CODE EXECUTION
AS
DATA

Zach Tellman
@ztellman

factual™

WE BUILD SYSTEMS

WHICH

CREATE DATA FASTER

THAN WE CAN

CONSUME DATA

A NEVERENDING COCKTAIL PARTY

A NEVERENDING COCKTAIL PARTY

```
[alice,  "curiouser",  [1.2, 3.5], 1234567890]
[hamlet, "to",         [1.3, 0.8], 1234567891]
[alice,  "and",        [1.2, 3.5], 1234567892]
[hamlet, "be",         [1.3, 0.8], 1234567893]
                     .
                     .
                     .
```

NARRATIVES

ARE BOTH

REVEALING

AND

REDUCTIVE

"The work has involved such things as file organizations, indexes, hierarchical structures..., and so on. After a while it dawned on me that **these are all just maps, being poor artificial approximations of some real underlying terrain**."

William Kent
Data and Reality

OUR TOOLS

- filter
- sample
- aggregate
- group

IMPLICIT ASSUMPTIONS

- filter
- sample
- aggregate
- group

LOGGING

- immediate presentation
- merged into a single stream

IMPLICIT ASSUMPTIONS

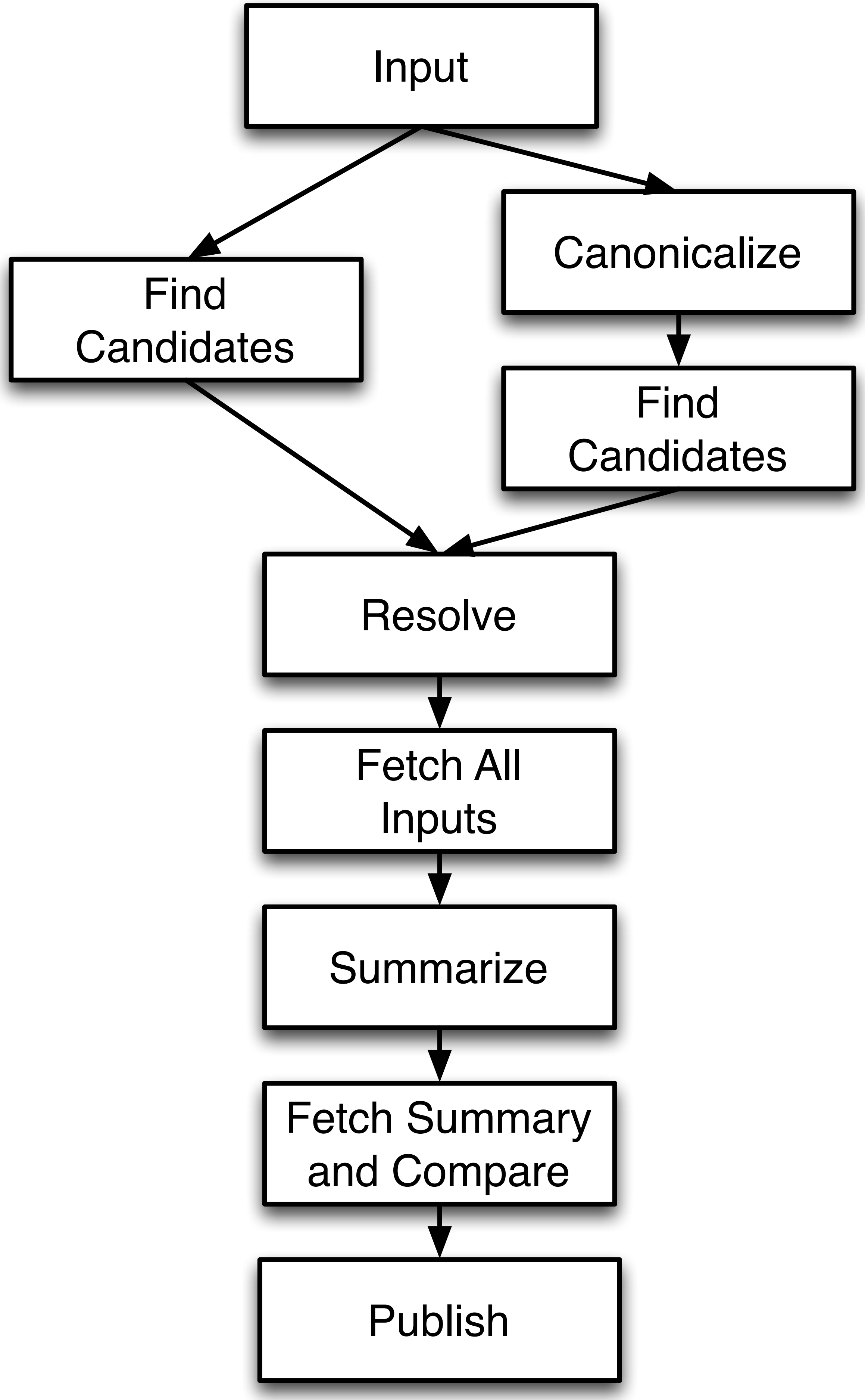
- anything we don’t log, we’ll never need
- the logs are useful for humans
- the logs are useful for machines

MY OWN NARRATIVE

MY IMPLICIT ASSUMPTIONS

- code execution can be treated as streams of events
- most events are related, at least a little
- we often confuse the map for the place itself

factual™


Input
Find Candidates
Canonicalize
Find Candidates
Resolve
Fetch All Inputs
Summarize
Fetch Summary and Compare
Publish

MY GOALS

- to create a stable, fast system
- to see if my assumptions are sound

GENERATING EVENTS

```
(trace :foo:bar
  {:value 123
   :description "a number!"})
```

CONSUMING EVENTS

```
(probe-channel :foo:bar)

(select-probes "fo*" #"ba[r|z]")
```

FILTERING EVENTS

- map*
- filter*
- take-while*
- drop-while*
- transitions

SAMPLING EVENTS

- sample-every
- sample
- moving-sample
- quantiles
- moving-quantiles

AGGREGATING EVENTS

- moving-average
- rate
- sum
- reduce*
- reductions*

GROUPING EVENTS

- partition-every
- distribute-aggregate

THE GOOD

- single data structure, many operators
- distinct streams for distinct data
- decoupled data and presentation

THE BAD

- the only obvious relationships between streams are via name structure
- less noisy than logging, but still distracting

INSTRUMENTING FUNCTIONS

```
(instrument +
  {:name "plus"
   :capture :in-out})
```

```
(probe-channel :plus:enter)

  {:name "plus"
   :context {:host "xanadu"
             :pid 1234}
   :timestamp 1234567890
   :args [1 2 3]}
```

```
(probe-channel :plus:return)
```

```
{:name "plus"
 :context {:host "xanadu"
           :pid 1234}
 :timestamp 1234567890
 :duration 10000
 :sub-tasks nil
 :args [1 2 3]
 :result 6}
```

```
(probe-channel :plus:error)
```

```
{:name "plus"
 :context {:host "xanadu"
           :pid 1234}
 :timestamp 1234567890
 :duration 10000
 :sub-tasks nil
 :args [1 2 nil]
 :error ...}
```

```
(defn-instrumented bar
  [x y]
  (+ x y))

(defn-instrumented foo
  {:name "custom-foo"
   :capture :in-out}
  [x y]
  (bar x y))
```

```
{:name "custom-foo"
 :context {:host "xanadu"
           :pid 1234}
 :timestamp 1234567890
 :duration 10000
 :sub-tasks [{:name "user:bar"
              :context {:host "xanadu"
                        :pid 1234}
              :timestamp 1234567890
              :duration 8000
              :sub-tasks nil }]
  :args [1 2]
  :result 3}
```

THE GOOD

- easy annotation of significant functions
- first-class representation of related events

THE BAD

- pretty noisy

```
(distill-timing timing)
(merged-distilled-timings & timings)
```

```
{:name "custom-foo"
 :context {:host "xanadu"
           :pid 1234}
 :timestamp 1234567890
 :durations [10000]
 :sub-tasks [{:name "user:bar"
              :durations [8000]}]}
```

```
cat traces.gz | gunzip | pprint-traces
```

```
26.3%  ( 1.2% ) | ds-api-request-handler
50th 2.0ms, 75th 3.0ms, 95th 12.8ms, 99th 58.8ms, 99.9th 2247.3ms
67180142 calls
------
    9.0%  ( 0.3% ) | datastore-api-server:server:diffs:get-diffs-handler
    50th 2.0ms, 75th 2.8ms, 95th 10.0ms, 99th 47.8ms, 99.9th 644.4ms
    66236339 calls
    ------
        8.7%  ( 8.7% ) | datastore-api-server:services:diffs:diff-seq
        50th 2.0ms, 75th 2.0ms, 95th 11.0ms, 99th 37.8ms, 99.9th 550.4ms
        66236334 calls
        ------
```

cat traces.gz | gunzip | pprint-traces

```
26.3%  ( 1.2% ) | ds-api-request-handler
50th 2.0ms, 75th 3.0ms, 95th 12.8ms, 99th 58.8ms, 99.9th 2247.3ms
67180142 calls
------
    9.0%  ( 0.3% ) | datastore-api-server:server:diffs:get-diffs-handler
    50th 2.0ms, 75th 2.8ms, 95th 10.0ms, 99th 47.8ms, 99.9th 644.4ms
    66236339 calls
    ------
        8.7%  ( 8.7% ) | datastore-api-server:services:diffs:diff-seq
        50th 2.0ms, 75th 2.0ms, 95th 11.0ms, 99th 37.8ms, 99.9th 550.4ms
        66236334 calls
        ------
```

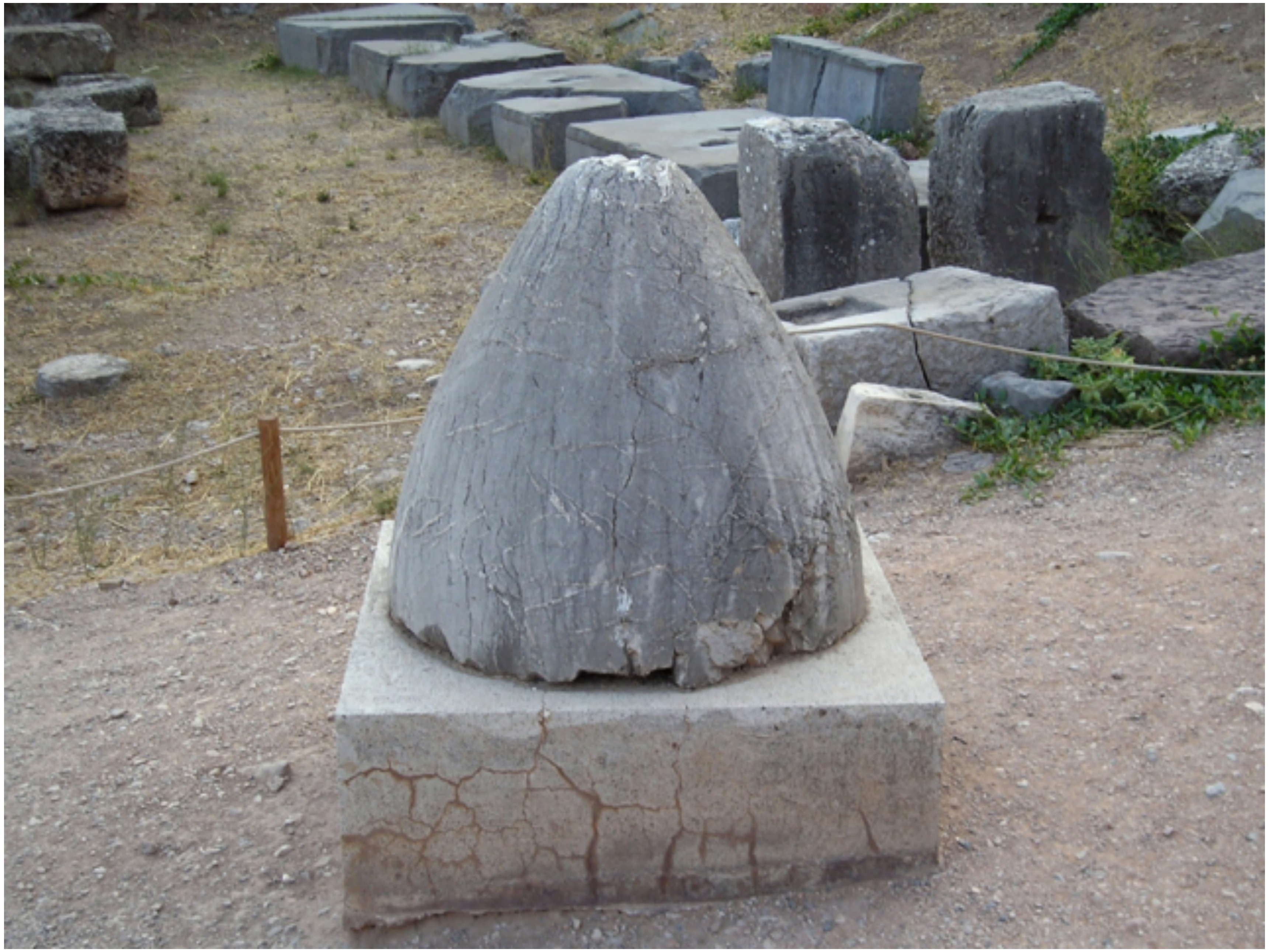
THE GOOD

- multiple high-level views of data
- doesn’t elide hierarchical structure

THE BAD

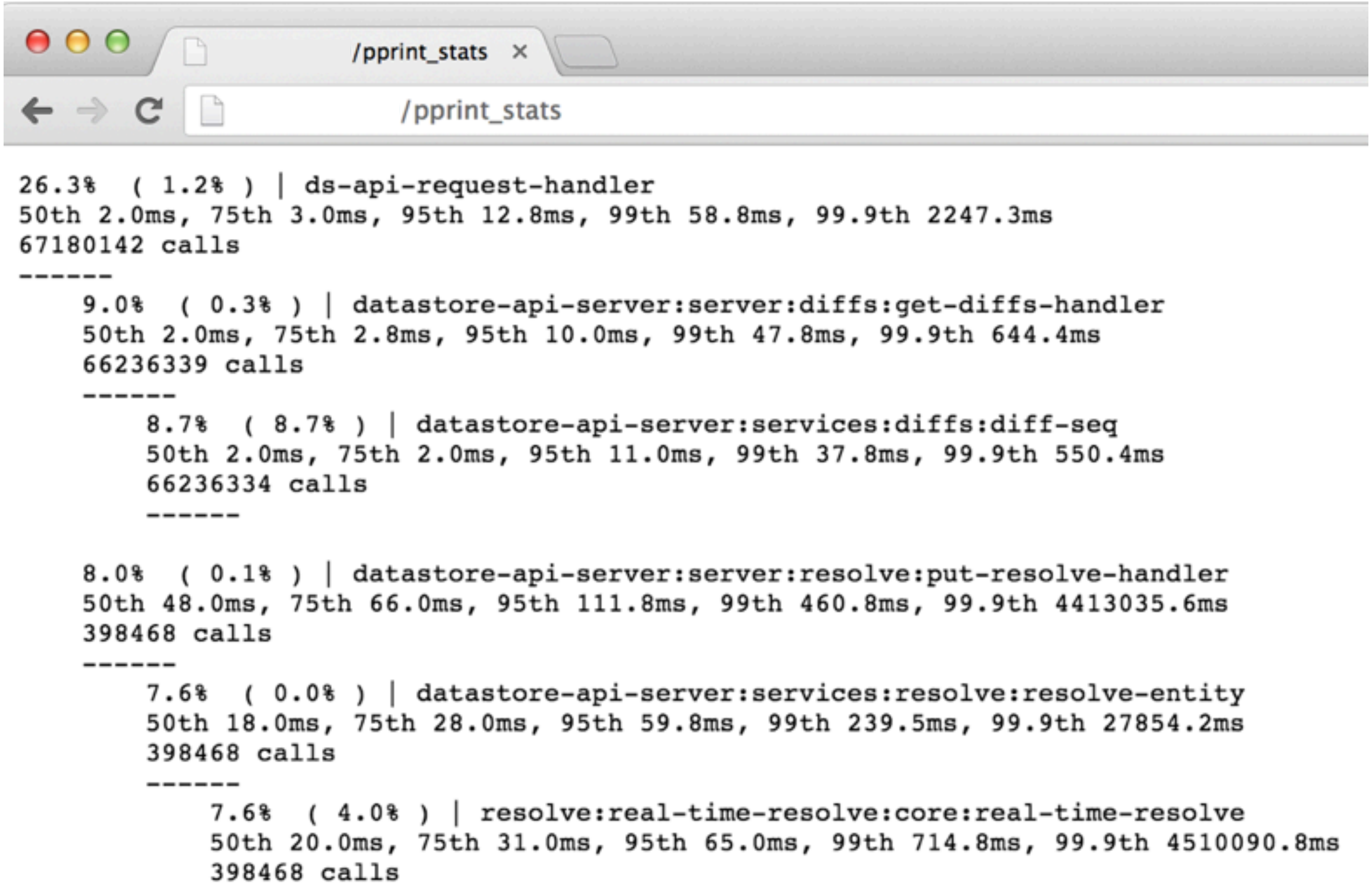
- memory intensive
- static presentation
- still only a subset of the entire process

OMPHALOS

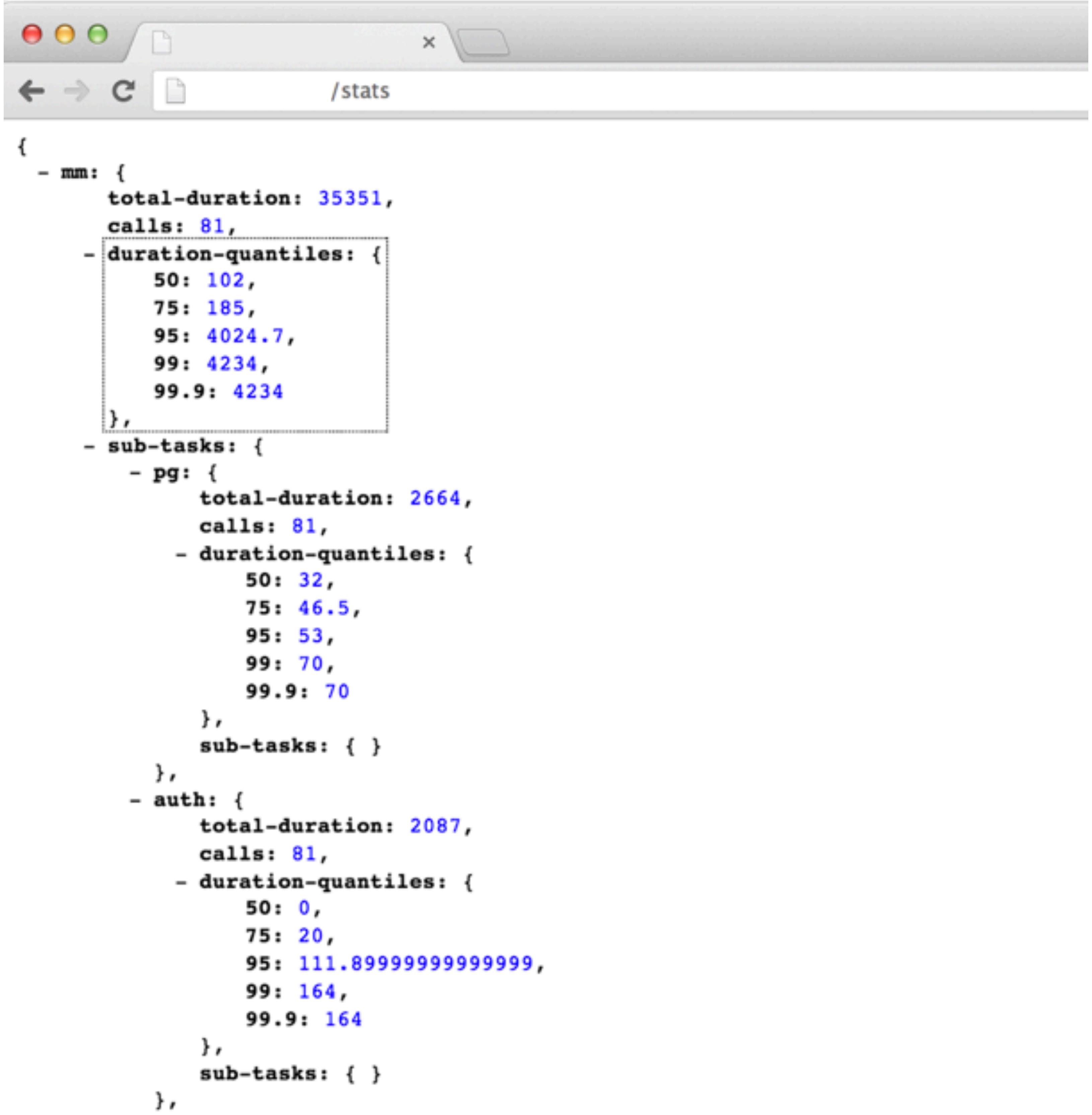


OMPHALOS

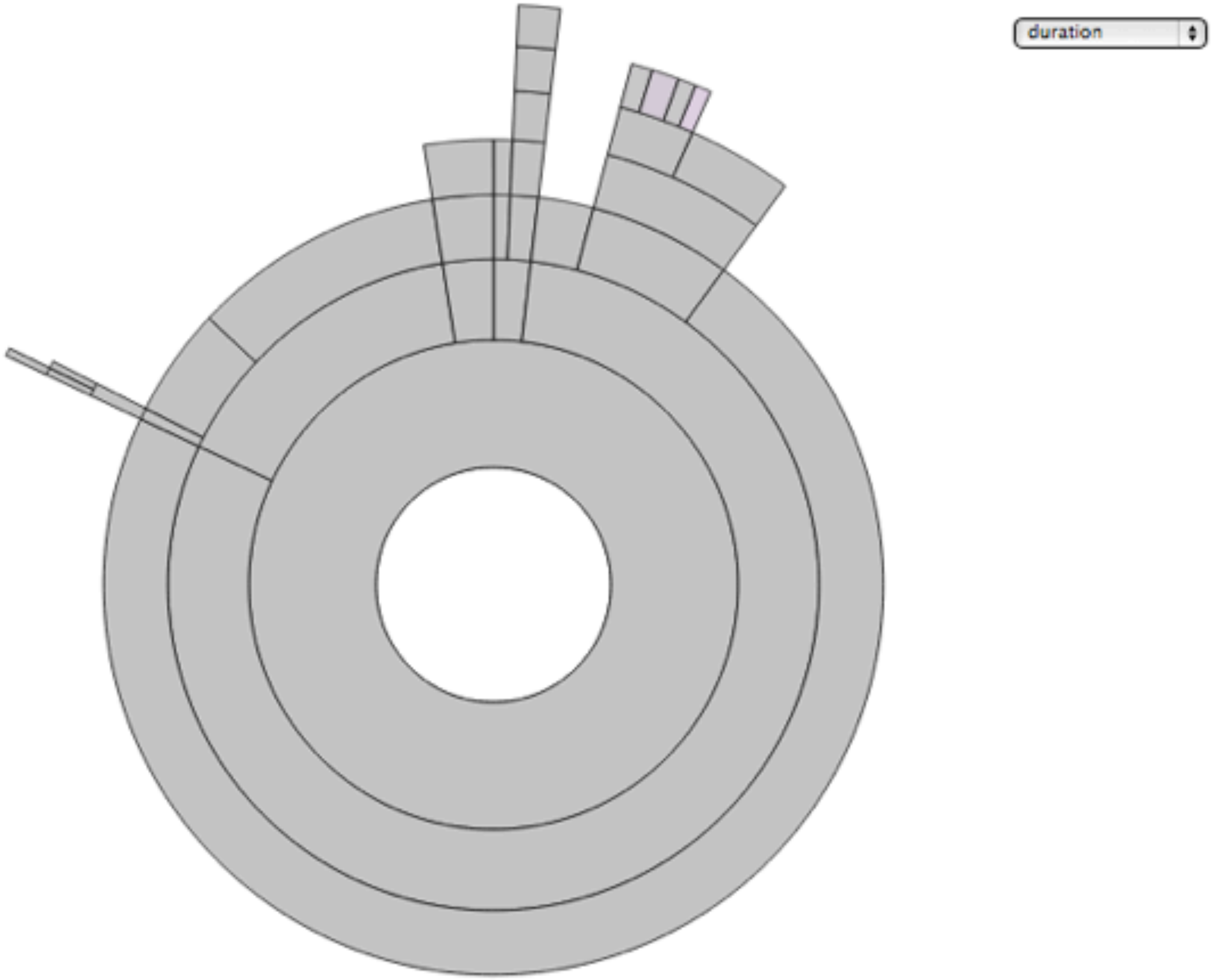
- add distilled-timing instrumentation to all libraries and processes
- deliver via UDP to central server



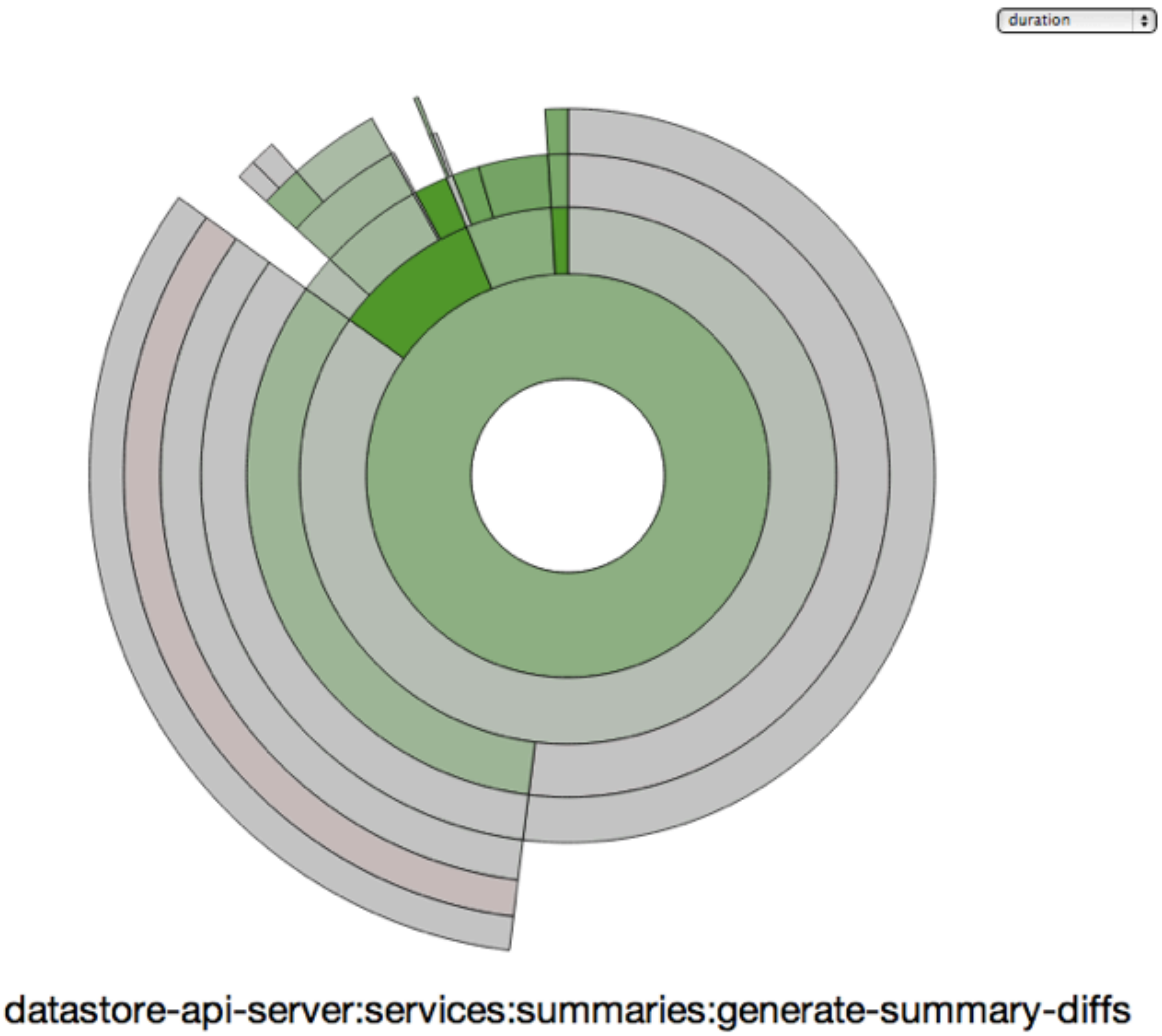
```
26.3%  ( 1.2% ) | ds-api-request-handler
50th 2.0ms, 75th 3.0ms, 95th 12.8ms, 99th 58.8ms, 99.9th 2247.3ms
67180142 calls
------
    9.0%  ( 0.3% ) | datastore-api-server:server:diffs:get-diffs-handler
    50th 2.0ms, 75th 2.8ms, 95th 10.0ms, 99th 47.8ms, 99.9th 644.4ms
    66236339 calls
    ------
        8.7%  ( 8.7% ) | datastore-api-server:services:diffs:diff-seq
        50th 2.0ms, 75th 2.0ms, 95th 11.0ms, 99th 37.8ms, 99.9th 550.4ms
        66236334 calls
        ------

    8.0%  ( 0.1% ) | datastore-api-server:server:resolve:put-resolve-handler
    50th 48.0ms, 75th 66.0ms, 95th 111.8ms, 99th 460.8ms, 99.9th 4413035.6ms
    398468 calls
    ------
        7.6%  ( 0.0% ) | datastore-api-server:services:resolve:resolve-entity
        50th 18.0ms, 75th 28.0ms, 95th 59.8ms, 99th 239.5ms, 99.9th 27854.2ms
        398468 calls
        ------
            7.6%  ( 4.0% ) | resolve:real-time-resolve:core:real-time-resolve
            50th 20.0ms, 75th 31.0ms, 95th 65.0ms, 99th 714.8ms, 99.9th 4510090.8ms
            398468 calls
```



```
{
 - mm: {
       total-duration: 35351,
       calls: 81,
     - duration-quantiles: {
          50: 102,
          75: 185,
          95: 4024.7,
          99: 4234,
          99.9: 4234
       },
     - sub-tasks: {
        - pg: {
              total-duration: 2664,
              calls: 81,
            - duration-quantiles: {
                 50: 32,
                 75: 46.5,
                 95: 53,
                 99: 70,
                 99.9: 70
              },
              sub-tasks: { }
          },
        - auth: {
              total-duration: 2087,
              calls: 81,
            - duration-quantiles: {
                 50: 0,
                 75: 20,
                 95: 111.89999999999999,
                 99: 164,
                 99.9: 164
              },
              sub-tasks: { }
          },
```



datastore-api-server:services:summaries:summarize-views



duration

datastore-api-server:services:summaries:generate-summary-diffs

HISTORICAL ANALYSIS

```
(query-seq
  lamina.stats/rate
  {:timestamp :timestamp}
  (log-seq))
```

CLOSING THOUGHTS

- mind your assumptions
- design with transparency and introspection as first class concerns
- always allow for further narratives

QUESTIONS?