

Python For Good

Python Microservice Application Performance Monitoring

柯振旭

Engineer,
Tetrade.io

柯振旭

@kezhenxu94



- Tetrade.io Engineer. Prev. Alibaba
- Open-source Enthusiast
 - Apache SkyWalking PMC Member
 - Apache Dubbo Committer
 - Apache Local Community Beijing (ALC) Member



Agenda

- Why APM System
- Apache SkyWalking
- Apache SkyWalking-Python Uncovered

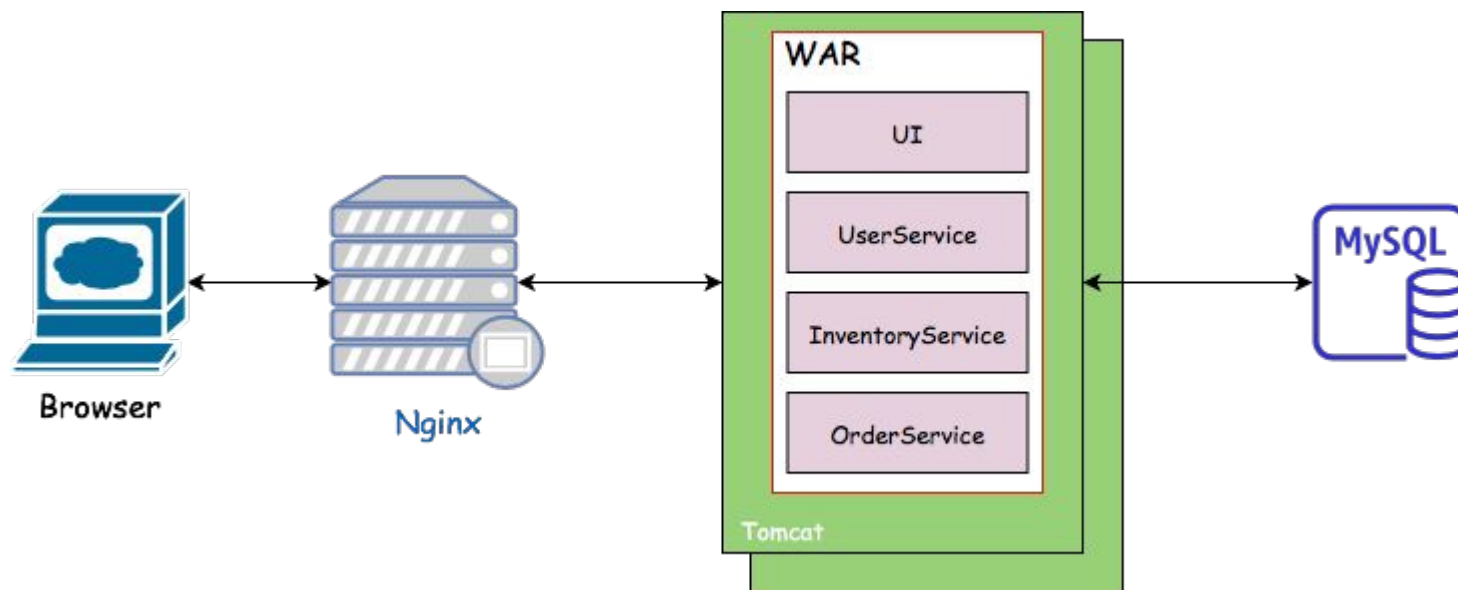
Traditional Monolithic Architecture

Easy to

Develop (Test)

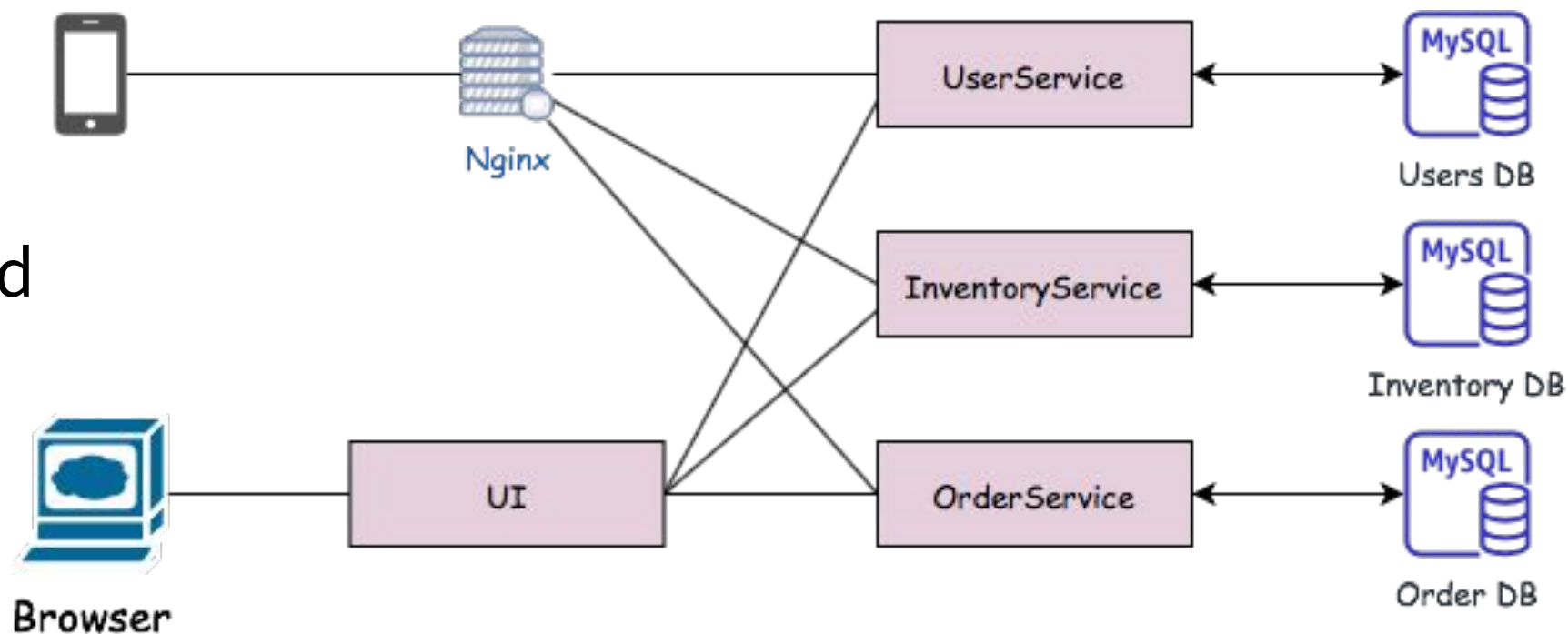
Deploy

Scale



Microservice Architecture

Difficult to
Debug
Comprehend
Deploy

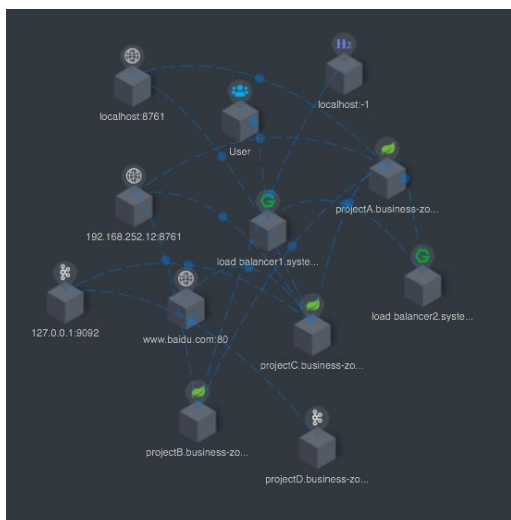


What Can APM System Do?

- **Distributed tracing and context propagation**
- **Service topology map analysis**
- **Service, service instance, endpoint metrics analysis**
- Root cause analysis. Profile the code at runtime
- Service, service instance and endpoint dependency analysis
- Slow services and endpoints detected
- Performance optimization

Apache SkyWalking

<http://122.112.182.72:8080>



A screenshot of the Apache SkyWalking dashboard showing a trace for a specific service. The top navigation bar includes Dashboard, Topology, Trace, Profile, Log, and Alarm. Below the navigation bar, there are tabs for APM, Database, SelfObservability, and Web Browser. The main content area shows a trace for the service 'projectC(value)' with a duration of 28ms. The trace includes several steps with their respective durations and status. The bottom part of the screenshot shows a detailed view of the trace, including the service name, endpoint, and the underlying database query.

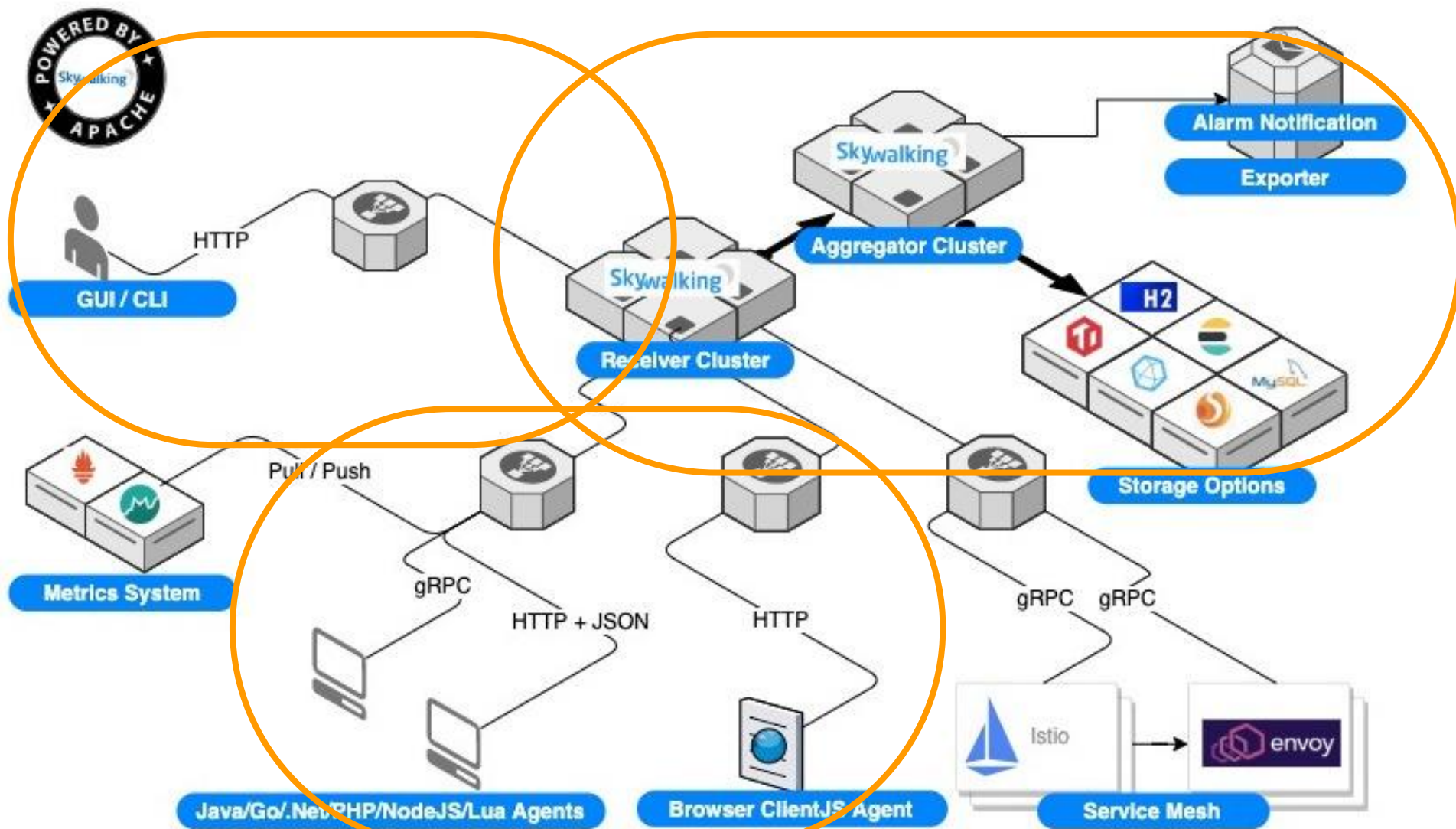
A screenshot of the Apache SkyWalking dashboard showing various performance metrics and charts. The top navigation bar includes Dashboard, Topology, Trace, Profile, Log, and Alarm. Below the navigation bar, there are tabs for APM, Database, SelfObservability, and Web Browser. The main content area shows several charts and tables:

- Services Load (CPM - calls per minute)**: A table showing the load for various services.
- Slow Services (ms)**: A table showing the slowest services.
- Un-Health Services (Apdex)**: A table showing the health status of services.
- Slow Endpoints (ms)**: A table showing the slowest endpoints.
- Global Response Latency (percentile in ms)**: A line chart showing the response latency for various percentiles (P50, P75, P90, P95, P99) over time.
- Global Heatmap (ms)**: A heatmap showing the response latency over time.

A quick look at Apache SkyWalking

2020-11-11 14:52:42 ~ 2020-11-12 14:52:42 En Server Zone UTC +8

Apache SkyWalking



Apache SkyWalking Python

The Python Agent for Apache SkyWalking, which provides the native tracing abilities for Python applications.



Quick Start

```
$ pip install apache-skywalking
```

```
from skywalking import agent, config
```

```
if __name__ == '__main__':  
    config.init(collector='127.0.0.1:11800', service='your service')  
    agent.start()  
    # your codes ...
```

Quick Start

```
$ pip install apache-skywalking
```

```
from skywalking.decorators import trace
```

```
@trace() # the operation name is the method name('my_method') by default  
def my_method():  
    print('Hello World')
```

Quick Start

```
$ pip install apache-skywalking
```

```
context: SpanContext = get_context() # get a tracing context
```

```
with context.new_entry_span(op='https://github.com/apache') as span:
```

```
    span.component = Component.Flask
```

```
    span.tag(Tag(key='Org', val='Apache'))
```

```
# context.new_exit_span()/context.new_local_span()
```

Agent Core Uncovered

```
# Client Side, Exit Span
context = get_context()
carrier = Carrier()
with context.new_exit_span(op=url_param.path or "/", peer=url_param.netloc, carrier=carrier) as span:
    for item in carrier:
        headers[item.key] = item.val

# Server Side, Entry Span
context = get_context()
carrier = Carrier()
for item in carrier:
    sw_http_header_key = ...
    if sw_http_header_key in request.META:
        item.val = request.META[sw_http_header_key]
```

Agent Plugin Uncovered

Everything in Python is an object
Nearly all objects are mutable in Python

Agent Plugin Uncovered

```
from django.core.handlers.base import BaseHandler

_get_response = BaseHandler.get_response

def _sw_get_response(this, request):
    context = get_context()
    carrier = Carrier()
    # ...
    with context.new_entry_span(op=request.path, carrier=carrier) as span:
        resp = _get_response(this, request)
        # ...
        return resp

BaseHandler.get_response = _sw_get_response
```

Plugins

- http.server
- urllib.request
- requests
- Flask
- PyMySQL
- Django
- redis-py
- kafka-python
- tornado
- pika
- pymongo
- elasticsearch
- urllib3

THANK YOU



@kezhenxu94



kezhenxu94



kezhenxu94



@ASFSkyWalking

