Scalable web services using message queues

with MsgFlo & GuvScale

Orchestrate 2017
Barcelona

Jon Nordby
@jononor

msgflo.org
@flowhub_io

whoami

Engineer

- Electronics → Software
- Embedded → Web

Creator

- Bicycle mechanic → Digital fabrication

→ Software-Defined Everything

Flowhub UG

Making software systems more understandable through data-driven programming and visual tools.

msgflo.org noflojs.org microflo.org imgflo.org

DevOps?

Everyone is able to

0) understand the system
1) debug problems
2) get changes into production
3) high confidence that change → better

Reality not there yet
- we got to improve tools & practices

This talk

1. Background (The Grid)
2. Message queue basics (RabbitMQ)
3. The MsgFlo way (flow-based-programming)
4. LIVE: Image resizing service
5. Autoscaling workers (GuvScale)
6. LIVE: Enabling GuvScale

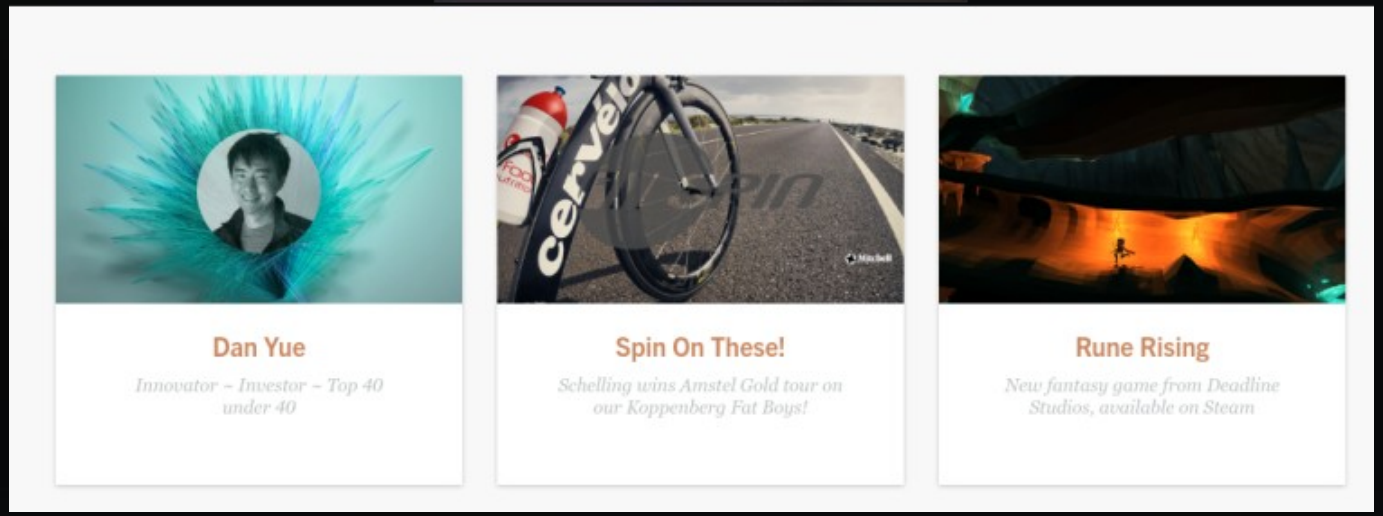
Chapter 1:
Background story

TheGrid

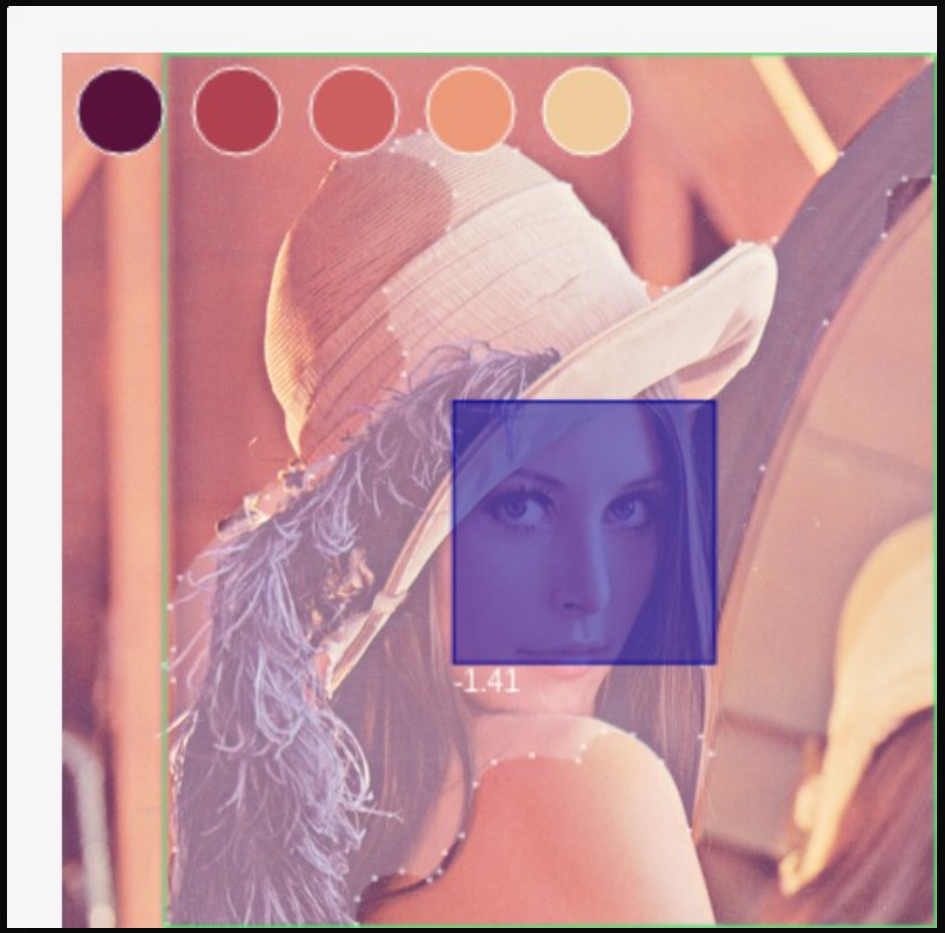
Content

Stylistic guidelines

Purpose



Content analysis



Documents

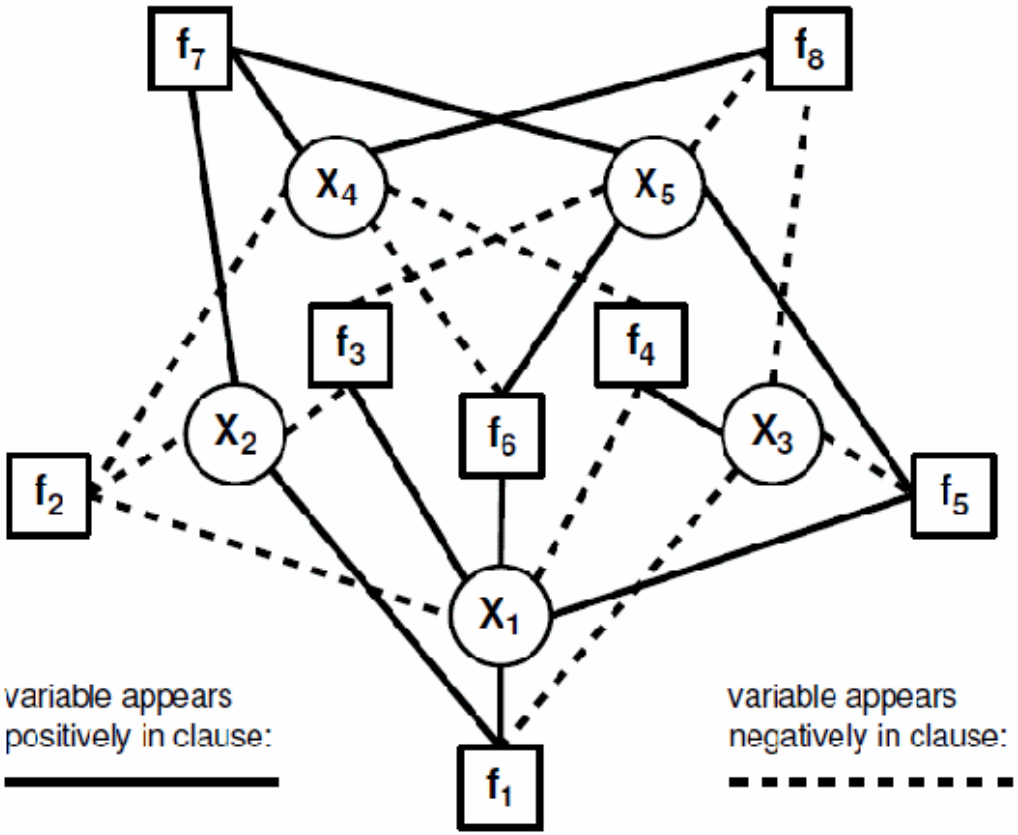
Video

Image

Text

...

Constraint solving



$$\Phi = C_1 \wedge \ldots \wedge C_8$$

$$\begin{aligned}
C_1 &= (\ x_1 \vee \ x_2 \vee \neg x_3) \\
C_2 &= (\neg x_1 \vee \neg x_2 \vee \neg x_4) \\
C_3 &= (\ x_1 \vee \neg x_2 \vee \neg x_5) \\
C_4 &= (\neg x_1 \vee \ x_3 \vee \neg x_4) \\
C_5 &= (\ x_1 \vee \neg x_3 \vee \ x_5) \\
C_6 &= (\ x_1 \vee \neg x_4 \vee \ x_5) \\
C_7 &= (\ x_2 \vee \ x_4 \vee \ x_5) \\
C_8 &= (\neg x_3 \vee \ x_4 \vee \neg x_5)
\end{aligned}$$

$$n - 5,\ m - 8,\ \alpha - 1.6$$

Image processing

https://github.com/imgflo/imgflo-server

"A good rule of thumb is to avoid web requests which run longer than 500ms".

Heroku - Worker Dynos, Background Jobs and Queueing

Examples

- CPU intensive tasks
- External & 3rd-party services

Chapter 2:
Message queues

Message queuing systems

Amazon SDS

Google PubSub

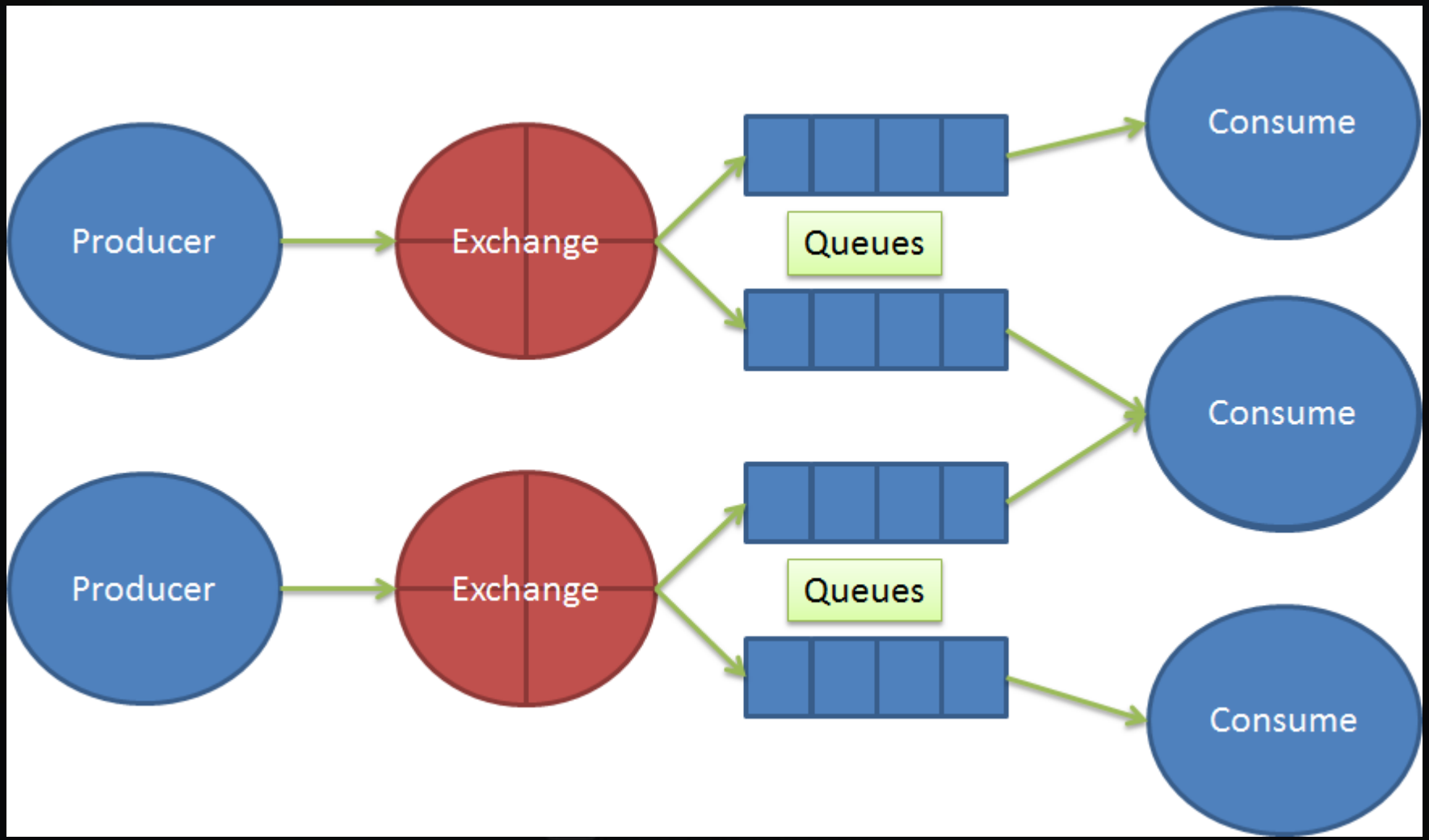
IronMQ

Mosquitto (MQTT)

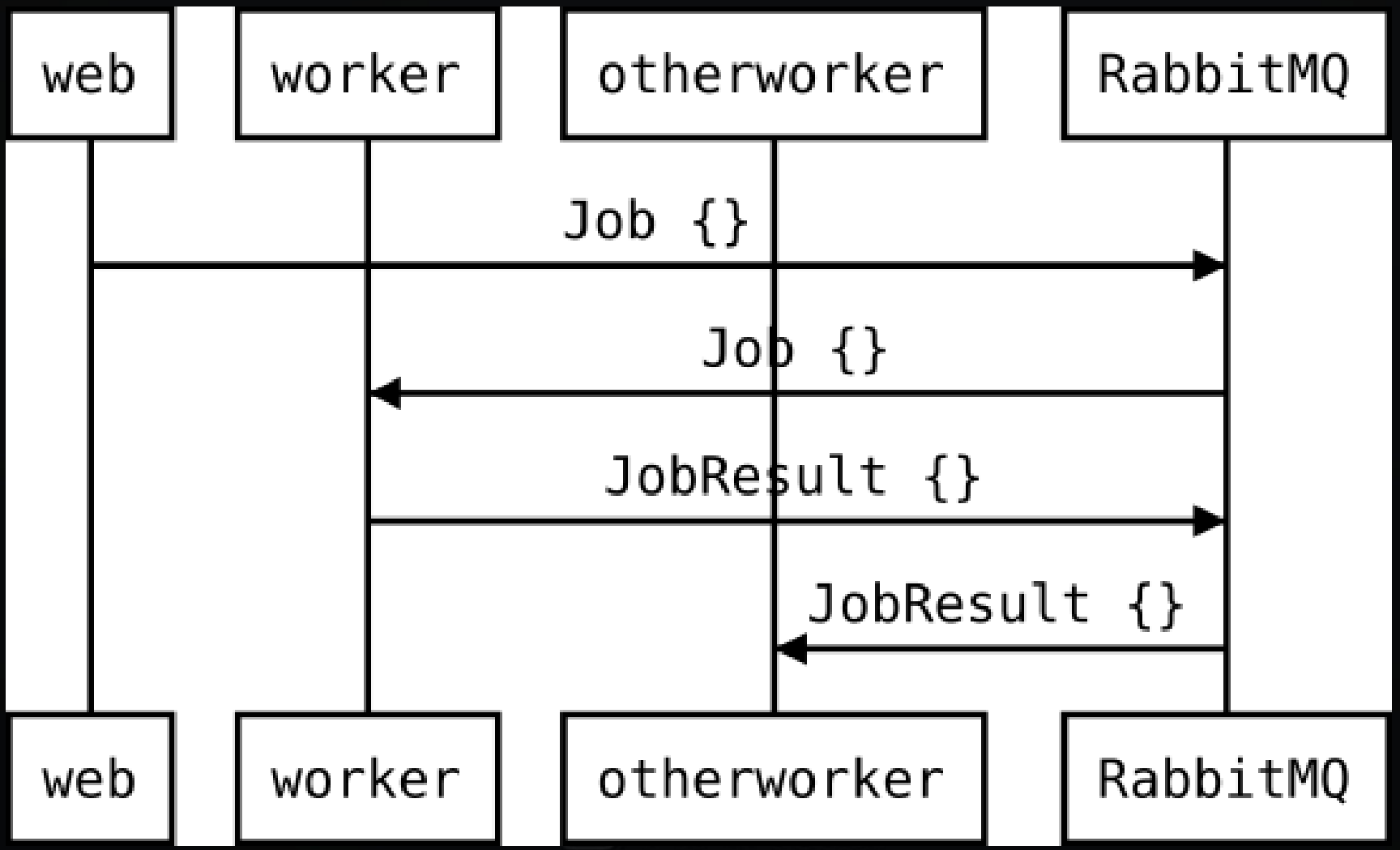
ZeroMQ

....

Message queue communication



Broker communication model



Basic AMQP communication

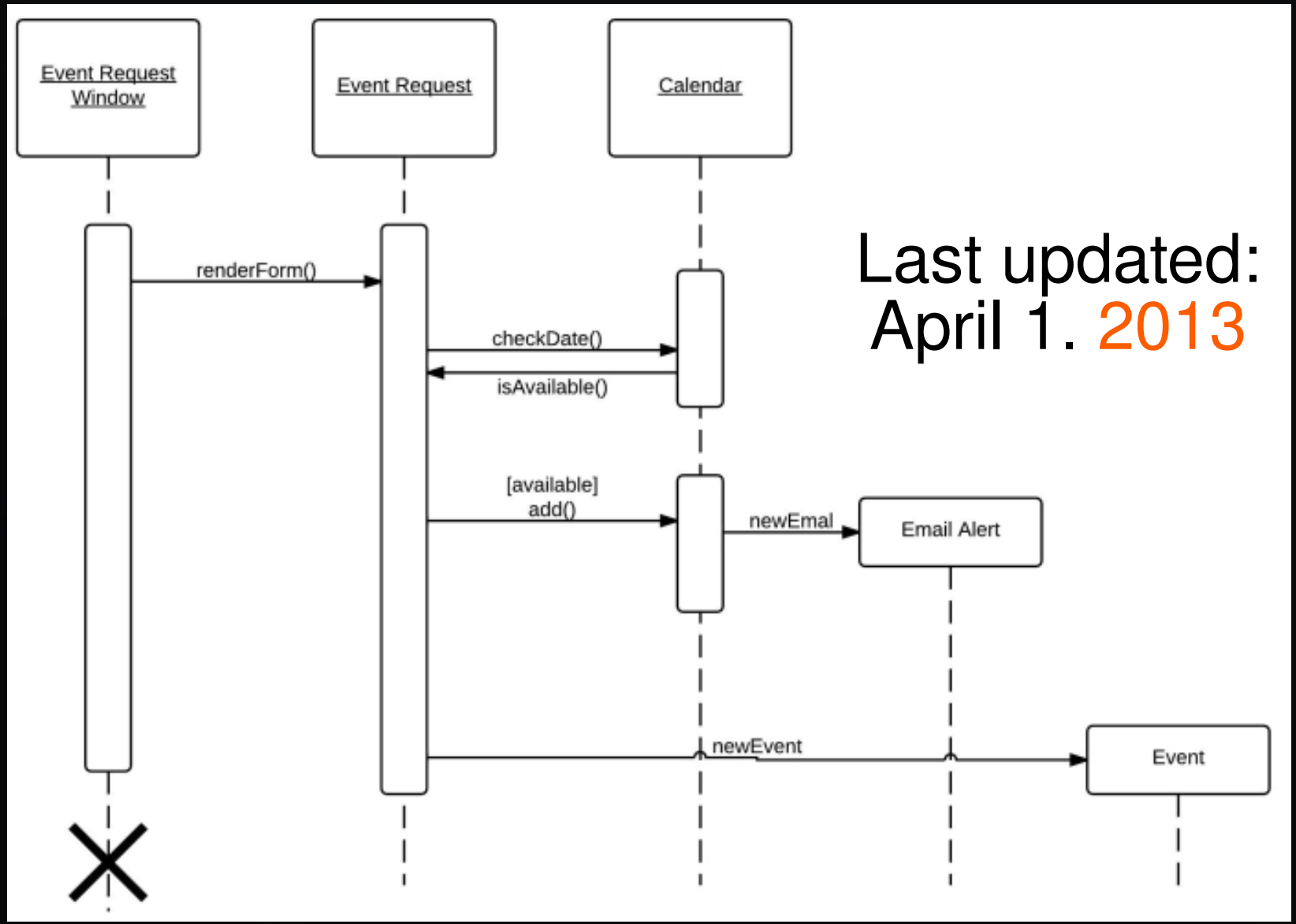
```
// worker.js
amqp.subscribe(“myjobs”, runJob);
```

```
// api.js
amqp.publish(“myjobs”, jobData);
```

+ Works OK

- Connections hidden in code
- Hardcoded functionality

Documentation...



Chapter 3:
“the MsgFlo way”

MsgFlo participant

```
// worker.js

amqp.subscribe("worker/job", runJob);
amqp.publish("msgflo/discover", workerInfo);
// function runJob
amqp.publish("worker/result", res)
```

```
// api.js

amqp.publish("msgflo/discover", apiInfo);
amqp.publish("api/newjob", job);
```

\+ Software describes itself
\+ Reusable components
! Something need to connect the topics

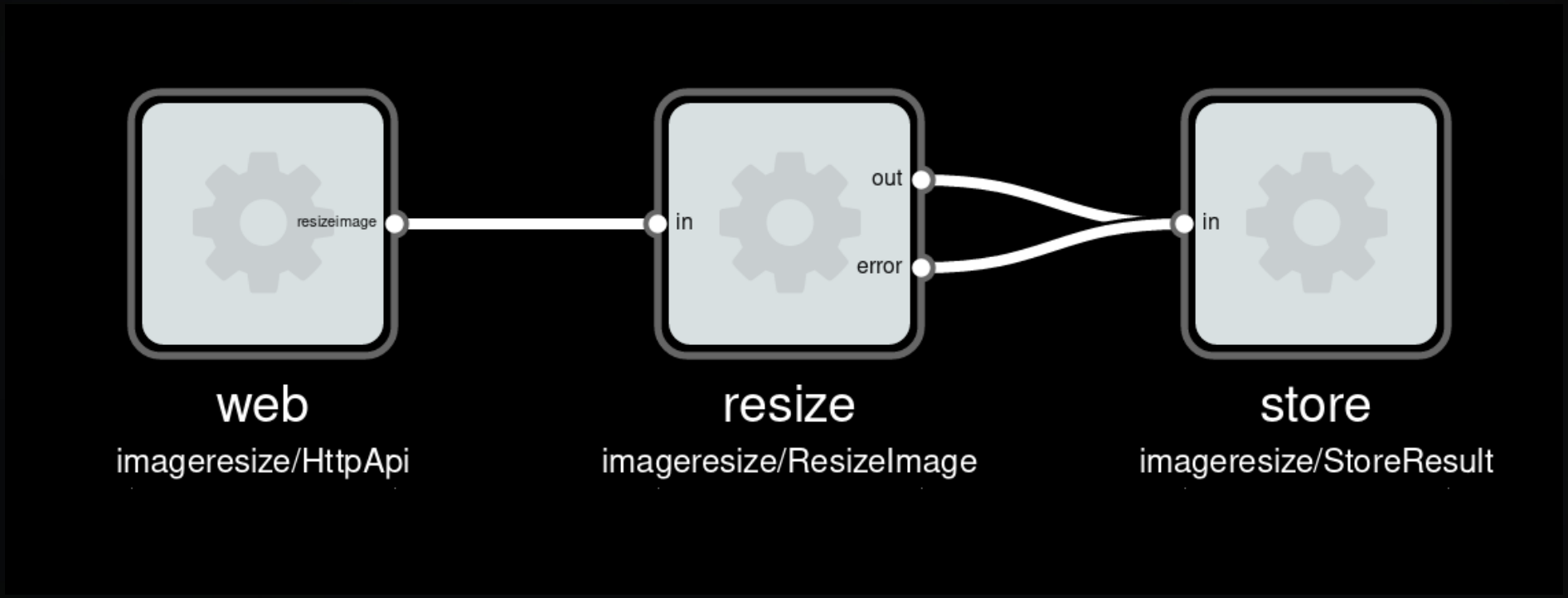
MsgFlo discovery message

```
{
  "component": "ResizeImage",
  "role": "resize",
  "id": "resize-1",
  "label": “This is a button, it can be pressed”,
  "icon": null,
  "outports": [{
      "id": "job",
      "type": "boolean"
      "queue": "resize/job"
  }],
  "inports": []
}
```

Connect workers

```
# service.fbp
api(HttpApi) NEWJOB → JOB resize(ResizeImage)
...
resize RESULT →  JOBDONE api
```

```
# bind the queues together
# configures RabbitMQ exchange bindings
$ msgflo-setup service.fbp
```


resizeimage
in
out
error
in
web
imageresize/HttpApi
resize
imageresize/ResizeImage
store
imageresize/StoreResult

Polyglot

MsgFlo support in $favoritelang ?

= Receive on AMQP queue(s)

+ Send on AMQP exchange(s)

+ MsgFlo discovery message

Convenience libraries

msgflo-nodejs: JavaScript on Node.js

noflo-runtime-msgflo: NoFlo

msgflo-cpp: C++11 on Linux

msgflo-python: Python 2.x

! msgflo-arduino: ESP8266. *MQTT-only*

msgflo-rust: Rust

Foreign participants

```
# existingthing.yml
component: c-base/siri
label: c-base siri data rescue probe
inports:
    openurl:
        queue: siri/open
        type: string
```

```
# Send MsgFlo discovery message
# on behalf of code
$ msgflo-register-foreign existingthing.yml
```

Message payload agnostic

JSON

Protobuf

XML

Binary

...

Chapter 5:
MsgFlo example
Image resizing service

Chapter 5:
Autoscaling with GuvScale

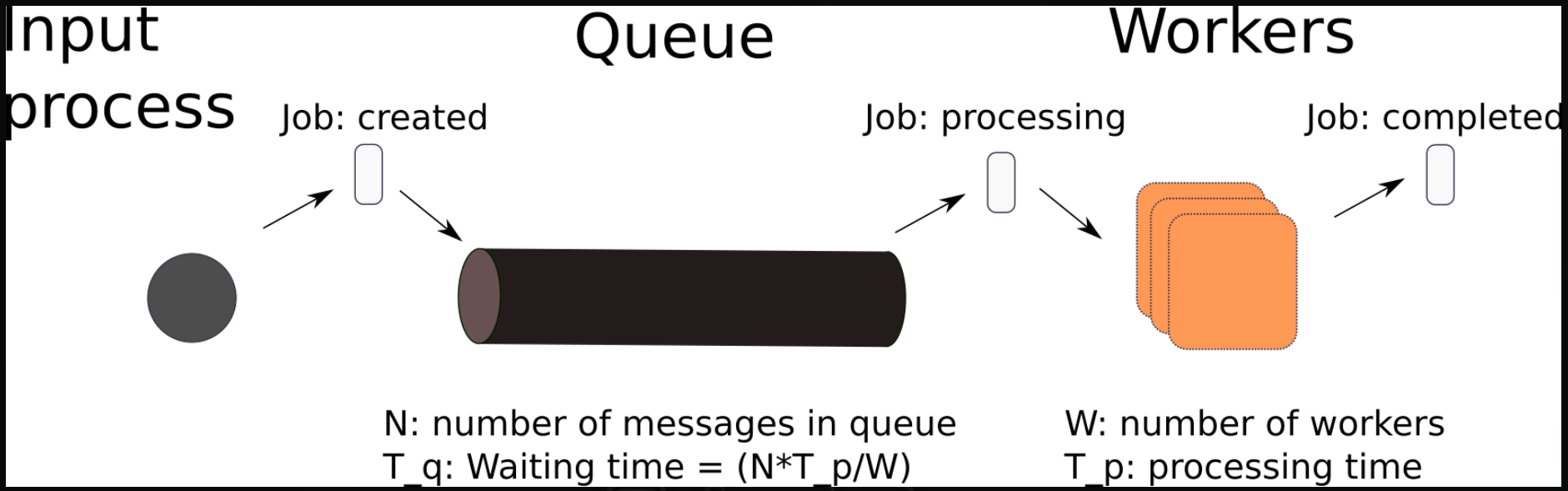
Conventional autoscaling

System metrics based
No application knowledge
Web frontend (HTTP API) focused

Typically: CPU threshold
Some: Queue-based on/off

GuvScale

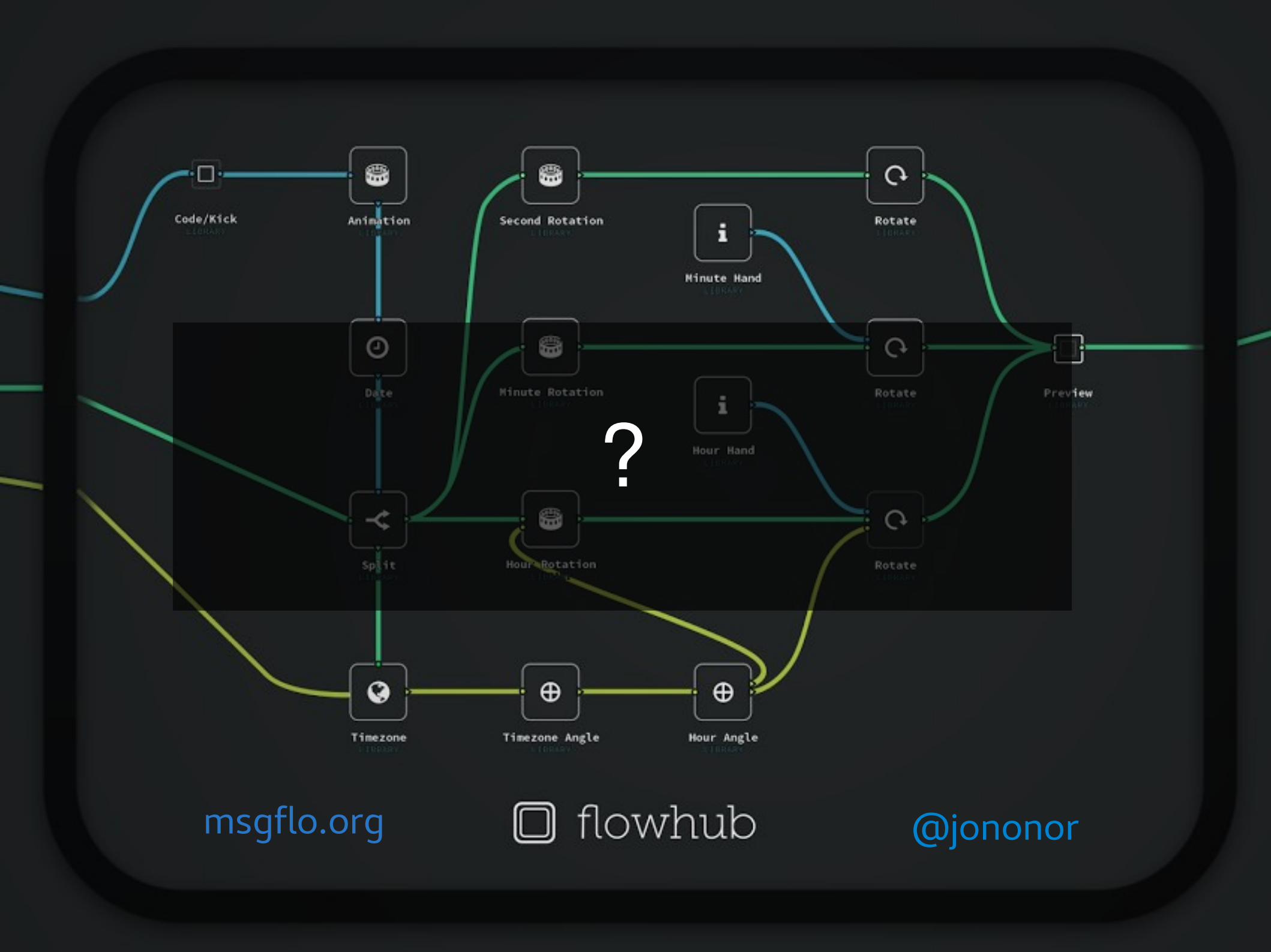
Monitor number of jobs in queue
scale workers to the number needed
to completed within specified deadline
using job processing statistics



Configuration

```
emailsender:
  queue: "send-email" # The AMQP queue name
  deadline: 180       # 3 minutes, in seconds
  minimum: 0          # Minimum number of workers
  maximum: 5          # Maximum number of workers
  concurrency: 10     # How many messages are processed concurrently
  processing: 0.300   # 300 ms, in seconds
```

Chapter 6:
Using GuvScale



msgflo.org

flowhub

@jononor

Participant modelling best practices FBP + FSM style

Component library model & tools