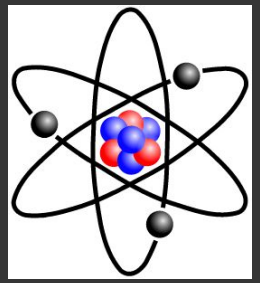
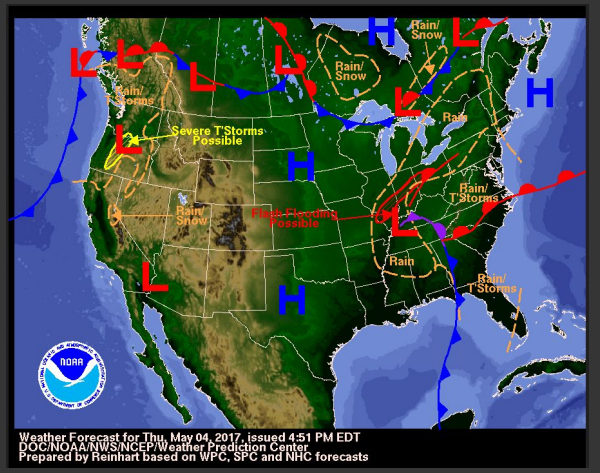
Logging and Observability in Distributed Systems

David Jahn
(VMware, Meta)

How do we know
about the state of systems?

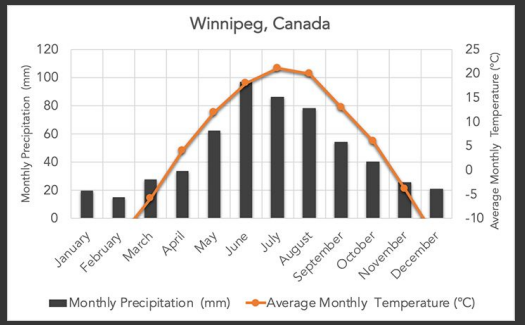




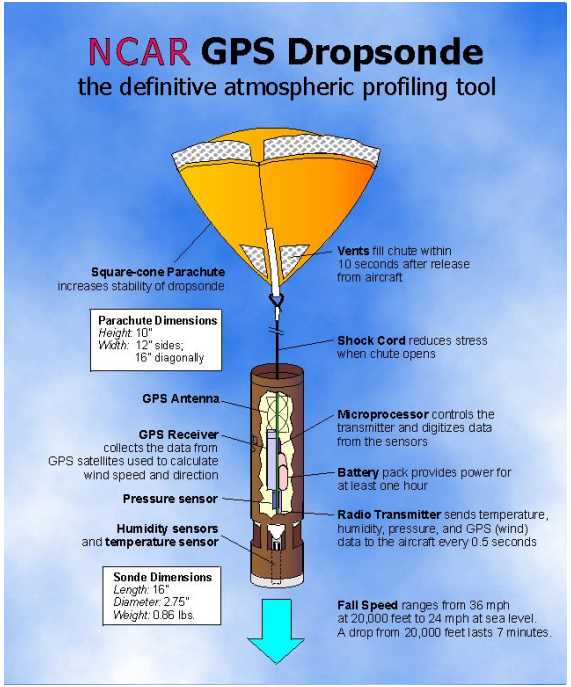
Measurements (observables)

- Numbers
  - “The thermometer reads 78 degrees”
  - Usually a function of time / space

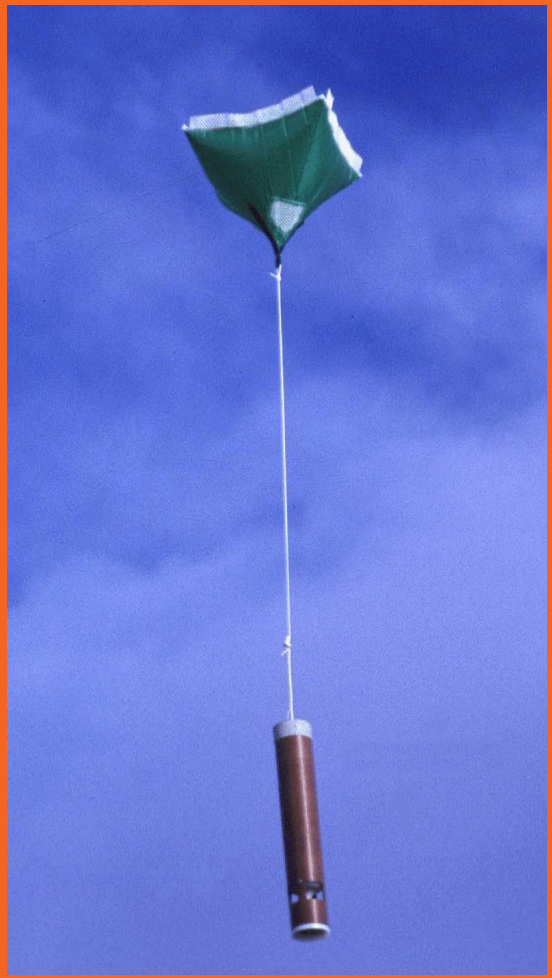
- Not numbers
  - “The radiator looks broken”
  - “I see error lines in the log file”




```
Started GET "/users" for 127.0.0.1 at 2019-12-21 19:58:56 +0100
Processing by UsersController#index as */*
UsersController => index
  Rendering users/index.html.erb within layouts/application
  Rendered users/index.html.erb within layouts/application (81.9ms)
Completed 500 Internal Server Error in 0ms
ZeroDivisionError (divided by 0):
app/controllers/v1/tokens_controller.rb:6:in `/'
app/controllers/v1/tokens_controller.rb:6:in `create'
app/middleware/throttling.rb:13:in `call'
app/middleware/token_parser.rb:8:in `call'
```



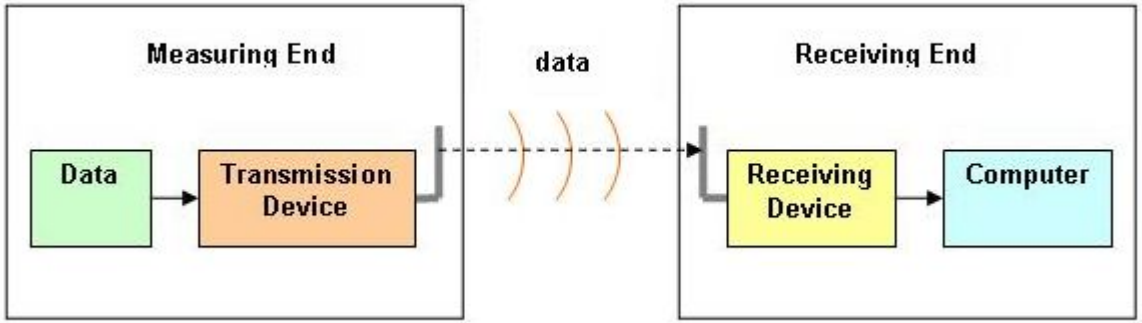
- Sensors (**instrumentation**)
  - Collect measurement
- **Transmitter**
  - Send data home



Telemetry

Telemetry refers to

1) **Collection** of data
2) **Transmission / Storage**
3) Analysis / Usage



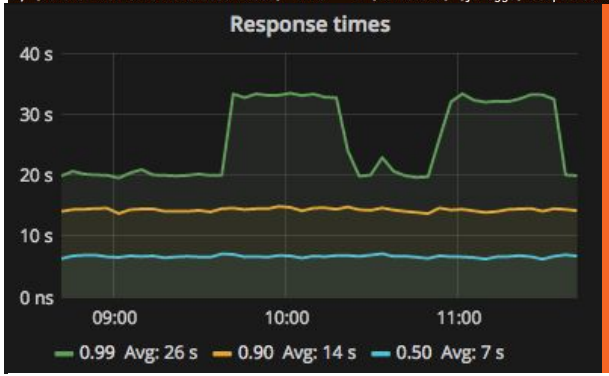
Examples: Aviation, Weather Research, Mining/Drilling, Vehicles, etc.

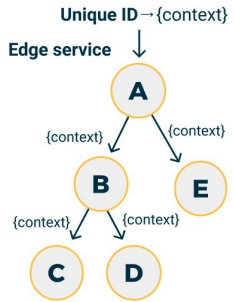
Telemetry in Software

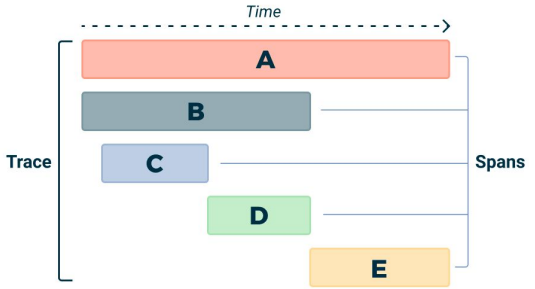
(also called **Observability**)

- **Logging**
  Log lines from a web server
- **Monitoring / Metrics**
  Requests Per Second, Bytes Per Second,
  Errors Per Second, Latency, CPU usage
- **Distributed Tracing**
  Graph structure
  (service A called service B)
- **APM (Application Performance Monitoring/Metrics/Management)**
  Catch-all buzzword

```
I, [2016-02-26T15:43:26.394890 #2333]  INFO -- : [127.0.0.1] [jbloggs] Started GET "/groups" for 127.0.0.1 at 2016-02-26 15:43:26 +0000
I, [2016-02-26T15:43:26.404830 #2333]  INFO -- : [127.0.0.1] [jbloggs] Processing by GroupsController#index as HTML
I, [2016-02-26T15:43:26.456395 #2333]  INFO -- : [127.0.0.1] [jbloggs]   Rendered cascading_gals/_groups_select.html.haml (33.1ms)
I, [2016-02-26T15:43:26.471248 #2333]  INFO -- : [127.0.0.1] [jbloggs]   Rendered search/_search_form.html.haml (50.4ms)
I, [2016-02-26T15:43:26.504085 #2333]  INFO -- : [127.0.0.1] [jbloggs]   Rendered action_links/_crud.html.haml (5.9ms)
I, [2016-02-26T15:43:26.504693 #2333]  INFO -- : [127.0.0.1] [jbloggs]   Rendered groups/_group.html.haml (25.5ms)
I, [2016-02-26T15:43:26.509209 #2333]  INFO -- : [127.0.0.1] [jbloggs]   Rendered groups/index.html.haml within layouts/application (89.4ms)
I, [2016-02-26T15:43:26.579240 #2333]  INFO -- : [127.0.0.1] [jbloggs]   Rendered layouts/_ie_conditional.html.haml (61.3ms)
I, [2016-02-26T15:43:26.610566 #2333]  INFO -- : [127.0.0.1] [jbloggs]   Rendered layouts/_menu.html.haml (29.1ms)
I, [2016-02-26T15:43:26.612351 #2333]  INFO -- : [127.0.0.1] [jbloggs]   Rendered layouts/_header.html.haml (0.2ms)
I, [2016-02-26T15:43:26.614910 #2333]  INFO -- : [127.0.0.1] [jbloggs]   Rendered layouts/_logout_warning.html.haml (0.7ms)
I, [2016-02-26T15:43:26.619572 #2333]  INFO -- : [127.0.0.1] [jbloggs]   Rendered layouts/_announcement.html.haml (2.7ms)
I, [2016-02-26T15:43:26.621150 #2333]  INFO -- : [127.0.0.1] [jbloggs]   Rendered layouts/_reviewable_control_files.html.haml (0.1ms)
I, [2016-02-26T15:43:26.622449 #2333]  INFO -- : [127.0.0.1] [jbloggs]   Rendered layouts/_flasher.html.haml (0.1ms)
I, [2016-02-26T15:43:26.624343 #2333]  INFO -- : [127.0.0.1] [jbloggs]   Rendered layouts/_footer.html.haml (0.5ms)
I, [2016-02-26T15:43:26.625069 #2333]  INFO -- : [127.0.0.1] [jbloggs] Completed 200 OK in 220ms (Views: 207.9ms | ActiveRecord: 5.3ms | Mongo:
```







How does it work?

(almost) all software telemetry is something like:

- Import logging library/package into your code
- Configure
- Add custom metrics/logs (if desired)
- Many libraries are designed to work “out of the box” in popular app frameworks

Once your app is “instrumented”, you need:

- Processing system to receive logs
- Storage + querying system
- UI/UX to visualize, search logs
- Alerting / paging system

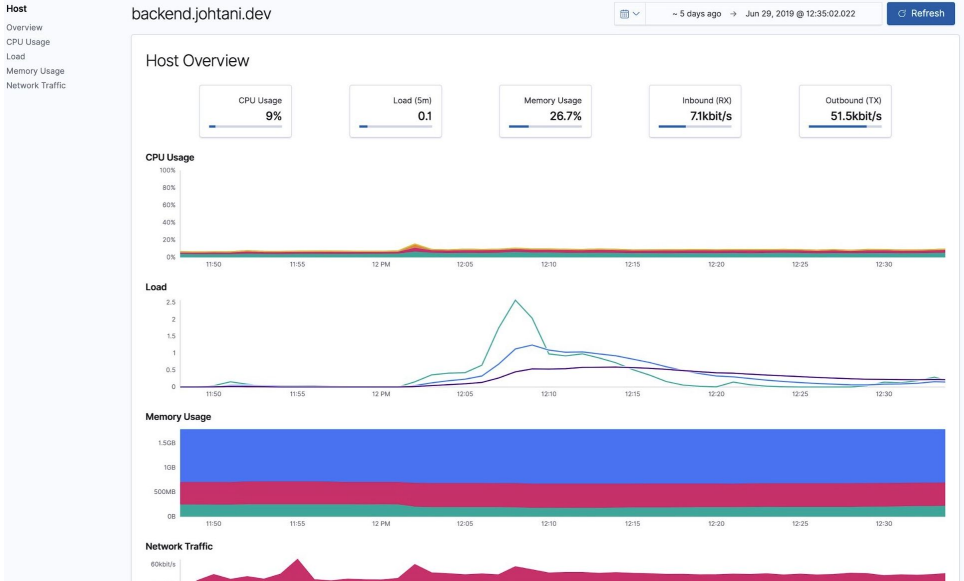
When using Elastic APM with Ruby on Rails, configure the agent as follows. First, add to use to the Gemfile.

```
# For Elastic APM
gem 'elastic-apm'
```

Then, create a file called `elastic_apm.yml` in the `config` directory, and add your serve

```
server_url: <%= ENV['ELASTIC_APM_SERVER_URL'] %>
secret_token: <%= ENV['ELASTIC_APM_SECRET_TOKEN'] %>
```

That’s it! The agent will automatically measure the processing time of actions and SQ



Telemetry Value

➔ **Operations / Devops**

Know when things break (ideally before they break)

Reduce downtime

→ Some sites lose millions in revenue per hour in outages

➔ **Performance**

Reduce latency

Improve perf / Reduce expenses

→ Some sites have high infra costs, millions or billions of $

➔ **Security**

Monitor potential attacks / threats

Company	Founded	Funding	Today
Splunk	2003	$2.3B	market cap $10B
Dynatrace	2005	$22M	market cap $10B
App Dynamics	2008	$364M	$3.7B acquisition, 2017 by Cisco
New Relic	2008	$214M	market cap $3.5B
Datadog	2010	$148M	market cap $20B
Elastic	2012	$162M	market cap $5B
Logz.io	2014	$122M	-
Instana	2015	$57M	$??B acquisition, 2020 by IBM
Honeycomb	2016	$85M	-
Observe	2017	$112M	-

This is a large industry.

There are many companies and startups (hundreds) that focus on observability / logging / APM.

VCs have invested billions of dollars.

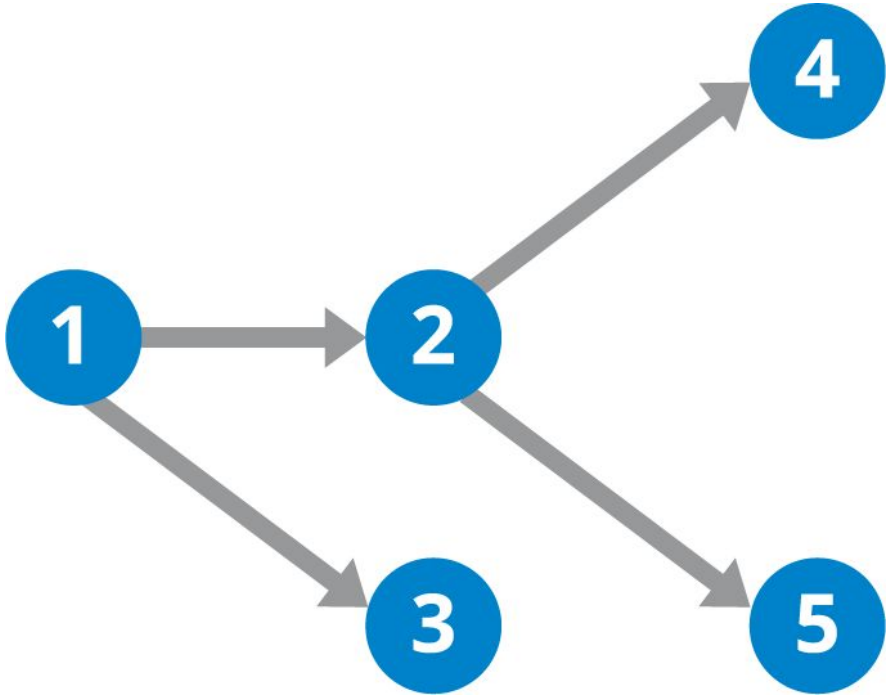
APM companies focus on developing logging libraries, along with processing backends + frontends.

Distributed Tracing: a (small) deep dive

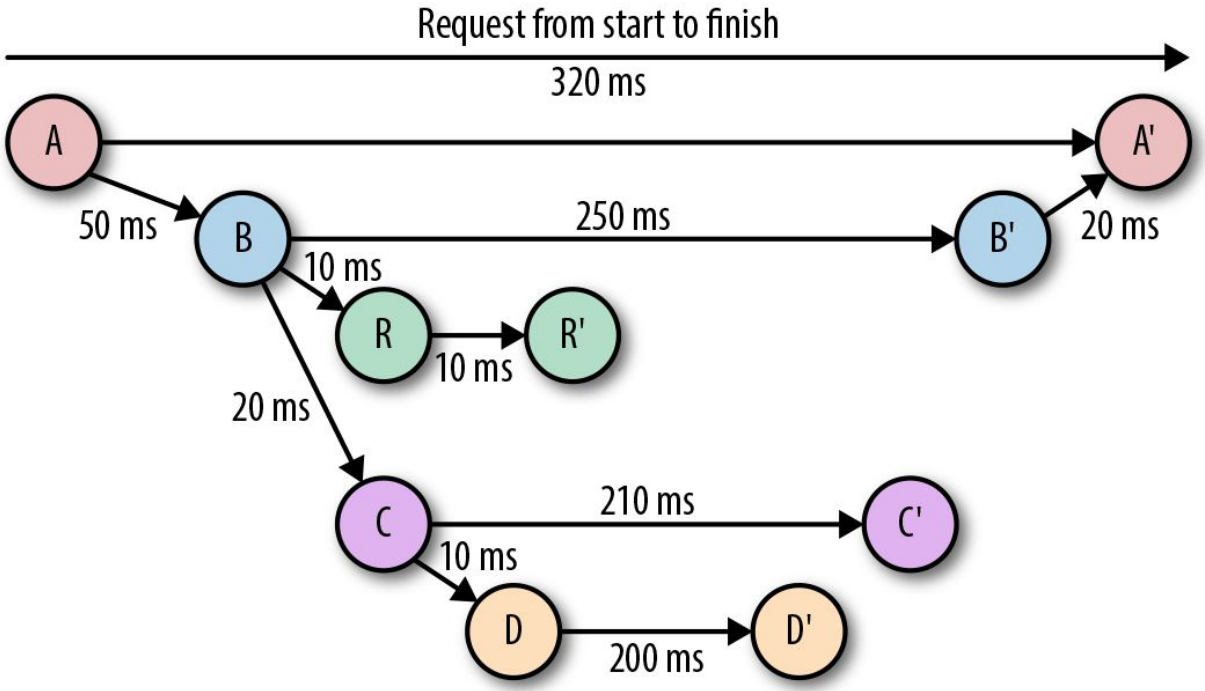
Websites / Apps are often **multiple services** that talk to each other.

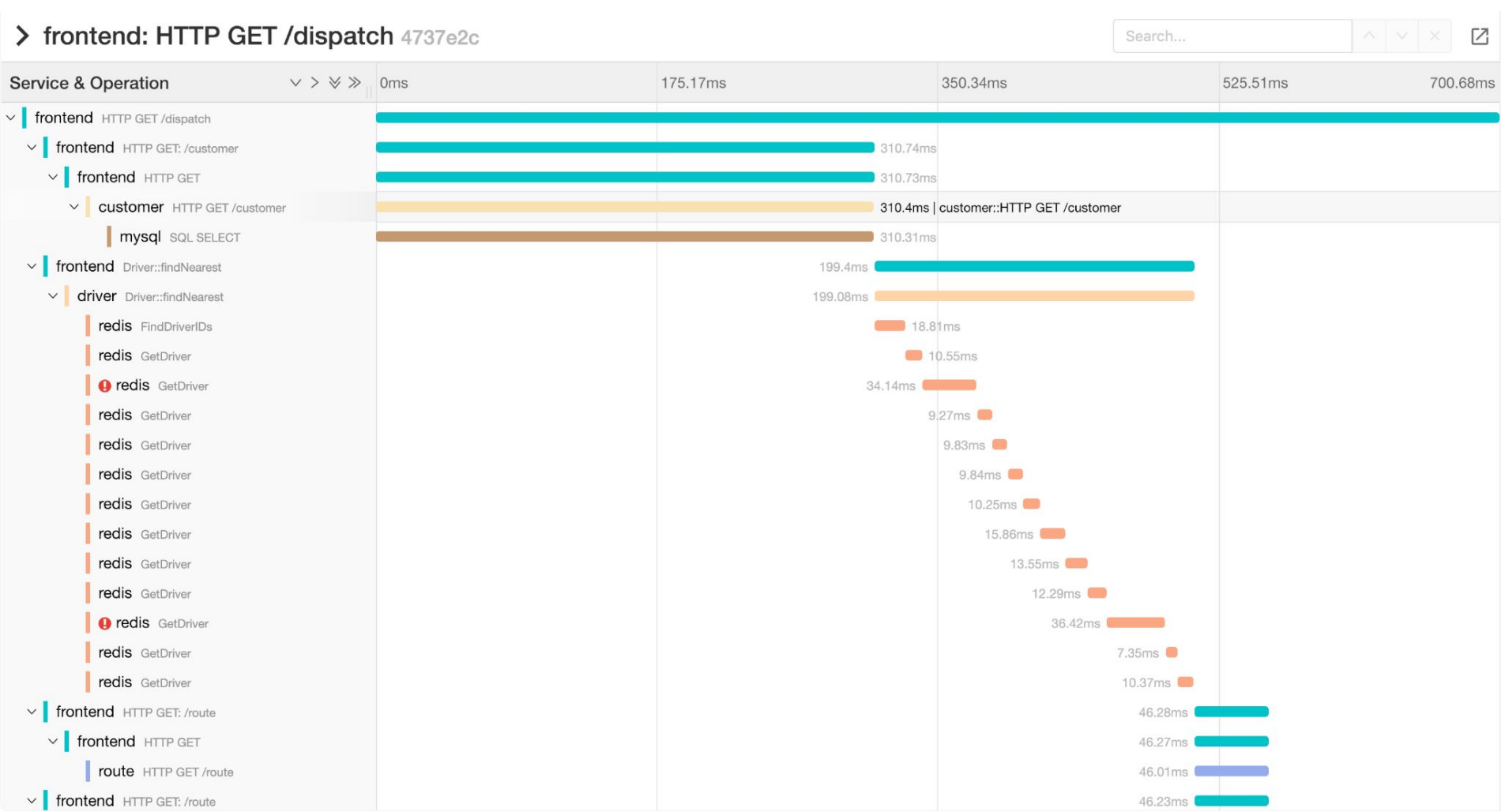
Web server → Database
Web server → Cache
Web server
→ Authentication Service
→ Shopping Cart Service
→ Database

Each call forms a (mathematical) **graph**, in particular a **DAG** (directed acyclic graph)



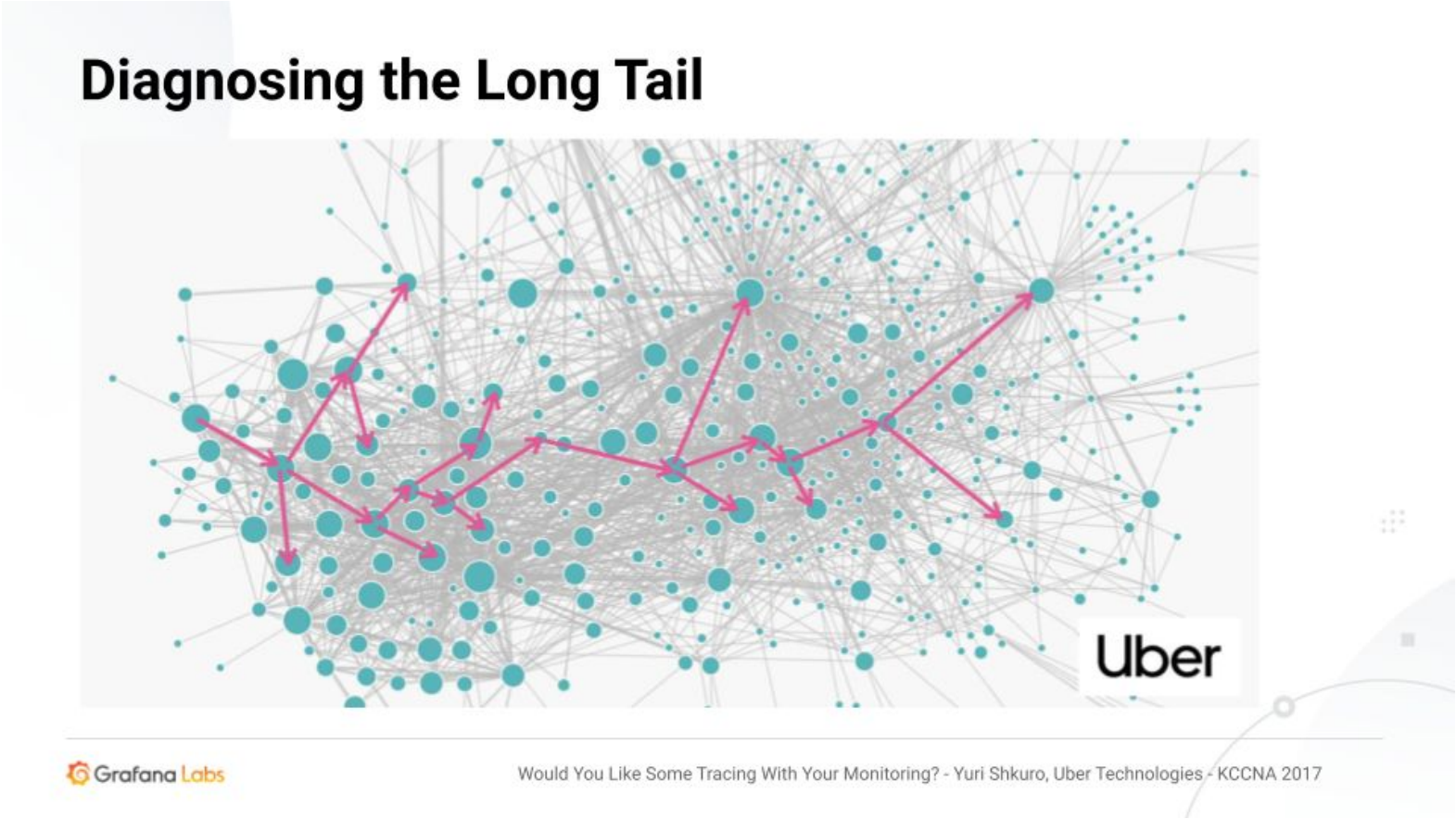
A **distributed trace** is a log of the call graph from an application request





(Jaeger open-source tracing system)

Depending on the scale of a company, and their architectural decisions… Traces can be large

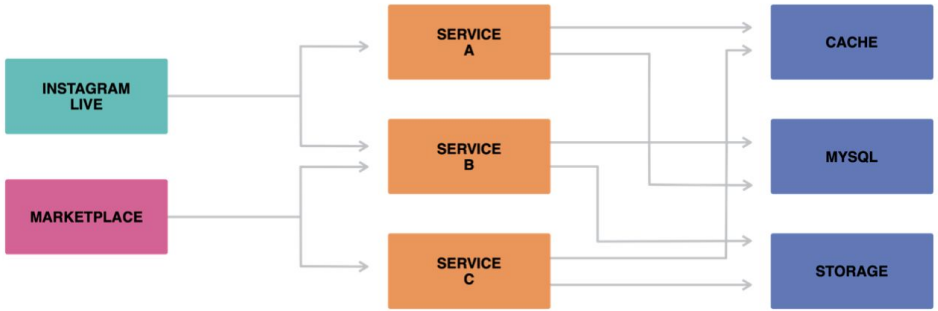


Use Cases for Distributed Traces

Latency Debugging (why is my request slow?)

Performance Analysis (what is the global cost of a request?)

Debugging system failures (what service failed?)



Allocating cost to different products when resources are shared (resource accounting)

Akanksha Bansal and Richard Cornew Systems@Scale 2021 presentation "End-to-End Resource Accounting Leveraging Distributed Tracing"

Thanks!

Further reading:

- Dapper whitepaper - Google distributed tracing system
- Canopy whitepaper - Meta/Facebook distributed tracing system
- OpenTelemetry - open source standard + toolset for logging and tracing
- The Three Pillars of Observability - Chapter from O'Reilly book
- The Butterfly Effect in Distributed Systems - Distributed resource accounting