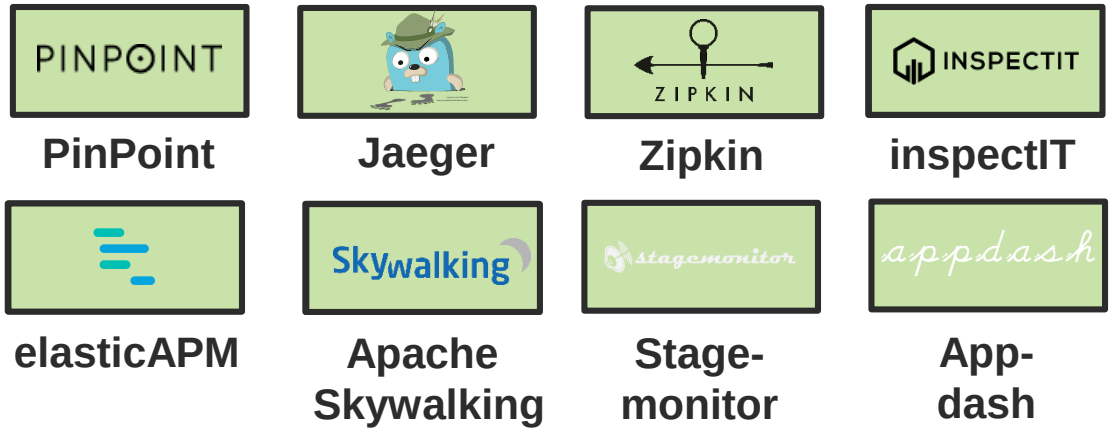
Open Source Application Performance Monitoring (APM)

Ein Überblick über APM Tools und Standards für Java-basierte Enterprise-Anwendungen

Dr. Andreas Brunnert
RETIT GmbH

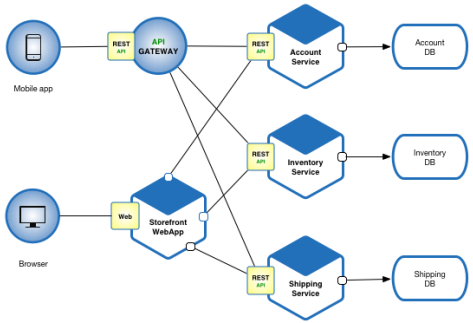
Motivation

The amount of open source APM tools has grown dramatically in the last four years:



Motivation

Complexity increase in modern software systems



Services might need to interact with each other in ways that might not be obvious at the time of development or deployment.

Growing importance of IT for more business models

Downtimes or bad software performance have a direct impact on revenue.

Development of tracing standards

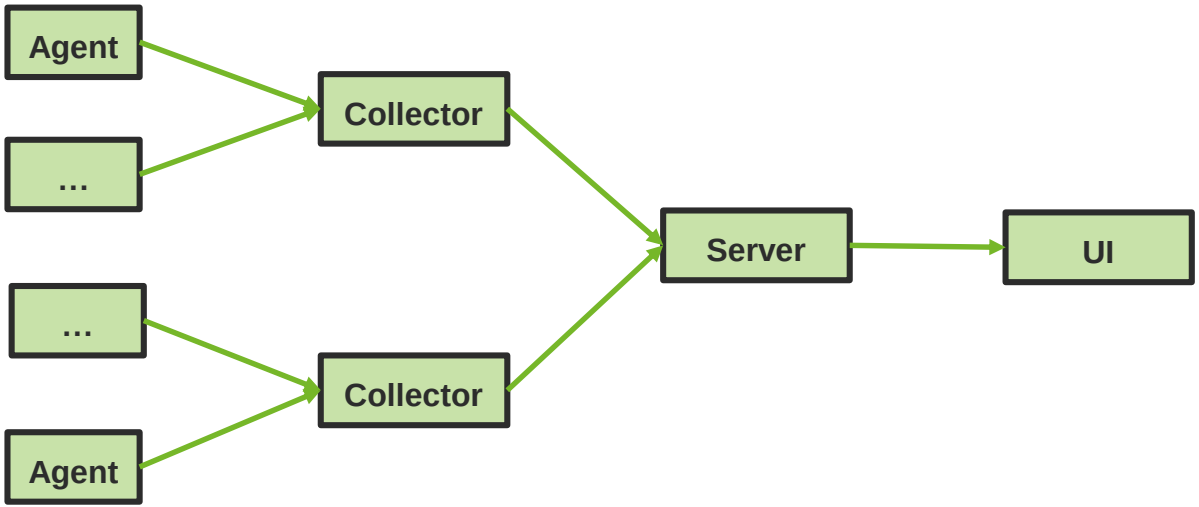
OPENTRACING

OpenCensus

Which allow to easily exchange the tracing tool in use. Furthermore, they reduce the effort for each vendor.

Context

Anatomy of an APM Solution

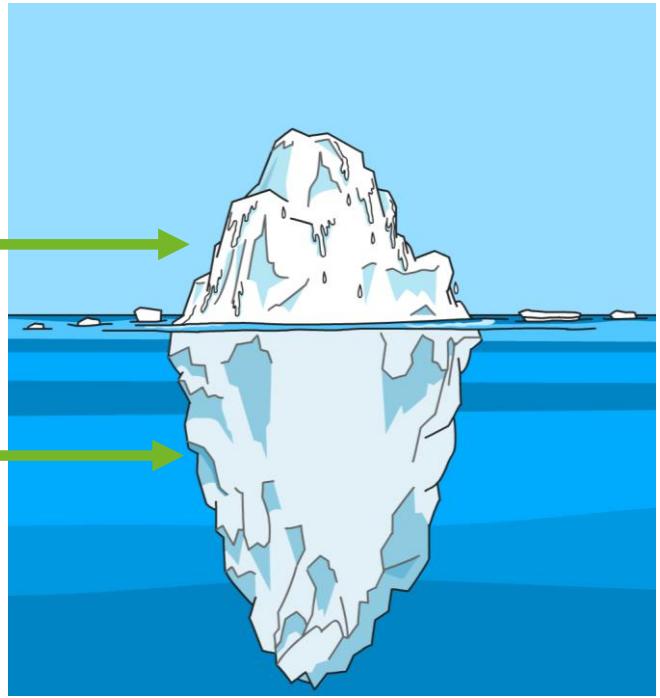


Context

Code and Effort distribution of an APM Solution

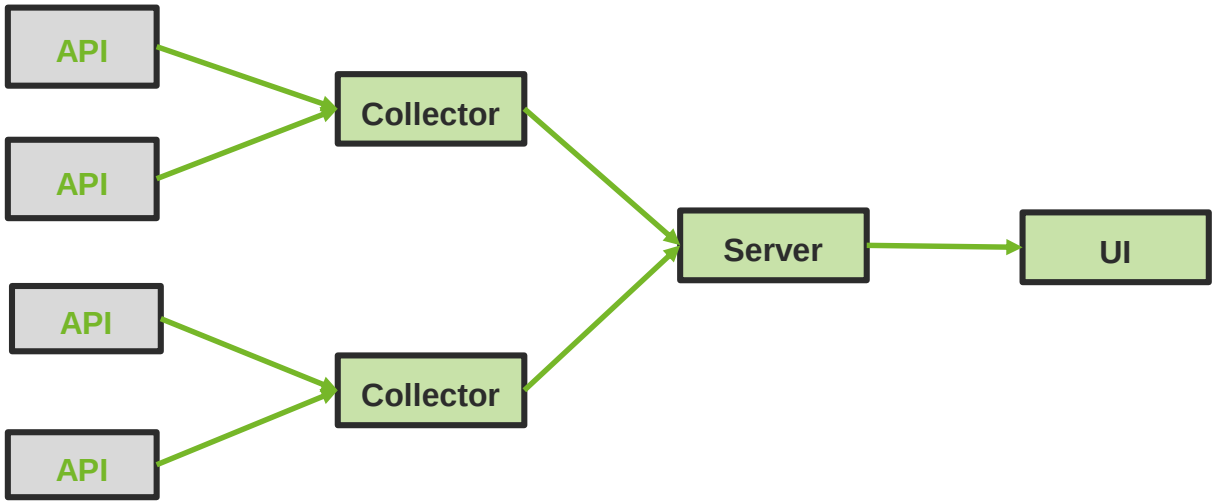
UI + Server + Collectors

Agents



Context

Scope of many open source APM solutions



Context

- Some tools build upon the same concepts or even fork each other:
  - https://research.google.com/pubs/pub36356.html
  - Basis for: Pinpoint, Jaeger and Zipkin
  - Zipkin is again the basis for Jaeger

Context

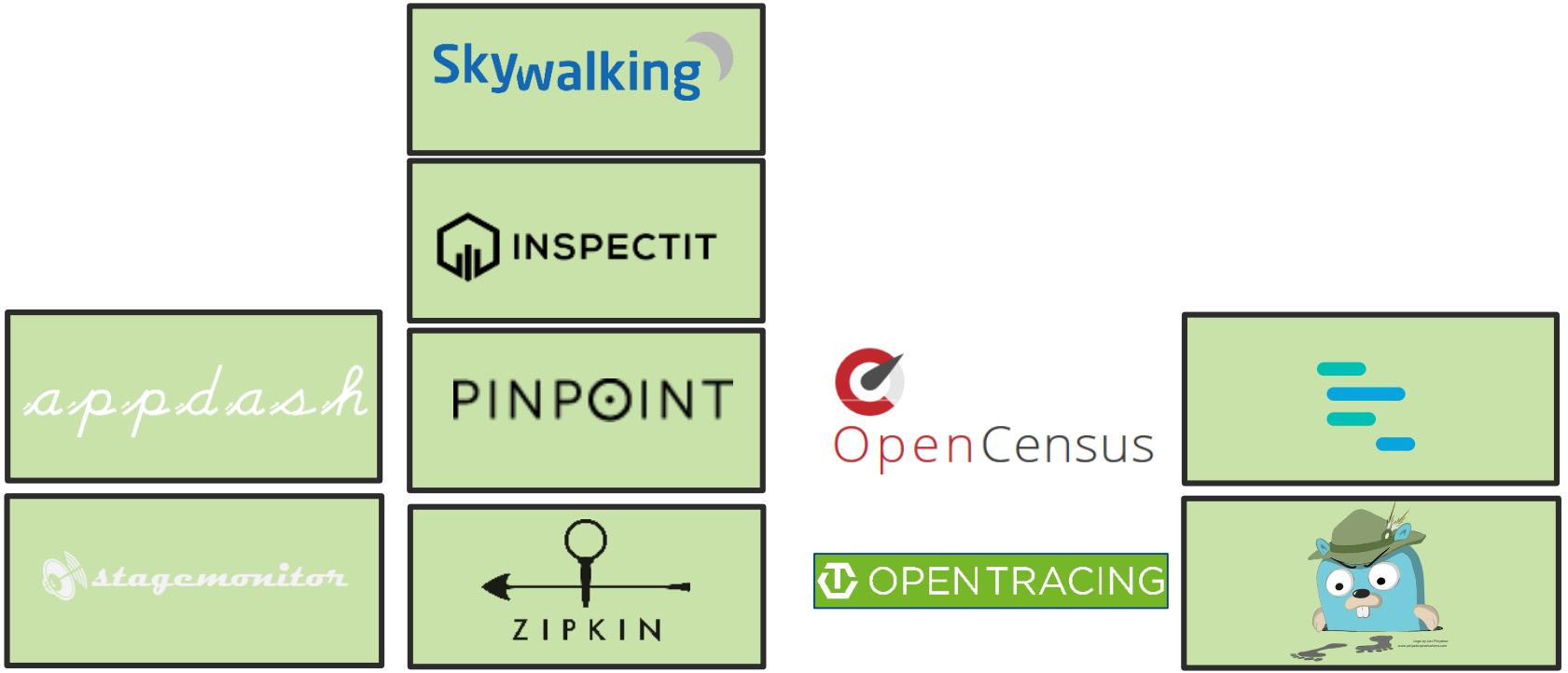
But how do these open source APM tools compare?

- Age
- Popularity
- Supported Technologies
- Standards Support

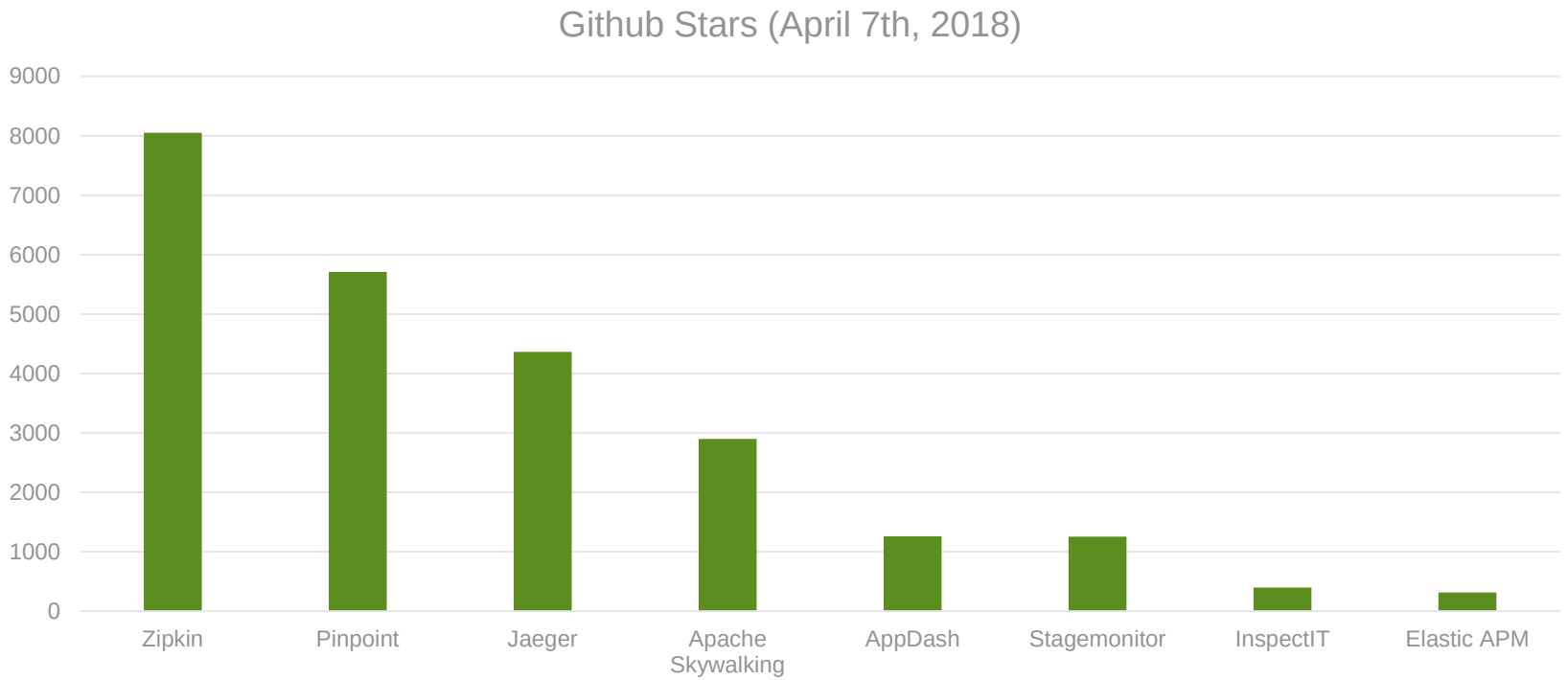
- Not in presentation (will be covered in later blog articles):
  - Setup – Effort
  - Integration Capabilities with other tools
  - License

What are reasons for a closed source alternative?

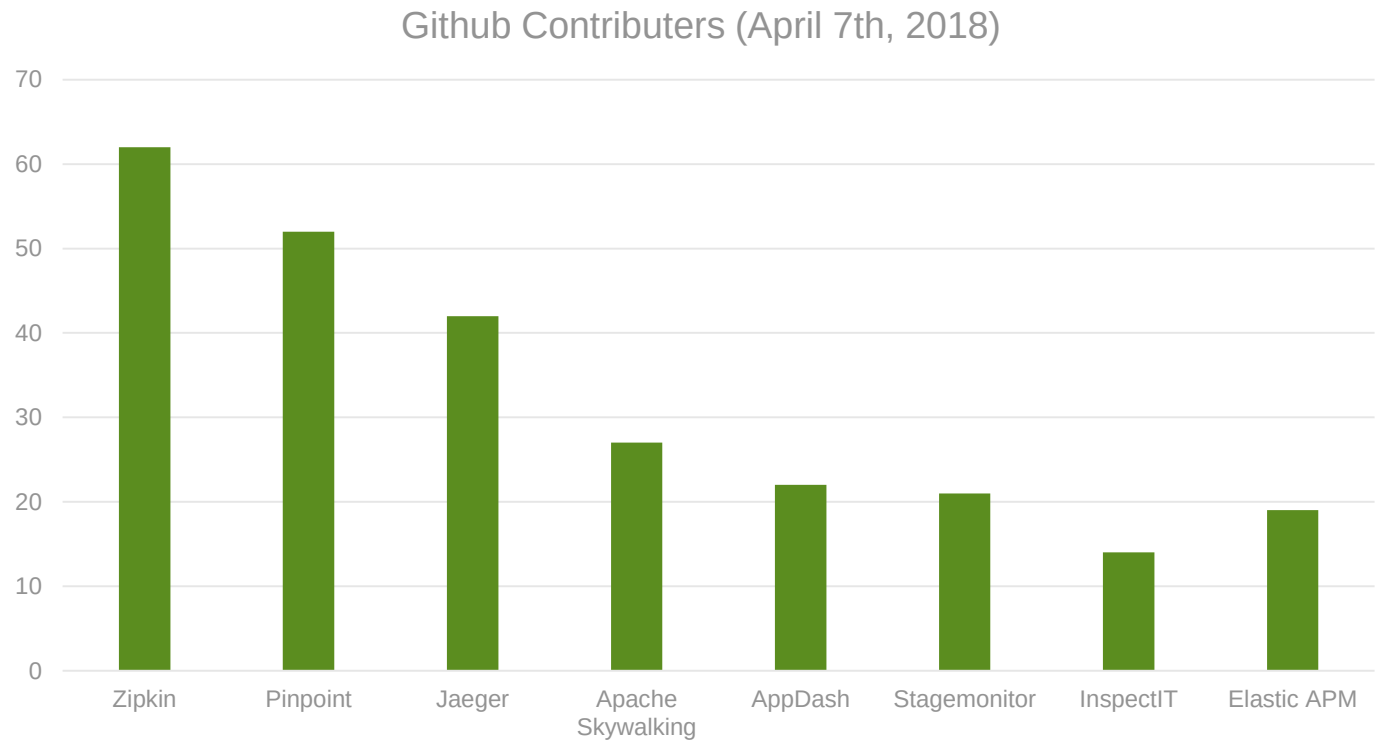
A brief timeline of tool availability (since 2014)



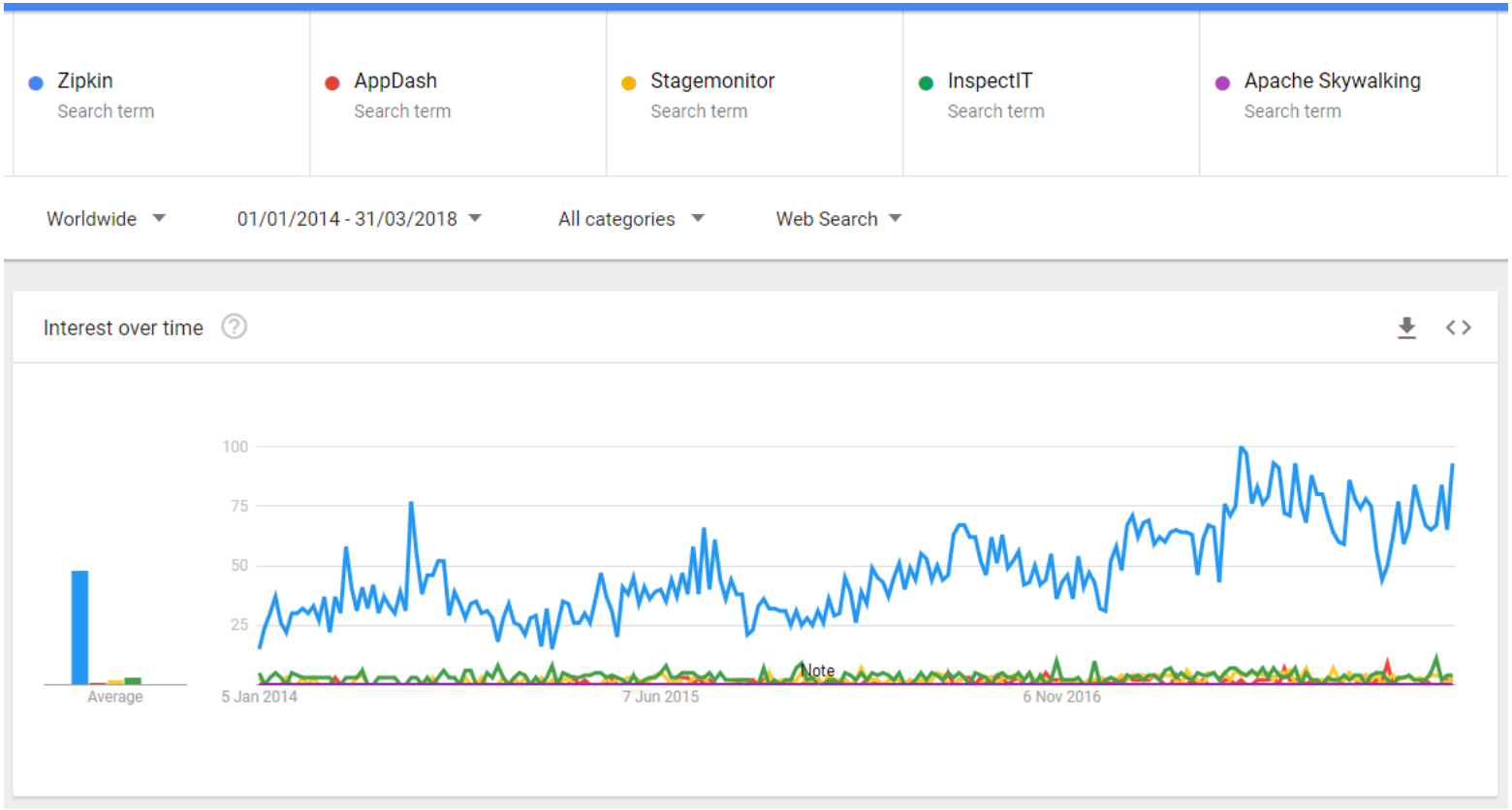
A ranking of github stars



A ranking of github contributes



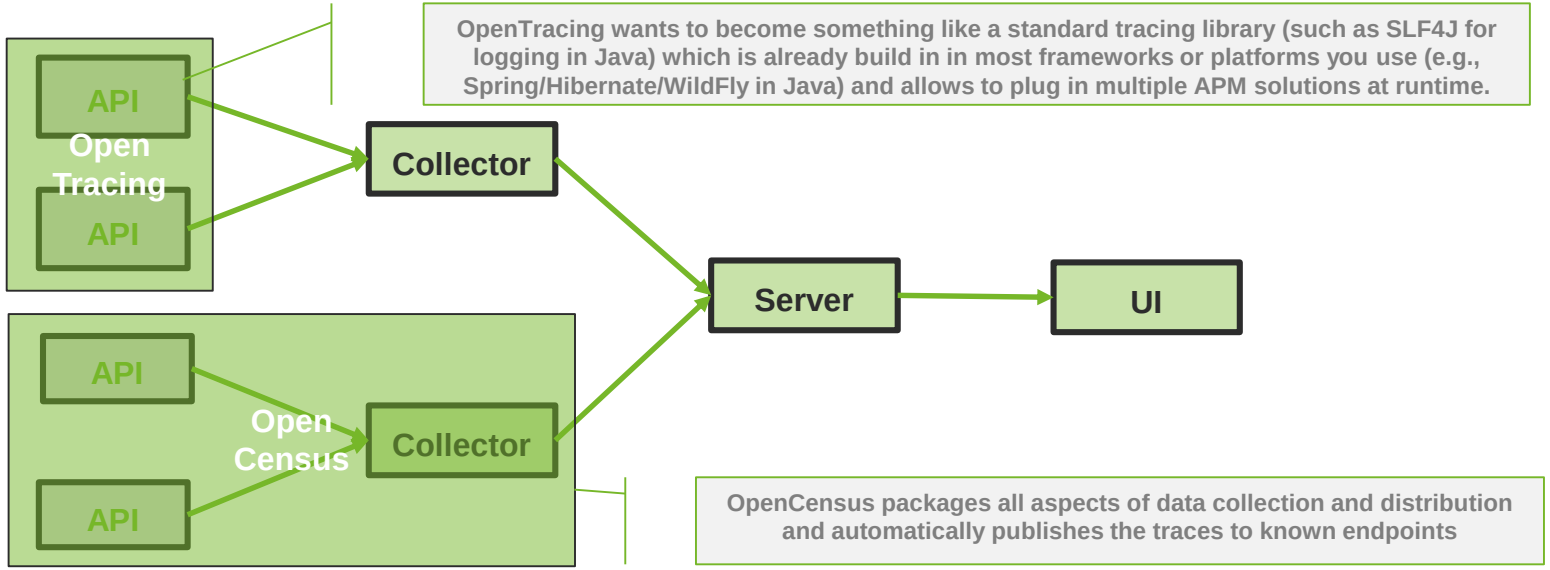
Google Trends Analysis



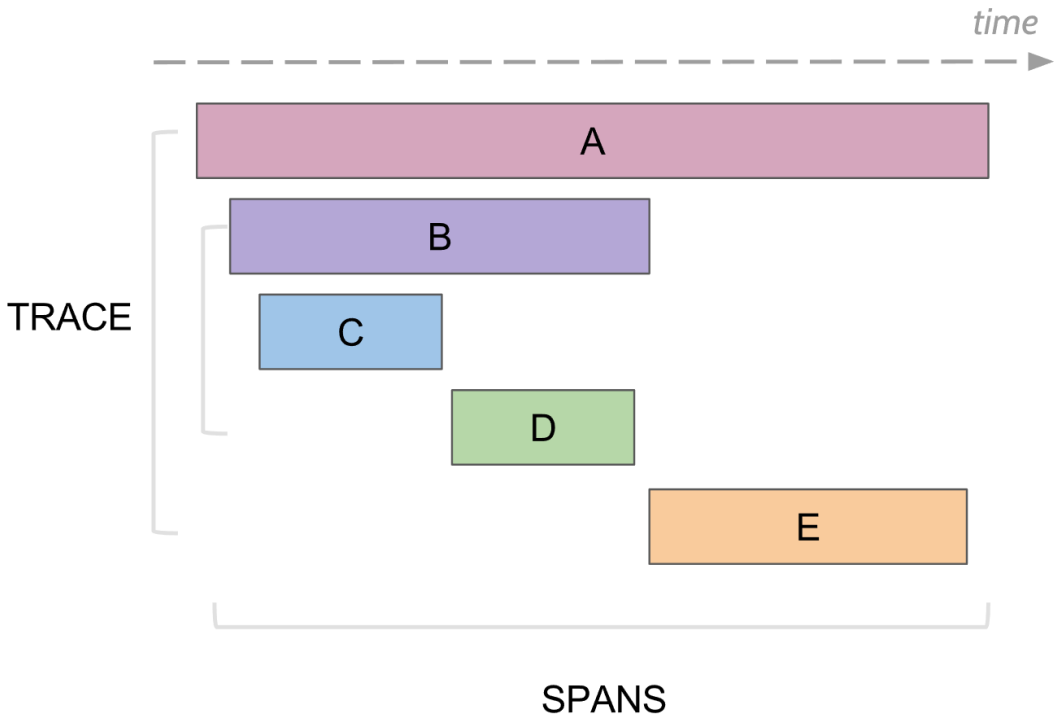
Open Source “Standards”

Open Source “Standards”

Scope of OpenTracing vs. OpenCensus (Simplified)

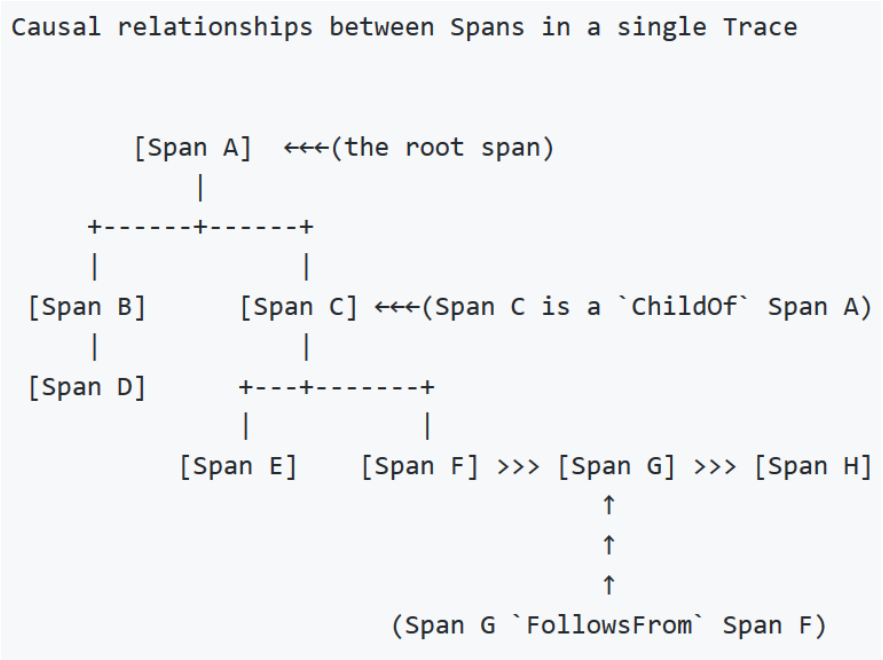


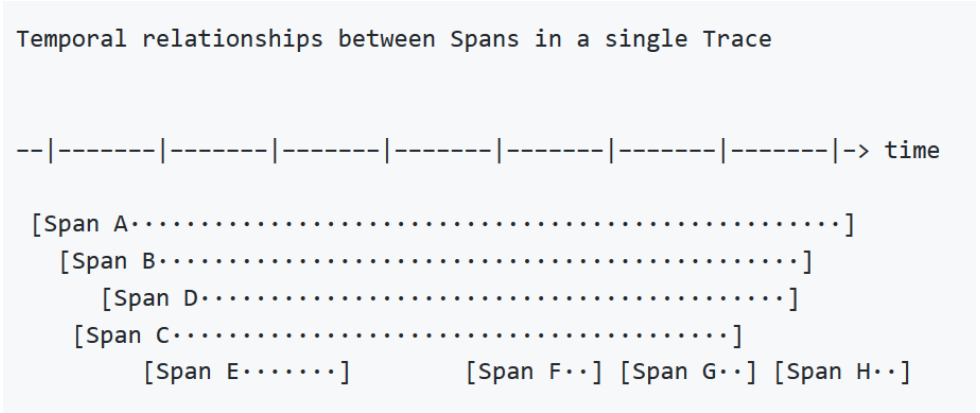
Open Source “Standards” - OpenTracing



Source: https://www.jaegertracing.io/docs/architecture/

Open Source “Standards” - OpenTracing





Source: https://github.com/opentracing/specification/blob/master/specification.md

Open Source “Standards” - OpenTracing

```
import com.uber.jaeger.Configuration;
import io.opentracing.Span;
import io.opentracing.util.GlobalTracer;

...

GlobalTracer.register(
   new Configuration(
      "your_service_name",
      new Configuration.SamplerConfiguration("const", 1), new Configuration.ReporterConfiguration(false, "localhost", null, 1000, 10000)
   ).getTracer());

...

try (Span parent = GlobalTracer.get()
         .buildSpan("hello")
         .start()) {
   try (Span child = GlobalTracer.get()
         .buildSpan("world")
         .asChildOf(parent)
         .start()) {
   }
}
```

You only need to do this once

For each individual span

Source: http://opentracing.io/documentation/pages/quick-start.html

Open Source “Standards” - OpenCensus

LANGUAGES

LANGUAGE	TRACING	STATS
C++	Supported	Supported
Erlang	Supported	Supported
Go	Supported	Supported
Java (JVM, OpenJDK, Android)	Supported	Supported
PHP	Supported	Planned
Python	Supported	In Progress
Ruby	Supported	Planned

Source: **https://opencensus.io/roadmap/index.html**

Open Source “Standards” - OpenCensus

EXPORTERS

BACKEND	GO	JAVA	ERLANG	C++	PYTHON
SignalFX	No (open issue)	Yes	No	No	No
Prometheus	Yes	Yes	Yes	No	No
Jaeger	Yes	No	No	No	No
Stackdriver	Yes	Yes	Yes (trace only)	No	Yes
Zipkin	Yes	Yes	Yes	No	No

Source: https://opencensus.io/roadmap/index.html

Open Source “Standards” - OpenCensus

TraceZ Summary

Span Name	Running	Latency Samples								Error Samples
		[>0us]	[>10us]	[>100us]	[>1ms]	[>10ms]	[>100ms]	[>1s]	[>10s]	[>100s]	
HttpServer/tracecconfigz	0	0	0	0	0	0	0	0	0	0	0
HttpServer/tracez	1	0	0	0	0	1	0	0	0	0	0
Recv.helloworld.Greeter.SayHello	0	0	10	10	10	7	1	0	0	0	0

Span Name: Recv.helloworld.Greeter.SayHello

Finished Requests 10

```
When                            Elapsed(s)
-------------------------------------------
2017/12/02-21:37:57.472000      0.002787     TraceId: 27845009b7298988e90207e89802c8ce SpanId: 274398e41b4a06d5 ParentSpanId: 1b8cd50723d30705
          21:37:57.472110         .   110 ... Received message_id=0 message_size=0
          21:37:57.474761         .  2651 ... Sent message_id=0 message_size=60414
                                               Status{canonicalCode=OK, description=null}
                                               Attributes:{}
2017/12/02-21:37:32.335000      0.001002     TraceId: 0e0f2910418068502dafef9f7610483c SpanId: fca7b1ed3ff53a0f ParentSpanId: 86d5e57bd86c4611
          21:37:32.335268         .   268 ... Received message_id=0 message_size=0
          21:37:32.335957         .   689 ... Sent message_id=0 message_size=49331
                                               Status{canonicalCode=OK, description=null}
                                               Attributes:{}
2017/12/02-21:37:21.259000      0.005406     TraceId: 460eb57a99e97c46809fbe3e0780d5d6 SpanId: eb54811c23eac071 ParentSpanId: 07294967d63f43b0
          21:37:21.259083         .    83 ... Received message_id=0 message_size=0
          21:37:21.264380         .  5296 ... Sent message_id=0 message_size=61007
                                               Status{canonicalCode=OK, description=null}
                                               Attributes:{}
2017/12/02-21:27:45.180000      0.006234     TraceId: 67d266124964df9dce976573ab1bcafc SpanId: 520aa99cbbc3286c ParentSpanId: 18e9b2ad811f8761
          21:27:45.180117         .   117 ... Received message_id=0 message_size=0
          21:27:45.186216         .  6099 ... Sent message_id=0 message_size=47647
                                               Status{canonicalCode=OK, description=null}
                                               Attributes:{}
2017/12/02-21:27:16.932000      0.002692     TraceId: 8ed6ded04e1cd74ee8eaf95424e8b760 SpanId: 5c6d9017c179866a ParentSpanId: ecd40cbff0ac0a3c
          21:27:16.932318         .   318 ... Received message_id=0 message_size=0
          21:27:16.934676         .  2358 ... Sent message_id=0 message_size=64123
                                               Status{canonicalCode=OK, description=null}
                                               Attributes:{}
2017/12/02-21:21:26.941000      0.003771     TraceId: c9a3aca3de139a9d935b12f2b1737929 SpanId: c0a5ce510b765ae0 ParentSpanId: 5d36a77a23213219
          21:21:26.941081         .    81 ... Received message_id=0 message_size=0
          21:21:26.944761         .  3679 ... Sent message_id=0 message_size=62305
                                               Status{canonicalCode=OK, description=null}
                                               Attributes:{}
2017/12/02-21:03:51.139000      0.004916     TraceId: b8a3dbd7e62835c2f9cadc4291094c7c SpanId: cfd1e0b6316c10c2 ParentSpanId: 829f09e82e65324d
          21:03:51.139089         .    89 ... Received message_id=0 message_size=0
          21:03:51.143896         .  4807 ... Sent message_id=0 message_size=59838
                                               Status{canonicalCode=OK, description=null}
                                               Attributes:{}
2017/12/02-21:00:09.285000      0.003660     TraceId: 6e2ab7e63a200c4d5acd0080debeba43 SpanId: 45a2e81616b92d70 ParentSpanId: 9625cc4b05e1f2b8
          21:00:09.285061         .    61 ... Received message_id=0 message_size=0
          21:00:09.288649         .  3587 ... Sent message_id=0 message_size=35810
                                               Status{canonicalCode=OK, description=null}
                                               Attributes:{}
2017/12/02-20:48:57.628000      0.004779     TraceId: 94468847c8d7c35f4c84610aa7a1b5330 SpanId: c91eb86726a22dc1 ParentSpanId: a4fca6b1fb3bbe98
          20:48:57.632695         .  4695 ... Received message_id=0 message_size=0
          20:48:57.632768         .    72 ... Sent message_id=0 message_size=8554
                                               Status{canonicalCode=OK, description=null}
                                               Attributes:{}
2017/12/02-20:45:11.804000      0.003995     TraceId: 3022b79c2e635d8745300ceea4130300 SpanId: 8ff9e8a6af70fcaf ParentSpanId: 445b763c74a7da6c
          20:45:11.804057         .    57 ... Received message_id=0 message_size=0
          20:45:11.807983         .  3926 ... Sent message_id=0 message_size=46489
                                               Status{canonicalCode=OK, description=null}
                                               Attributes:{}
```

Source: https://opencensus.io/overview/index.html

Open Source “Standards” - OpenCensus

```
io.opencensus.exporter.trace.zipkin.ZipkinTraceExporter.createAndRegister("http://127.0.0.1:9411/api/v2/spans", "my-service");
io.opencensus.trace.Tracer tracer = io.opencensus.trace.Tracer.getTracer();
```

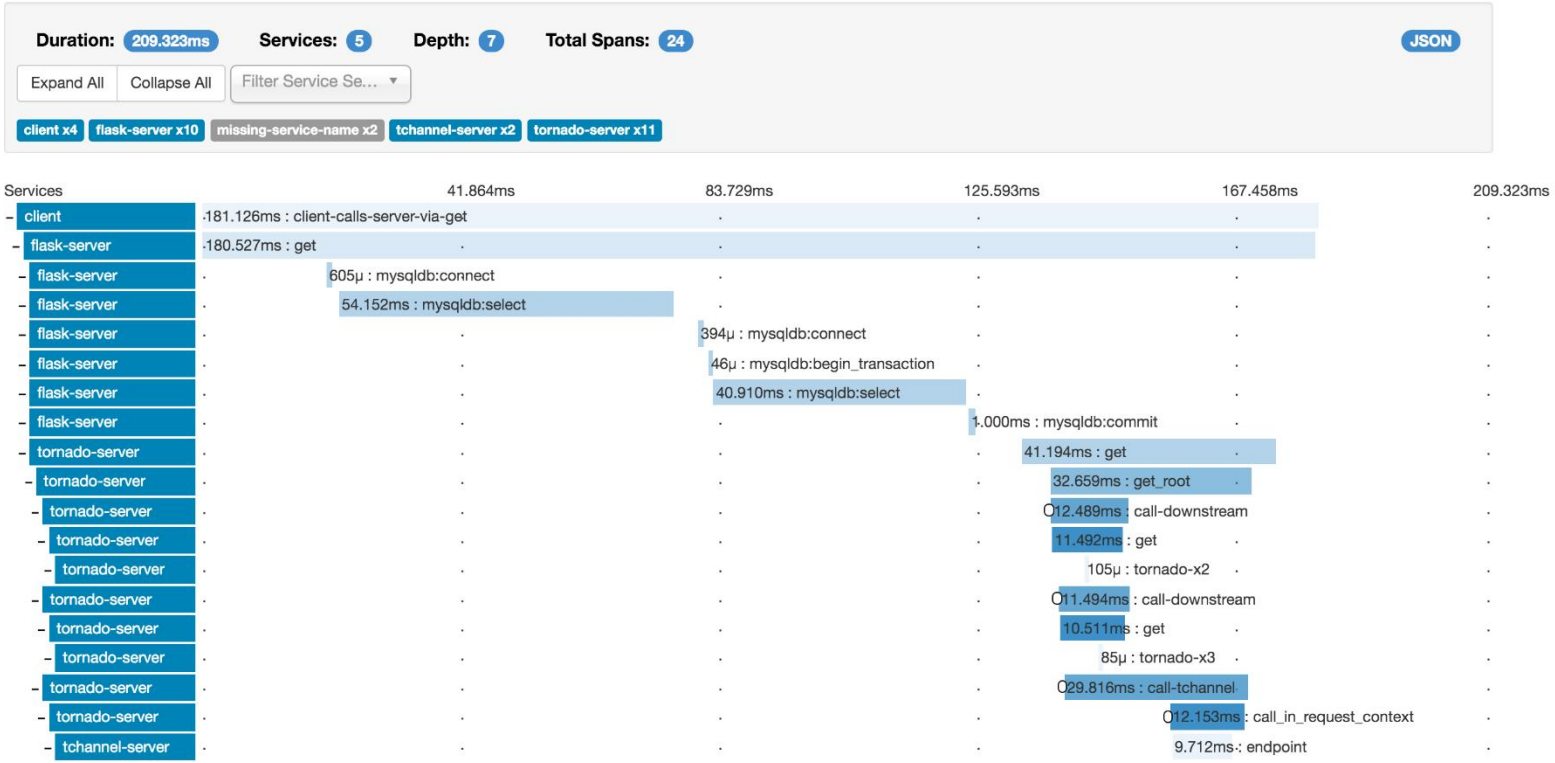
You only need to do this once

```
io.opencensus.trace.Span rootSpan = tracer.spanBuilderWithExplicitParent(„MyRootSpan“, null).startSpan();
io.opencensus.trace.Span childSpan = tracer.spanBuilderWithExplicitParent(„MyChildSpan“, rootSpan).startSpan();
childSpan.end();
rootSpan.end();
```

For each individual span

Source: https://opencensus.io/java/index.html

ZIPKIN (zipkin.io)

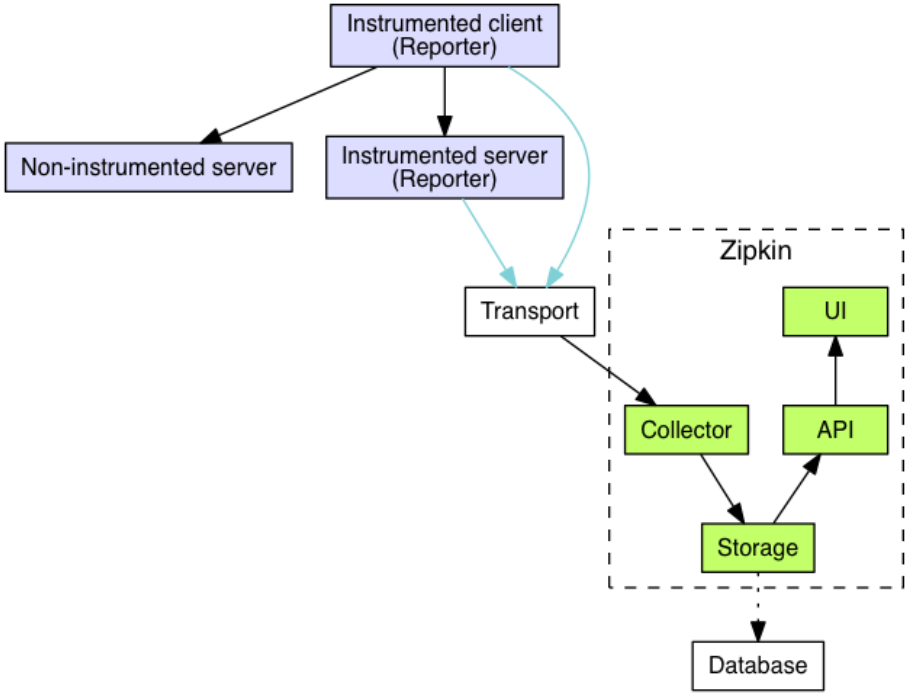


Source: https://zipkin.io/public/img/web-screenshot.png

ZIPKIN (zipkin.io)

Supported Languages:
C#, Go, Java, JavaScript, Ruby, Scala

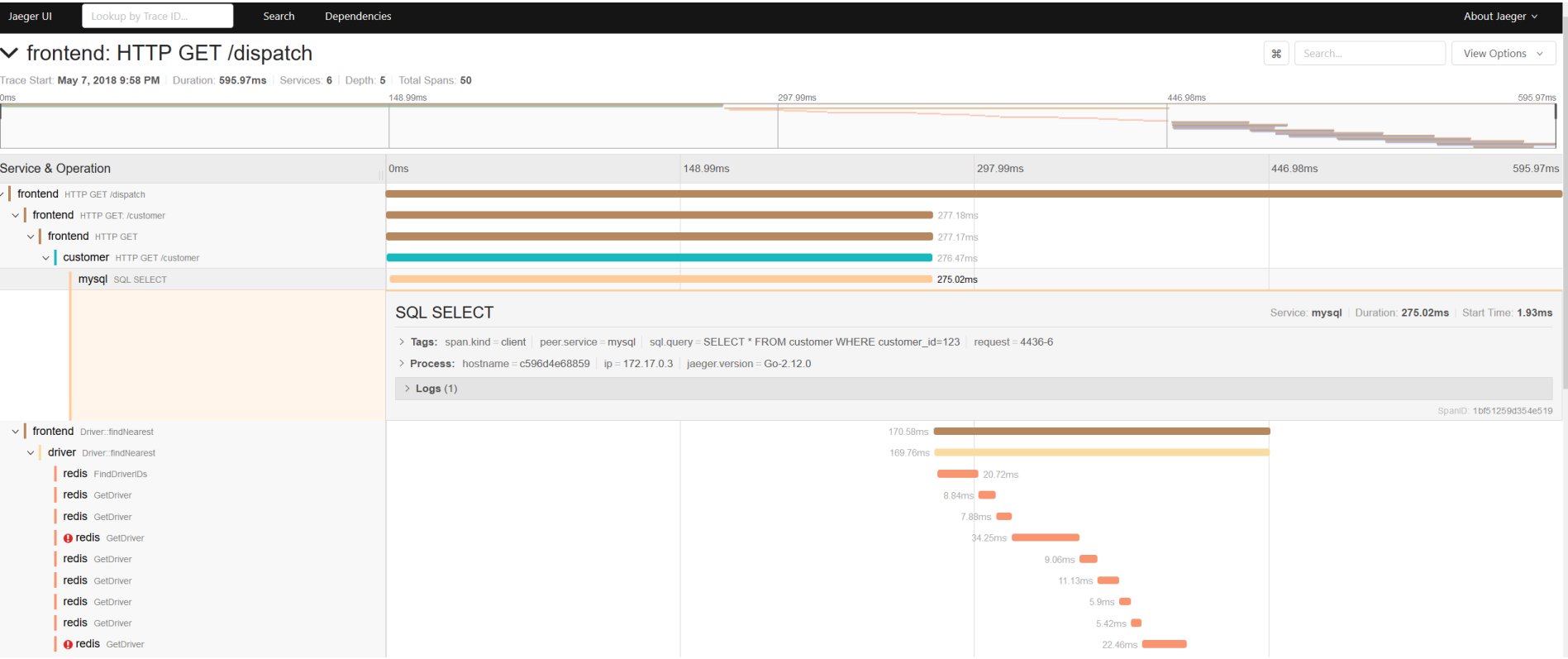
Supported Languages (Community Contributions):
C, C++, Elixir, Python, Scala, PHP



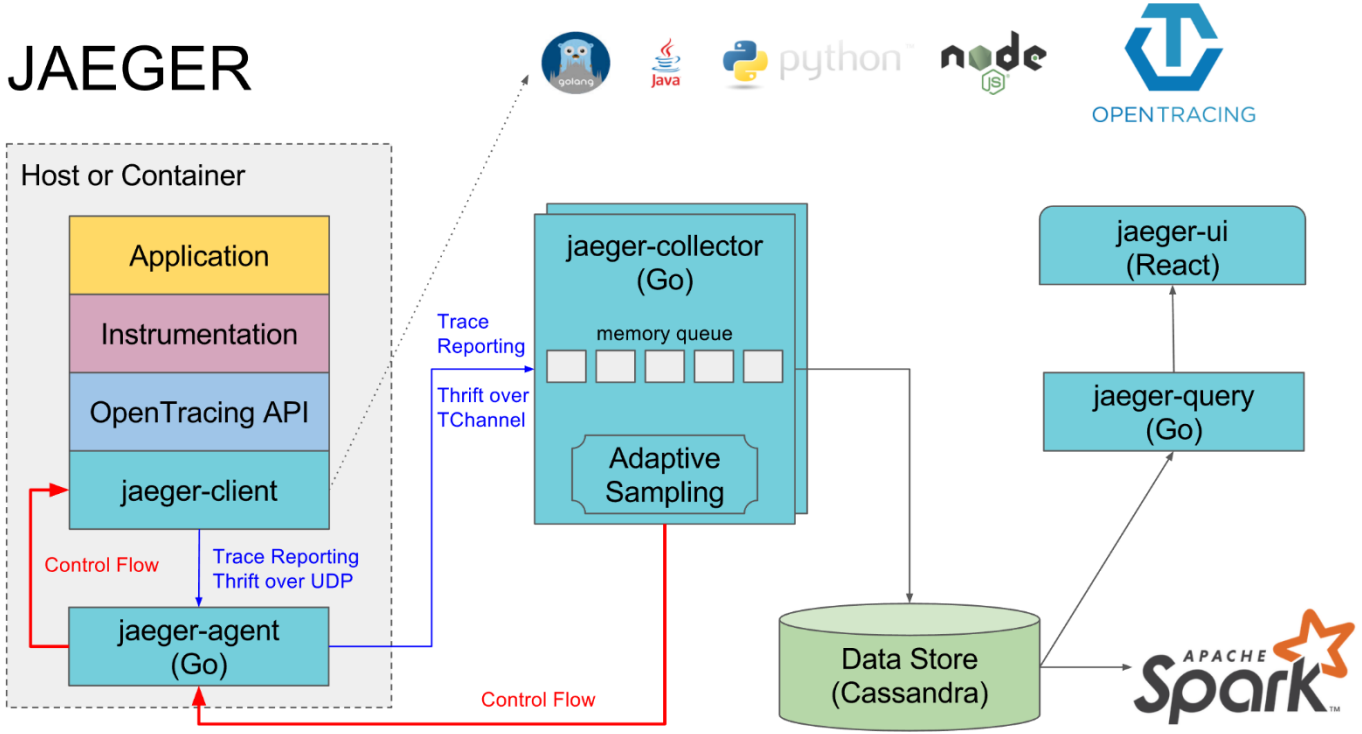
Source: https://zipkin.io/pages/architecture.html

Jaeger (jaegertracing.io)

OPENTRACING

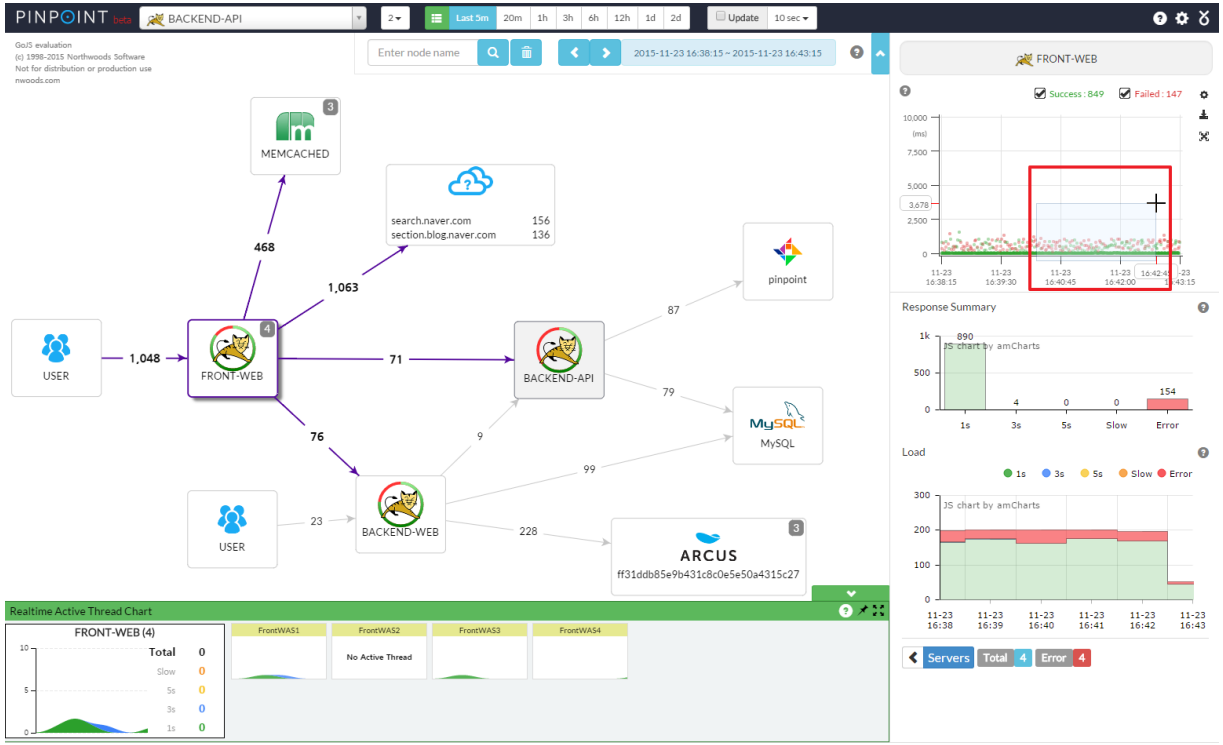


Jaeger (jaegertracing.io)



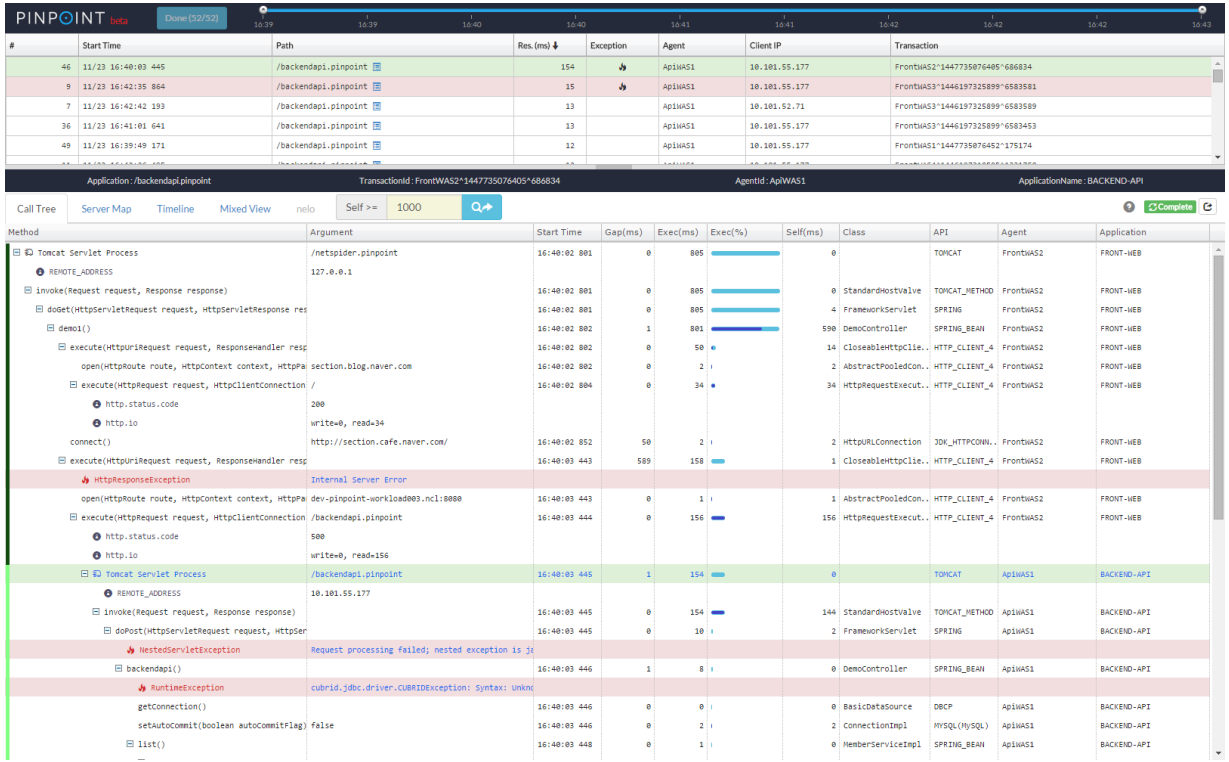
Source: https://www.jaegertracing.io/docs/architecture/

PINPOINT (http://naver.github.io/pinpoint/)



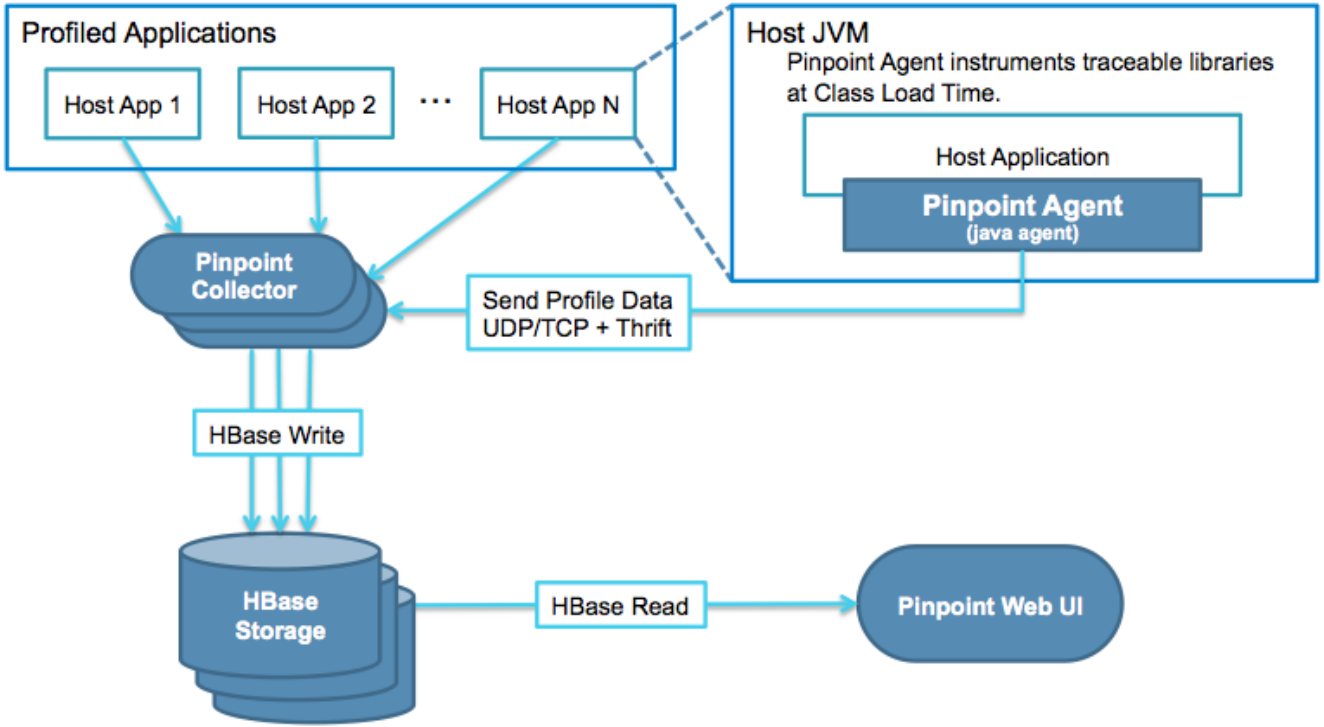
Source: http://naver.github.io/pinpoint/overview.html

PINPOINT (http://naver.github.io/pinpoint/)



Source: http://naver.github.io/pinpoint/overview.html

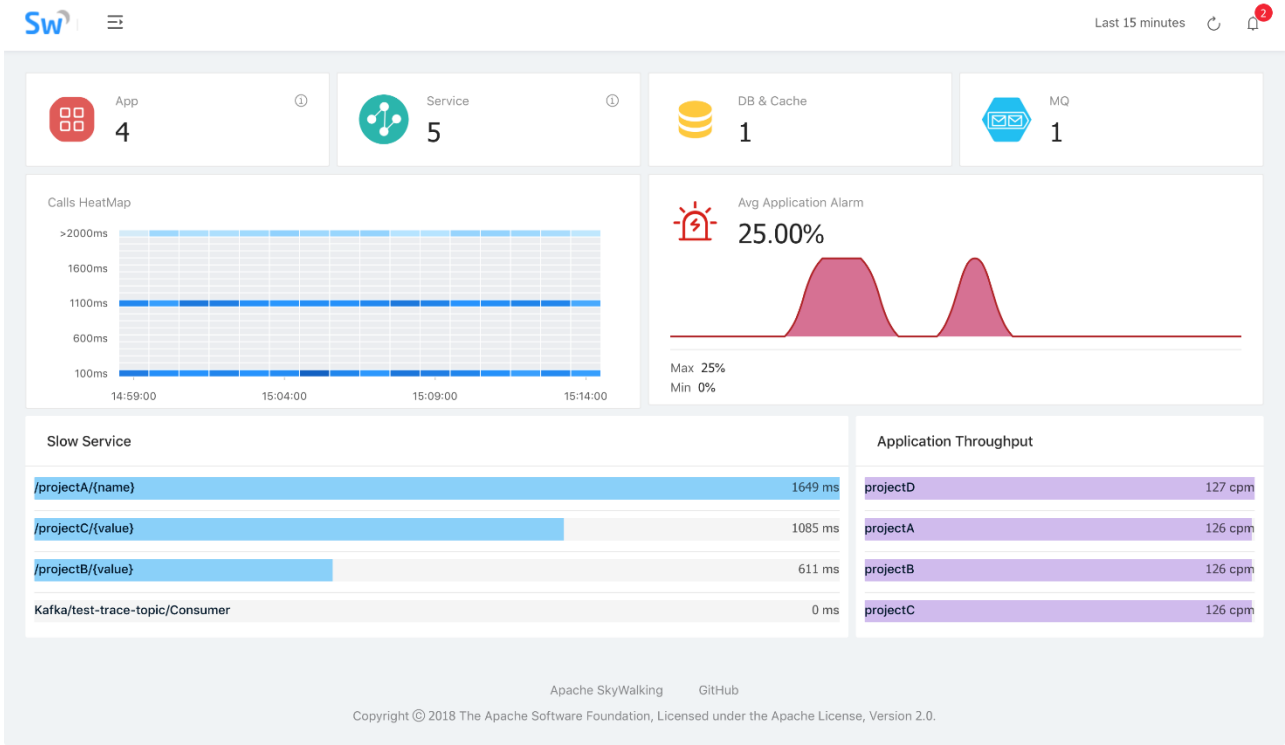
PINPOINT (http://naver.github.io/pinpoint/)



Source: http://naver.github.io/pinpoint/overview.html

Apache Skywalking (skywalking.apache.org)

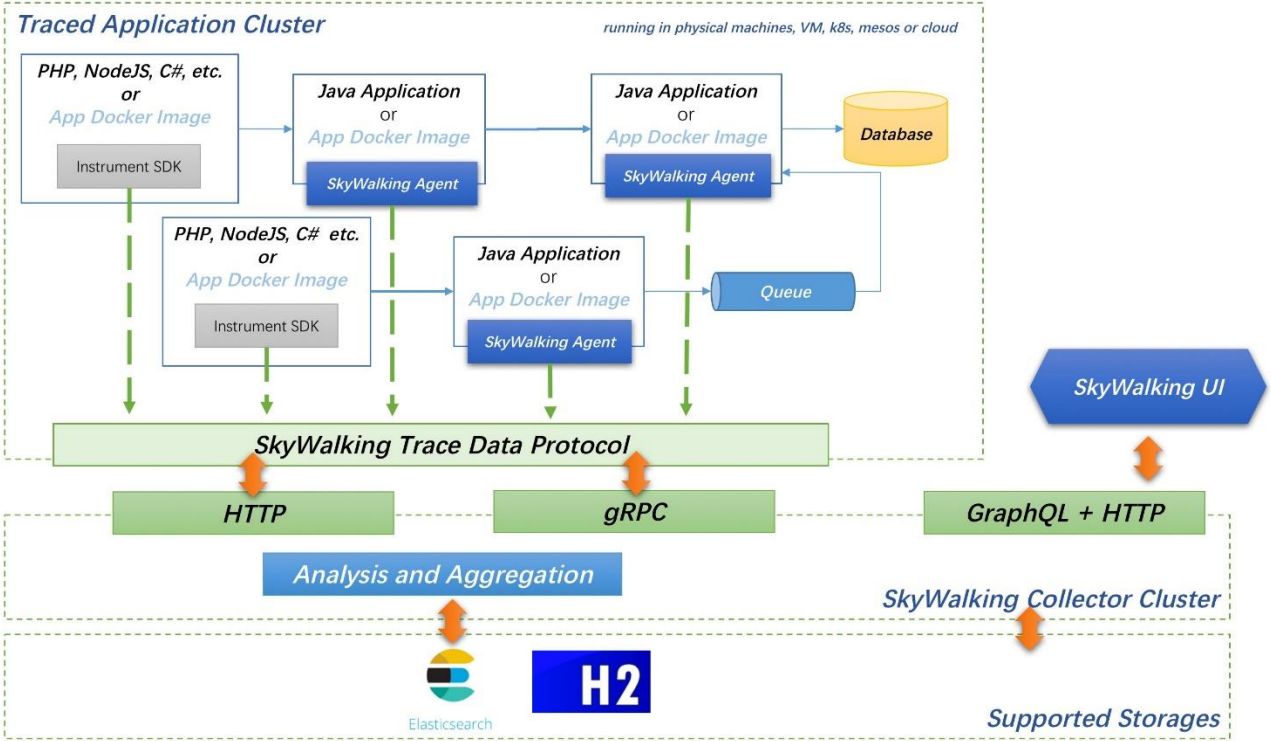
OPENTRACING



Source: https://github.com/apache/incubator-skywalking

Apache Skywalking (skywalking.apache.org)

OPENTRACING



Source: https://github.com/apache/incubator-skywalking

Apache Skywalking (skywalking.apache.org)

OPENTRACING

Agent for Java, Instrumentation SDK for PHP, C#, NodeJS

HTTP Server
- Tomcat 7
- Tomcat 8
- Tomcat 9
- Spring Boot Web 4.x
- Spring MVC 3.x, 4.x with servlet 3.x
- Nutz Web Framework 1.x
- Struts2 MVC 2.3.x -> 2.5.x
- Resin 3 (Optional[1])
- Resin 4 (Optional[1])
- Jetty Server 9

HTTP Client
- Feign 9.x
- Netflix Spring Cloud Feign 1.1.x, 1.2.x, 1.3.x
- Okhttp 3.x
- Apache httpcomponent HttpClient 4.2, 4.3
- Spring RestTemplete 4.x
- Jetty Client 9
- Apache httpcomponent AsyncClient 4.x

JDBC
- Mysql Driver 5.x, 6.x
- Oracle Driver (Optional[1])
- H2 Driver 1.3.x -> 1.4.x
- Sharding-JDBC 1.5.x
- PostgreSQL Driver 8.x, 9.x, 42.x

RPC Frameworks
- Dubbo 2.5.4 -> 2.6.0
- Dubbox 2.8.4
- Motan 0.2.x -> 1.1.0
- gRPC 1.x
- Apache ServiceComb Java Chassis 0.1 -> 0.5,1.0.x

MQ
- RocketMQ 4.x
- Kafka 0.11.0.0 -> 1.0

NoSQL
- Redis
  - Jedis 2.x
- MongoDB Java Driver 2.13-2.14,3.3+
- Memcached Client
  - Spymemcached 2.x
  - Xmemcached 2.x

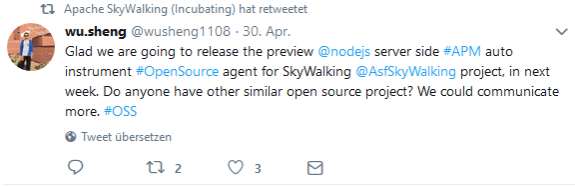
Service Discovery
- Netflix Eureka

Spring Ecosystem
- Spring Bean annotations(@Bean, @Service, @Component, @Repository) 3.x and 4.x (Optional[2])
- Spring Core Async SuccessCallback/FailureCallback/ListenableFutureCallback 4.x

Hystrix: Latency and Fault Tolerance for Distributed Systems 1.4.20 -> 1.5.12

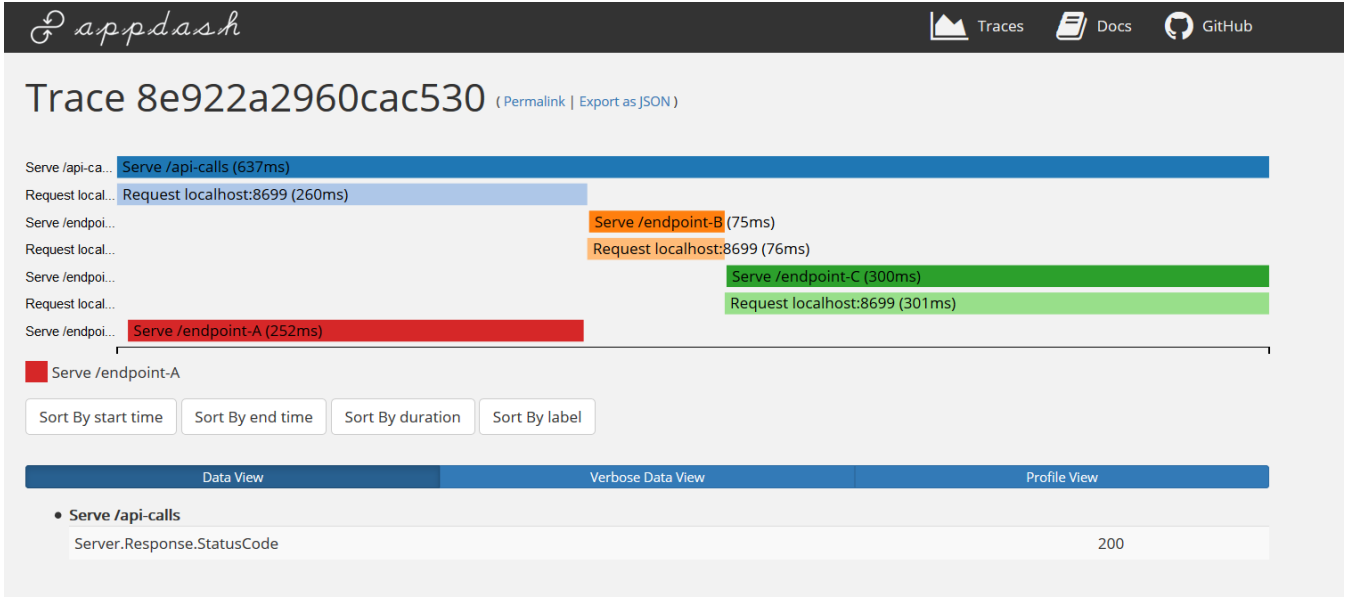
Scheduler
- Elastic Job 2.x

OpenTracing community supported



AppDash (github.com/sourcegraph/appdash)

OPENTRACING

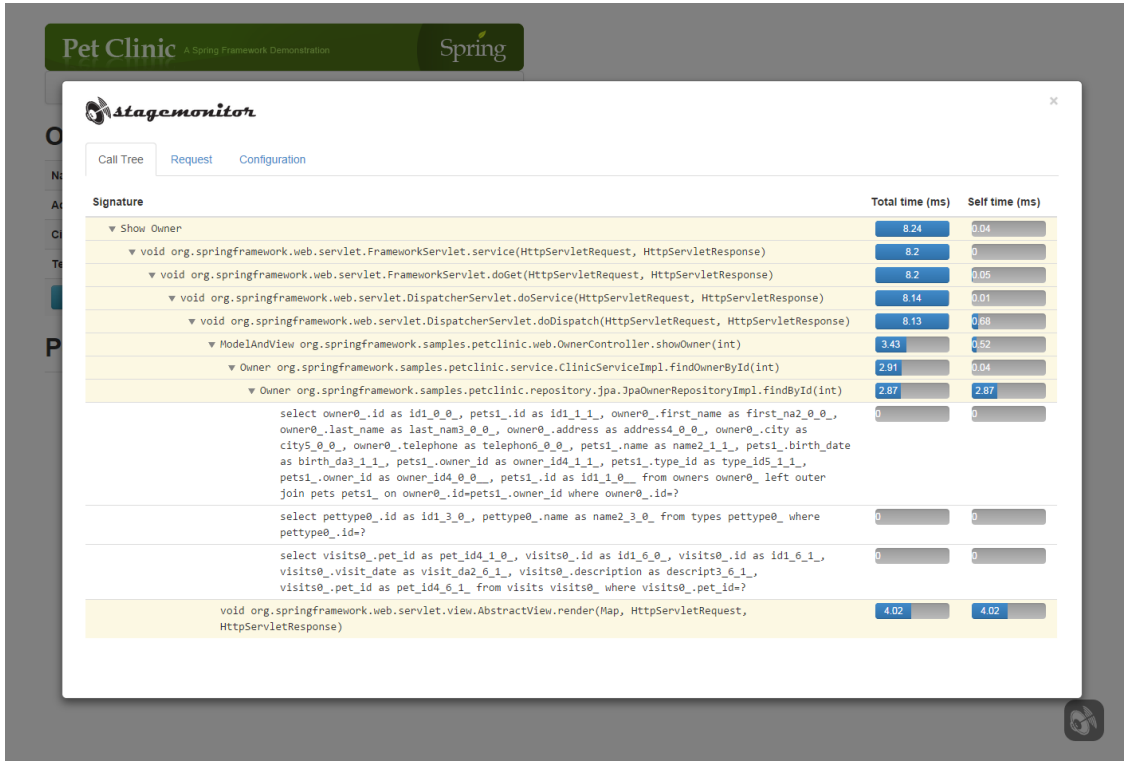


Supported Modules:

Go (https://medium.com/opentracing/distributed-tracing-in-10-minutes-51b378ee40f1 ,
(Python - https://github.com/sourcegraph/appdash/tree/master/python),
(Ruby - https://github.com/bsm/appdash-rb)

Stagemonitor (www.stagemonitor.org)

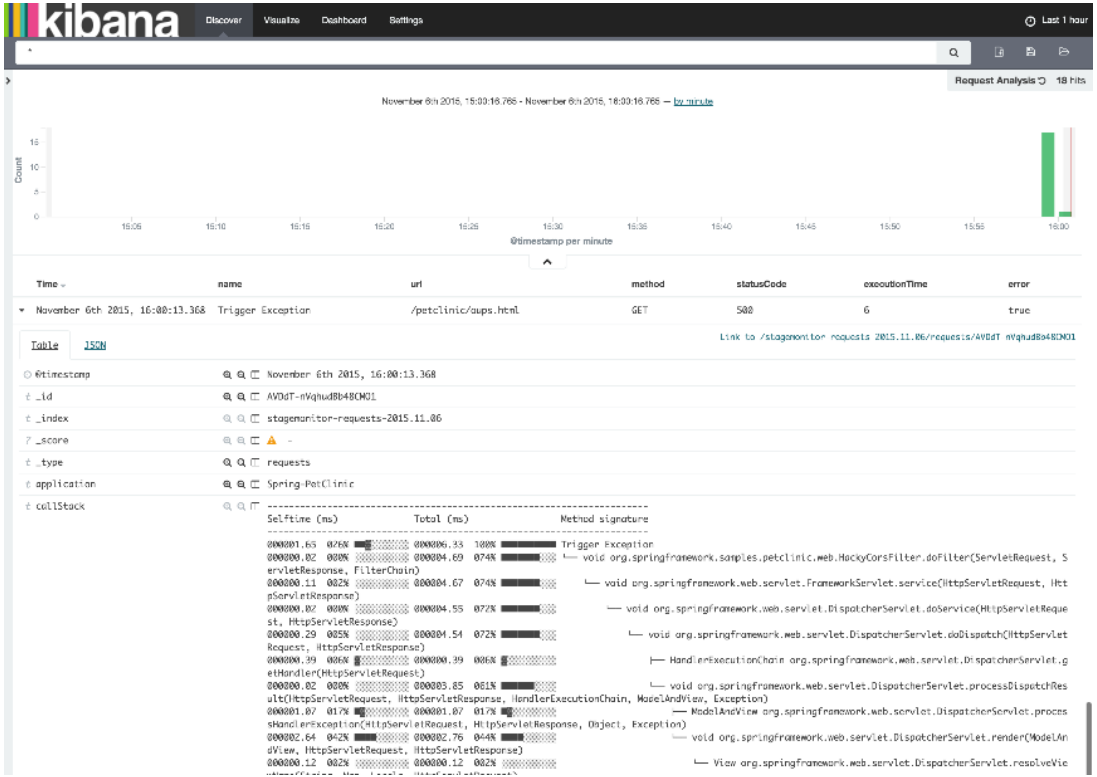
OPENTRACING



Source: http://www.stagemonitor.org/de/#overview

Stagemonitor (www.stagemonitor.org)

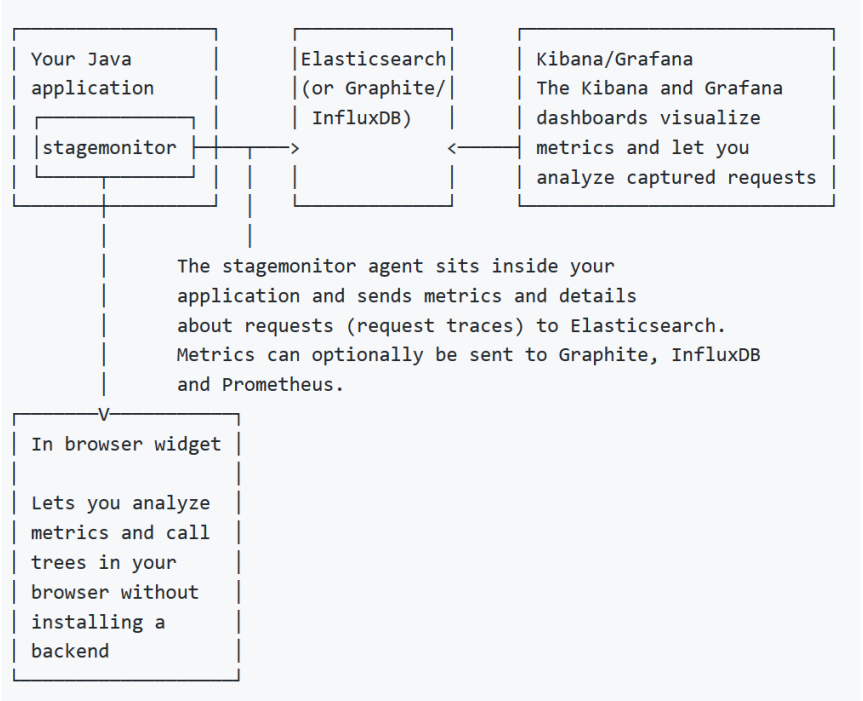
OPENTRACING



Source: https://github.com/stagemonitor/stagemonitor/wiki/Request-Analysis-Dashboard

Stagemonitor (www.stagemonitor.org)

OPENTRACING

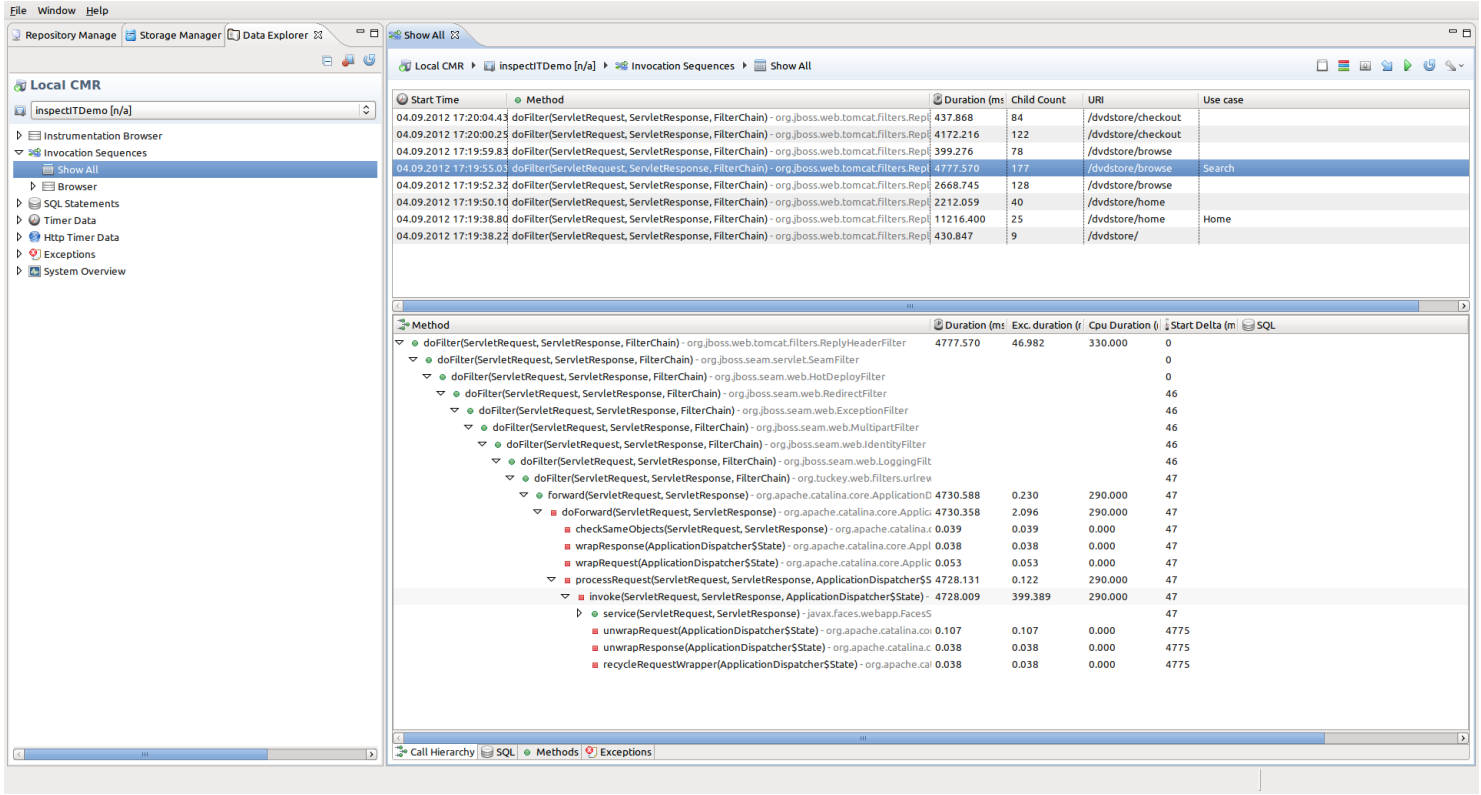


Supported Modules:

Java (https://github.com/stagemonitor/stagemonitor/wiki)

InspectIT (inspectit.rocks)

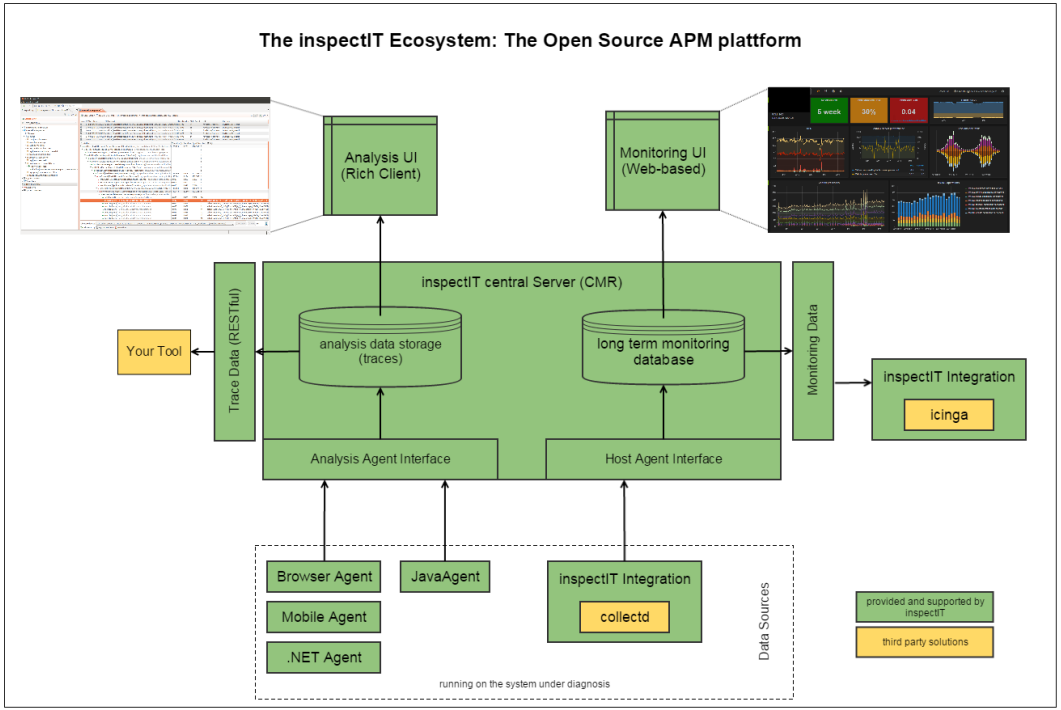
OPENTRACING



Source: https://inspectit-performance.atlassian.net/wiki/spaces/DOC18/pages/93009319/Working+with+invocation+sequences

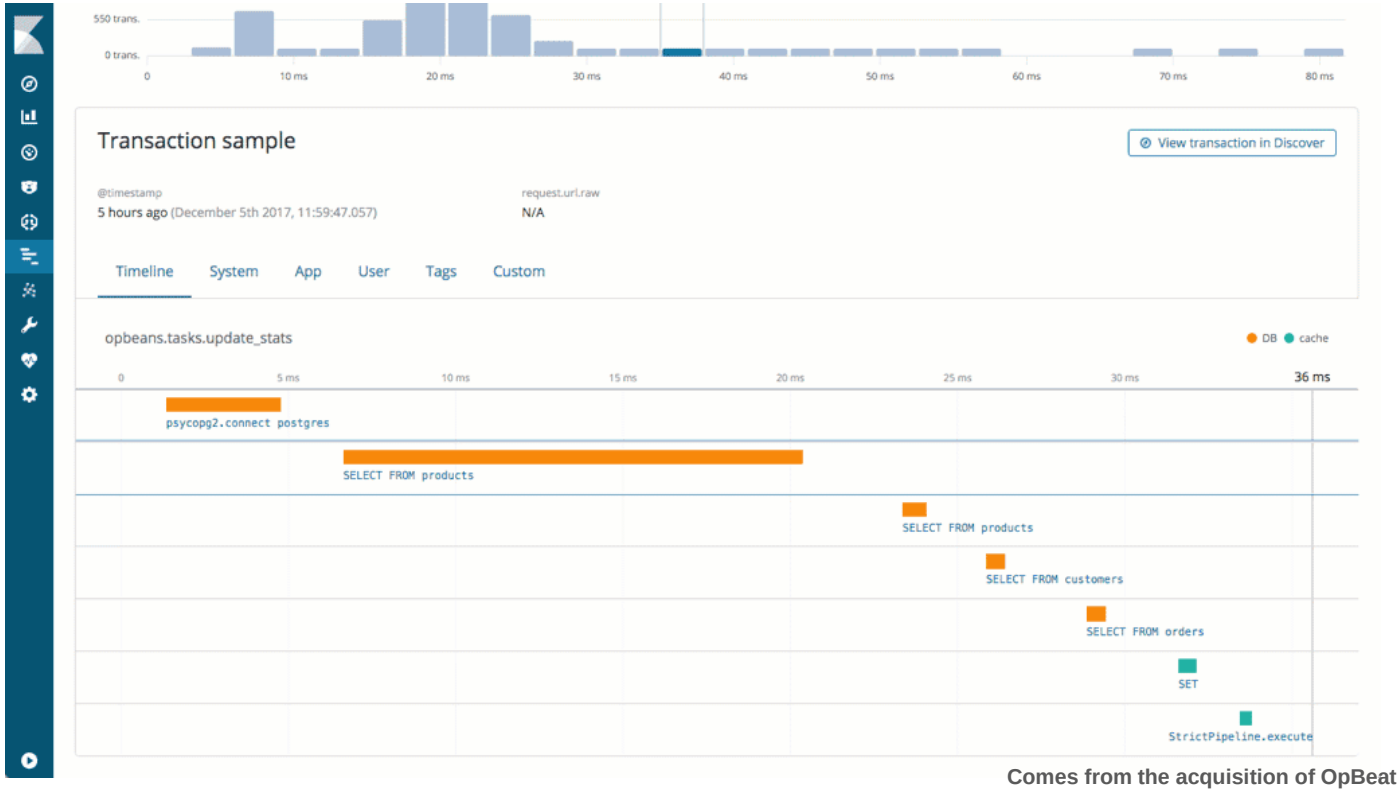
InspectIT (inspectit.rocks)

OPENTRACING



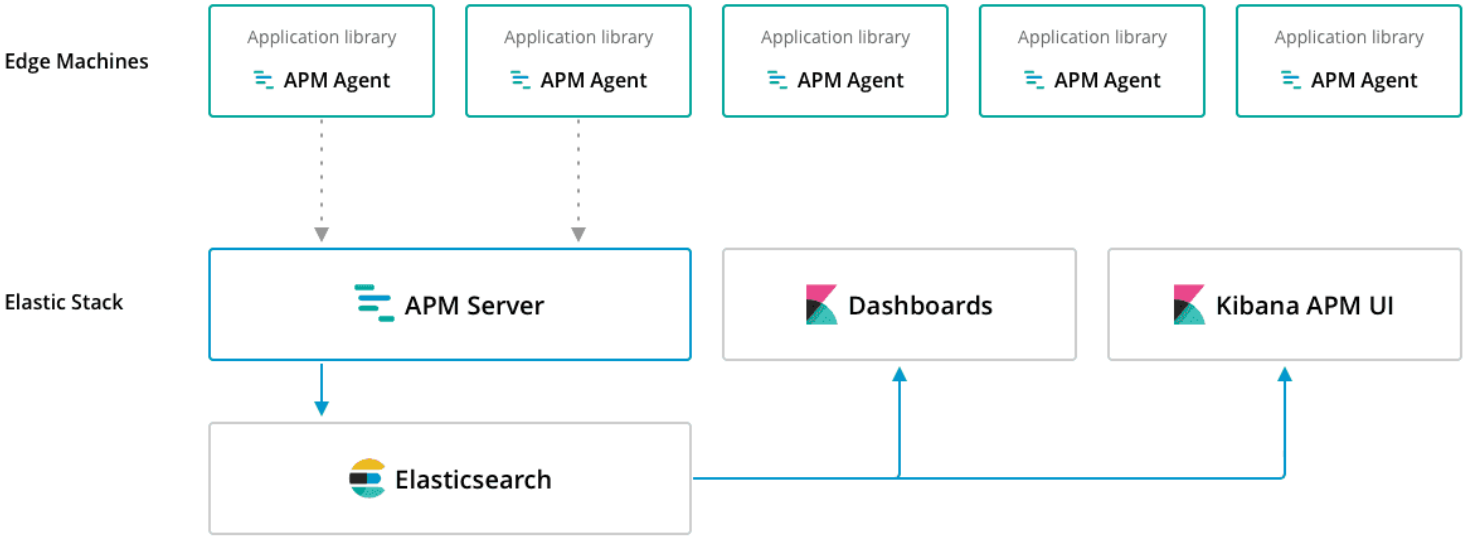
Supported Languages: Java, (.NET)

Elastic APM (www.elastic.co/solutions/apm)



Comes from the acquisition of OpBeat (part of Elastic Stack from 6.2):
https://www.elastic.co/de/blog/elastic-apm-ga-released

Elastic APM (www.elastic.co/solutions/apm)



Agents: Node.js, Python, Ruby, JavaScript, Java (Beta) (https://www.elastic.co/guide/en/apm/agent/index.html)

Source: **https://www.elastic.co/guide/en/apm/get-started/current/overview.html**

What are reasons for a proprietary alternative?

- There is also cost associated with setting up and maintaining an open source APM solution (taken from https://sematext.com/blog/performance-monitoring-comparison-build-vs-buy/) :

- **Build Your Own Monitoring System — Cost Scenario**

  - Hourly rate: 100 € (ballpark figure; could be much higher)
  - Installation: 2 hours (very optimistic)
  - Configuration: 8 hours (very optimistic)
  - Maintenance: 2 hours/month (optimistic)
  - Upgrading: 2 days (i.e., ~20 hours)/year (IF all goes well!)
  - # of servers to run this configuration: 3 (monitoring 10 total servers*)
  - Cost per server (hardware): 1,000 € each (i.e., 3,000 € total)

  - Total Cost in Year 1: 6,200 €
  - Total Cost in Year 2: 3,200 € (not including any additional server purchases)
  - Total Cost in Year 3: 3,200 € (at least, though most likely higher)

What are reasons for a proprietary alternative?

- **Easier problem resolution**:
  - You do have someone to investigate and fix issues
  - Less risk in production as tools are (mostly) more thoroughly tested

- **Broader technology support**:
  - Developing agents is very time consuming and, thus, costly – the open source community cannot spend the same amount of manpower into this effort for each and every version of a technology (e.g., supporting Tomcat, 5,6,7,8, …)

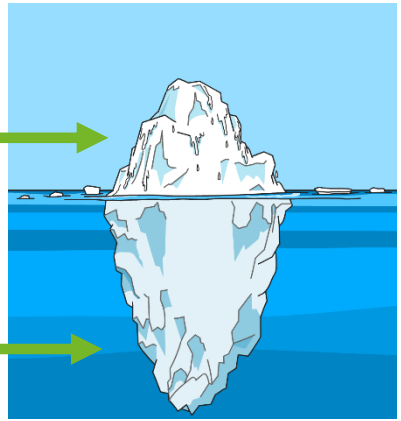
- **You can plan ahead**:
  - Vendors typically communicate the time until which a software version is supported and support the transition phase as well, this is not always the case for open source software

What are reasons for a proprietary alternative?

Remember: Code and Effort distribution of an APM Solution

UI + Server + Collectors

Agents



- Some things might change, as some open source projects (e.g., istio/Ingres/ WildFly) are already supporting OpenTracing natively
- Furthermore, there are default implementations for Spring Boot or WildFly Swarm to automatically capture traces that can be packaged in your application

Dr. Andreas Brunnert
brunnert@retit.de

RETIT
Resource Efficient Technologies & IT Systems