A world map with a dark green background, overlaid with a complex network of glowing purple and white lines representing global connections or data flow.

@ultrabug

Gentoo Linux developer
CTO at **Numberly**



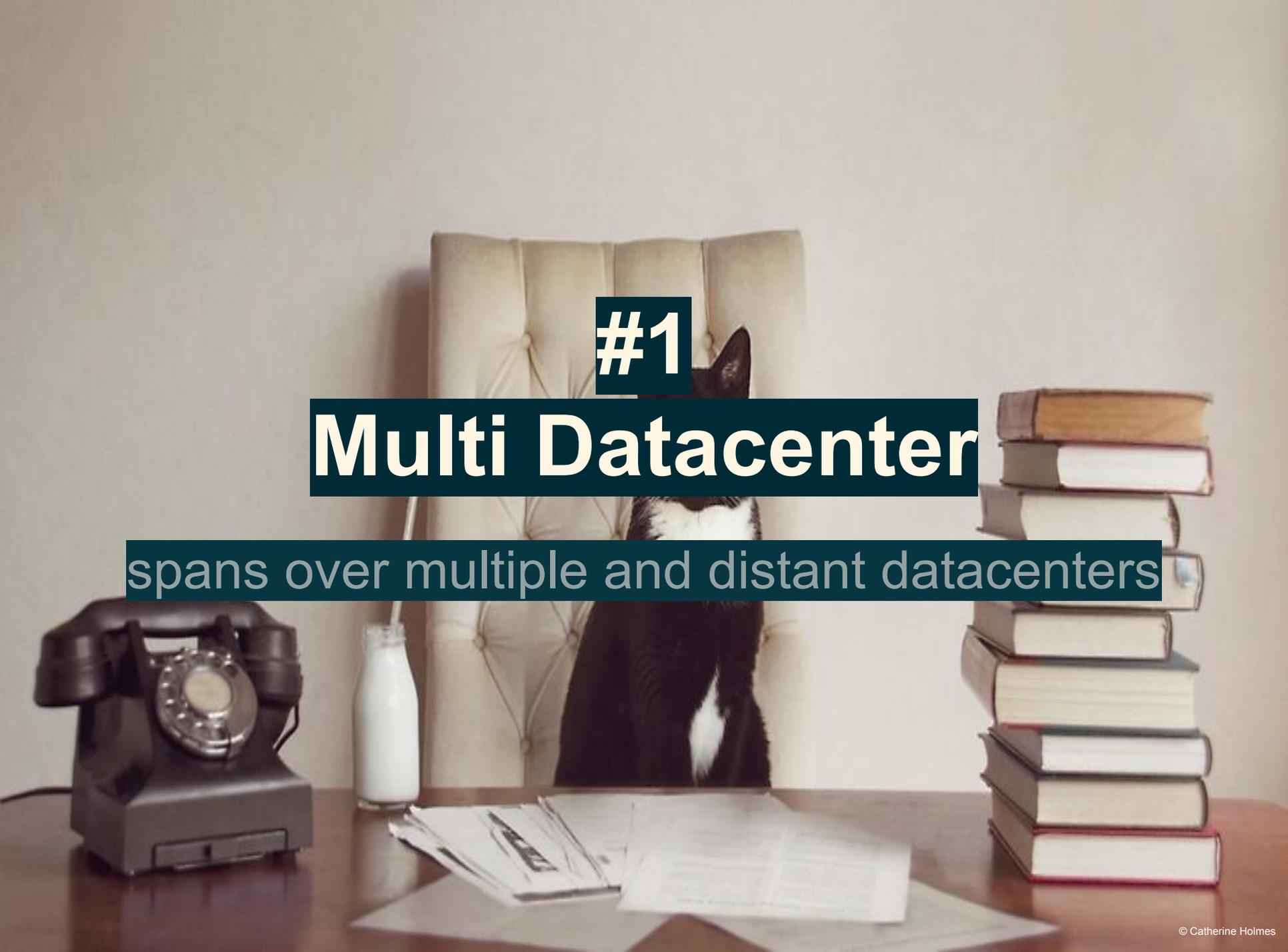
Designing a scalable and distributed application

EuroPython 2015



Geo distributed page hit counter web application

USA ↔ EUROPE

A black and white dog is sitting on a tufted, light-colored chair behind a wooden desk. On the desk, there is a vintage black rotary telephone on the left, a glass of white milk in the center, and a tall stack of several books on the right. The background is a plain, light-colored wall. The text "#1" is overlaid on the dog's head area.

#1

Multi Datacenter

spans over multiple and distant datacenters



#2

A unique page hit count

sum of the page hits from all the datacenters

A black and white dog is sitting in a tufted, light-colored chair behind a wooden desk. On the desk, there is a vintage rotary telephone on the left, a glass of milk in the center, and a stack of several books on the right. The dog's head and front paws are visible, looking towards the camera. The background is a plain, light-colored wall.

#3

Automatic resizing

no manual application (re)configuration

A black and white dog is sitting on a tufted, light-colored chair behind a wooden desk. On the desk, there is a vintage black rotary telephone on the left, a glass of white milk in the center, and a stack of several books on the right. The background is a plain, light-colored wall.

#4

Distributed configuration

immediately available to all web services



Hell...

...that's some kind of contract !

A ginger tabby cat is lying on its back on a black, textured surface, possibly a bedsheet. The cat's front paws are raised and its hind legs are also raised, with its tail curled. The cat's eyes are closed, and it appears to be resting or sleeping. Overlaid on the image are two text boxes: a large white one with a dark background and a smaller grey one with a dark background.

Rely on your stack

be lazy !

```
>>> import this
```

Zen of python

if the *technology* is hard to explain,
it's a bad idea

```
SLMIN .SYS      12P 20-Dec-85    VN .SYS        3P 13-Aug-86
XL .SYS         4P 20-Dec-85    LD .SYS        8P 23-Aug-86
SP .SYS         6P 13-Aug-88    DL .SYS        5P 13-Aug-86
RT11SJ.SYS     78P 13-Aug-88    DU .SYS        8P 13-Aug-86
NL .SYS         2P 13-Aug-86    TT .SYS        2P 13-Aug-86
SD .SYS         5P 31-May-85    RL02DC.SYS    71P 21-Nov-84
BASIC .SAV     56 24-May-79    BINCOM.SAV    24 20-Dec-85
DATE .SAV      4 20-Dec-85    DIR .SAV      19 20-Dec-85
DUMP .SAV      9 20-Dec-85    DUP .SAV      47 20-Dec-85
TSXMD .SAV     78 27-Nov-82    FORTRA.SAV    206 21-May-85
HARRIS.SAV     41 12-Jun-85    LET .SAV      5 20-Dec-85
START .P3G     2 21-Dec-91    RETRO.OBJ     1519P 16-May-88
EN1A .STH     15P 02-Feb-93    TSMUCL.TSX    22 07-Mar-92
STAND .LIN     12P 15-Aug-83    TSXV6.HSC     1P 04-Sep-95
EN1B .STH     19 11-Feb-93    JKFLIP.SAV    30 08-Mar-96
JKFLIP.FOR     3 08-Mar-96    RT11FB.SYS    93P 20-Dec-85
ANMOT .SAV     38 18-Apr-93    TSMF23.NEM    1200P 27-Nov-92
DUO .DIR       18 16-Jul-96    DL .DIR       7 16-Jul-96
DIR .DIF       26 16-Jul-96    DEMOFG.OBJ    1 -BAD-
DEMOBC.OBJ     1 -BAD-
EVAN .ID       1 1
5949 Blocks
```

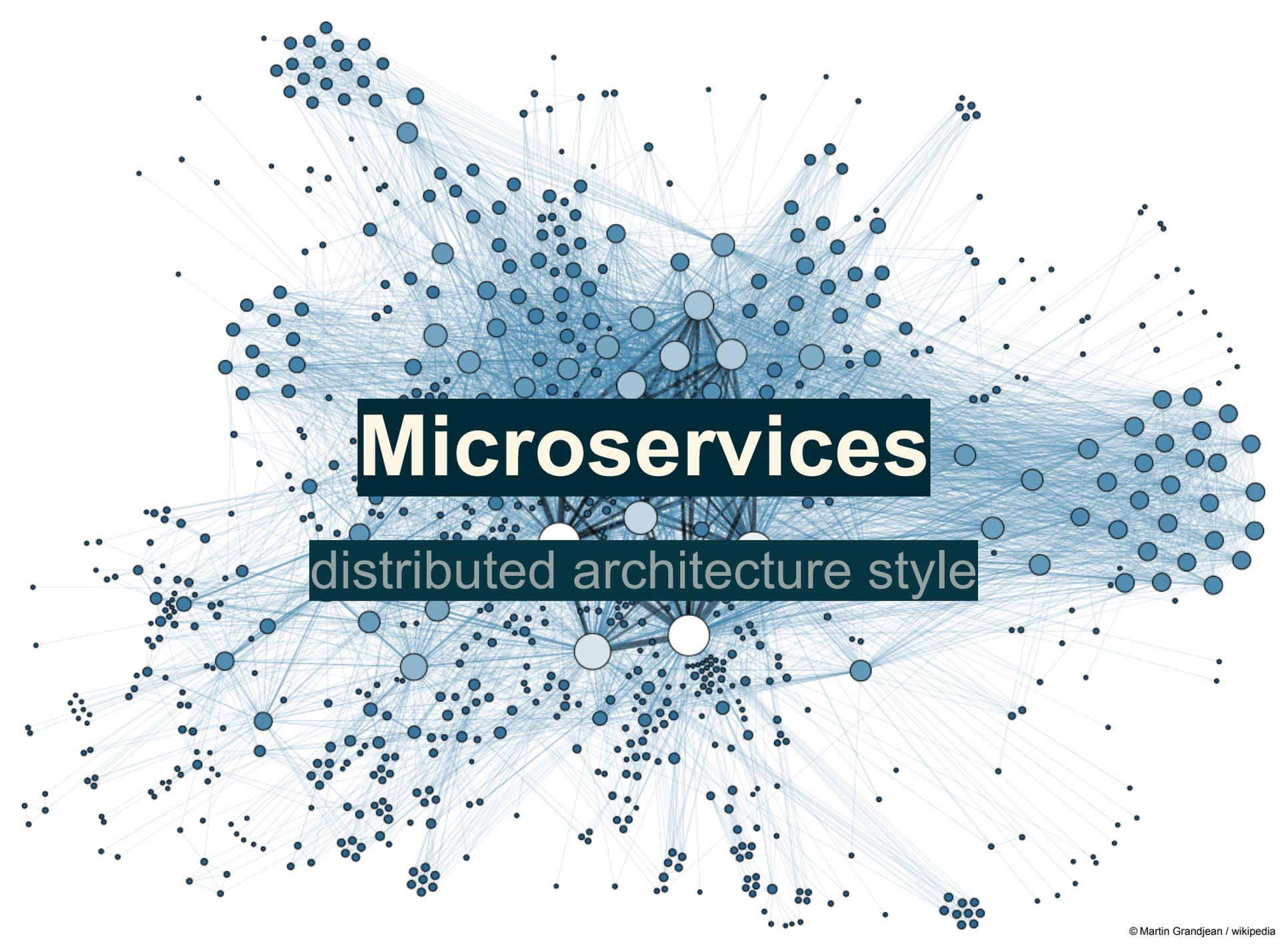
UNIX tools philosophy

keep it small and simple



Isolated components

breaking it down to pieces



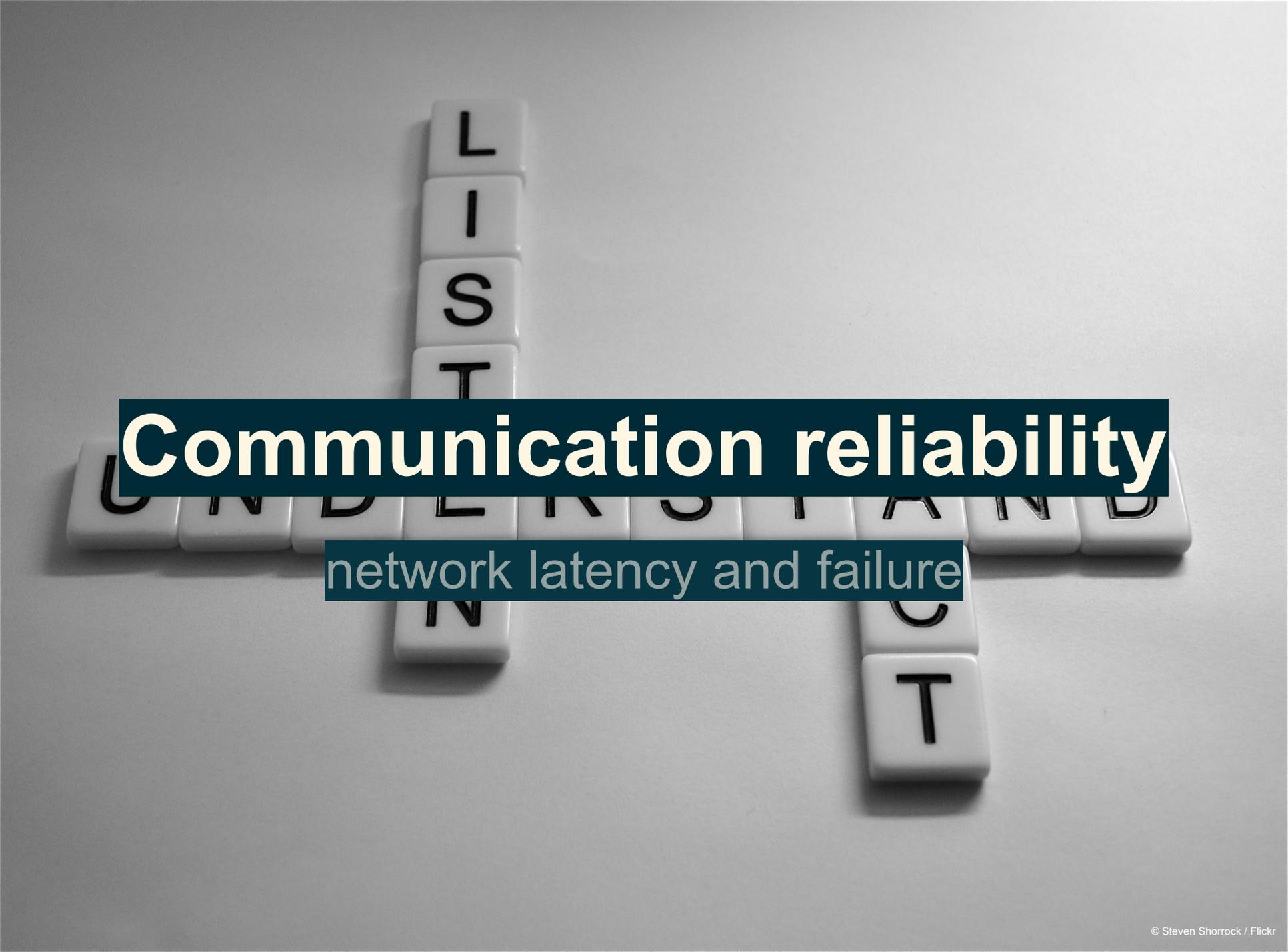
Microservices

distributed architecture style



Microservices

micromanagement = high operational cost

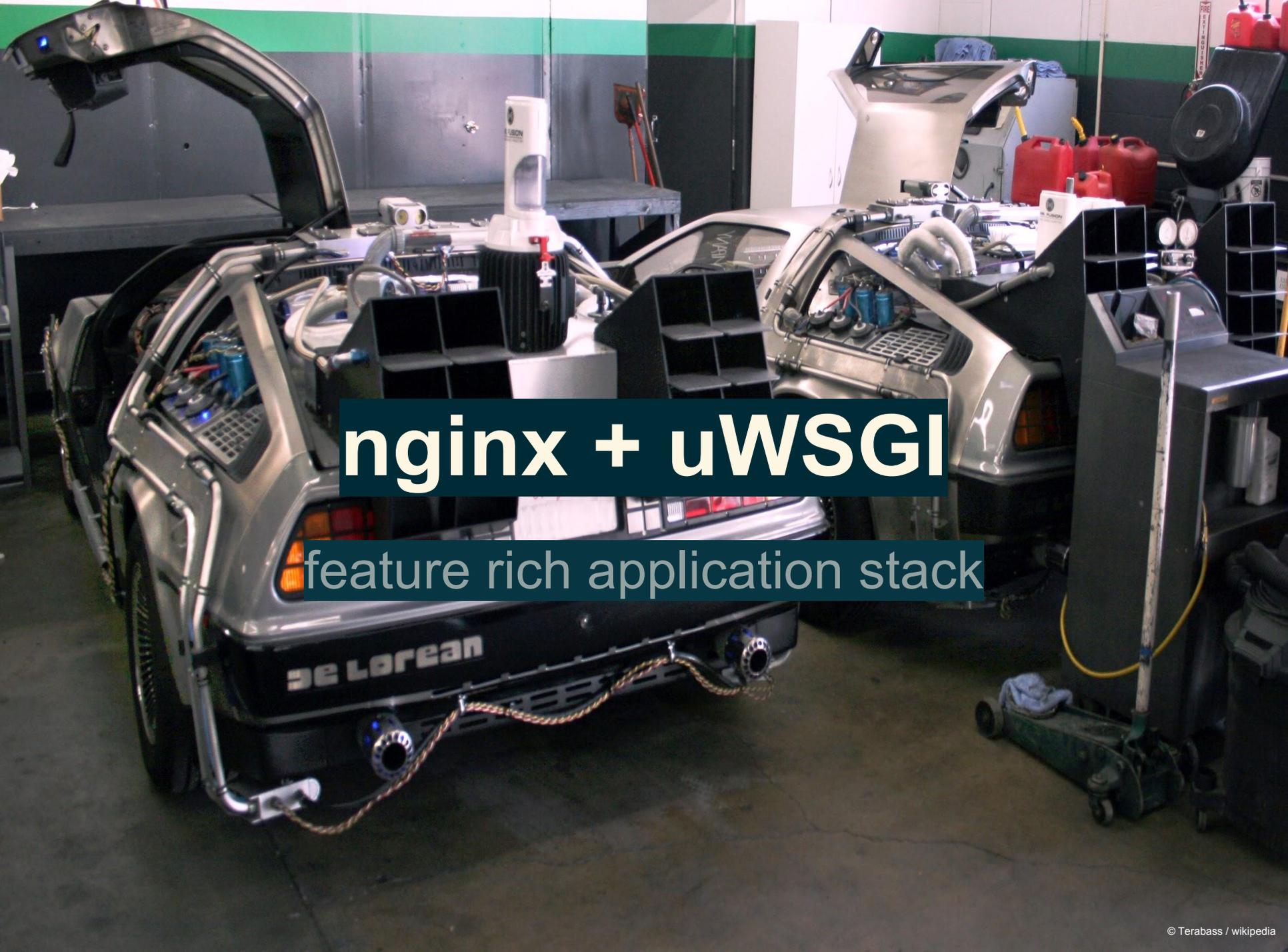


Communication reliability

network latency and failure

Locality vs consistency

data access lag or outage



nginx + uWSGI

feature rich application stack

```
1 [uwsgi]
2 enable-threads = true
3 lazy-apps = true
4 master = true
5 plugins = python27,gevent27
6 socket = /run/uwsgi/%n.socket
7 uid = nginx
8
9 # process
10 processes = 2
11 gevent = 1000
12 gevent-monkey-patch = true
13
14 # reload
15 reload-mercy = 10
16 worker-reload-mercy = 1
17
18 # automatic reload
19 touch-reload = /var/www/ep2015_collector/http_worker.py
20
21 # logging
22 log-syslog = uwsgi.%n
23 disable-logging = true
24 log-slow = 500
25
26 # http worker
27 buffer-size = 32768
28 max-requests = 10000
29 listen = 5000
30 #
31 chdir = /var/www/ep2015_collector/
32 file = http_worker.py
33 callable = ep2015
34 subscribe-to = 127.0.0.1:3615:ep2015.ultrabug.org
```

collector

⇒ gets HTTP requests ⇒ produces jobs

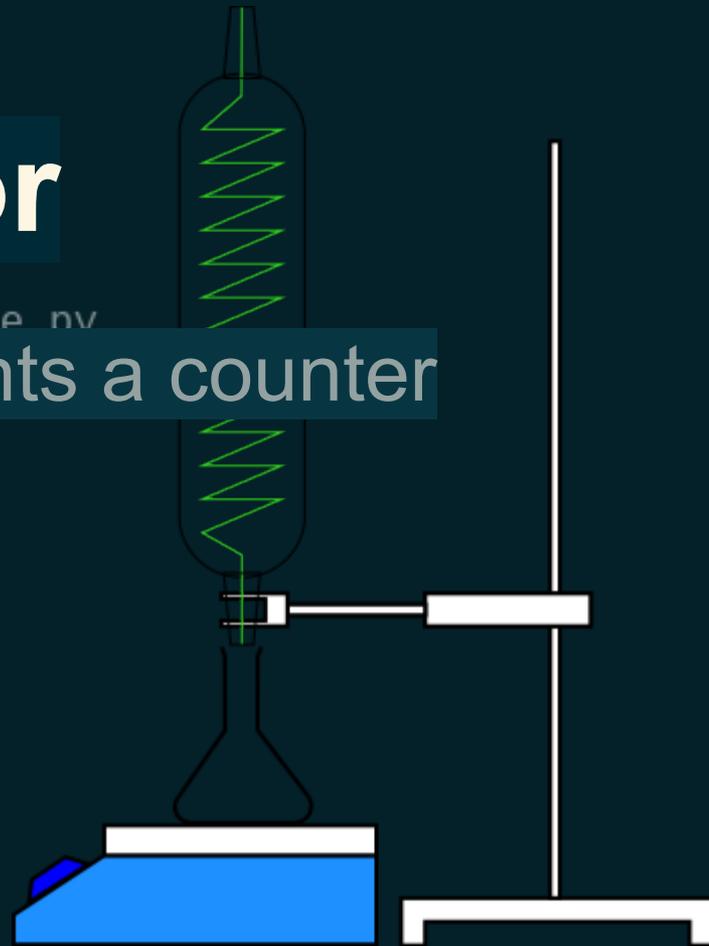
⇐ returns the total hit count

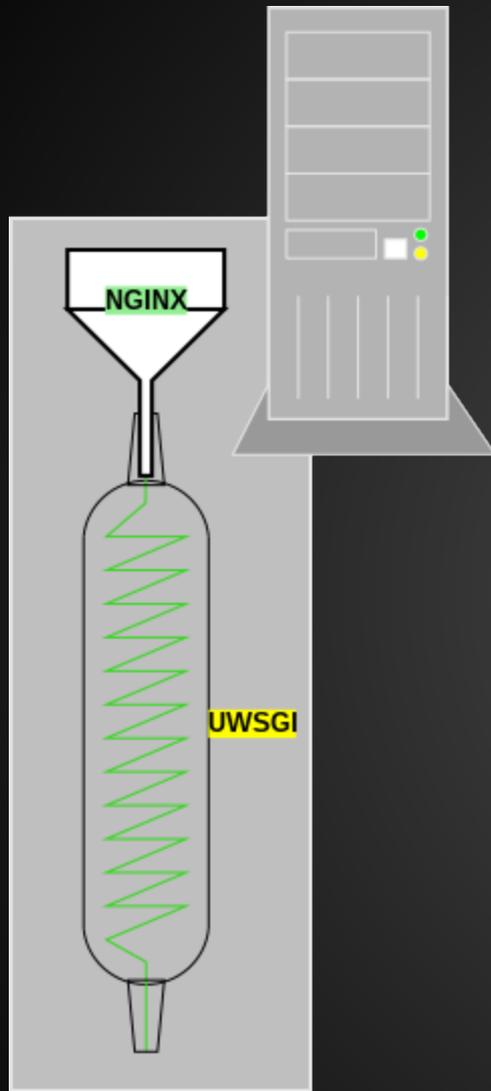
A diagram of a collector is shown on the right side of the image. It consists of a white funnel at the top, connected to a vertical tube. Inside the tube is a green zigzag line representing a spring. The tube ends in a small opening at the bottom. The word 'collector' is written in large white letters over a dark blue background, positioned to the left of the diagram.

```
1 [uwsgi]
2 enable-threads = true
3 master = true
4 plugins = python27
5 socket = /run/uwsgi/%n.socket
6
7 # process (one per default, duplicate for more)
8 mule = /usr/lib/ep2015_processor/mule.py
9
10 # reload
11 reload-mercy = 10
12 mule-reload-mercy = 1
13 py-call-osafterfork =
14
15 # automatic reload
16 touch-reload = /usr/lib/ep2015_processor/mule.py
17
18 # logging
19 log-syslog = uwsgi.%n
20 disable-logging = true
21
```

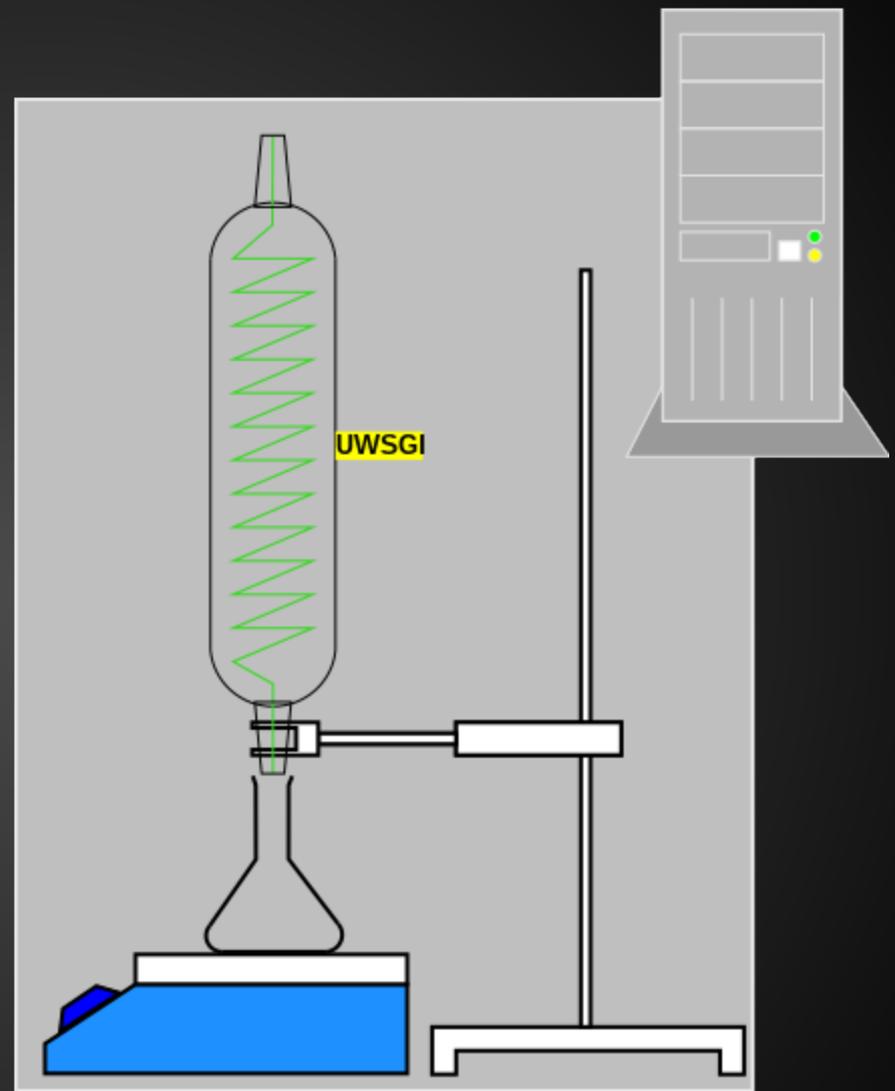
processor

consumes jobs ⇨ increments a counter





COLLECTOR



PROCESSOR



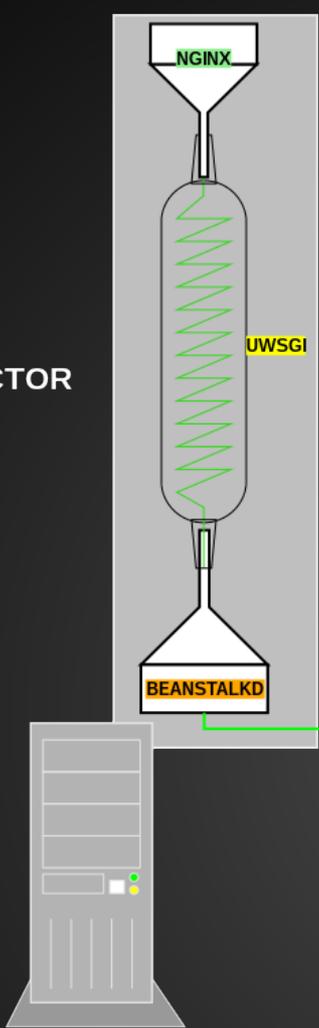
beanstalkd

simple and reliable job queue server

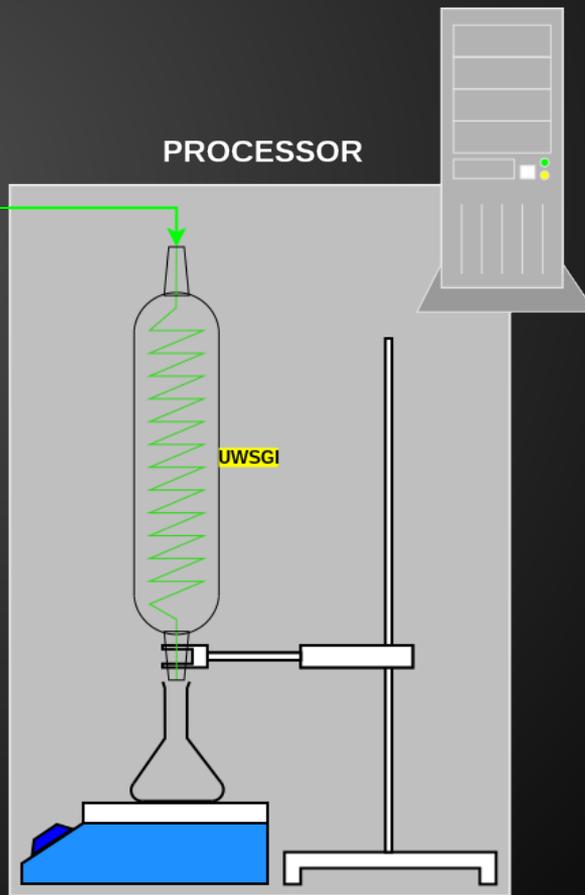
```
1 [uwsgi]
2 master = true
3 socket = /run/uwsgi/%n.socket
4
5 # daemon
6 attach-daemon2 = pidfile=/run/beanstalkd.pid,cmd=beanstalkd -b /
  var/lib/beanstalkd -p 11300 -l 0.0.0.0 -z 65536 -u beanstalk,
  daemonize=1
```

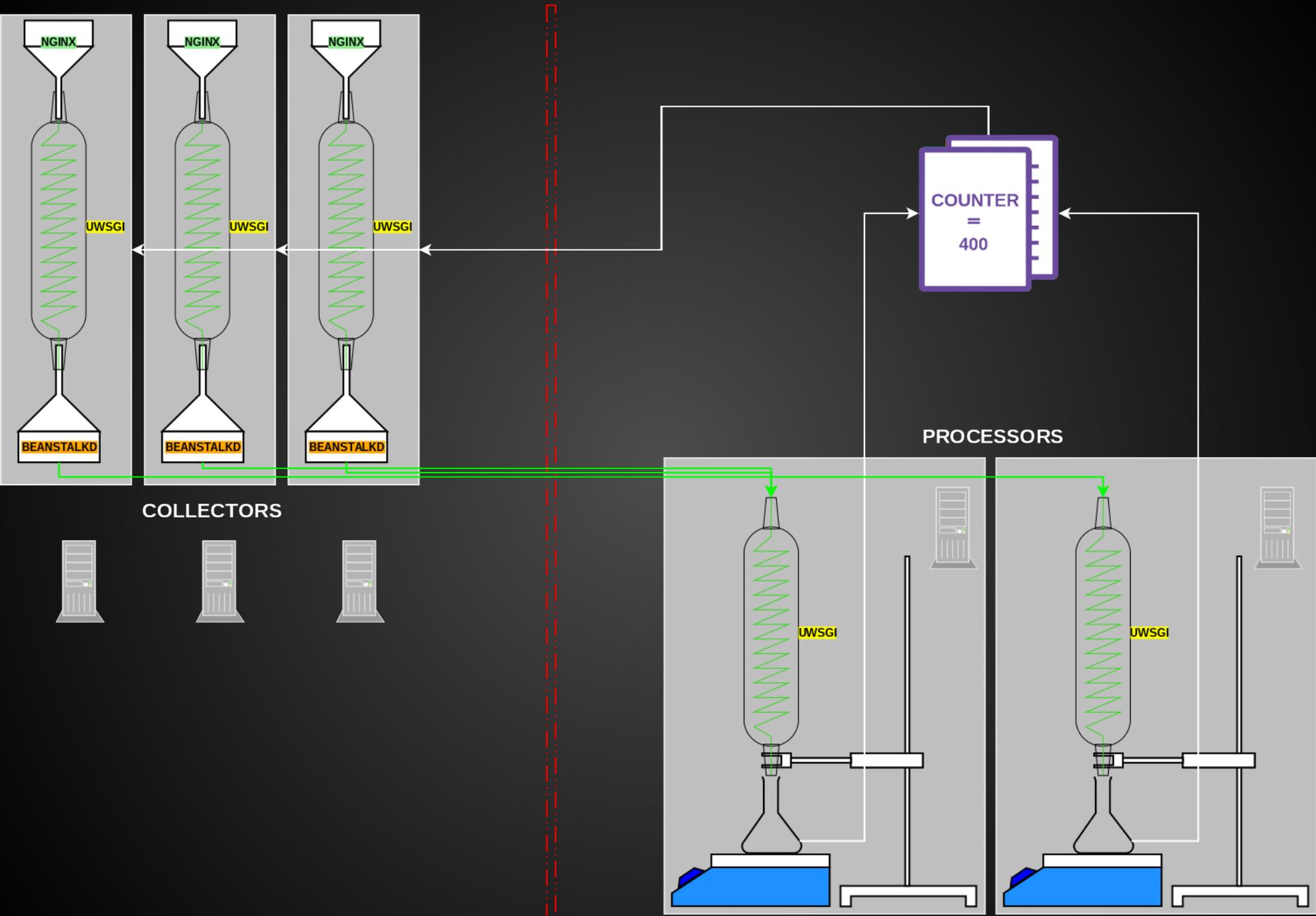
```
7
```

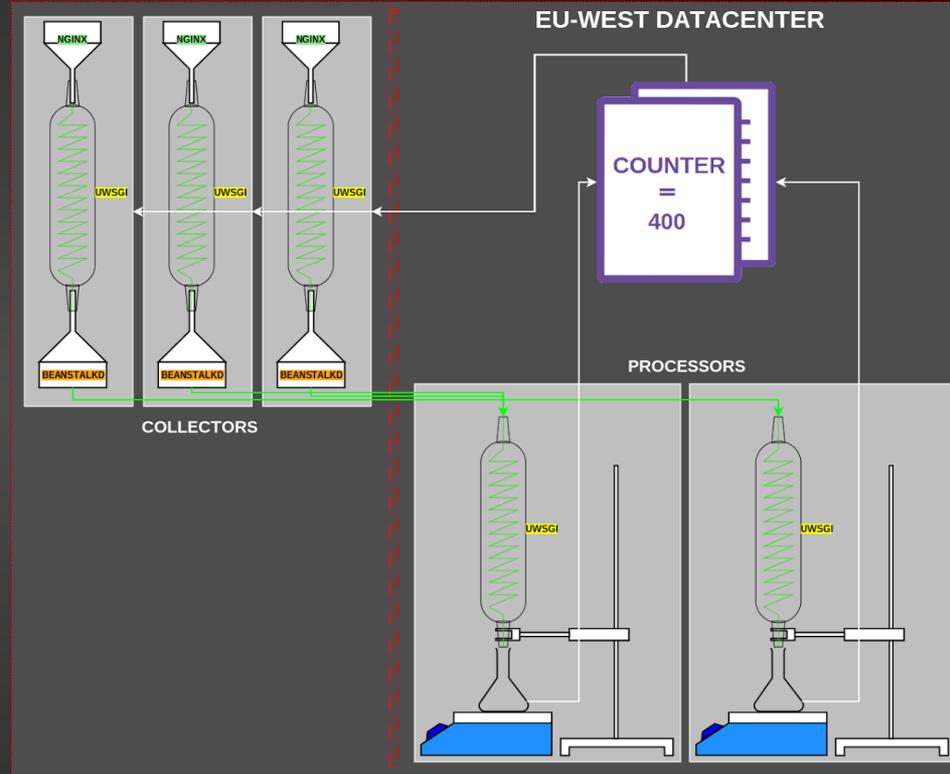
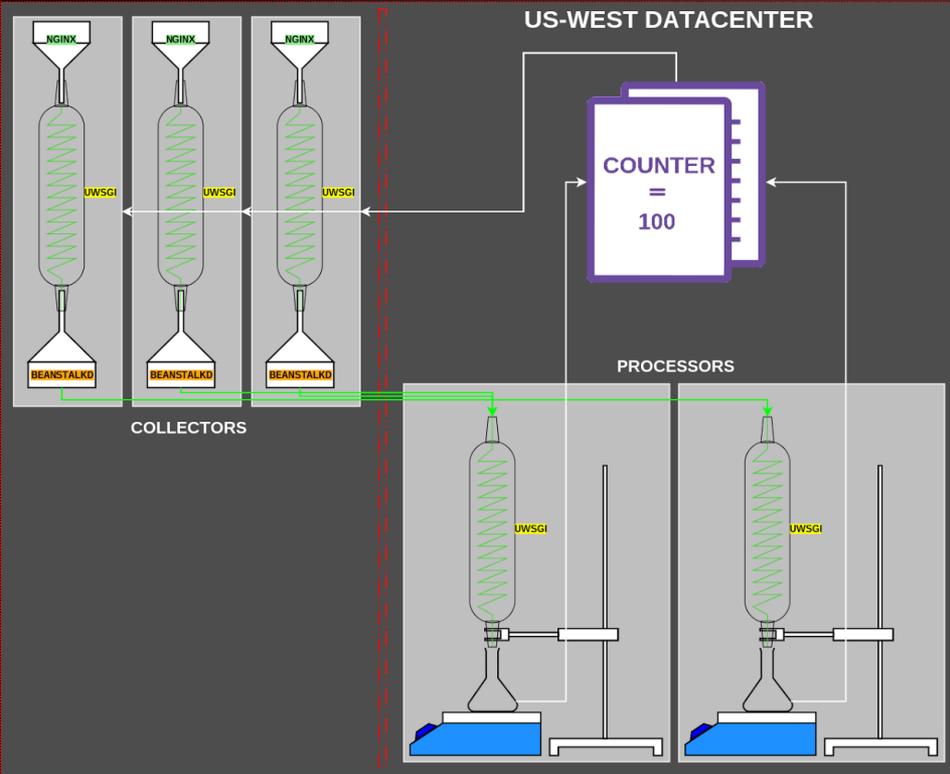
COLLECTOR



PROCESSOR



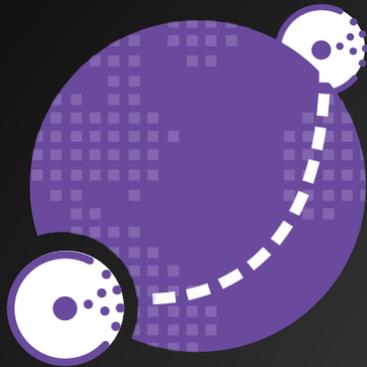






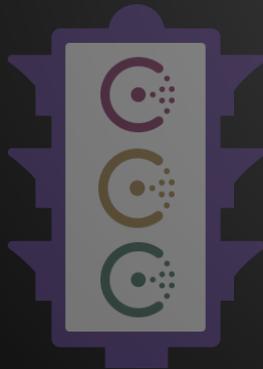
Service discovery

automated scaling and fault tolerance

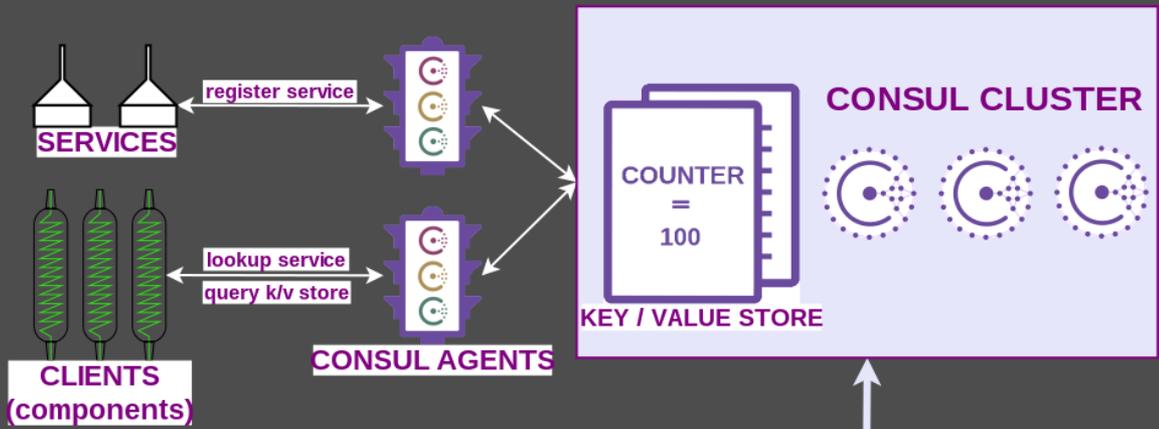


Consul

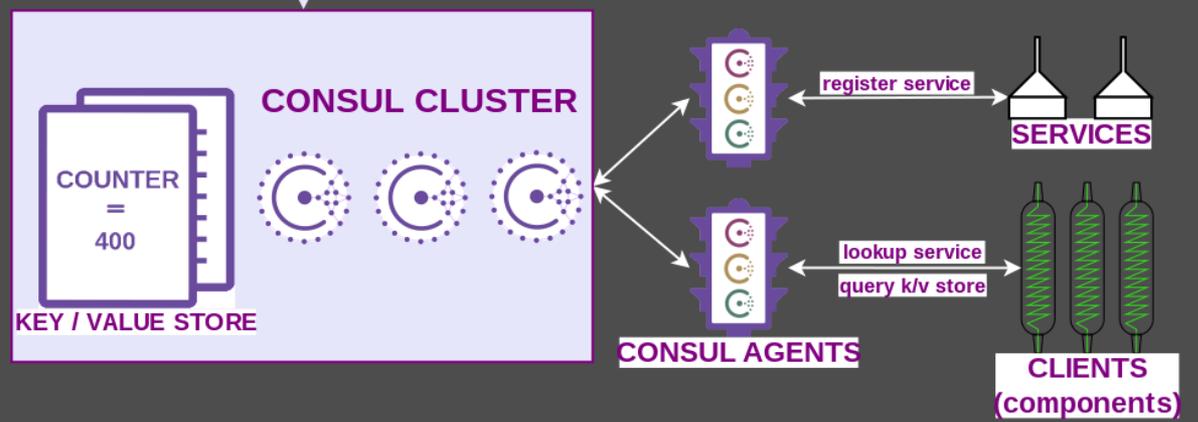
multi datacenter + key / value storage

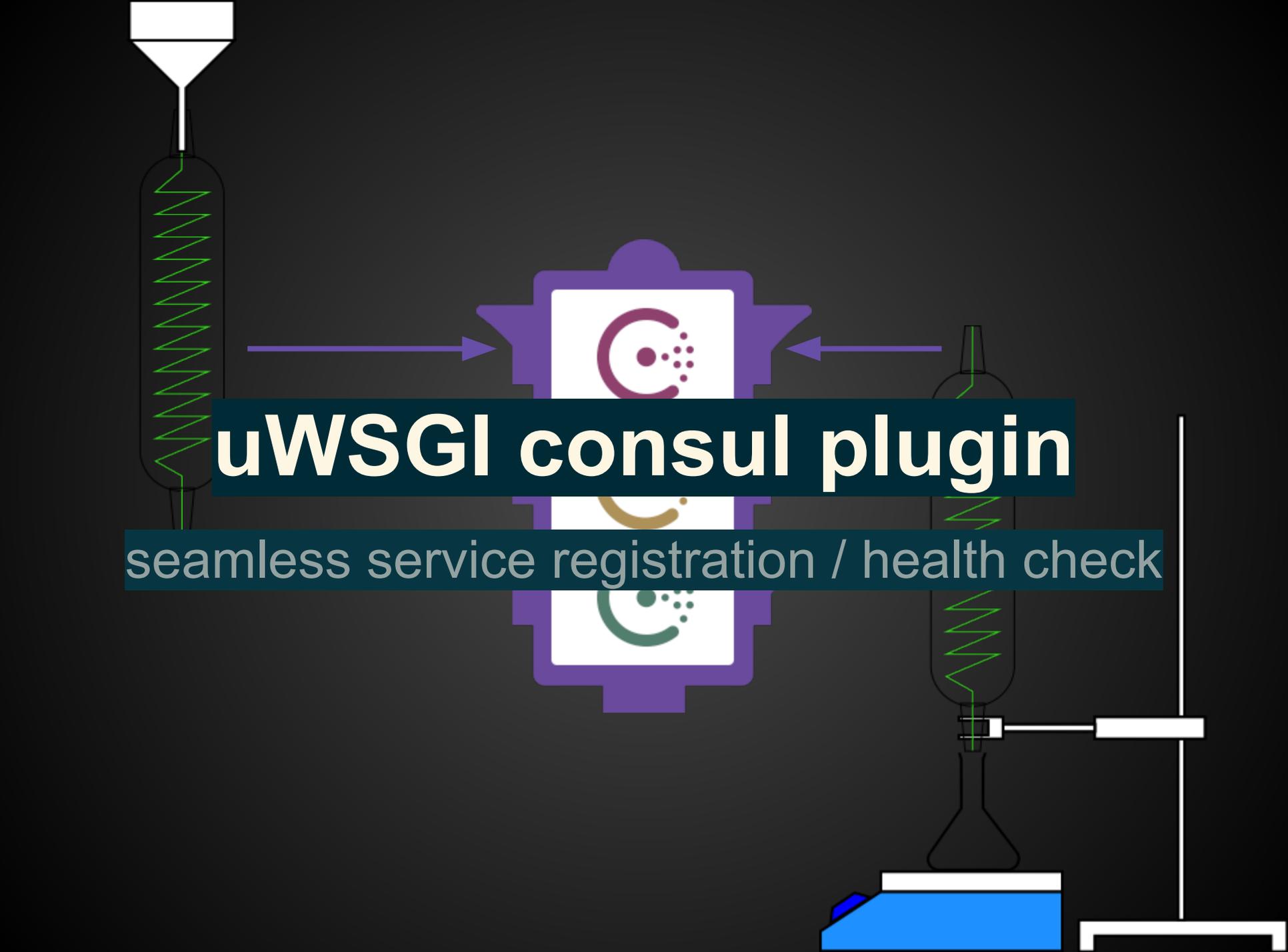


US-WEST DATACENTER



EU-WEST DATACENTER





uWSGI consul plugin

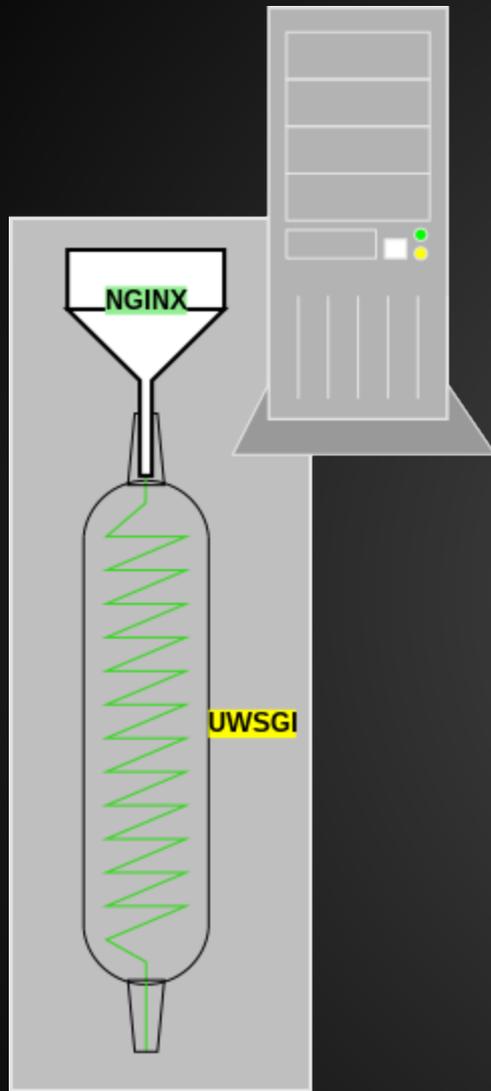
seamless service registration / health check

```
consul-register = url=http://localhost:8500,name=my_service
```

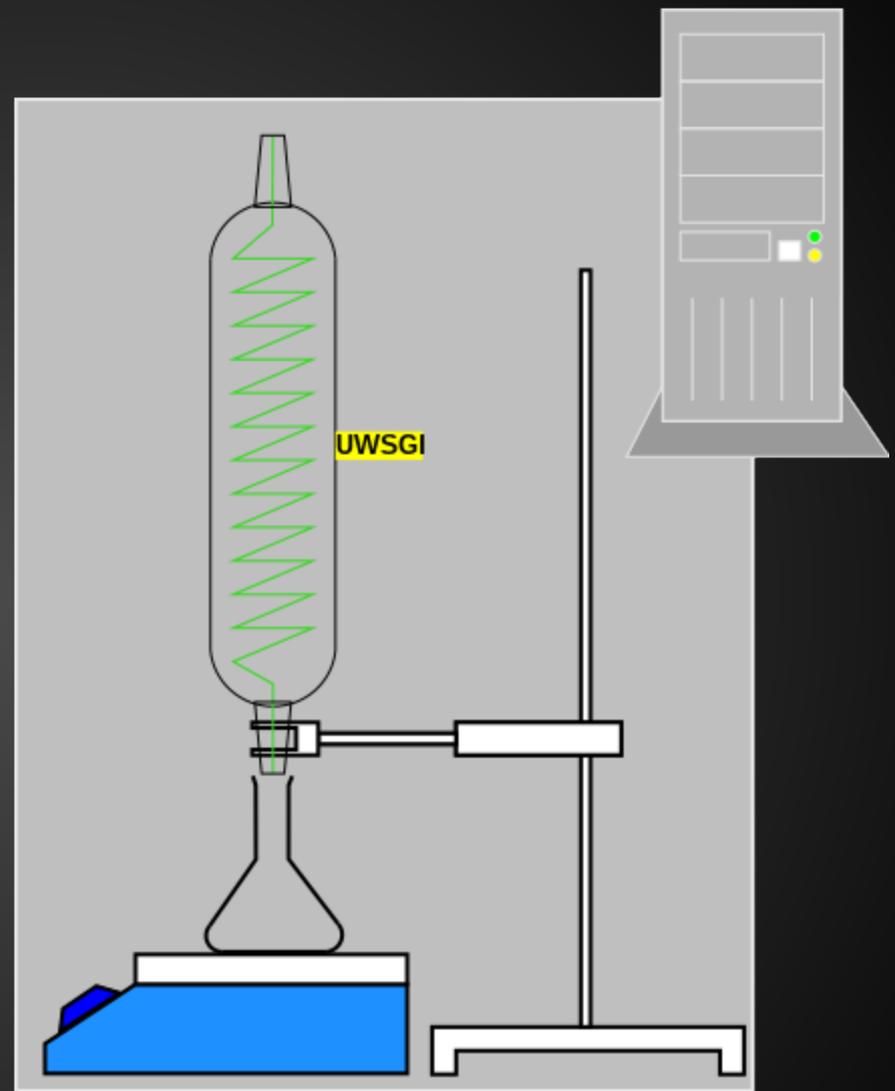


Let's build this up

to victory

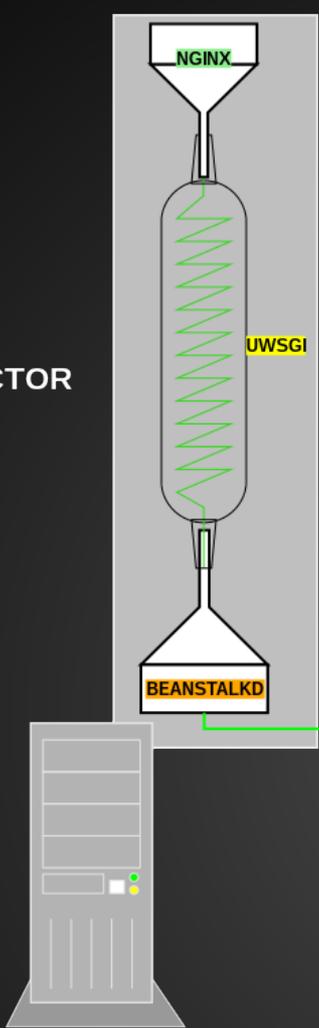


COLLECTOR

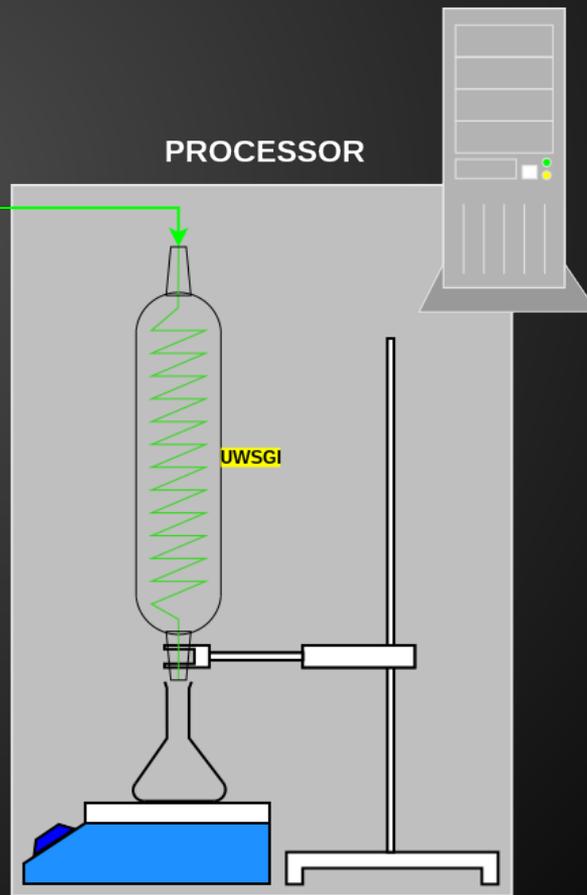


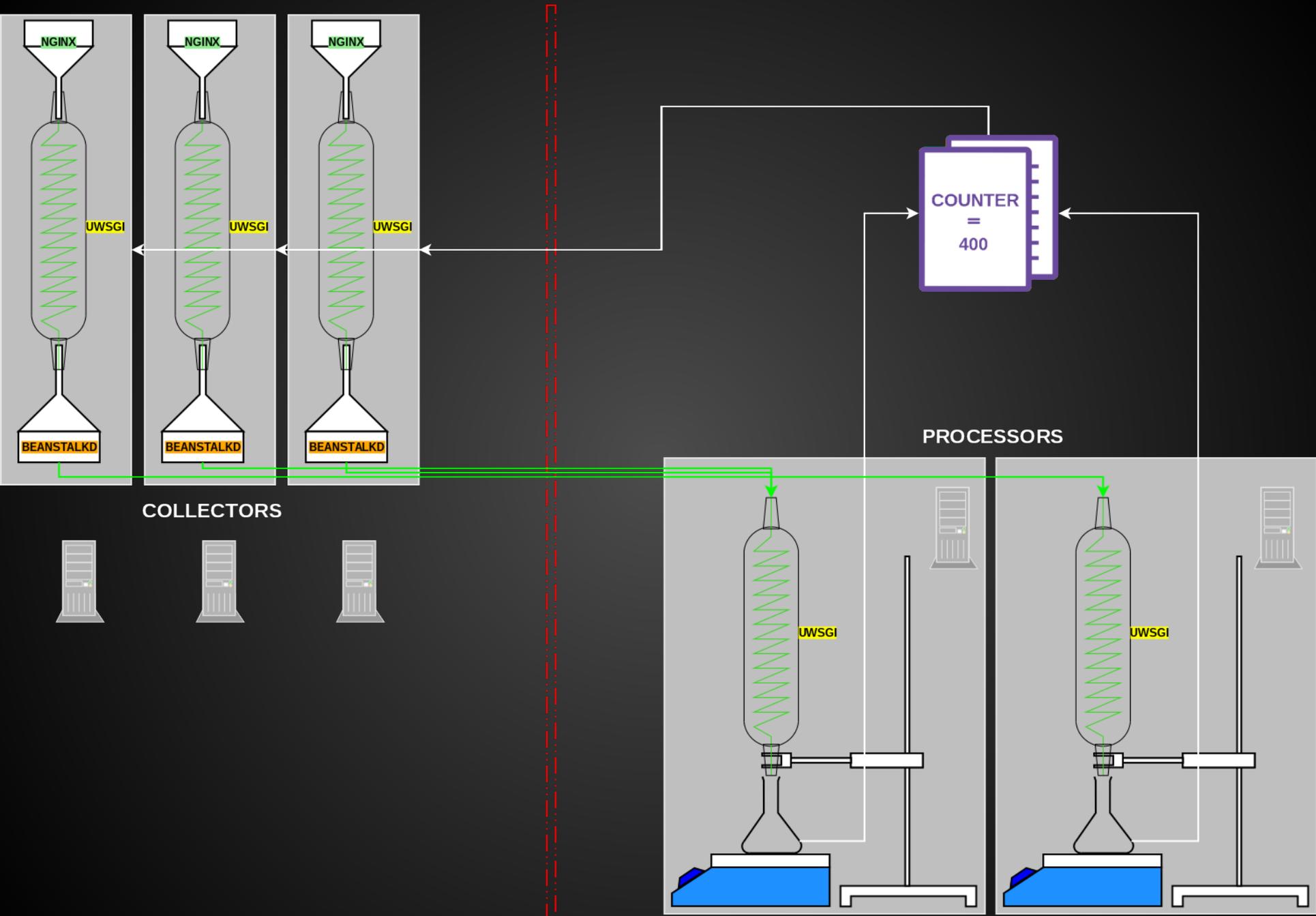
PROCESSOR

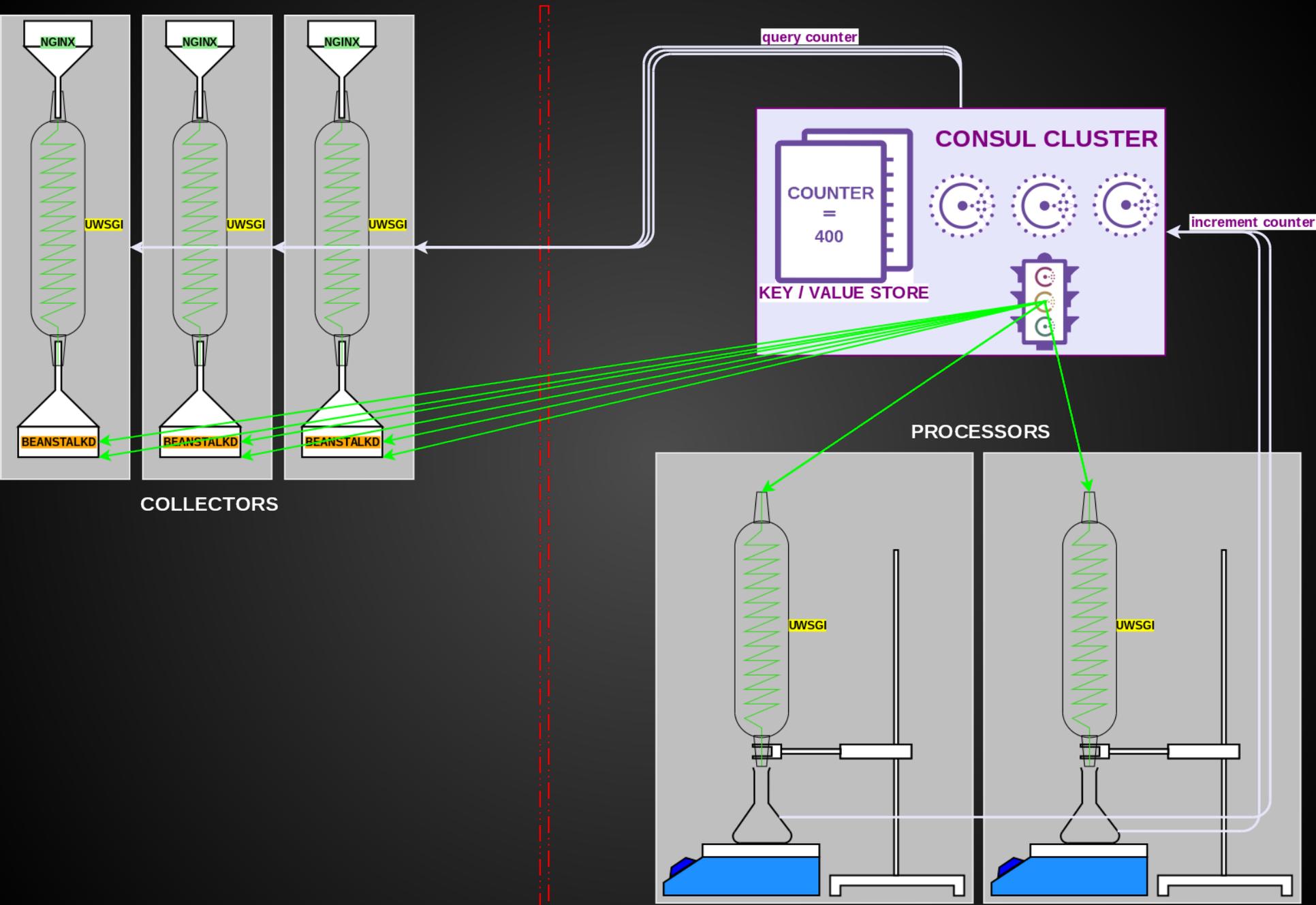
COLLECTOR

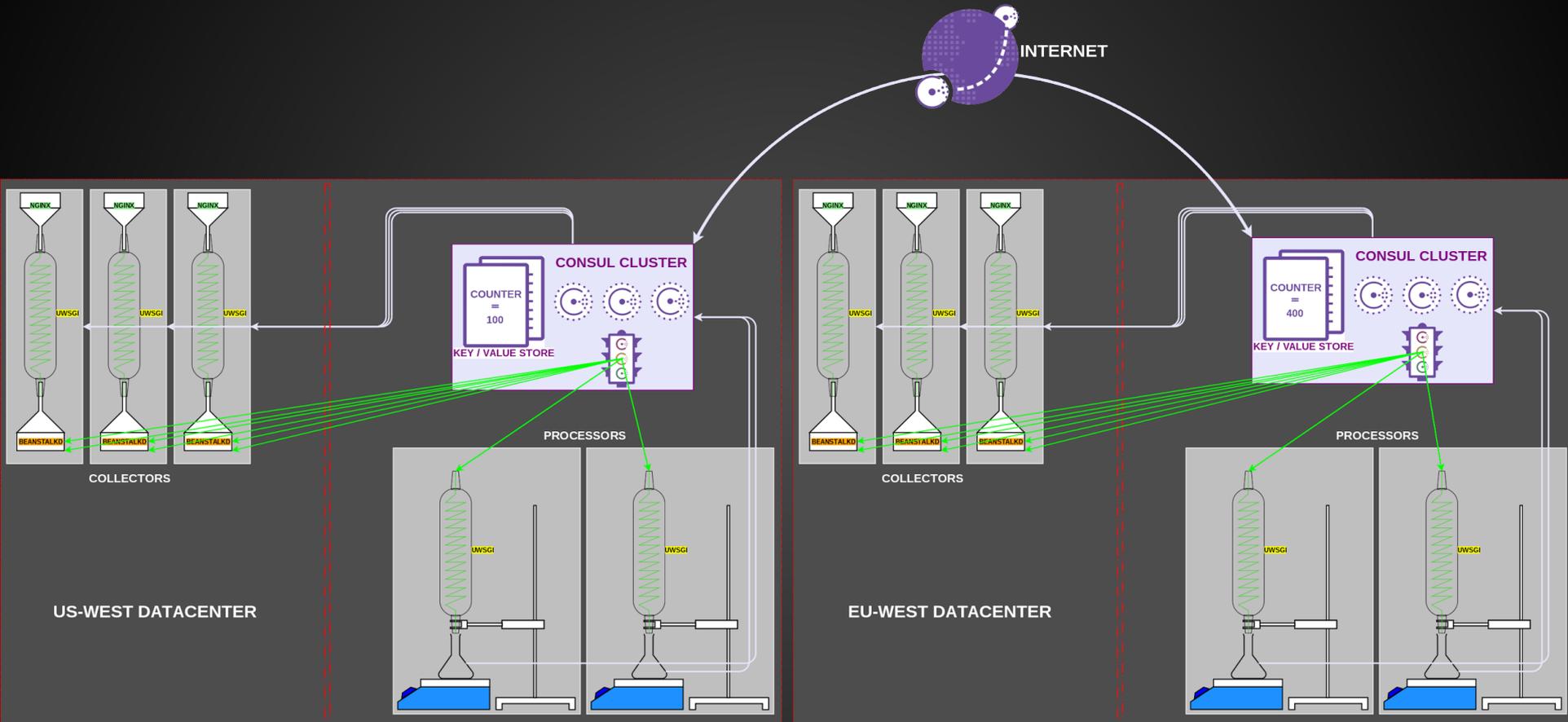


PROCESSOR













Live demo !

(hopefully)

A world map with a dark green background, overlaid with a complex network of thin, glowing purple lines that represent global connections or data flow. The lines are most dense in North America and Europe, with many lines radiating outwards to other parts of the world.

Thanks

source code : github.com/ultrabug/ep2015
[@ultrabug](https://twitter.com/ultrabug)