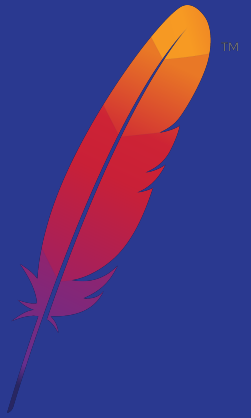
Apache SkyWalking

Sheng Wu 吴晟

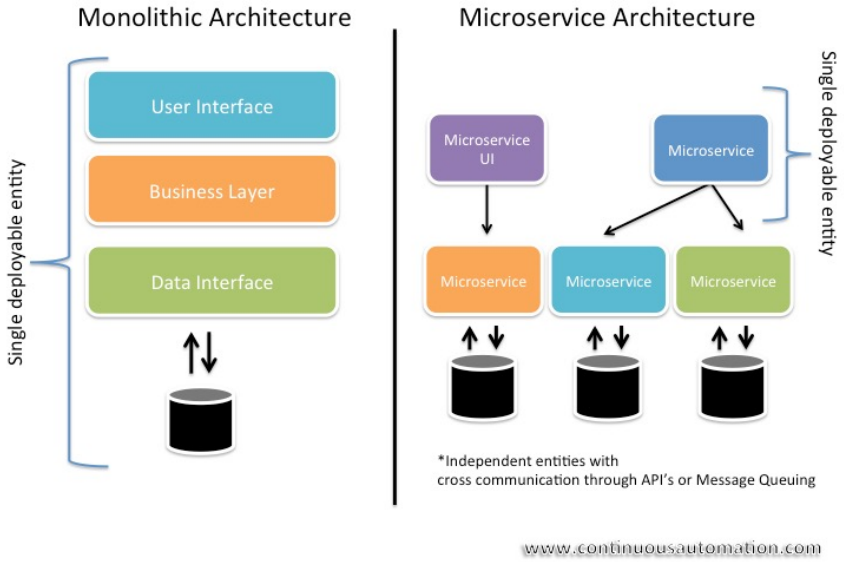
Tetrate Founding Engineer
Apache Board Director
Apache SkyWalking Founder & V.P.

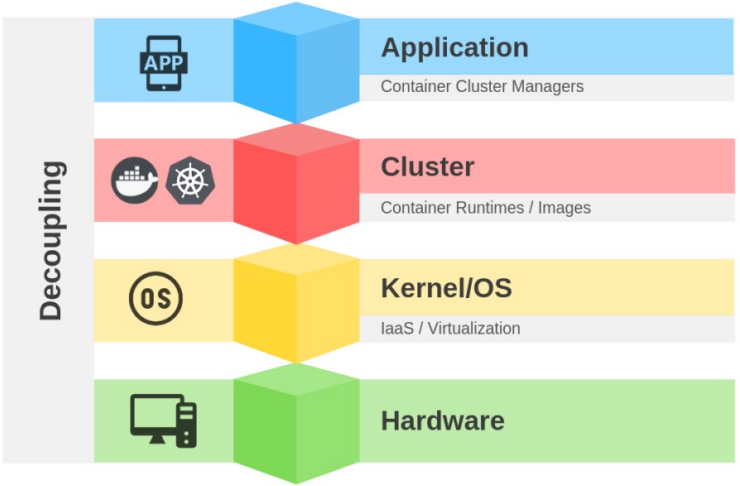


Profile

- Founding Engineer, Tetrate
- Work on Service Mesh observability
- Board Director of Apache Software Foundation
- VP and PMC member, Apache SkyWalking
- PMC member of Apache APISIX , Incubator, ShardingSphere, ECharts, DolphinScheduler
- Member of CNCF KubeCon Program Committee, 2018-2020

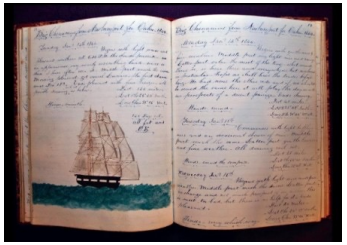
Modern Distributed System





Metrics

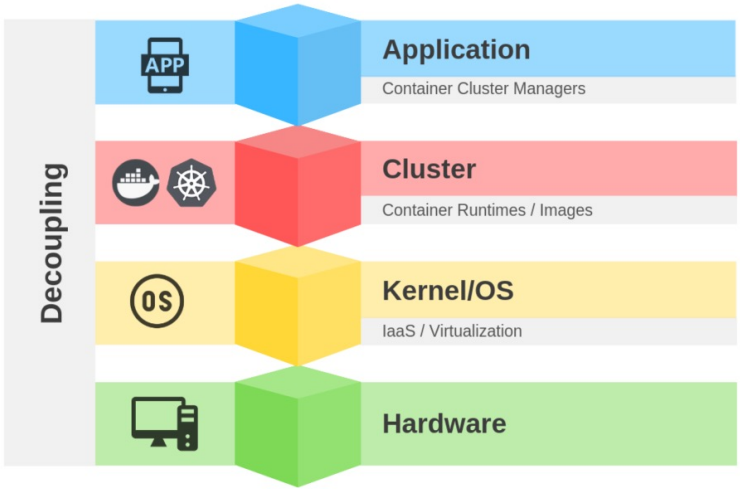
Tracing



Logging

Event/Alarm

Monitoring Landscape and Solution



Language Agents for In-Process Monitoring

K8s Metrics Monitoring

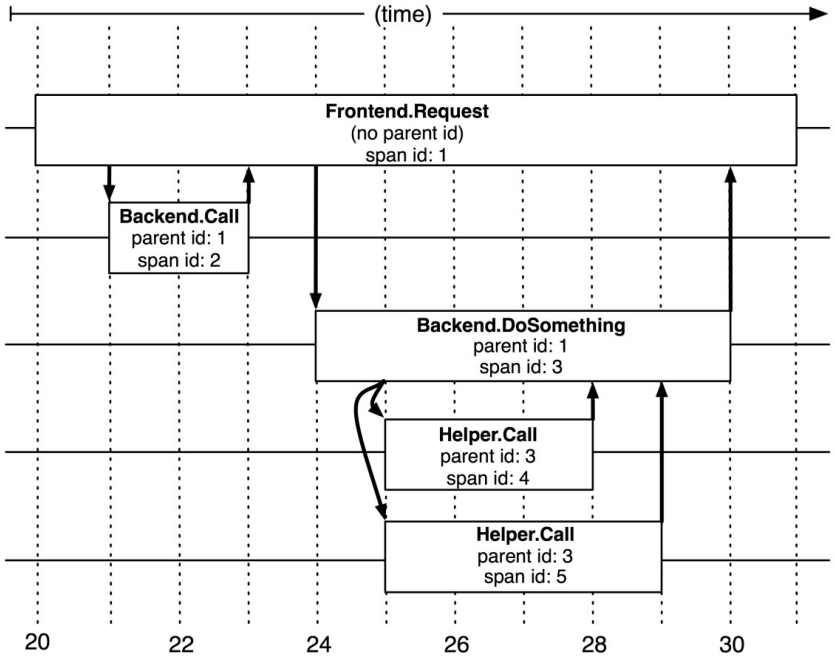
Traditional OS Monitoring

Tracing

The hottest topic in Obs. in these 3 years

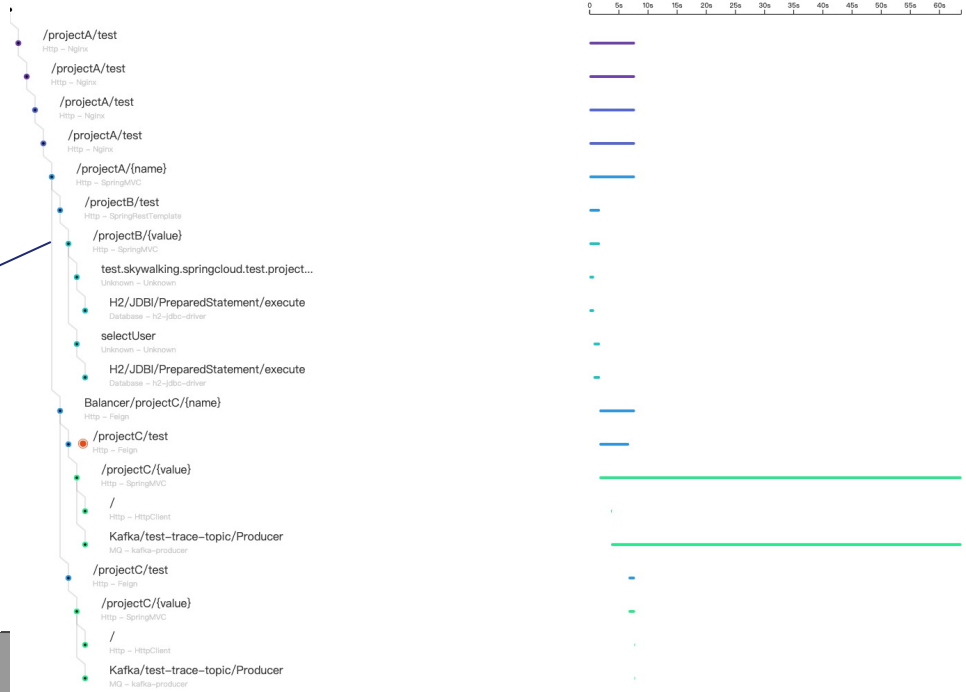
Dapper – 10 years ago Google Research

- Trace Concept
- Span Concept
- Context-enhanced Logs
- Low sampling rate



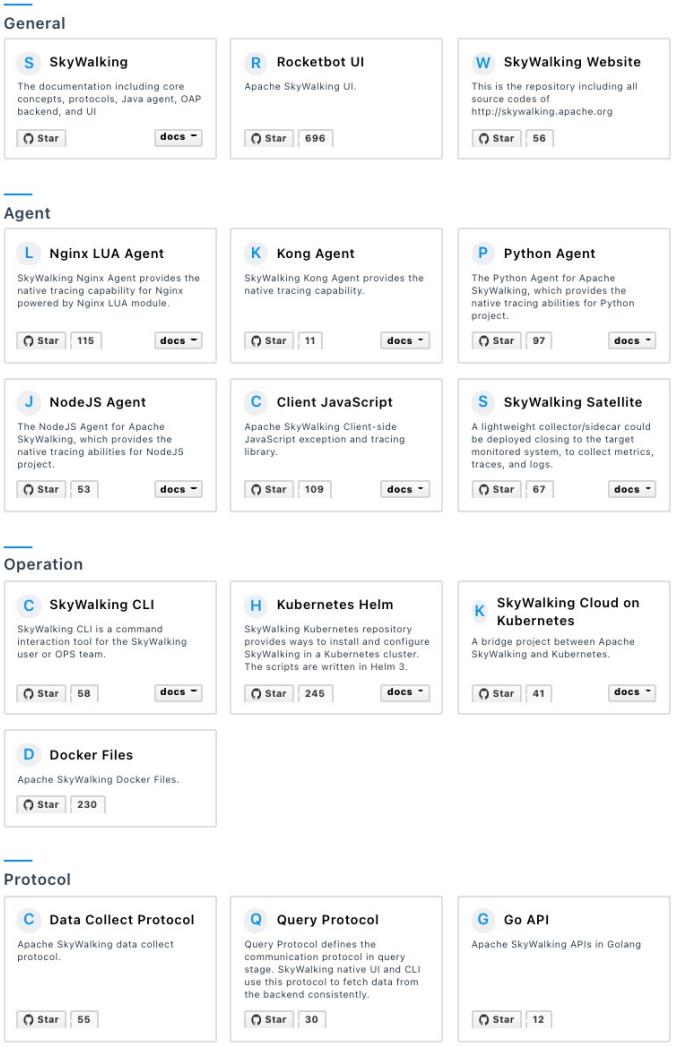
Tracing

- Dapper+
- Streaming Topology Analysis
- Customize Metrics Script - OAL



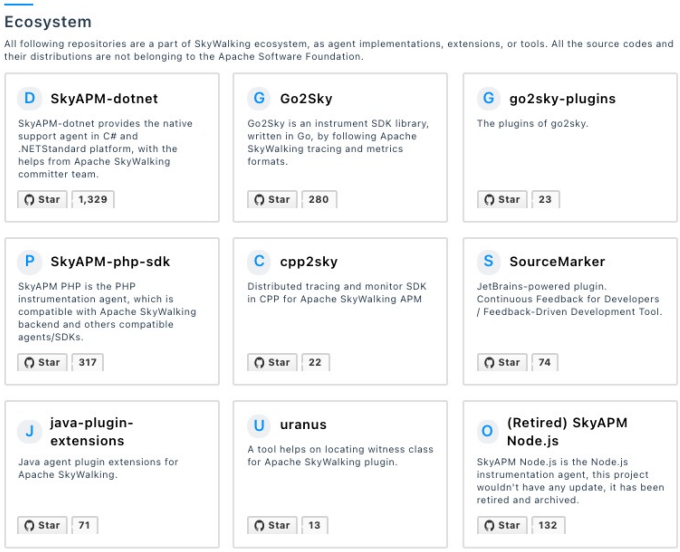
Tags.

Service:	business-zone::projectB
ServiceInstance:	3bb7cf3e14244200b1a8c3b381f2e0be@10.170.0.13
Endpoint:	/projectB/{value}
Span Type:	Entry
Component:	SpringMVC
Peer:	No Peer
Error:	false
url:	http://skywalking-demo-app2.c.skywalking-live-demo.internal:8762/projectB/test
http.method:	GET

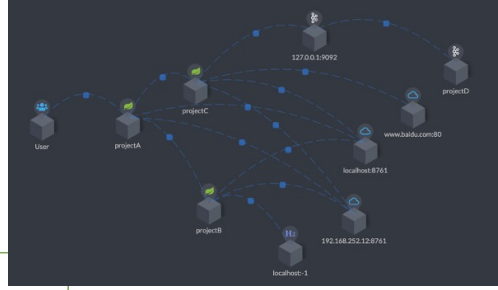


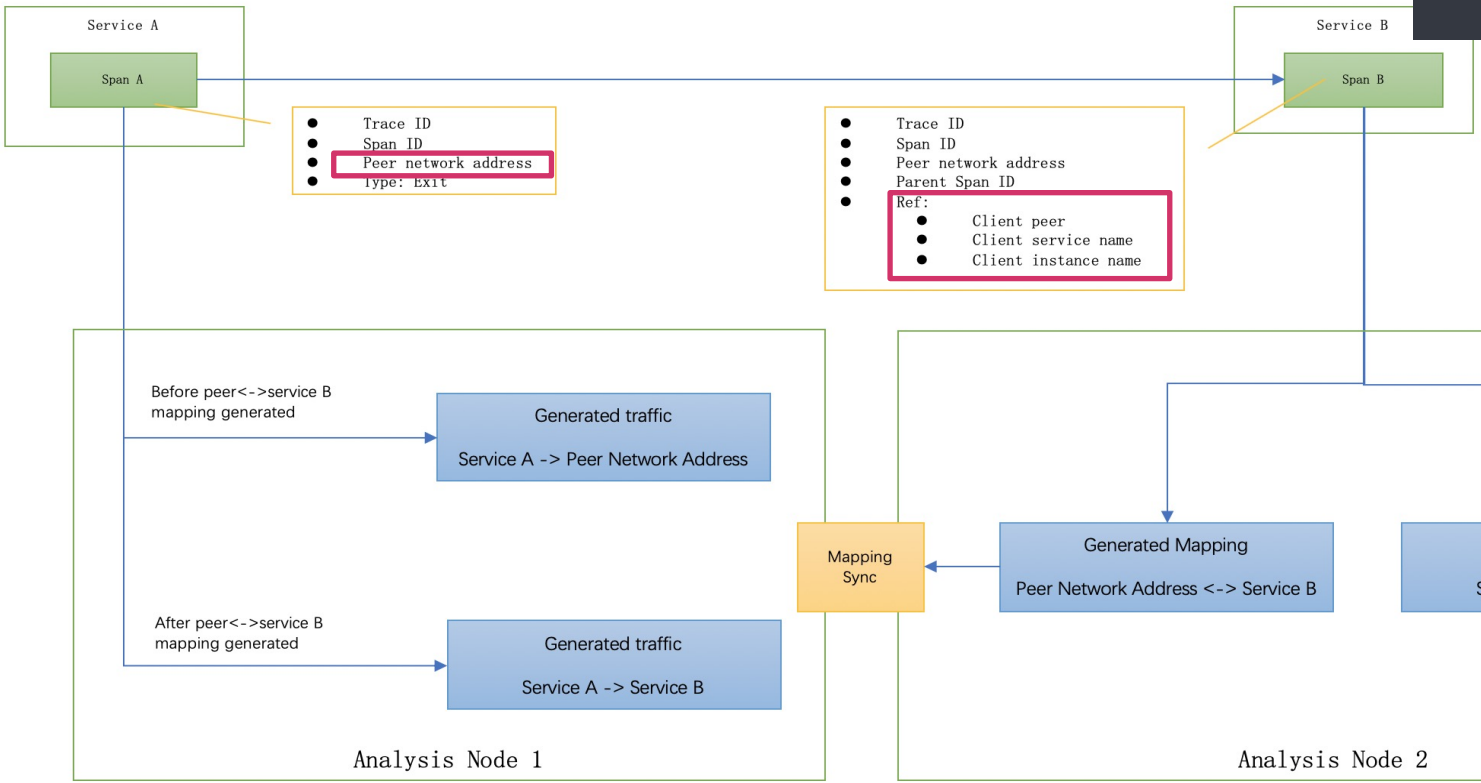
All language agents are maintained by the community

- Java
- Dot Net Core
- PHP
- Python
- NodeJS
- Golang
- Nginx LUA(APISIX & Kong)
- C++, Browser



Why Need Dapper+ Tracing?





OAL – Observability Analysis Language

```
// Declare the metrics.
METRICS_NAME = from(SCOPE.(* | [FIELD][,FIELD ...]))
[.filter(FIELD OP [INT | STRING])]
.FUNCTION([PARAM][, PARAM ...])

// Disable hard code
disable(METRICS_NAME);
```

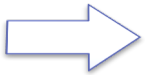
```
// Calculate the p50, p75, p90, p95 and p99 of each Endpoint by 50 ms steps.
endpoint_percentile = from(Endpoint.latency).percentile(10)

// Calculate the percent of response status is true, for each service.
endpoint_success = from(Endpoint.*).filter(status == true).percent()

// Calculate the sum of response code in [404, 500, 503], for each service.
endpoint_abnormal = from(Endpoint.*).filter(responseCode in [404, 500, 503]).count()
```

```
// Calculate the CPM with the GET method for each service.The value is made up with `tagKey:tagValue`.
service_cpm_http_get = from(Service.*).filter(tags contain "http.method:GET").cpm()

// Calculate the CPM with the HTTP method except for the GET method for each service.The value is made up with `tagKey:tagValue`.
service_cpm_http_other = from(Service.*).filter(tags not contain "http.method:GET").cpm()
```

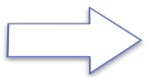
- Avg
- Count
- Percent
- CPM – Calls Per Minute
- Rate
- Histogram
- Apdex
- Percentile

Metrics

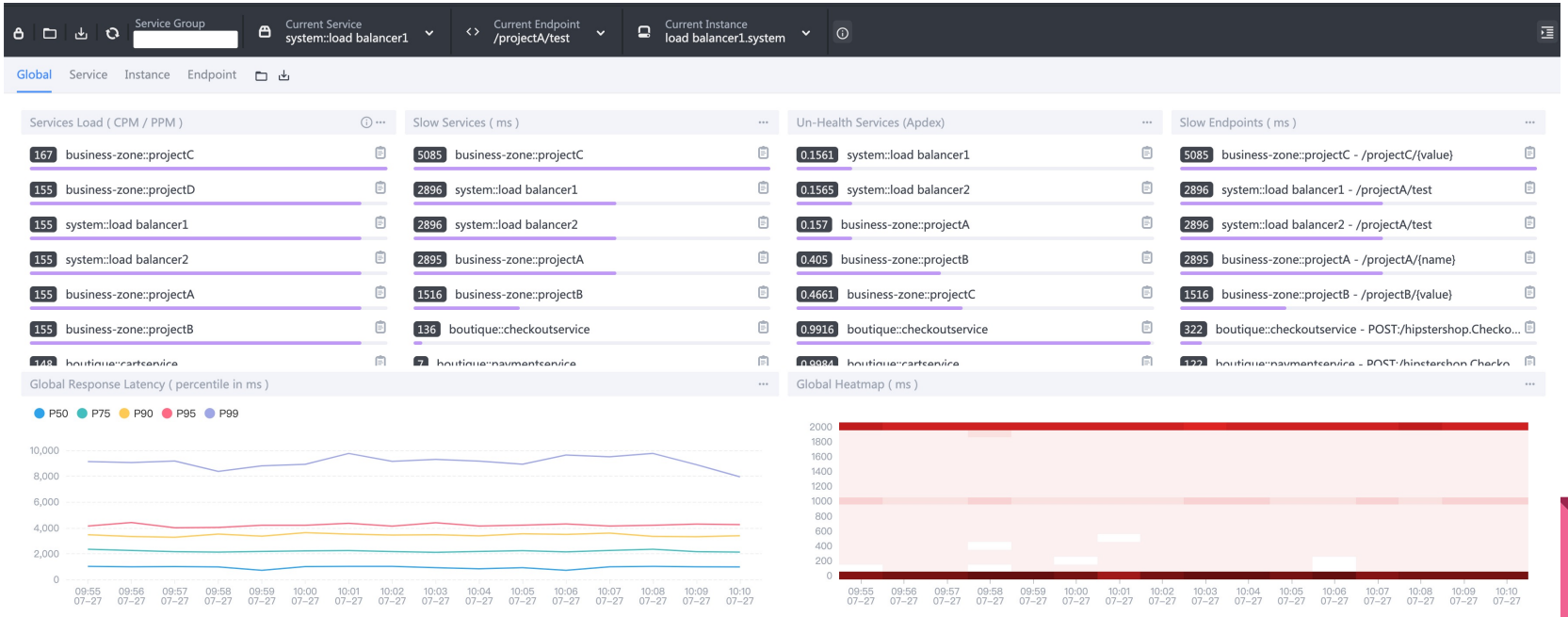
Metrics – Native format & Ecosystem



- Prometheus Fetcher
- Zabbix Format
- OpenCensus gRPC
- Envoy Metrics Service



Metrics Dashboard



MAL Example for self-observability


```yaml
fetcherInterval: PT15S
fetcherTimeout: PT10S
metricsPath: /metrics
staticConfig:
  # targets will be labeled as "instance"
  targets:
    - url: http://localhost:1234
      sslCaFilePath:
  labels:
    service: oap-server
expSuffix: tag({tags -> tags.service = 'oap::' + tags.service}).instance(['service'], ['instance'])
metricPrefix: meter_oap
metricsRules:
  - name: instance_cpu_percentage
    exp: (process_cpu_seconds_total * 100).sum(['service', 'instance']).rate('PT1M')
  - name: instance_jvm_memory_bytes_used
    exp: jvm_memory_bytes_used.sum(['service', 'instance'])
  - name: instance_jvm_young_gc_count
    exp: jvm_gc_collection_seconds_count.tagMatch('gc', 'PS Scavenge|Copy|ParNew|G1 Young Generation').sum(['service', 'instance']).increase('PT1M')
  - name: instance_jvm_young_gc_time
    exp: jvm_gc_collection_seconds_sum.tagMatch('gc', 'PS Scavenge|Copy|ParNew|G1 Young Generation').sum(['service', 'instance']) * 1000
```

MAL – Metrics Analysis Language


```
instance_trace_count{region="us-west",az="az-1"} 100
instance_trace_count{region="us-east",az="az-3"} 20
instance_trace_count{region="asia-north",az="az-1"} 33
```

Filter (by tags and value)

```
instance_trace_count.tagMatch("region", "us-west|asia-north").tagEqual("az", "az-1")
```

```
instance_trace_count.valueGreaterEqual(33)
```

Expression

```
instance_trace_count + 2
```

```
instance_trace_count{region="us-west",az="az-1"} 102 // 100 + 2
instance_trace_count{region="us-east",az="az-3"} 22 // 20 + 2
instance_trace_count{region="asia-north",az="az-1"} 35 // 33 + 2
```

```
instance_trace_count{region="us-west",az="az-1"} 100
instance_trace_count{region="us-east",az="az-3"} 20
instance_trace_count{region="asia-north",az="az-1"} 33
```

```
instance_trace_analysis_error_count{region="us-west",az="az-1"} 20
instance_trace_analysis_error_count{region="asia-north",az="az-1"} 11
```

```
instance_trace_analysis_error_count / instance_trace_count
```

```
{region="us-west",az="az-1"} 0.8  // 20 / 100
{region="asia-north",az="az-1"} 0.3333  // 11 / 33
```

MAL – Metrics Analysis Language


```
instance_trace_count{region="us-west",az="az-1"} 100
instance_trace_count{region="us-east",az="az-3"} 20
instance_trace_count{region="asia-north",az="az-1"} 33
```

Aggregation

```
instance_trace_count.sum(by: ['az'])
```

→

```
instance_trace_count{az="az-1"} 133 // 100 + 33
instance_trace_count{az="az-3"} 20
```

Function

ISO-8601 duration format {@code PnDTnHnMn.nS}

→

- “PT20.345S” – parses as “20.345 seconds”
- “PT15M” – parses as “15 minutes” (where a minute is 60 seconds)
- “PT10H” – parses as “10 hours” (where an hour is 3600 seconds)
- “P2D” – parses as “2 days” (where a day is 24 hours or 86400 seconds)
- “P2DT3H4M” – parses as “2 days, 3 hours and 4 minutes”
- “P-6H3M” – parses as “-6 hours and +3 minutes”
- “-P6H3M” – parses as “-6 hours and -3 minutes”
- “-P-6H+3M” – parses as “+6 hours and -3 minutes”

- increase(Duration)
- rate(Duration)
- irate()
- tag({allTags -> })
- histogram(le: '<the tag name of le>’)
- histogram_percentile([<p scalar>])
- time()

MAL – Metrics Analysis Language

Down Sampling Operation

MAL should instruct meter-system on how to downsample for metrics. It doesn't only refer to aggregate raw samples to minute level, but also expresses data from minute in higher levels, such as hour and day.

> last_server_state_sync_time_in_seconds.tagEqual('production', 'catalog').downsampling(LATEST)

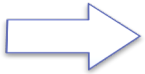
Metric level function

- servcie([svc_label1, svc_label2...])
- instance([svc_label1, svc_label2...], [ins_label1, ins_label2...])
- endpoint([svc_label1, svc_label2...], [ep_label1, ep_label2...])

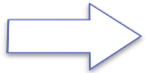
Logging

Streaming log process

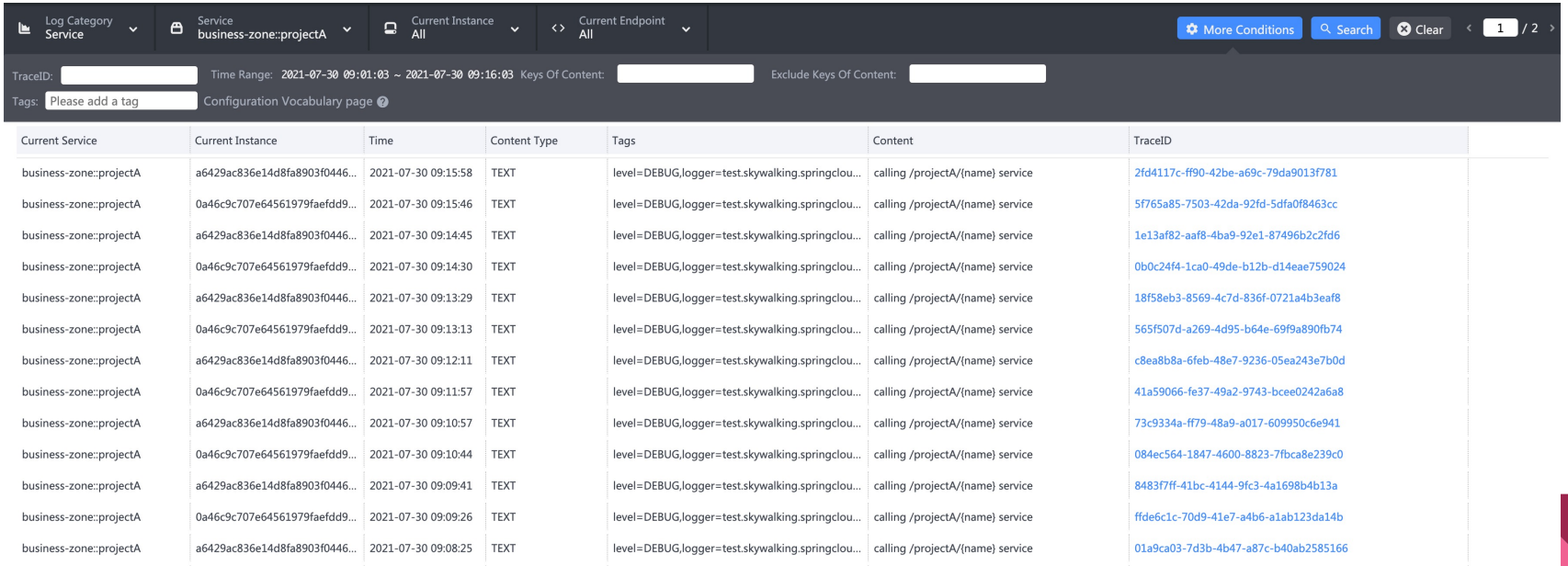
Multiple Agents



- Fluentd
- Fluent-bit
- Elastic FileBeat
- Envoy Access Log



Collected Logs With Trace Collaboration



Current Service	Current Instance	Time	Content Type	Tags	Content	TraceID
business-zone::projectA	a6429ac836e14d8fa8903f0446...	2021-07-30 09:15:58	TEXT	level=DEBUG,logger=test.skywalking.springclou...	calling /projectA/{name} service	2fd4117c-ff90-42be-a69c-79da9013f781
business-zone::projectA	0a46c9c707e64561979faefdd9...	2021-07-30 09:15:46	TEXT	level=DEBUG,logger=test.skywalking.springclou...	calling /projectA/{name} service	5f765a85-7503-42da-92fd-5dfa0f8463cc
business-zone::projectA	a6429ac836e14d8fa8903f0446...	2021-07-30 09:14:45	TEXT	level=DEBUG,logger=test.skywalking.springclou...	calling /projectA/{name} service	1e13af82-aaf8-4ba9-92e1-87496b2c2fd6
business-zone::projectA	0a46c9c707e64561979faefdd9...	2021-07-30 09:14:30	TEXT	level=DEBUG,logger=test.skywalking.springclou...	calling /projectA/{name} service	0b0c24f4-1ca0-49de-b12b-d14eae759024
business-zone::projectA	a6429ac836e14d8fa8903f0446...	2021-07-30 09:13:29	TEXT	level=DEBUG,logger=test.skywalking.springclou...	calling /projectA/{name} service	18f58eb3-8569-4c7d-836f-0721a4b3eaf8
business-zone::projectA	0a46c9c707e64561979faefdd9...	2021-07-30 09:13:13	TEXT	level=DEBUG,logger=test.skywalking.springclou...	calling /projectA/{name} service	565f507d-a269-4d95-b64e-69f9a890fb74
business-zone::projectA	a6429ac836e14d8fa8903f0446...	2021-07-30 09:12:11	TEXT	level=DEBUG,logger=test.skywalking.springclou...	calling /projectA/{name} service	c8ea8b8a-6feb-48e7-9236-05ea243e7b0d
business-zone::projectA	0a46c9c707e64561979faefdd9...	2021-07-30 09:11:57	TEXT	level=DEBUG,logger=test.skywalking.springclou...	calling /projectA/{name} service	41a59066-fe37-49a2-9743-bcee0242a6a8
business-zone::projectA	a6429ac836e14d8fa8903f0446...	2021-07-30 09:10:57	TEXT	level=DEBUG,logger=test.skywalking.springclou...	calling /projectA/{name} service	73c9334a-ff79-48a9-a017-609950c6e941
business-zone::projectA	0a46c9c707e64561979faefdd9...	2021-07-30 09:10:44	TEXT	level=DEBUG,logger=test.skywalking.springclou...	calling /projectA/{name} service	084ec564-1847-4600-8823-7fbca8e239c0
business-zone::projectA	a6429ac836e14d8fa8903f0446...	2021-07-30 09:09:41	TEXT	level=DEBUG,logger=test.skywalking.springclou...	calling /projectA/{name} service	8483f7ff-41bc-4144-9fc3-4a1698b4b13a
business-zone::projectA	0a46c9c707e64561979faefdd9...	2021-07-30 09:09:26	TEXT	level=DEBUG,logger=test.skywalking.springclou...	calling /projectA/{name} service	ffde6c1c-70d9-41e7-a4b6-a1ab123da14b
business-zone::projectA	a6429ac836e14d8fa8903f0446...	2021-07-30 09:08:25	TEXT	level=DEBUG,logger=test.skywalking.springclou...	calling /projectA/{name} service	01a9ca03-7d3b-4b47-a87c-b40ab2585166

Parse logging from Text


```
filter {
    text {
        abortOnFailure true // this is optional because it's default behaviour
        // this is just a demo pattern
        regexp "(?<timestamp>\\d{8}) (?<thread>\\w+) (?<level>\\w+) (?<traceId>\\w+) (?<msg>.+)"
    }
    extractor {
        tag level: parsed.level
        // we add a tag called `level` and its value is parsed.level, captured from the regexp above
        traceId parsed.traceId
        // we also extract the trace id from the parsed result, which will be used to associate the log with the trace
    }
    // ...
}
```

Generate Metrics from Logs


```
filter {
    // ...
    extractor {
        service parsed.serviceName
        metrics {
            name "log_count"
            timestamp parsed.timestamp
            labels level: parsed.level, service: parsed.service, instance: parsed.instance
            value 1
        }
        metrics {
            name "http_response_time"
            timestamp parsed.timestamp
            labels status_code: parsed.statusCode, service: parsed.service, instance: parsed.instance
            value parsed.duration
        }
    }
    // ...
}
```

```
# ... other configurations of MAL

metrics:
  - name: log_count_debug
    exp: log_count.tagEqual('level', 'DEBUG').sum(['service', 'instance']).increase('PT1M')
  - name: log_count_error
    exp: log_count.tagEqual('level', 'ERROR').sum(['service', 'instance']).increase('PT1M')
```

```
metrics:
  - name: response_time_percentile
    exp: http_response_time.sum(['le', 'service', 'instance']).increase('PT5M').histogram().histogram_percentile([50,70,90,99])
```

Sampling Logs


```
filter {
    // ... parser

    sink {
        sampler {
            if (parsed.service == "ImportantApp") {
                rateLimit("ImportantAppSampler") {
                    qps 30  // samples 30 pieces of logs every second for service "ImportantApp"
                }
            } else {
                rateLimit("OtherSampler") {
                    qps 3   // samples 3 pieces of logs every second for other services than "ImportantApp"
                }
            }
        }
    }
}
```

Metrics-Focused, Log Dropper

```
filter { // filter A: this is for persistence
    // ... parser

    sink {
        sampler {
            // .. sampler configs
        }
    }
}
filter { // filter B:
    // ... extractors to generate many metrics
    extractors {
        metrics {
            // ... metrics
        }
    }
    sink {
        dropper {} // drop all logs because they have been saved in "filter A" above.
    }
}
```

Enforcer
- sample logs with specific conditions

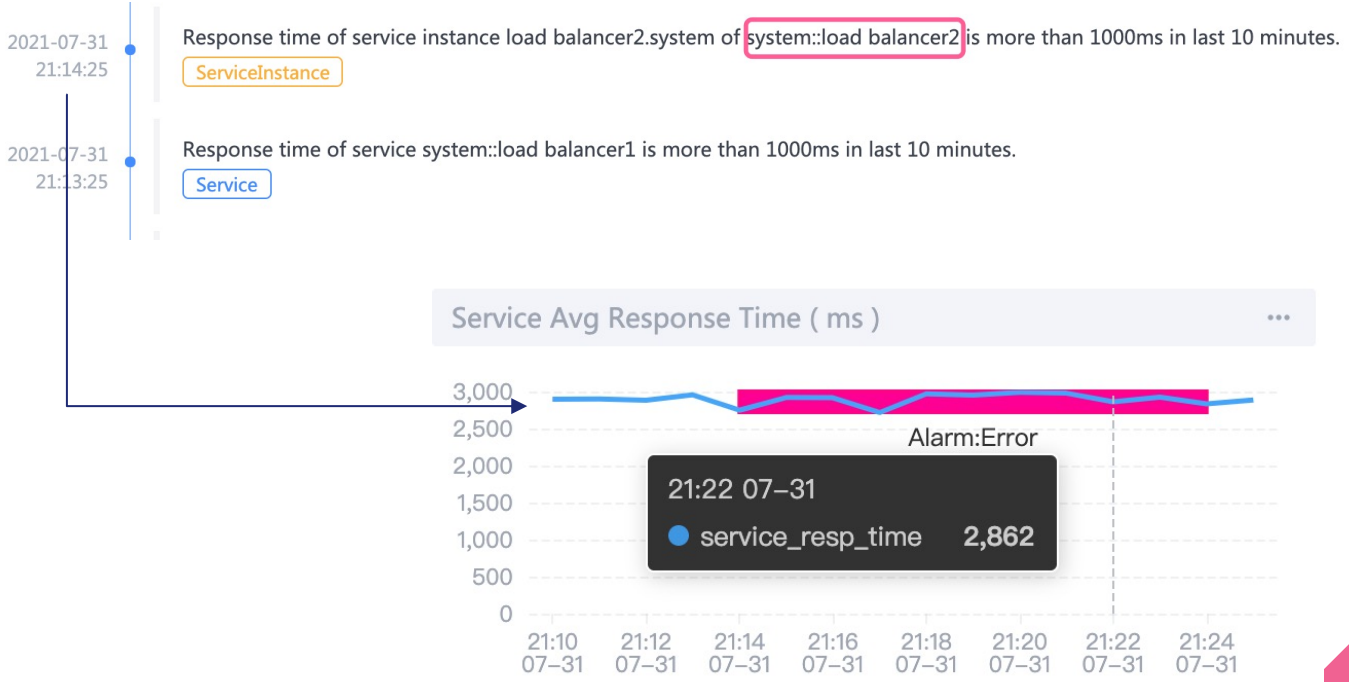
```
filter {
    // ... parser

    sink {
        sampler {
            // ... sampler configs
        }
        if (parserd.level == "ERROR" || parsed.userId == "TestingUserId") {
            enforcer {
            }
        }
    }
}
```

// sample error logs or testing users' logs
// (userId == "TestingUserId")
// even if the sampling strategy is configured

Event

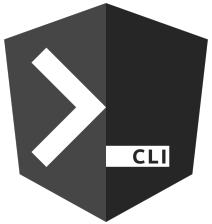
A trigger to chaos.

Event Timeline and Dashboard Highlight



Event Detection

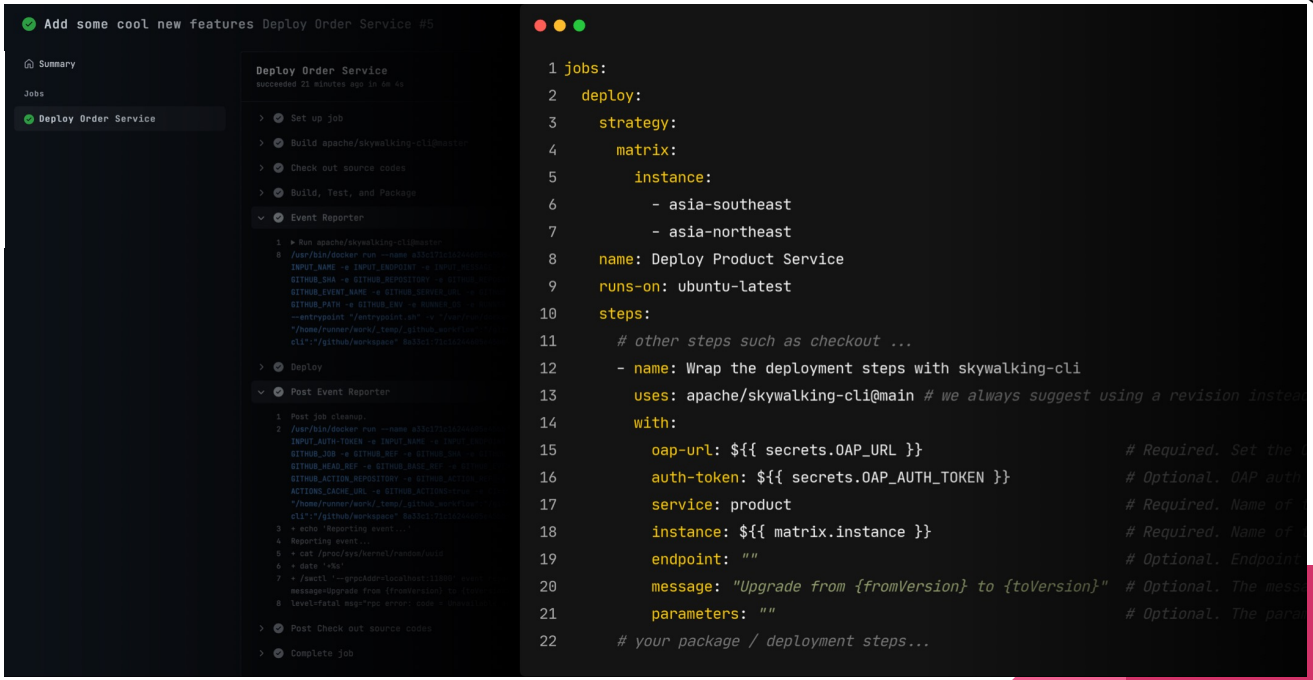
- JVM Start up
- JVM Shut down
- JVM Options



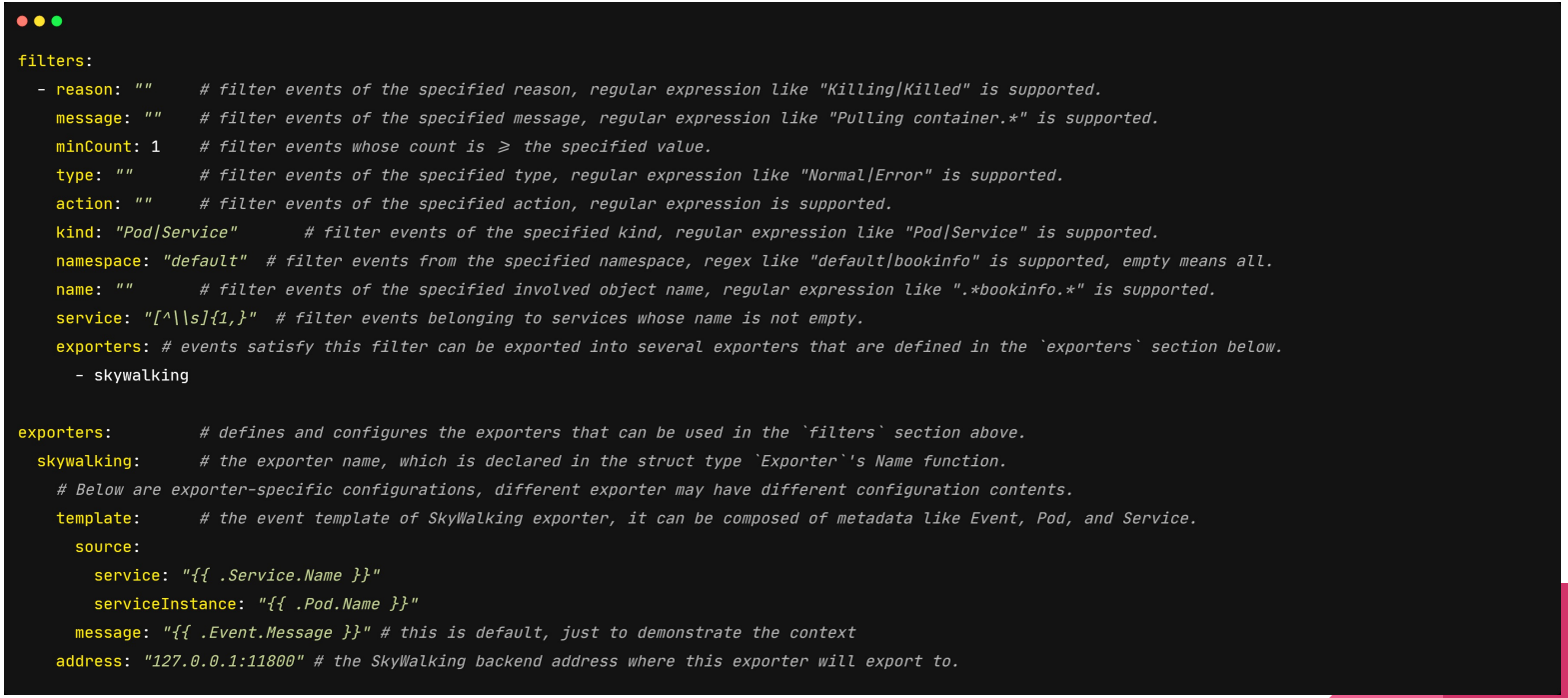

```
swctl event report --grpcAddr=localhost:11800 \
                   --uuid=$(uuidgen) \
                   --name=Upgrade \
                   --service=order-service \
                   --instance=order-service-instance-1 \
                   --endpoint='/purchase' \
                   --message='Upgrade from {fromVersion} to {toVersion}' \
                   --startTime=$(date +%s)000 \
                   --endTime=$(date +%s)000 \
                   fromVersion=v1 toVersion=v2
```

CD Integration - GitHub Action

GitHub Actions

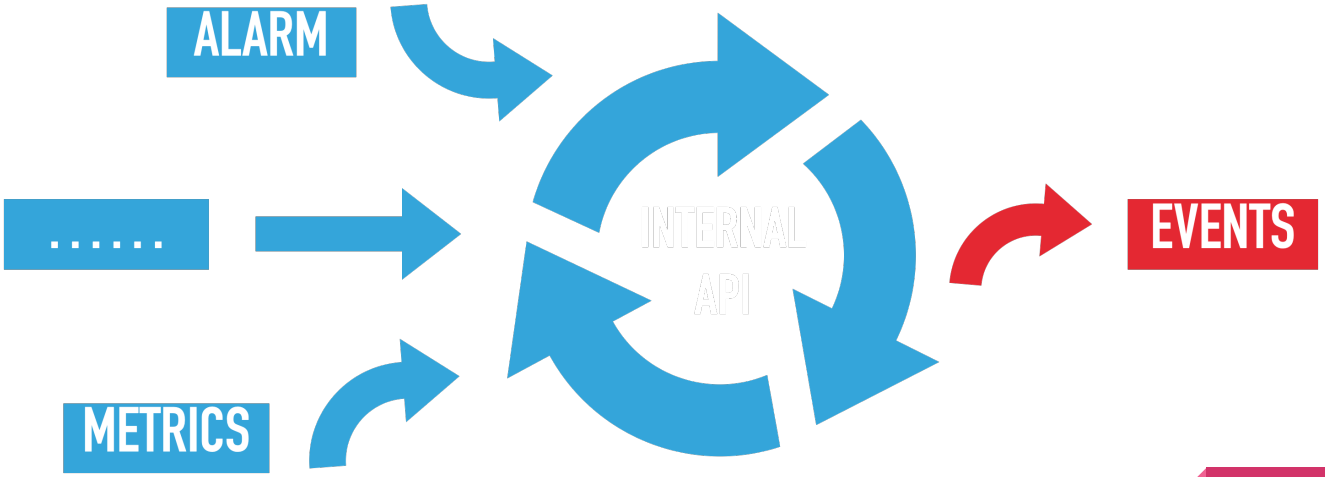


Kubernetes Event Exporter



```yaml
filters:
  - reason: ""     # filter events of the specified reason, regular expression like "Killing|Killed" is supported.
    message: ""    # filter events of the specified message, regular expression like "Pulling container.*" is supported.
    minCount: 1    # filter events whose count is ≥ the specified value.
    type: ""       # filter events of the specified type, regular expression like "Normal|Error" is supported.
    action: ""     # filter events of the specified action, regular expression is supported.
    kind: "Pod|Service"       # filter events of the specified kind, regular expression like "Pod|Service" is supported.
    namespace: "default"  # filter events from the specified namespace, regex like "default|bookinfo" is supported, empty means all.
    name: ""       # filter events of the specified involved object name, regular expression like ".*bookinfo.*" is supported.
    service: "[^\\s]{1,}"  # filter events belonging to services whose name is not empty.
    exporters: # events satisfy this filter can be exported into several exporters that are defined in the `exporters` section below.
      - skywalking

exporters:         # defines and configures the exporters that can be used in the `filters` section above.
  skywalking:      # the exporter name, which is declared in the struct type `Exporter`'s Name function.
    # Below are exporter-specific configurations, different exporter may have different configuration contents.
    template:      # the event template of SkyWalking exporter, it can be composed of metadata like Event, Pod, and Service.
      source:
        service: "{{ .Service.Name }}"
        serviceInstance: "{{ .Pod.Name }}"
      message: "{{ .Event.Message }}" # this is default, just to demonstrate the context
    address: "127.0.0.1:11800" # the SkyWalking backend address where this exporter will export to.
```

SkyWalking Internal Event Detecting



Alarm for event

- Configure alerts for events as if they were metrics
  - Once “Crash” event occurs
  - Count (“Unhealthy”) > 5 within 5 minutes

```
unhealthy_event_rule:
  metrics-name: Unhealthy
  threshold: 5
  op: ">"
  period: 10
  count: 1
  message: Service instance has been
           unhealthy for 10 minutes
```

How to make sure Quality

APM was used to always commercial product only

Community Driven

- 500+ Code Contributors (counted by GitHub)
- Different groups(SIG) for different fields
- Tracing specific testing framework
- E2E testing framework

Q & A

https://twitter.com/ASFSkyWalking

https://twitter.com/wusheng1108