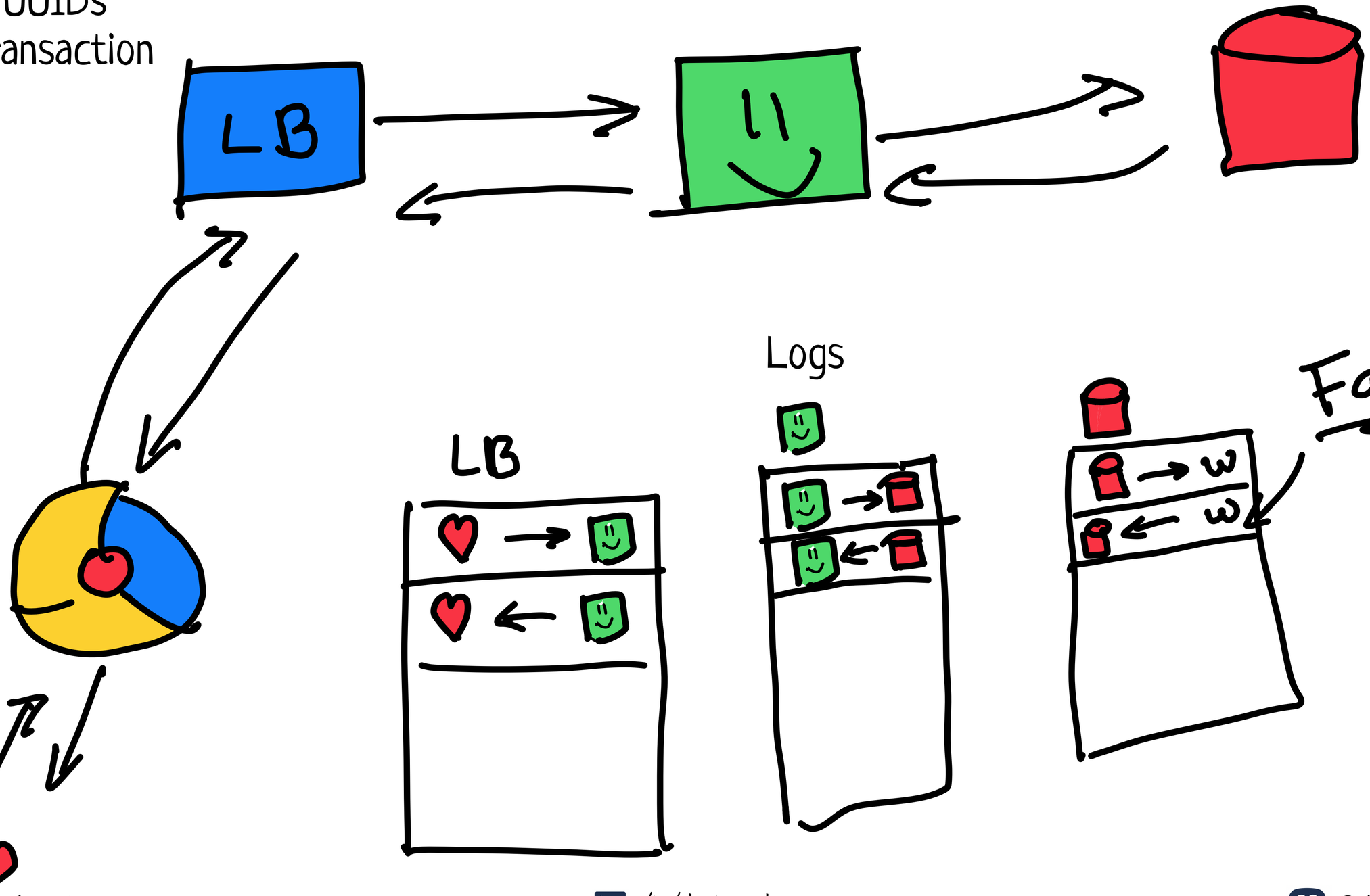


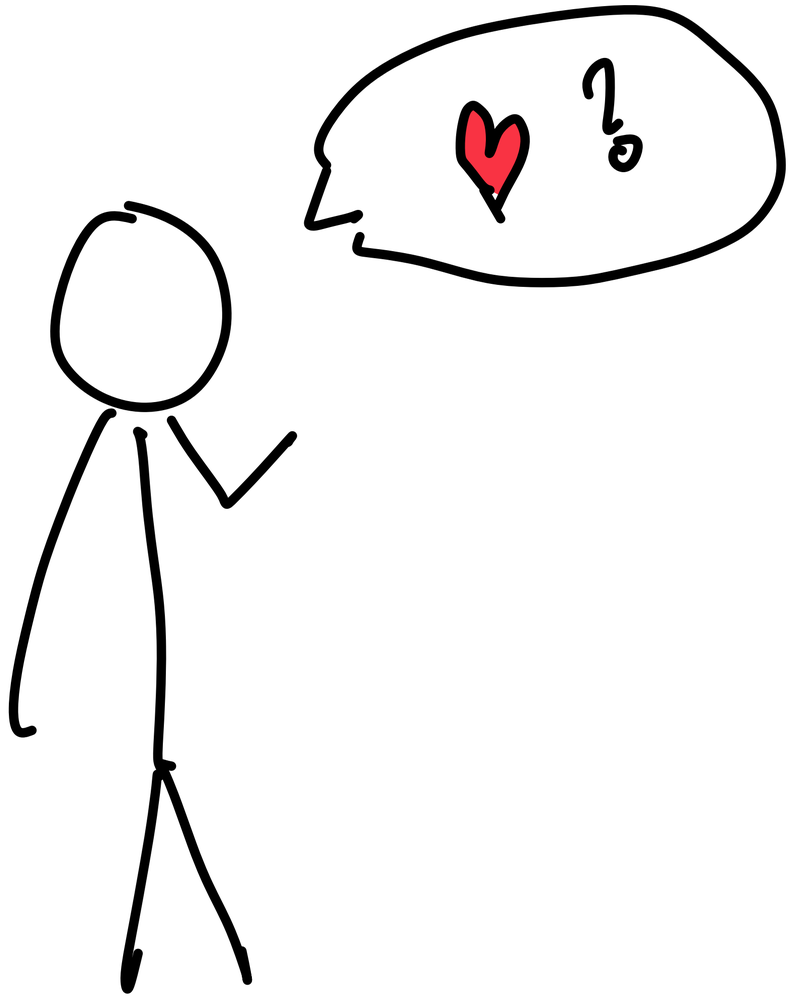


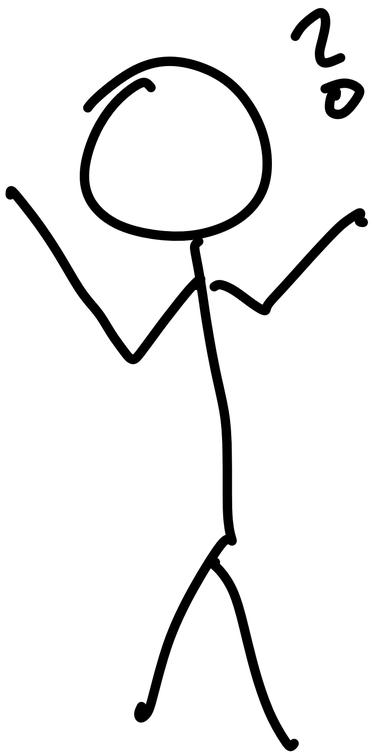
Logs



No UUIDs
Per transaction







Error 502

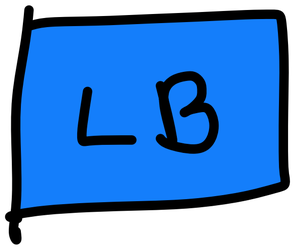
Ray ID: 2936e43f56e7440e • 2016-04-14 11:41:57 UTC

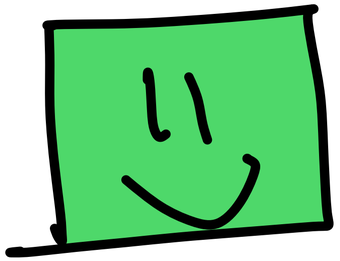
Bad gateway

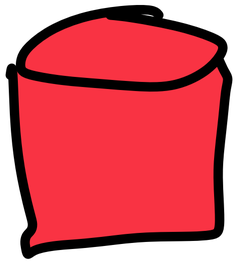
Ray ID: 2936e43f56e7440e • 2016-04-14 11:41:5

You	POP: undef	.org
Browser	CloudFlare	Host
Working	Working	Error

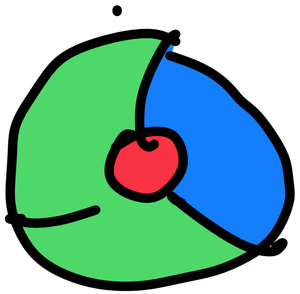
Atomic UUIDs

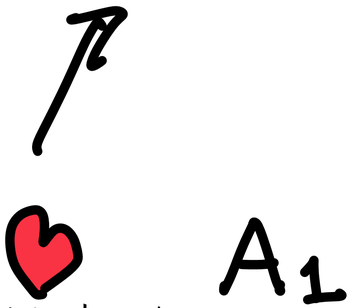




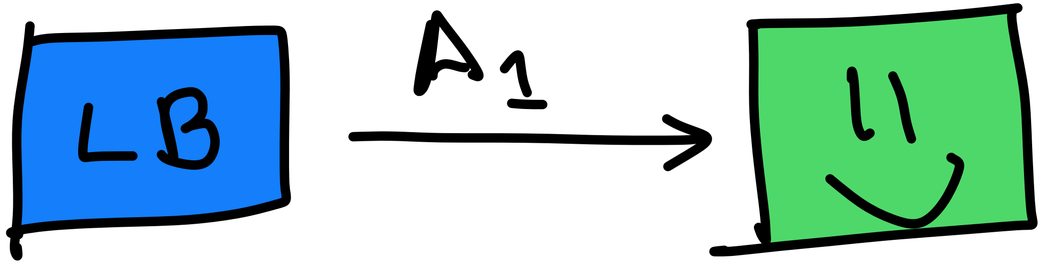


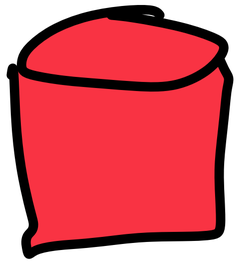
Logs

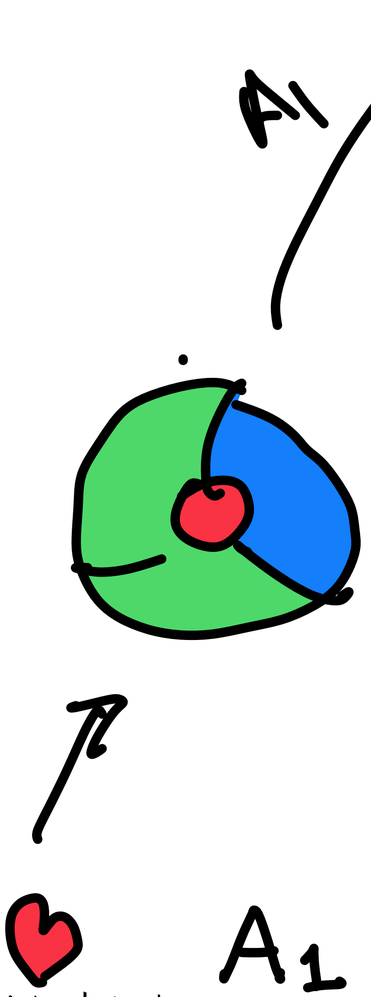




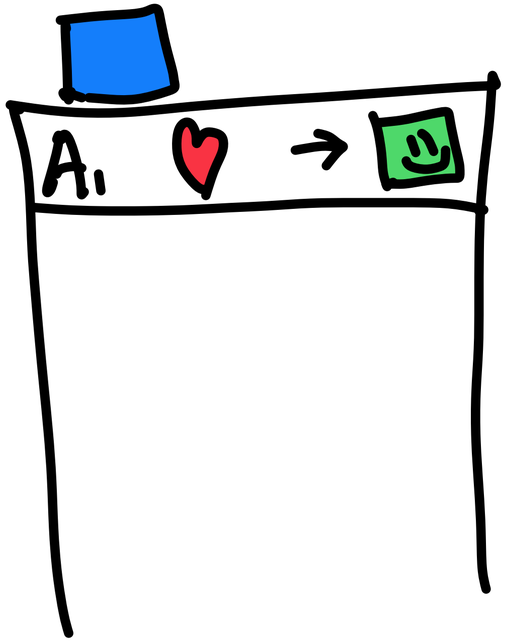
Atomic UUIDs



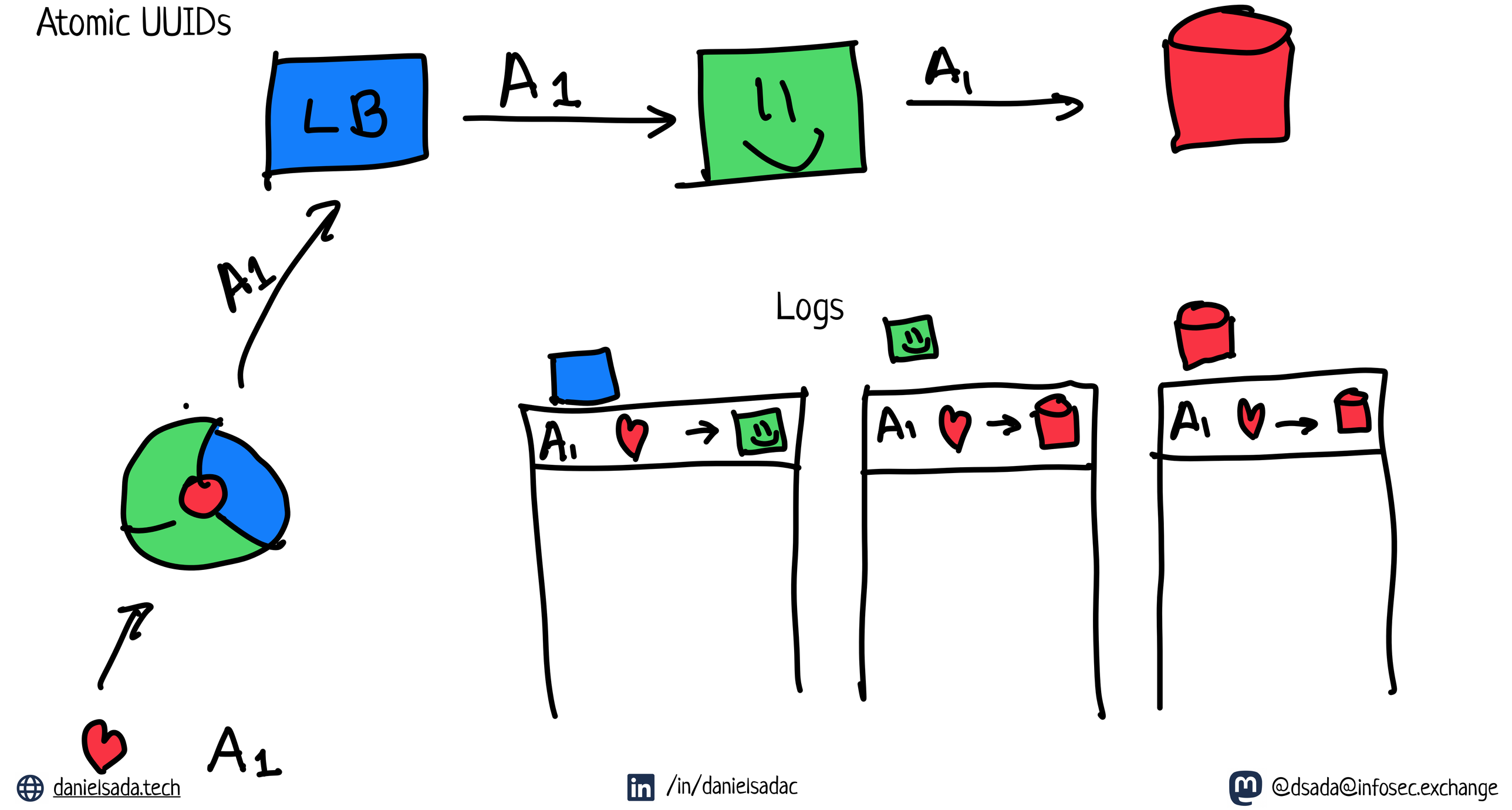




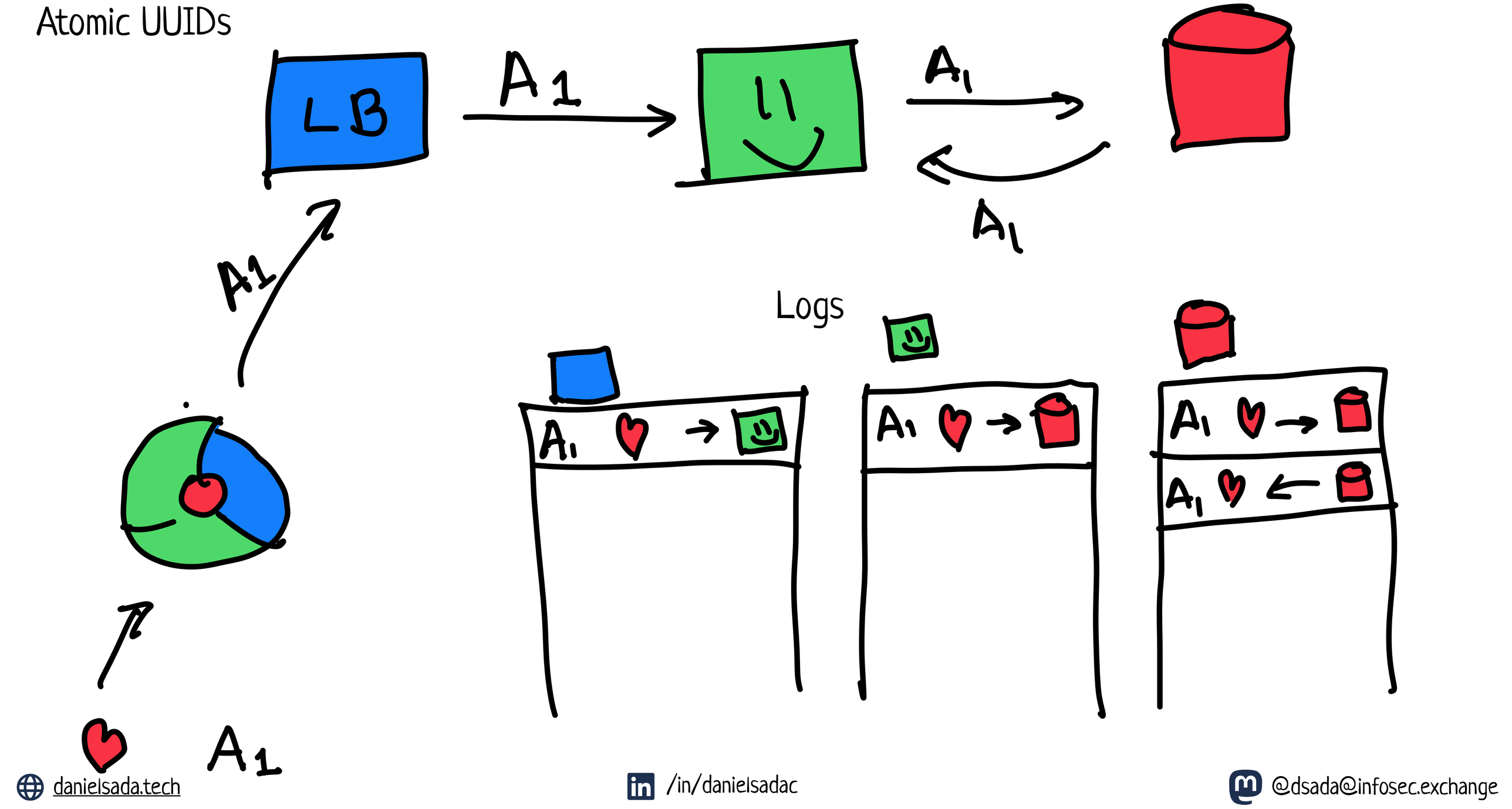
Logs



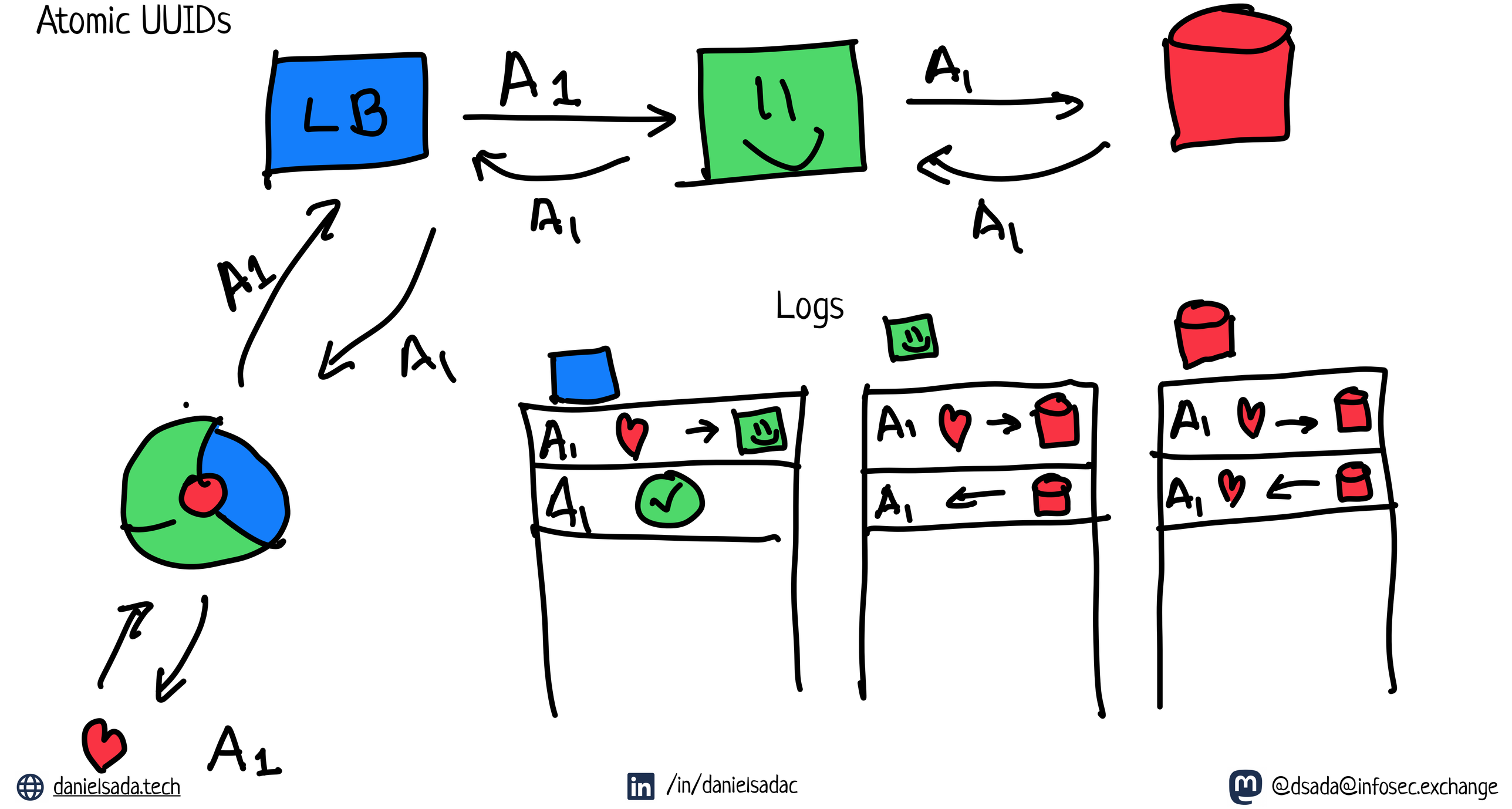
Atomic UUIDs



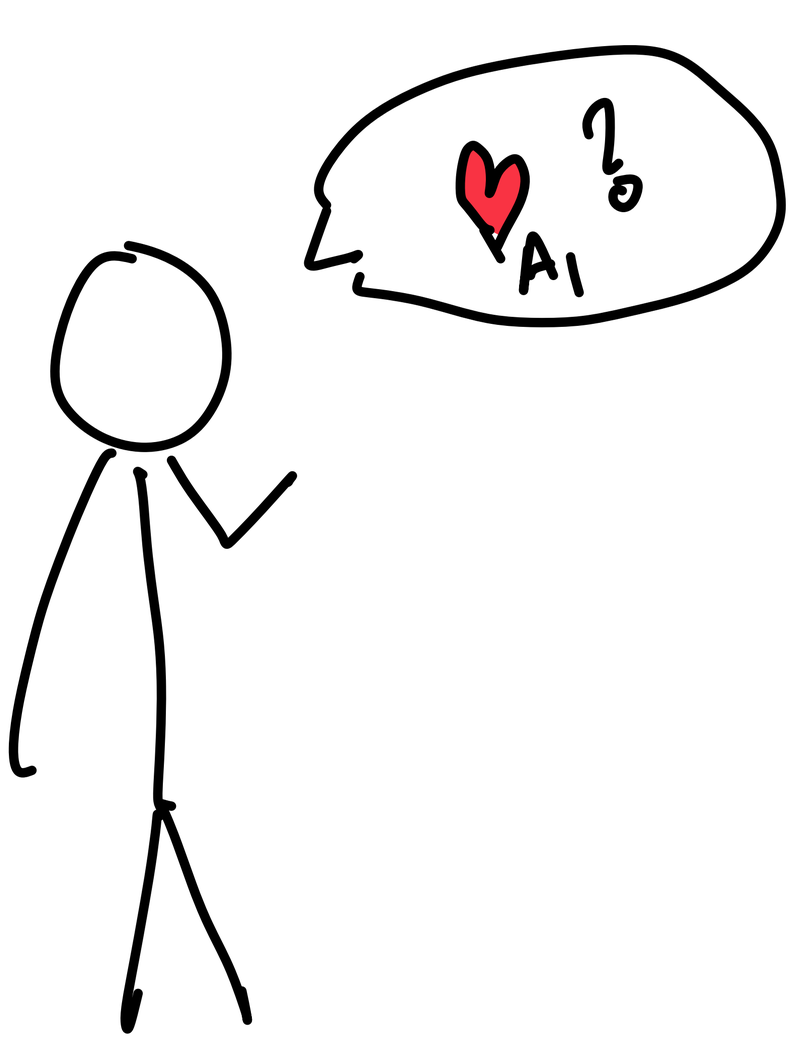
Atomic UUIDs

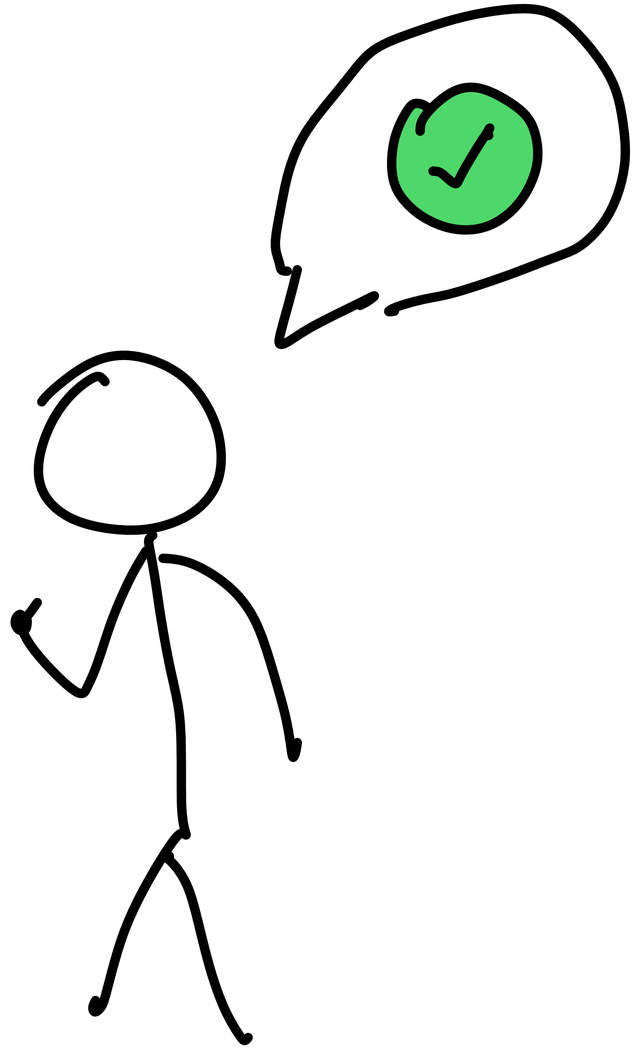


Atomic UUIDs

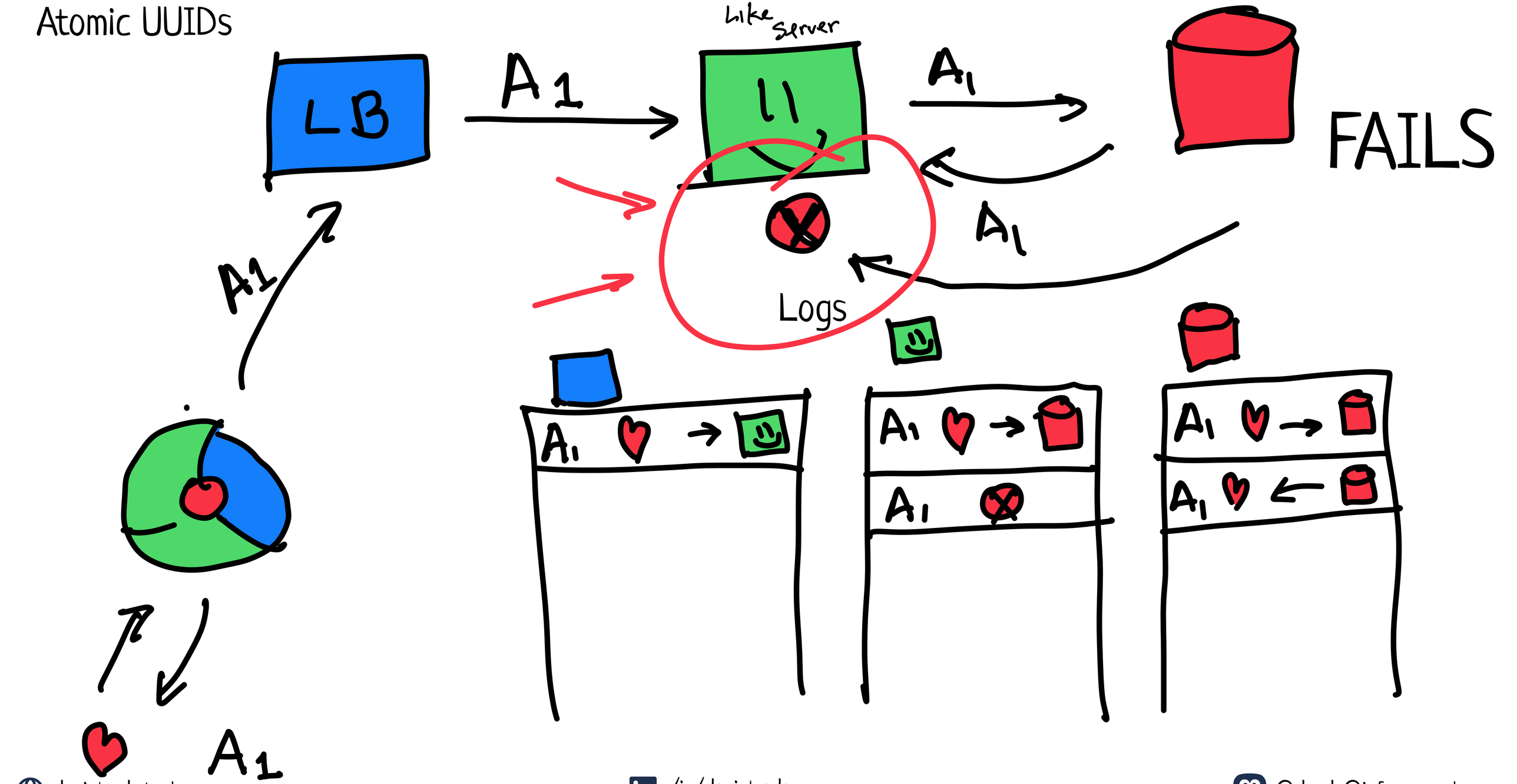


Atomic UUIDs

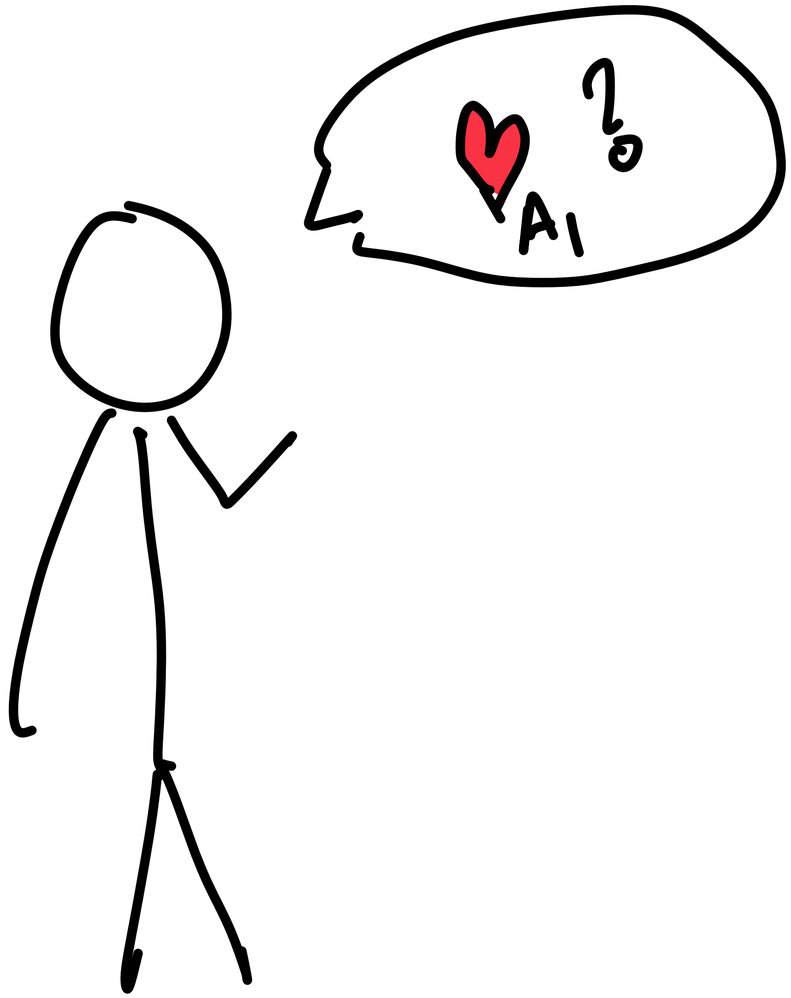




Atomic UUIDs



Atomic UUIDs



UUID

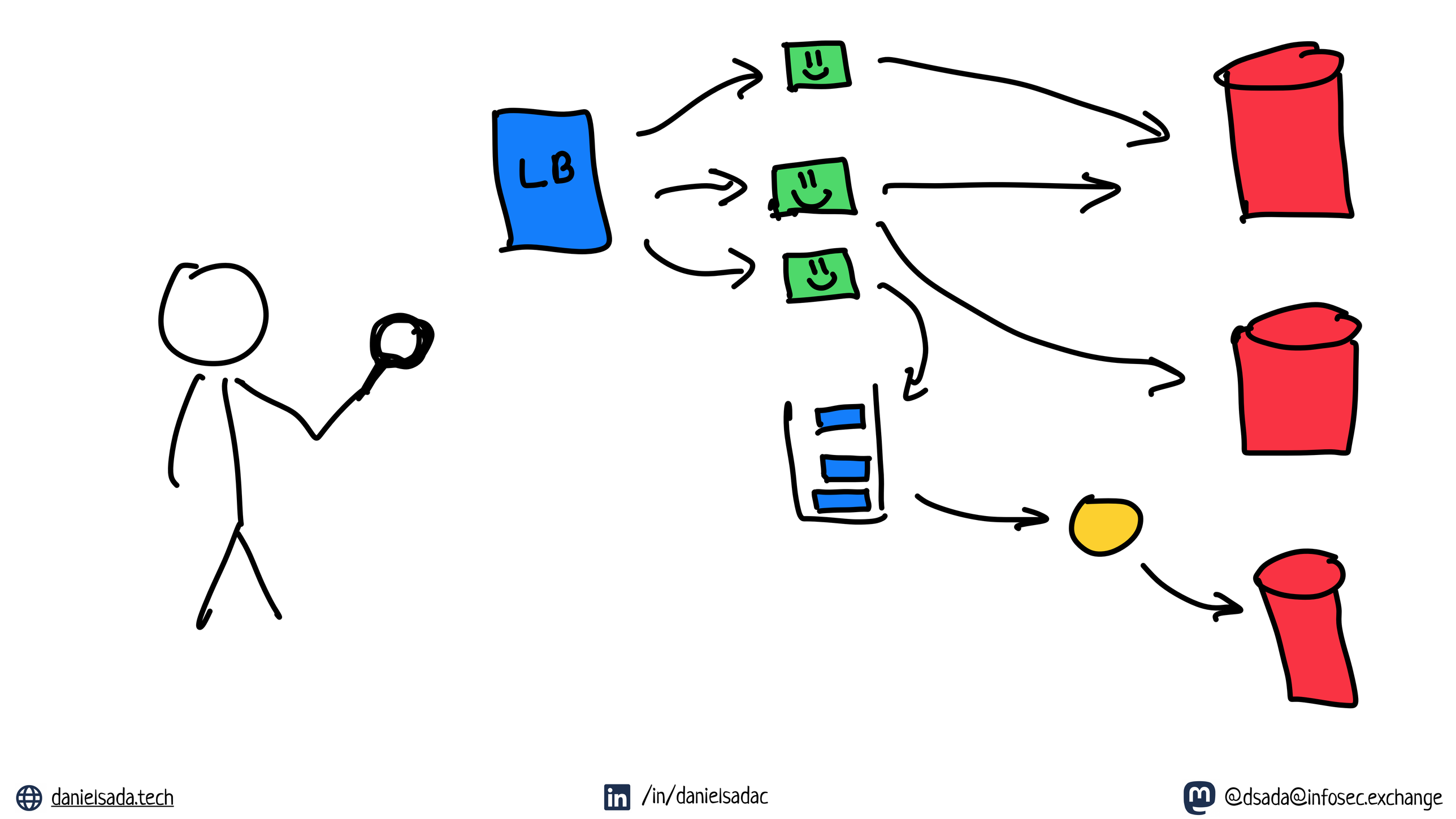
Universal Unique Identifier

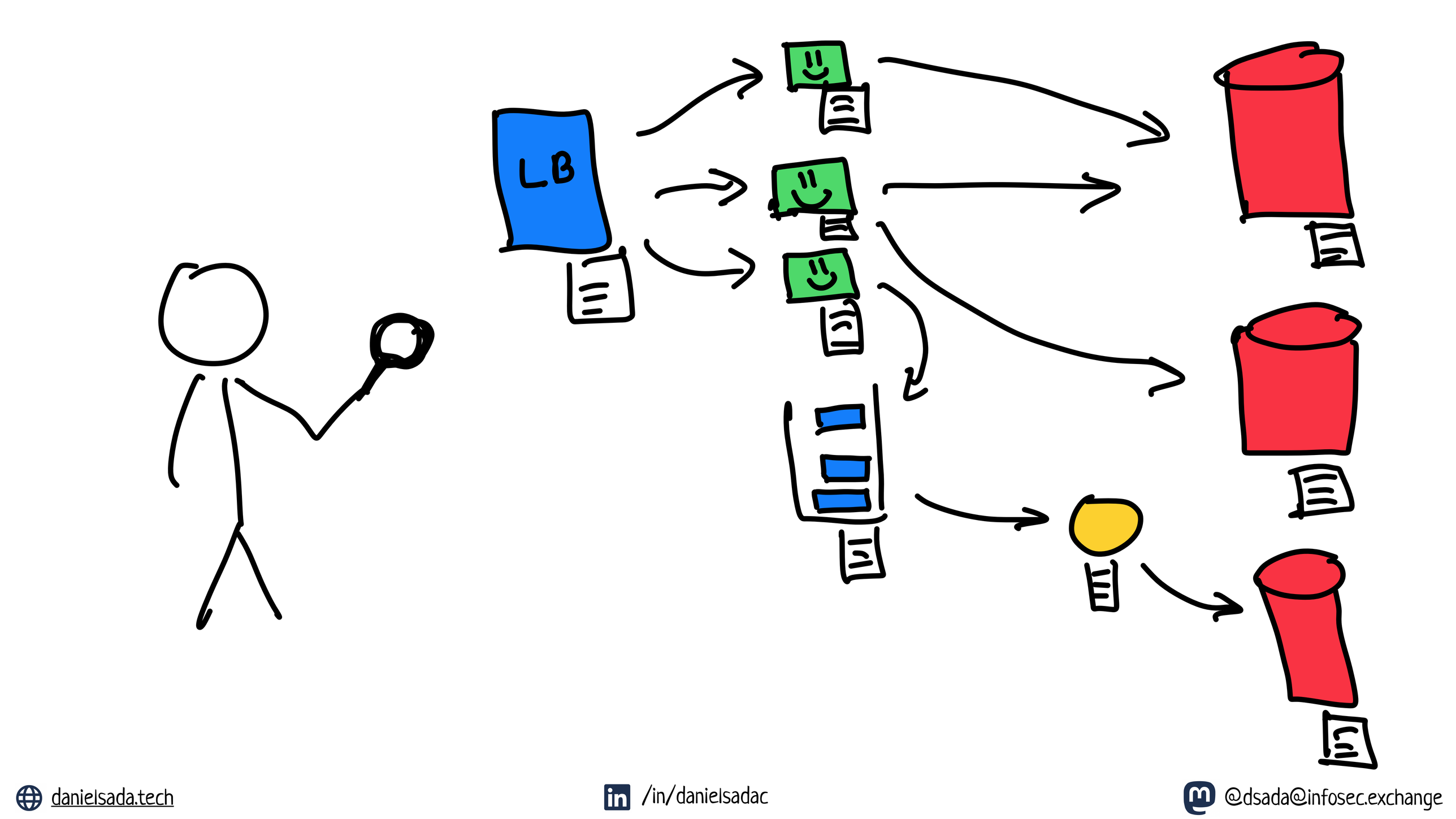
- Implement it yourself
- Avoid generating your own IDs
- You can generate them client side.

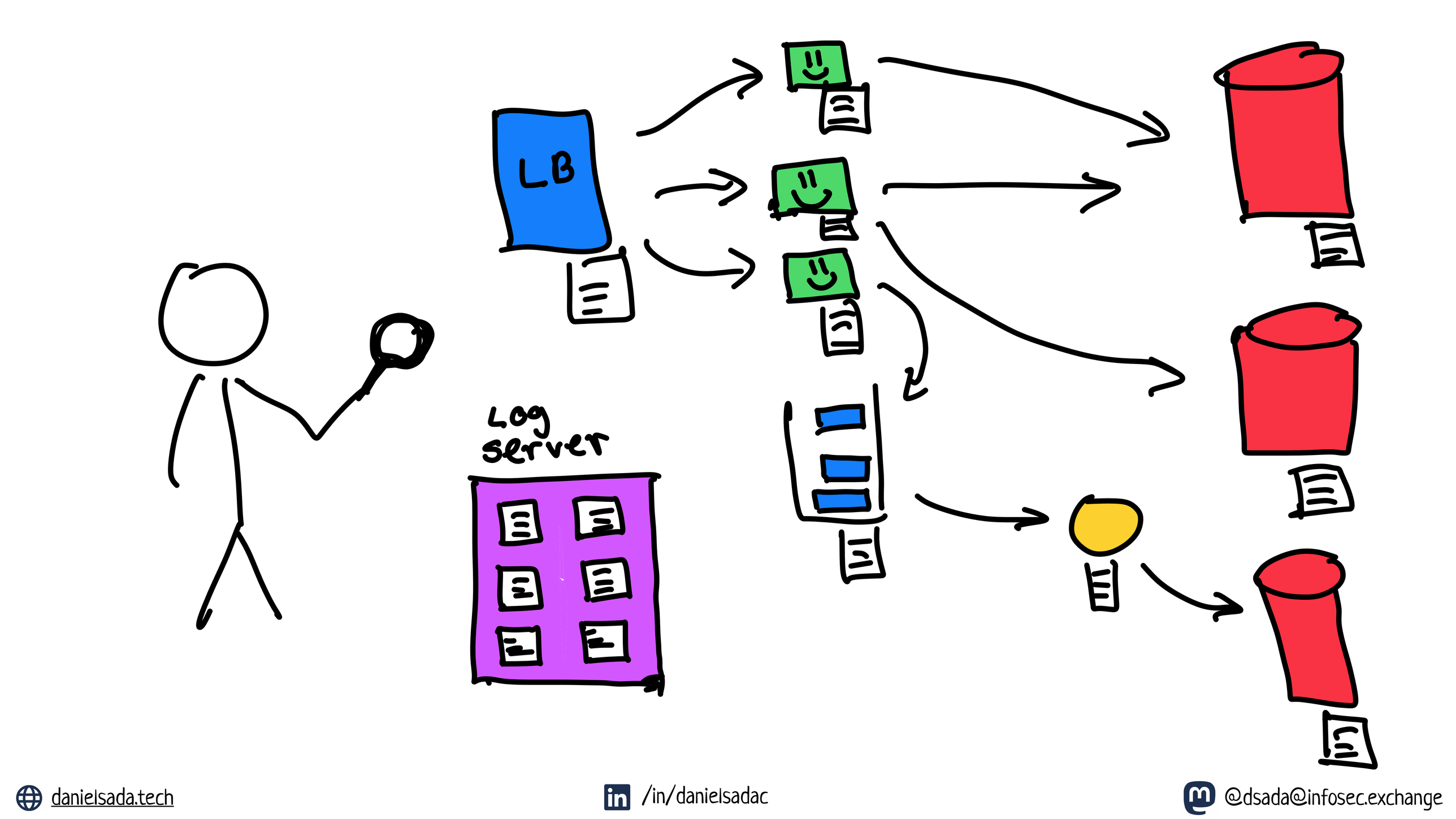
Medium

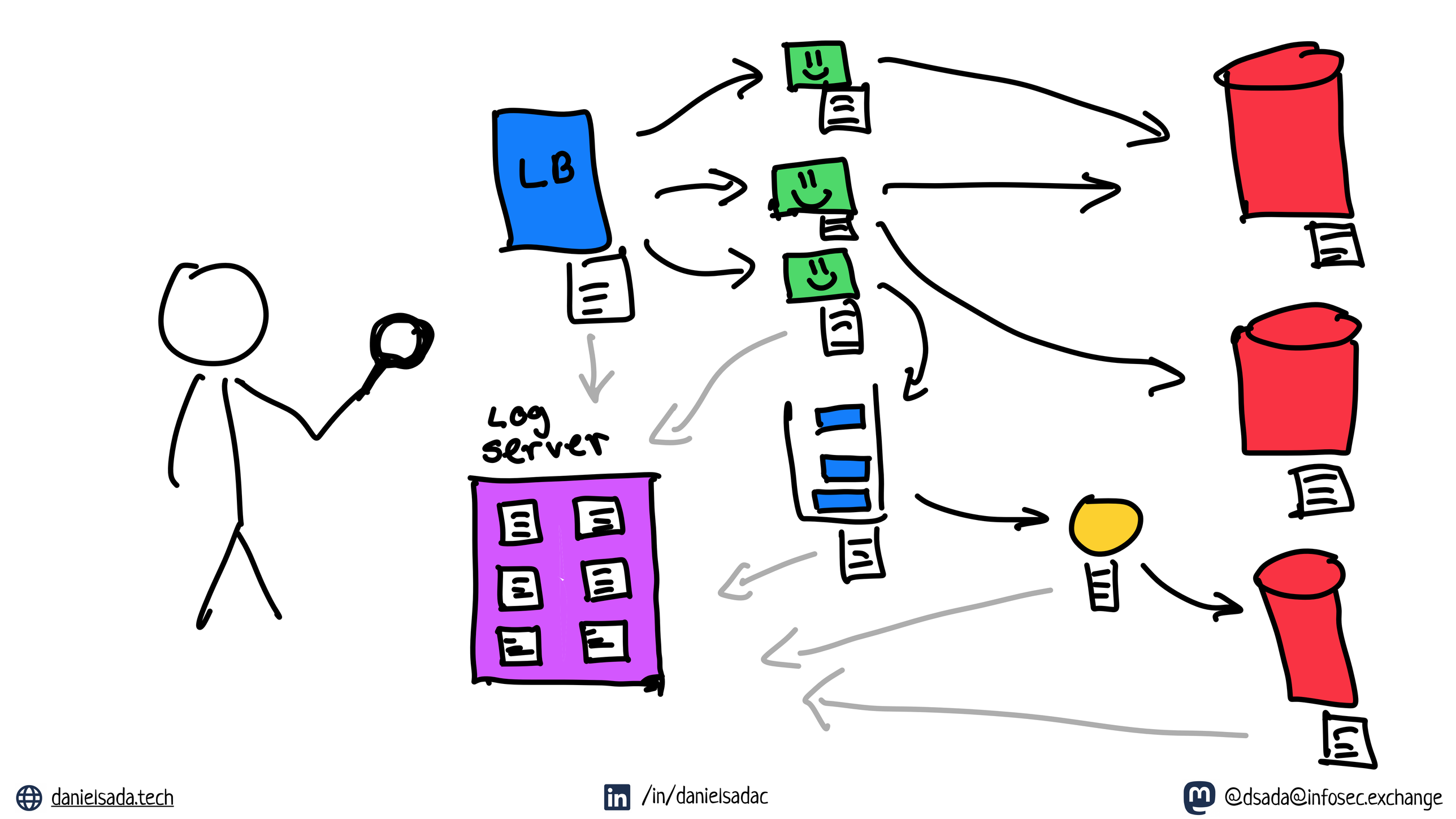
Starts getting spicy

Centralized Logging


LB

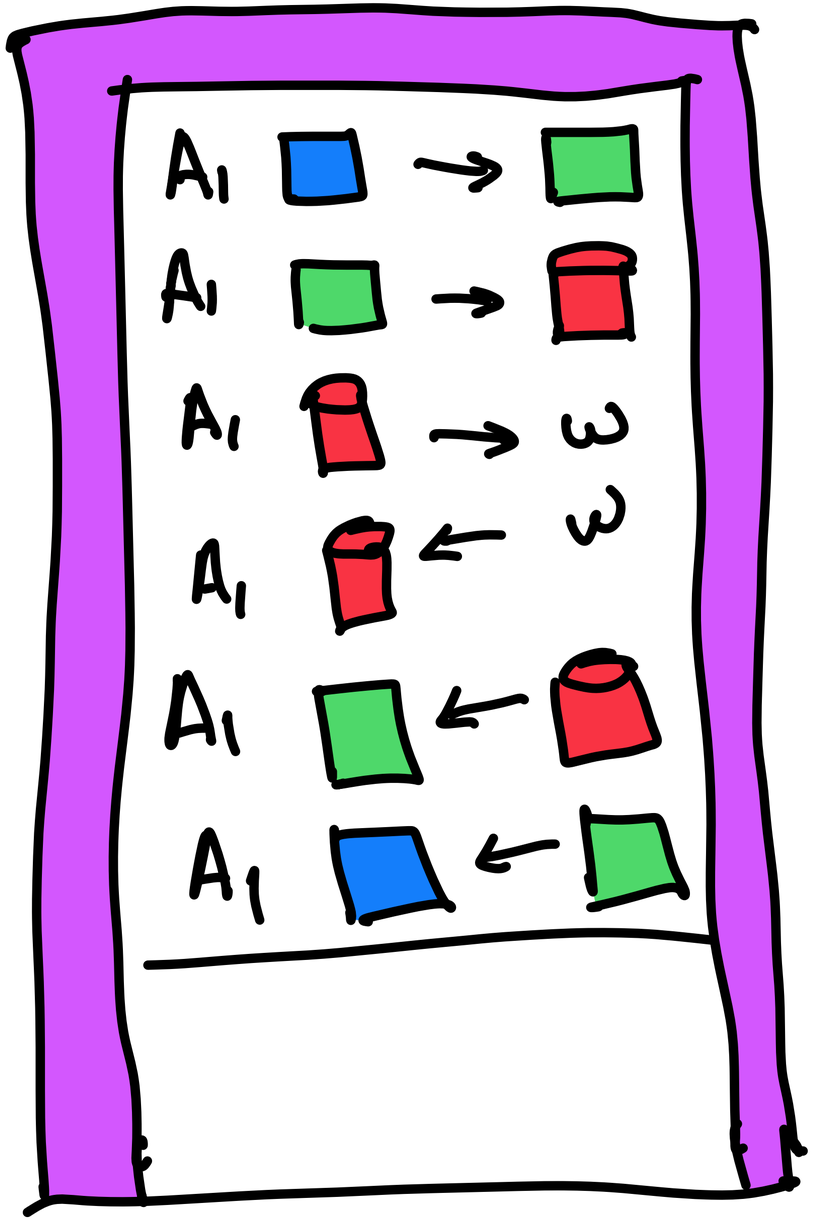

LB

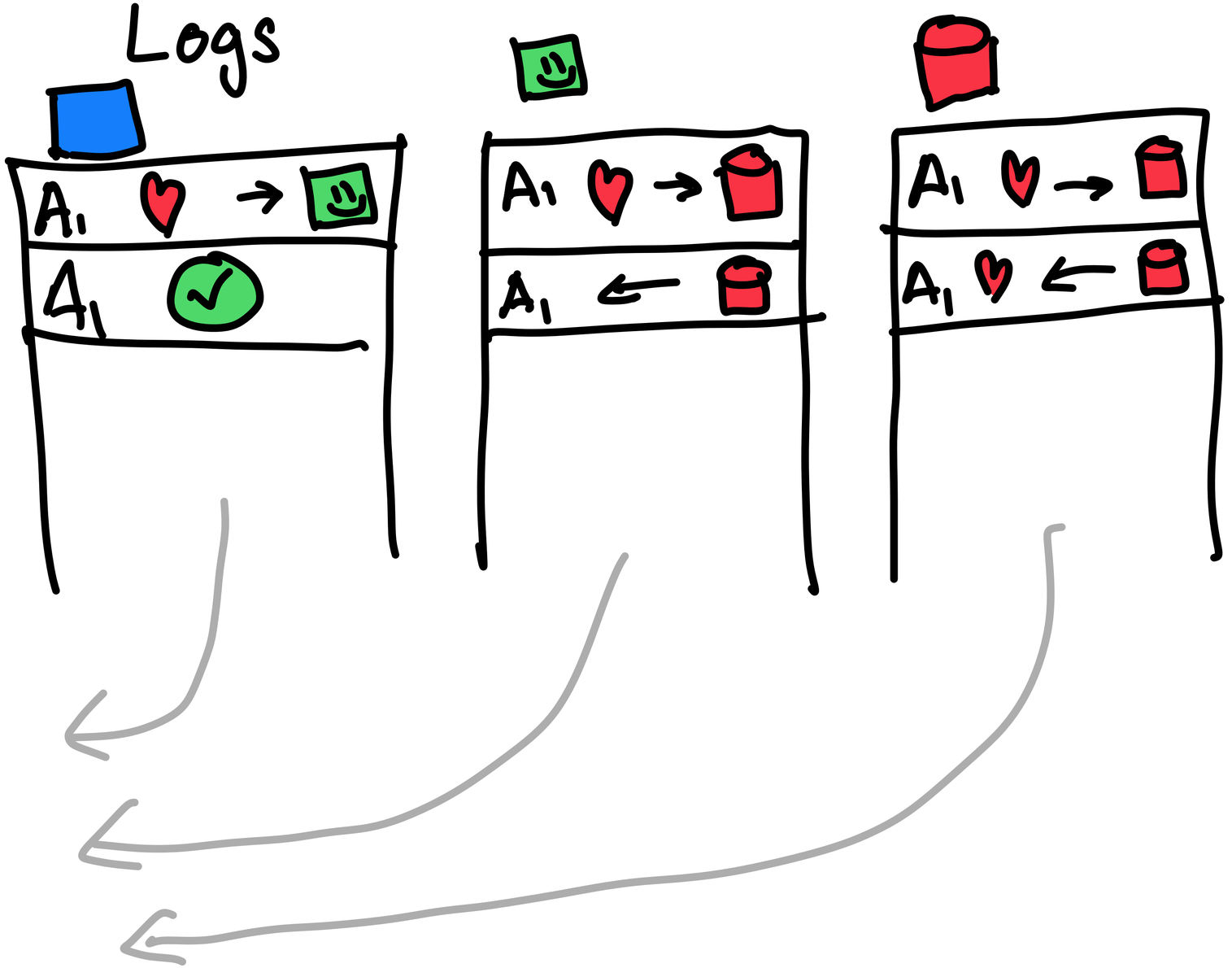

LB
Log server


LB
Log
server

Logs

Atomic UUIDs example

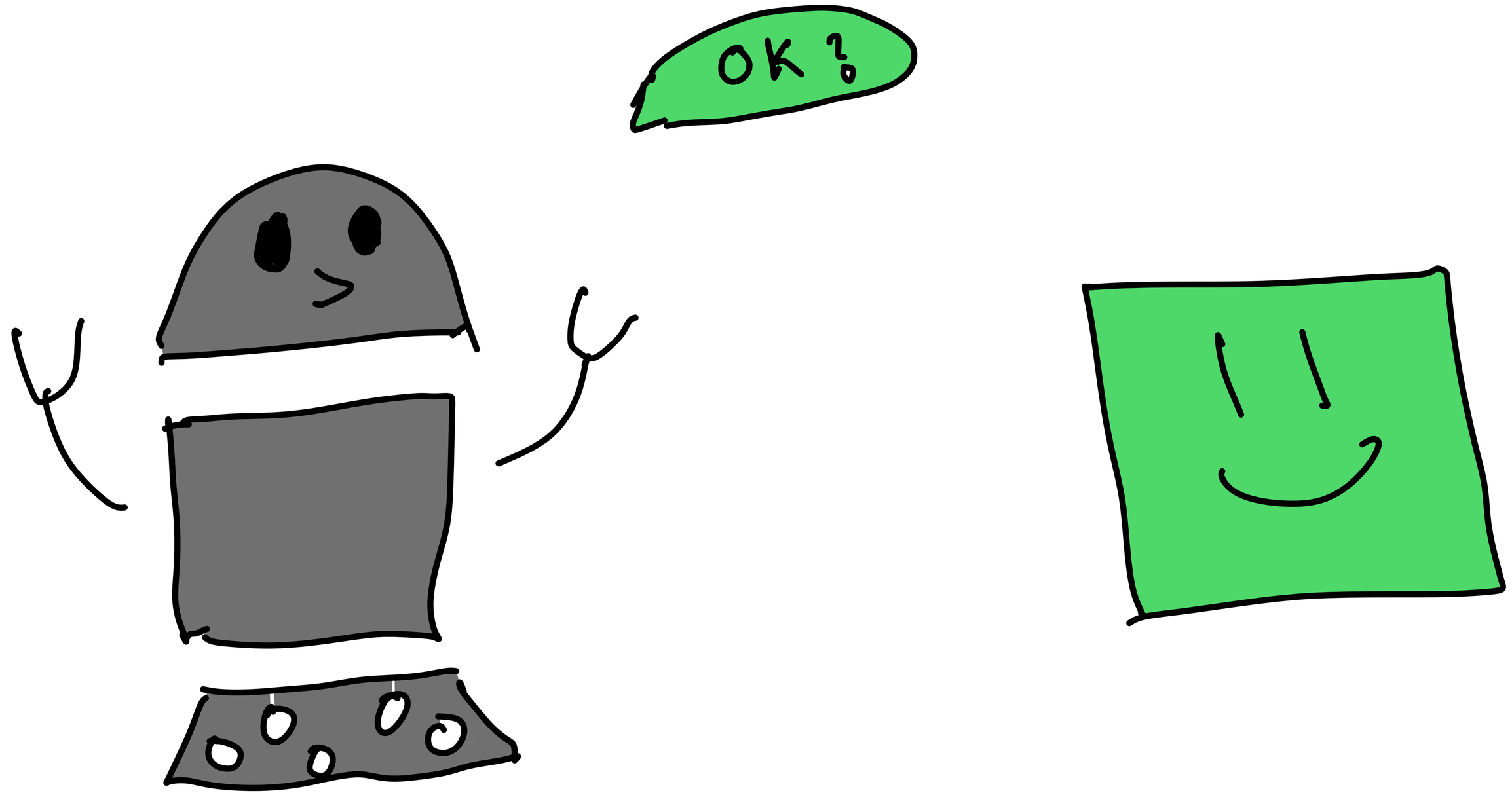


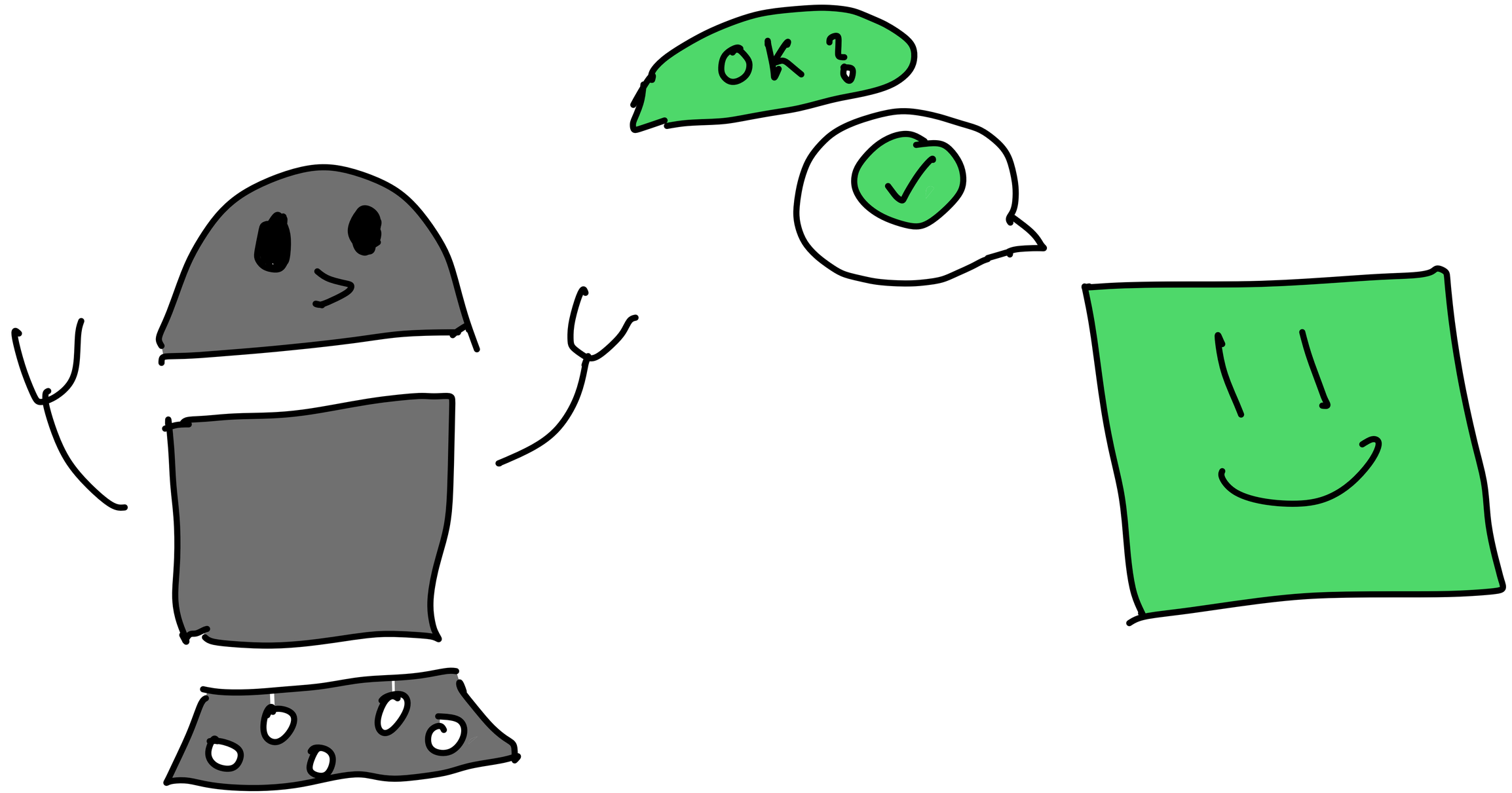


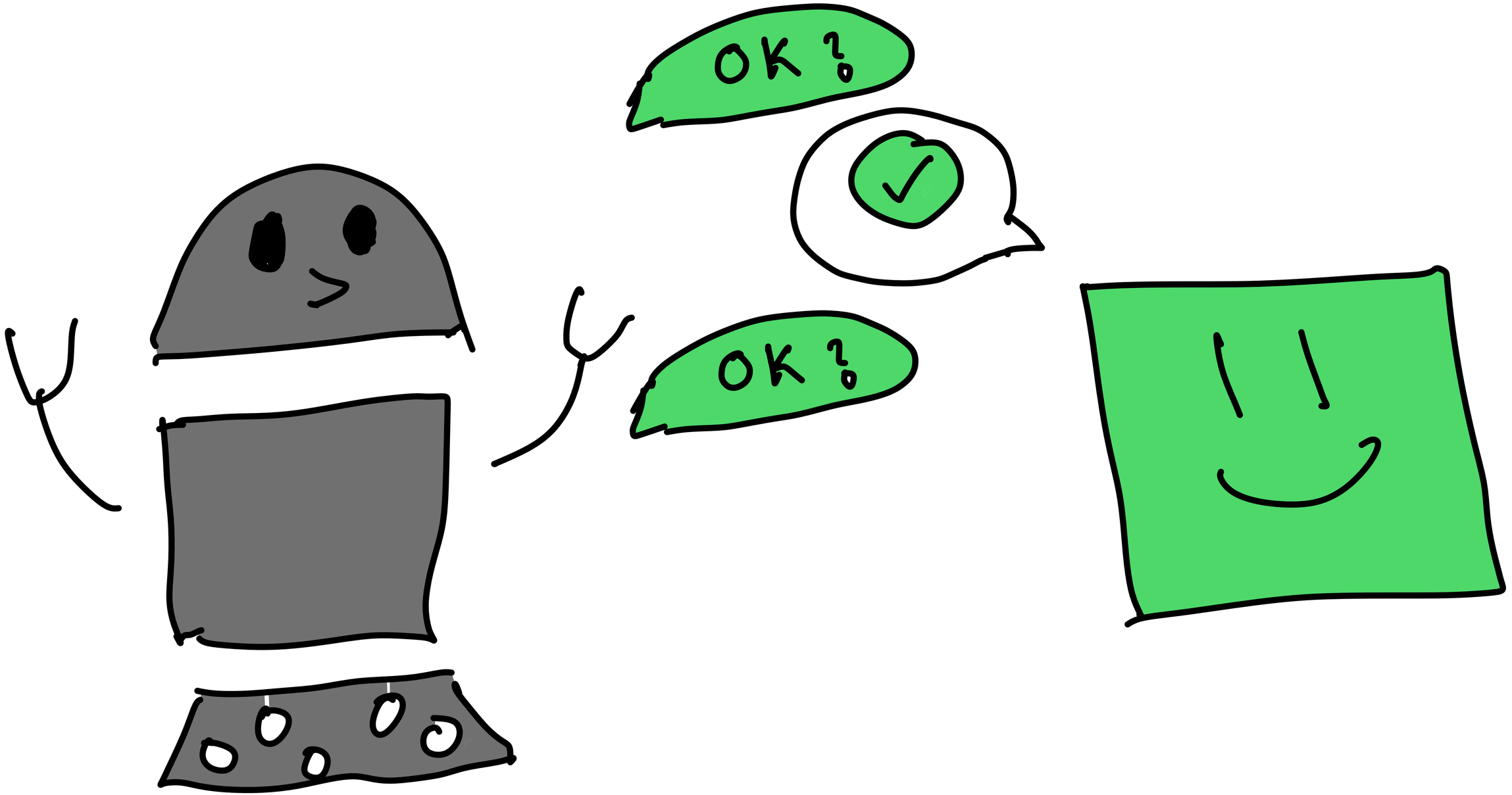
Centralized Logging

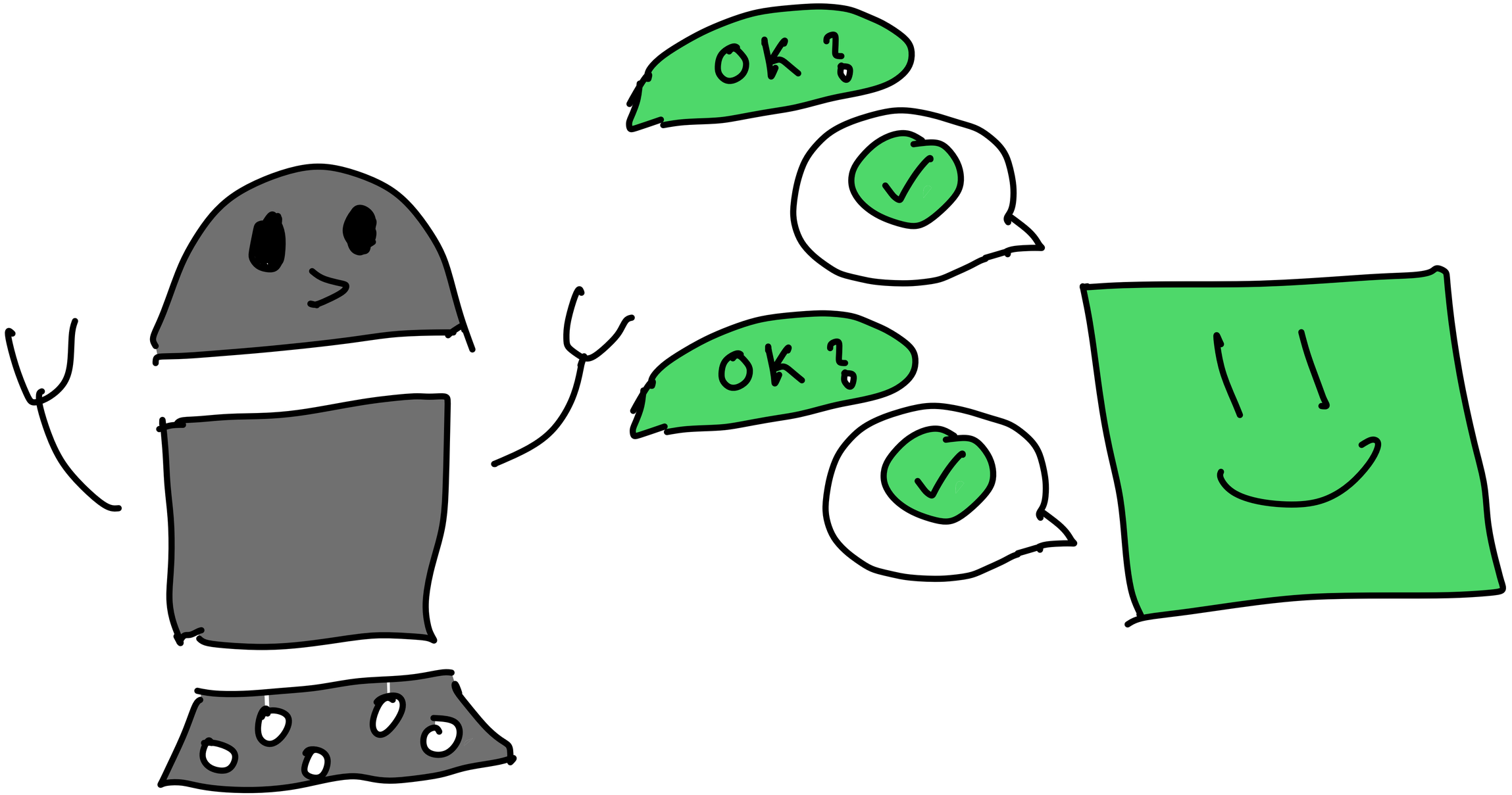
- ELK stack
(Elastic Search, Logstash, Kibana)
  The "E" in ELK is an
  Sponsor in this conference,
  Go talk to elastic!
- Grafana
- Splunk
- Prometheus

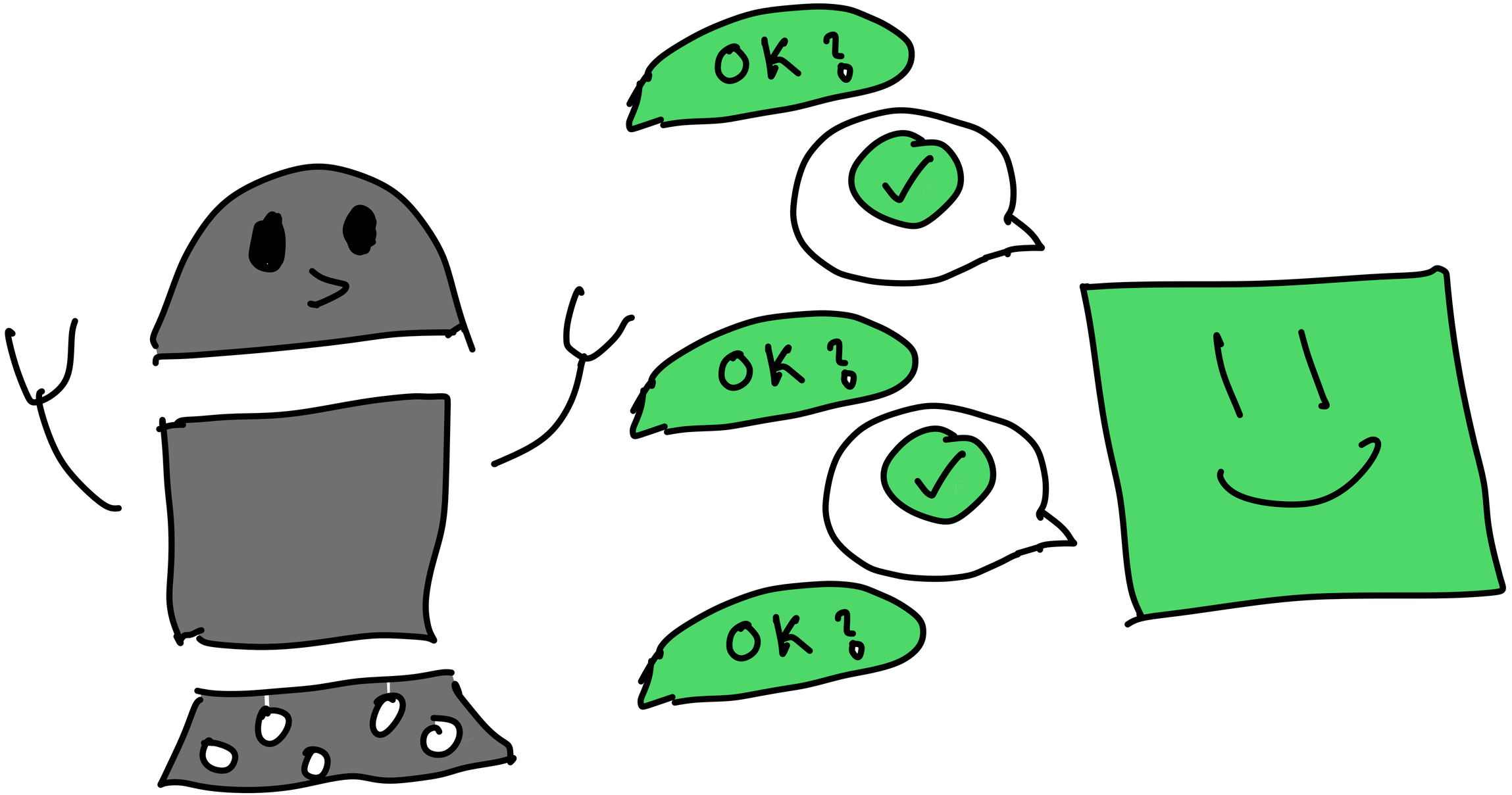
Health Checks / Automation

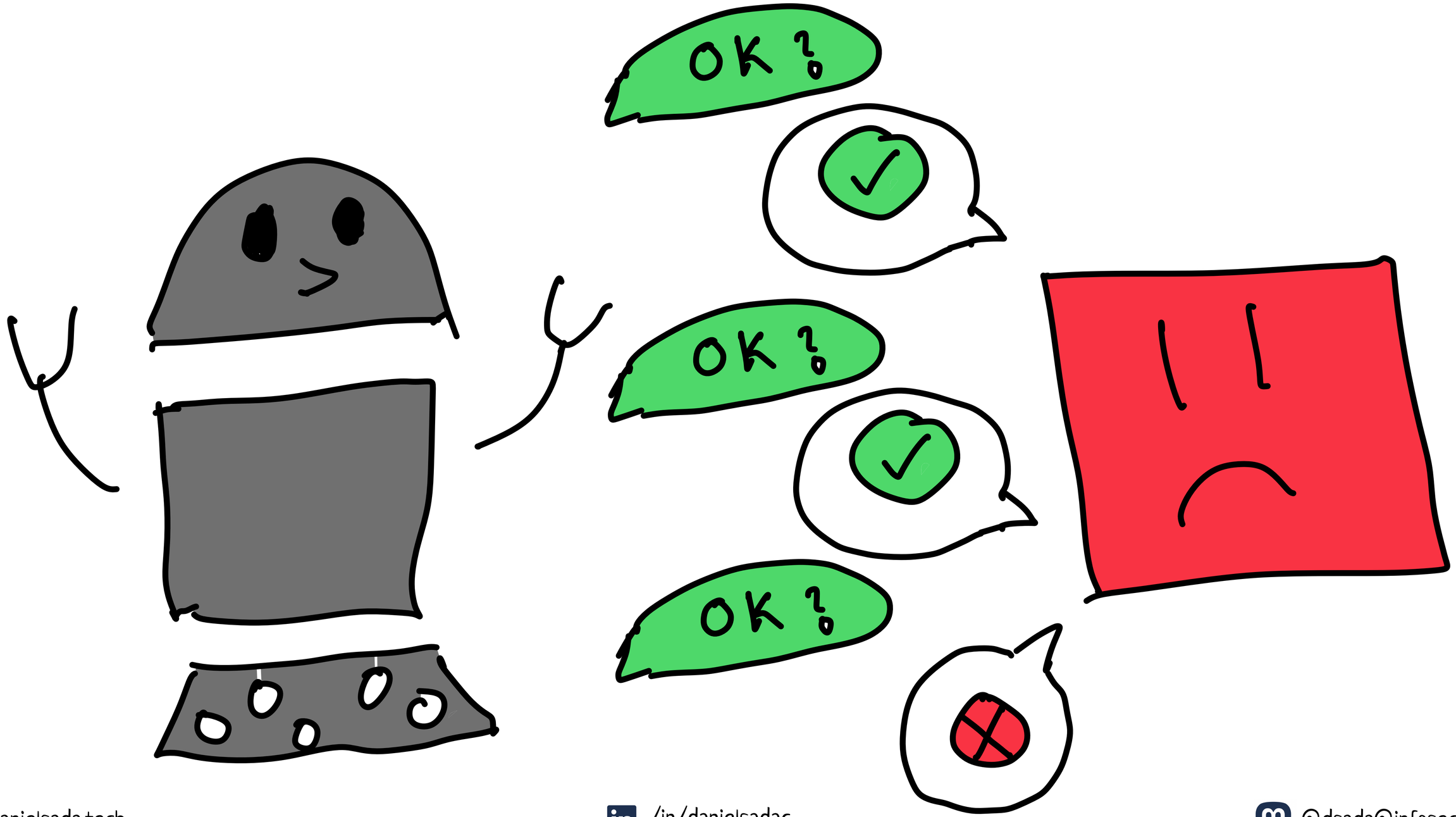

OK?

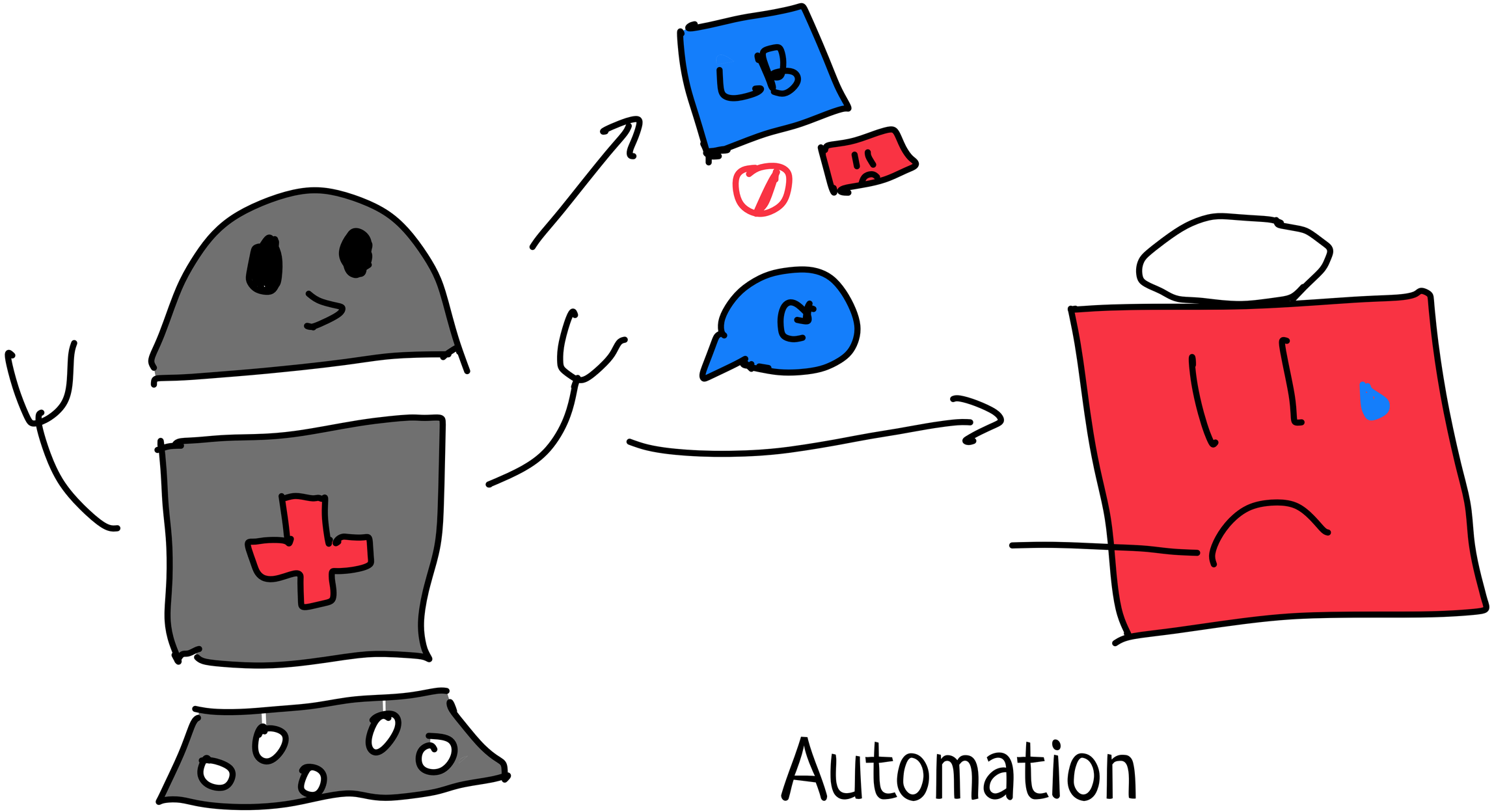

OK?


OK?
OK?

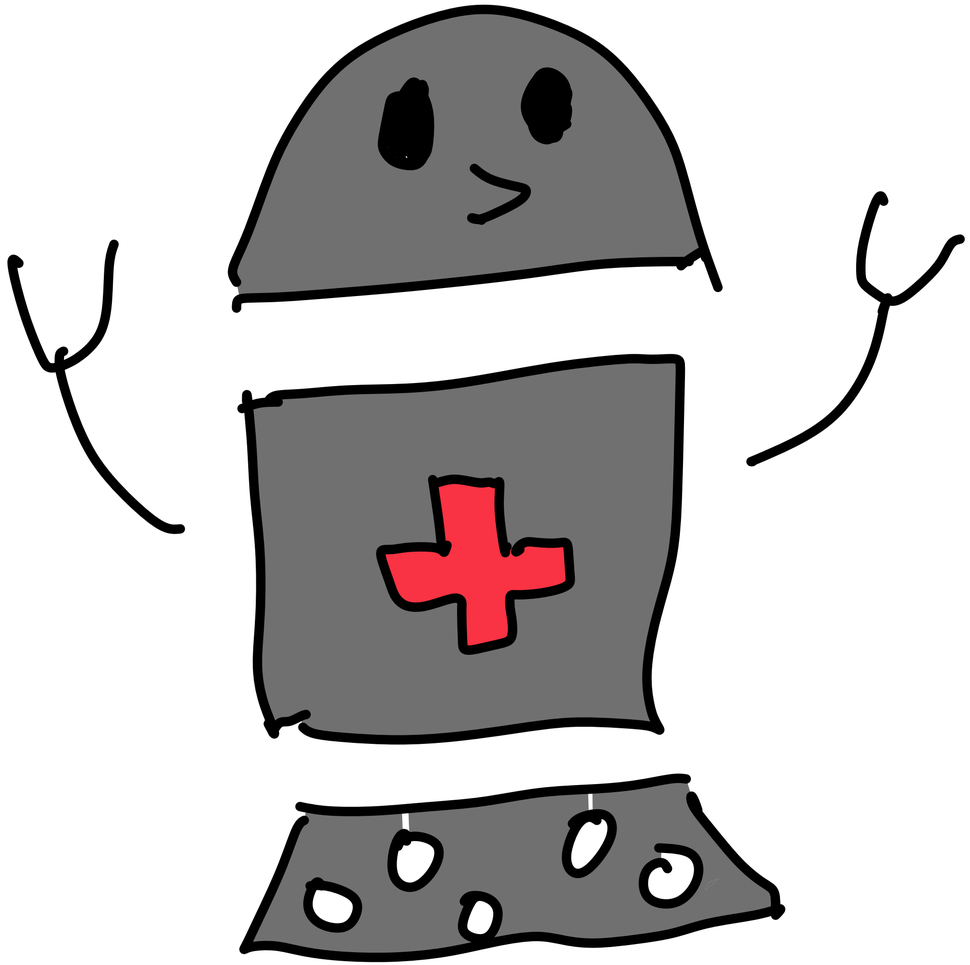

OK ?
OK ?

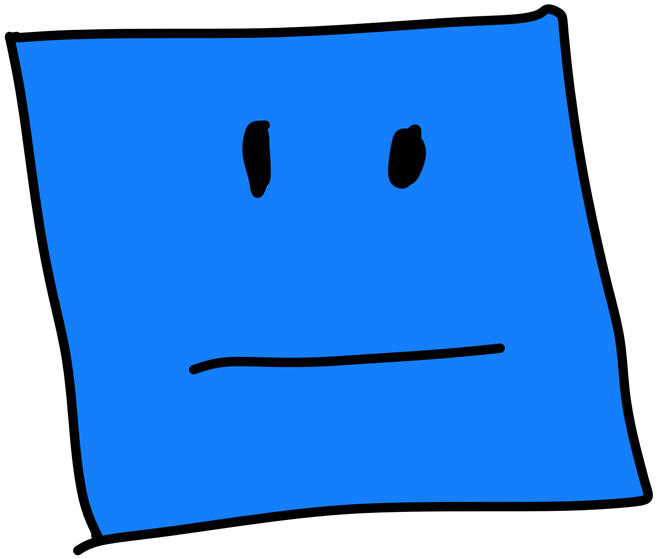

OK ?
OK ?
OK ?

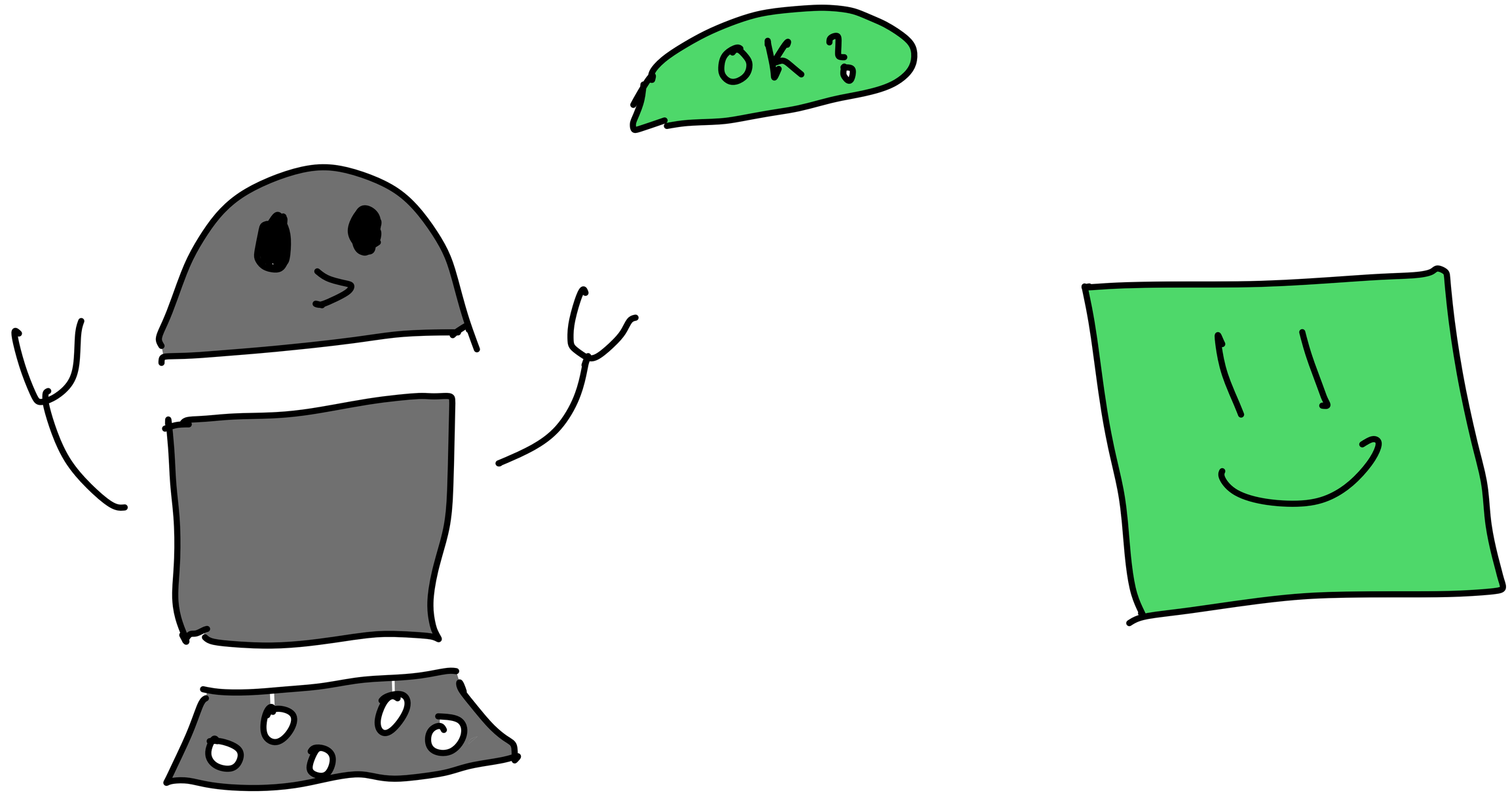

OK?
OK?
OK?

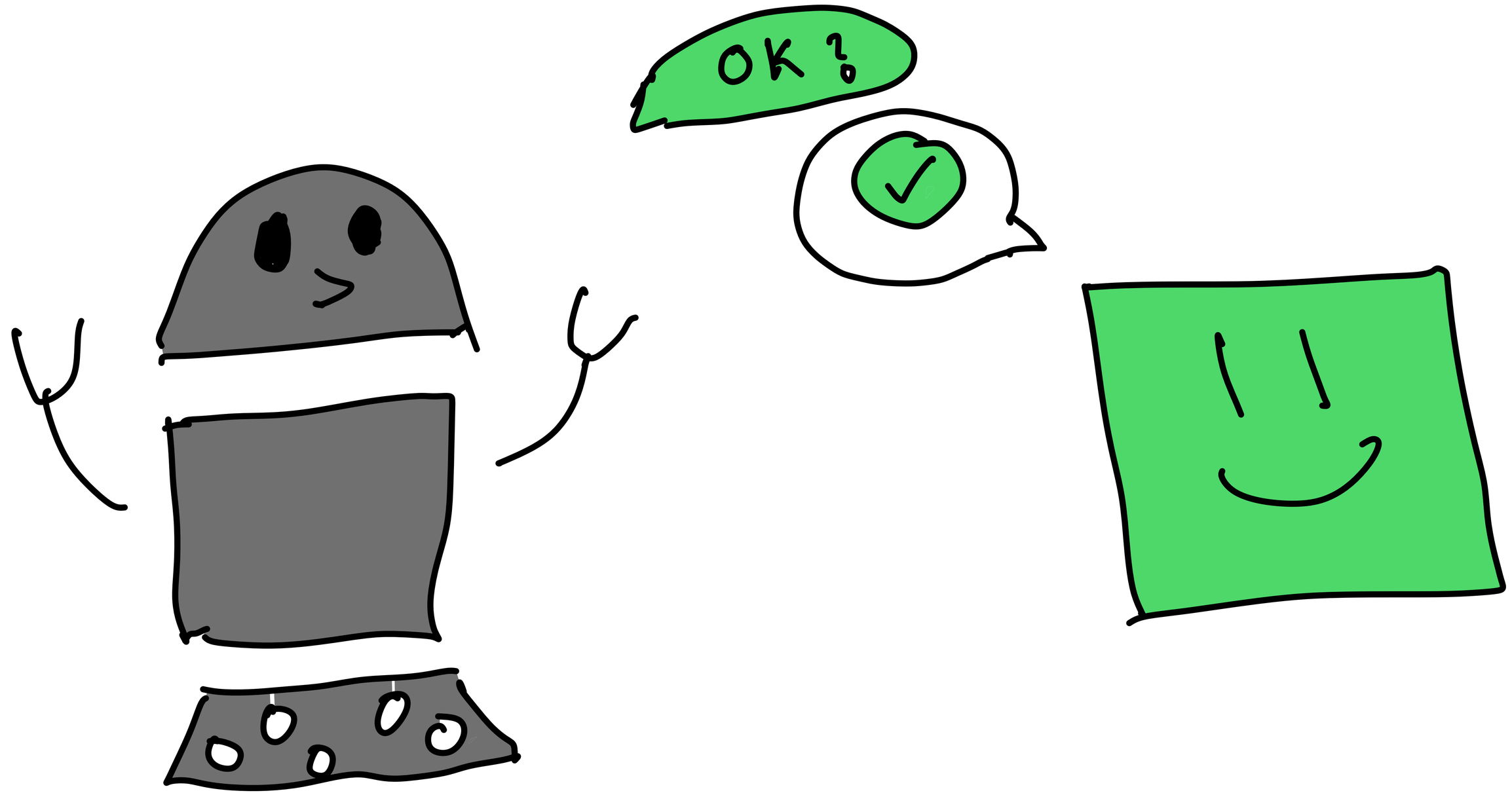


Automation





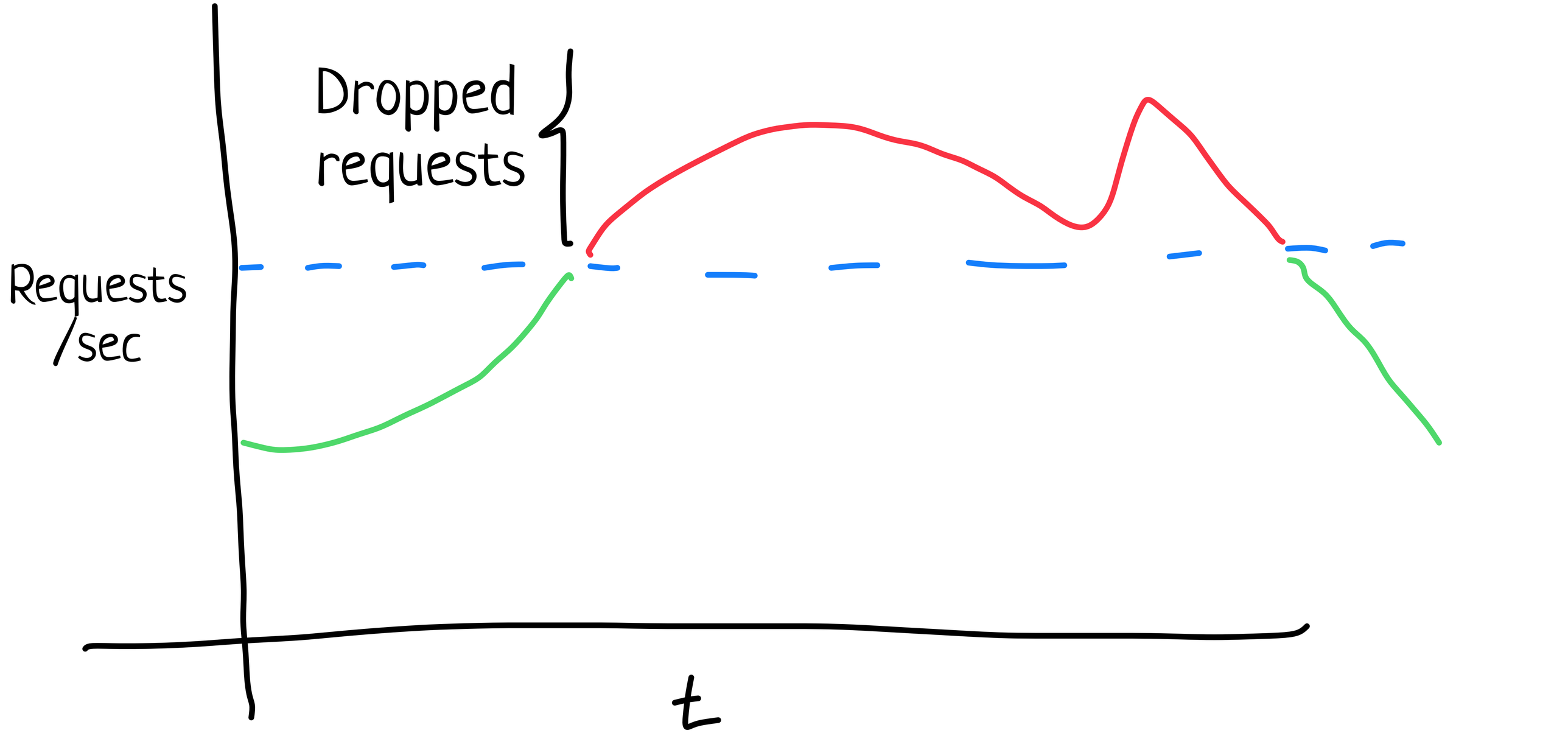

OK?

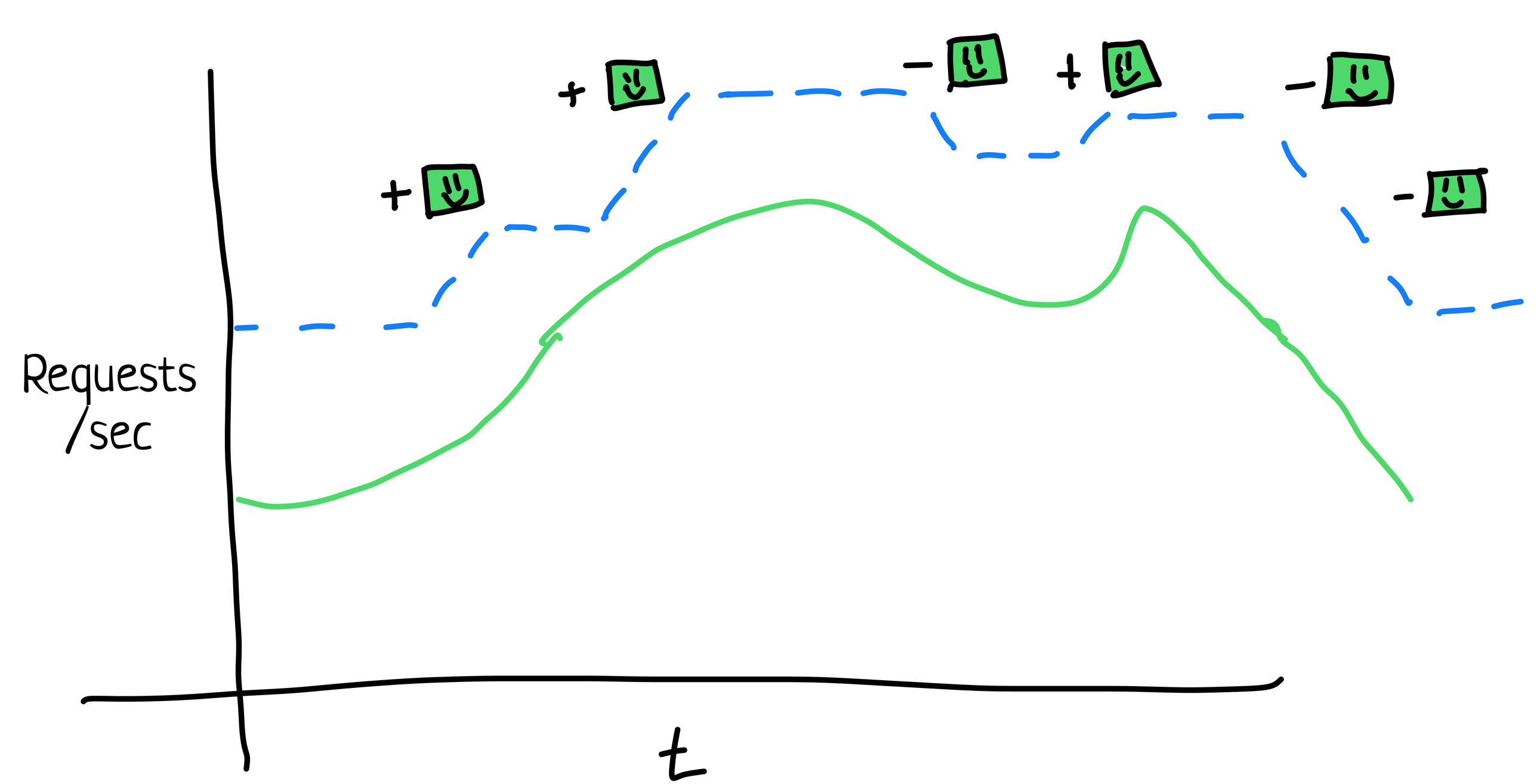

OK?

Health Checks / Automation

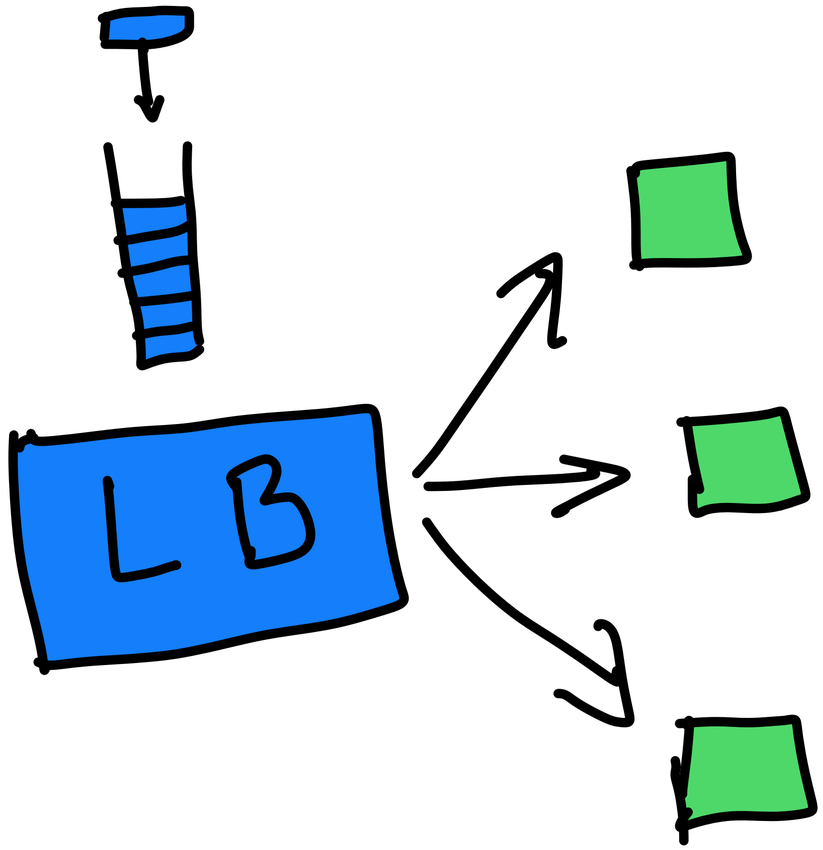
- Mostly using orchestrators
- On other environments, you can use your load balancer or expose endpoints.
- Cloud specific offerings.

Autoscaling

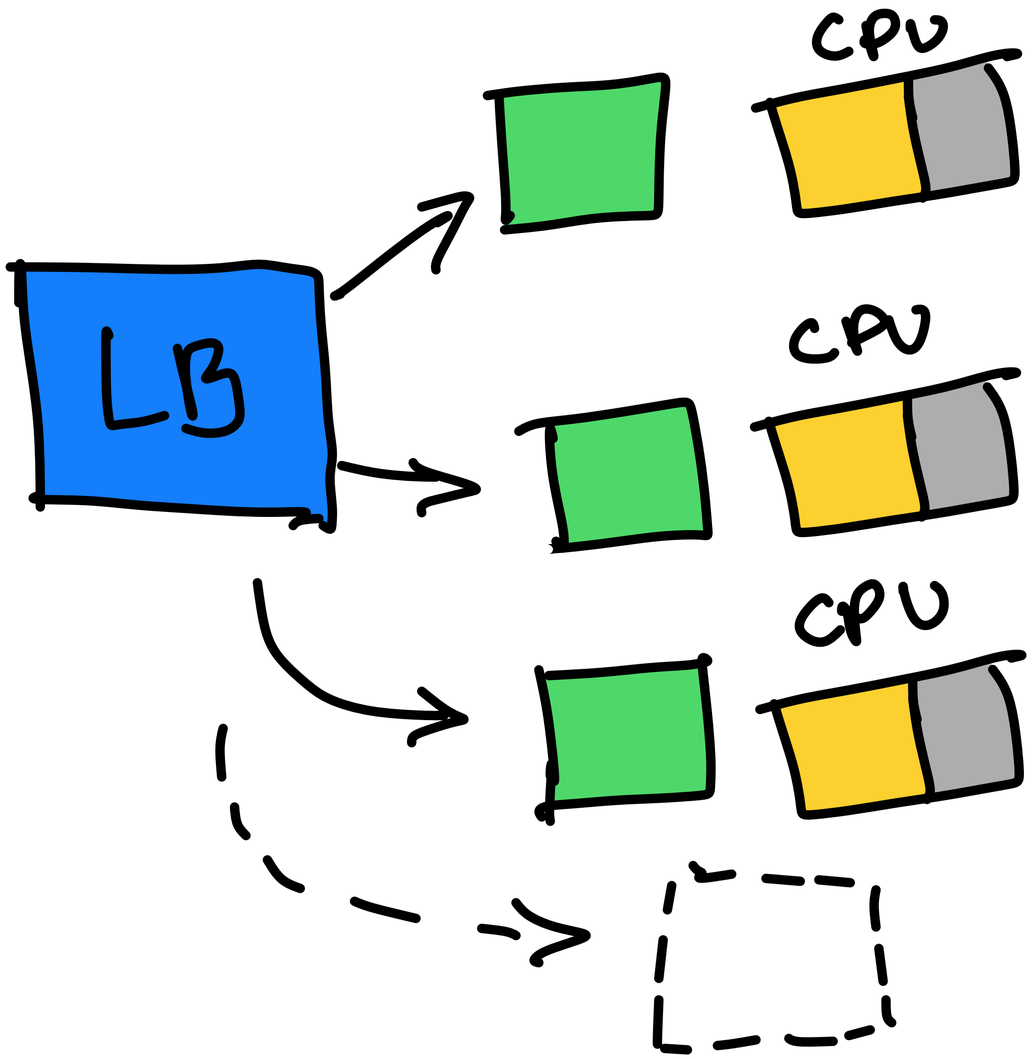

Dropped
requests
Requests
/sec
t


+
+
−
+
−
−
Requests
/sec
t

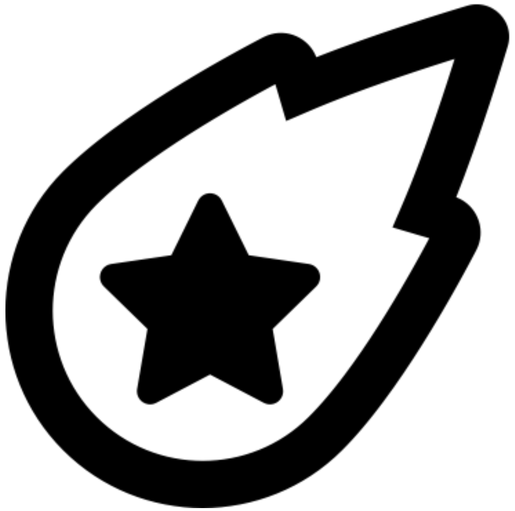
Autoscaling based On queue length



Autoscaling based On resource usage

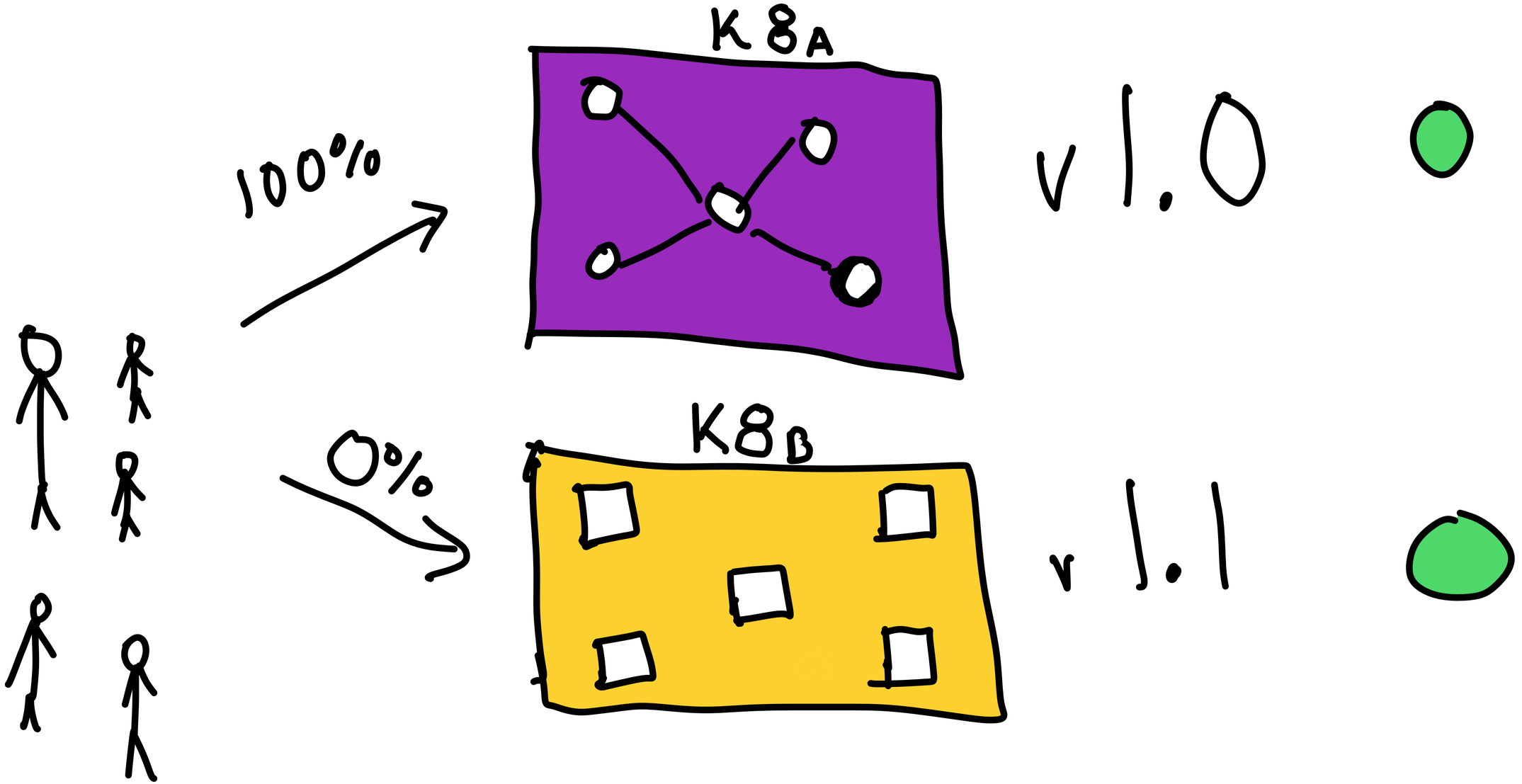


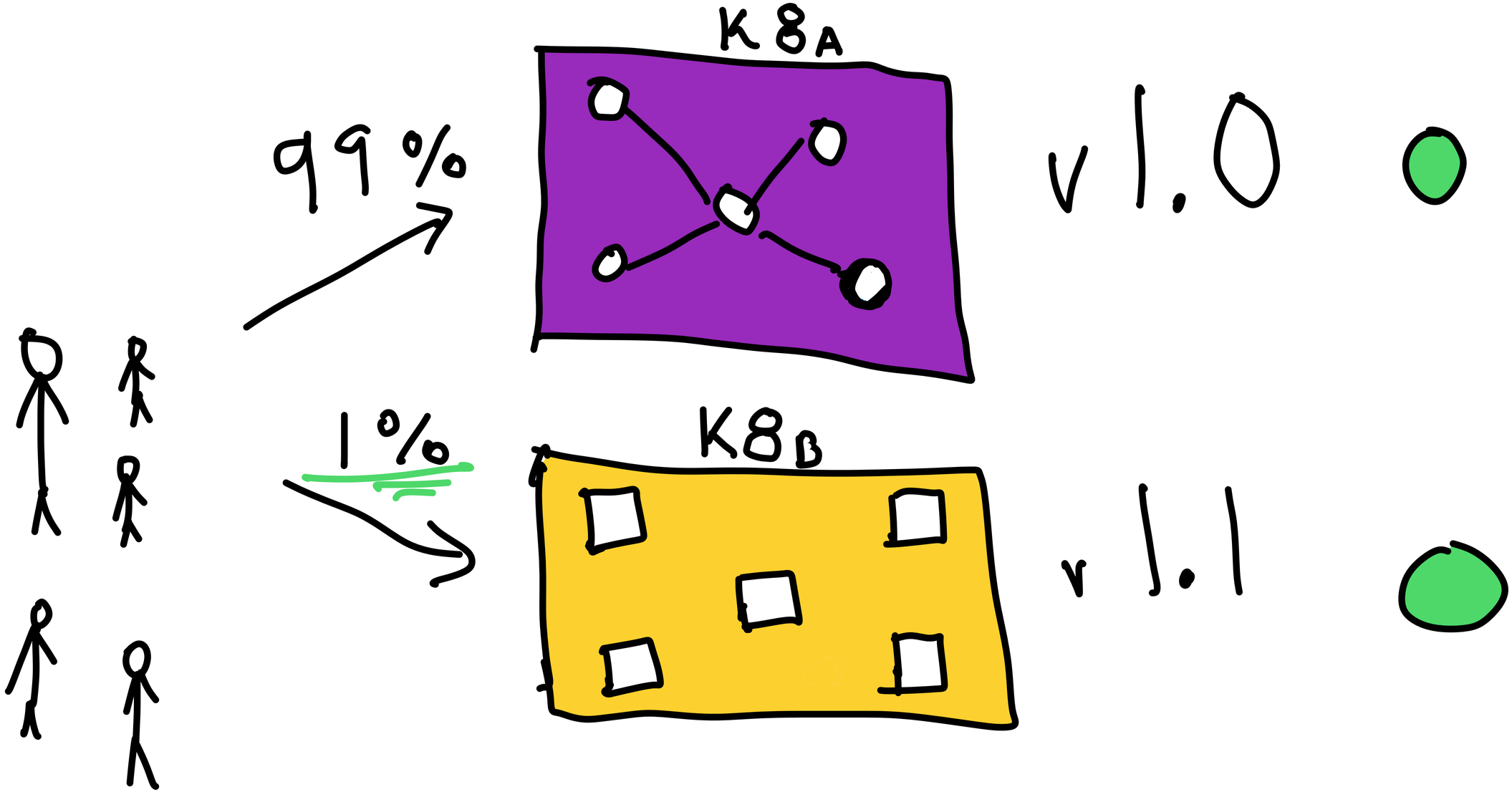
Hard

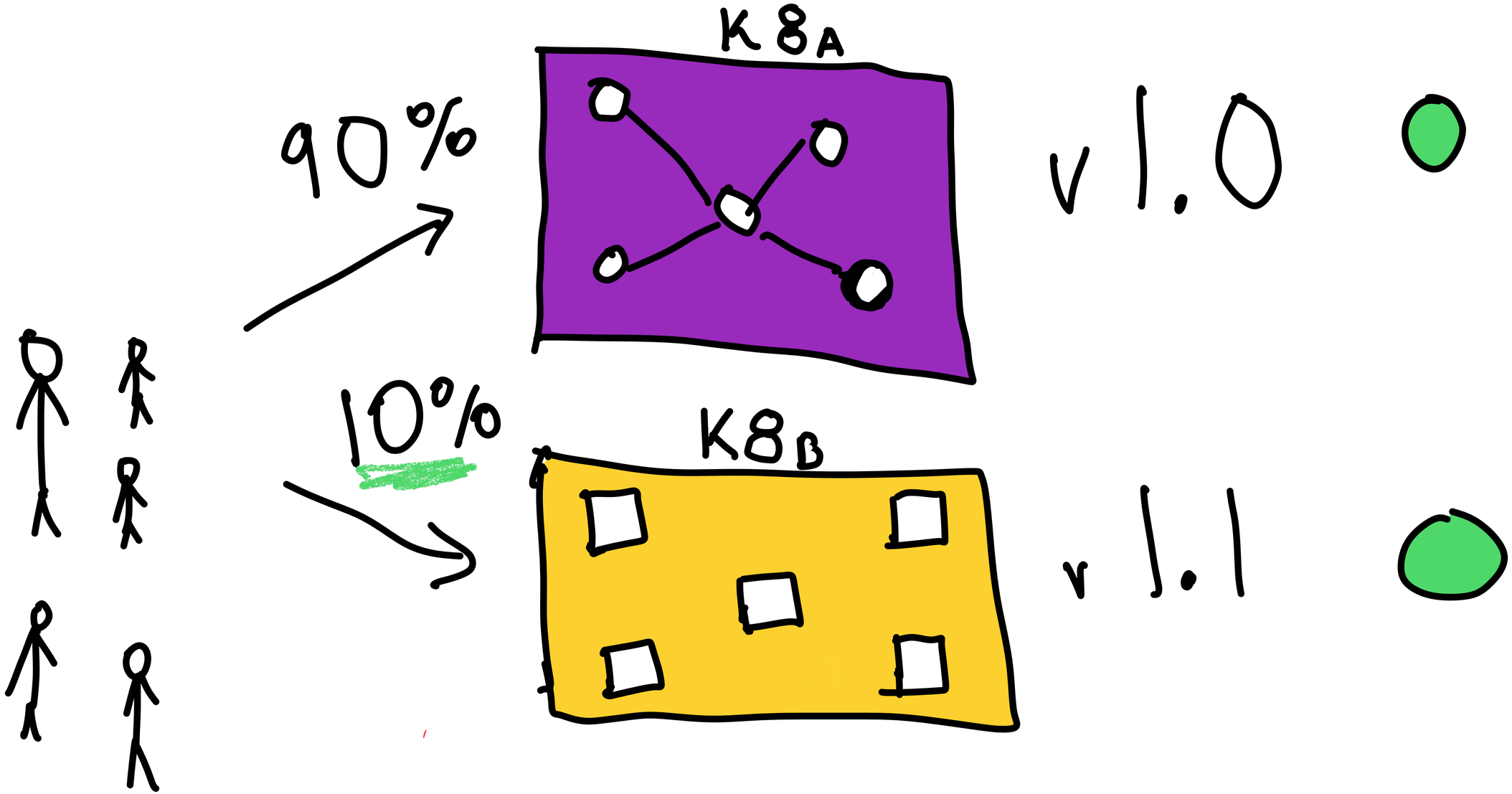


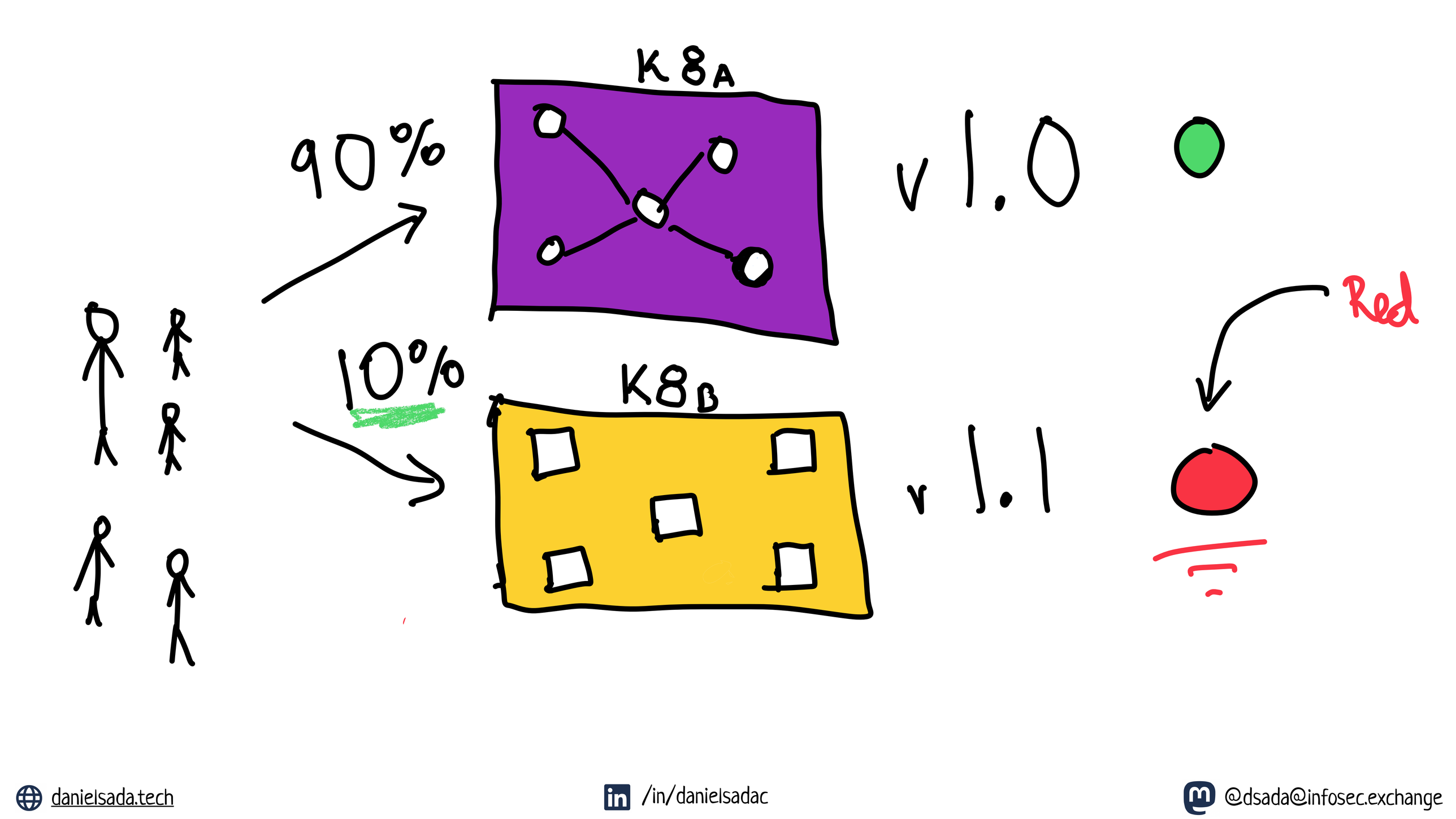
This is expensive and requires fine-tuning

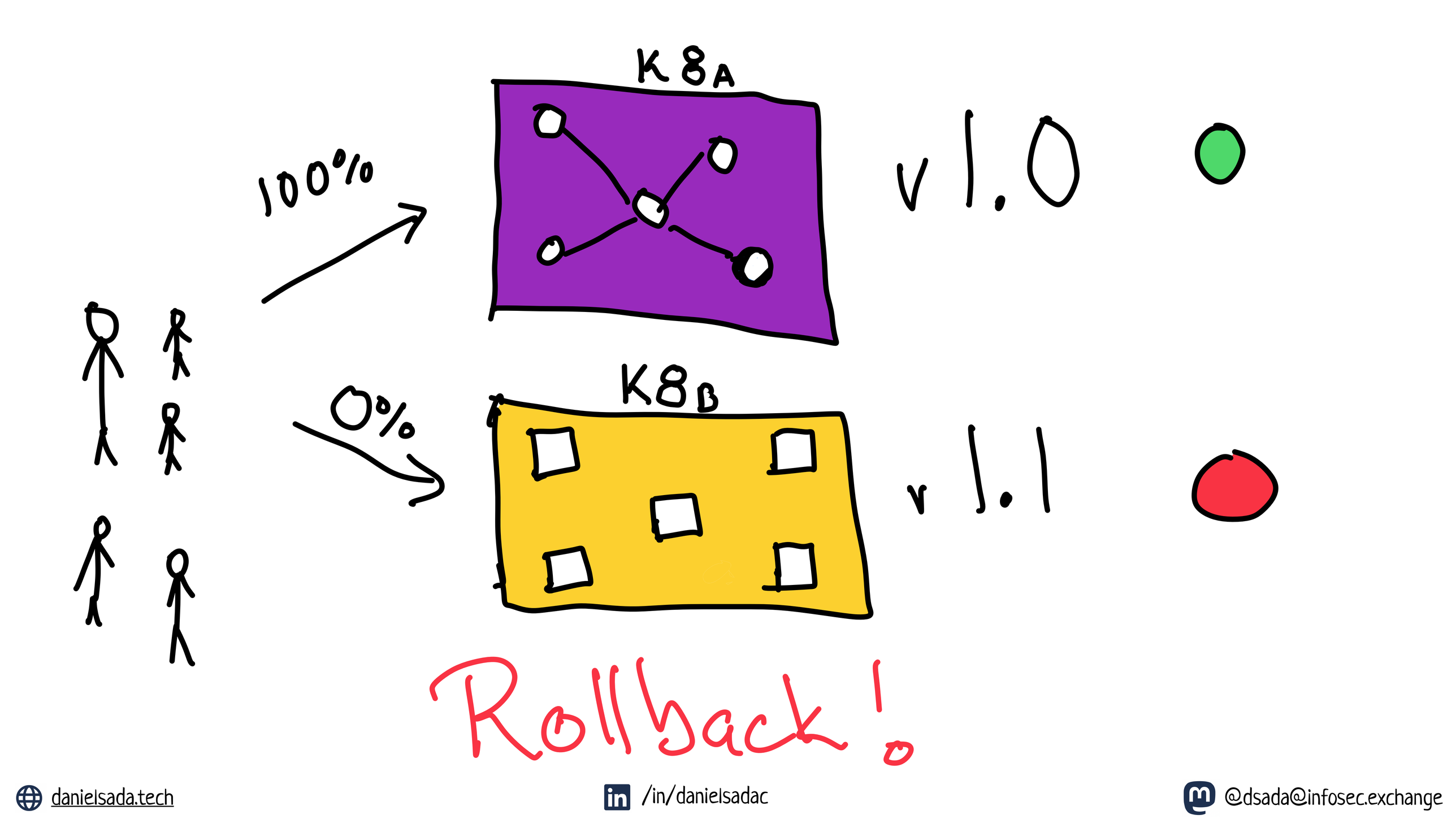
Green-Blue Deployments

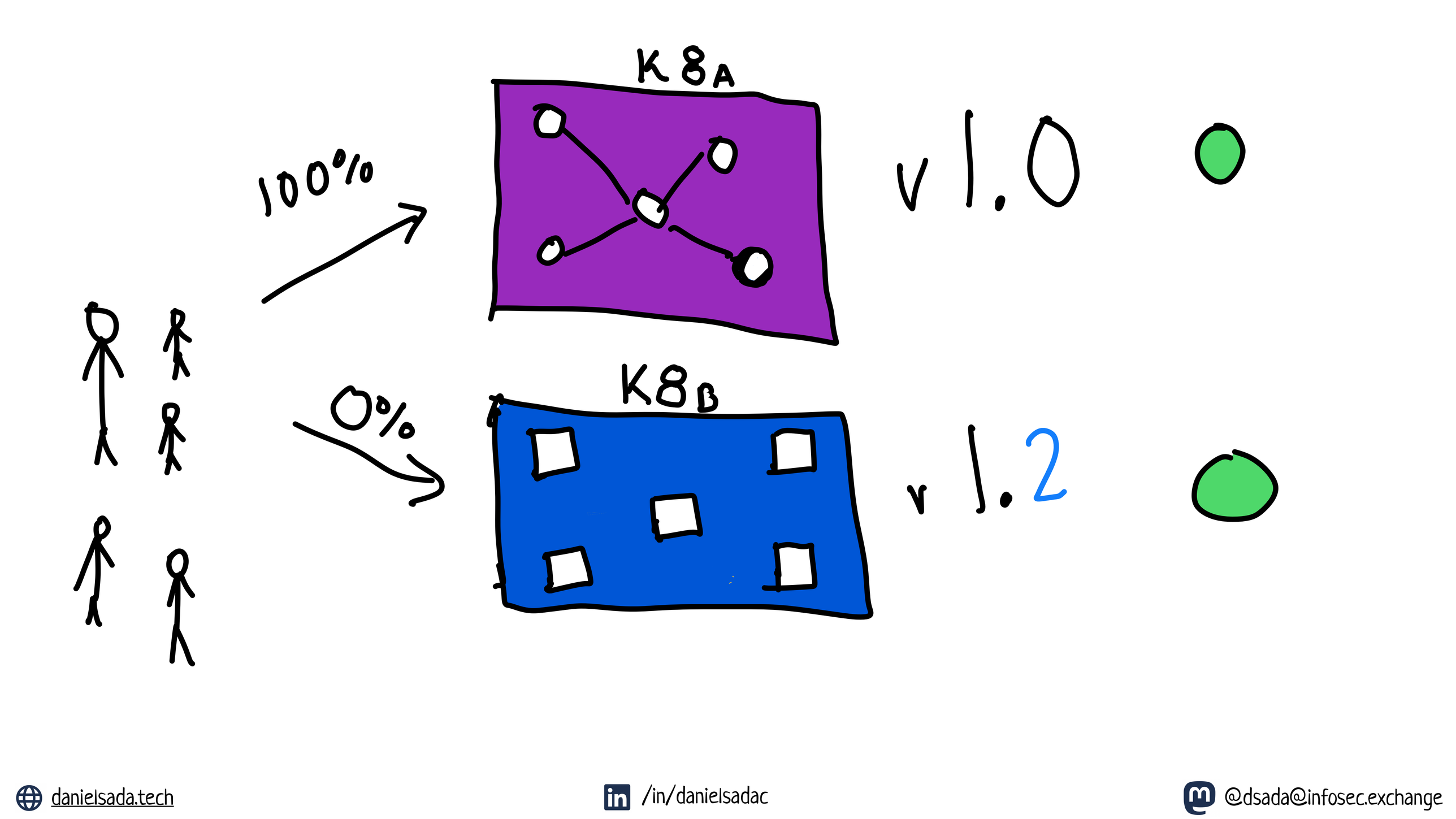

K8A
100%
v1.0
K8B
0%
v1.1

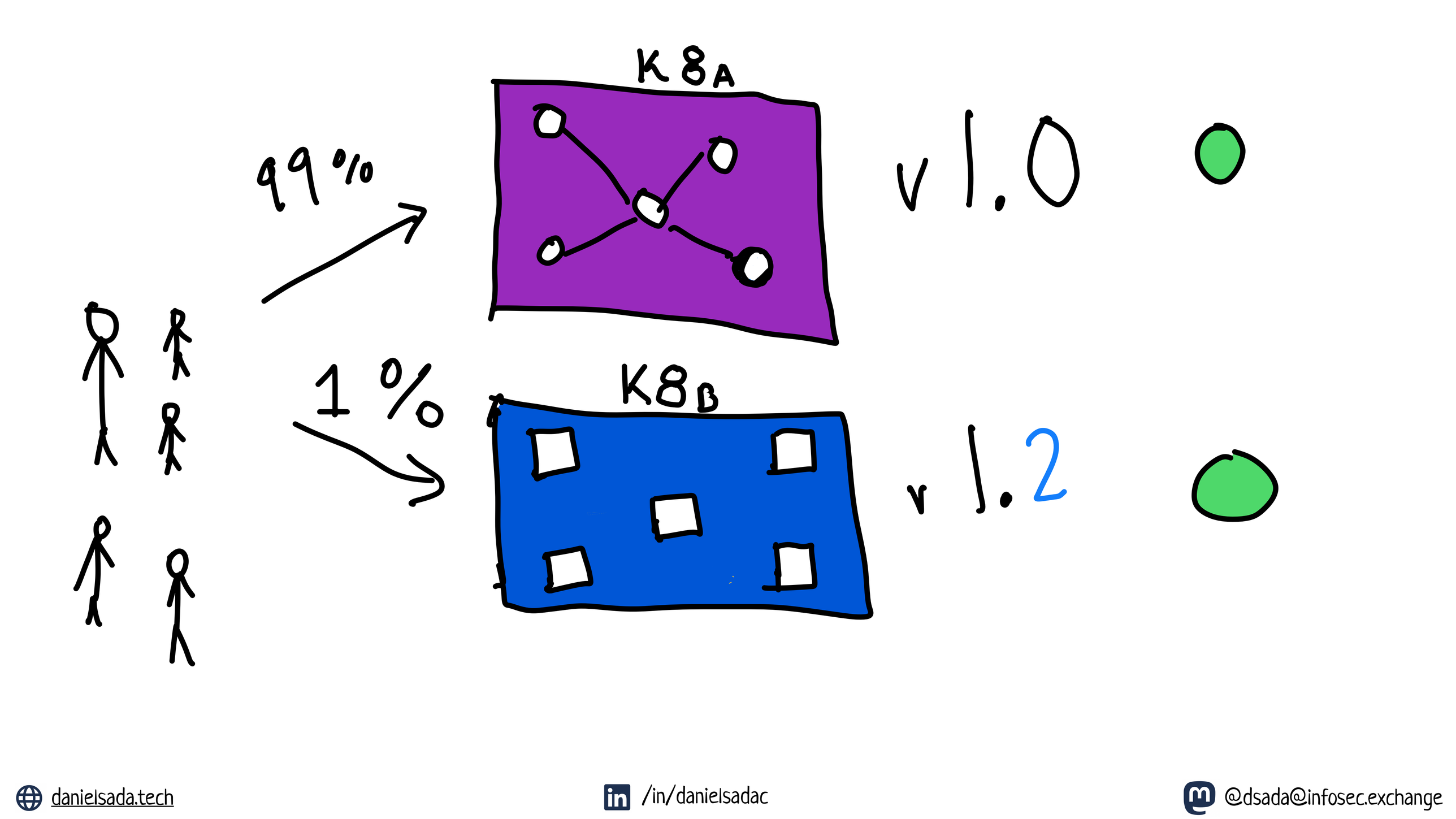

K8A
99%
v1.0
1%
K8B
v1.1

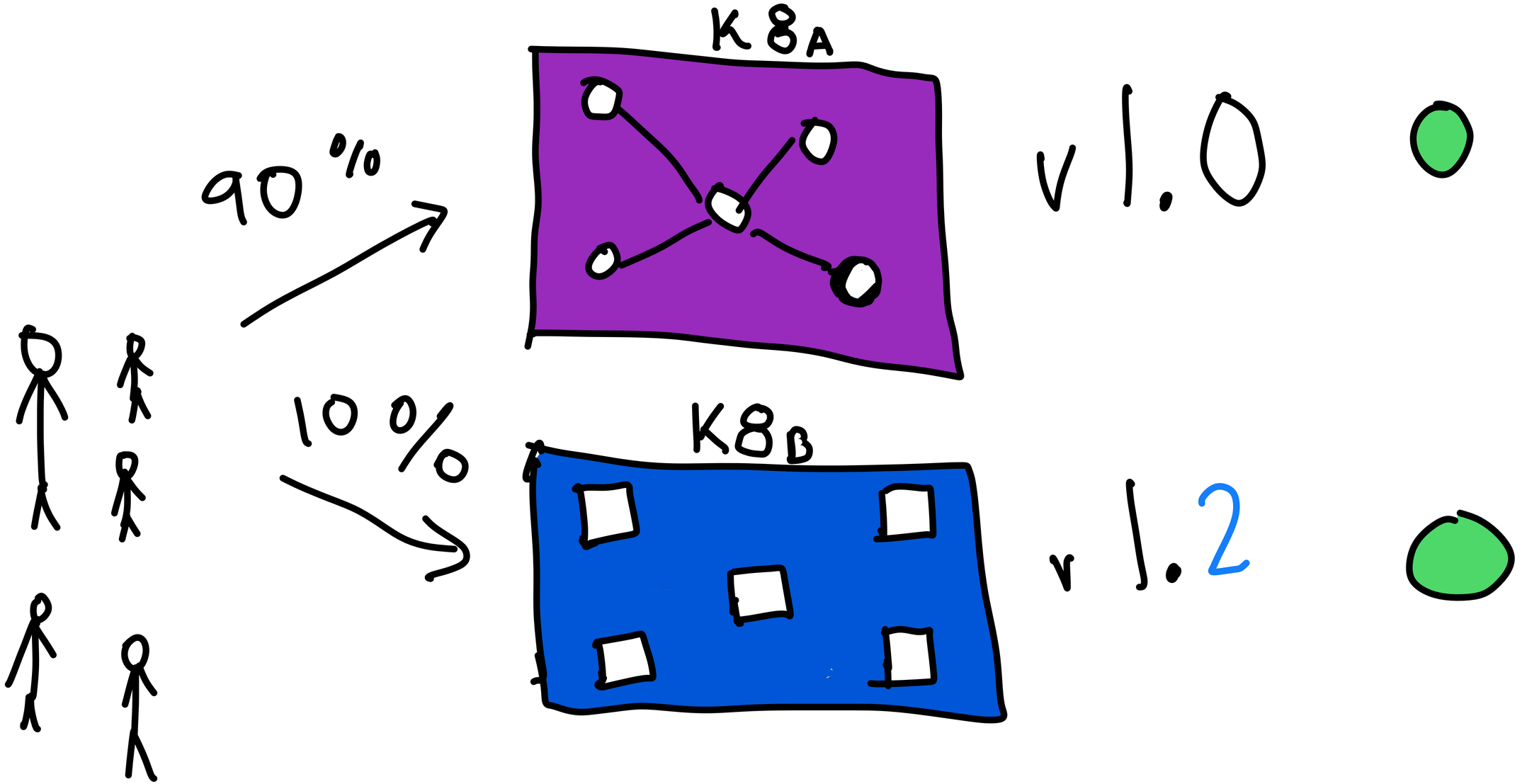

K8A
90%
v1.0
10%
K8B
v1.1

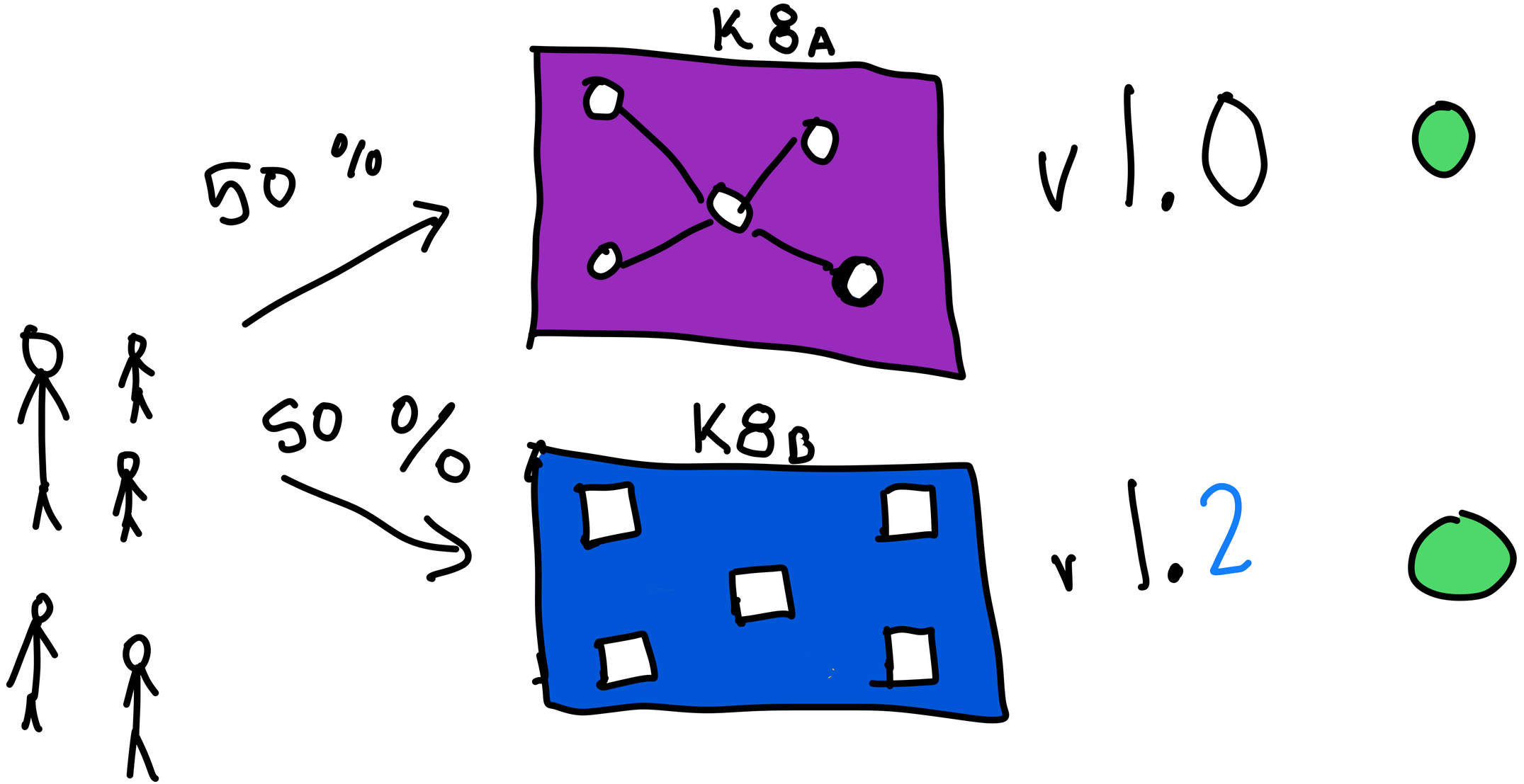

K8A
90%
v1.0
10%
K8B
v1.1
Red

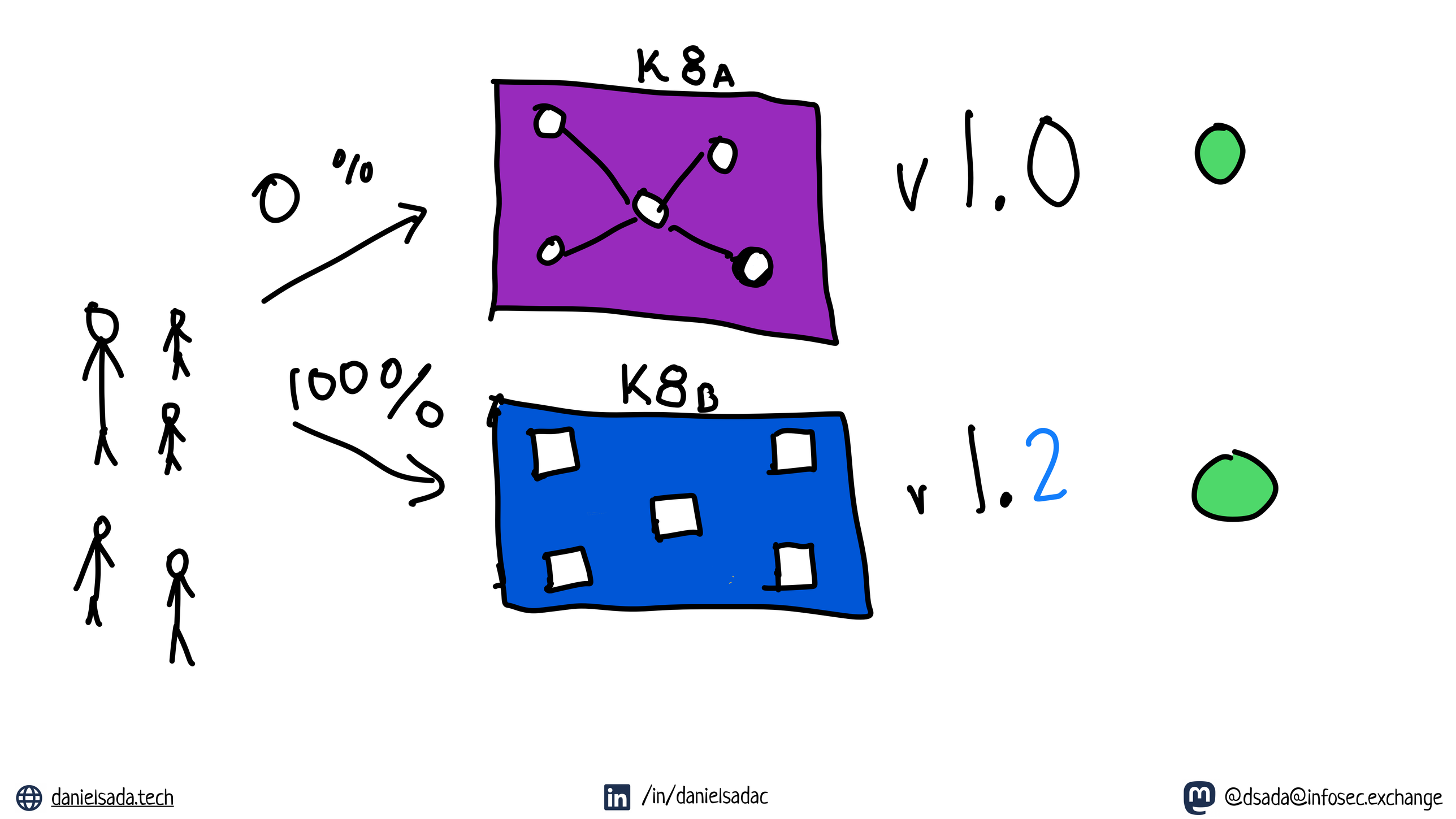

K8A
100%
v1.0
0%
K8B
v1.1
Rollback!


K8A
100%
v1.0
K8B
0%
v1.2

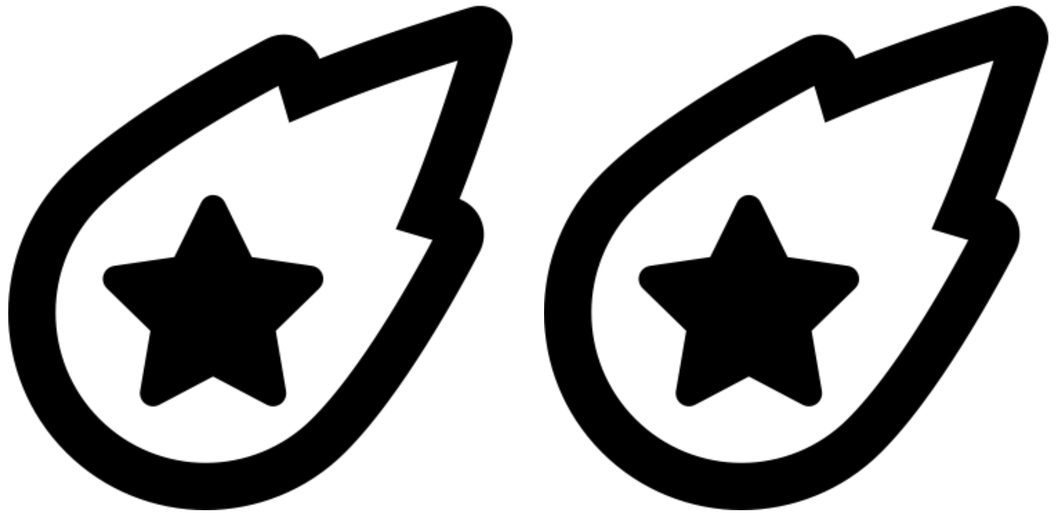

K8A
99%
v1.0
1%
K8B
v1.2


K8A
90 %
v 1.0
10 %
K8B
v 1.2


K8A
50 %
v1.0
50 %
K8B
v1.2


K8A
0 %
v1.0
100%
K8B
v1.2

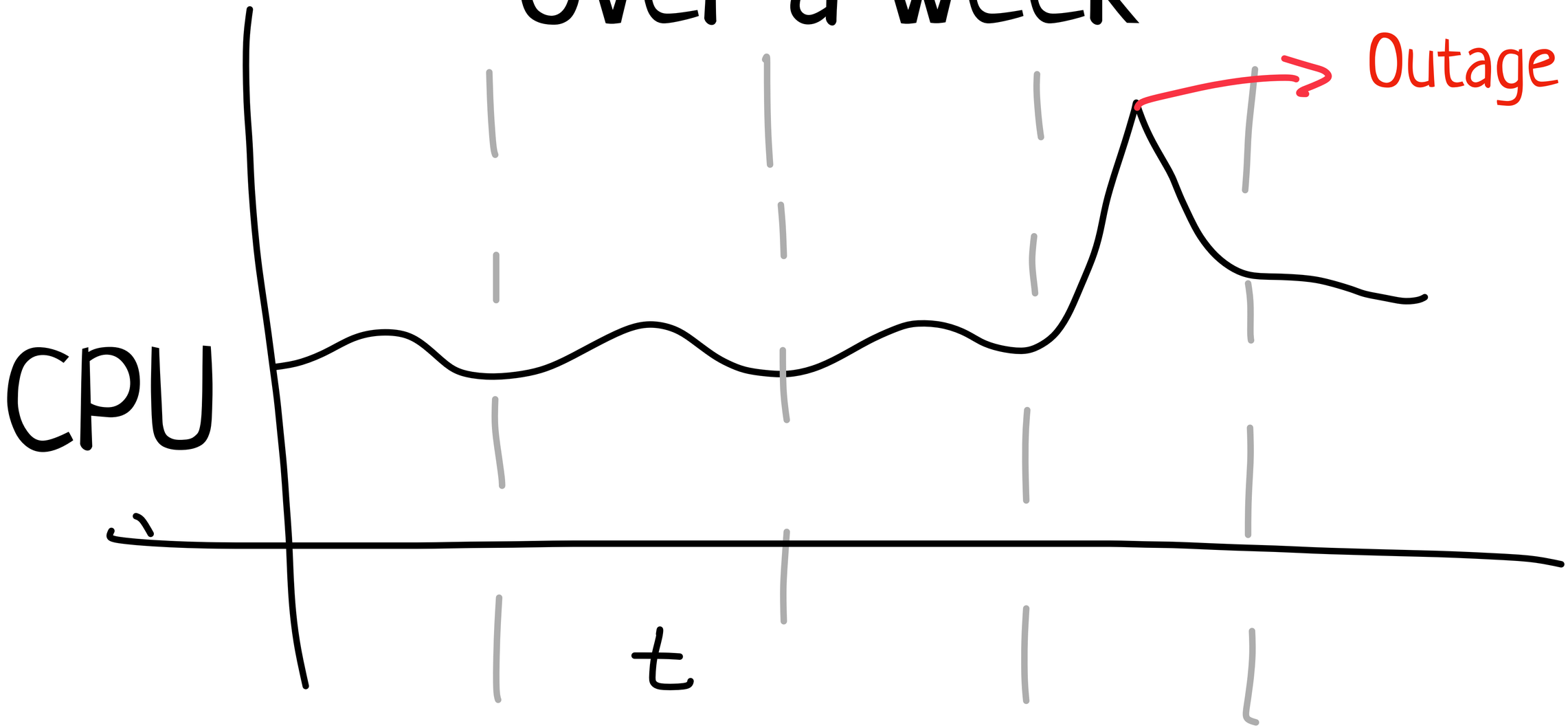
EXPERT

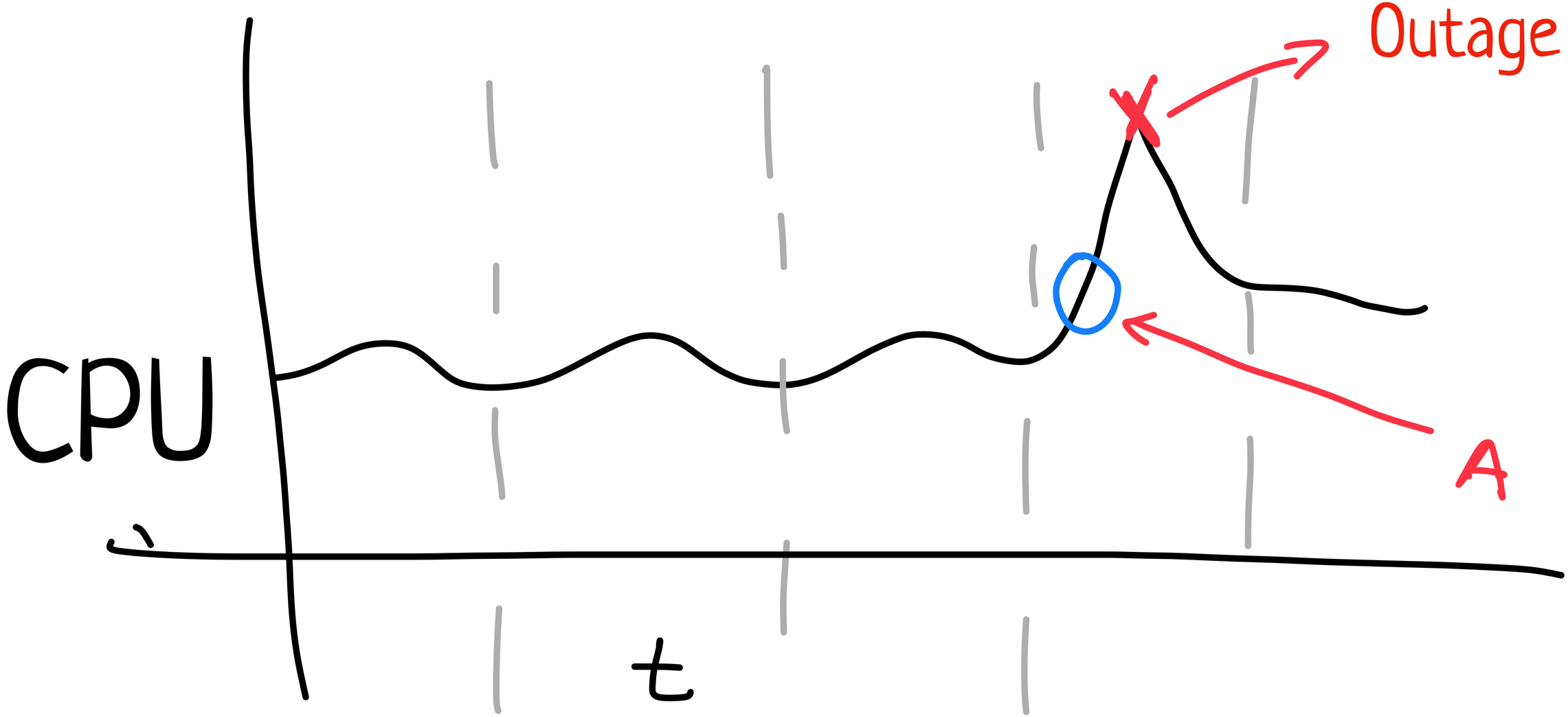


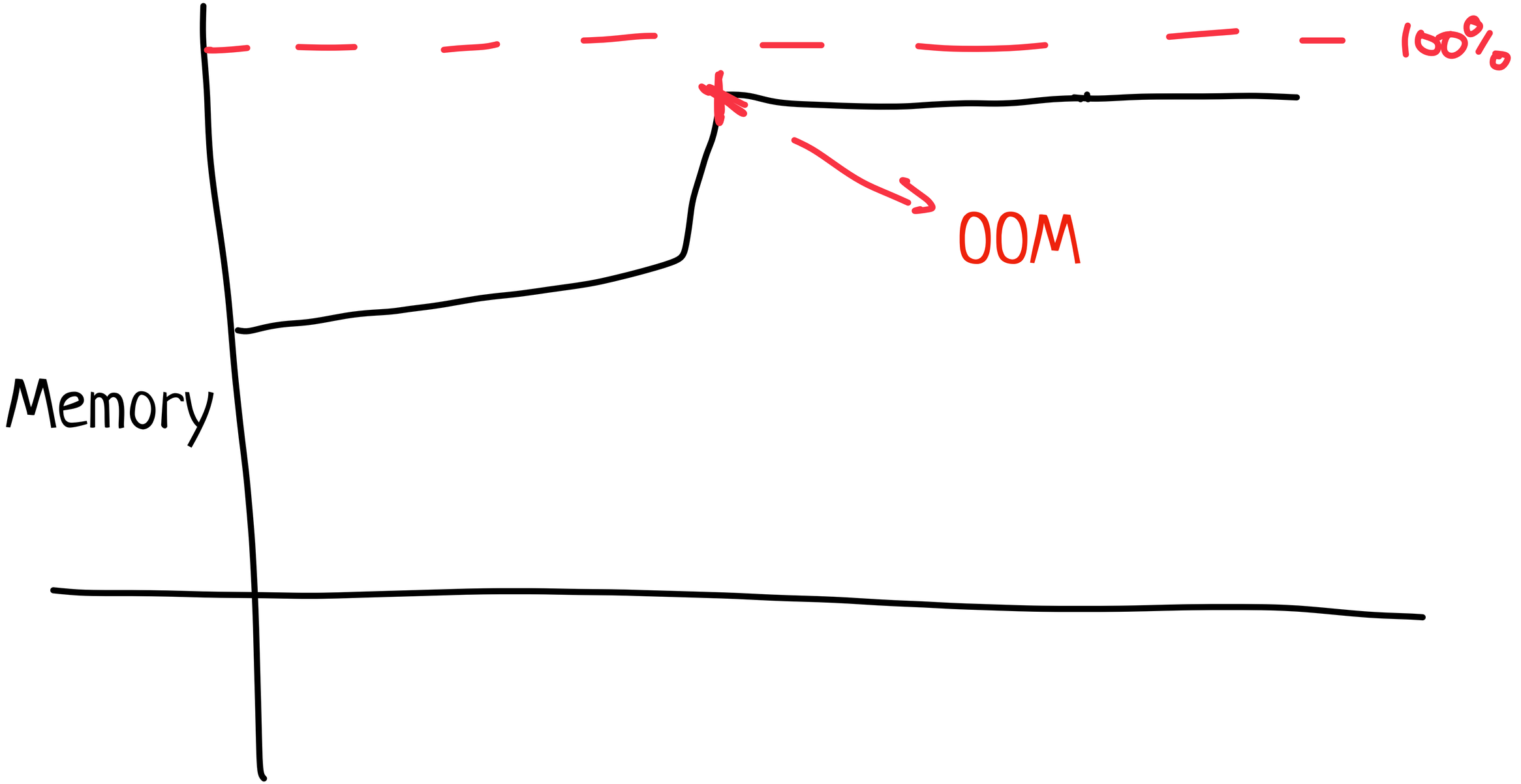
These are hard problems in the field.

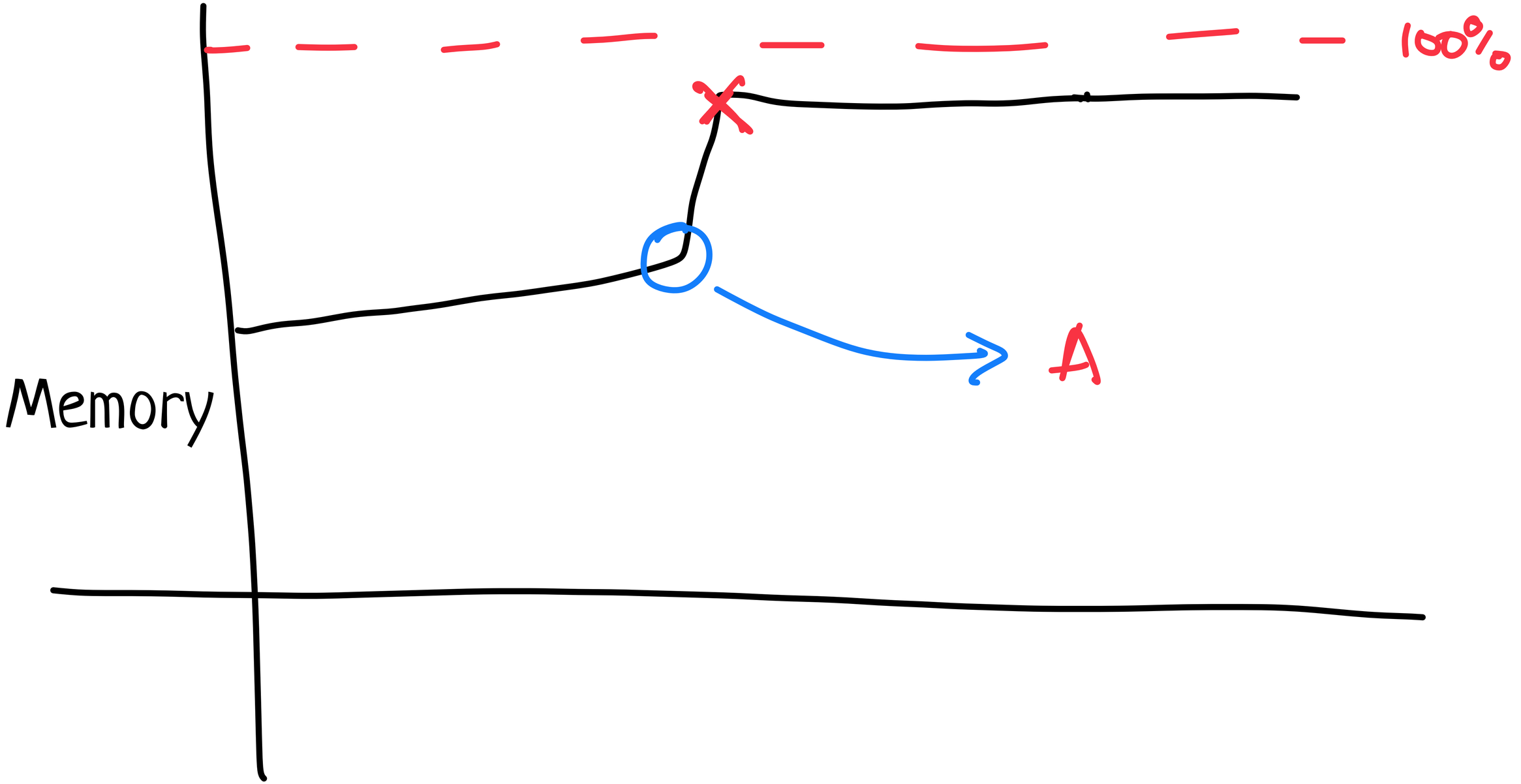
Anomaly detection
with automation

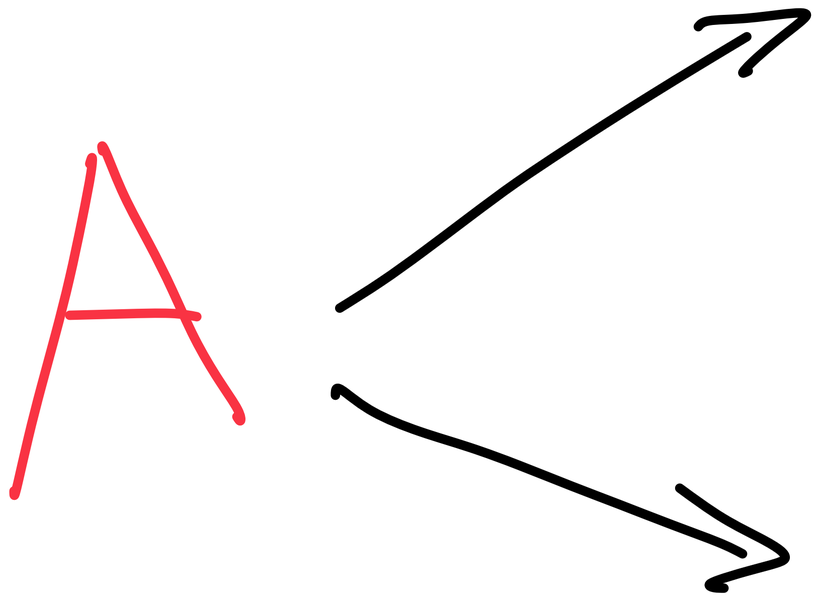
Over a week




Outage
CPU
A
t


100%
OOM
Memory


100%
Memory
A



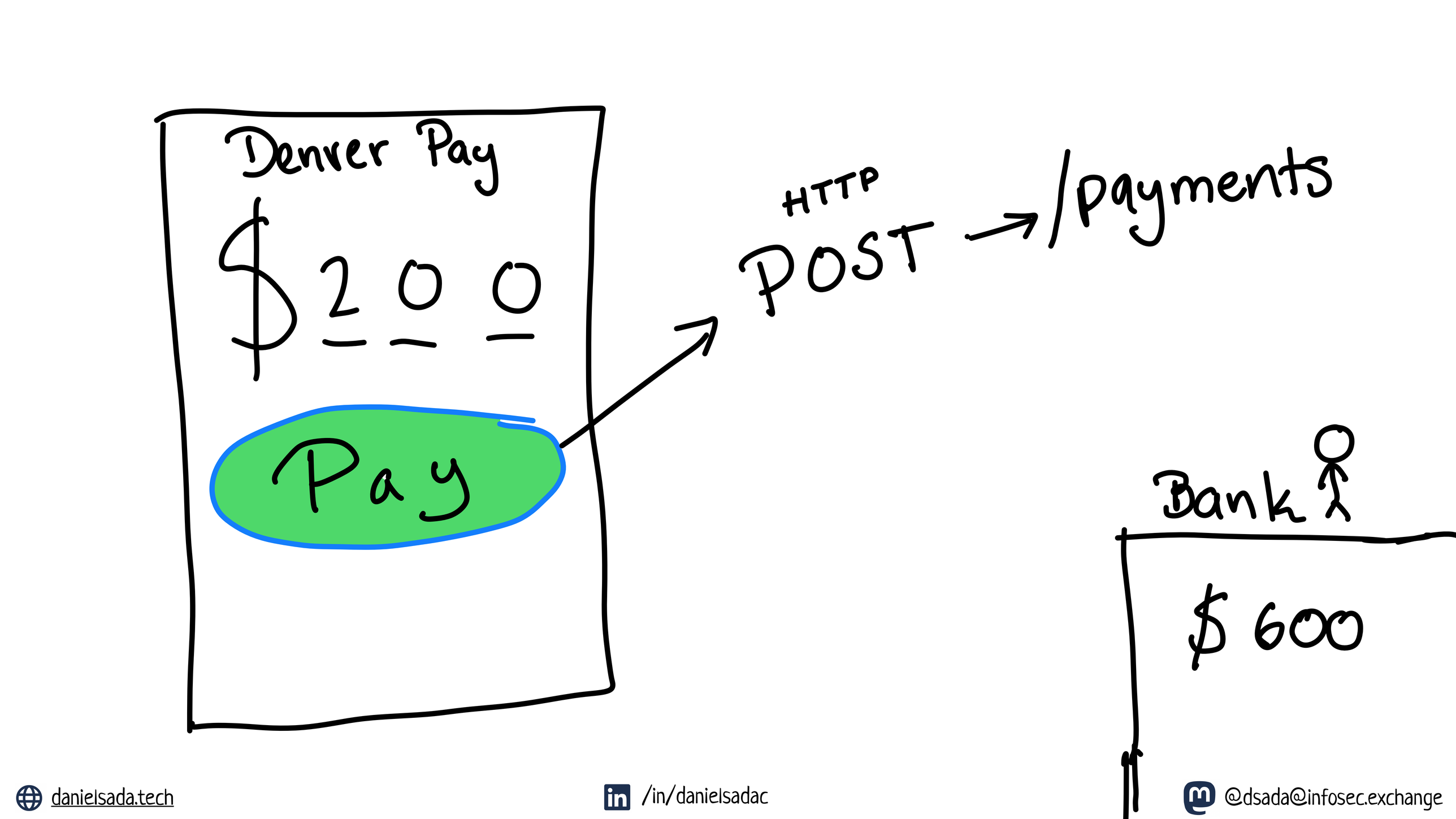
Corrective Action

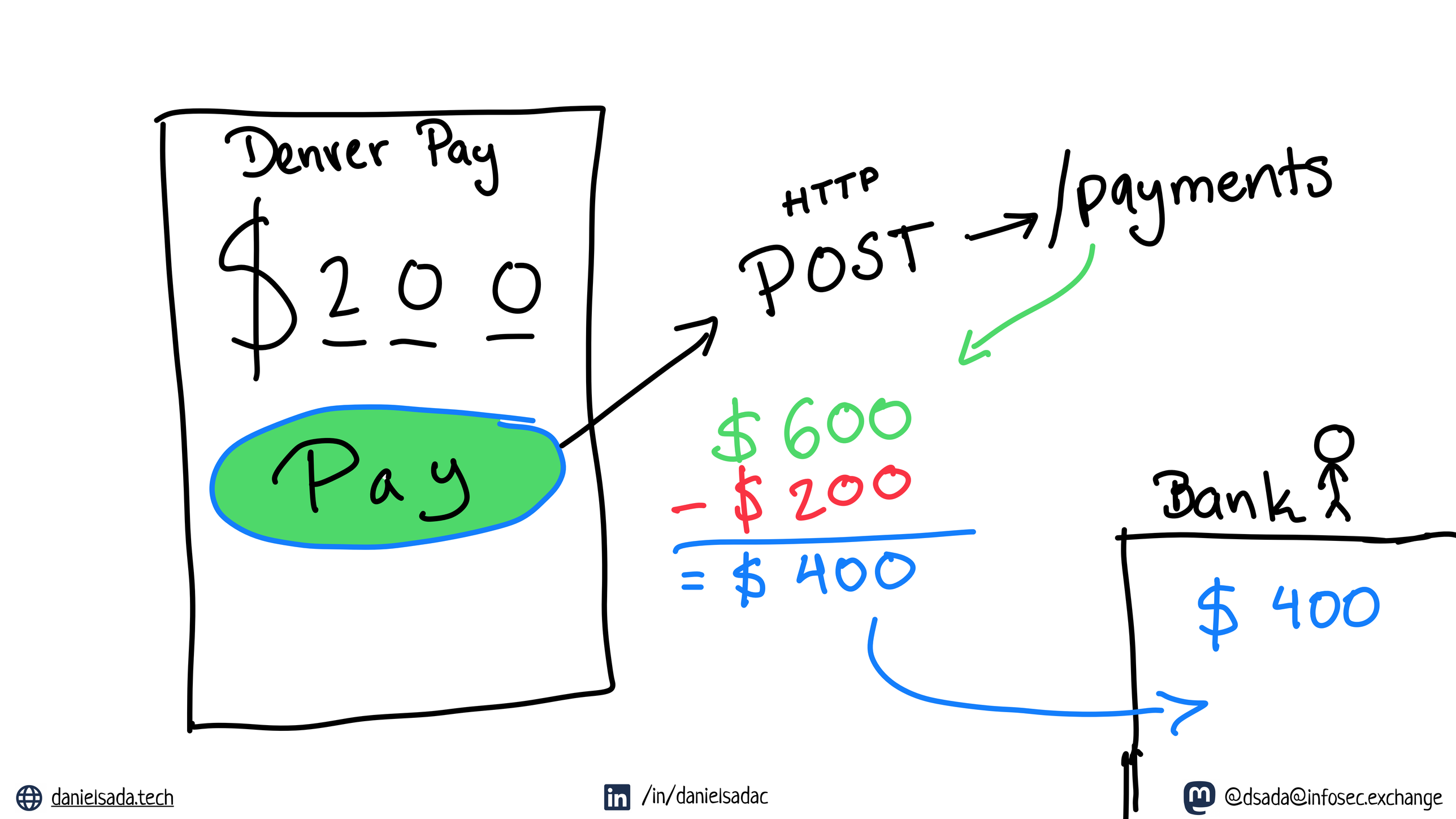
Alert*

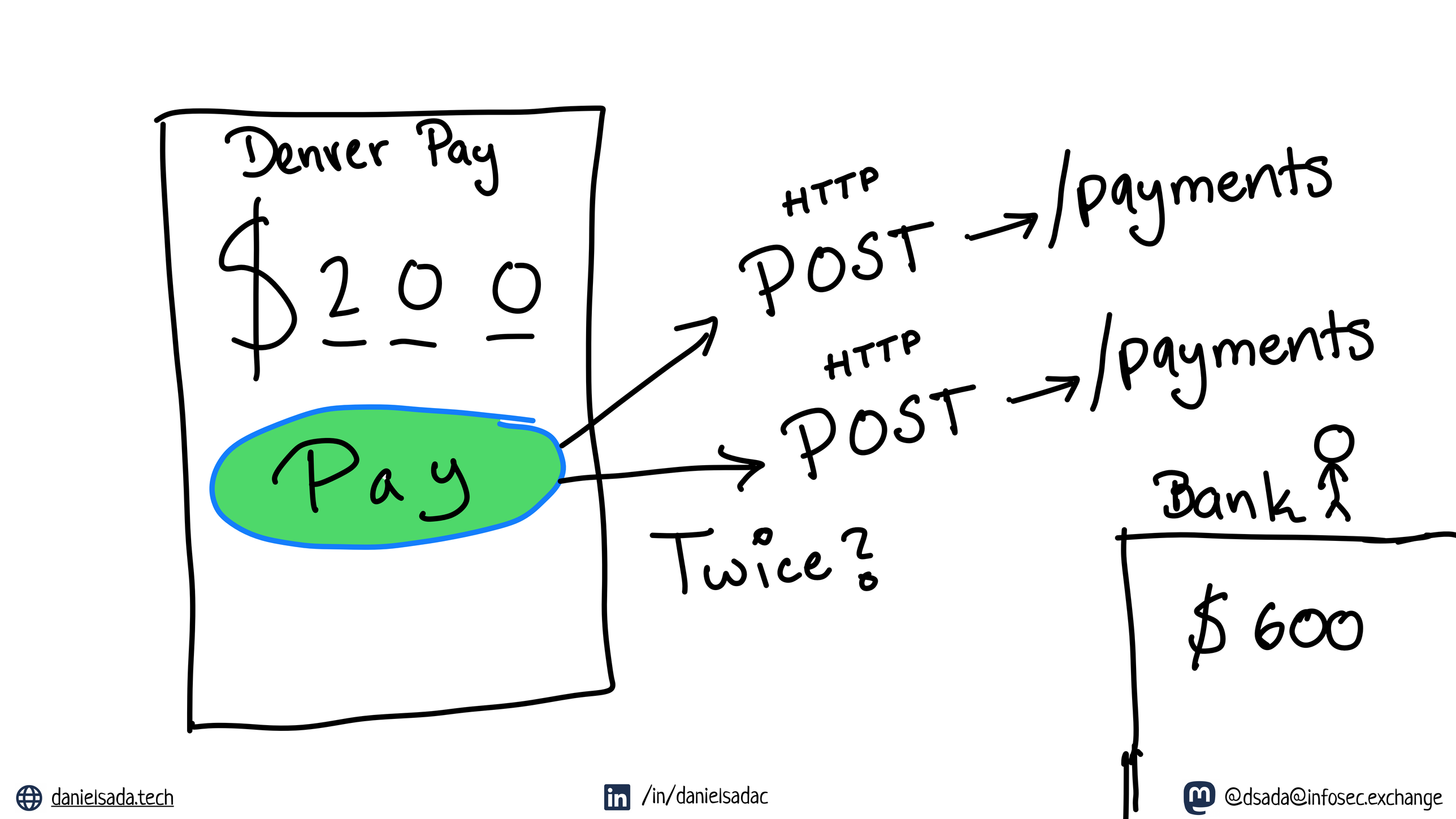
🔔

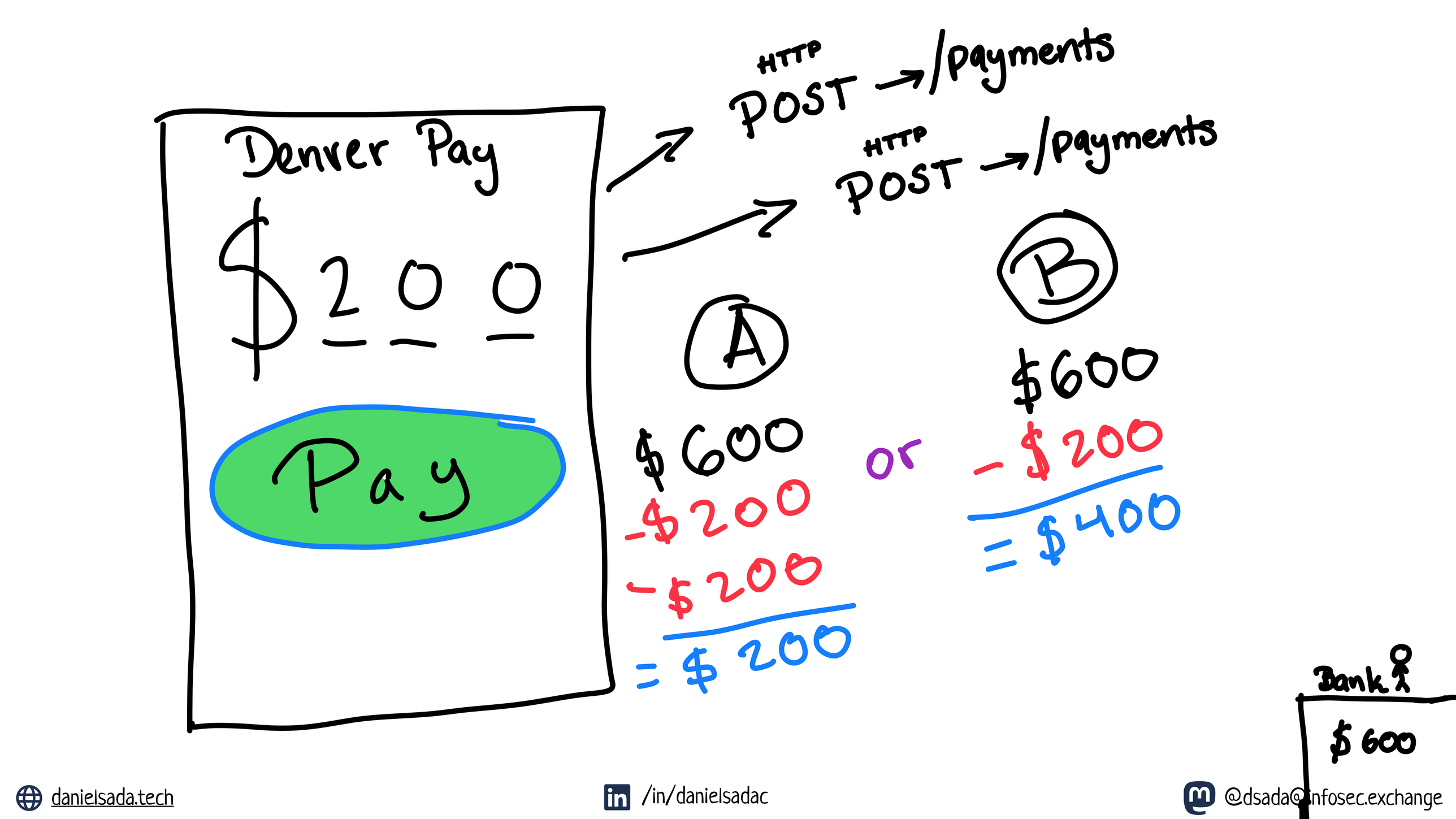
*But alerts don't let you sleep at night :(

Idempotence


Denver Pay
$200
Pay
HTTP
POST
/payments
Bank
$ 600


Denver Pay
$200
Pay
HTTP
POST
/payments
$600
- $200
= $400
Bank
$400


Denver Pay
$200
Pay
HTTP
POST → /payments
HTTP
POST → /payments
Twice?
Bank
$ 600


Denver Pay
$200
Pay
HTTP POST → /payments
HTTP POST → /payments
A
$600
-$200
-$200
= $200
or
B
$600
- $200
= $400
Bank
$600

Idempotence

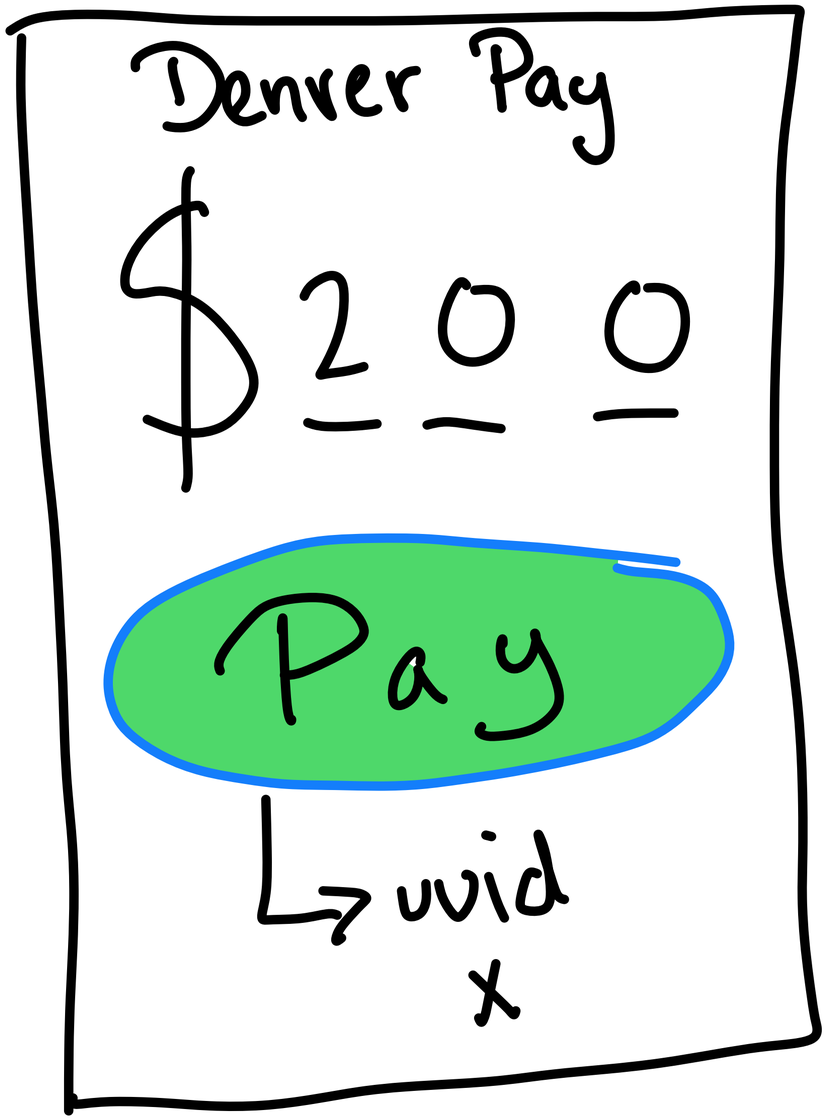
"the property of certain operations in mathematics and computer science that they can be applied multiple times without changing the result beyond the initial application."

https://en.wikipedia.org/wiki/Idempotence

Times ran		Effect
1	`UPDATE students SET first_name = 'John' WHERE id = 123;`	
2	`UPDATE students SET first_name = 'John' WHERE id = 123;`	
3	`UPDATE students SET first_name = 'John' WHERE id = 123;`	
4	`UPDATE students SET first_name = 'John' WHERE id = 123;`	
5	`UPDATE students SET first_name = 'John' WHERE id = 123;`	
…		

https://ieftimov.com/posts/understand-how-why-add-idempotent-requests-api/

One of many solutions: Idempotency Keys

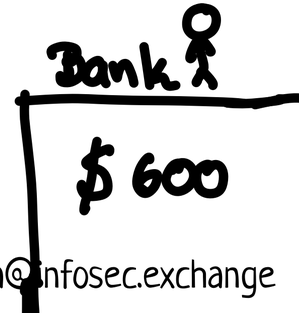


HTTP POST → /payments · id = uuid x ✓

HTTP POST → /payments · id = uuid x ✗

HTTP POST → /payments · id = uuid x ✗

$600
- $200
= $400



We could still talk about:

Fault tolerant applications
Chaos engineering
How orchestration platforms solve some of this problems
Observability.

But that is left as an exercise to the viewer.

And with all this architecture tips...

I wish you a good night's sleep...

(Maybe?)

DANIEL SADA

Developer Productivity & Culture

Thank you!

danielsada.tech

/in/danielsadac

@dsada@infosec.exchange