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:

PinPoint

Jaeger

Zipkin

inspectIT

elasticAPM

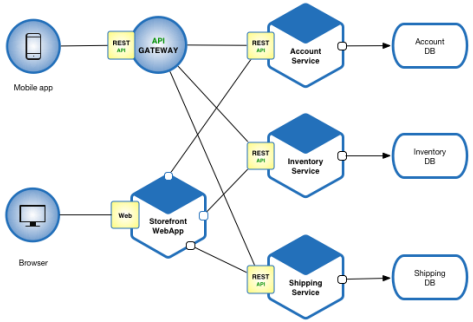
Apache Skywalking

Stage-monitor

App-dash

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

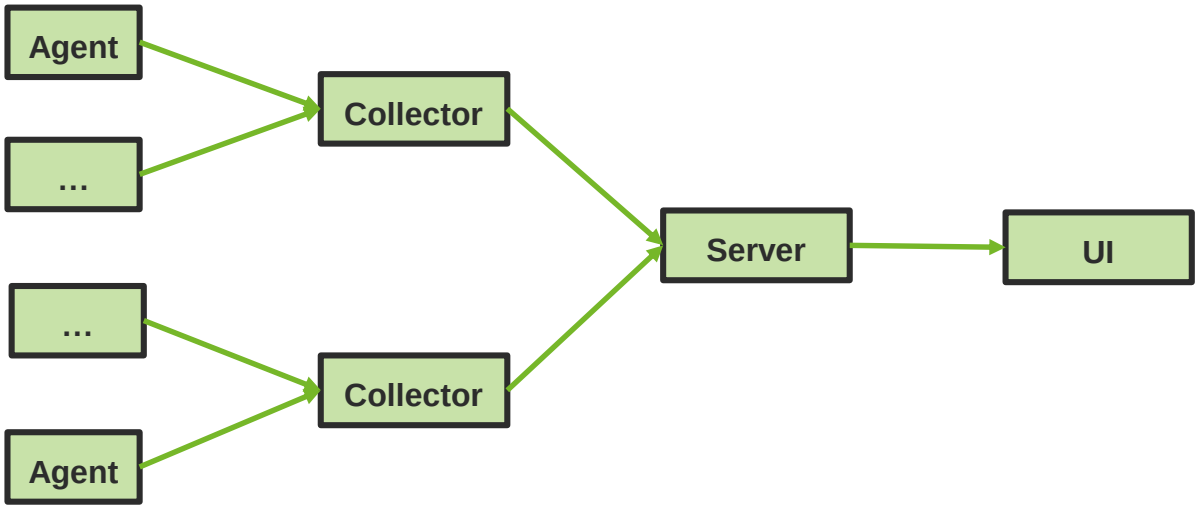
OpenCensus

OpenTelemetry

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

Context

Anatomy of an APM Solution

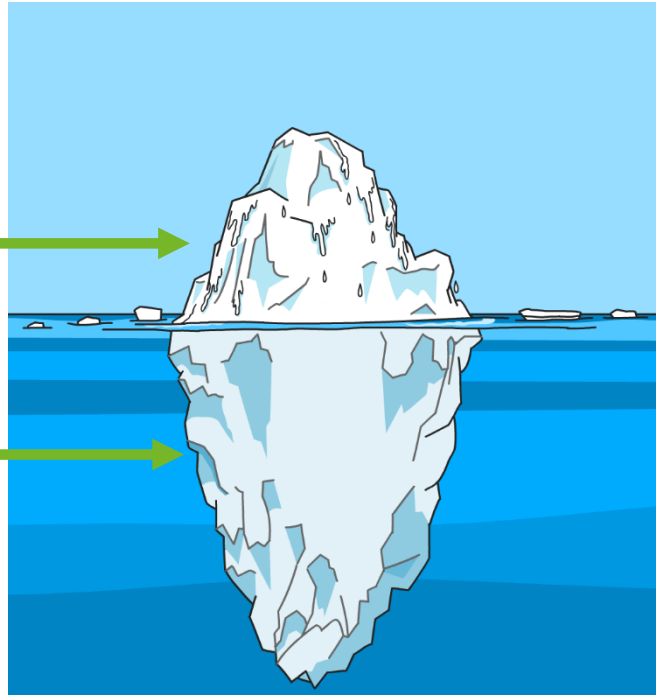


Context

Code and Effort distribution of an APM Solution

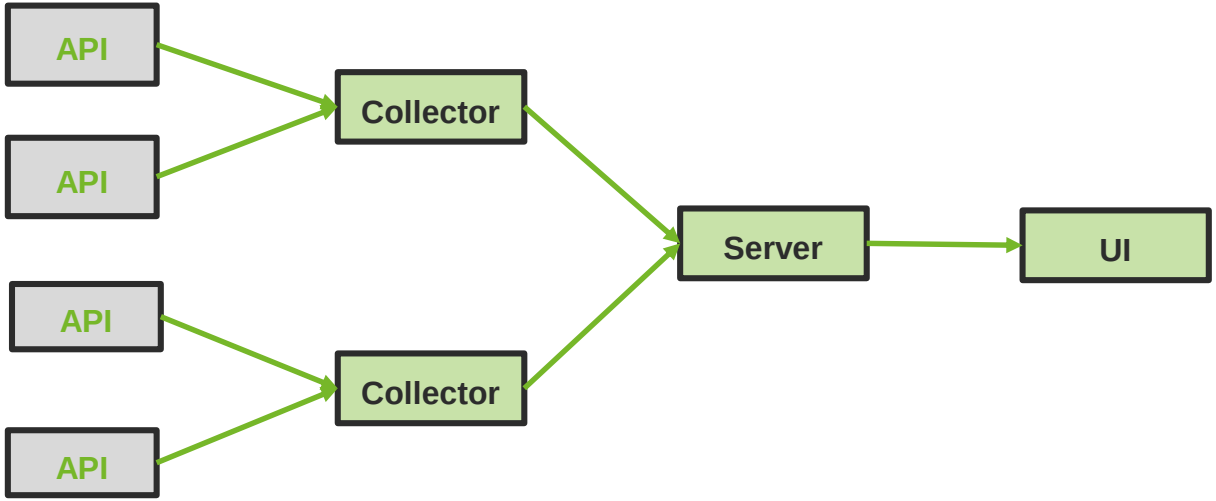
UI + Server + Collectors

Agents



Context

Scope of many open source APM solutions



Context

RESEARCH › PUBLICATIONS ›

Dapper, a Large-Scale Distributed Systems Tracing Infrastructure

Download Search Copy Bibtex

- 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

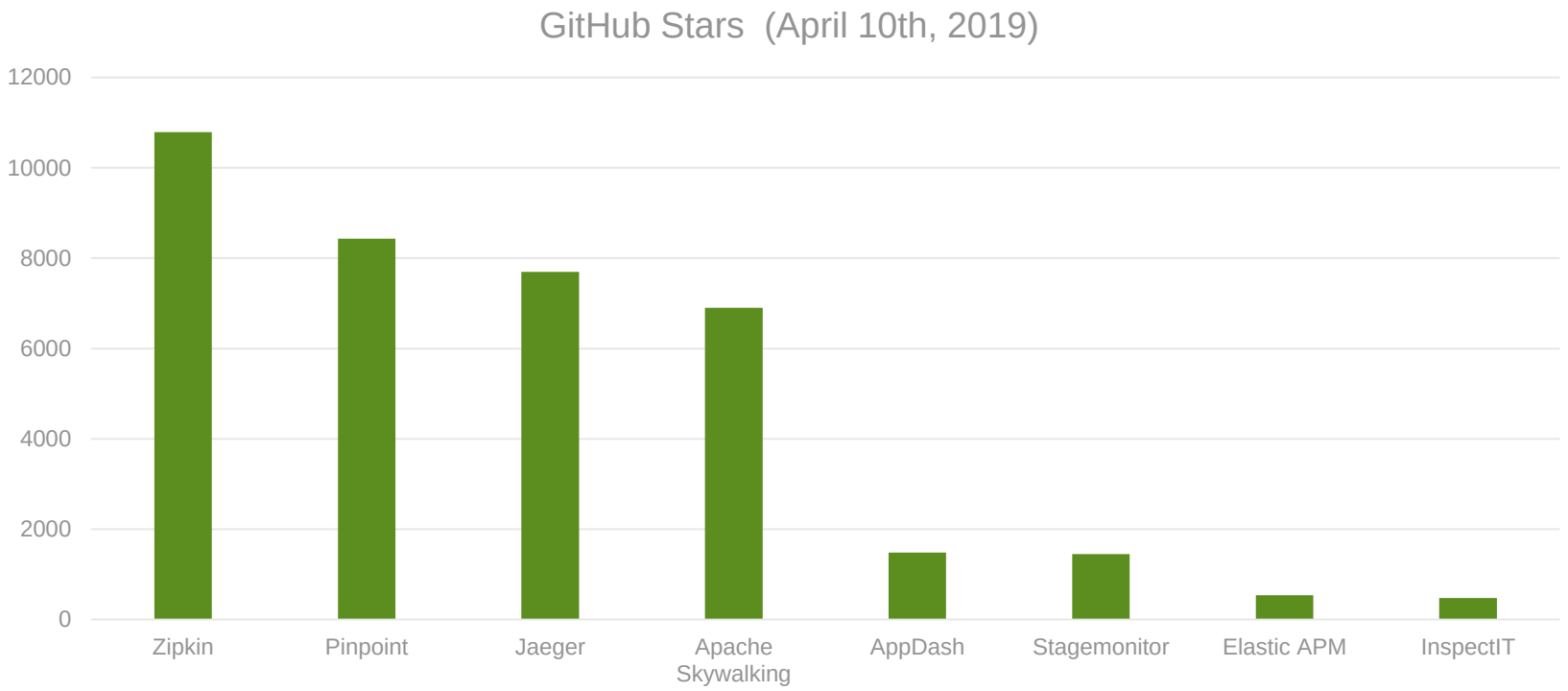
- Not in presentation (will be covered in later blog articles):
  - Setup – Effort
  - Integration Capabilities with other tools
  - License

What are reasons for a closed source alternative?

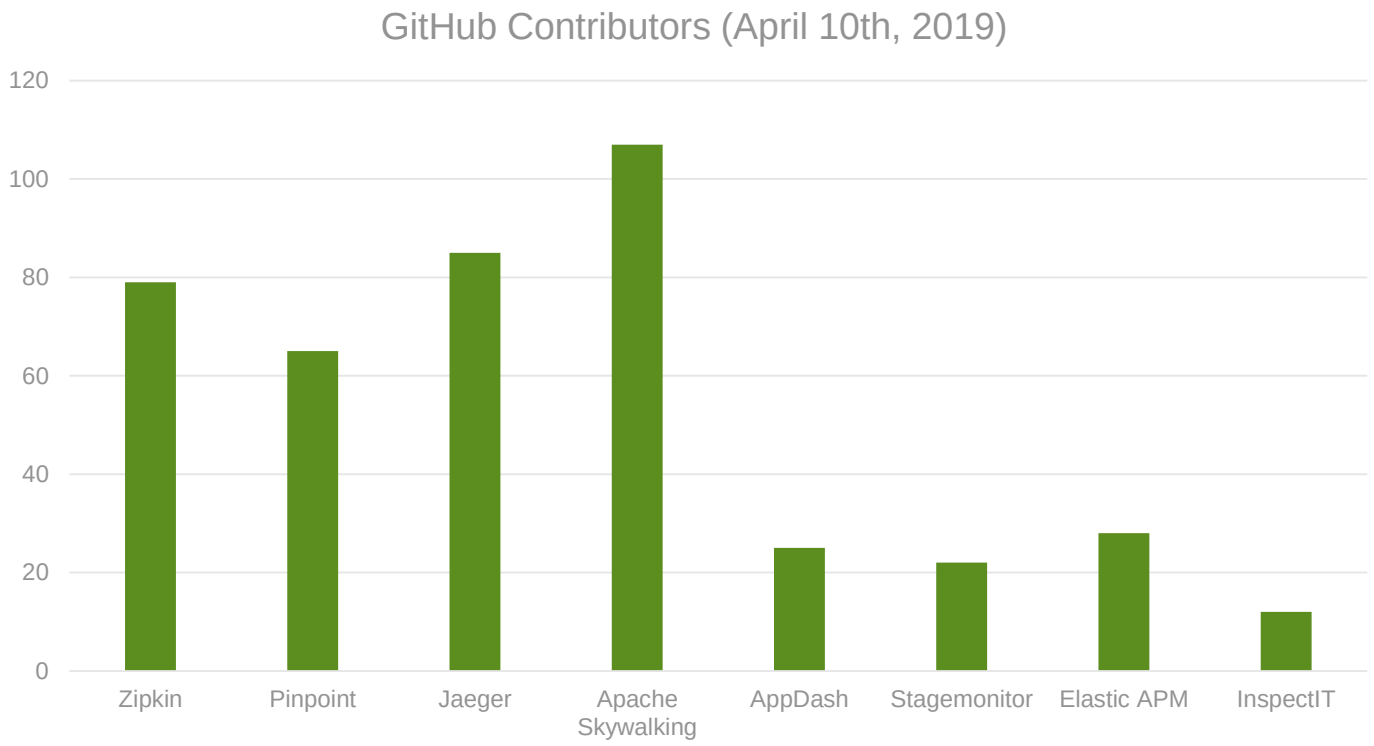
A brief timeline of tool availability (since 2014)

2014	2015	2016	2017
appdash	Skywalking	OpenCensus	
stagemonitor	INSPECTIT	OPENTRACING	
	PINPOINT		
	ZIPKIN		

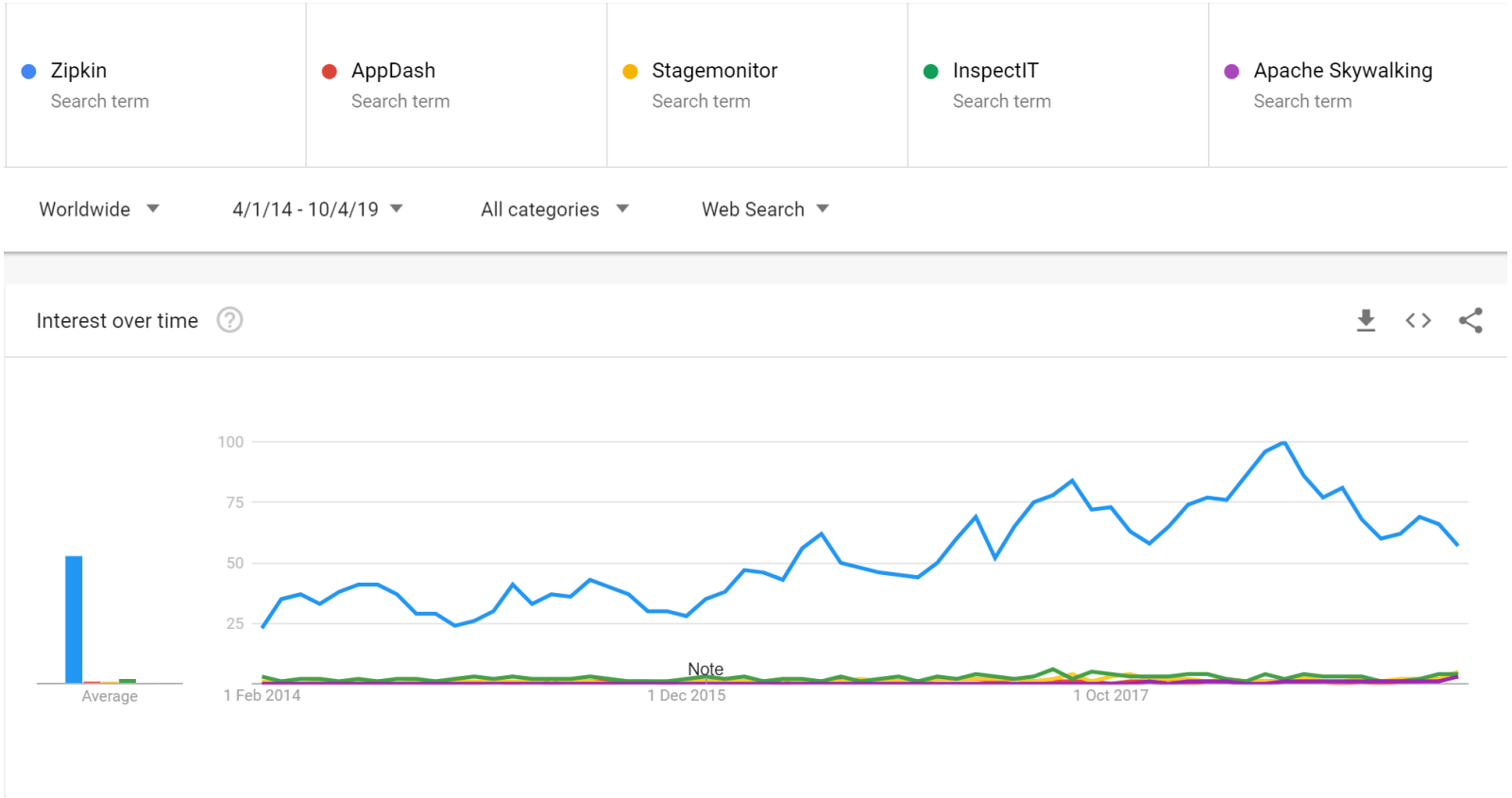
A ranking of GitHub stars



A ranking of GitHub contributes



Google Trends Analysis



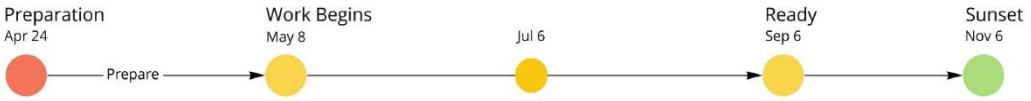
Open Source “Standards”

OPENTRACING

+

OpenCensus

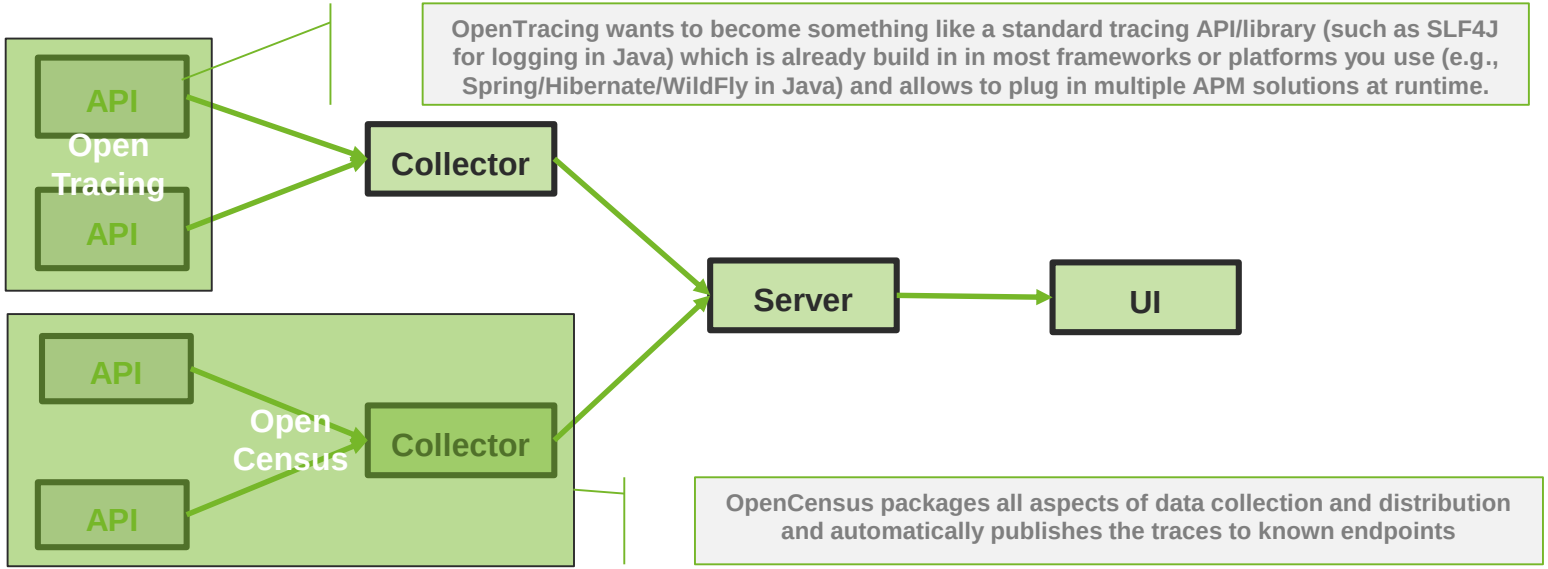
OpenTelemetry



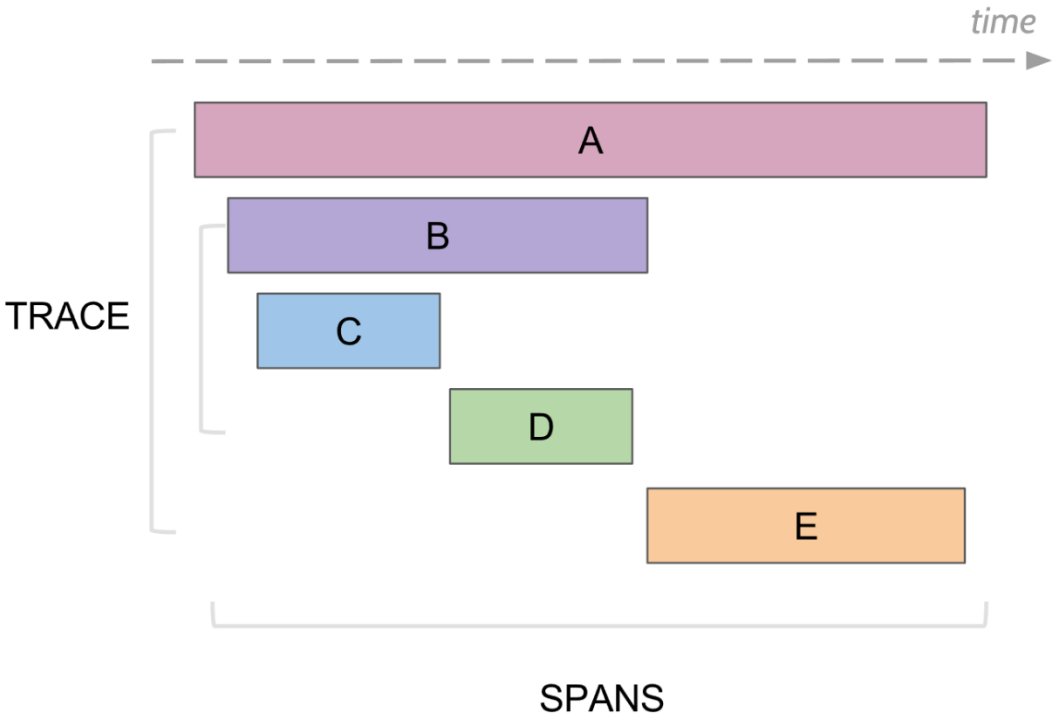
https://medium.com/opentracing/a-roadmap-to-convergence-b074e5815289

Open Source “Standards”

Scope of OpenTracing vs. OpenCensus (Simplified)



Open Source “Standards” - OpenTracing



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

Open Source “Standards” - OpenTracing

```
Causal relationships between Spans in a single Trace


        [Span A]  ←←←(the root span)
            |
     +------+------+
     |             |
 [Span B]      [Span C] ←←←(Span C is a `ChildOf` Span A)
     |             |
 [Span D]      +---+-------+
               |           |
           [Span E]    [Span F] >>> [Span G] >>> [Span H]
                                         ↑
                                         ↑
                                         ↑
                         (Span G `FollowsFrom` Span F)
```

```
Temporal relationships between Spans in a single Trace


--|-------|-------|-------|-------|-------|-------|-------|-> time

 [Span A···························································]
   [Span B·················································]
      [Span D······································]
    [Span C·································]
         [Span E·******]       [Span F··] [Span G··] [Span H··]
```

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

Open Source “Standards” - OpenTracing

```
import io.jaegertracing.Configuration;
import io.opentracing.Span;
import io.opentracing.util.GlobalTracer;

...

GlobalTracer.register(
    Configuration.fromEnv().getTracer());
);

...

try (Scope scope = tracer.buildSpan("parentSpan").startActive(true)) {
    try (Scope innerScope = tracer.buildSpan("childSpan").startActive(true)) {
        // "child" is automatically a child of "parent".
    }
}
```

You only need to do this once

Tracer configuration loaded from environment properties, but can be customized programmatically

For each individual span

Source:
https://github.com/jaegertracing/jaeger-client-java/blob/master/jaeger-core/README.md
https://opentracing.io/guides/java/

Open Source “Standards” - OpenCensus

Languages

LANGUAGE	STATS	TRACING
C#	Supported	Supported
C++	Supported	Supported
Erlang/Elixir	Supported	Supported
Go	Supported	Supported
Java (JVM, OpenJDK,	Supported	Supported
Node.js	Supported	Supported
PHP	Planned	Supported
Python	Supported	Supported
Ruby	Planned	Supported

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

Open Source “Standards” - OpenCensus

Exporters

T Backend supports Tracing

S Backend supports Stats

BACKEND	C#	C++	ERLANG	GO	JAVA	NODE.JS	PHP	PYTHON	RUBY
AWS X-Ray	-	–	–	T		–	–	–	–
Azure Monitor	T S	–	–	T	T	–	–	T	–
Datadog	–	–	T S	T S	T	–	–	–	–
Elasticsearch	-	–	–	–	T	–	–	–	–
Honeycomb	–	–	–	T	–	–	–	–	–
Instana	–	–	–	–	T	T	–	–	–
Jaeger	–	–	–	T	T	T	–	T	–
Prometheus	S	S	S	S	S	S	–	S	–
SignalFX	–	–	–	–	S	–	–	–	–
Stackdriver	T	T S	T	T S	T S	T S	–	T S	–
Zipkin	T	T	T	T	T	T	–	T	–

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

Open Source “Standards” - OpenCensus

- zPages: in process web pages, displaying collected data from process
- No backend necessary.
- Useful for debugging.
- Available for Go, Java and Node.js.

OpenCensus

TraceZ Summary

Span Name	Running	Latency Samples									Error Samples
		[>0us]	[>10us]	[>100us]	[>1ms]	[>10ms]	[>100ms]	[>1s]	[>10s]	[>100s]	
ExampleSpan	0	0	0	0	0	0	1	0	0	0	0
HttpServer/rpcz	0	0	0	0	0	1	0	0	0	0	0
HttpServer/statsz	0	0	0	0	7	3	0	0	0	0	0
HttpServer/traceconfigz	0	0	0	0	0	1	0	0	0	0	0
HttpServer/tracez	0	0	0	0	0	0	0	0	0	0	0

OpenCensus

RPC Stats

Sent

Method	Count			Avg latency (ms)			Rate (rpc/s)			Input (kb/s)			Output (kb/s)			Errors		
	Min.	Hr.	Tot.	Min.	Hr.	Tot.	Min.	Hr.	Tot.	Min.	Hr.	Tot.	Min.	Hr.	Tot.	Min.	Hr.	Tot.
ExampleMethod/abc.123/xyz.890/example	1	1	1	1.000	1.000	1.000	0.017	0.000	0.042	1.628	0.027	4.123	1.628	0.027	4.122	1	1	1

Received

Method	Count			Avg latency (ms)			Rate (rpc/s)			Input (kb/s)			Output (kb/s)			Errors		
	Min.	Hr.	Tot.	Min.	Hr.	Tot.	Min.	Hr.	Tot.	Min.	Hr.	Tot.	Min.	Hr.	Tot.	Min.	Hr.	Tot.
ExampleMethod/abc.123/xyz.890/example	1	1	1	1.000	1.000	1.000	0.017	0.000	0.042	1.628	0.027	4.118	1.628	0.027	4.118	1	1	1

Source: https://opencensus.io/zpages/#zpages

Open Source “Standards” - OpenCensus

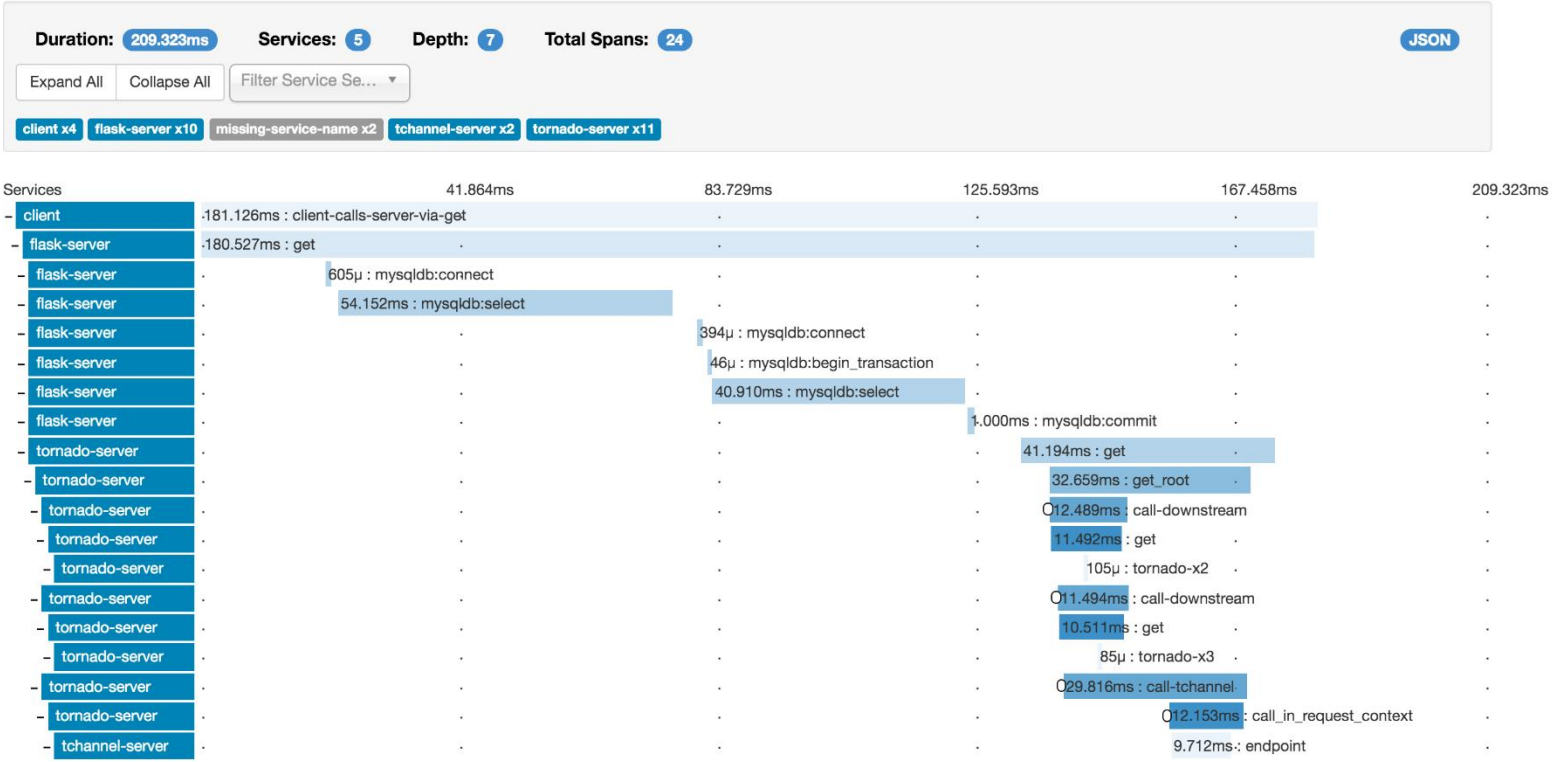
```
import io.opencensus.common.Scop
import io.opencensus.exporter.trace.zipkin.ZipkinTraceExporter;
import io.opencensus.trace.Tracer;
…
ZipkinTraceExporter.createAndRegister("http://127.0.0.1:9411/api/v2/spans", "my-service");
Tracer tracer = Tracing.getTracer(); // Global singleton Tracer object
…
try (Scope scope = tracer.spanBuilder("main").startScopedSpan()) {
    System.out.println("About to do some busy work...");
    for (int i = 0; i < 10; i++) {
        doWork(i);
    }
}
…
public void doWork(int i) {
    // Starts another span, which will be a child span if another span is already active
    try (Scope scope = tracer.spanBuilder("main").startScopedSpan()) {
        // work
    }
}
```

You only need to do this once

For each individual span

Source: **https://opencensus.io/quickstart/java/tracing/**

ZIPKIN (zipkin.io)

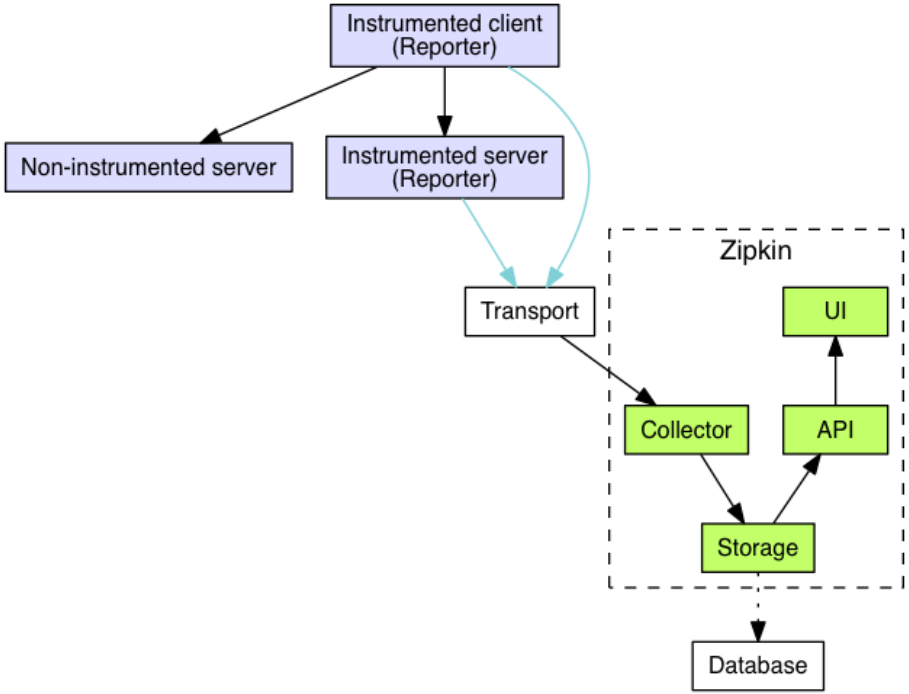


Source: https://zipkin.io/public/img/web-screenshot.png

ZIPKIN (zipkin.io)

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

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



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

Jaeger (jaegertracing.io)

OPENTRACING

Jaeger UI | Lookup by Trace ID... | Search | Dependencies | About Jaeger

frontend: HTTP GET /dispatch

Search... | View Options

Trace Start: **May 7, 2018 9:58 PM** | Duration: **595.97ms** | Services: **6** | Depth: **5** | Total Spans: **50**

0ms | 148.99ms | 297.99ms | 446.98ms | 595.97ms

Service & Operation	0ms	148.99ms	297.99ms	446.98ms	595.97ms
frontend HTTP GET /dispatch					
frontend HTTP GET /customer	277.18ms				
frontend HTTP GET	277.17ms				
customer HTTP GET /customer	276.47ms				
mysql SQL SELECT	275.02ms				
frontend Driver::findNearest	170.58ms				
driver Driver::findNearest	169.76ms				
redis FindDriverIDs	20.72ms				
redis GetDriver	8.84ms				
redis GetDriver	7.88ms				
redis GetDriver	34.25ms				
redis GetDriver	9.06ms				
redis GetDriver	11.13ms				
redis GetDriver	5.9ms				
redis GetDriver	5.42ms				
redis GetDriver	22.46ms				

SQL SELECT

Service: **mysql** | Duration: **275.02ms** | Start Time: **1.93ms**

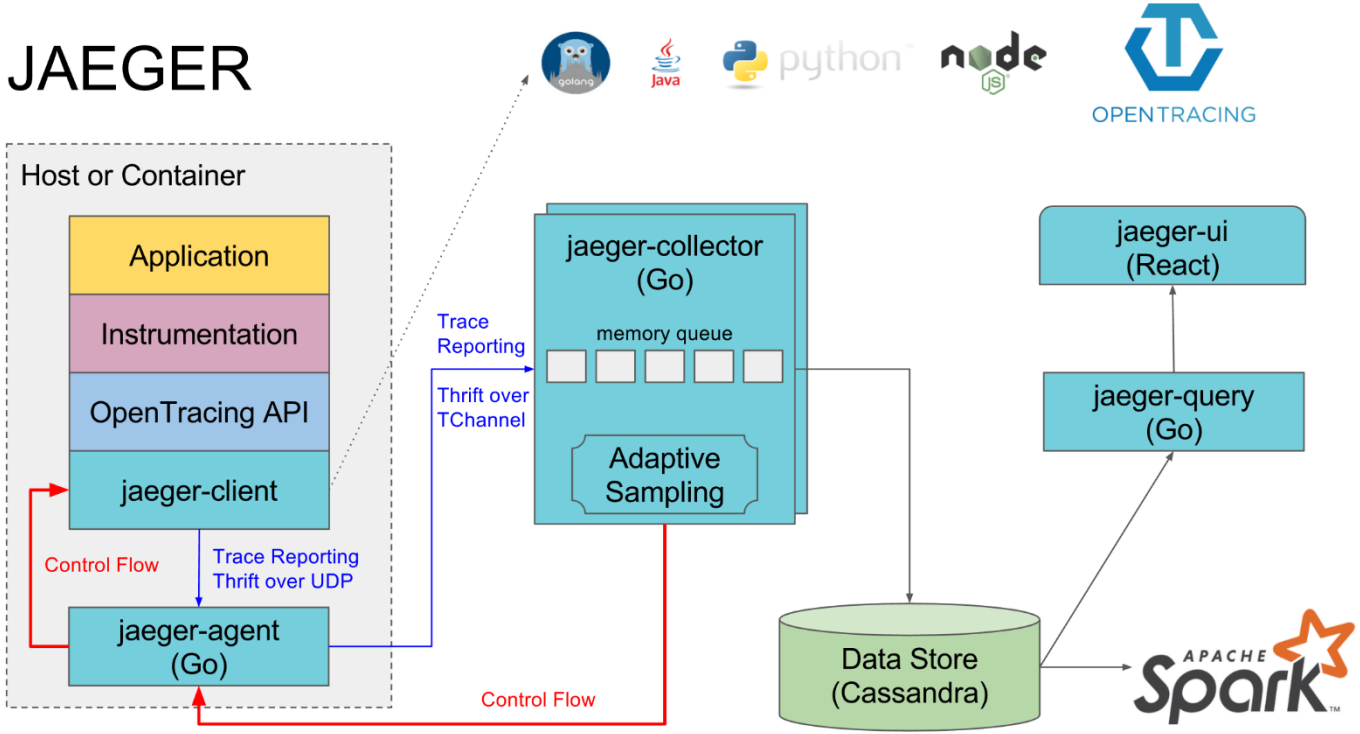
> **Tags:** span.kind = client | peer.service = mysql | sql.query = SELECT * FROM customer WHERE customer_id=123 | request = 4436-6

> **Process:** hostname = c596d4e68859 | ip = 172.17.0.3 | jaeger.version = Go-2.12.0

> **Logs** (1)

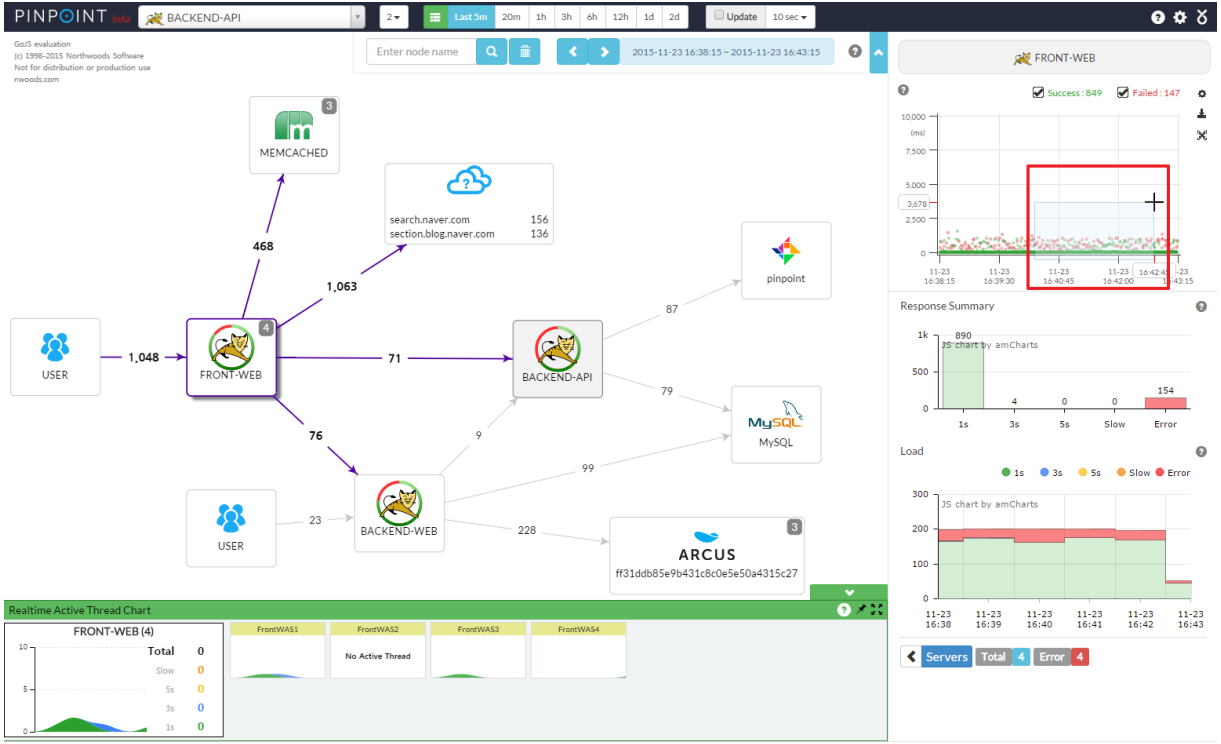
SpanID: 1bf51259d354e519

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/)

PINPOINT beta | Done (52/52) | 16:39 16:39 16:40 16:40 16:41 16:41 16:42 16:42 16:42 16:43

#	Start Time	Path	Res. (ms)	Exception	Agent	Client IP	Transaction
46	11/23 16:40:03 445	/backendapi.pinpoint	154		ApiWAS1	10.101.55.177	FrontWAS2^1447735076405^686834
9	11/23 16:42:35 864	/backendapi.pinpoint	15		ApiWAS1	10.101.55.177	FrontWAS3^1446197325899^6583581
7	11/23 16:42:42 193	/backendapi.pinpoint	13		ApiWAS1	10.101.52.71	FrontWAS3^1446197325899^6583589
36	11/23 16:41:01 641	/backendapi.pinpoint	13		ApiWAS1	10.101.55.177	FrontWAS3^1446197325899^6583453
49	11/23 16:39:49 171	/backendapi.pinpoint	12		ApiWAS1	10.101.55.177	FrontWAS1^1447735076452^175174

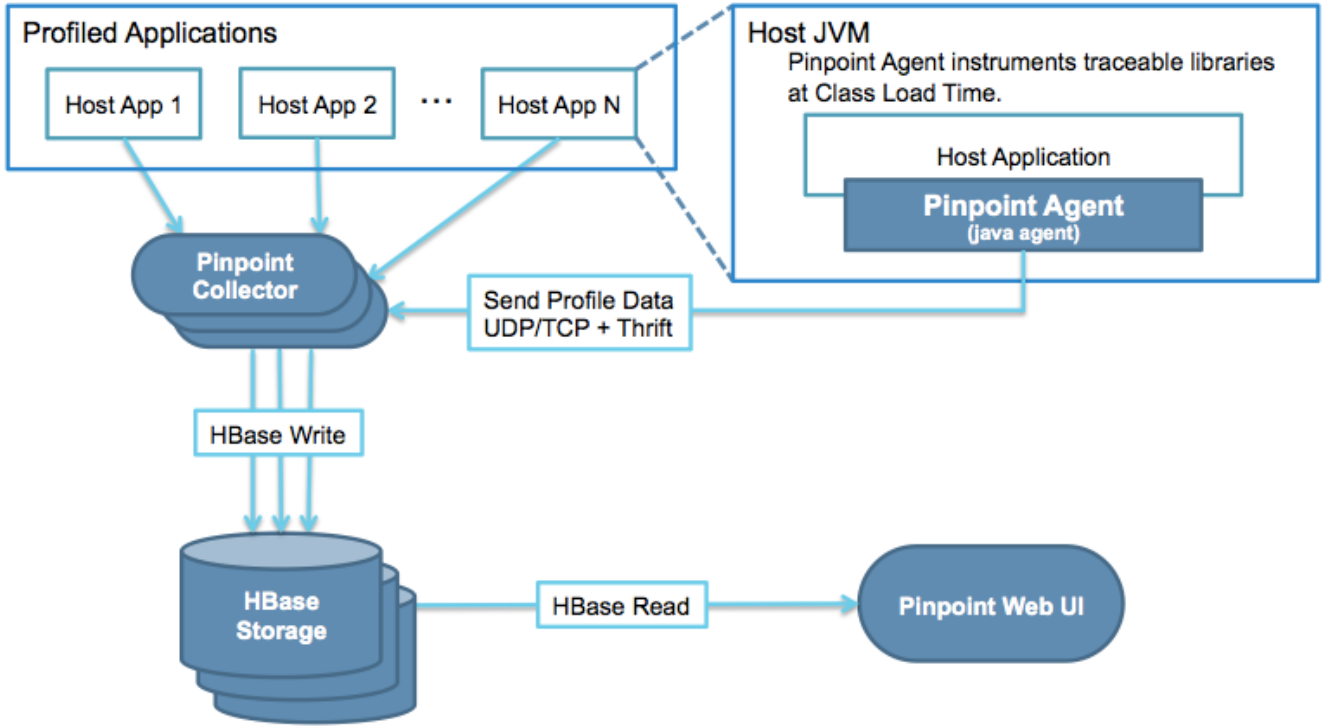
Application : /backendapi.pinpoint | TransactionId : FrontWAS2^1447735076405^686834 | AgentId : ApiWAS1 | ApplicationName : BACKEND-API

Call Tree | Server Map | Timeline | Mixed View | nelo | Self >= 1000 | Complete

Method	Argument	Start Time	Gap(ms)	Exec(ms)	Exec(%)	Self(ms)	Class	API	Agent	Application
Tomcat Servlet Process	/netspider.pinpoint	16:40:02 801	0	805		0		TOMCAT	FrontWAS2	FRONT-WEB
REMOTE_ADDRESS	127.0.0.1									
invoke(Request request, Response response)		16:40:02 801	0	805		0	StandardHostValve	TOMCAT_METHOD	FrontWAS2	FRONT-WEB
doGet(HttpServletRequest request, HttpServletResponse res		16:40:02 801	0	805		4	FrameworkServlet	SPRING	FrontWAS2	FRONT-WEB
demo1()		16:40:02 802	1	801		590	DemoController	SPRING_BEAN	FrontWAS2	FRONT-WEB
execute(HttpUriRequest request, ResponseHandler resp		16:40:02 802	0	50		14	CloseableHttpClie..	HTTP_CLIENT_4	FrontWAS2	FRONT-WEB
open(HttpRoute route, HttpContext context, HttpPa	section.blog.naver.com	16:40:02 802	0	2		2	AbstractPooledCon..	HTTP_CLIENT_4	FrontWAS2	FRONT-WEB
execute(HttpRequest request, HttpClientConnection	/	16:40:02 804	0	34		34	HttpRequestExecut..	HTTP_CLIENT_4	FrontWAS2	FRONT-WEB
http.status.code	200									
http.io	write=0, read=34									
connect()	http://section.cafe.naver.com/	16:40:02 852	50	2		2	HttpURLConnection	JDK_HTTPCONN..	FrontWAS2	FRONT-WEB
execute(HttpUriRequest request, ResponseHandler resp		16:40:03 443	589	158		1	CloseableHttpClie..	HTTP_CLIENT_4	FrontWAS2	FRONT-WEB
HttpResponseException	Internal Server Error									
open(HttpRoute route, HttpContext context, HttpPa	dev-pinpoint-workload003.ncl:8080	16:40:03 443	0	1		1	AbstractPooledCon..	HTTP_CLIENT_4	FrontWAS2	FRONT-WEB
execute(HttpRequest request, HttpClientConnection	/backendapi.pinpoint	16:40:03 444	0	156		156	HttpRequestExecut..	HTTP_CLIENT_4	FrontWAS2	FRONT-WEB
http.status.code	500									
http.io	write=0, read=156									
Tomcat Servlet Process	/backendapi.pinpoint	16:40:03 445	1	154		0		TOMCAT	ApiWAS1	BACKEND-API
REMOTE_ADDRESS	10.101.55.177									
invoke(Request request, Response response)		16:40:03 445	0	154		144	StandardHostValve	TOMCAT_METHOD	ApiWAS1	BACKEND-API
doPost(HttpServletRequest request, HttpSer		16:40:03 445	0	10		2	FrameworkServlet	SPRING	ApiWAS1	BACKEND-API
NestedServletException	Request processing failed; nested exception is jc									
backendapi()		16:40:03 446	1	8		0	DemoController	SPRING_BEAN	ApiWAS1	BACKEND-API
RuntimeException	cubrid.jdbc.driver.CUBRIDException: Syntax: Unknc									
getConnection()		16:40:03 446	0	0		0	BasicDataSource	DBCP	ApiWAS1	BACKEND-API
setAutoCommit(boolean autoCommitFlag)	false	16:40:03 446	0	2		2	ConnectionImpl	MYSQL(MySQL)	ApiWAS1	BACKEND-API
list()		16:40:03 448	0	1		0	MemberServiceImpl	SPRING_BEAN	ApiWAS1	BACKEND-API

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

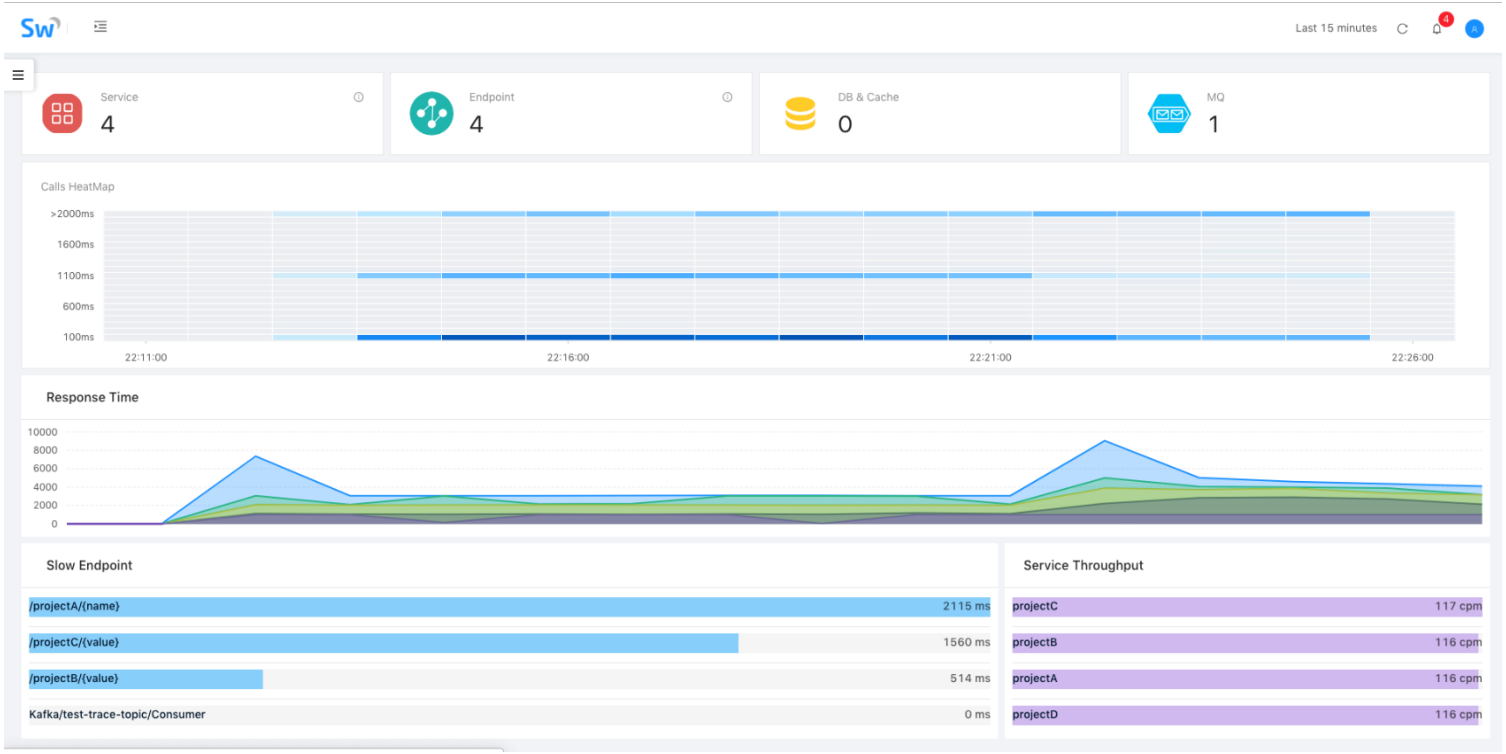
PINPOINT (http://naver.github.io/pinpoint/)



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

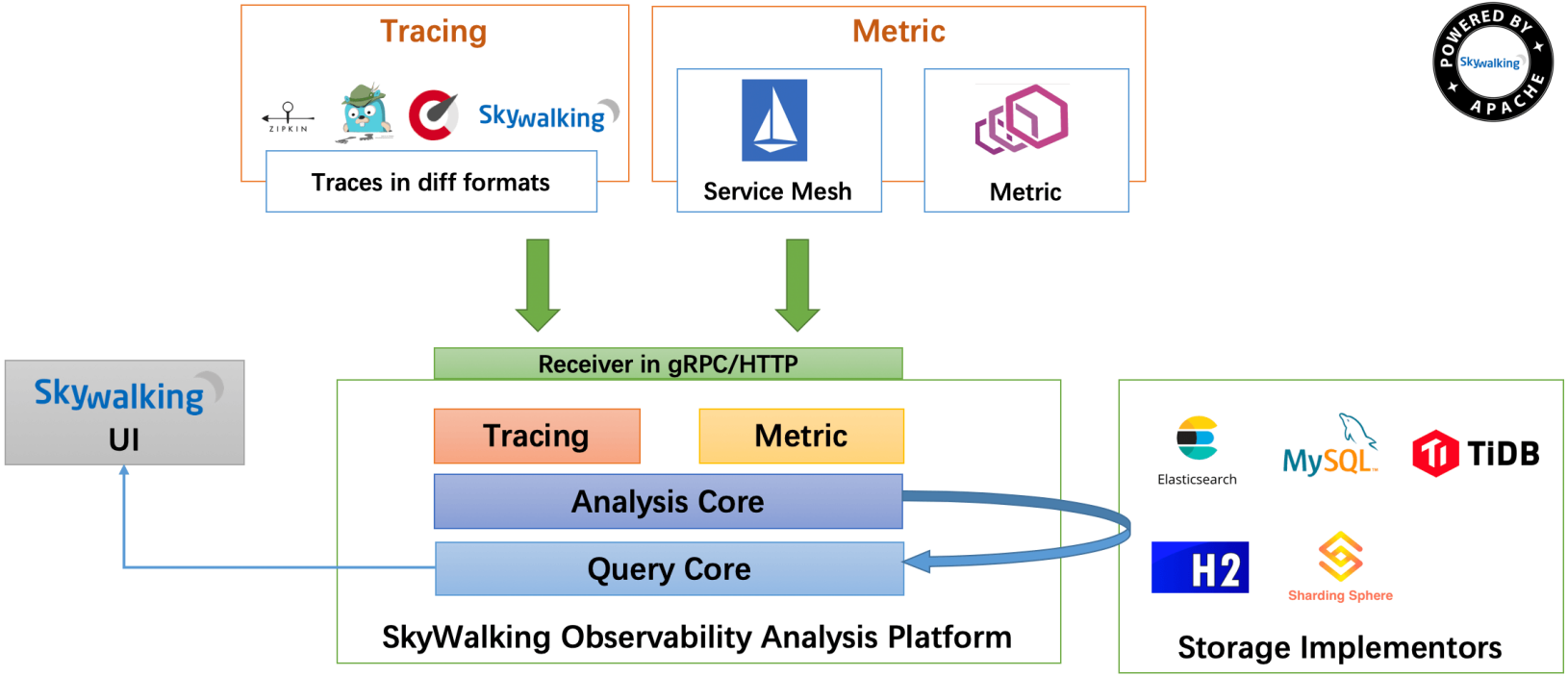
Apache Skywalking (skywalking.apache.org)

OPENTRACING



Source: https://github.com/apache/incubator-skywalking/blob/master/docs/Screenshots.md#agent

Apache Skywalking (skywalking.apache.org)



Source: https://github.com/apache/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

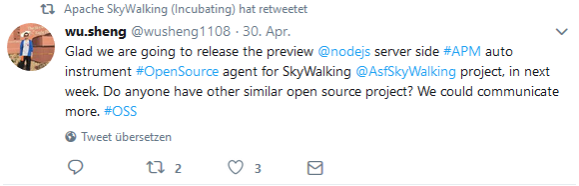
Service Discovery
- Netflix Eureka

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

appdash Traces Docs GitHub

Trace 8e922a2960cac530 (Permalink | Export as JSON)

Serve /api-ca... Serve /api-calls (637ms)
Request local... Request localhost:8699 (260ms)
Serve /endpoi... Serve /endpoint-B (75ms)
Request local... Request localhost:8699 (76ms)
Serve /endpoi... Serve /endpoint-C (300ms)
Request local... Request localhost:8699 (301ms)
Serve /endpoi... Serve /endpoint-A (252ms)

Serve /endpoint-A

Sort By start time | Sort By end time | Sort By duration | Sort By label

Data View | Verbose Data View | Profile View

- **Serve /api-calls**

Server.Response.StatusCode	200

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

Pet Clinic A Spring Framework Demonstration Spring

stagemonitor

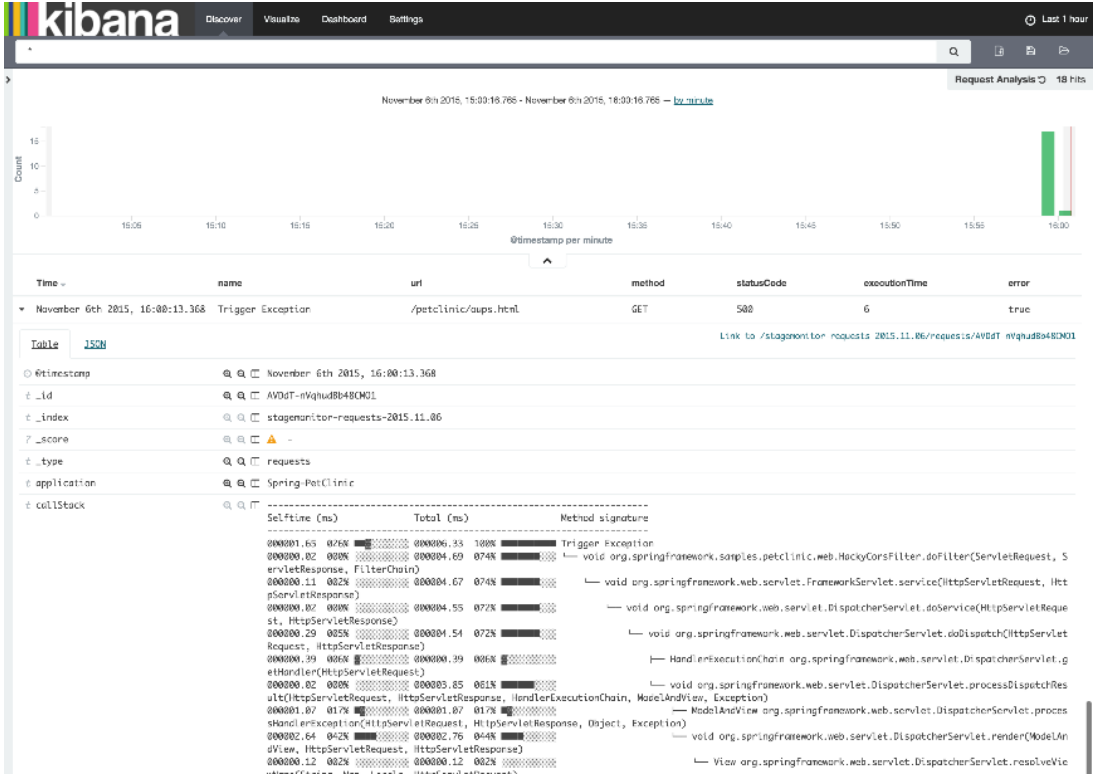
Call Tree | Request | Configuration

Signature	Total time (ms)	Self time (ms)
Show Owner	8.24	0.04
void org.springframework.web.servlet.FrameworkServlet.service(HttpServletRequest, HttpServletResponse)	8.2	0
void org.springframework.web.servlet.FrameworkServlet.doGet(HttpServletRequest, HttpServletResponse)	8.2	0.05
void org.springframework.web.servlet.DispatcherServlet.doService(HttpServletRequest, HttpServletResponse)	8.14	0.01
void org.springframework.web.servlet.DispatcherServlet.doDispatch(HttpServletRequest, HttpServletResponse)	8.13	0.68
ModelAndView org.springframework.samples.petclinic.web.OwnerController.showOwner(int)	3.43	0.52
Owner org.springframework.samples.petclinic.service.ClinicServiceImpl.findOwnerById(int)	2.91	0.04
Owner org.springframework.samples.petclinic.repository.jpa.JpaOwnerRepositoryImpl.findById(int)	2.87	2.87
select owner0_.id as id1_0_0_, pets1_.id as id1_1_1_, owner0_.first_name as first_na2_0_0_, owner0_.last_name as last_nam3_0_0_, owner0_.address as address4_0_0_, owner0_.city as city5_0_0_, owner0_.telephone as telephon6_0_0_, pets1_.name as name2_1_1_, pets1_.birth_date as birth_da3_1_1_, pets1_.owner_id as owner_id4_1_1_, pets1_.type_id as type_id5_1_1_, pets1_.owner_id as owner_id4_0_0__, pets1_.id as id1_1_0__ from owners owner0_ left outer join pets pets1_ on owner0_.id=pets1_.owner_id where owner0_.id=?	0	0
select pettype0_.id as id1_3_0_, pettype0_.name as name2_3_0_ from types pettype0_ where pettype0_.id=?	0	0
select visits0_.pet_id as pet_id4_1_0_, visits0_.id as id1_6_0_, visits0_.id as id1_6_1_, visits0_.visit_date as visit_da2_6_1_, visits0_.description as descript3_6_1_, visits0_.pet_id as pet_id4_6_1_ from visits visits0_ where visits0_.pet_id=?	0	0
void org.springframework.web.servlet.view.AbstractView.render(Map, HttpServletRequest, HttpServletResponse)	4.02	4.02

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

```
┌──────────────────┐       ┌──────────────┐      ┌──────────────────────────┐
│ Your Java        │       │Elasticsearch│      │ Kibana/Grafana           │
│ application      │       │(or Graphite/│      │ The Kibana and Grafana   │
│ ┌──────────────┐ │       │ InfluxDB)   │      │ dashboards visualize     │
│ │stagemonitor ├─┼──┬────>             <──────┤ metrics and let you      │
│ └──────────────┘ │  │    │              │      │ analyze captured requests │
└────────┬─────────┘  │    └──────────────┘      └──────────────────────────┘
         │            │
         │            │
         │      The stagemonitor agent sits inside your
         │      application and sends metrics and details
         │      about requests (request traces) to Elasticsearch.
         │      Metrics can optionally be sent to Graphite, InfluxDB
         │      and Prometheus.
┌────────V──────────┐
│ In browser widget │
│                   │
│ Lets you analyze  │
│ metrics and call  │
│ trees in your     │
│ browser without   │
│ installing a      │
│ backend           │
└───────────────────┘
```

Supported Modules:

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

InspectIT (inspectit.rocks)

OPENTRACING

File Window Help

Repository Manage | Storage Manager | Data Explorer

Local CMR

inspectITDemo [n/a]

- Instrumentation Browser
- Invocation Sequences
  - Show All
  - Browser
- SQL Statements
- Timer Data
- Http Timer Data
- Exceptions
- System Overview

Show All

Local CMR ▸ inspectITDemo [n/a] ▸ Invocation Sequences ▸ Show All

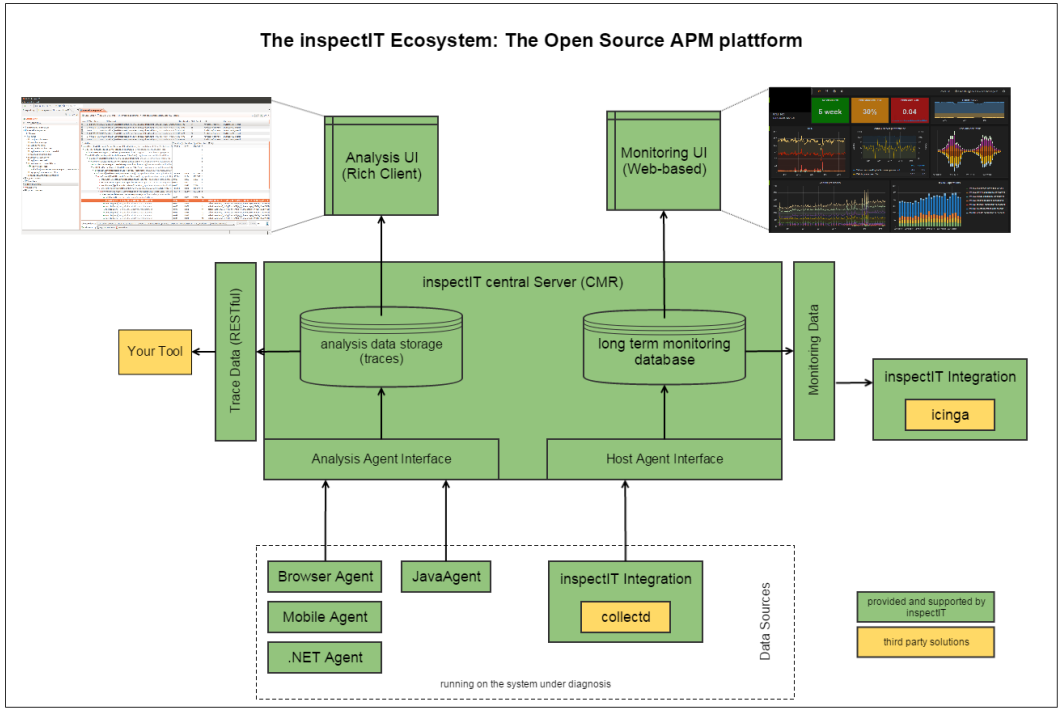
Start Time	Method	Duration (ms	Child Count	URI	Use case
04.09.2012 17:20:04.43	doFilter(ServletRequest, ServletResponse, FilterChain) - org.jboss.web.tomcat.filters.Repl	437.868	84	/dvdstore/checkout	
04.09.2012 17:20:00.25	doFilter(ServletRequest, ServletResponse, FilterChain) - org.jboss.web.tomcat.filters.Repl	4172.216	122	/dvdstore/checkout	
04.09.2012 17:19:59.83	doFilter(ServletRequest, ServletResponse, FilterChain) - org.jboss.web.tomcat.filters.Repl	399.276	78	/dvdstore/browse	
04.09.2012 17:19:55.03	doFilter(ServletRequest, ServletResponse, FilterChain) - org.jboss.web.tomcat.filters.Repl	4777.570	177	/dvdstore/browse	Search
04.09.2012 17:19:52.32	doFilter(ServletRequest, ServletResponse, FilterChain) - org.jboss.web.tomcat.filters.Repl	2668.745	128	/dvdstore/browse	
04.09.2012 17:19:50.10	doFilter(ServletRequest, ServletResponse, FilterChain) - org.jboss.web.tomcat.filters.Repl	2212.059	40	/dvdstore/home	
04.09.2012 17:19:38.80	doFilter(ServletRequest, ServletResponse, FilterChain) - org.jboss.web.tomcat.filters.Repl	11216.400	25	/dvdstore/home	Home
04.09.2012 17:19:38.22	doFilter(ServletRequest, ServletResponse, FilterChain) - org.jboss.web.tomcat.filters.Repl	430.847	9	/dvdstore/	

Method	Duration (ms	Exc. duration (r	Cpu Duration ((	Start Delta (m	SQL
doFilter(ServletRequest, ServletResponse, FilterChain) - org.jboss.web.tomcat.filters.ReplyHeaderFilter	4777.570	46.982	330.000	0	
doFilter(ServletRequest, ServletResponse, FilterChain) - org.jboss.seam.servlet.SeamFilter				0	
doFilter(ServletRequest, ServletResponse, FilterChain) - org.jboss.seam.web.HotDeployFilter				0	
doFilter(ServletRequest, ServletResponse, FilterChain) - org.jboss.seam.web.RedirectFilter				46	
doFilter(ServletRequest, ServletResponse, FilterChain) - org.jboss.seam.web.ExceptionFilter				46	
doFilter(ServletRequest, ServletResponse, FilterChain) - org.jboss.seam.web.MultipartFilter				46	
doFilter(ServletRequest, ServletResponse, FilterChain) - org.jboss.seam.web.IdentityFilter				46	
doFilter(ServletRequest, ServletResponse, FilterChain) - org.jboss.seam.web.LoggingFilt				46	
doFilter(ServletRequest, ServletResponse, FilterChain) - org.tuckey.web.filters.urlrew				47	
forward(ServletRequest, ServletResponse) - org.apache.catalina.core.ApplicationD	4730.588	0.230	290.000	47	
doForward(ServletRequest, ServletResponse) - org.apache.catalina.core.Applic	4730.358	2.096	290.000	47	
checkSameObjects(ServletRequest, ServletResponse) - org.apache.catalina.c	0.039	0.039	0.000	47	
wrapResponse(ApplicationDispatcher$State) - org.apache.catalina.core.Appl	0.038	0.038	0.000	47	
wrapRequest(ApplicationDispatcher$State) - org.apache.catalina.core.Applic	0.053	0.053	0.000	47	
processRequest(ServletRequest, ServletResponse, ApplicationDispatcher$S	4728.131	0.122	290.000	47	
invoke(ServletRequest, ServletResponse, ApplicationDispatcher$State) -	4728.009	399.389	290.000	47	
service(ServletRequest, ServletResponse) - javax.faces.webapp.FacesS				47	
unwrapRequest(ApplicationDispatcher$State) - org.apache.catalina.co	0.107	0.107	0.000	4775	
unwrapResponse(ApplicationDispatcher$State) - org.apache.catalina.c	0.038	0.038	0.000	4775	
recycleRequestWrapper(ApplicationDispatcher$State) - org.apache.cal	0.038	0.038	0.000	4775	

Call Hierarchy | SQL | Methods | Exceptions

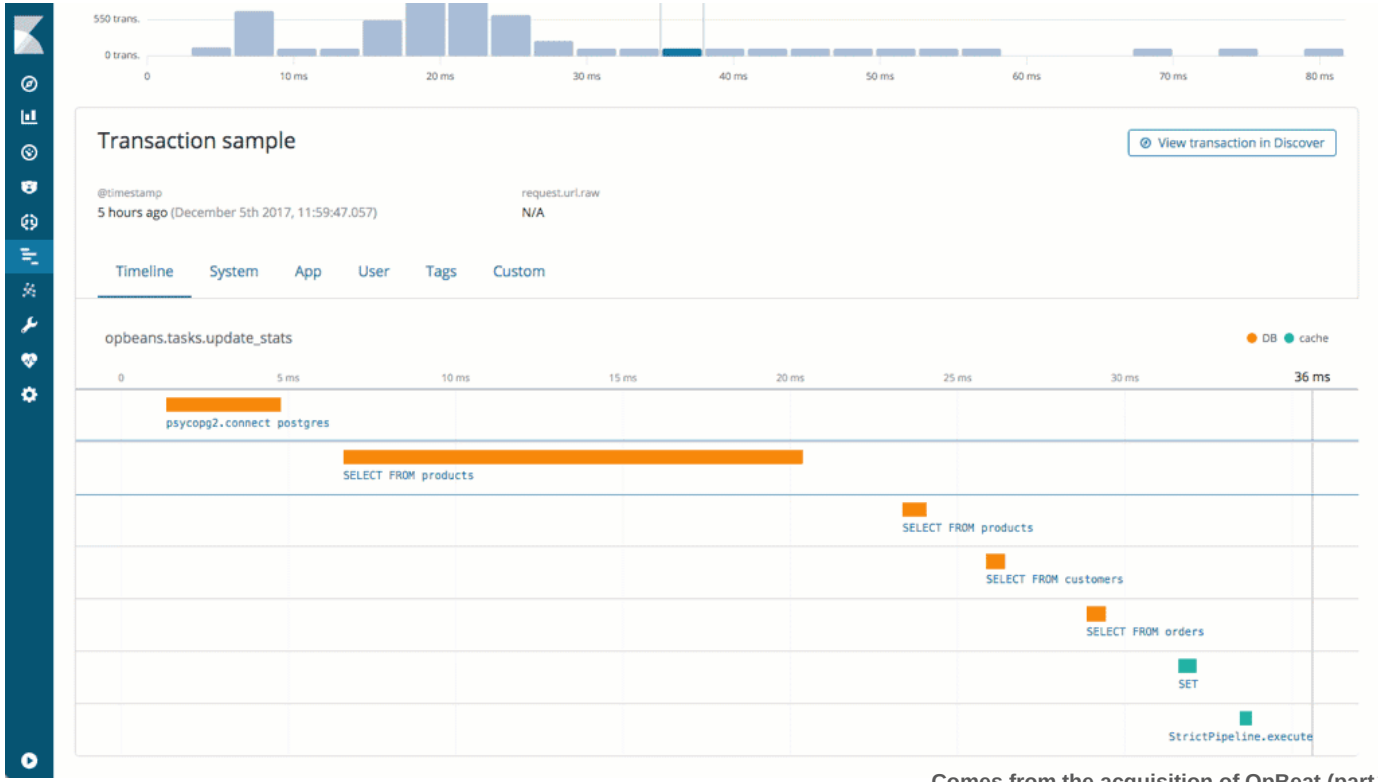
Source: https://inspectit-performance.atlassian.net/wiki/spaces/DOC18/pages/93009319/Working+with+invocation+sequences

InspectIT (inspectit.rocks)



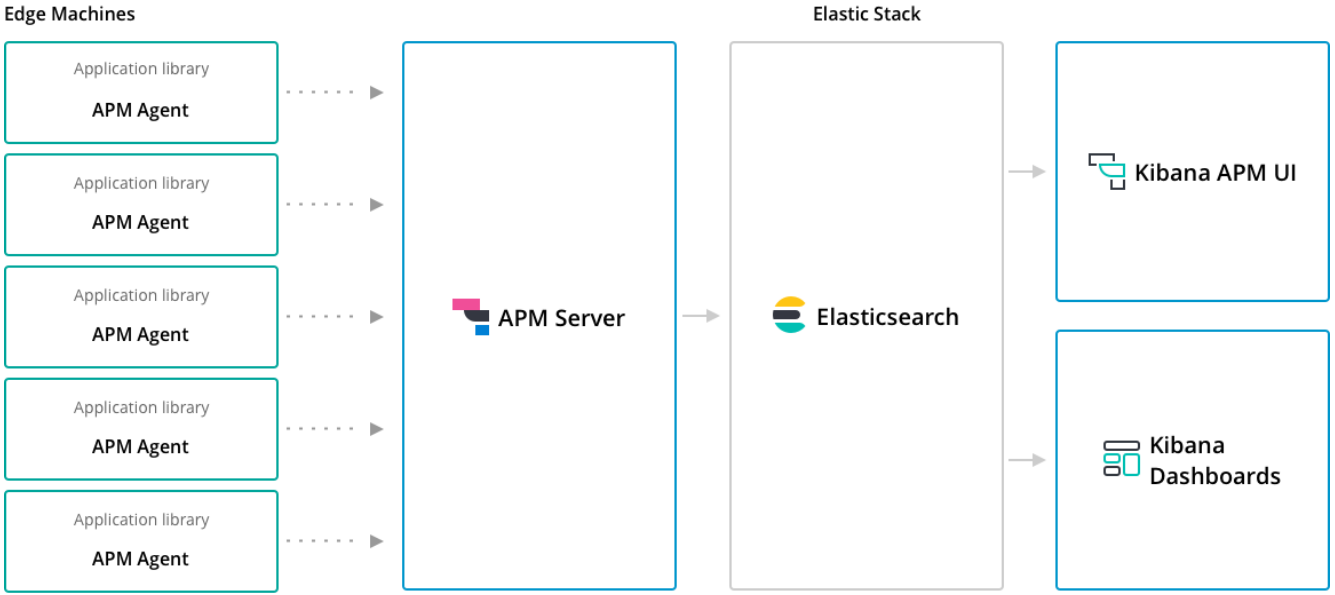
Supported Languages: Java, (.NET)

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, Go, Java, .NET (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, …)

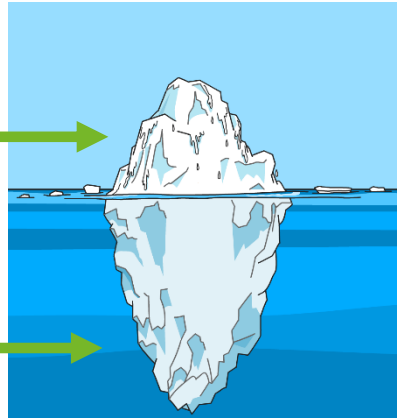
- **You can plan ahead**:
  - Vendors typically communicate the time until which a software version is supported and support the transition phase as well, this is not always the case for open source software

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 Thorntail (previously WildFly Swarm) to automatically capture traces that can be packaged in your application

Thank you!

Dr. Andreas Brunnert
brunnert@retit.de

RETIT

Resource Efficient Technologies & IT Systems