From eval to prod

How a service mesh helped us build production cloud-native services

@zot24 from @moltin

What will you learn today? 

You'll be able to understand the **needs** and **problems** we faced when we built our **distributed system.**

Summary 

- What we do at moltin?
- State of art before (micro)services
- State of art after (micro)services
  - Microservices
  - Service Mesh
- What's next?

What we do at moltin? 

- API-First
- Commerce experience

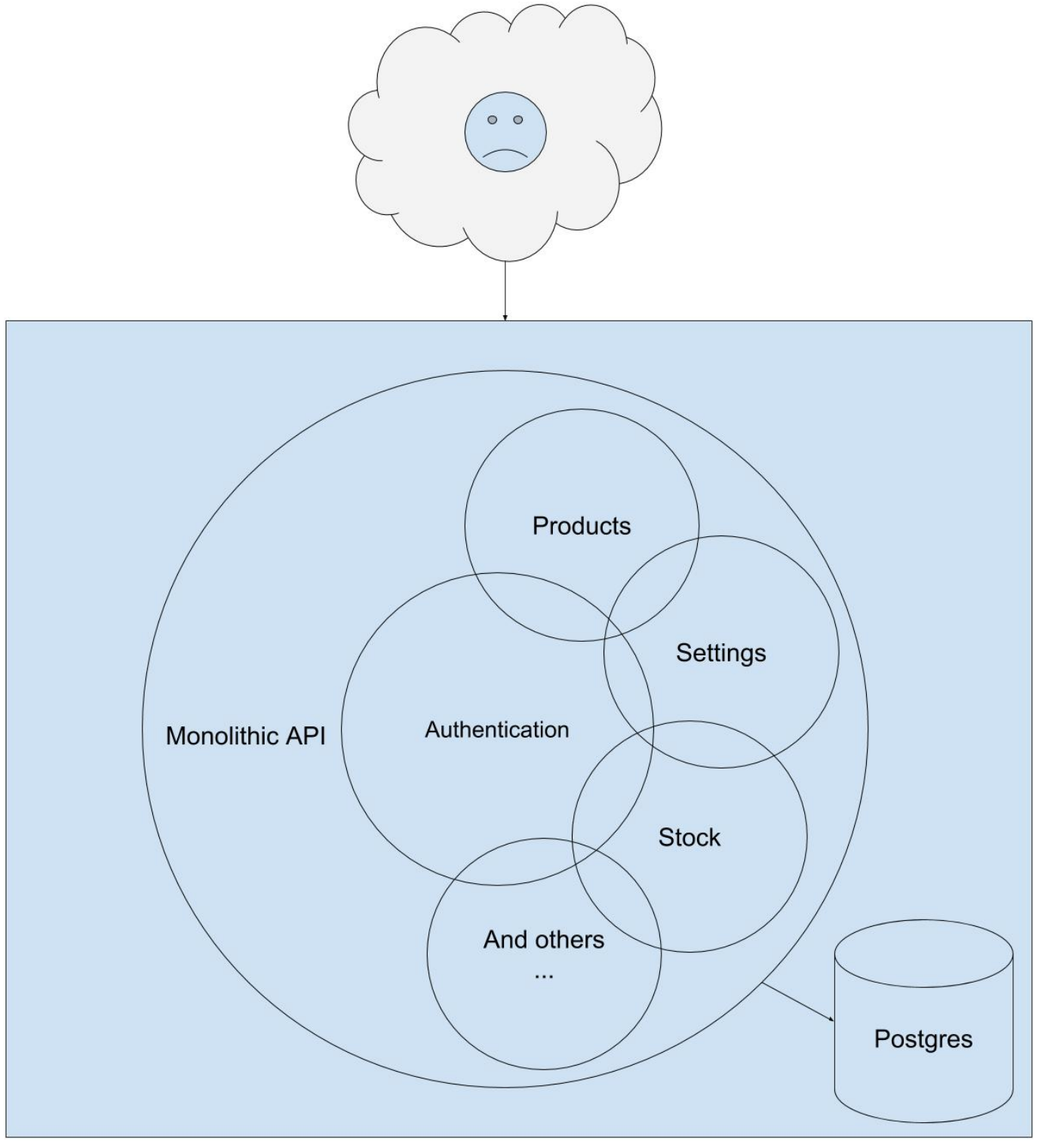
State of art before (micro)services

2 years ago ⏳

- A huge monolithic app
- Highly coupled code
- No tests at all
- Poor performance
- High latency responses

=

So this is how we looked like at that point


Products
Settings
Monolithic API
Authentication
Stock
And others
...
Postgres

Our first tasks 

- Increase performance 📈
- Decrease response latency 📉

How did we solve it?

- Identified the most use parts of our system
- Identified the slowest one
- Improve it!

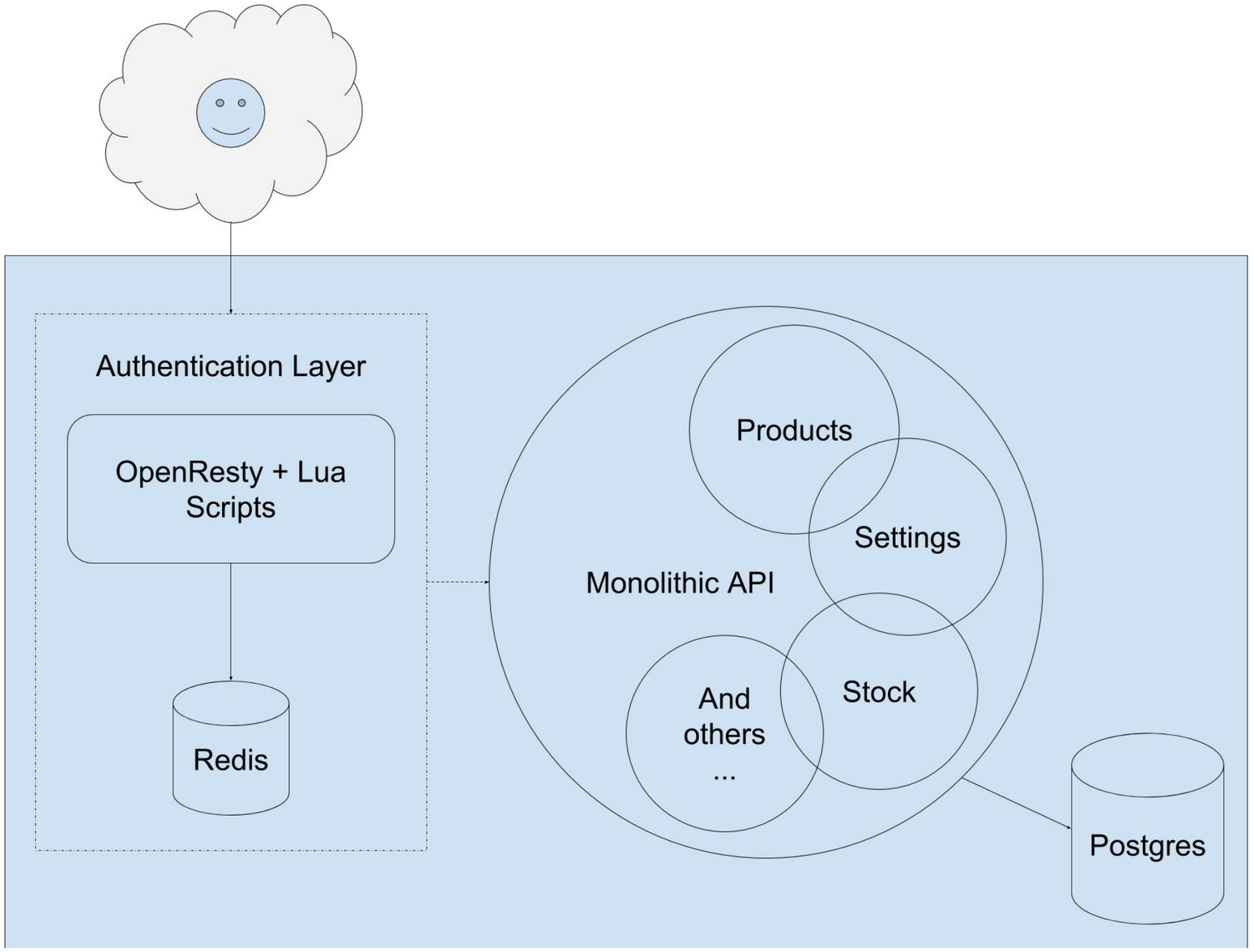
The most used and slowest performing part

- Authentication system 🔐
- Around ~500ms to request an access token 😰

How did we approach a solution?

- Tweaking 👷
- From 500ms to 250ms = 😐
- Decoupling 💔
- Our first "(micro)service" 👶
- The right tool 🔧 for the job 🚙
- From 250ms to ~40ms 😱 = 😀

So this is how we looked like at that point


Authentication Layer
OpenResty + Lua Scripts
Redis
Products
Settings
Monolithic API
And others ...
Stock
Postgres

State of art after (micro)services

Microservices

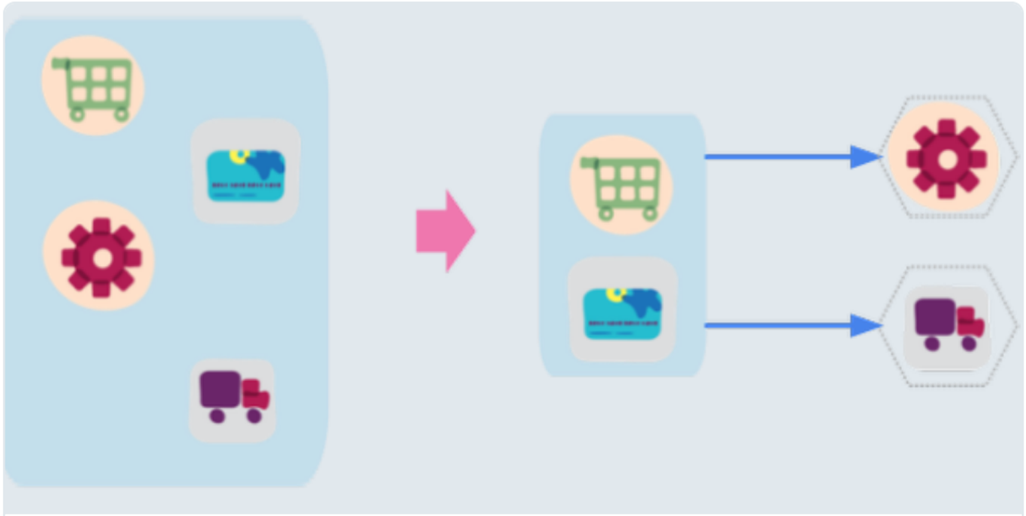
Why (micro)services?

- We needed to improve our system performance 📈 and reduce our request latencies 📉

but... how?

Martin Fowler @martinfowler · Apr 24

new post: Do you want to split a monolith into microservices? If so @zhamakd has been down that road and has lessons to share



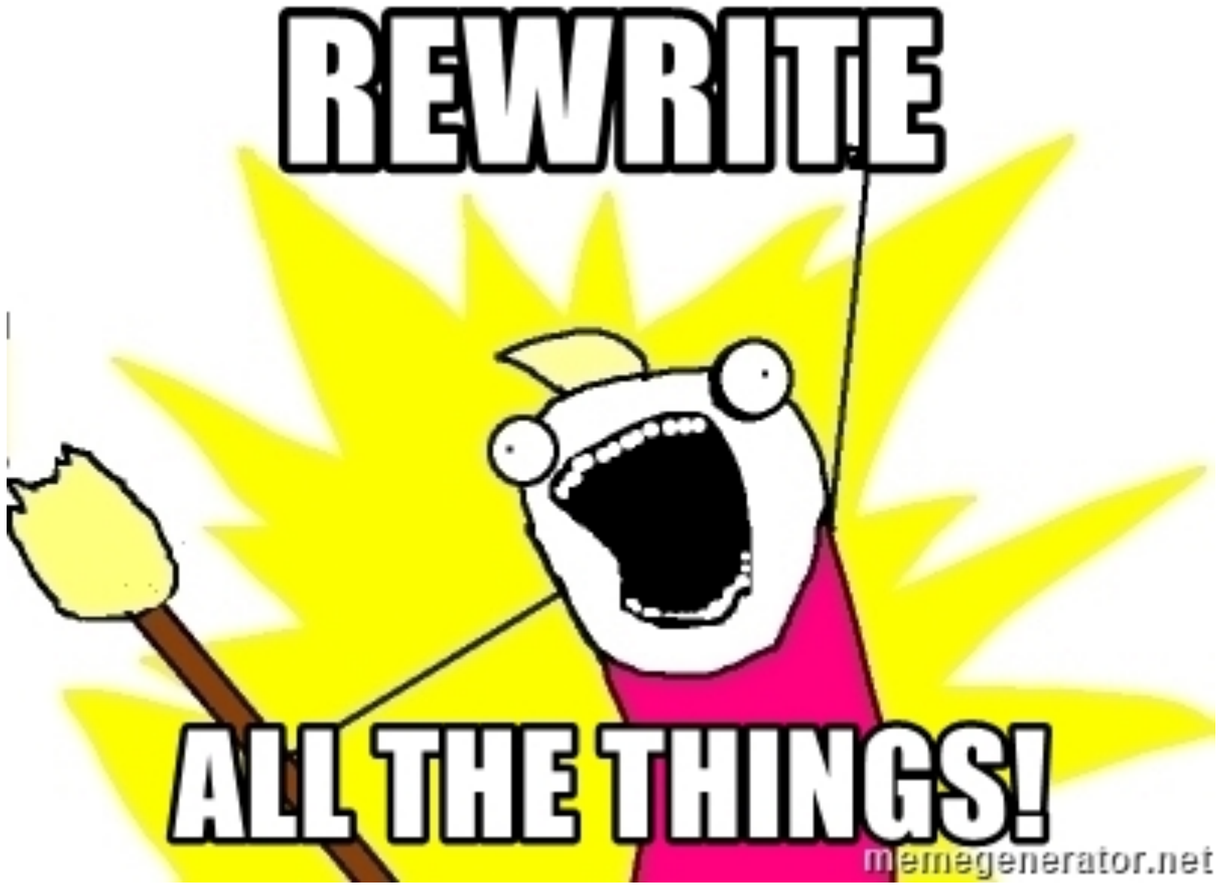
How to break a Monolith into Microservices

A guide to the common steps we've observed in breaking a monolithic application up into microservices

martinfowler.com

9 475 982

How did we approach it?


REWRITE
ALL THE THINGS!
memegenerator.net

Sandeep Dinesh
@SandeepDinesh

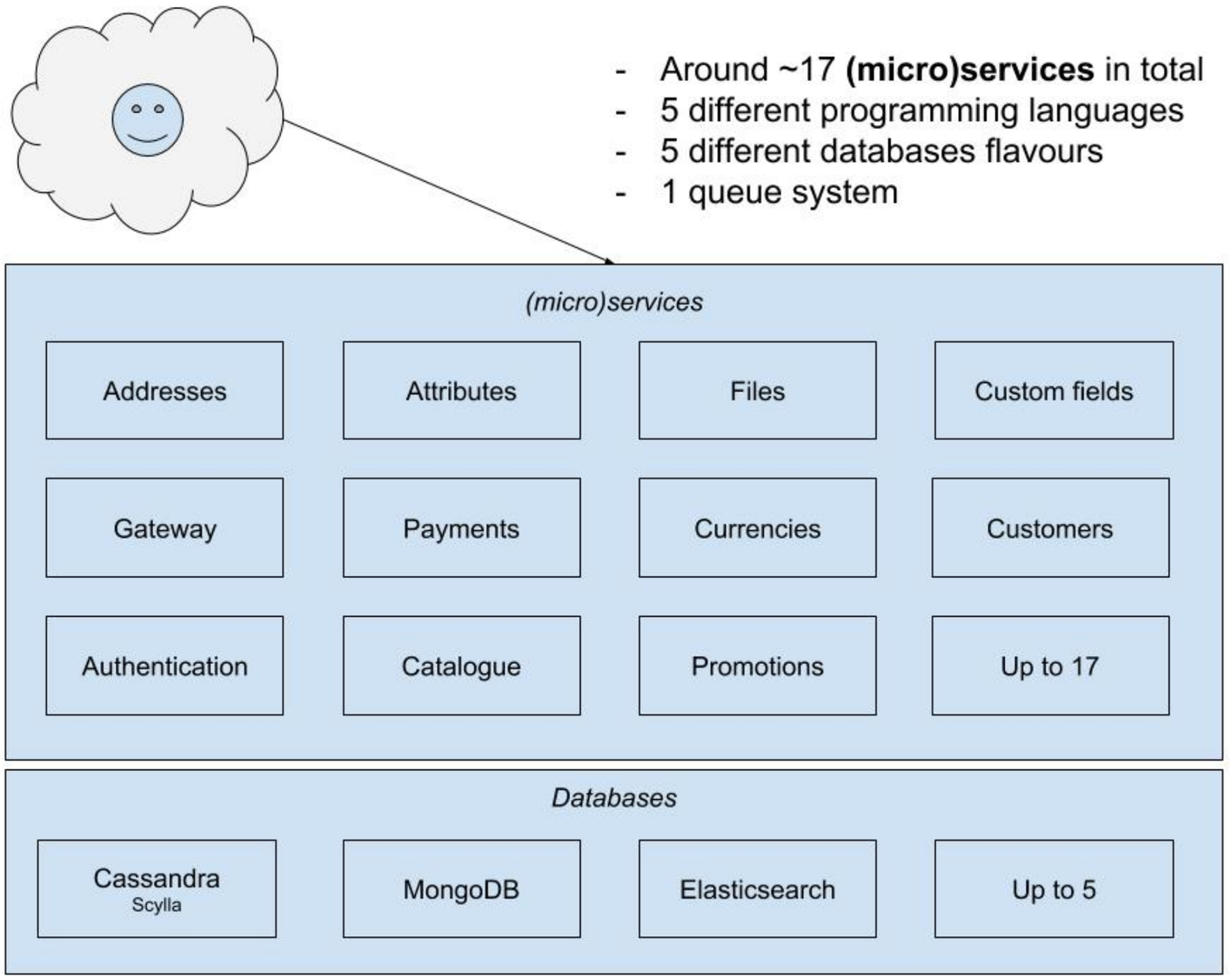
Following

Replying to @SandeepDinesh @cwienczek and 2 others

A good company/team will try to disrupt themselves by building the replacement WHILE maintaining the current system. Much easier said than done, but that's why software engineers are paid so well right?

7:23 PM - 27 Apr 2018

So this is how we looked at that point


- Around ~17 **(micro)services** in total
- 5 different programming languages
- 5 different databases flavours
- 1 queue system
(micro)services
Addresses
Attributes
Files
Custom fields
Gateway
Payments
Currencies
Customers
Authentication
Catalogue
Promotions
Up to 17
Databases
Cassandra
Scylla
MongoDB
Elasticsearch
Up to 5

State of art after (micro)services

Service Mesh

What's a Service Mesh?

A dedicated infrastructure layer for service to service communication decoupled from your application code and focus on services and requests

Why did we need a service mesh in the first place?

Service to service communication needs to be managed, monitored and controlled

Observability

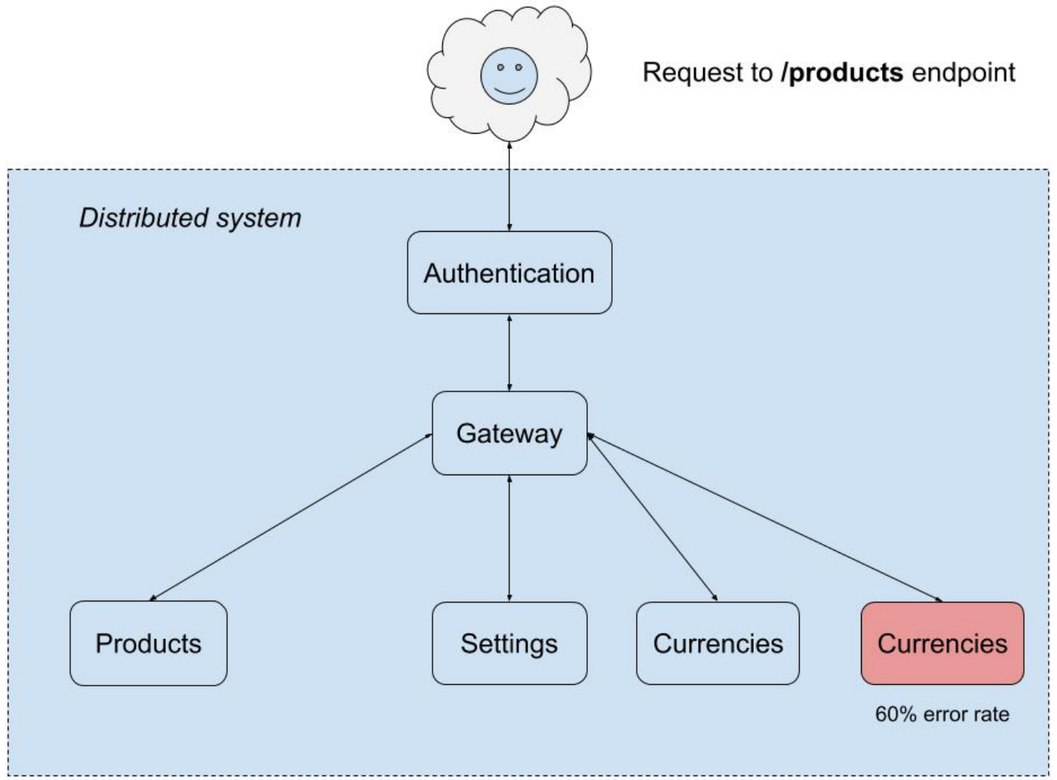
 Development

- Did my request got from A to B?
- Where is it hanging?
- Why my system is so slow?

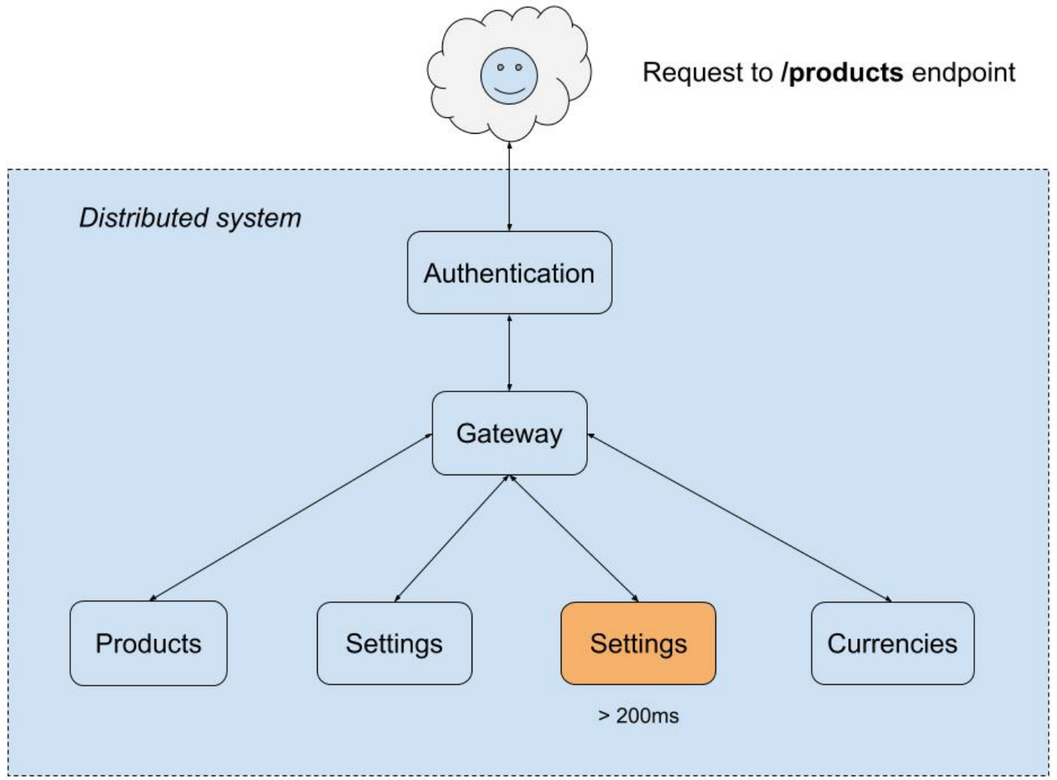
🚀 Production

- Improve resilience - Circuit breaking
- Improve success rate - Retries
- Reduce tail latency - Load Balancing
- Metrics and Tracing - Observability

Circuit Breaking problem



Latency aware as a problem



What's out there?

Service Mesh Solutions

- Linkerd
- Conduit (Linkerd for k8s)
- Envoy (from Lyft)
- Istio

Why Linkerd?

- Vendor agnostic
- Stronger option
  - Well adopted by the community
  - Trusted by multiple companies in production
- Great support

What is it really?

Is just **another proxy** but the feature set and focus is very different

A -> B

A -> L -> B

Will be my system's performance affected by introducing Linkerd?

YES it will!

Your best case will take a hit (a small performance implication)

Your worst case will have a huge performance improvement

What's your goal in a distributed system?

- Reduce your tail latencies p99
- Have a stable and resilient system

What else can we do with Linkerd?

Load Balancing

Logging

Error tracking

Tracing

Metrics

Retry budgets

Service Discovery

Expirations

Circuit Breaking

Canary Releases

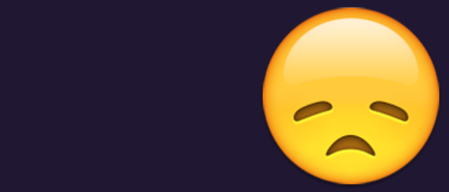
Retries

Back-offs

Dynamic routing

Timeouts

How much of all this are we using?

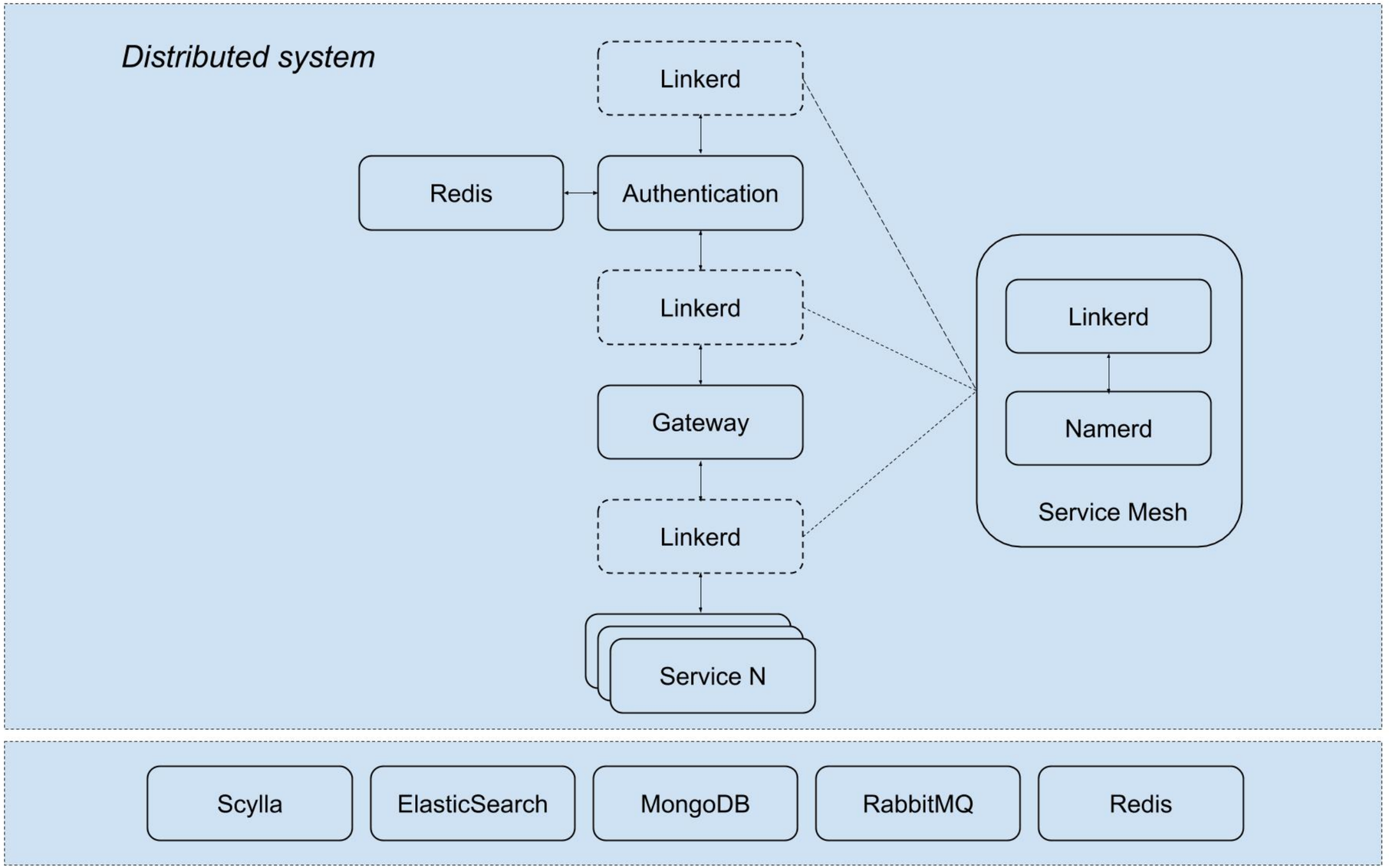


Not even the 15% of all features

📮 it just worked out of the box 🙌

with minimal configuration

So this is how we looked at that point

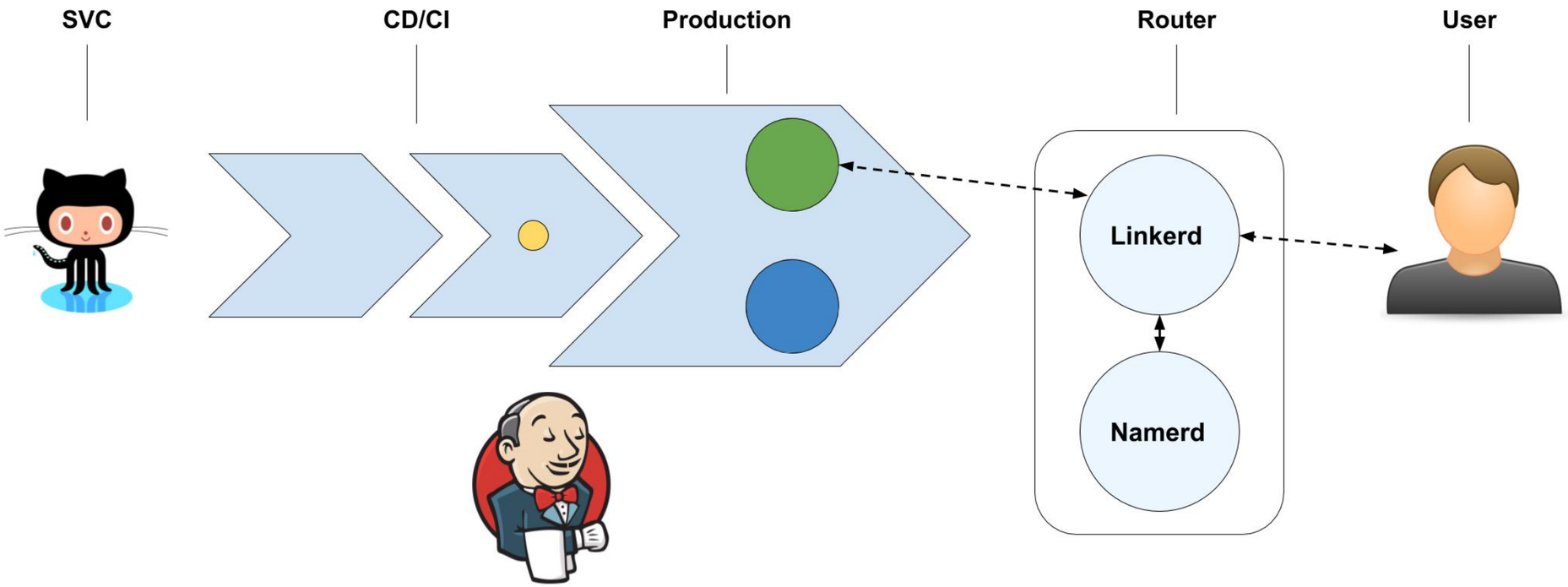

Distributed system
Linkerd
Redis
Authentication
Linkerd
Gateway
Linkerd
Service N
Linkerd
Namerd
Service Mesh
Scylla
ElasticSearch
MongoDB
RabbitMQ
Redis

What's next?

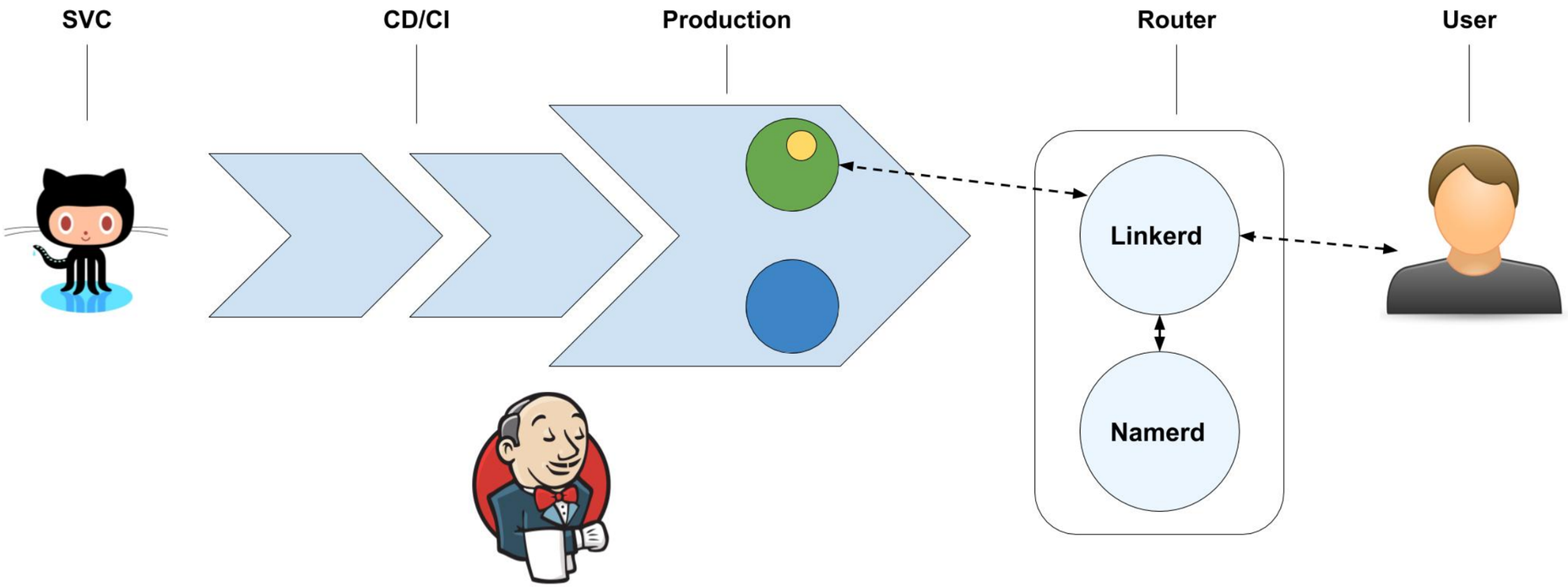
We want to do more cool 😎 things

- Canary deployment
- Shadowing traffic
- Dark Launches

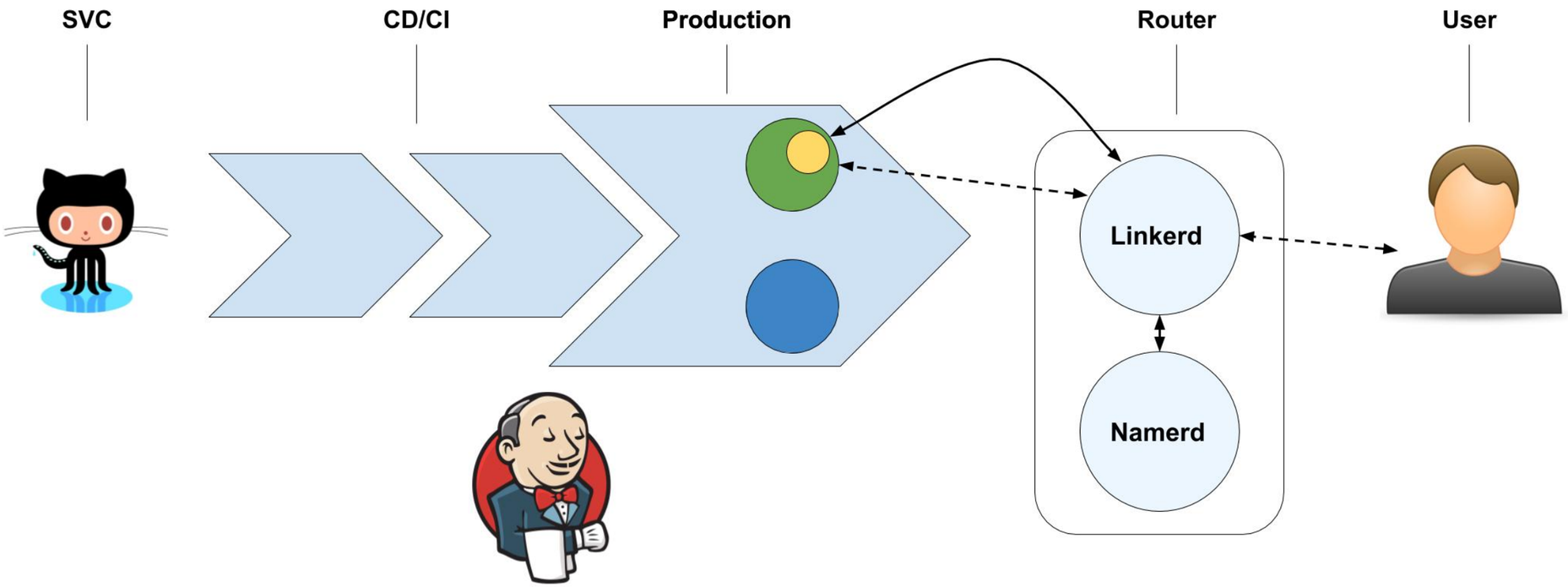
Canary deployment



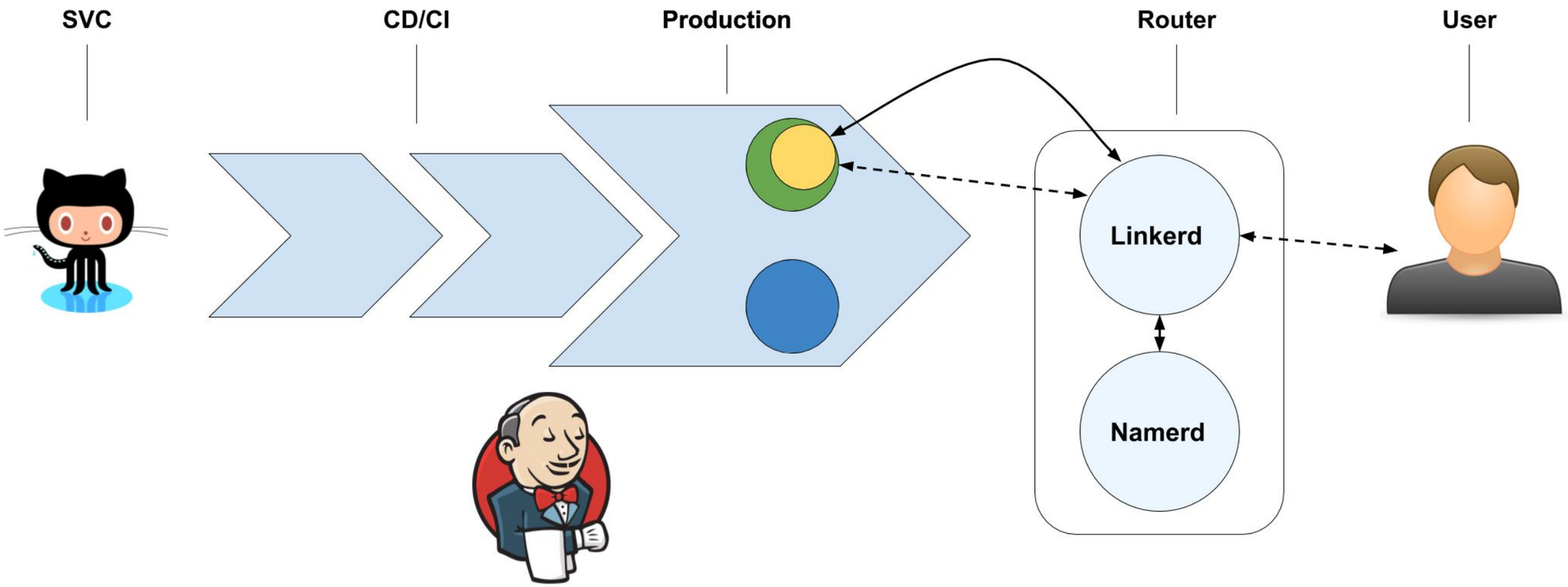
Canary deployment



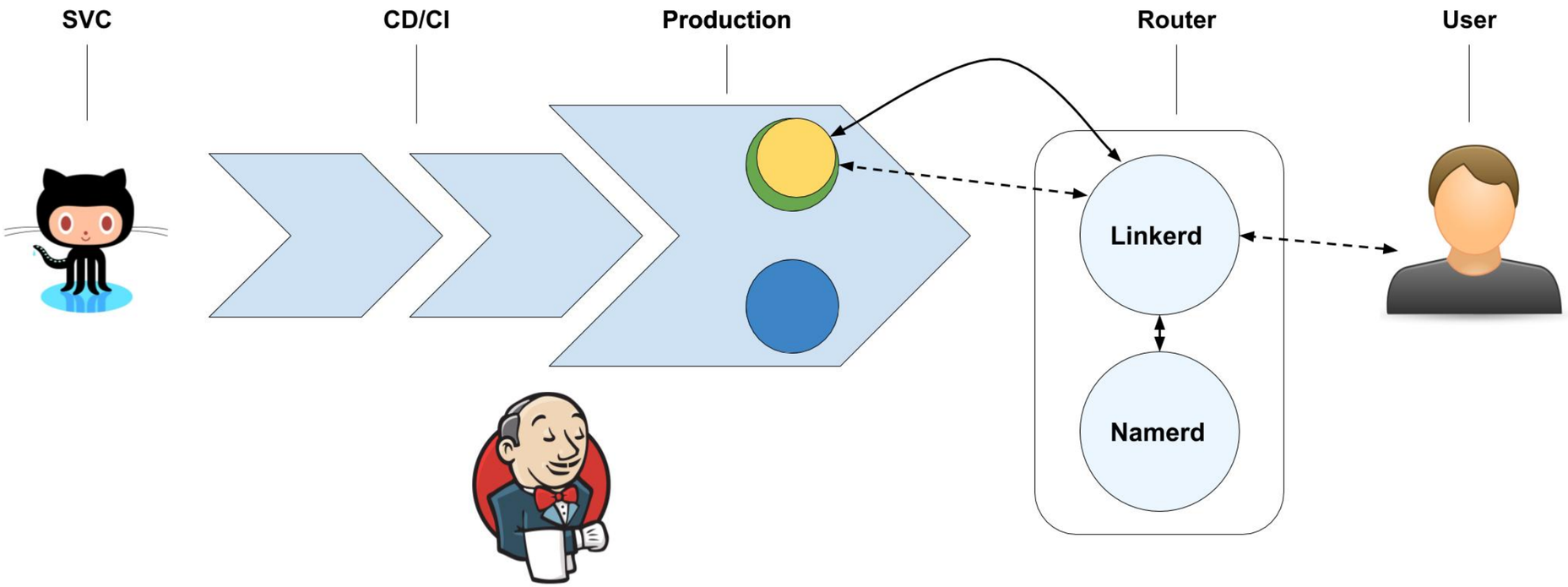
Canary deployment



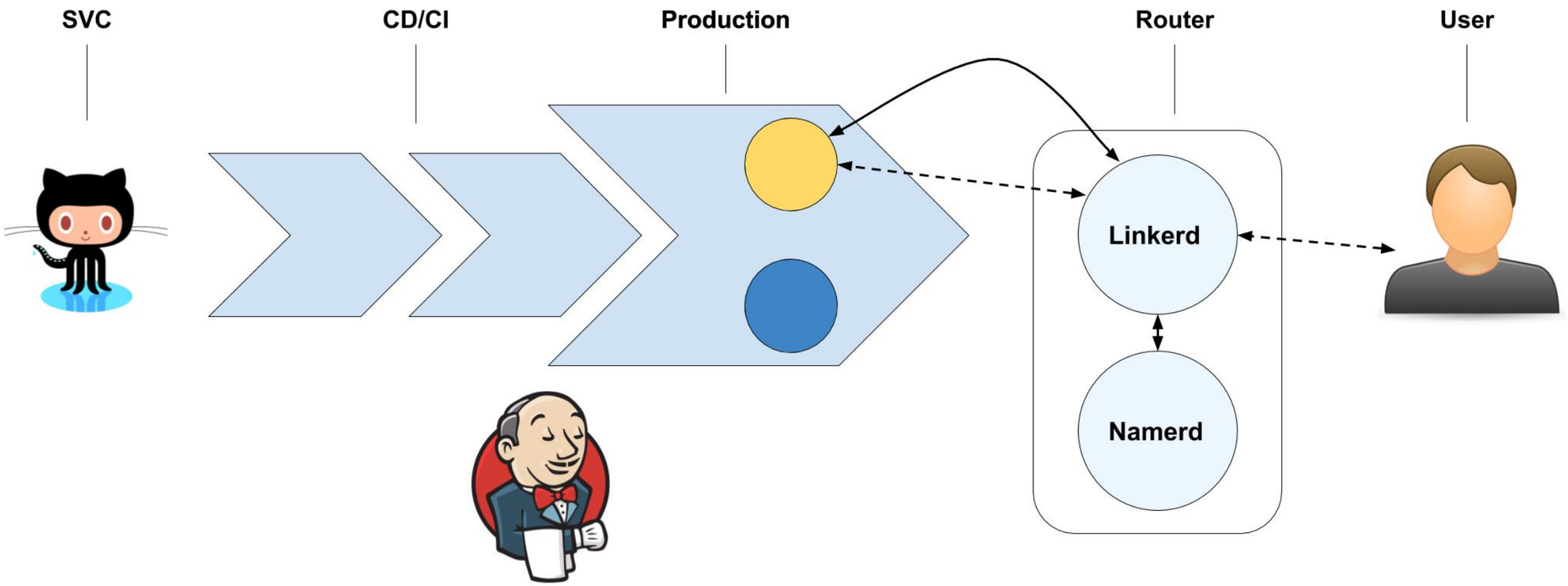
Canary deployment



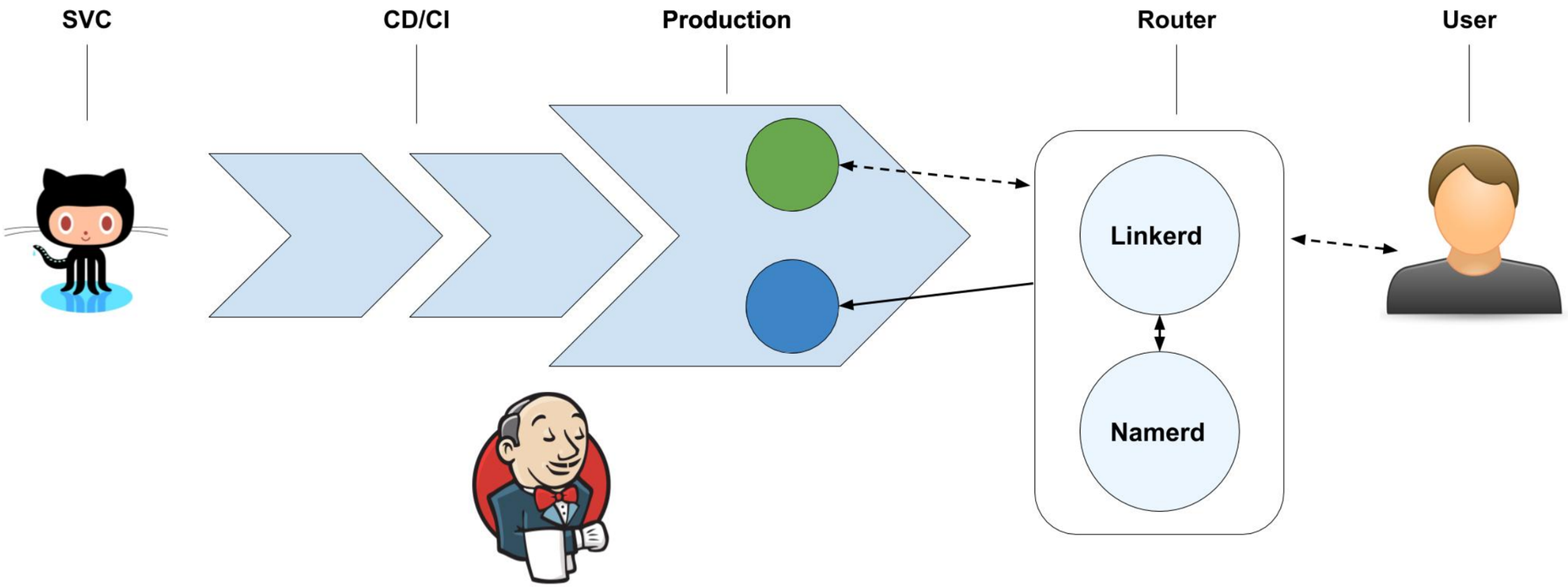
Canary deployment



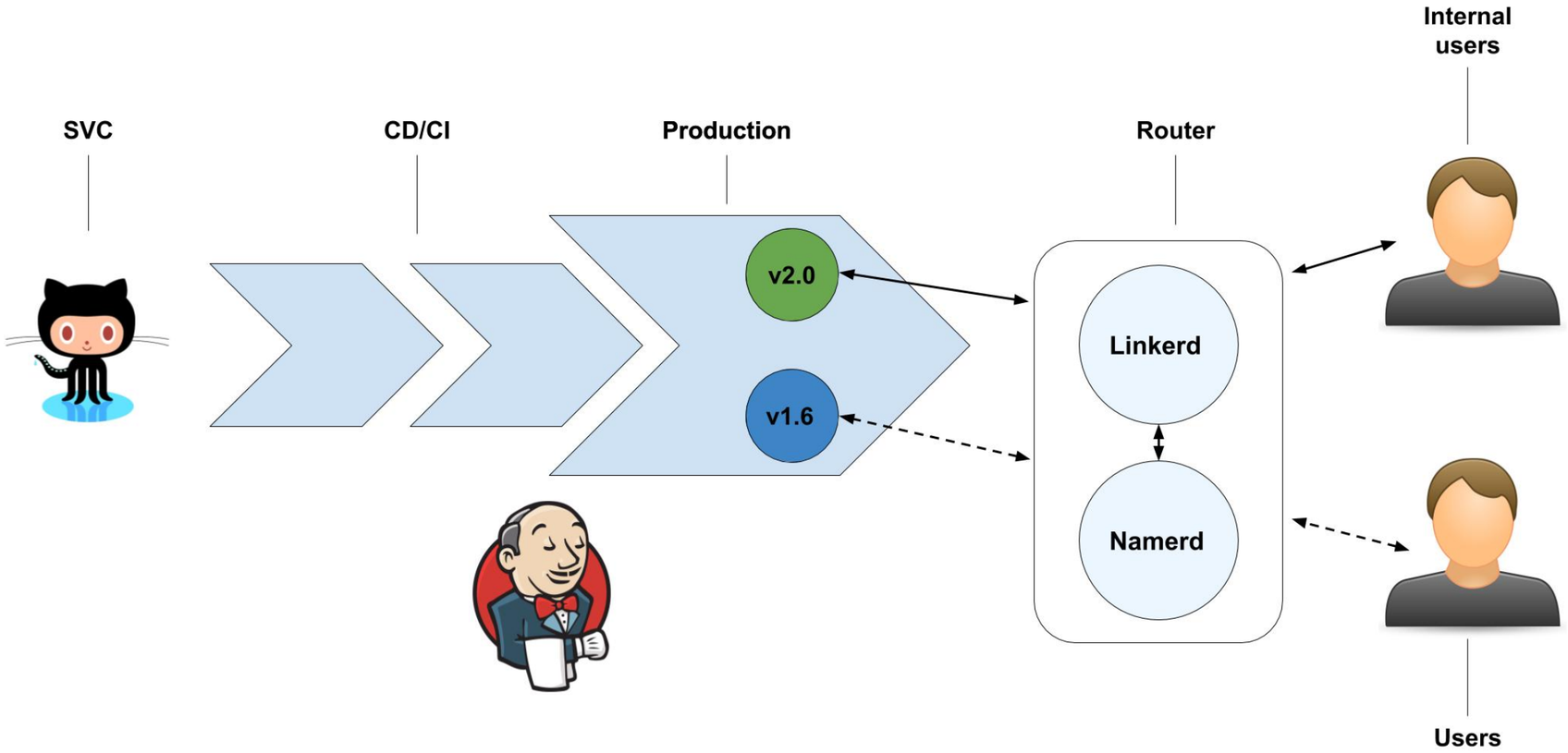
Canary deployment



Shadowing traffic



Dark Launches



Looking forward to...?

- Adopt to Kubernetes
- Eval Conduit

Take Away 🔑

- Network communication is complicated
- Apps shouldn't have to reinvent this wheel
- Libraries are language specific

so...

You will need a service mesh if you are building a distributed system

demo⌛ ❓

Thanks you! 🐲👏

- I'm Israel Sotomayor 🇻🇪🇪🇸
- Infrastructure Engineer 🌋
- Digital Nomad 🌍
- Enjoy automating things 🔧
- ❤️ Traveling ✈️

@zot24 🐲 from @moltin 🌋