The hotel wireless was terrible, sorry for the lack of pictures.

$ whoami

$ whoami

```
$ cat .profile | grep export
export GIT_AUTHOR="Florian Gilcher"
export GIT_AUTHOR_EMAIL="florian.gilcher@asquera.de"
export GITHUB_NICK="skade"
export GITHUB_ORGANIZATIONS="asquera,padrino"
export TWITTER_NICK="@argorak"
export TM_COMPANY="Asquera GmbH"
```

@argorak

$ whoami

Ruby Programmer since 2003

Now a consultant specialising in backends...

... and team building.

I run usergroups and organize conferences as a hobby.

http://asquera.de

http://padrinorb.com

http://eurucamp.org

“I don’t want to be woken up at night, so I call myself a developer.”

I set out to present a more dev-minded perspective on devops.

That was harder then I thought...

There's a talk about the "transforming devs to devops" later on.

```
git push developer mindsets/devops
```

Whats the benefit, if you don't do a lot of ops?

Vagrant
Puppet
Chef

~~Vagrant~~

~~Puppet~~

~~Chef~~

How did a devops mindset improve the software I write?

A bit of history about myself

I started my career in a typical agency job.

LAMP and the DEV/OPS split.

It wasn't even that bad...

...until projects got special.

scale

scope

Suddenly it turned out that one of the most efficient teams was an admin, a programmer and a cup of coffee.

Example

An example

One of our clients imports and reencodes videos from a constantly changing number of sources each day.

Sources

FTP upload

FTP fetch

RSS feeds

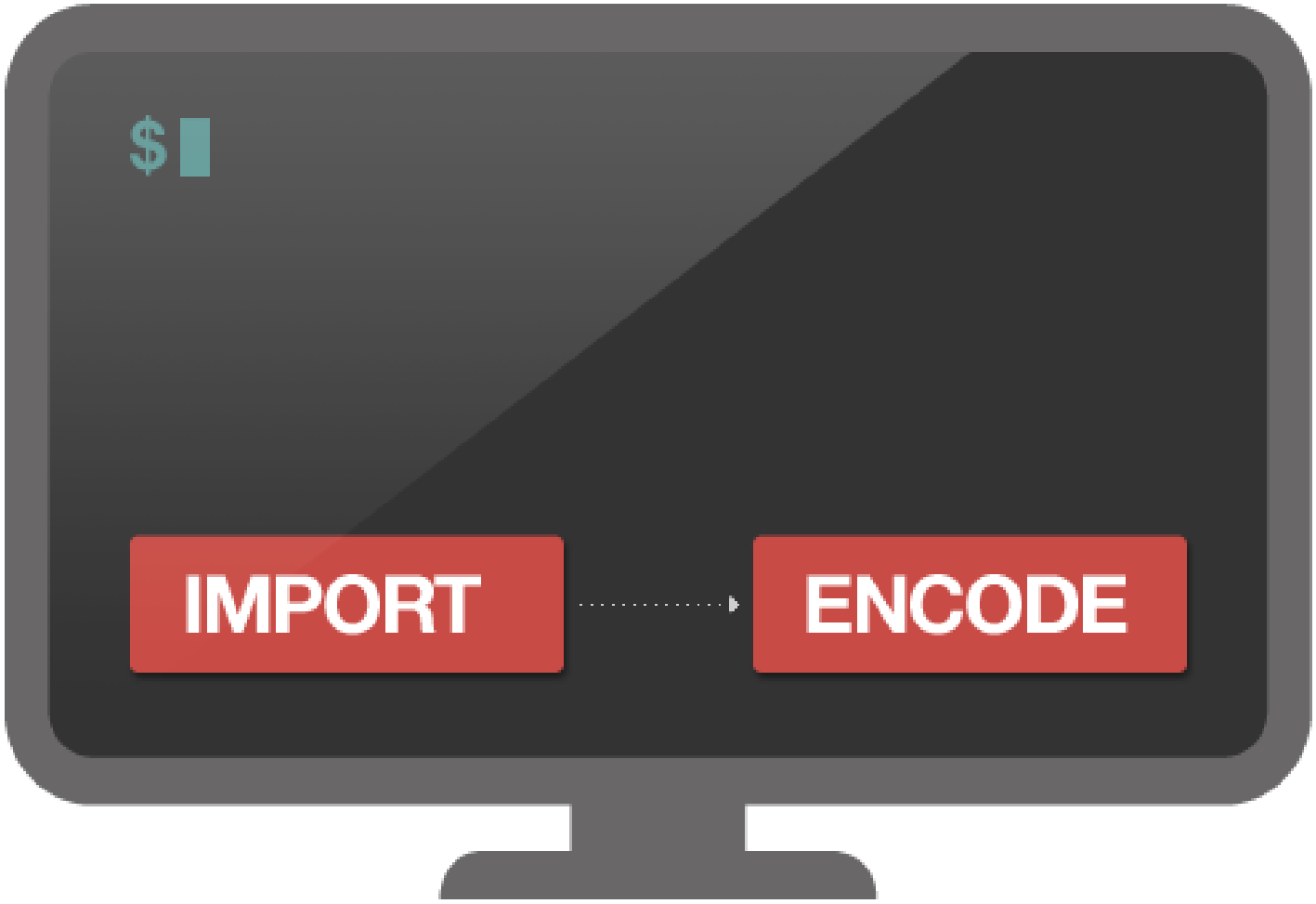
RSS feeds that are no RSS feeds

And some more...

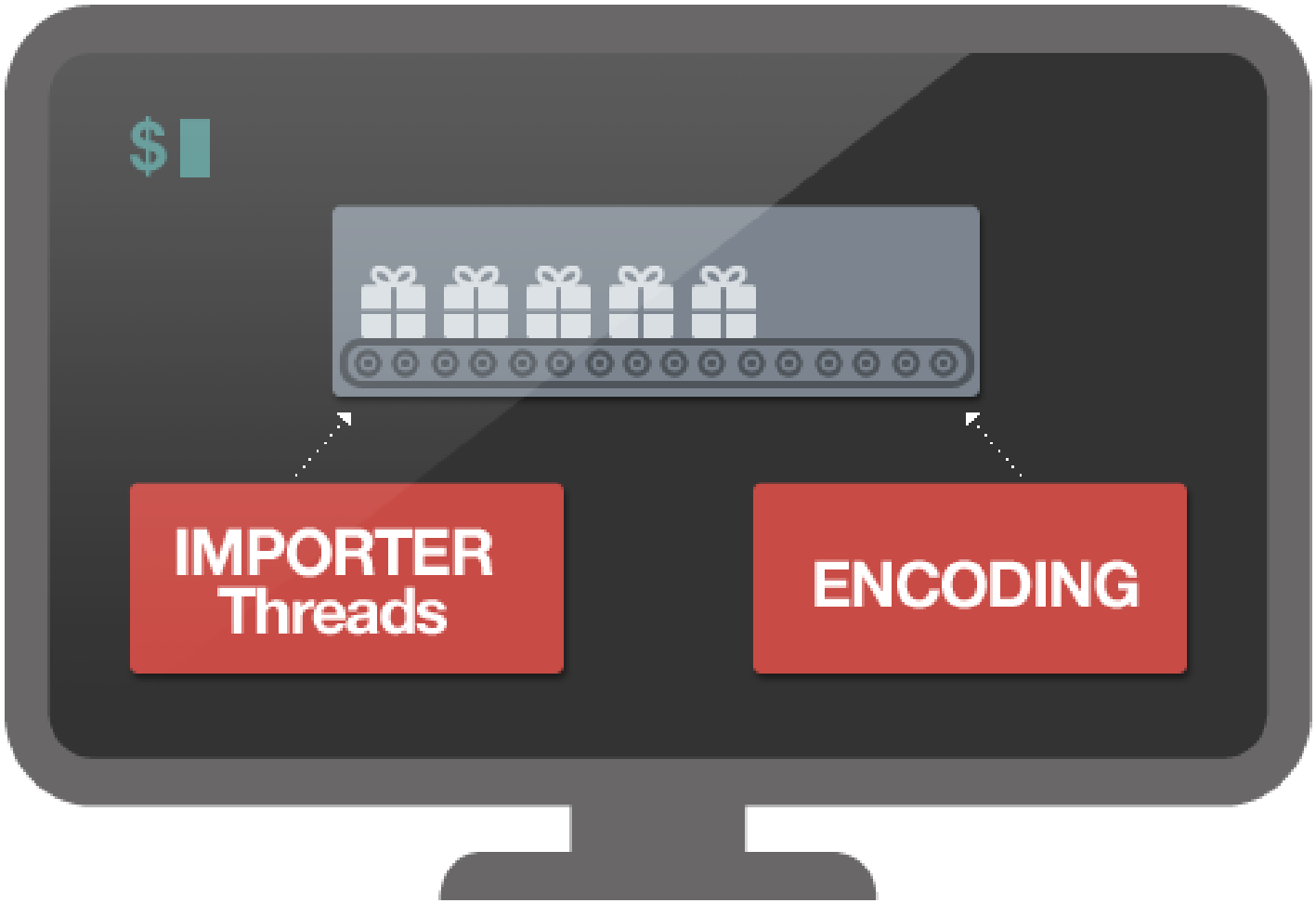
Destinations

All of them need to be reencoded to a standard set of sizes and bitrates.

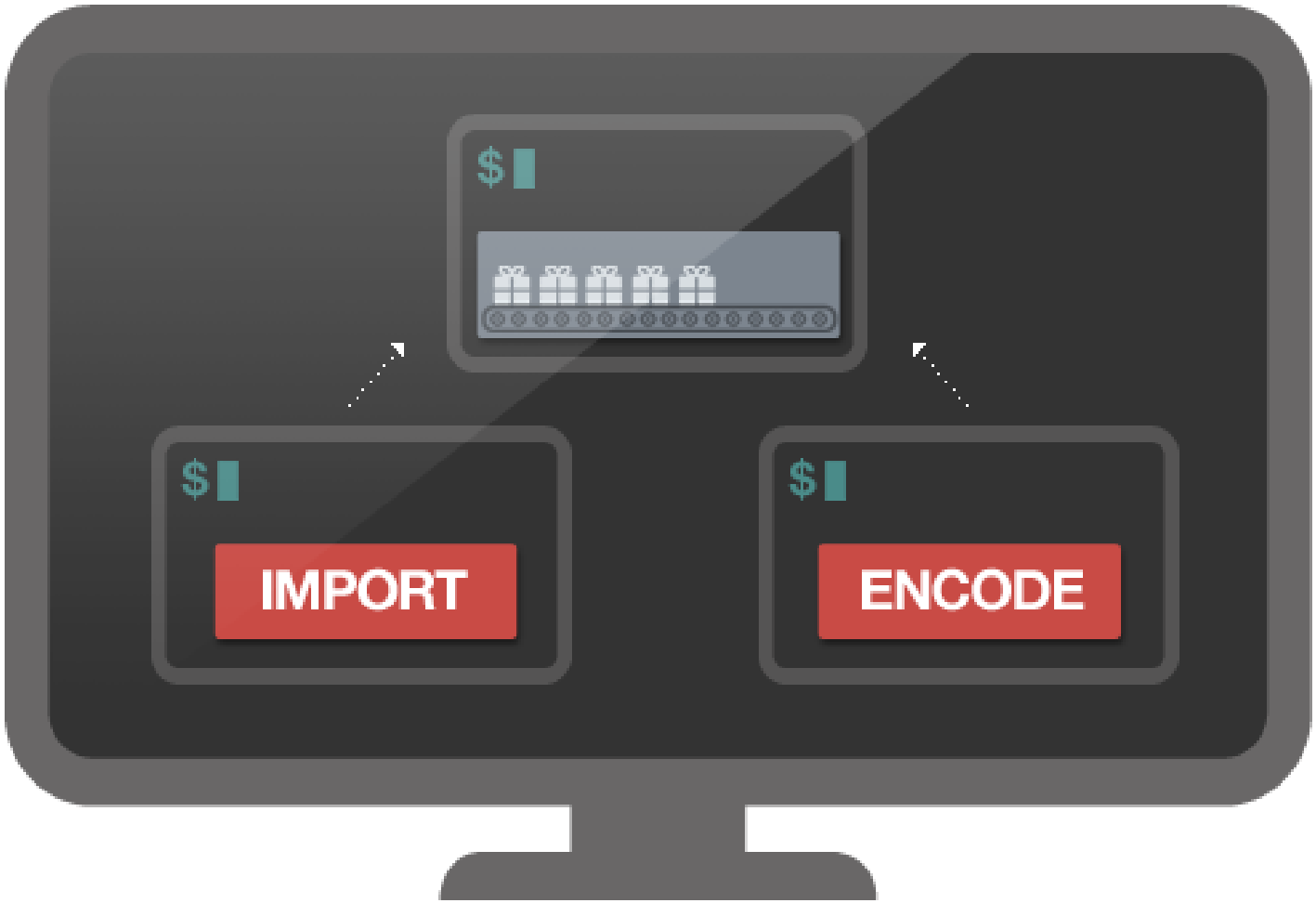
Simple approach



Single program with architecture

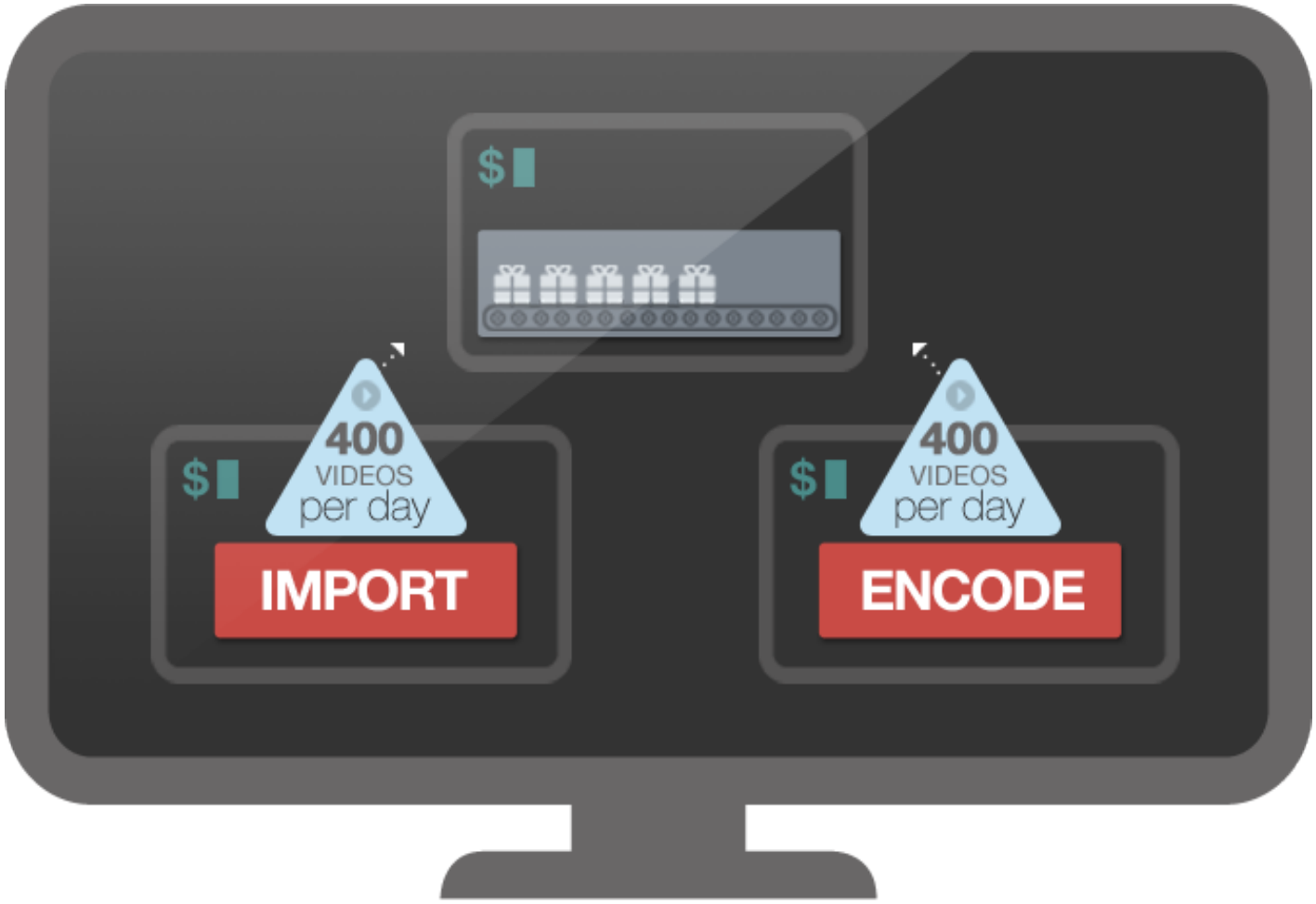


Same architecture, 3 processes

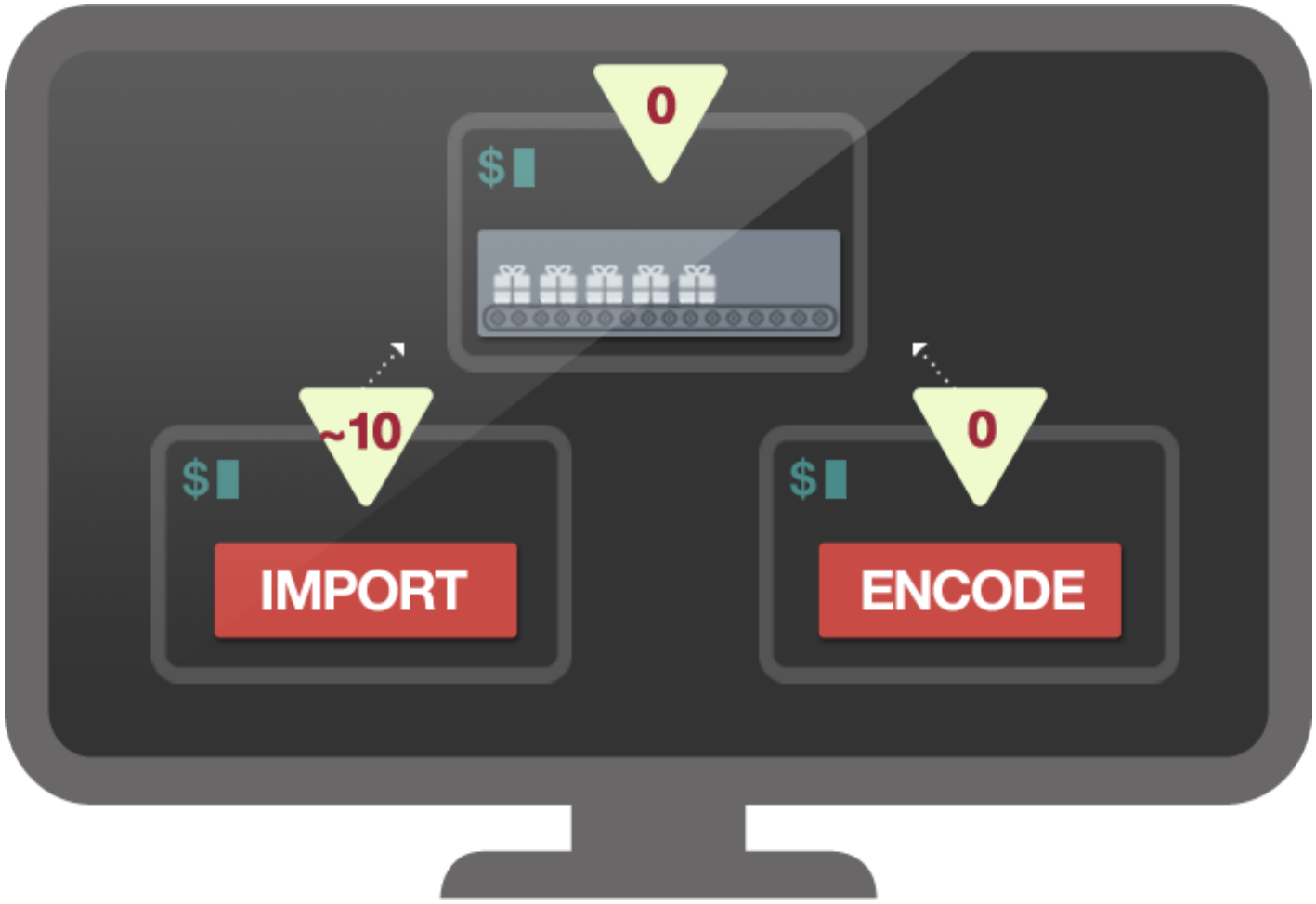


Why?

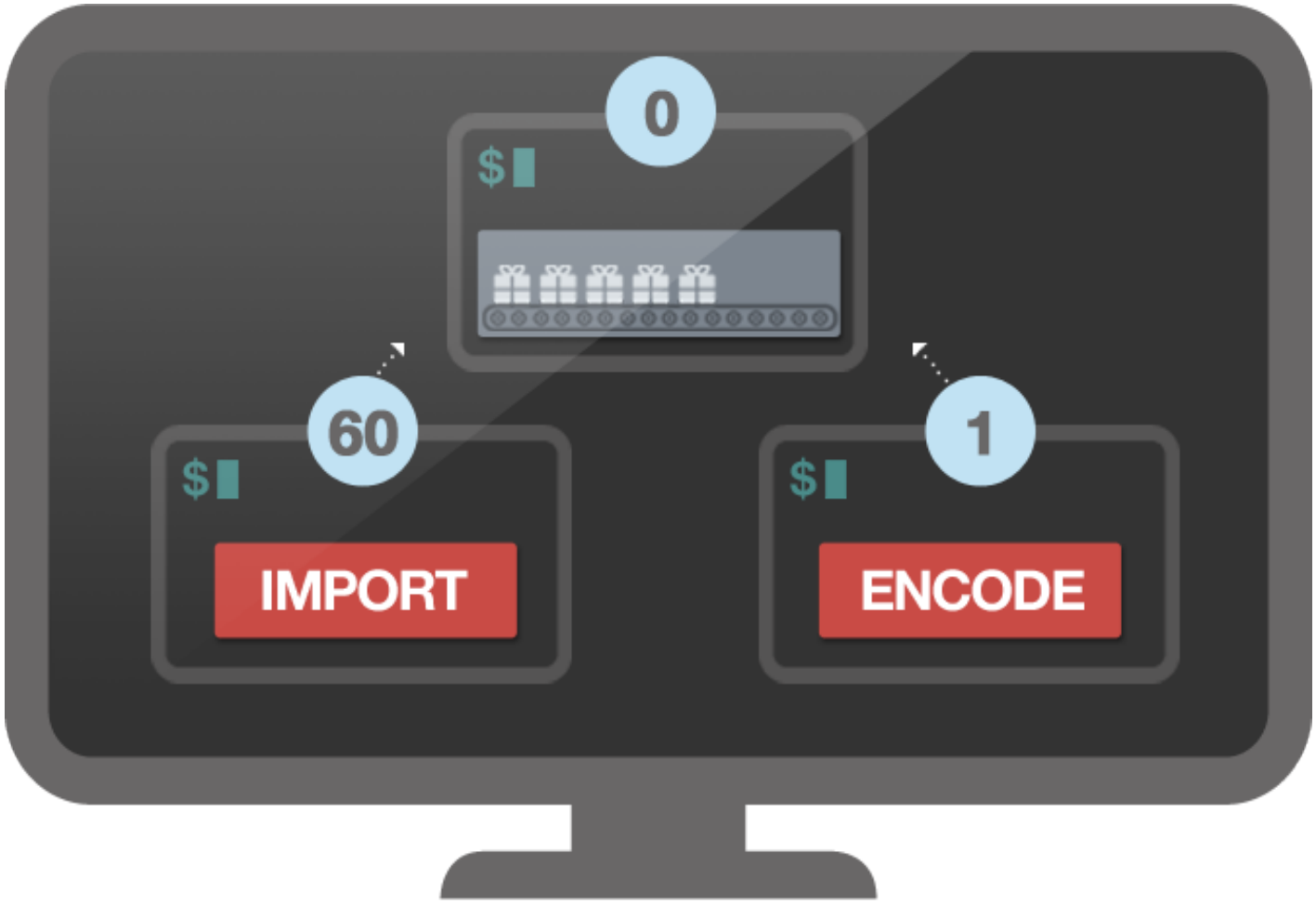
Videos per day



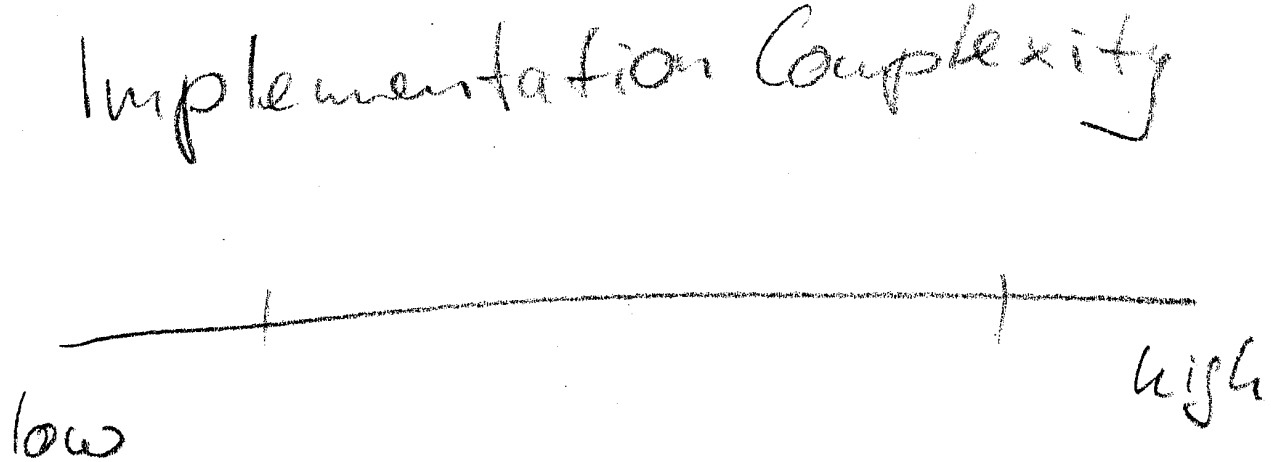
Critical failures last year

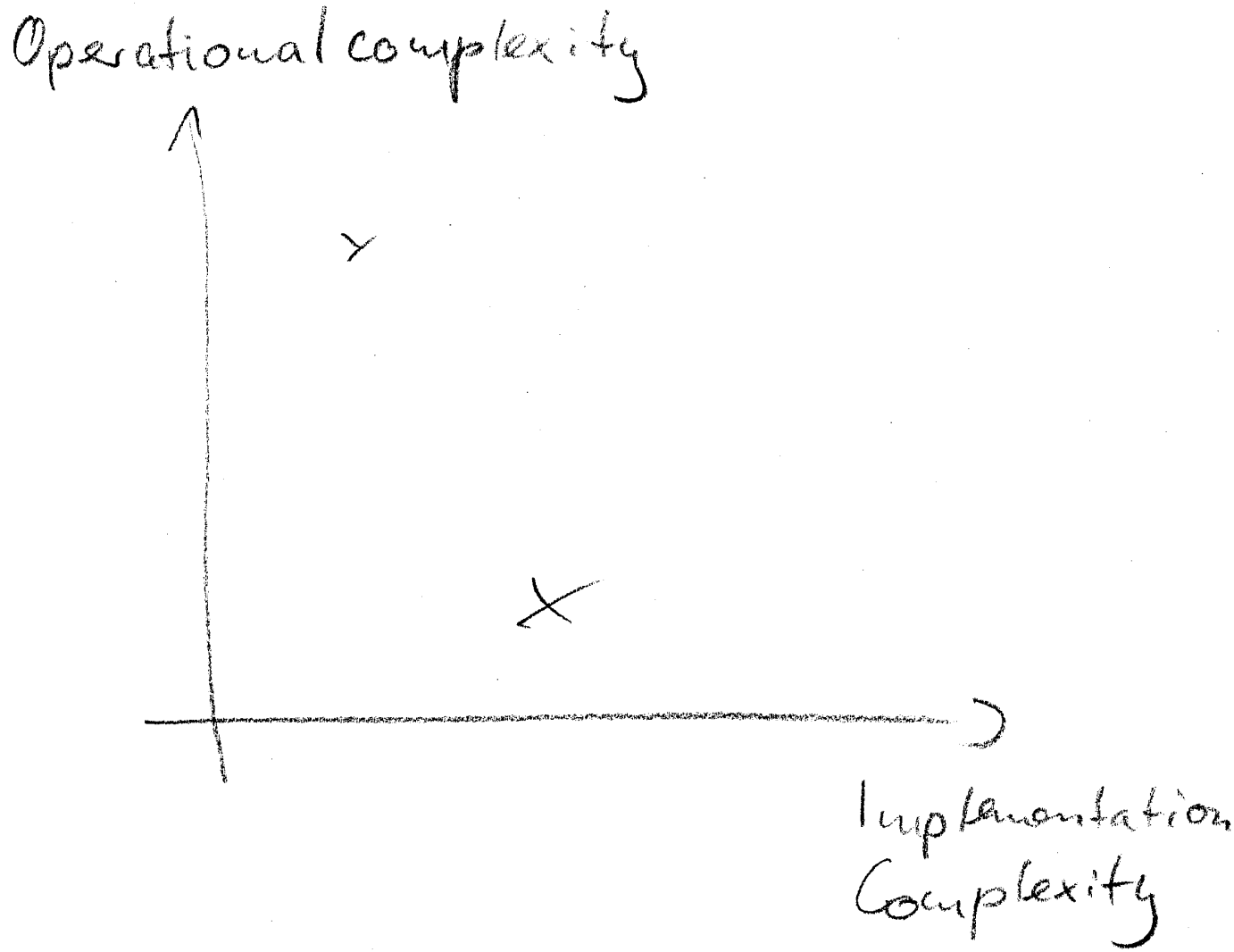


Deployments last year



New ways of discussing things.


Implementation Complexity
low
high


Operational complexity
Implementation
Complexity

Practical things learned in the process.

... beyond writing daemons and stuff.

A different perspective on code.

Infrastructure as code.

Code for infrastructure.

Common CLI tools
Common configurations styles
Common way of doing things

Gives insight

Well managable

Well automatable

In general, I care less about internal quality of programs nowadays then about external quality.

My ugliest piece of code ran 1,5 years in production without a change.

Nobody ever noticed how horrible it was.

I evaluate new software differently.

“Ease of setup” is a red flag.

This especially applies to new and fancy databases.

Not having to push any buttons to start working a database is problematic, if not dangerous.

You might miss things along the way.

“Ease of non-trivial configuration”
is far more important.

How to grade that?

Set up a production-like system.

Keep tally marks on how often it leaves you puzzled.

There is no such thing as “setting up production too early.”

Everything before that is childs play.

The big bangs always happen in production.

My favourite: Expensive loadbalancers that die during configuration and need to be shipped to the manufacturer.

Teams need to get used to their own systems.

Metrics and Logging are important in complex systems.

Most pure development teams underrate them and implement them too late.

They should be there from day one.

Last but not least: internal tooling can save you a lot of work.

Creative ways to talk about it even more.

Overview

New Page | Edit Page | Page History

Version in use

At the moment, we use ElasticSearch 0.17.6

About

ElasticSearch is a search indexer based on Apache Lucene. It is distributed and fault-tolerant, allowing indices to be distributed among multiple nodes. It sports a rich Query DSL in JSON, but can also be queried using the Lucene Query languages.

Important links

- Homepage: http://elasticsearch.org
- Announcements: http://www.elasticsearch.org/blog/
- Bugtracker: https://github.com/elasticsearch/elasticsearch/issues
- API: http://www.elasticsearch.org/guide/

Basic Functionality

ElasticSearch exposes an HTTP interface to one or more Lucene indices as well as a control interface. Each ElasticSearch node is fully functional, there are no master or slave nodes

But what about the humans?

Giving people say in many things makes them discuss many things.

There are two things I rarely see in teams with strict roles.

1. Platform refactorings

Why?

It always means that individual roles loose ground.

Why?

This can get political very quick.

Why?

Lack of skill.

2. Code reviews

Why?

Not enough staff that “is qualified” to review certain code.

The devops mindset takes away a lot of friction.

Less asking permission, more doing.

When your frontend developer changes your backend API, your varnish config, your deployment scripts and the puppet manifests before handing stuff off to review to implement a new feature, you are there.

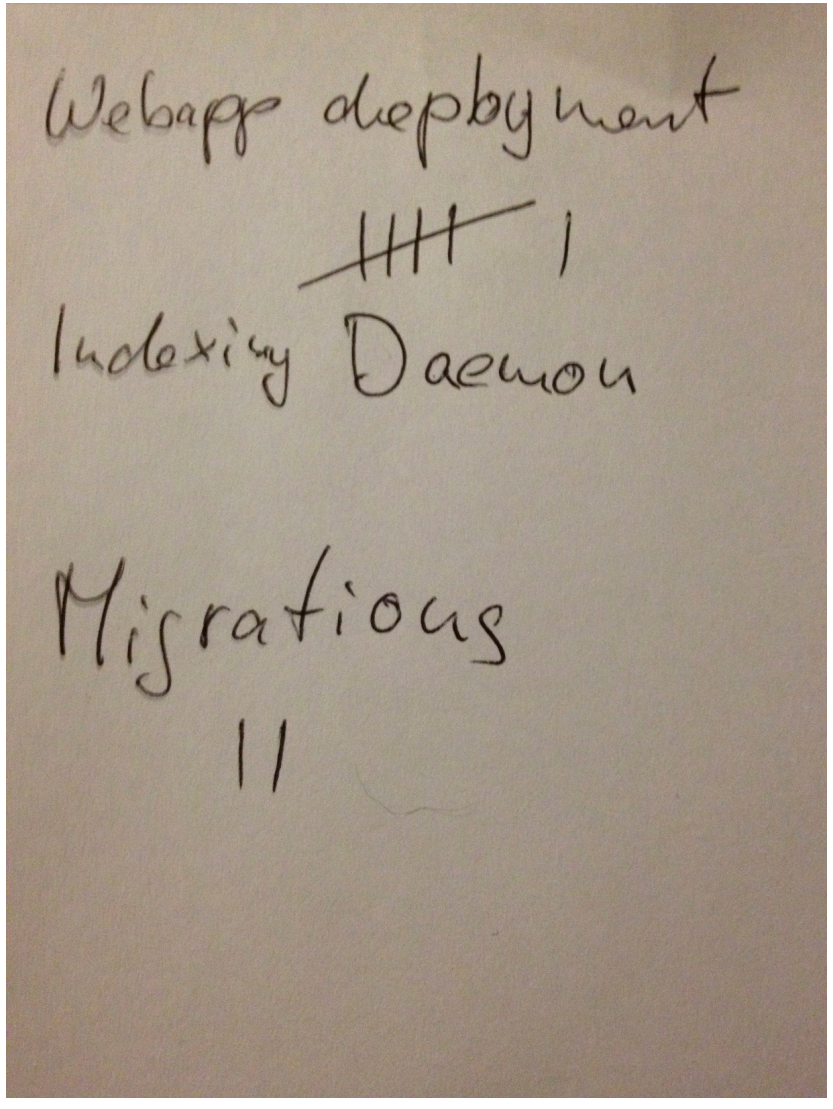
Bonus points if said developer is the companies apprentice.

The devops mindset can be incredibly empowering.

Devops-minded teams can cope with missing team members easier.

Everyone knows what everything is roughly doing anyways.

Find hacks to gather and spread that knowledge across your team!


Webapp deepbyment
IIII I
Indexing Daemon
Migrations
II

To sum up:

Teams with a strong devops culture:

can handle more complexity

Teams with a strong devops culture:

can handle more complexity

can find more alternative approaches to problems.

Teams with a strong devops culture:

can handle more complexity

can find more alternative approaches to problems.

are more likely to find solutions that handle well in production.

Their immediate answers are more complex.

Thank you for listening.

How did devops change your development style?