

Service Develop to Deploy



Important Quality Services



Important Quality Services

Observability
Configurability
Security



How to find an issue ?

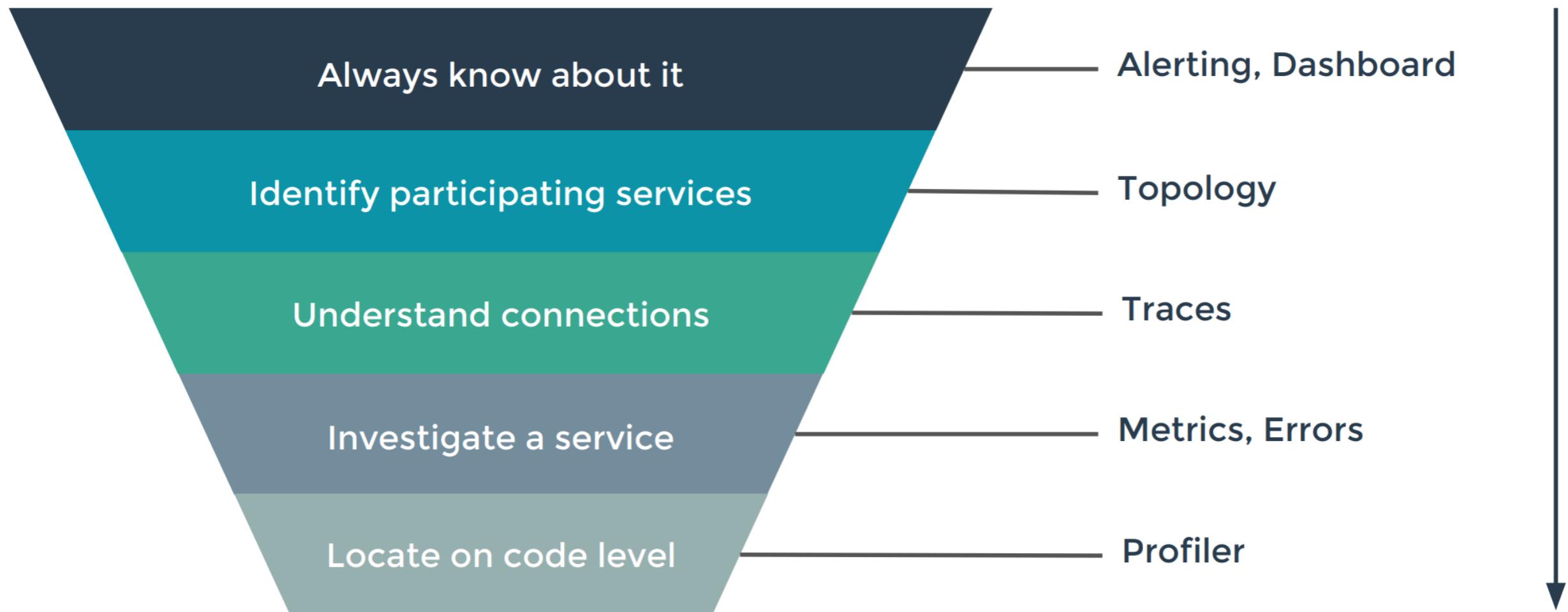
Reactive

Vs.

Proactive



How to find an issue ?



Observable services



Observability vs Monitoring



Design observable services

Health check API

Log aggregation

Distributed tracing

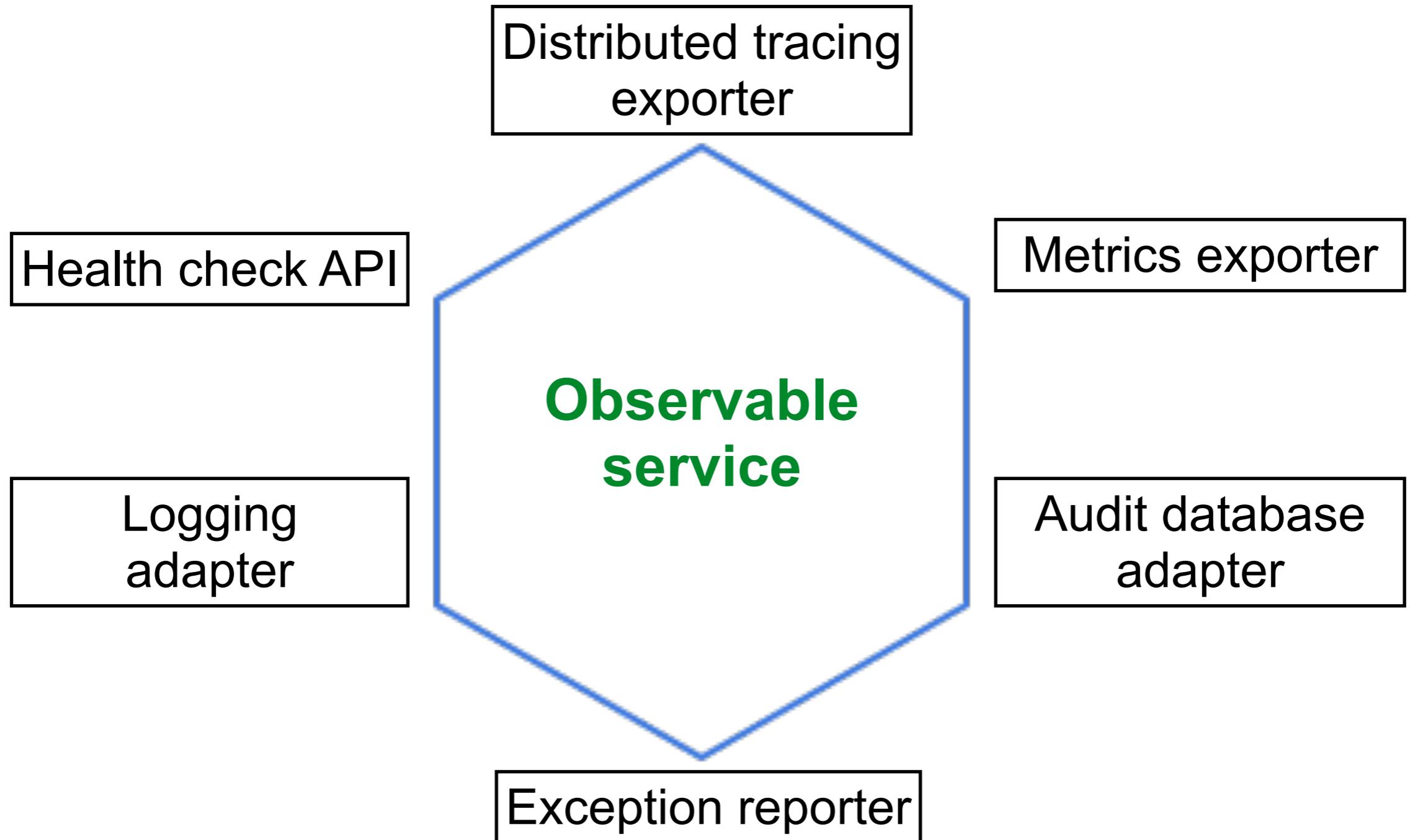
Exception tracking

Application metrics

Audit logging

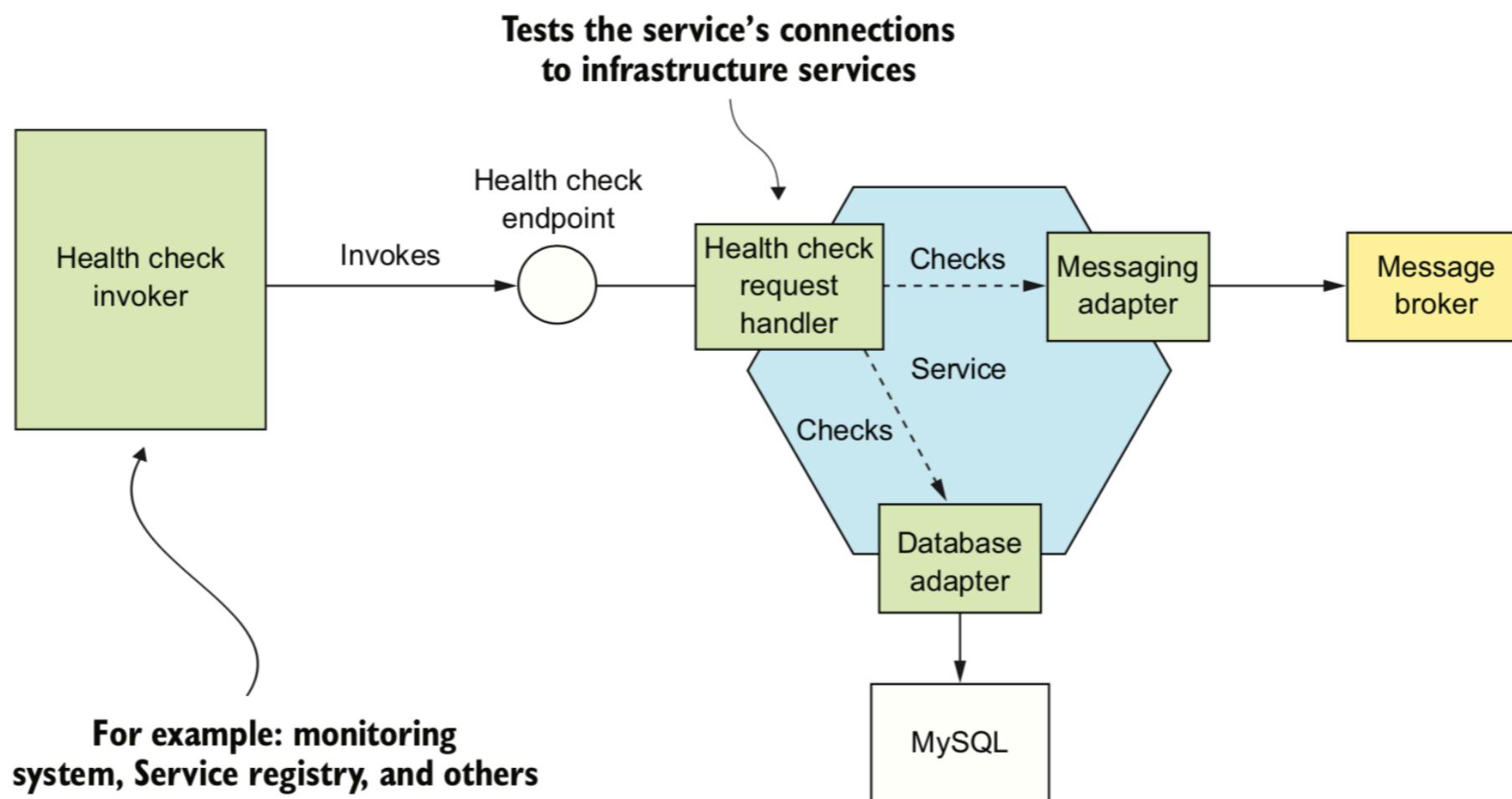


Observable services



Health check API

Expose an endpoint that return the health of service



Monitor your services

Service catalog

Uptime

Service discovery/registry



Service Catalog

The screenshot shows the Backstage Service Catalog interface. At the top, there's a header with a dark teal background. On the left, a sidebar has icons for navigation and settings. The main area has a light gray background. A banner at the top says "Bună ziua, guest!" and "Backstage Service Catalog". On the right, there are time zones for NYC, UTC, STO, and TYO. Below the banner, there are tabs for SERVICES, WEBSITES, LIBRARIES, DOCUMENTATION, and OTHER. The SERVICES tab is selected. In the center, there's a "Services" section with a "CREATE COMPONENT" button and a "SUPPORT" link. To the left of this, there's a "PERSONAL" sidebar with sections for "Owned" (3), "Starred" (0), and "SPOTIFY" (All, 6). The main content area shows a table titled "Owned (3)" with columns: NAME, OWNER, LIFECYCLE, DESCRIPTION, and ACTIONS. The table lists three services: playback-order, searcher, and shuffle-api, all owned by guest and in production lifecycle.

NAME	OWNER	LIFECYCLE	DESCRIPTION	ACTIONS
playback-order	guest	production	Playback Order	
searcher	guest	production	Searcher	
shuffle-api	guest	production	Shuffle API	

<https://backstage.io/>



Kuma Uptime

The screenshot shows the Uptime Kuma web interface. On the left, a sidebar lists several monitoring targets with their current status (e.g., Check Port, Example.com, Facebook, Google, Inbox by Gmail, MySQL, Ping) and a green progress bar. In the center, a detailed view for [LouisLam.net](https://louislam.net) is displayed. It includes a URL field, a search bar, and buttons for Pause, Edit, and Delete. A large green "Up" button indicates the service is currently up. Below this, a message says "Check every 60 seconds." A table provides performance metrics: Response (Current: 271 ms, 24-hour: 138 ms), Avg. Response (24-hour: 138 ms), Uptime (24-hour: 100%, 30-day: 100%), and Cert Exp. (2022-06-23). A line graph at the bottom tracks response time in milliseconds over time, showing a significant spike around 21:43.

<https://github.com/louislam/uptime-kuma>



Microservices

© 2022 - 2023 Siam Chamnankit Company Limited. All rights reserved.

Service Discovery

The screenshot displays a service discovery dashboard with several service instances listed:

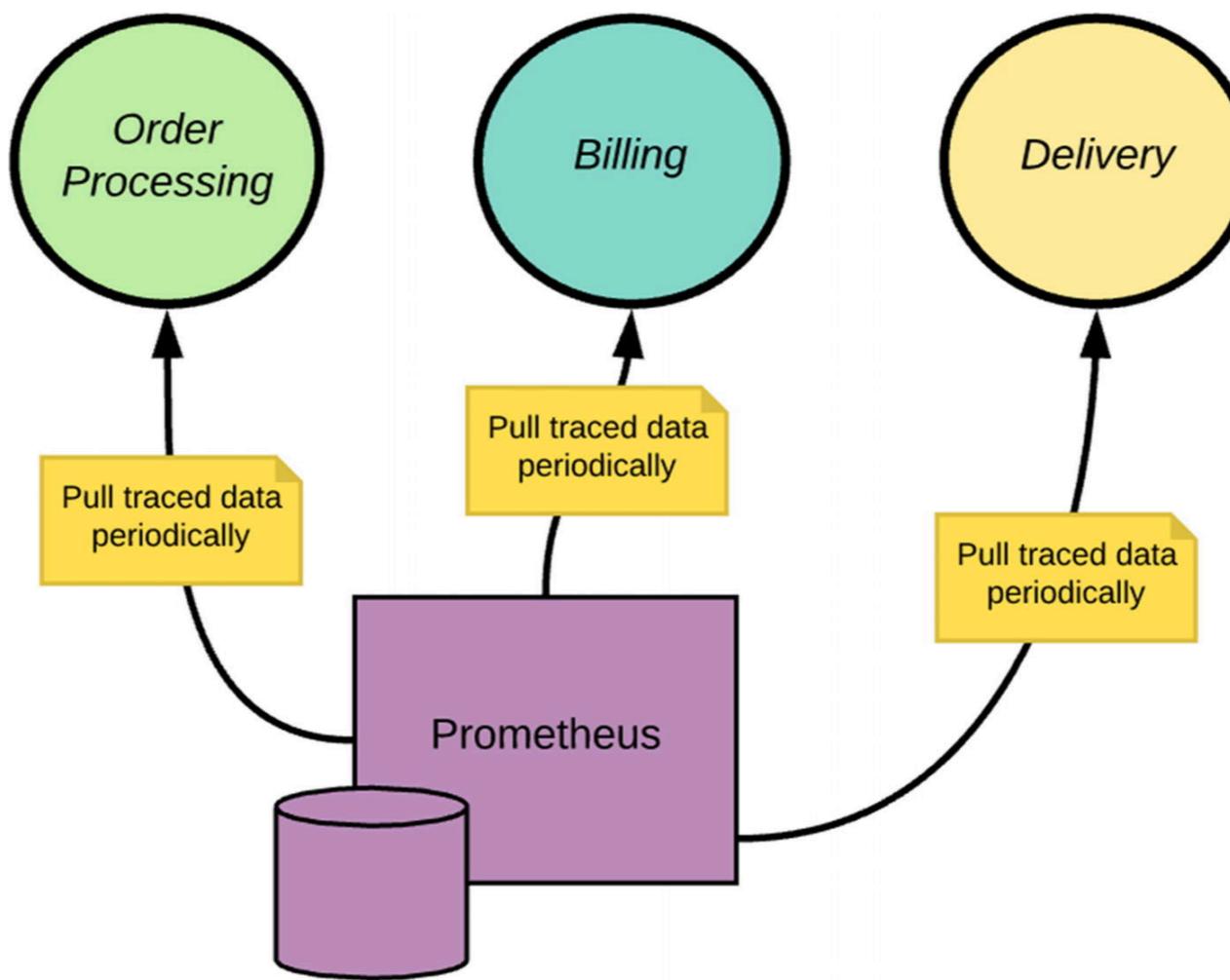
- api-auth-v2-production-10-120-10-87**
165.227.122.36
 - ✖ Load avg
 - ! File Descriptor Utilization
 - ✓ 12 other passing checks
- api-auth-10-120-10-23**
165.227.122.36
 - ✖ Serf health check
 - ✓ 12 other passing checks
- api-auth-10-120-123-43**
165.227.122.36
 - ✖ Serf health check
 - ! Memory usage
 - ! File Descriptor Utilization
 - ✓ 12 other passing checks
- v2-10-120-123-43**
165.227.122.36
 - ✖ Serf health check
 - ! Memory usage
 - ! Memory usage
 - ! File Descriptor Utilization
 - ✓ 12 other passing checks
- Api Client v3**
165.227.38.231
 - ✖ Serf health check
 - ✓ 12 other passing checks
- api-server-10-23-138-30**
165.227.122.36
 - ✖ Serf health check
 - ✖ Memory usage
 - ✓ 2 other passing checks
- consul-server-20-294**
165.227.122.36
 - ✖ Memory usage
- consul-client-20-294-357-34**
165.227.122.36
 - ✖ Serf Health Check

<https://www.consul.io/>



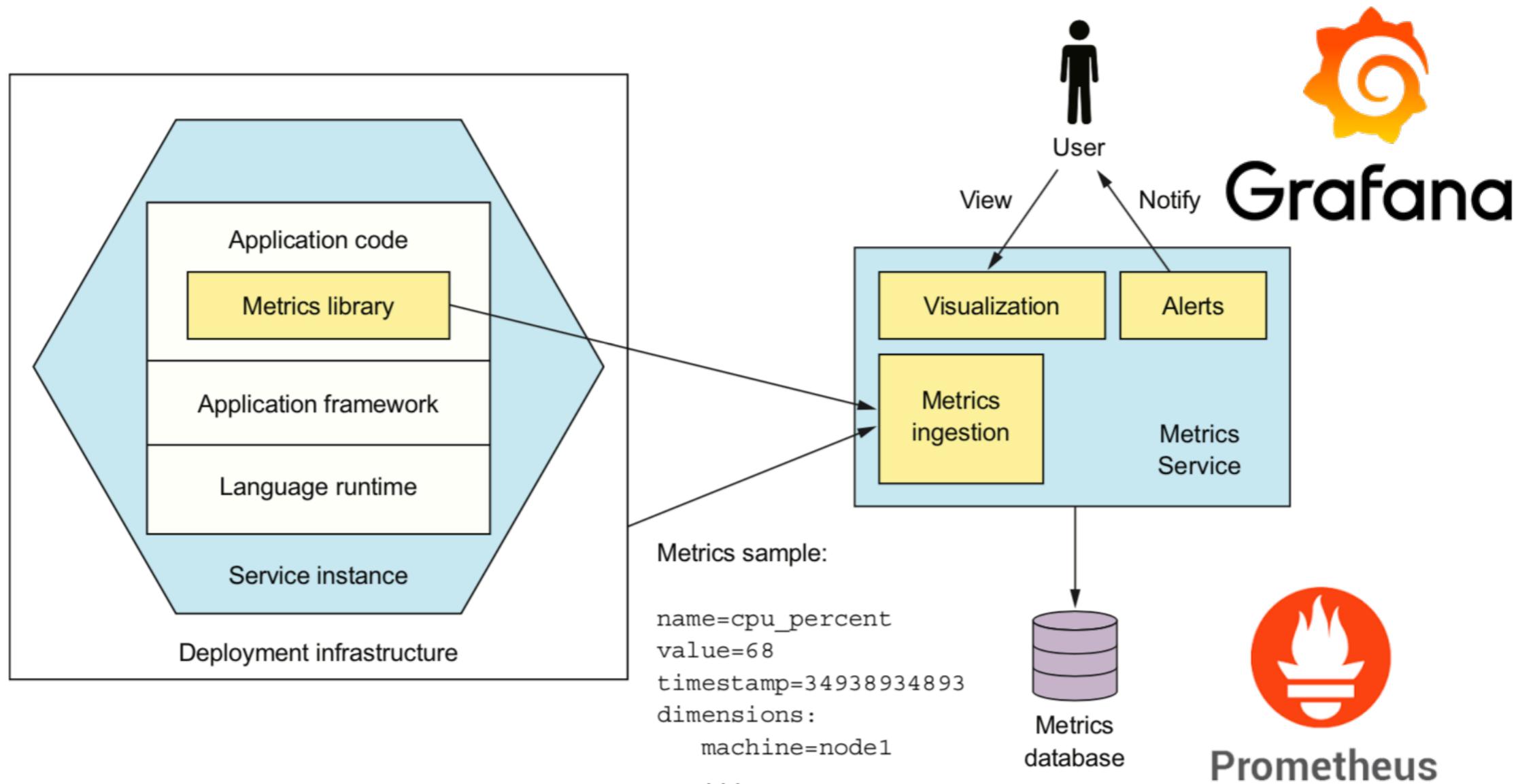
Application metrics

Services maintain metrics and expose to metric server
(counters, gauges)



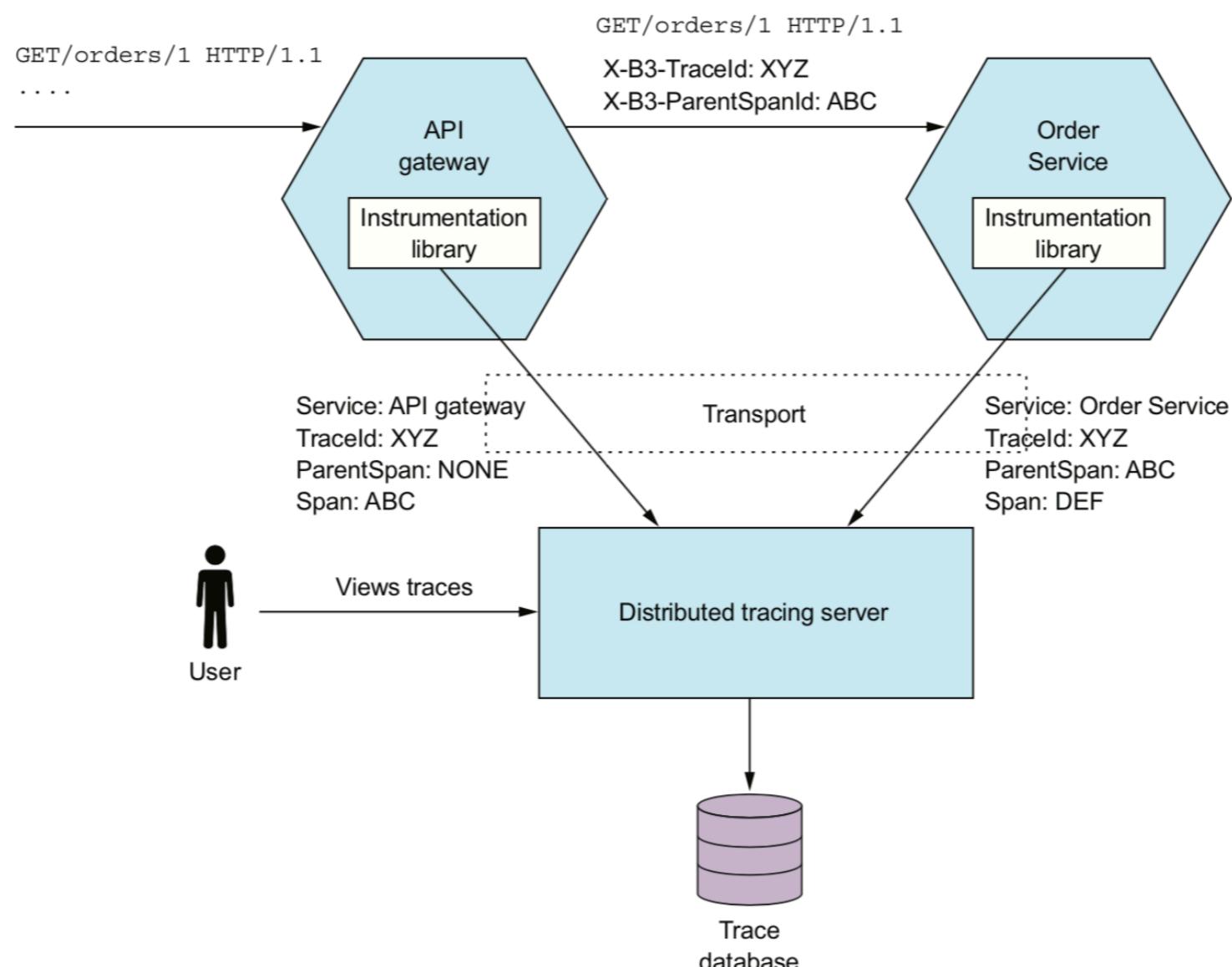
Application metrics

Services maintain metrics and expose to metric server
(counters, gauges)

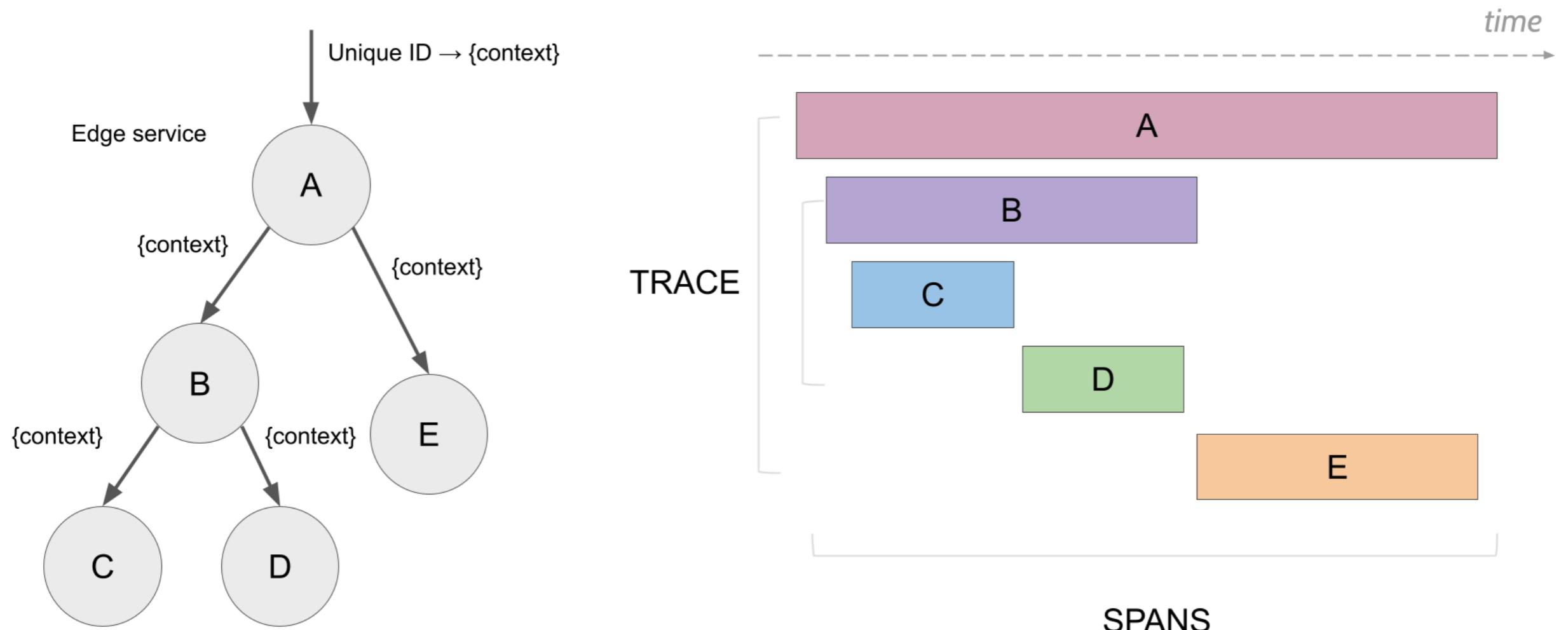


Distributed tracing

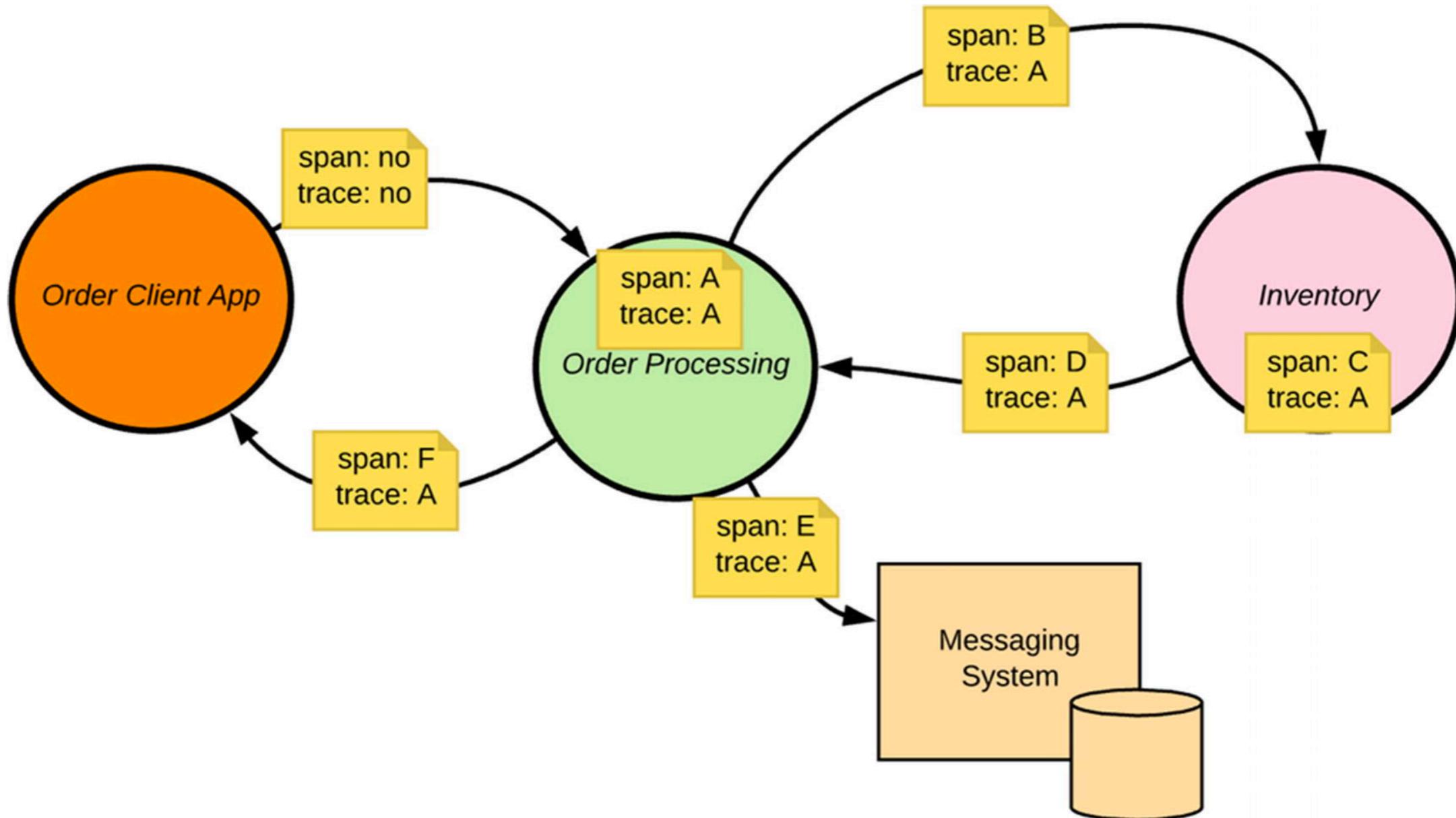
Assign each external request a **unique ID** and trace requests as flow between services



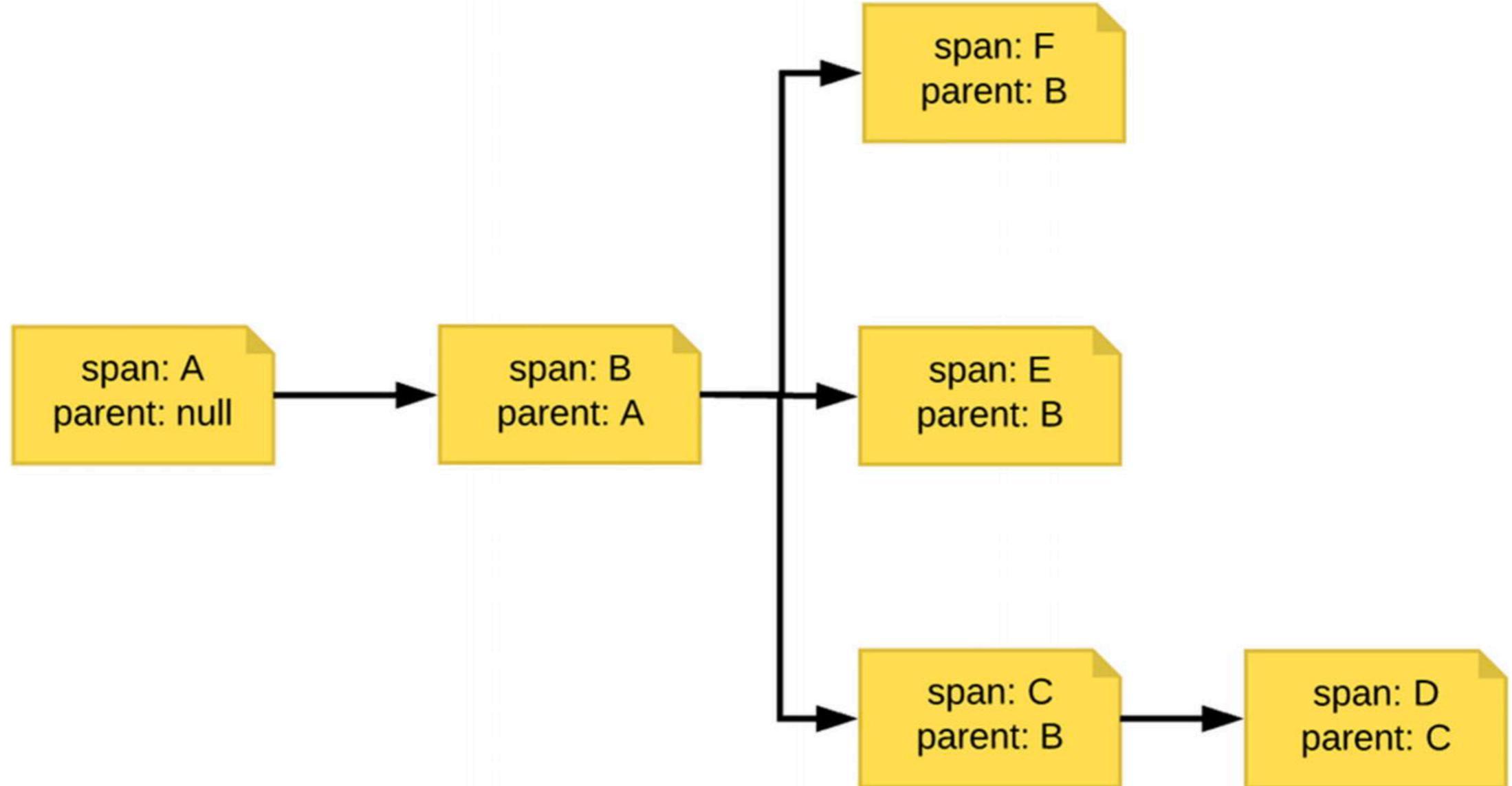
Distributed tracing



Distributed tracing



Distributed tracing



Distributed tracing tools

Format standard with OpenTelemetry
Zipkin, Jaeger, Tempo, AWS X-Ray, Elastic APM



JAEGER



Grafana Tempo

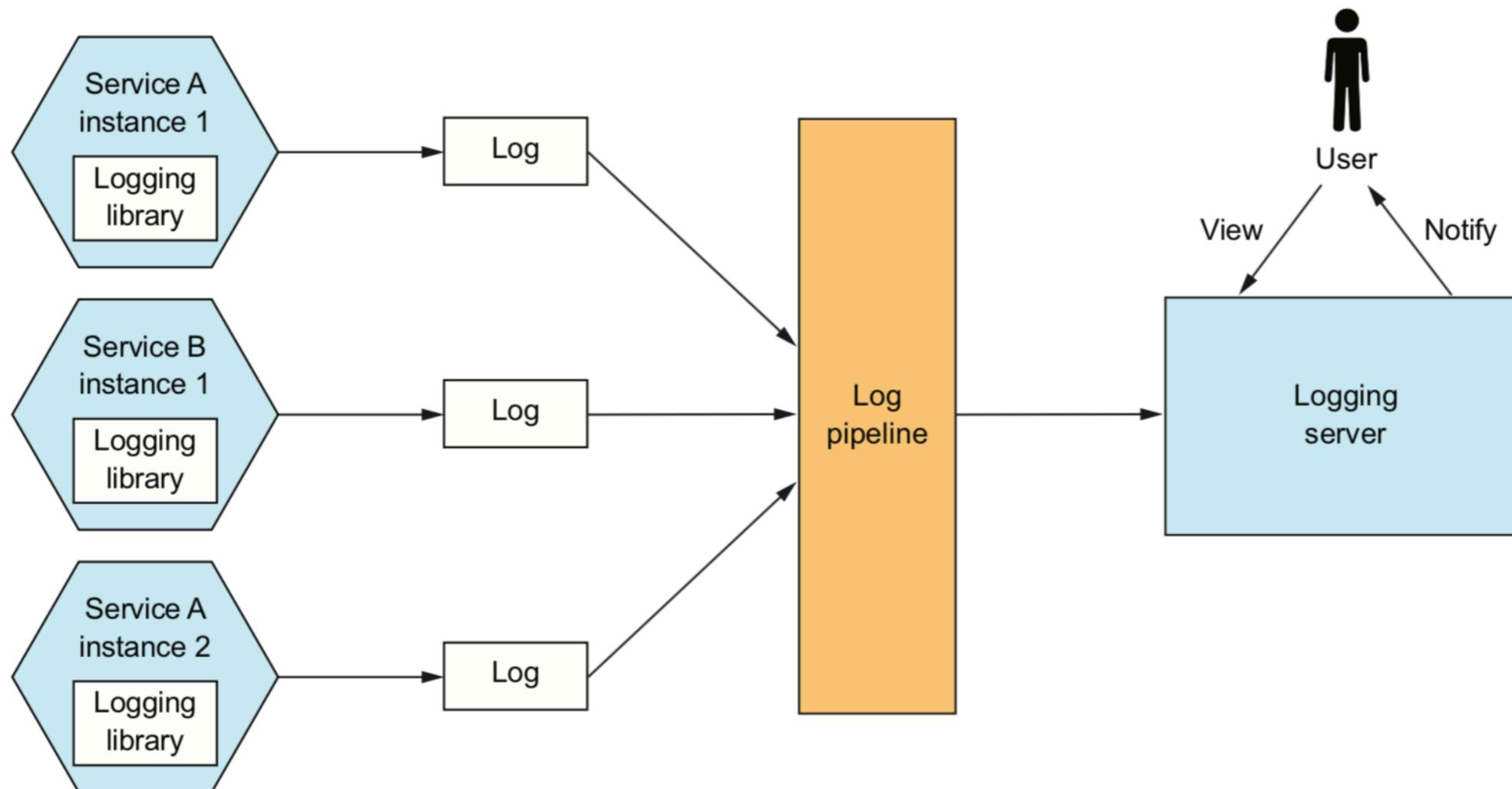


OpenTelemetry



Log aggregation

Log service activity and write logs into a centralized logging server. (searching, alerting)



Effective Log Aggregation

- Define event to log
- Use structured logging
- Exclude sensitive information
- Log at the correct level
- Be specific in your message
- Don't log large message
- Make sure you keep trace Id in the log

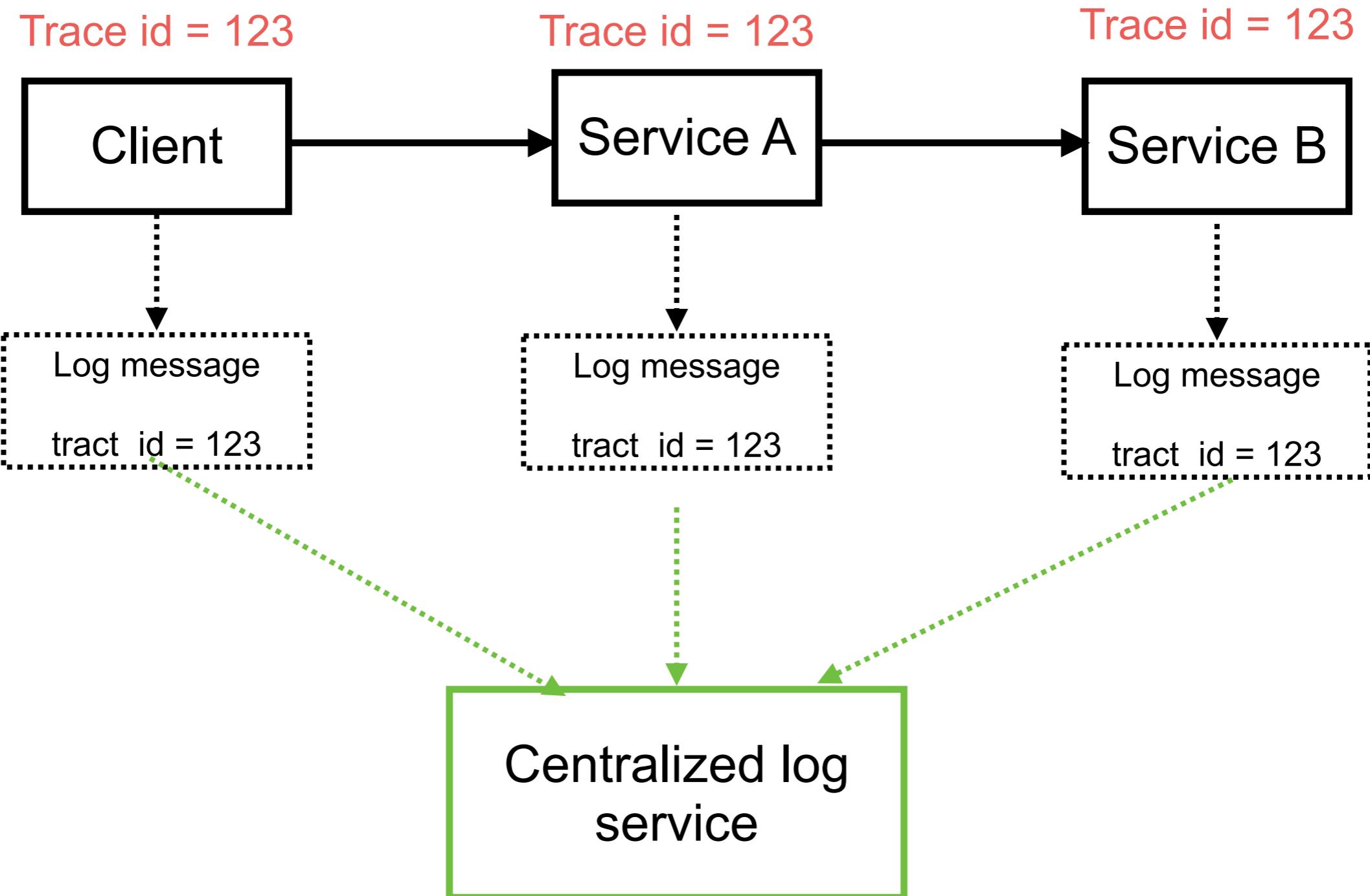
https://github.com/OWASP/CheatSheetSeries/blob/master/cheatsheets/Logging_Cheat_Sheet.md



Microservices

© 2022 - 2023 Siam Chamnankit Company Limited. All rights reserved.

Log aggregation



Consistent Structure across all logs

Property	Description	Example
Timestamp	Date and time of the log	2023-07-01
Log level	DEBUG, INFO, ERROR	
Trace Id or Correlation Id	Unique identifier that refer to other logs from all services	
Event/Action Name	Identify to event or action of log	Authentication fail
Service ID/ Name	Identify to service	
Request path	Path for the request	/api/products



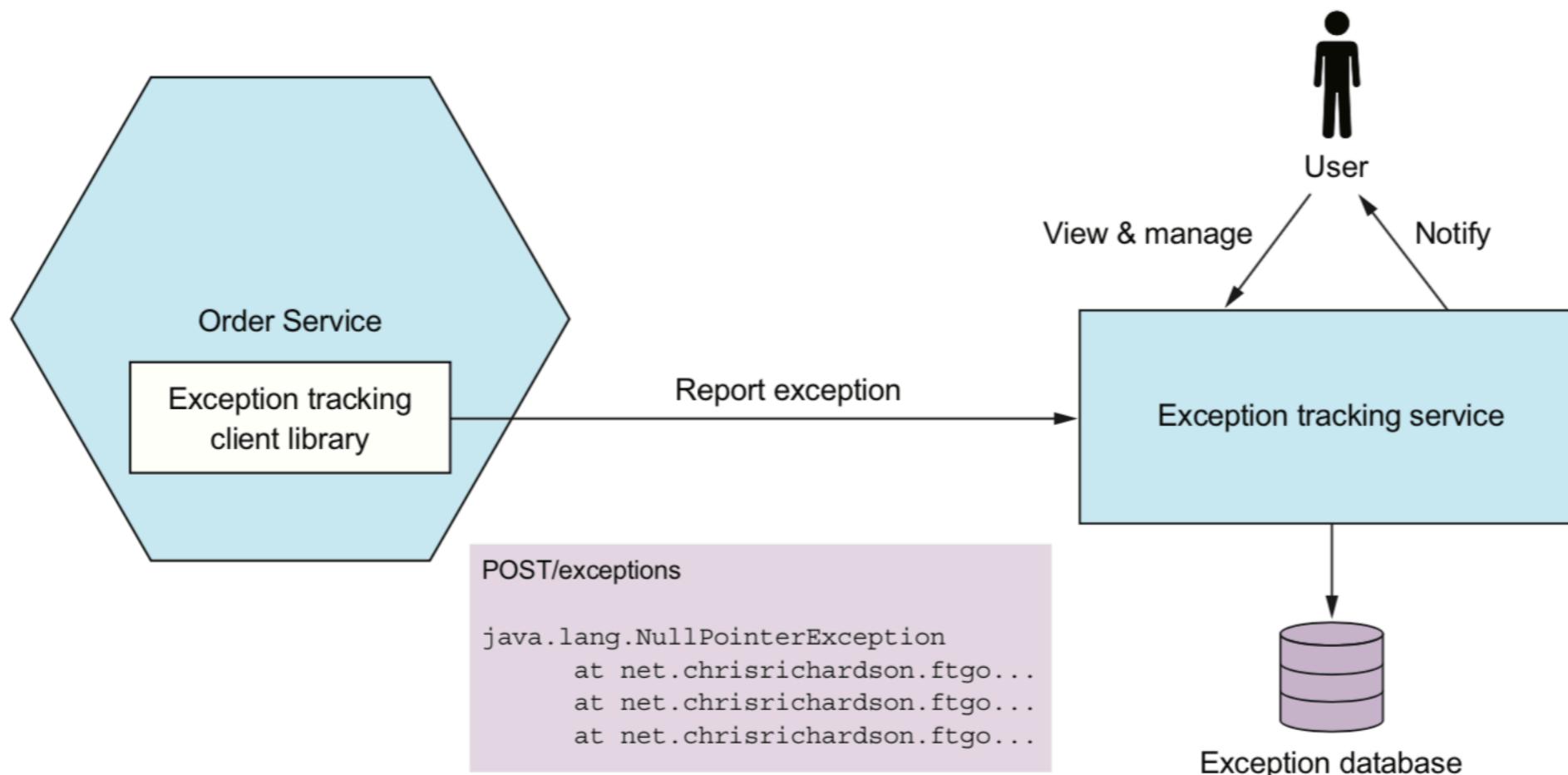
Don't keep !!

```
com.framework.FrameworkException: Error in web request
  at com.framework.ApplicationStarter.lambda$start$0(ApplicationStarter.java:15)
  at spark.RouteImpl$1.handle(RouteImpl.java:72)
  at spark.http.matching.Routes.execute(Routes.java:61)
  at spark.http.matching.MatcherFilter.doFilter(MatcherFilter.java:134)
  at spark.embeddedserver.jetty.JettyHandler.doHandle(JettyHandler.java:50)
  at org.eclipse.jetty.server.session.SessionHandler.doScope(SessionHandler.java:1568)
  at org.eclipse.jetty.server.handler.ScopedHandler.handle(ScopedHandler.java:144)
  at org.eclipse.jetty.server.handler.HandlerWrapper.handle(HandlerWrapper.java:132)
  at org.eclipse.jetty.server.Server.handle(Server.java:503)
  at org.eclipse.jetty.server.HttpChannel.handle(HttpChannel.java:364)
  at org.eclipse.jetty.server.HttpConnection.onFillable(HttpConnection.java:260)
  at org.eclipse.jetty.io.AbstractConnection$ReadCallback.succeeded(AbstractConnection.java:305)
  at org.eclipse.jetty.io.FillInterest.fillable(FillInterest.java:103)
  at org.eclipse.jetty.io.ChannelEndPoint$2.run(ChannelEndPoint.java:118)
  at org.eclipse.jetty.util.thread.QueuedThreadPool.runJob(QueuedThreadPool.java:765)
  at org.eclipse.jetty.util.thread.QueuedThreadPool$2.run(QueuedThreadPool.java:683)
  at java.base/java.lang.Thread.run(Thread.java:834)
Caused by: com.project.module.MyProjectFooBarException: The number of FooBars cannot be zero
  at com.project.module.MyProject.anotherMethod(MyProject.java:20)
  at com.project.module.MyProject.someMethod(MyProject.java:12)
  at com.framework.ApplicationStarter.lambda$start$0(ApplicationStarter.java:13)
  ... 16 more
Caused by: java.lang.ArithmetricException: The denominator must not be zero
  at org.apache.commons.lang3.math.Fraction.getFraction(Fraction.java:143)
  at com.project.module.MyProject.anotherMethod(MyProject.java:18)
  ... 18 more
```



Exception tracking

Report exceptions to exception tracking service
Help to identify the root cause



Exception tracking services



Audit logging

Log of user actions

Help customer support

Ensure compliance

Detect suspicious behavior



How to implement the audit logging ?

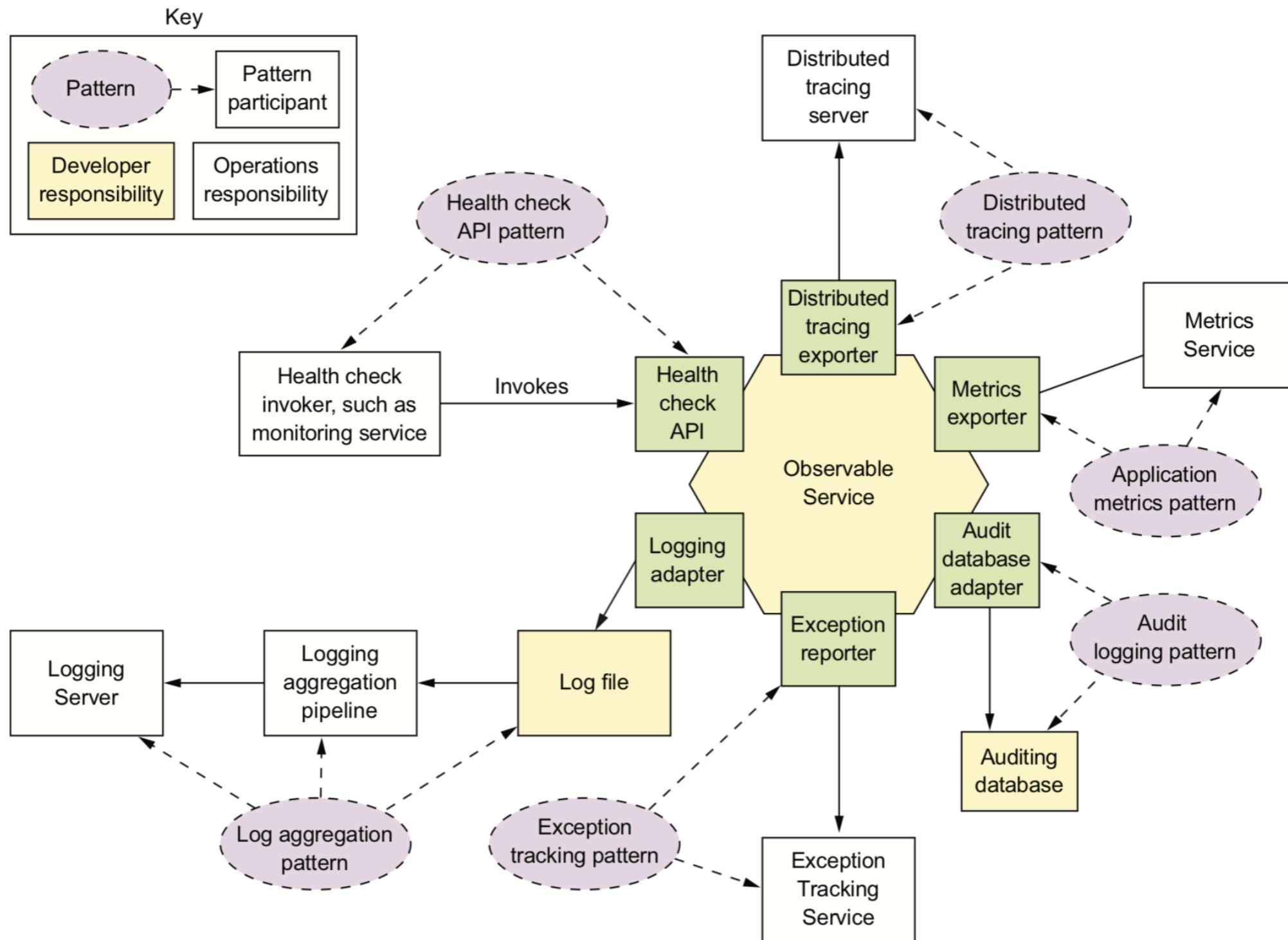
Add logging code in business logic

Use AOP (Aspect-Oriented Programming)

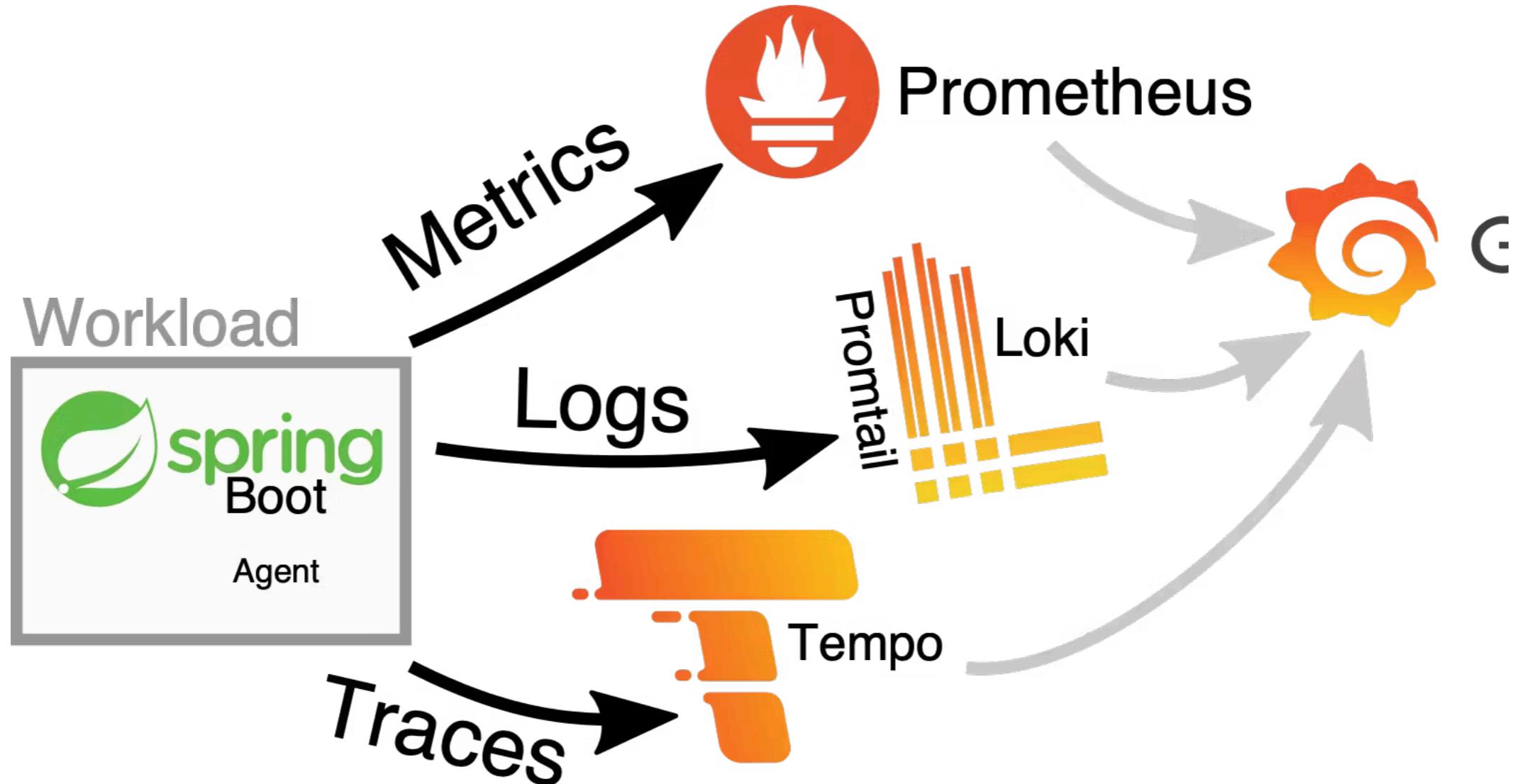
Use event sourcing



Observable services



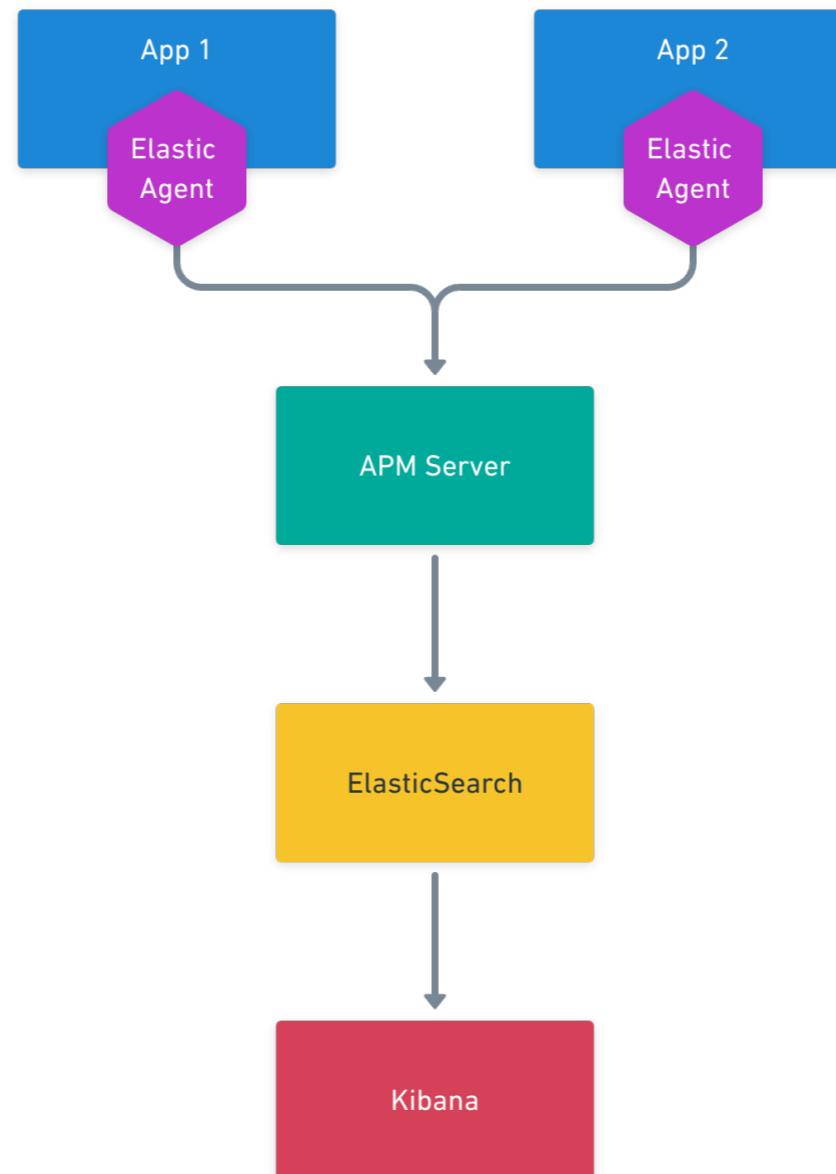
Grafana platform



<https://grafana.com/>



Elastic APM



<https://www.elastic.co/observability/application-performance-monitoring>



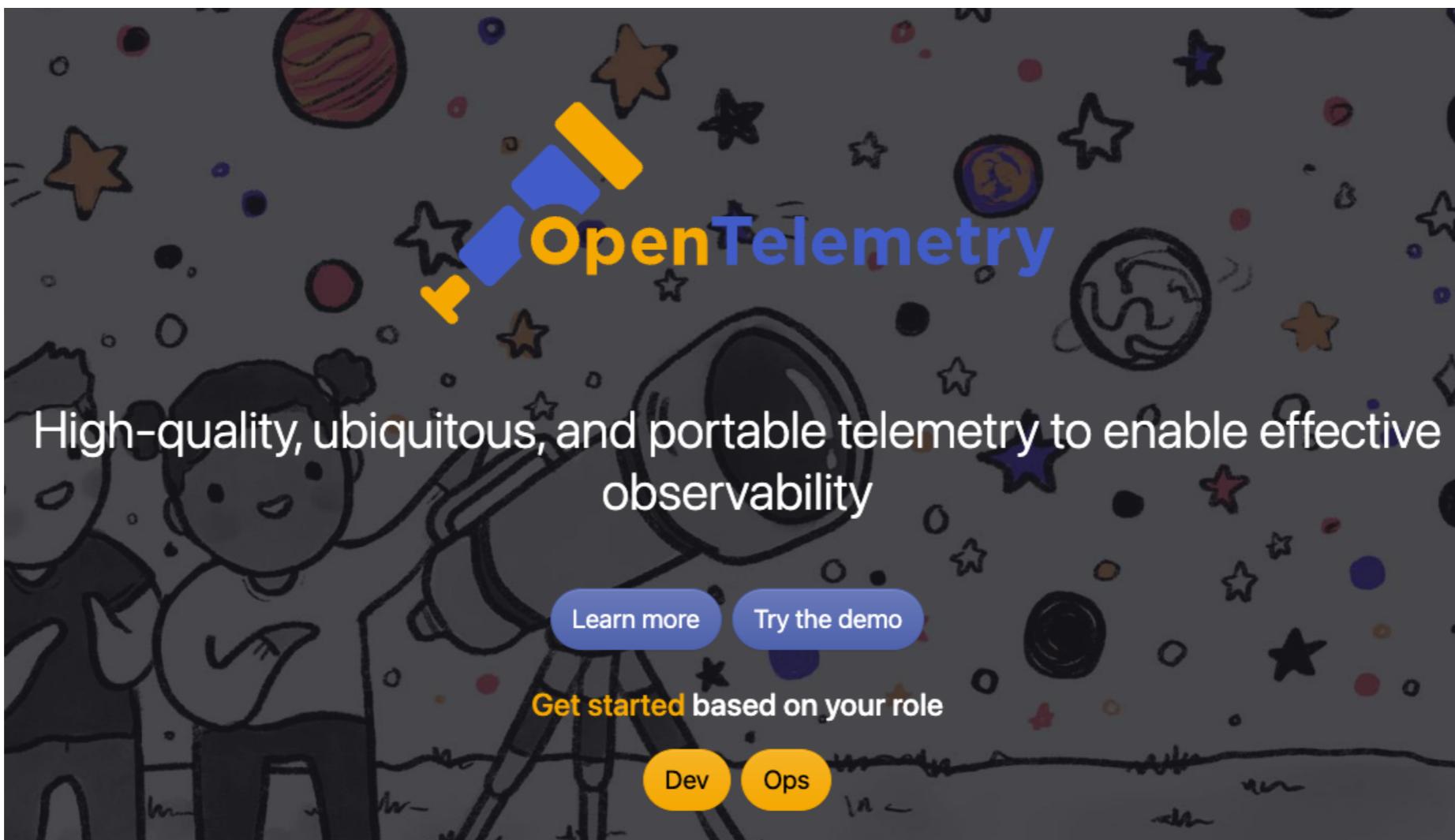
Observable service workshop

<https://github.com/up1/workshop-develop-microservices-2023>



Distributed Tracing

Working with OpenTelemetry



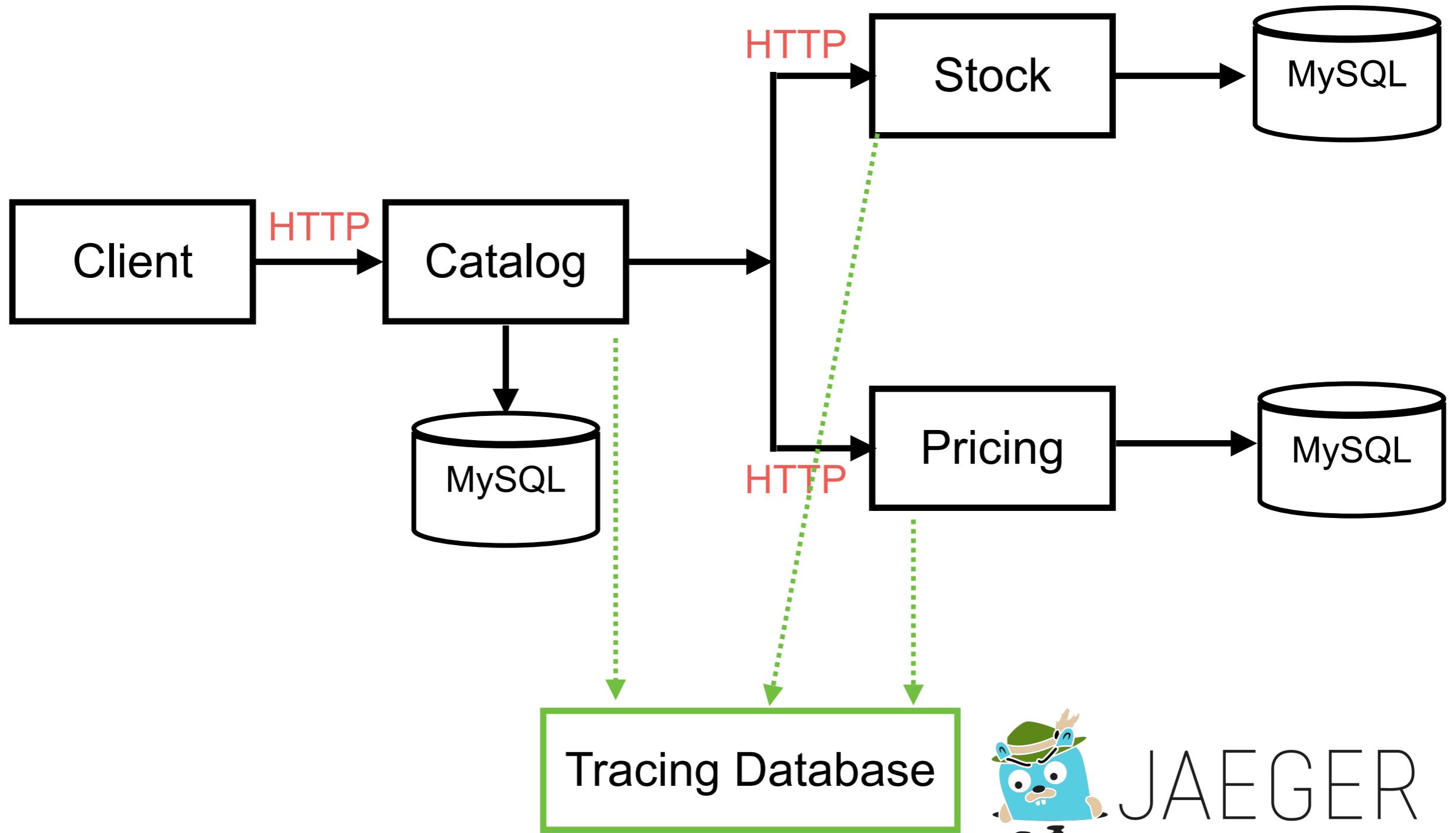
<https://opentelemetry.io/>



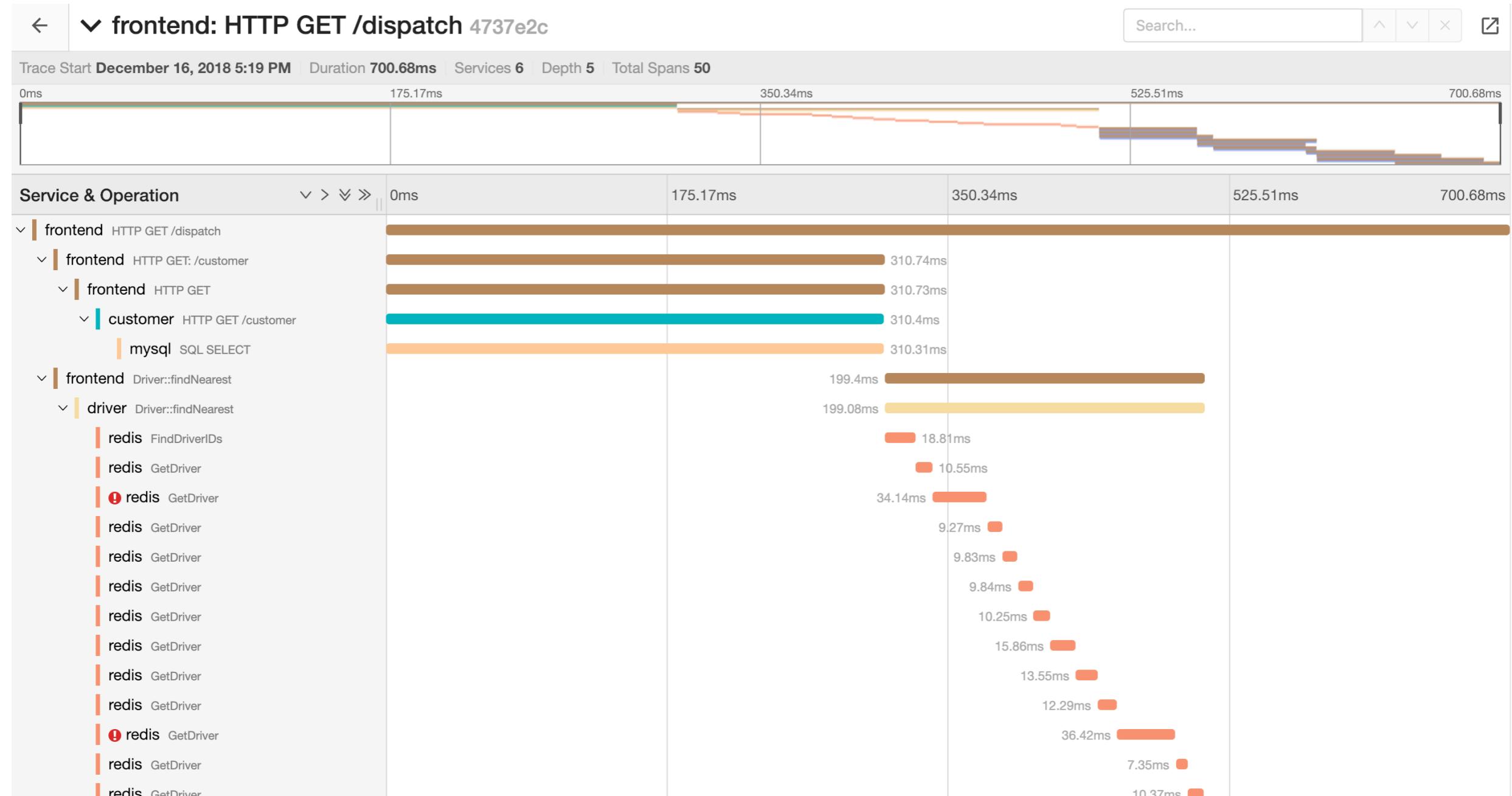
Microservices

© 2022 - 2023 Siam Chamnankit Company Limited. All rights reserved.

Distributed Tracing



Distributed Tracing with Jaeger



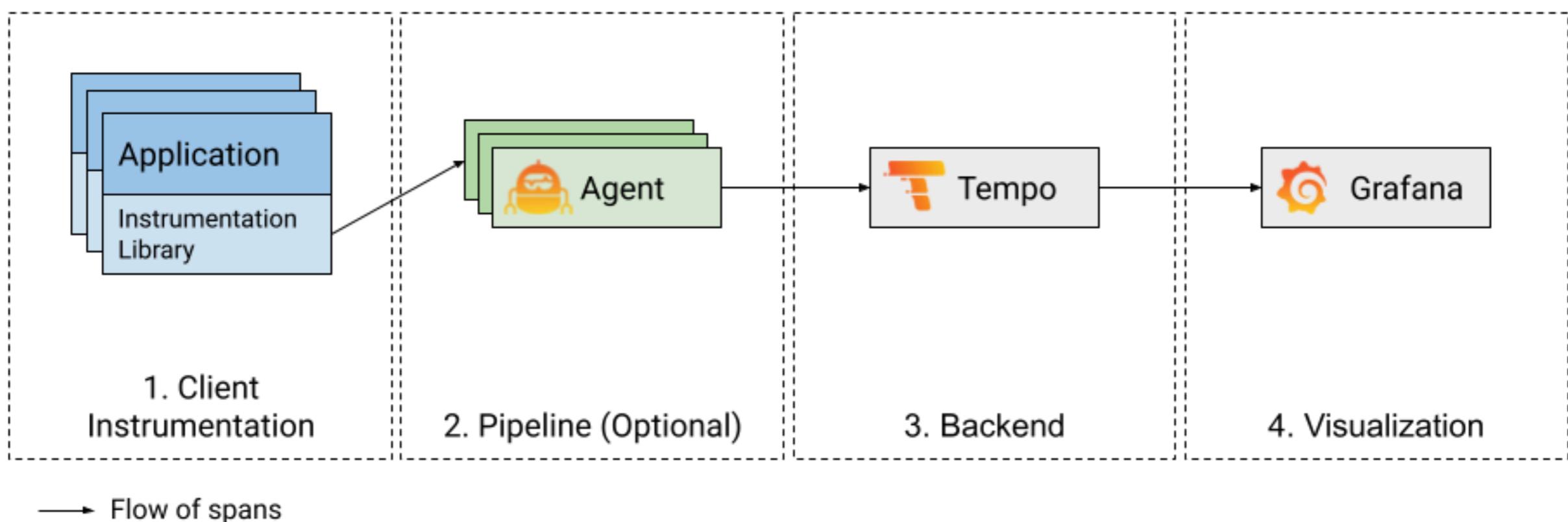
<https://www.jaegertracing.io>



Microservices

© 2022 - 2023 Siam Chamnankit Company Limited. All rights reserved.

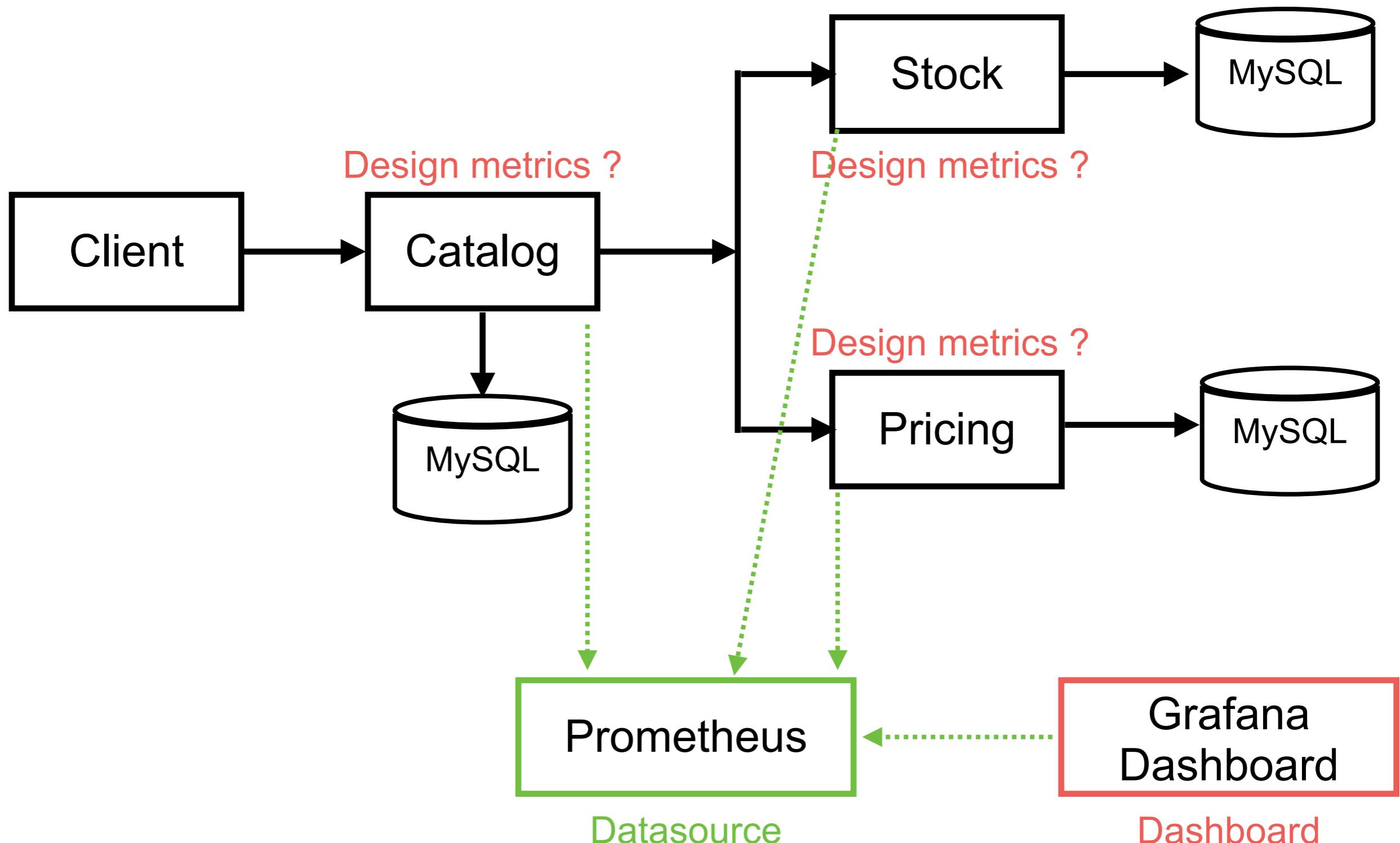
Distributed Tracing with Grafana



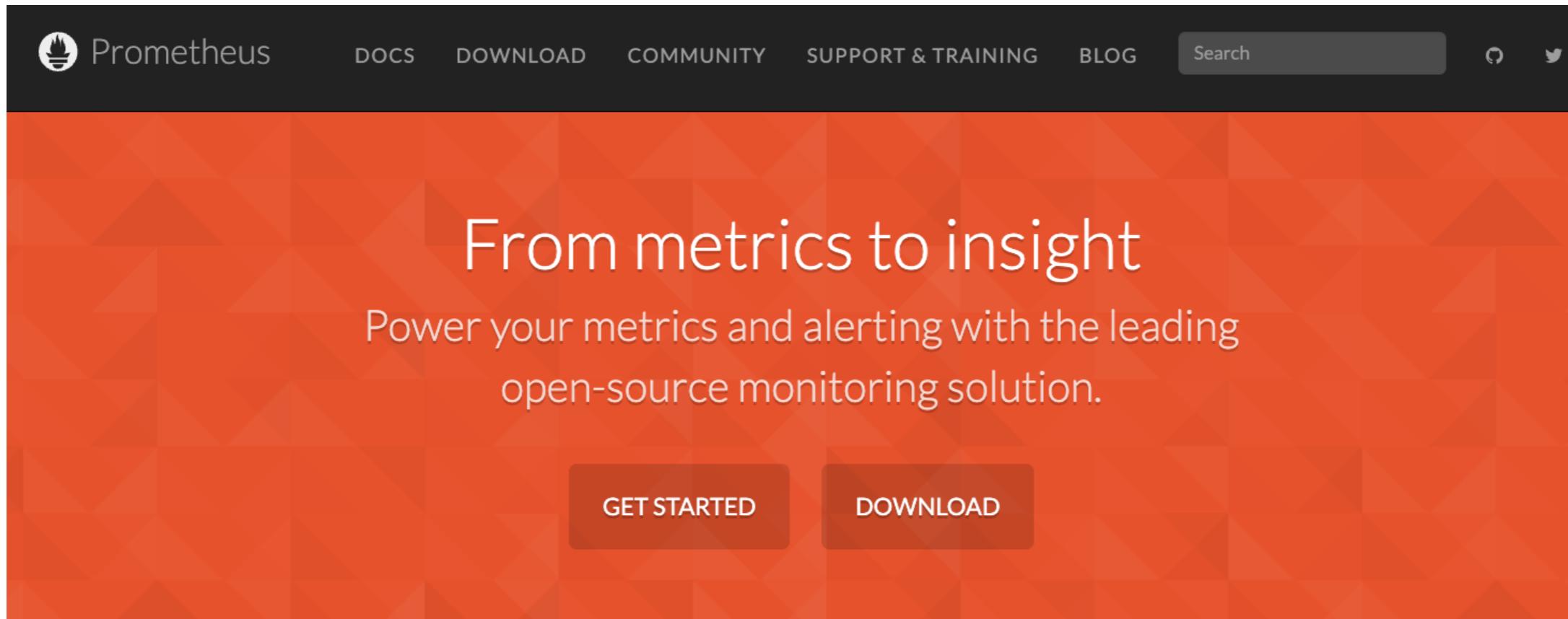
<https://www.jaegertracing.io>



Application metric



Prometheus



The banner features a dark header with the Prometheus logo and navigation links: DOCS, DOWNLOAD, COMMUNITY, SUPPORT & TRAINING, BLOG, and a Search bar. Below the header is a large orange background area with white text. The main headline reads "From metrics to insight" followed by the subtext "Power your metrics and alerting with the leading open-source monitoring solution." At the bottom of the orange area are two buttons: "GET STARTED" and "DOWNLOAD".

From metrics to insight

Power your metrics and alerting with the leading open-source monitoring solution.

GET STARTED DOWNLOAD

Dimensional data

Prometheus implements a highly dimensional data model. Time series are identified by a metric name and a set of key-value pairs.

Powerful queries

PromQL allows slicing and dicing of collected time series data in order to generate ad-hoc graphs, tables, and alerts.

Great visualization

Prometheus has multiple modes for visualizing data: a built-in expression browser, Grafana integration, and a console template language.

Efficient storage

Prometheus stores time series in memory and on local disk in an efficient custom format. Scaling is achieved by functional sharding and

<https://prometheus.io/>



Microservices

© 2022 - 2023 Siam Chamnankit Company Limited. All rights reserved.

Grafana

Grafana Labs Products Open source Solutions Learn Company Downloads Contact us Sign in

Compose and scale observability with one or all pieces of the stack

The diagram illustrates Grafana's observability stack. On the left, a sidebar shows a list of applications and infrastructure integrations, each with a '100+' badge. The main area shows the Grafana interface with various features: Plugins, Dashboards, Alerts, Usage insights, Reports, and Governance. Below the interface, three colored boxes represent the data types: Metrics (yellow), Logs (orange), and Traces (red). Arrows show the flow from the central interface down to the data types, and from the data types up to the interface. A large bracket on the right groups the central interface and data types, with the text 'Your observability wherever you need it' above it, and 'Cloud' and 'Self-managed' options below.

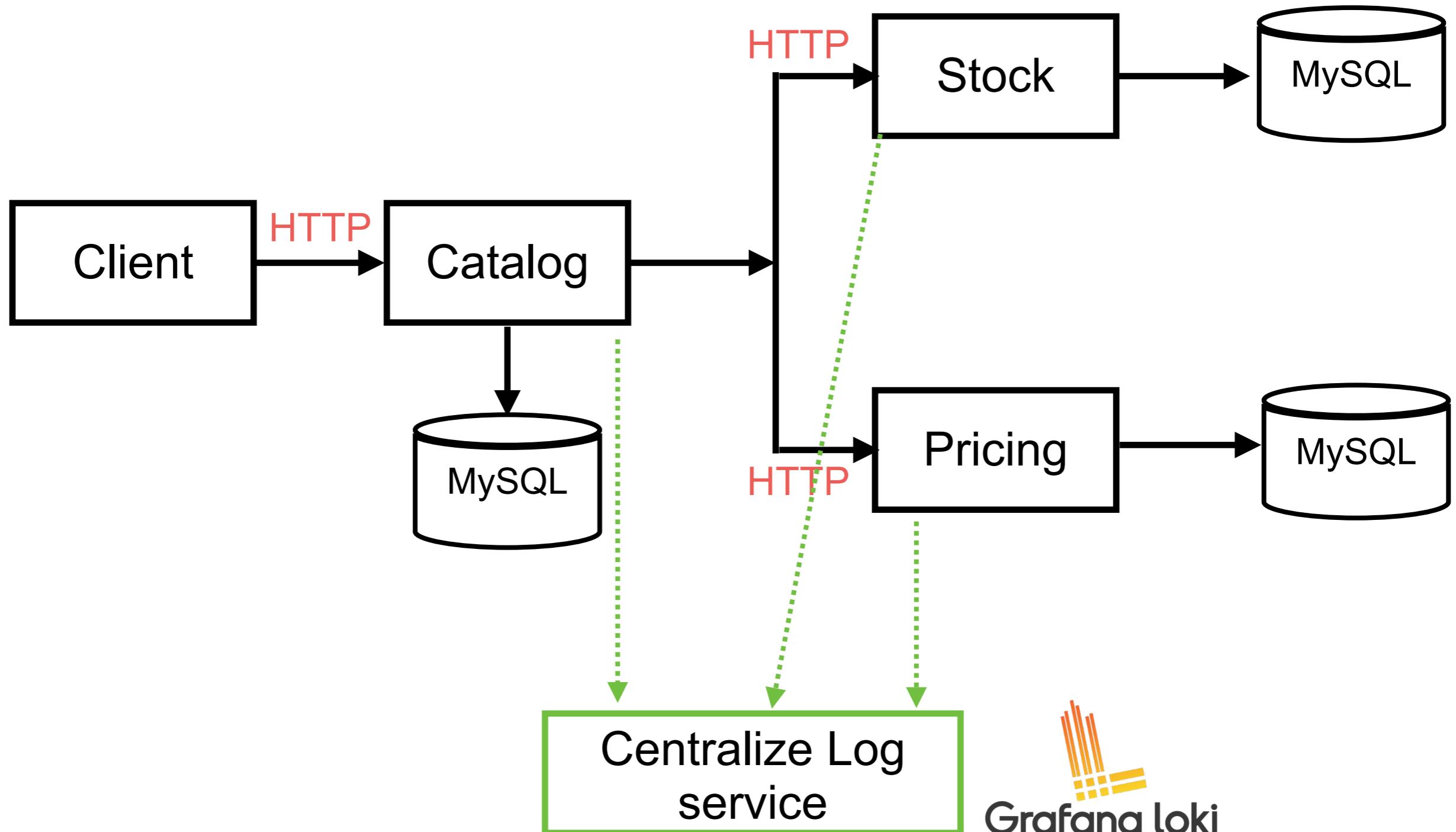
<https://grafana.com/>



Microservices

© 2022 - 2023 Siam Chamnankit Company Limited. All rights reserved.

Log aggregation



Secure services



Develop secure services

Authentication

Authorization

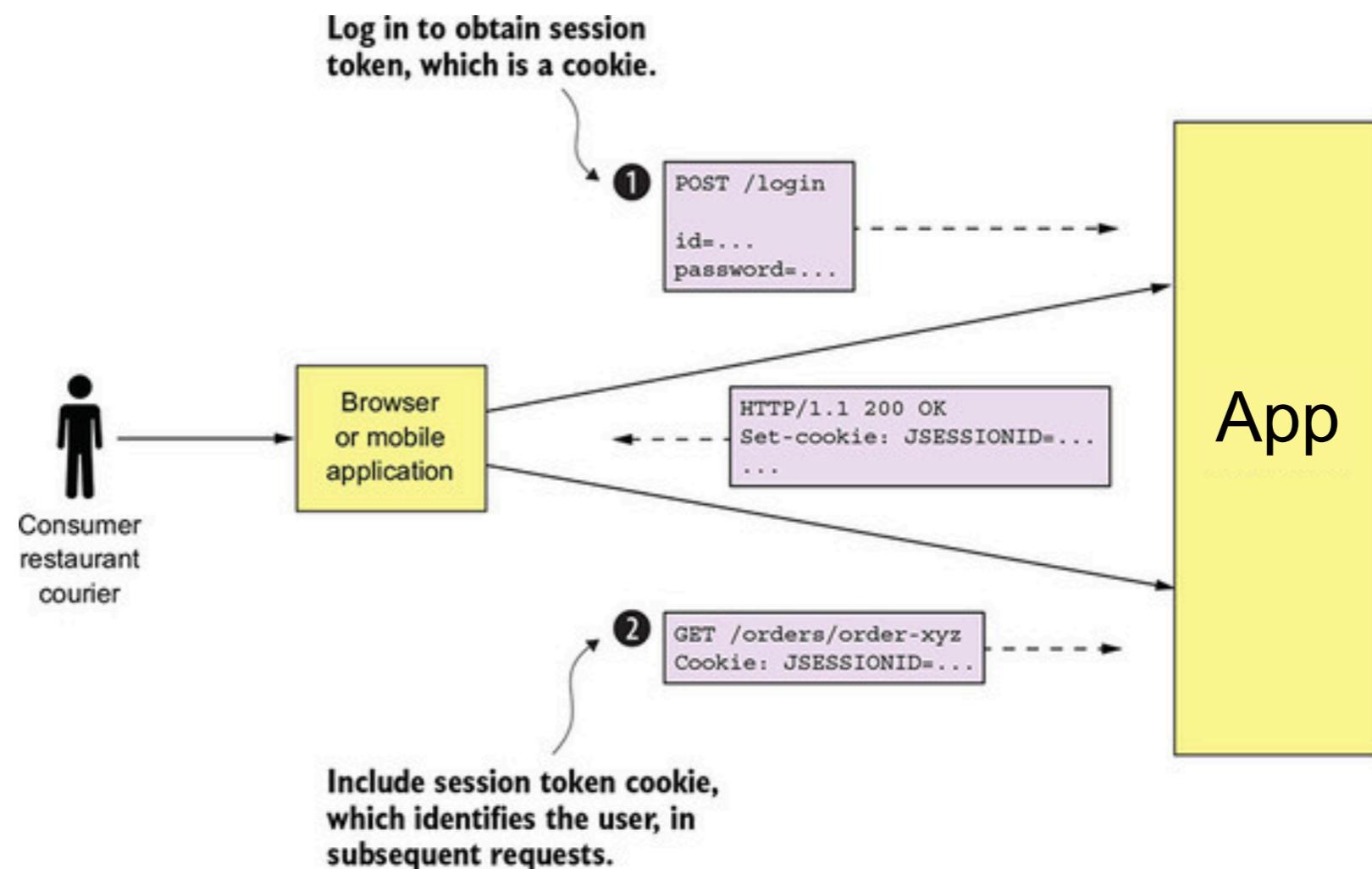
Auditing

Secure interprocess communication



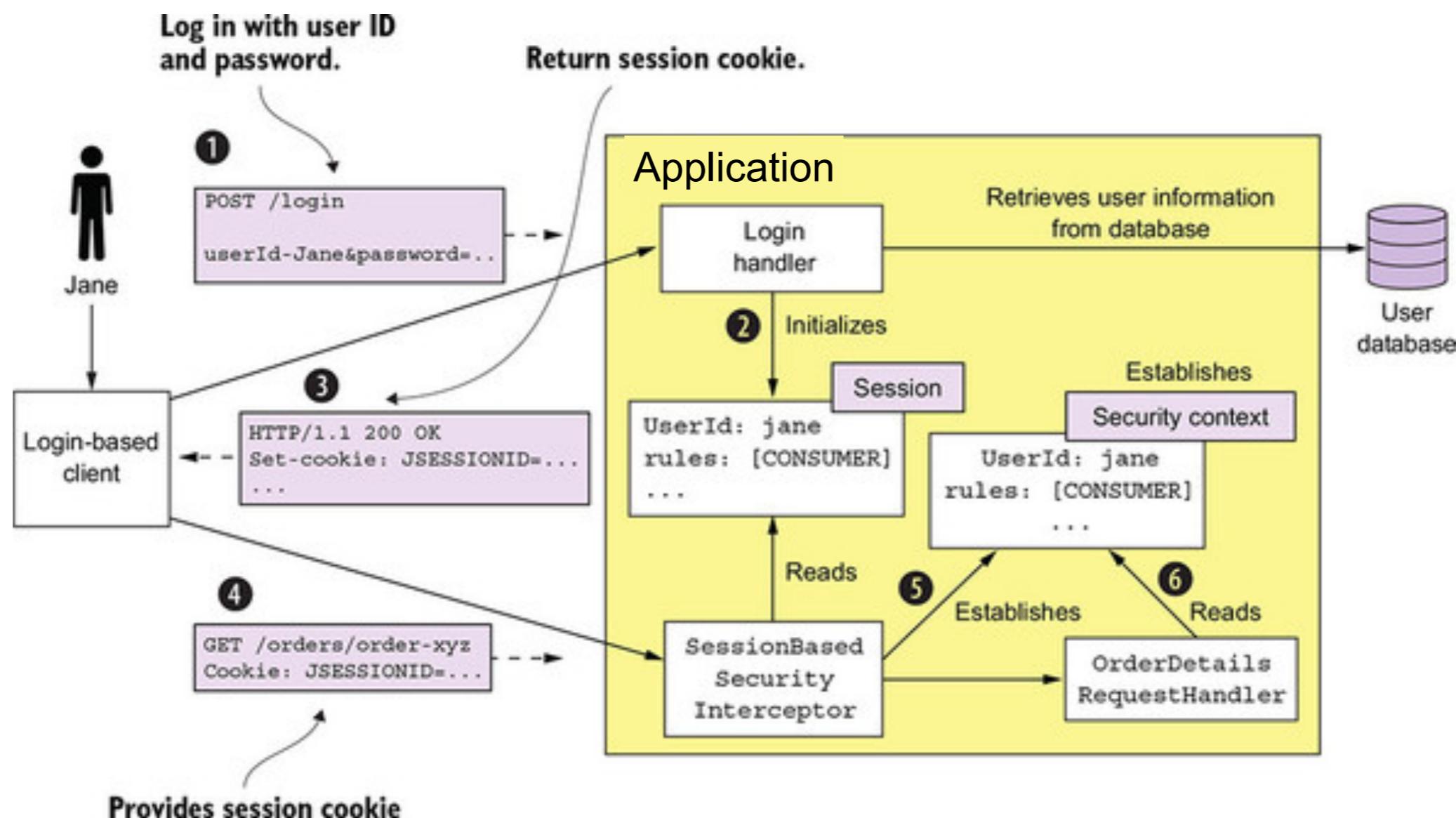
Security in traditional application

Keep security information in browser's cookie



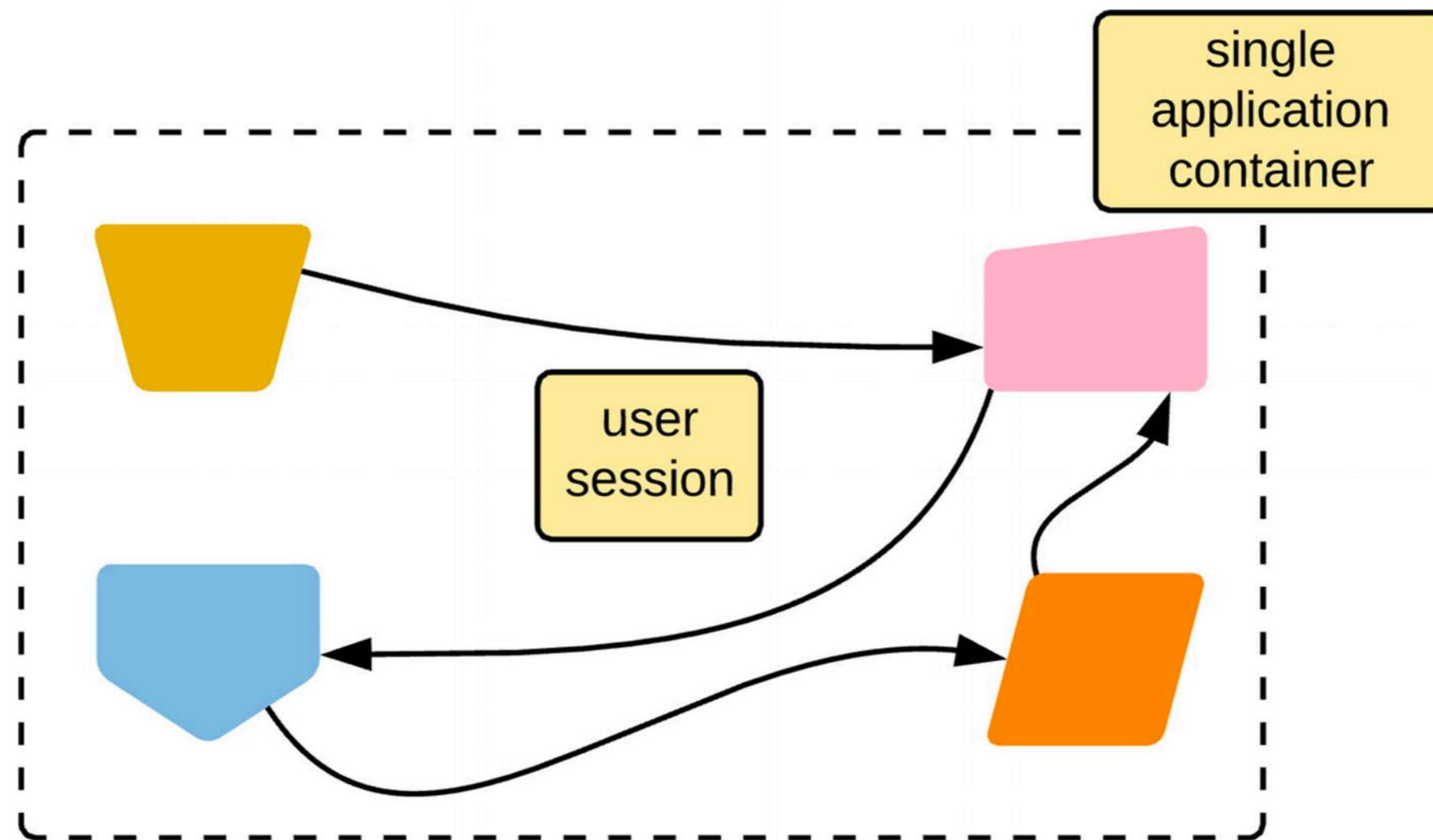
Security in traditional application

Implement security process in application



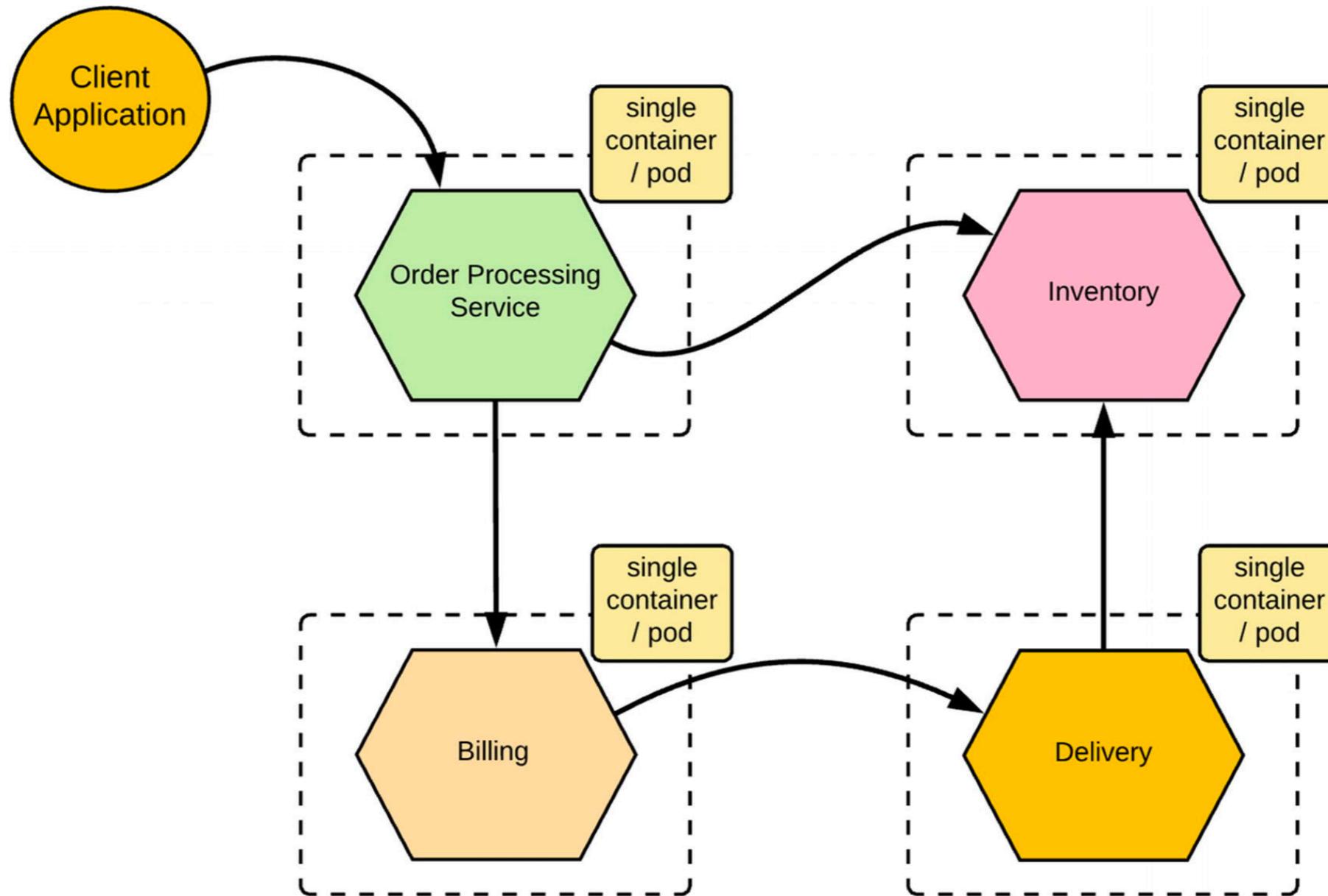
Security in traditional application

Sharing a user session in application



Security in multiple services ?

Interaction between multiple services



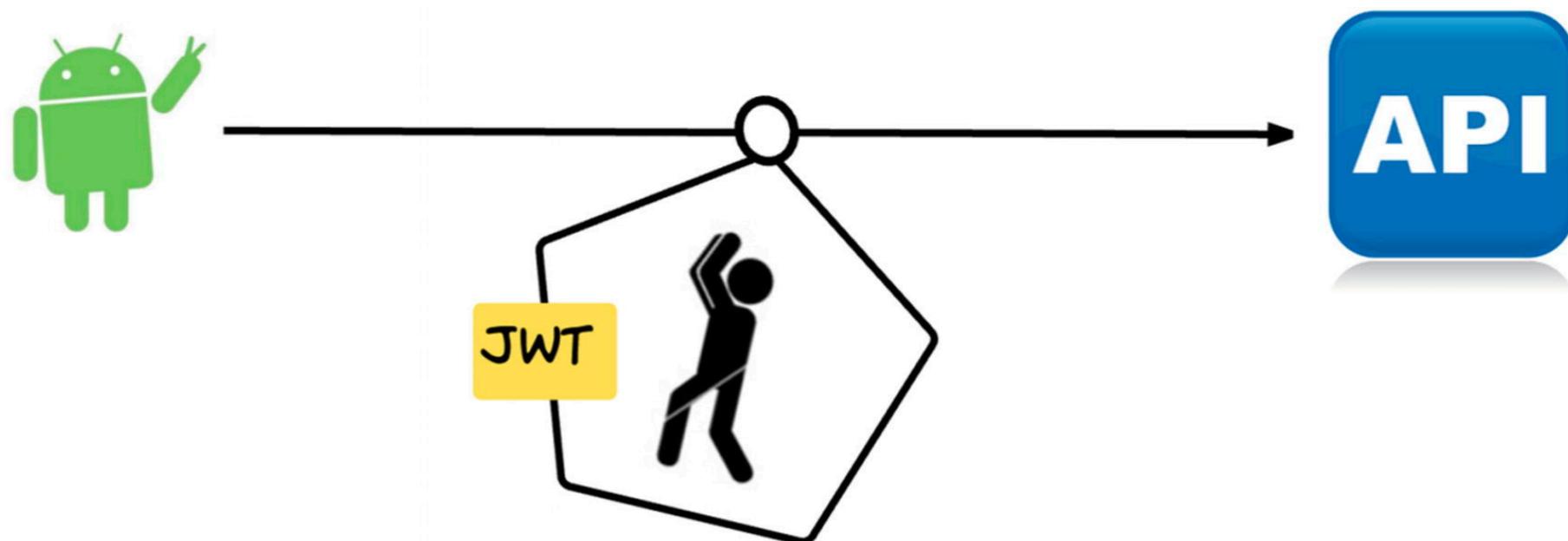
Secure service-to-service communication

JSON Web Token (JWT) + OAuth
Transport Layer Security (TLS) mutual authentication

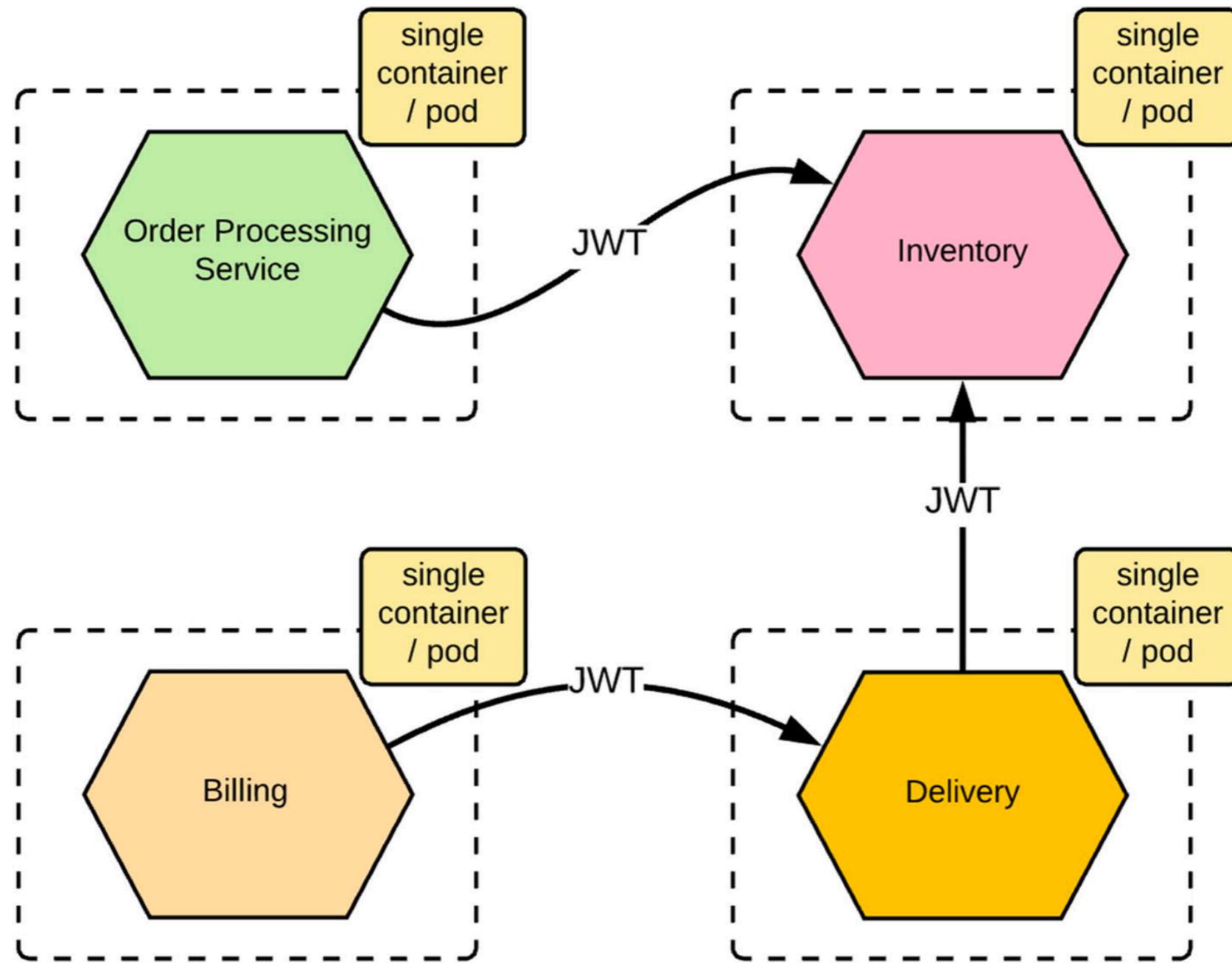


JSON Web Token (JWT)

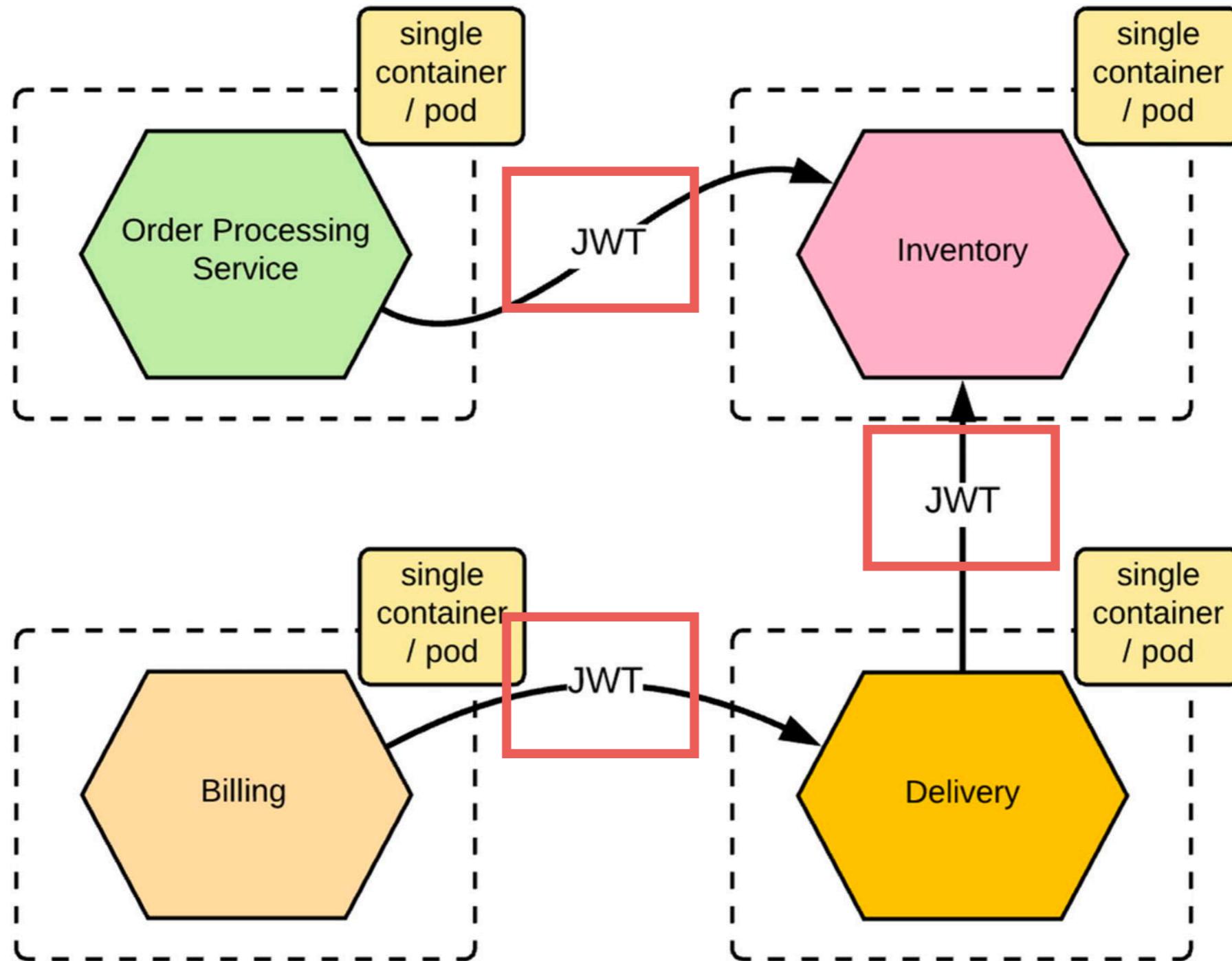
Define a container to transport data
between interested parties



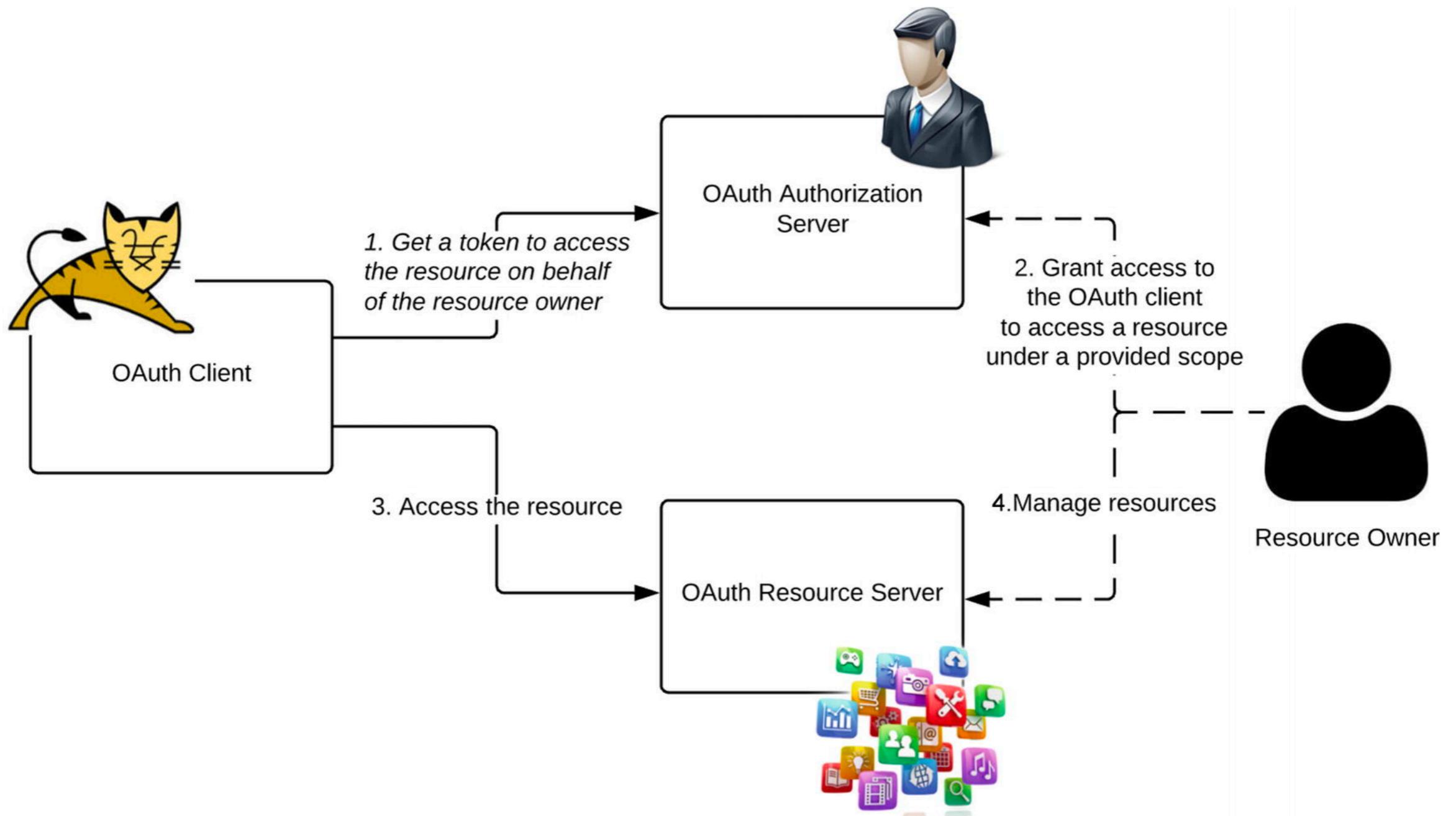
Passing user context as JWT



Problem ?

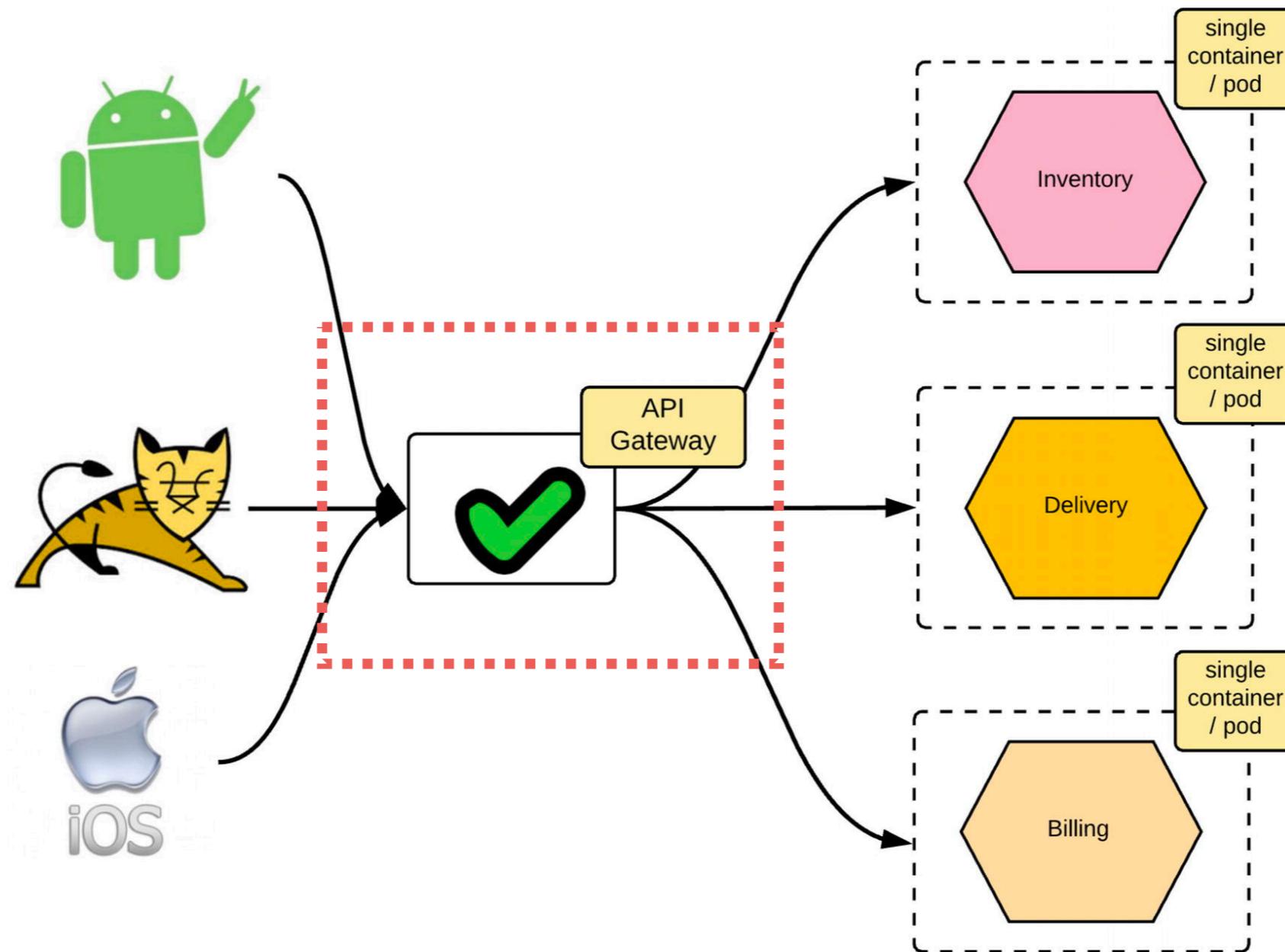


OAuth 2.0



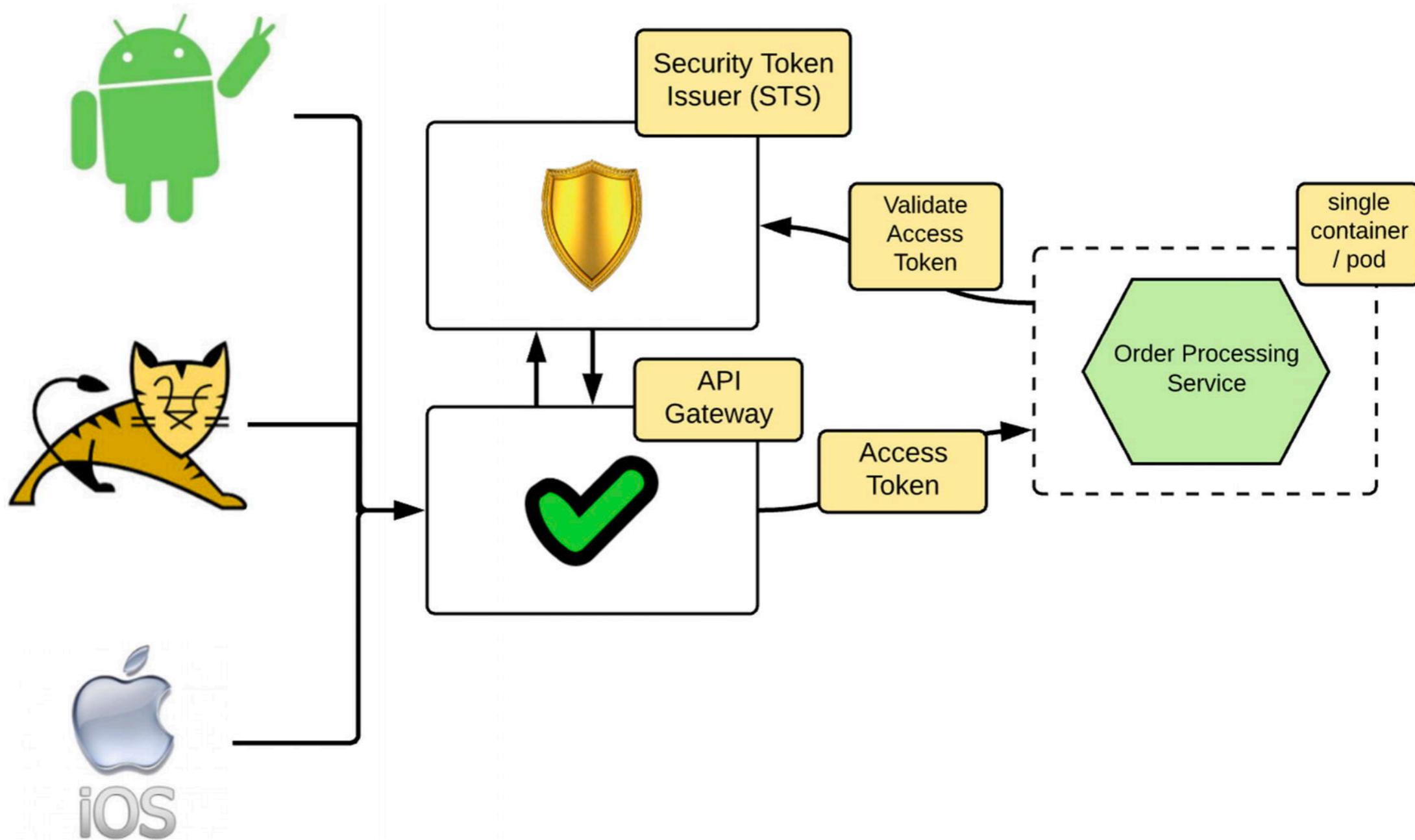
Centralize pattern

Using API gateway pattern

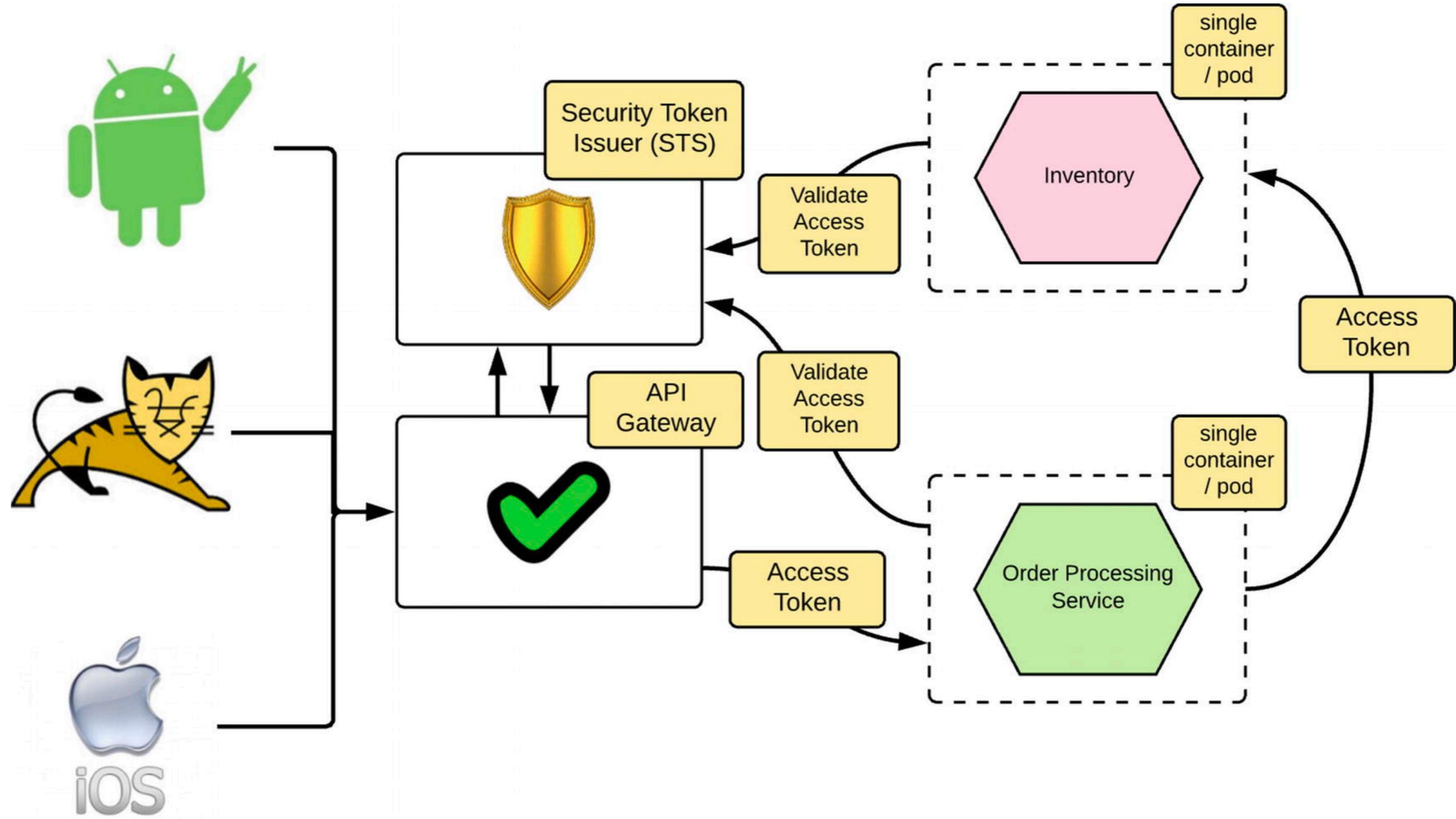


Centralize pattern

Using API gateway pattern

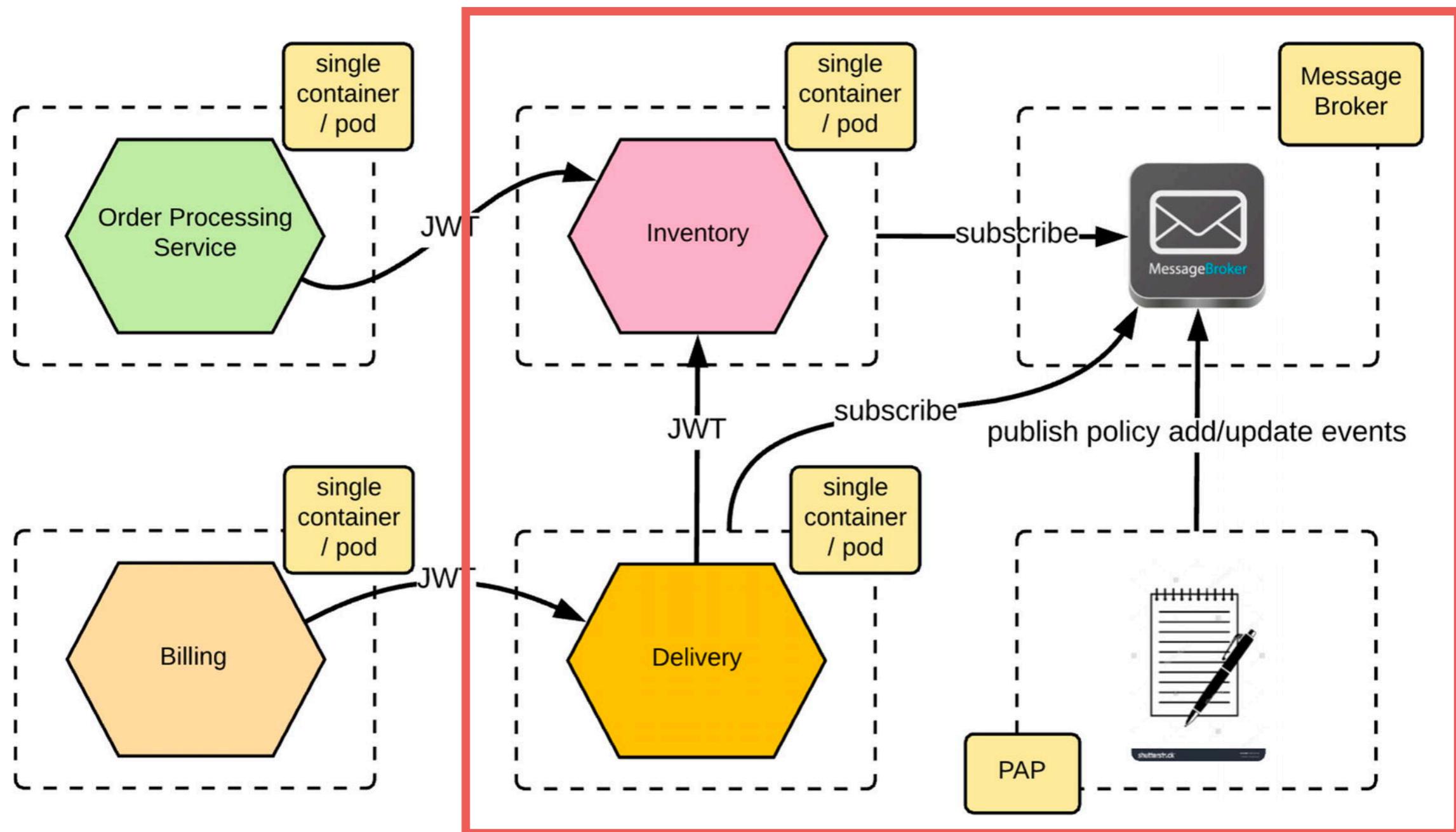


API gateway with OAuth 2.0



Access control of services

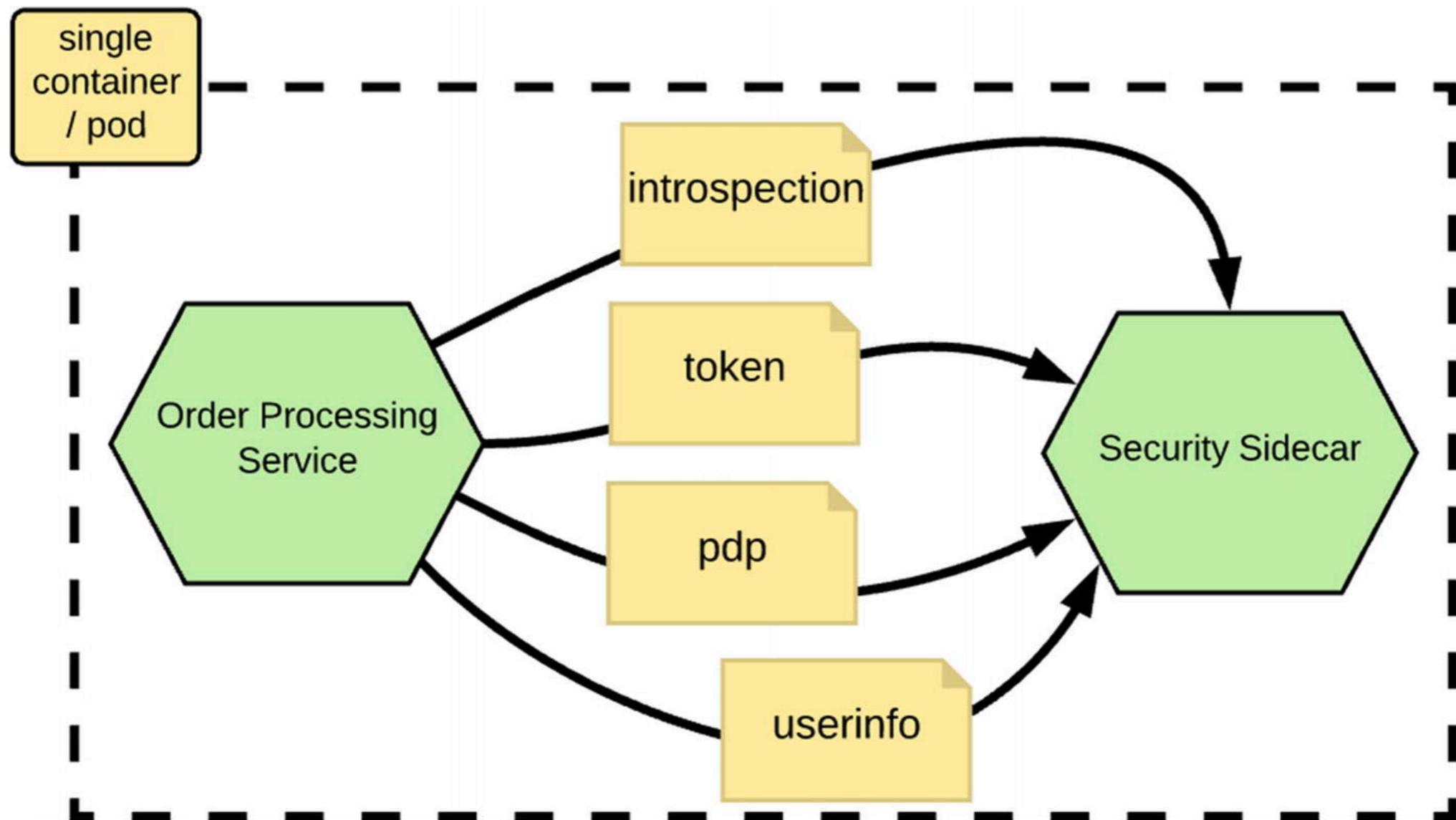
Policy Administration Point



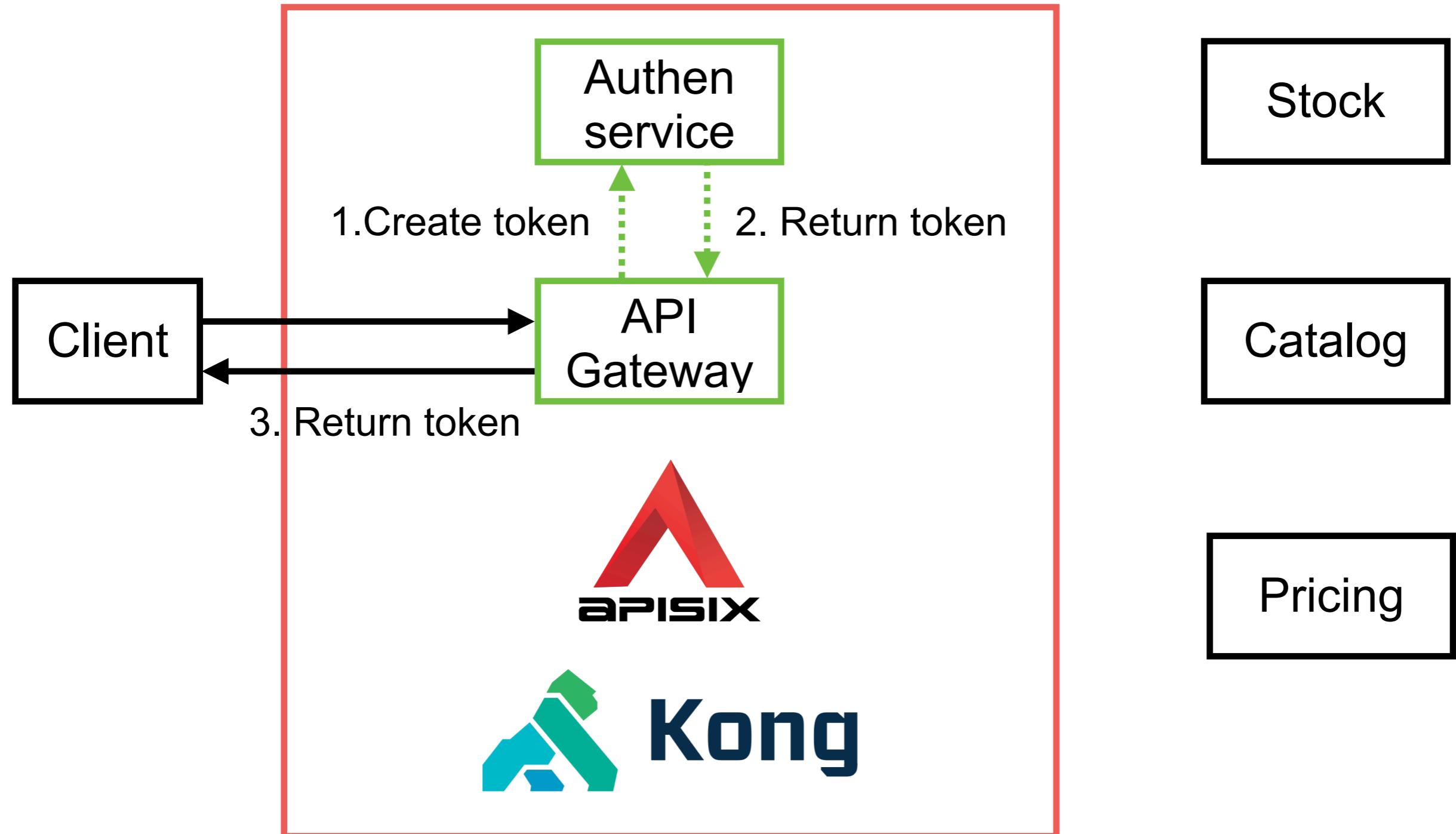
Working sidecar



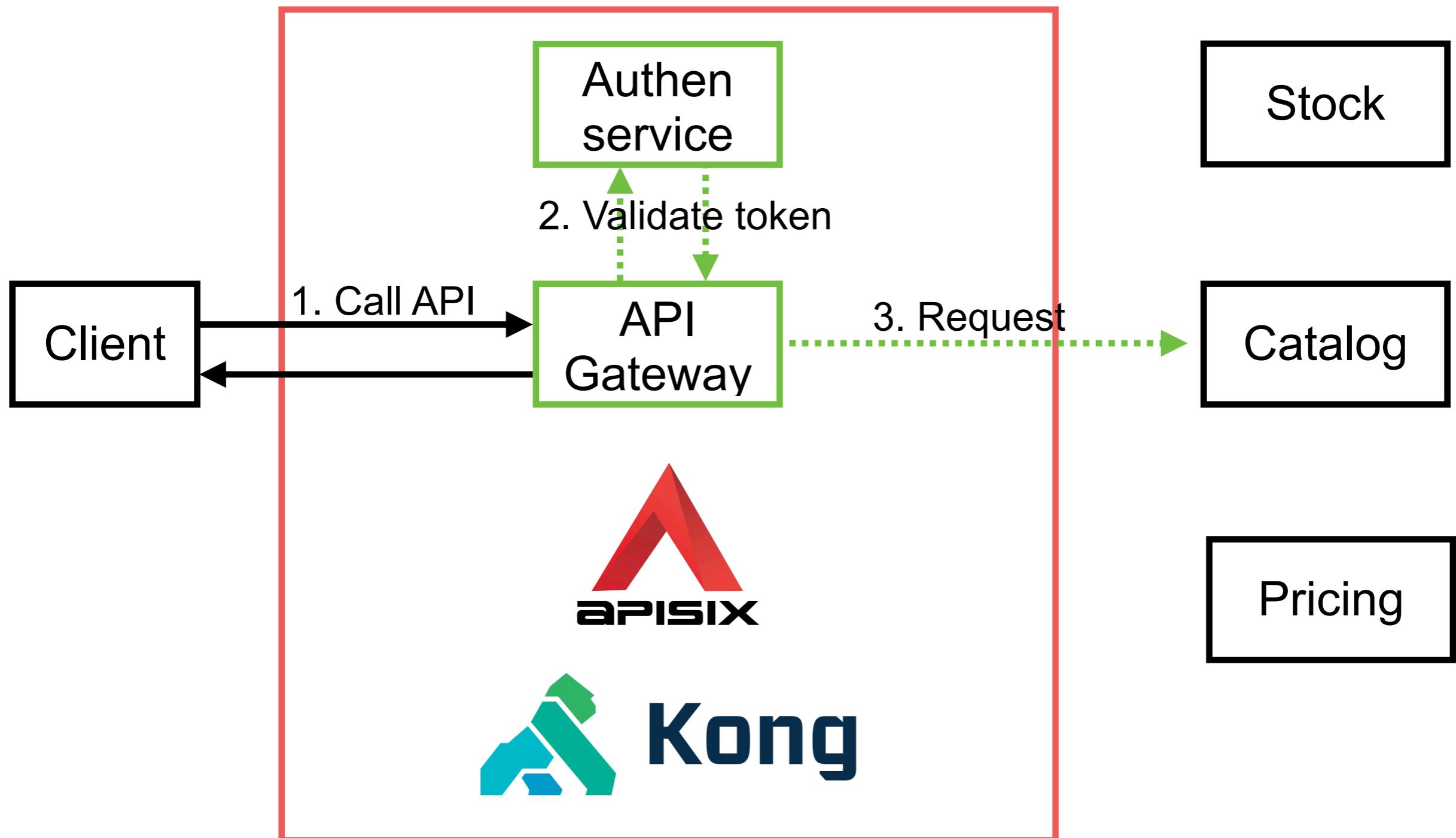
Security sidecar

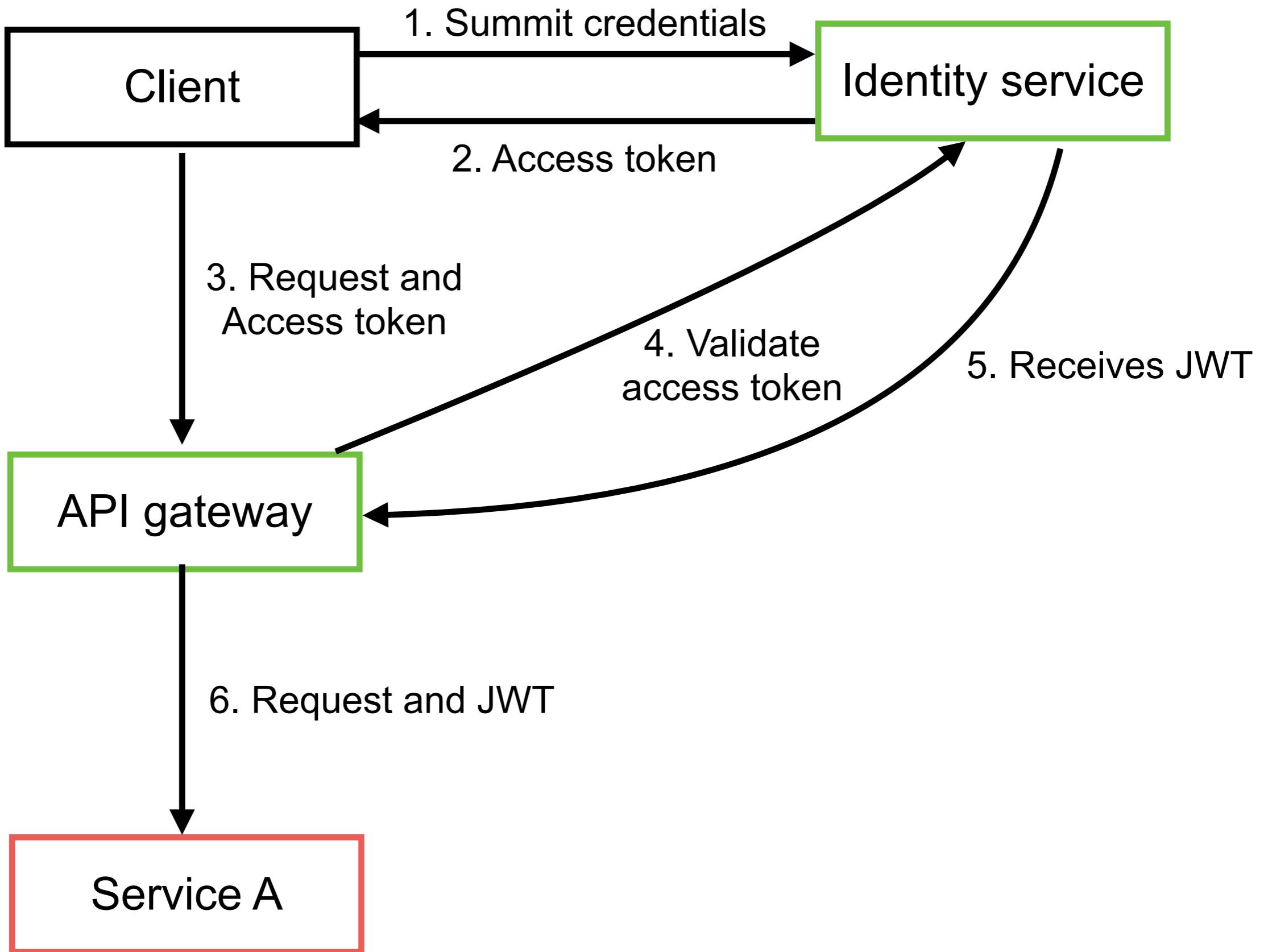


Workshop with Secure service



Workshop with Secure service

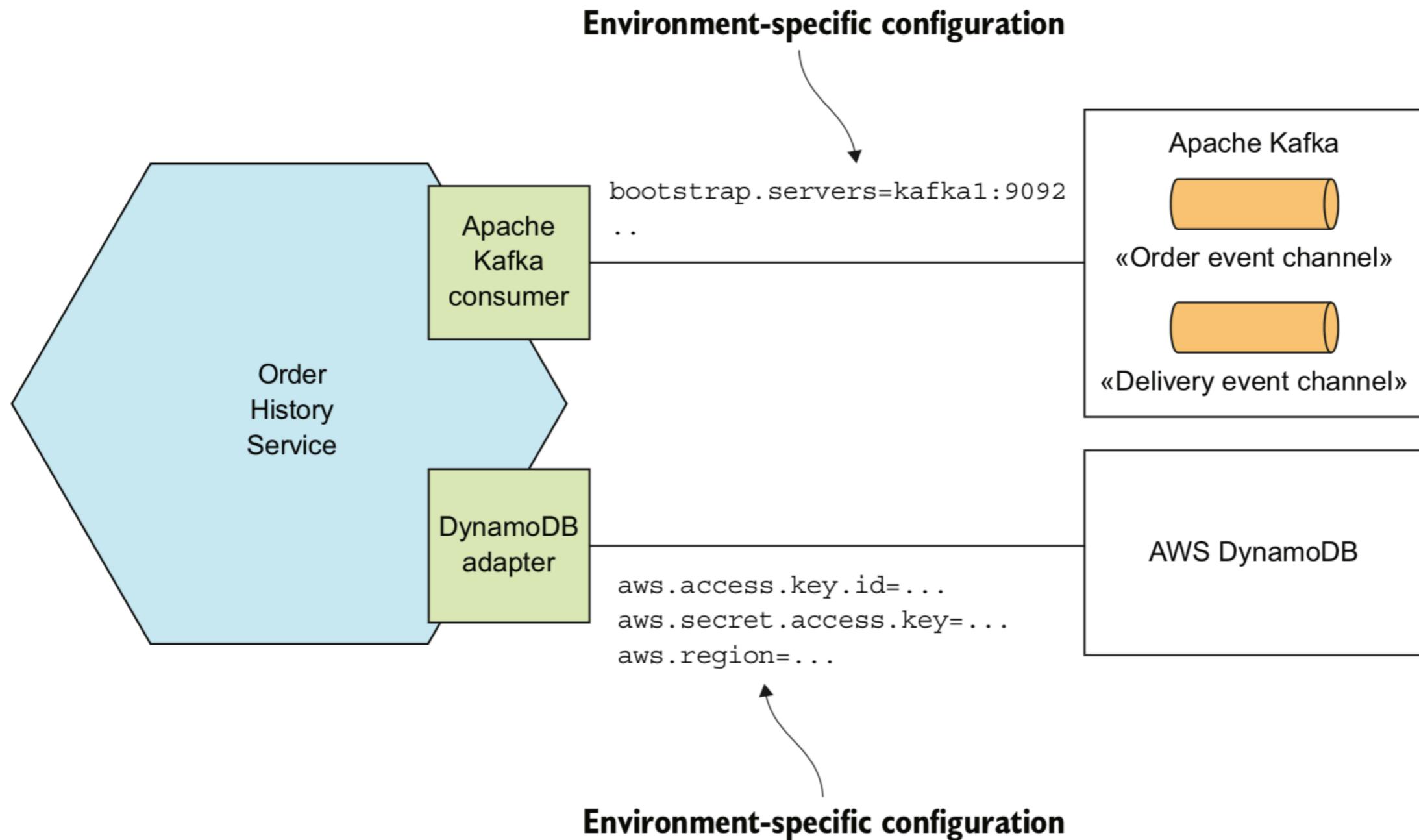




Configurable services



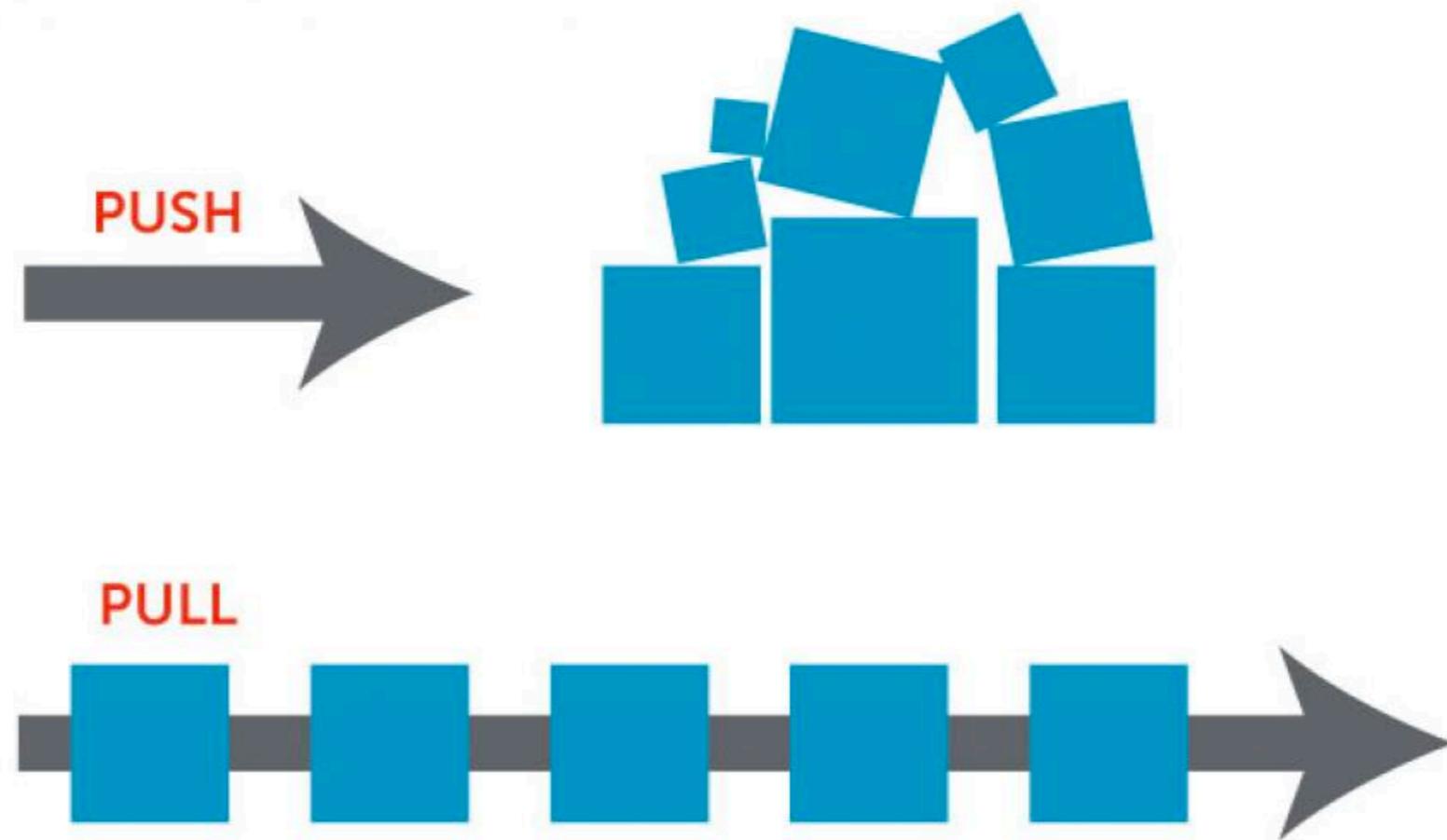
Design configurable services



External configuration models

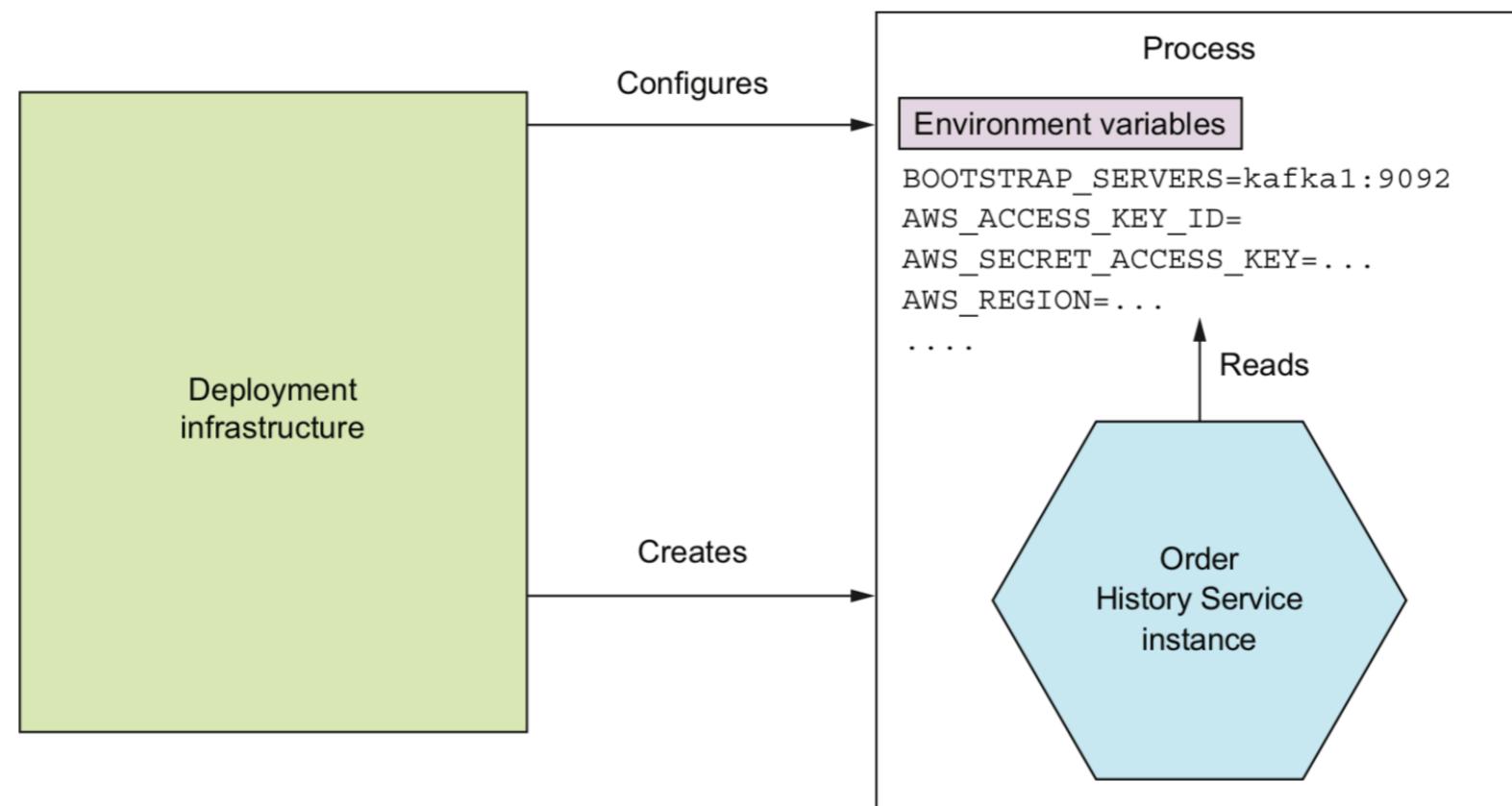
Push model

Pull model



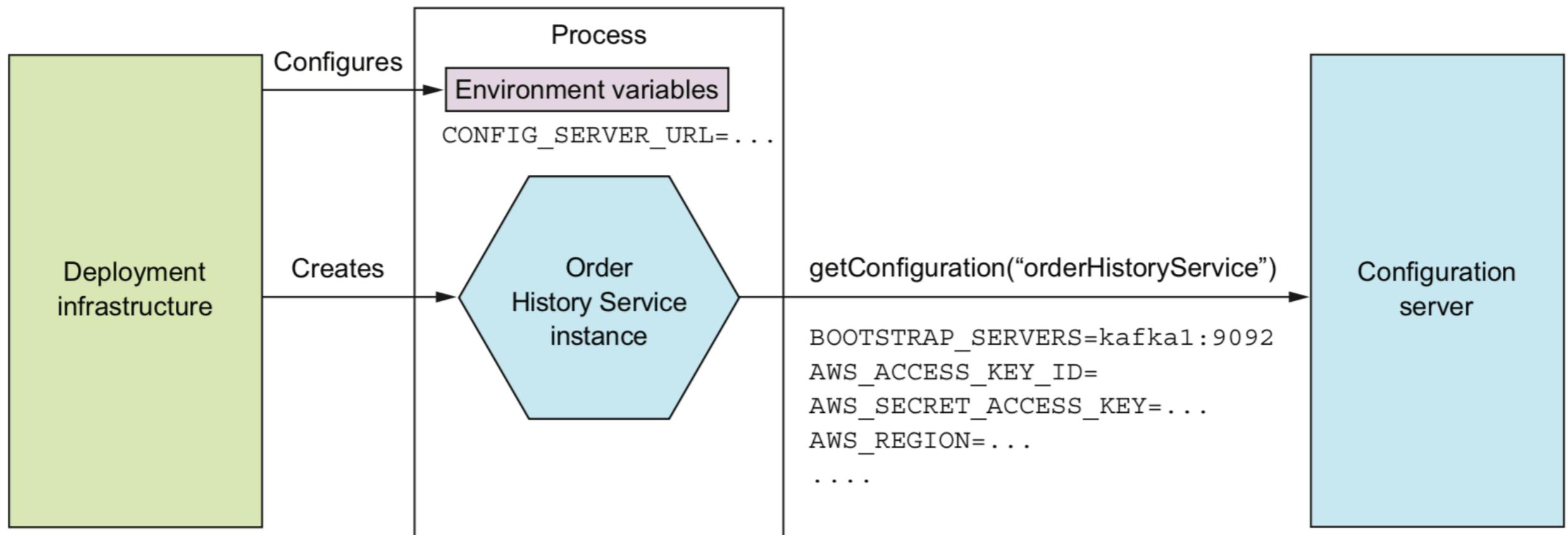
Push model

Pass the configuration to service
OS environment variables
Configuration files



Pull model

Service read configuration from configuration server



Benefits of configuration server

Centralized configuration

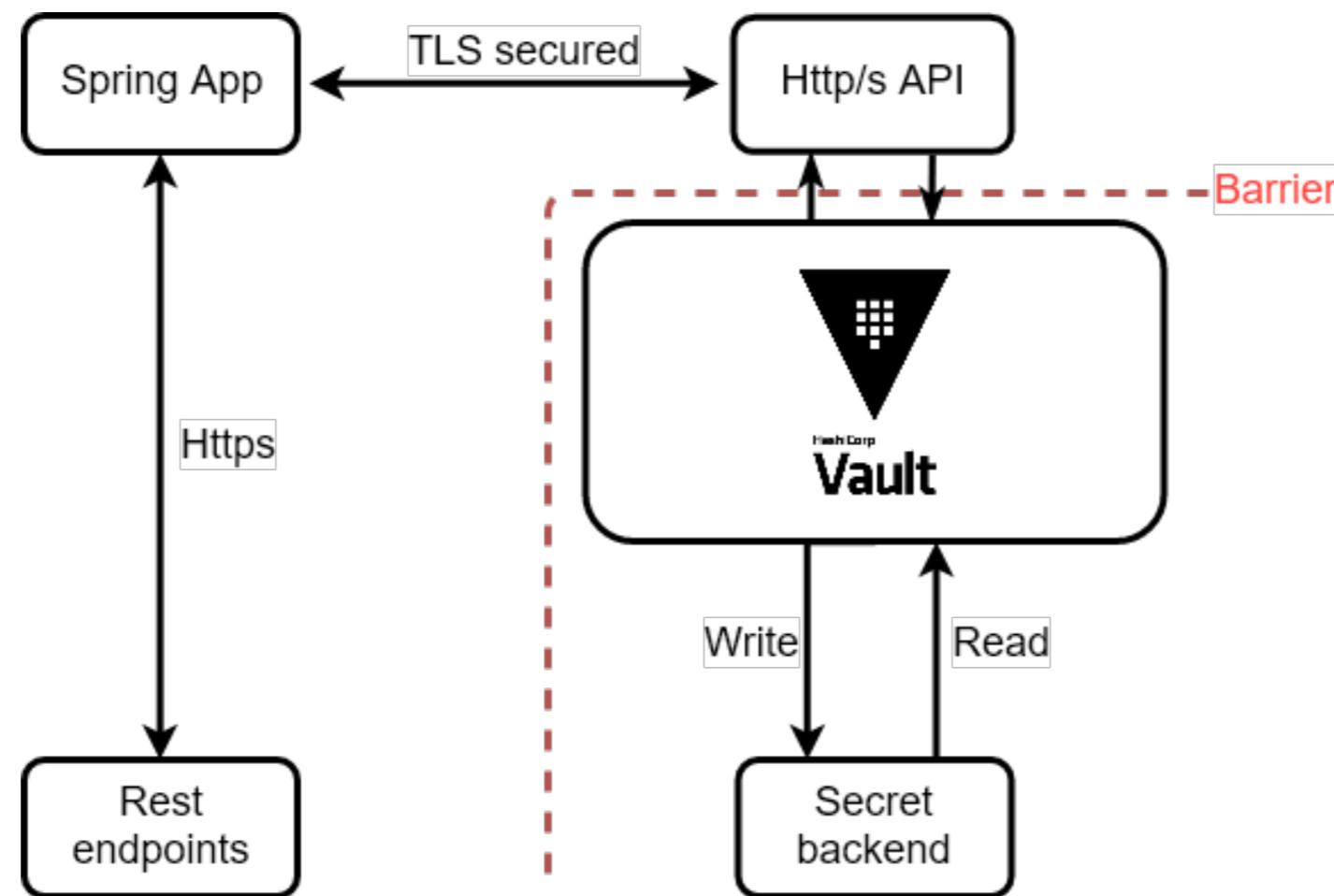
Transparent decryption of sensitive data

Dynamic reconfiguration



Workshop with Configuration

Plain text
Secure data



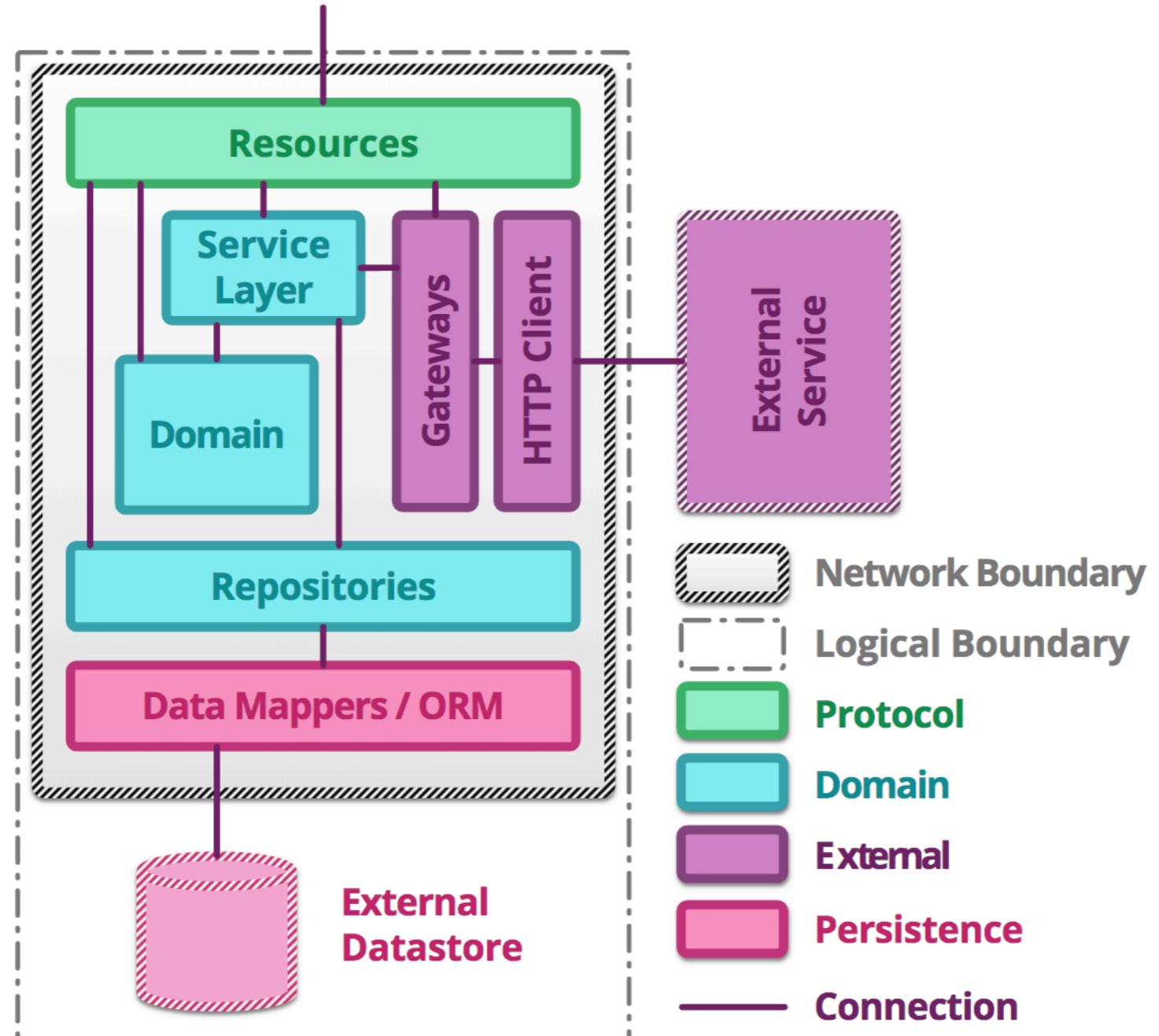
<https://www.vaultproject.io/>



Microservice Testing ?



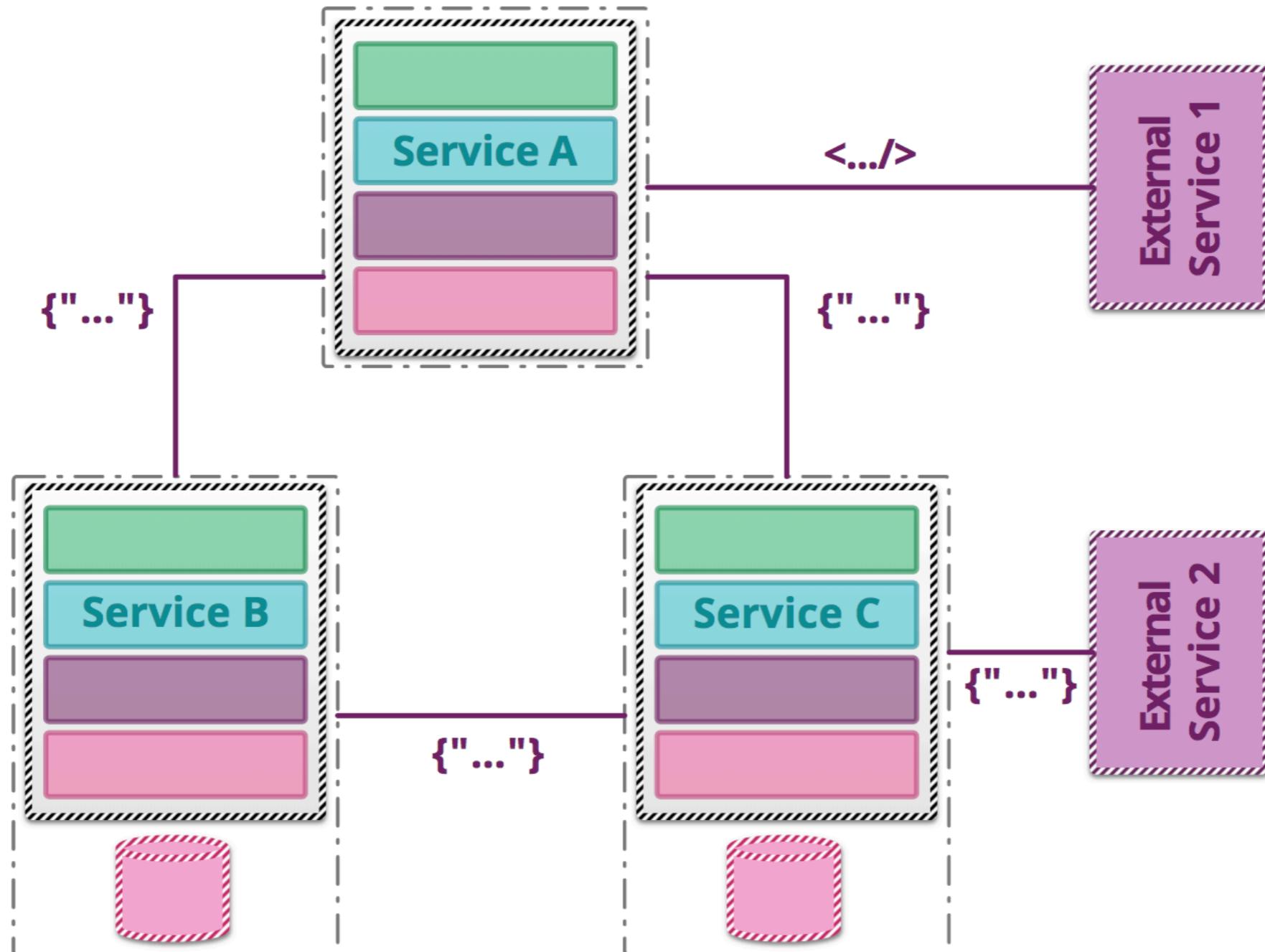
Service structure



<https://martinfowler.com/articles/microservice-testing>

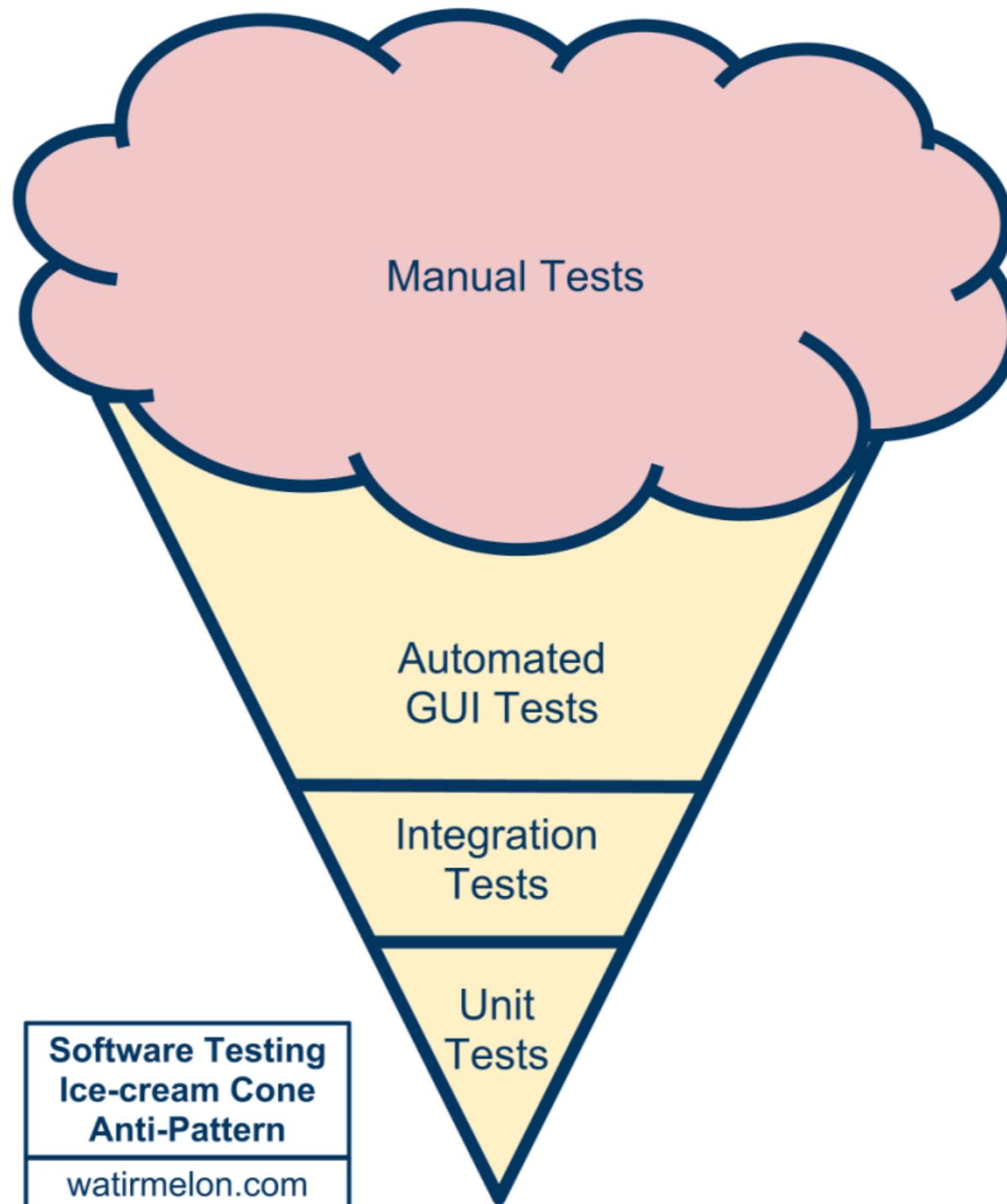


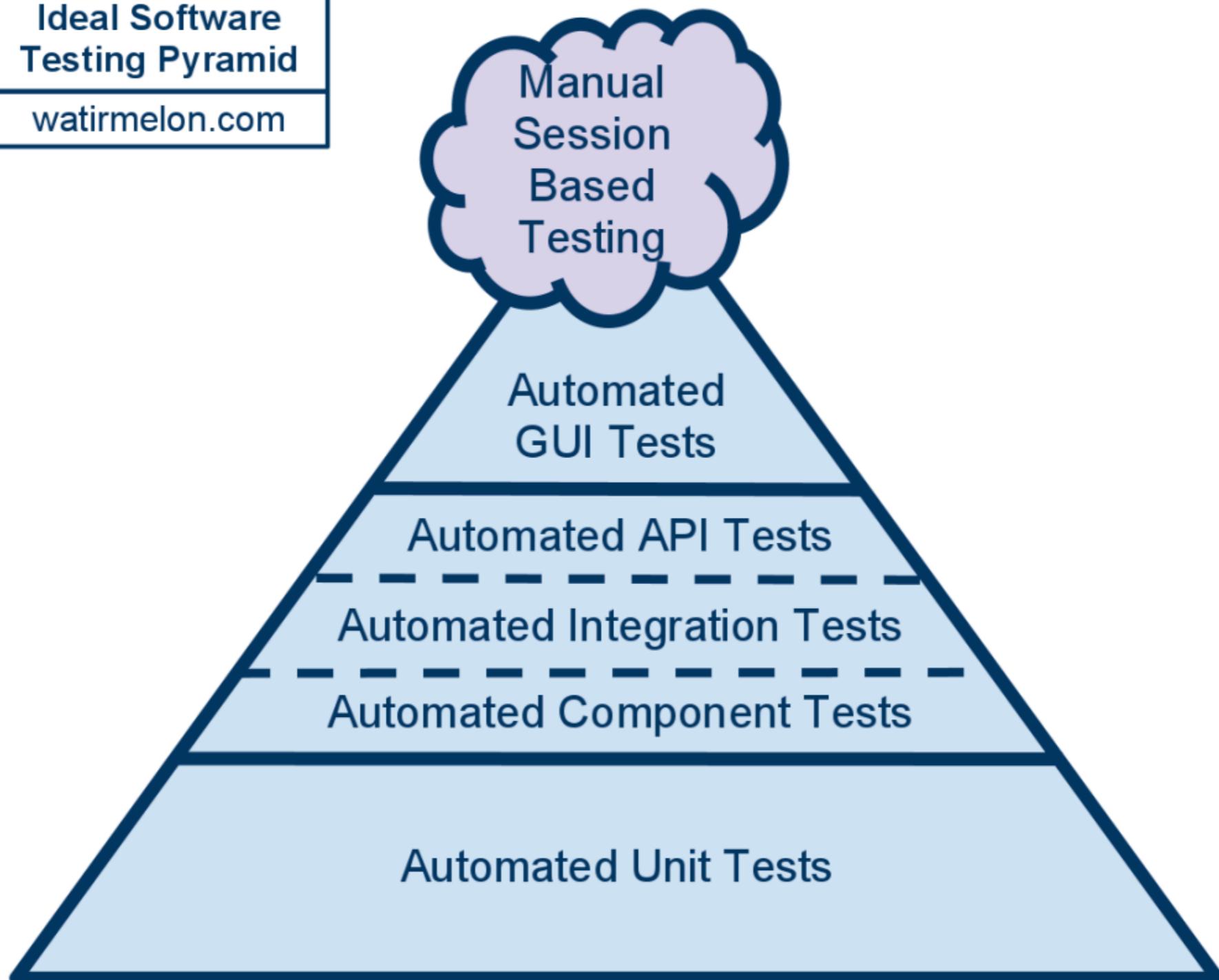
Multiple services

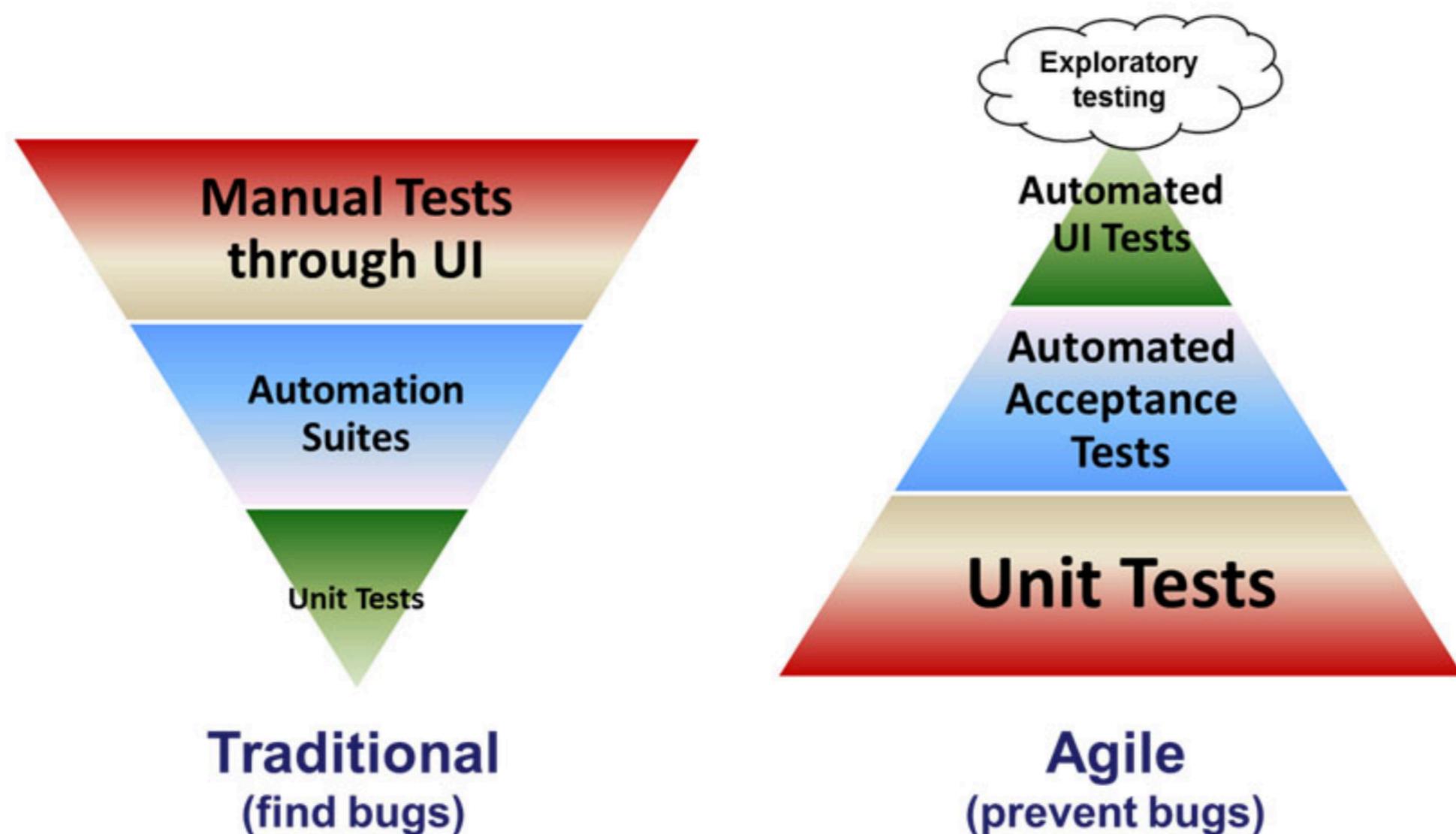


<https://martinfowler.com/articles/microservice-testing>

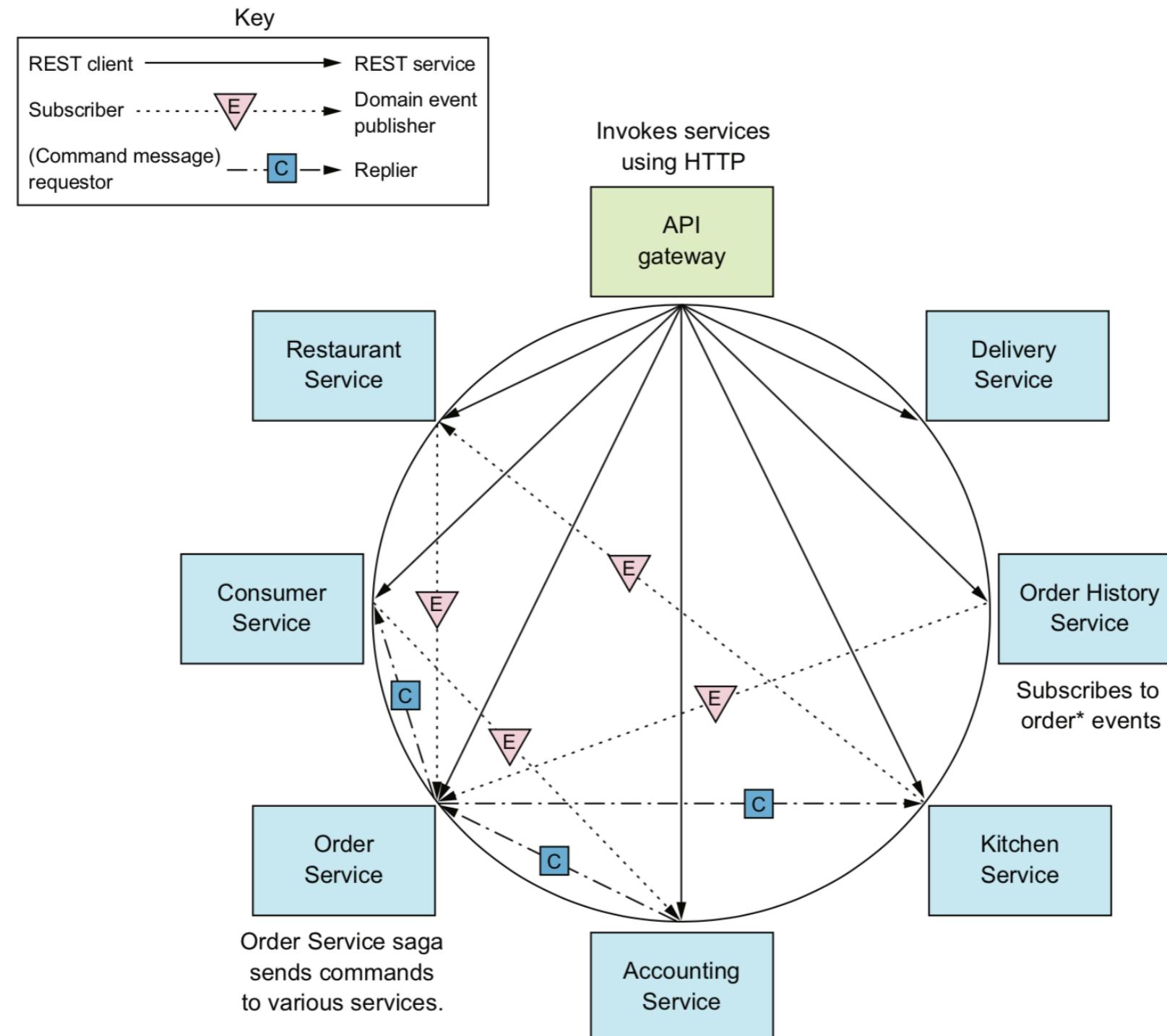




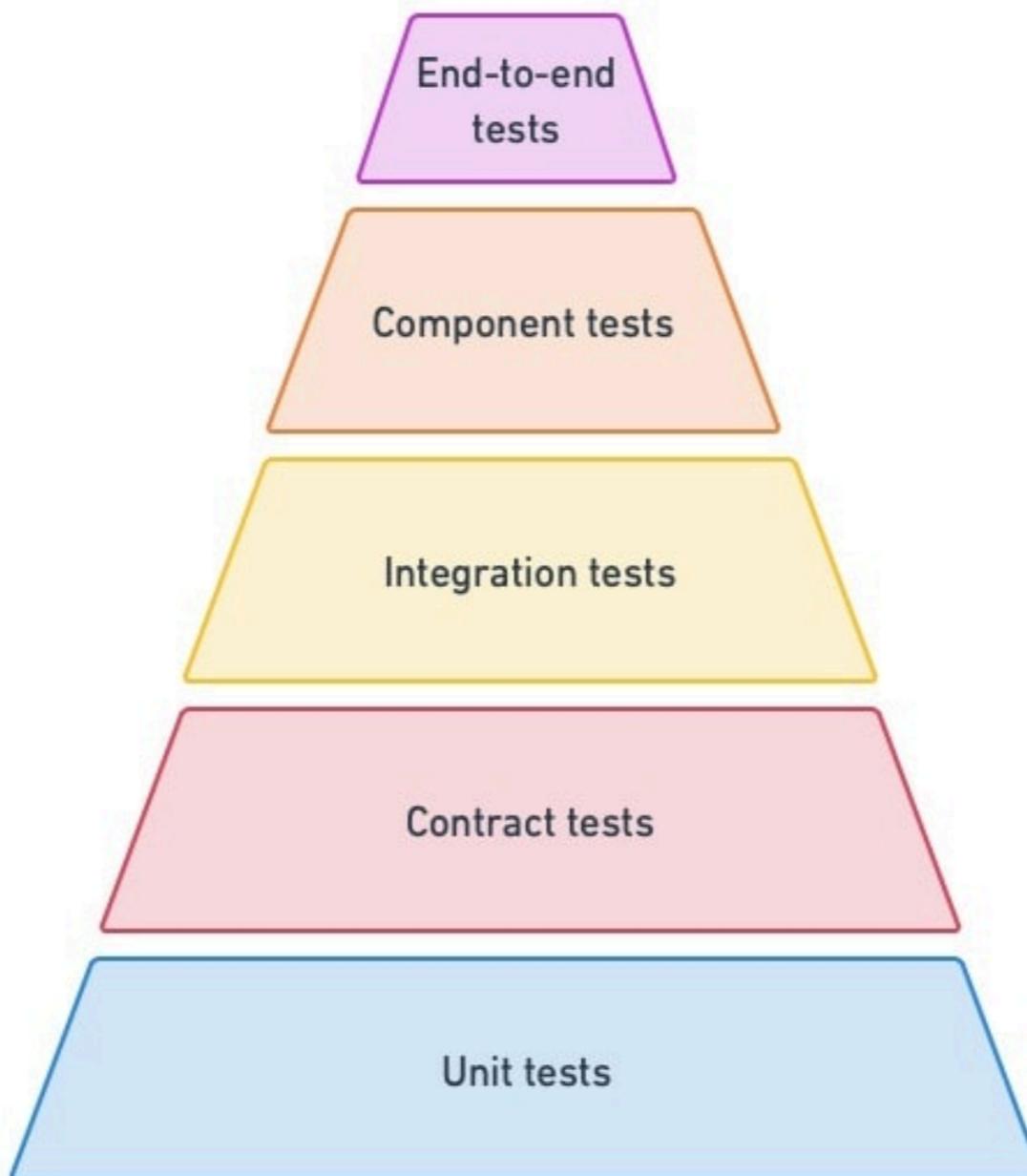




Testing in Microservice ?



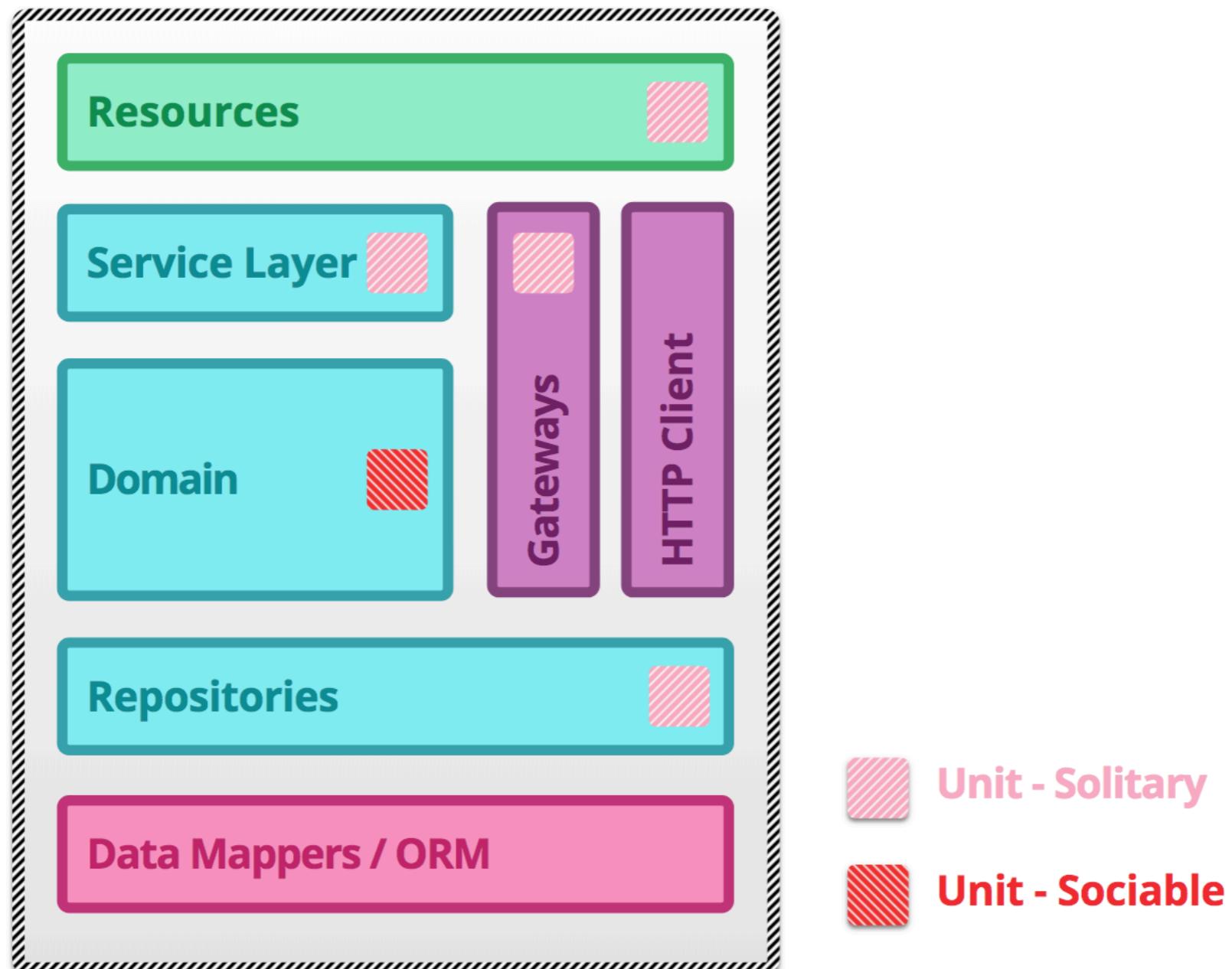
Microservice Testing



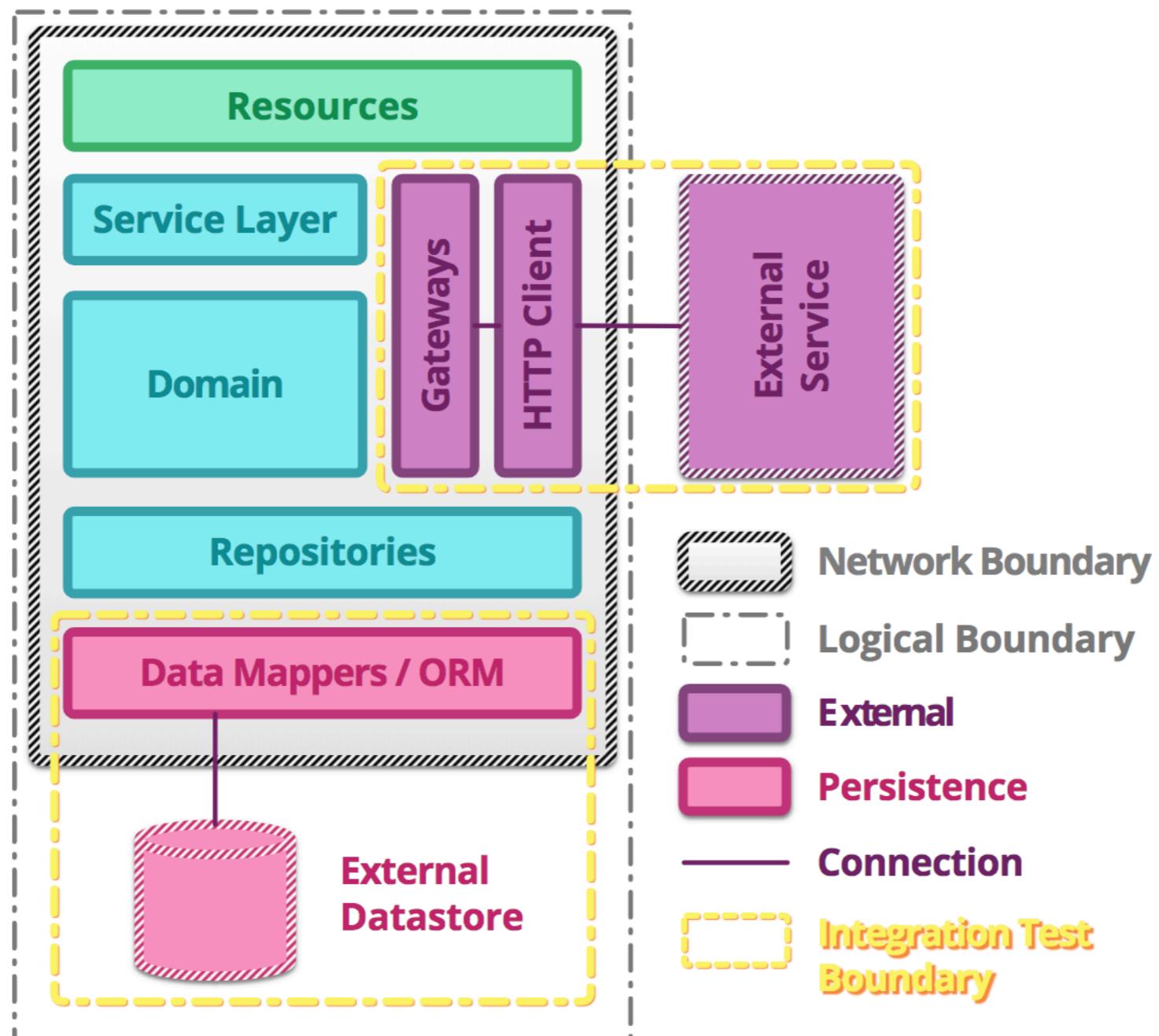
<https://semaphoreci.com/blog/test-microservices>



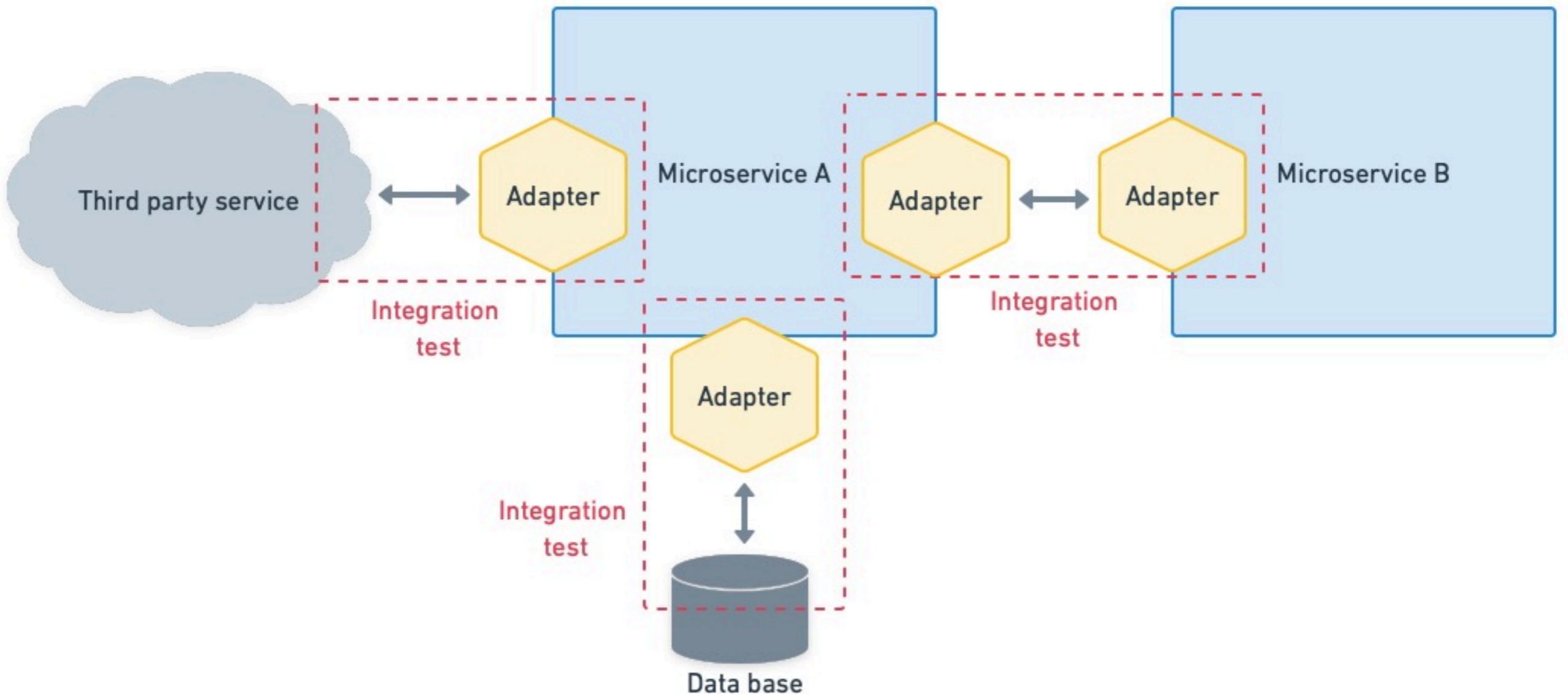
Unit testing



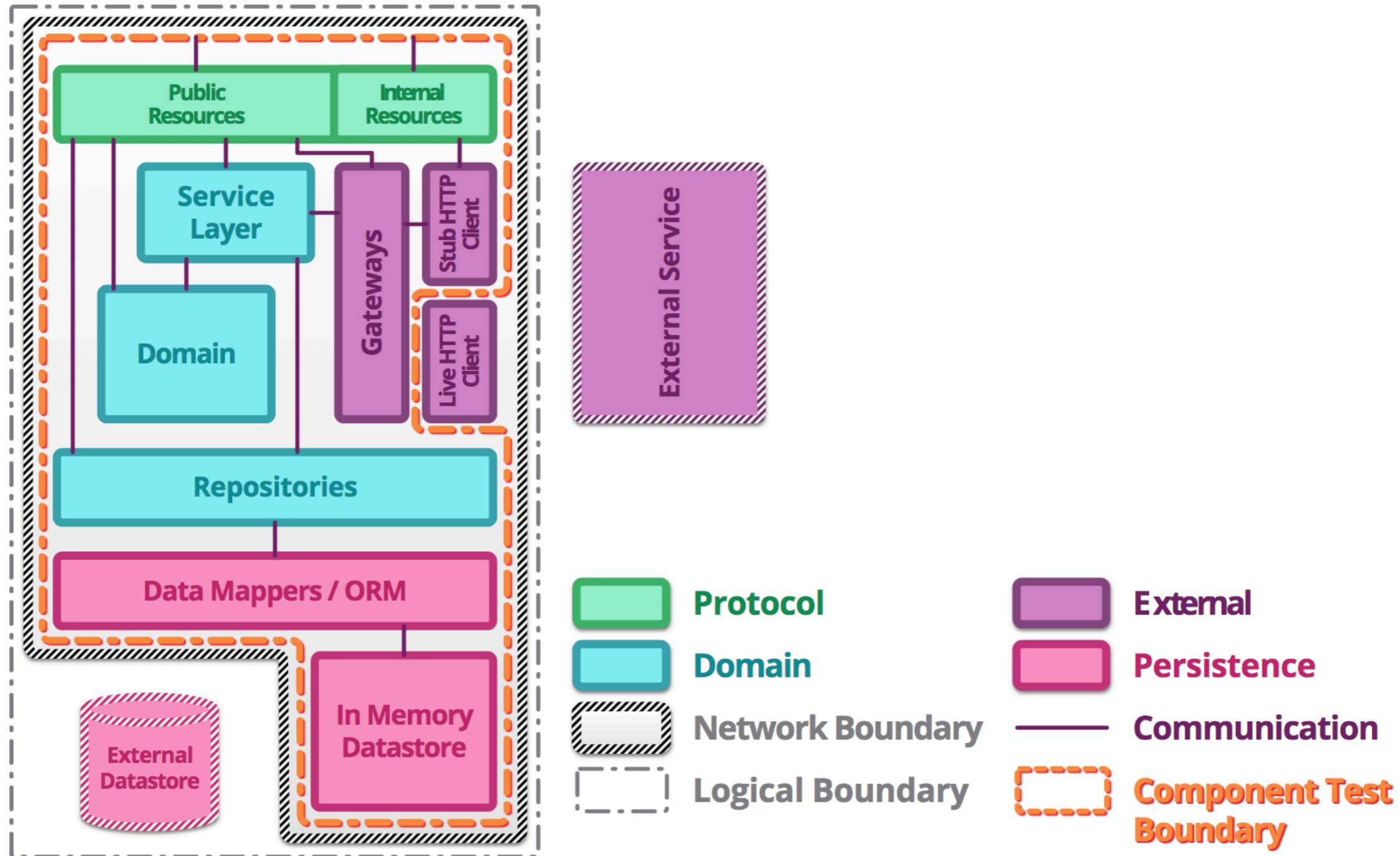
Integration testing



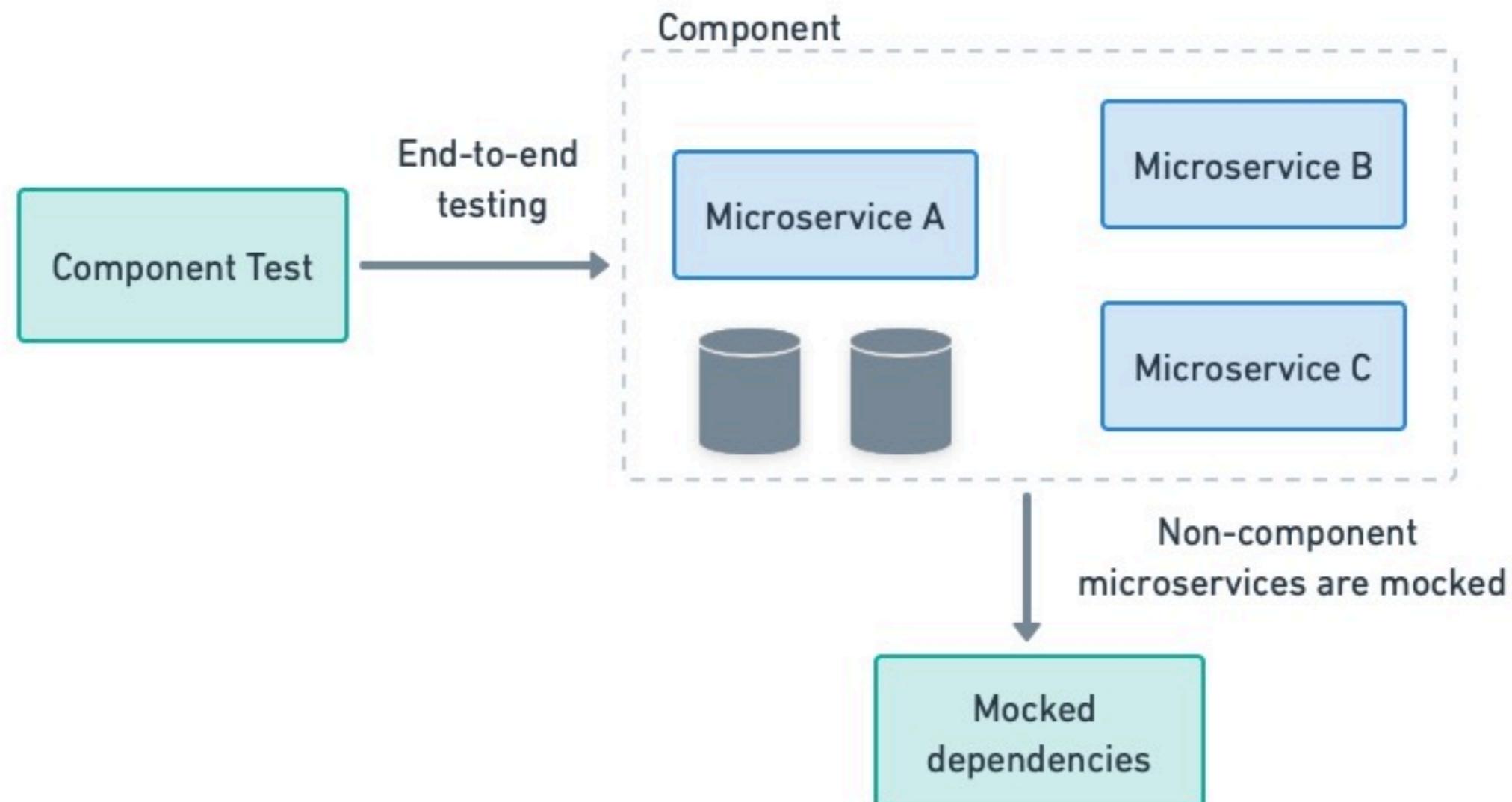
Integration testing



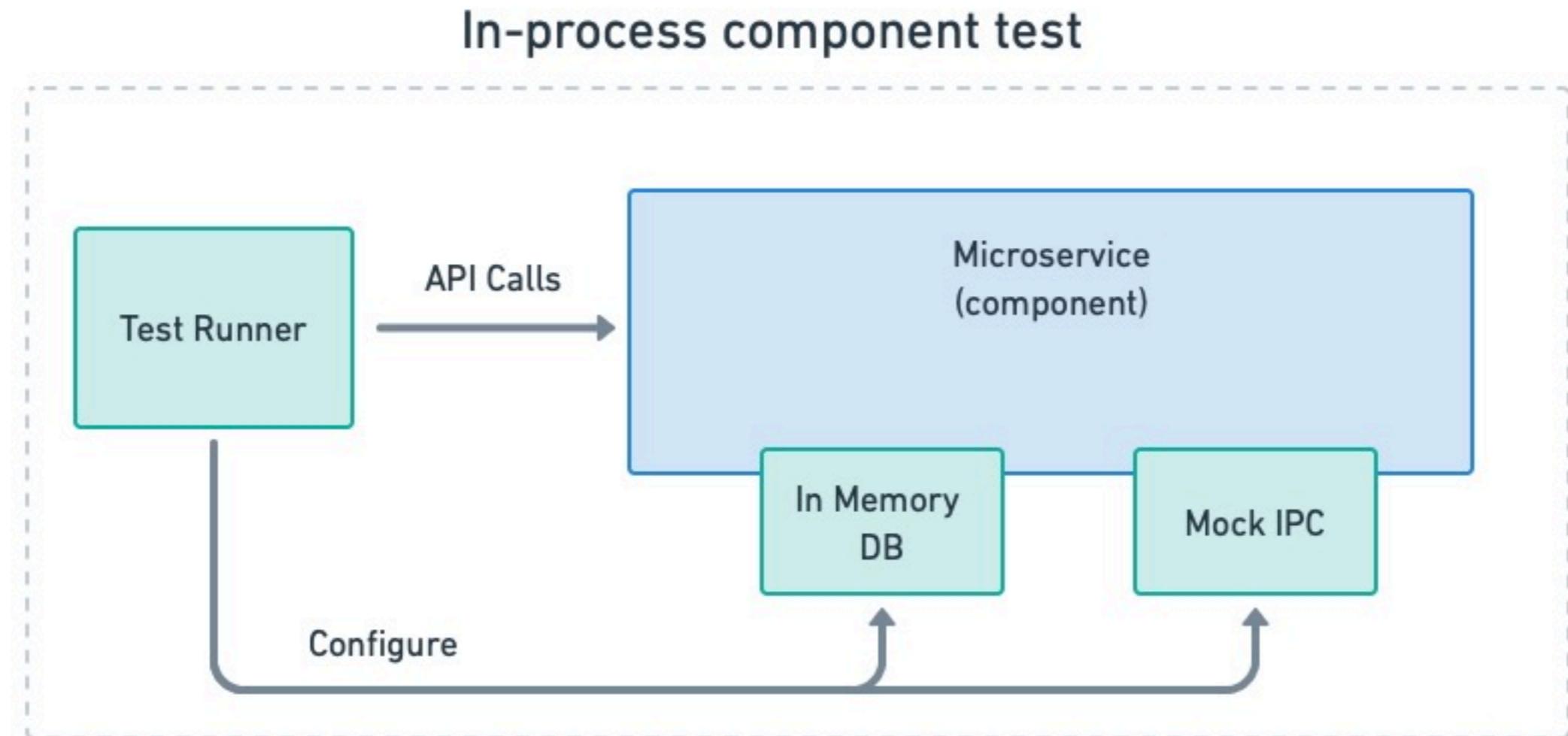
Component testing



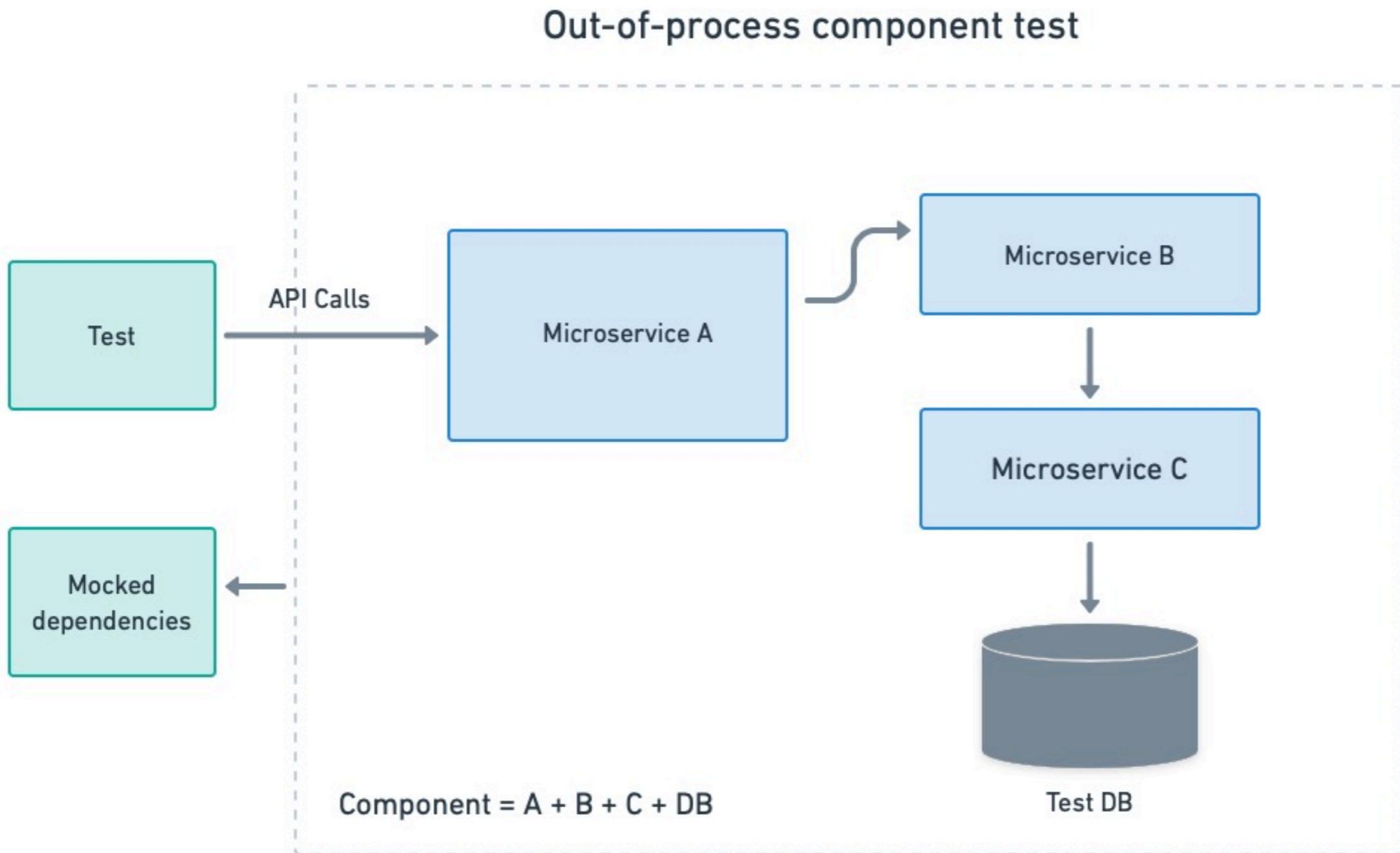
Component testing



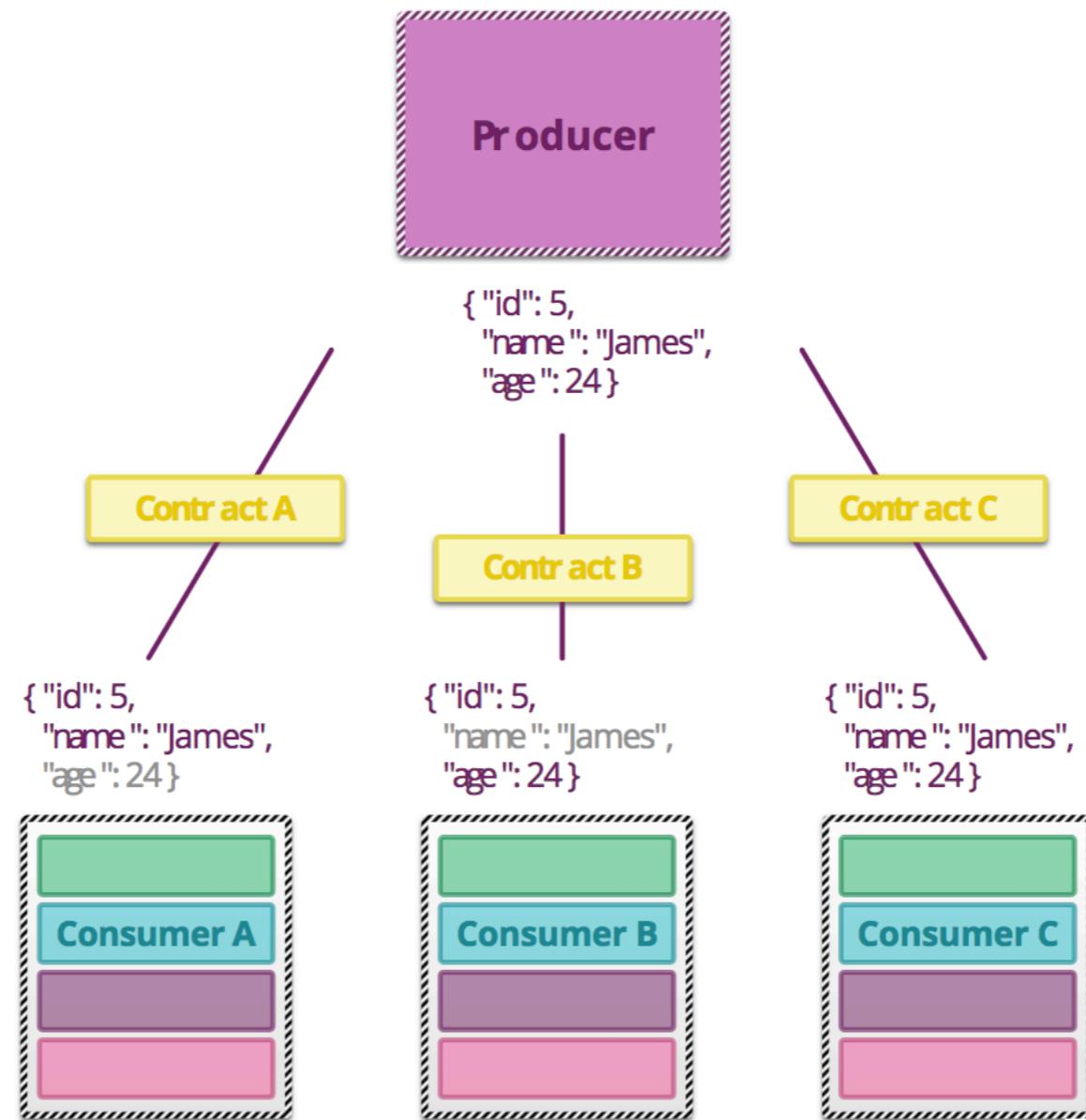
Component testing (in-process)



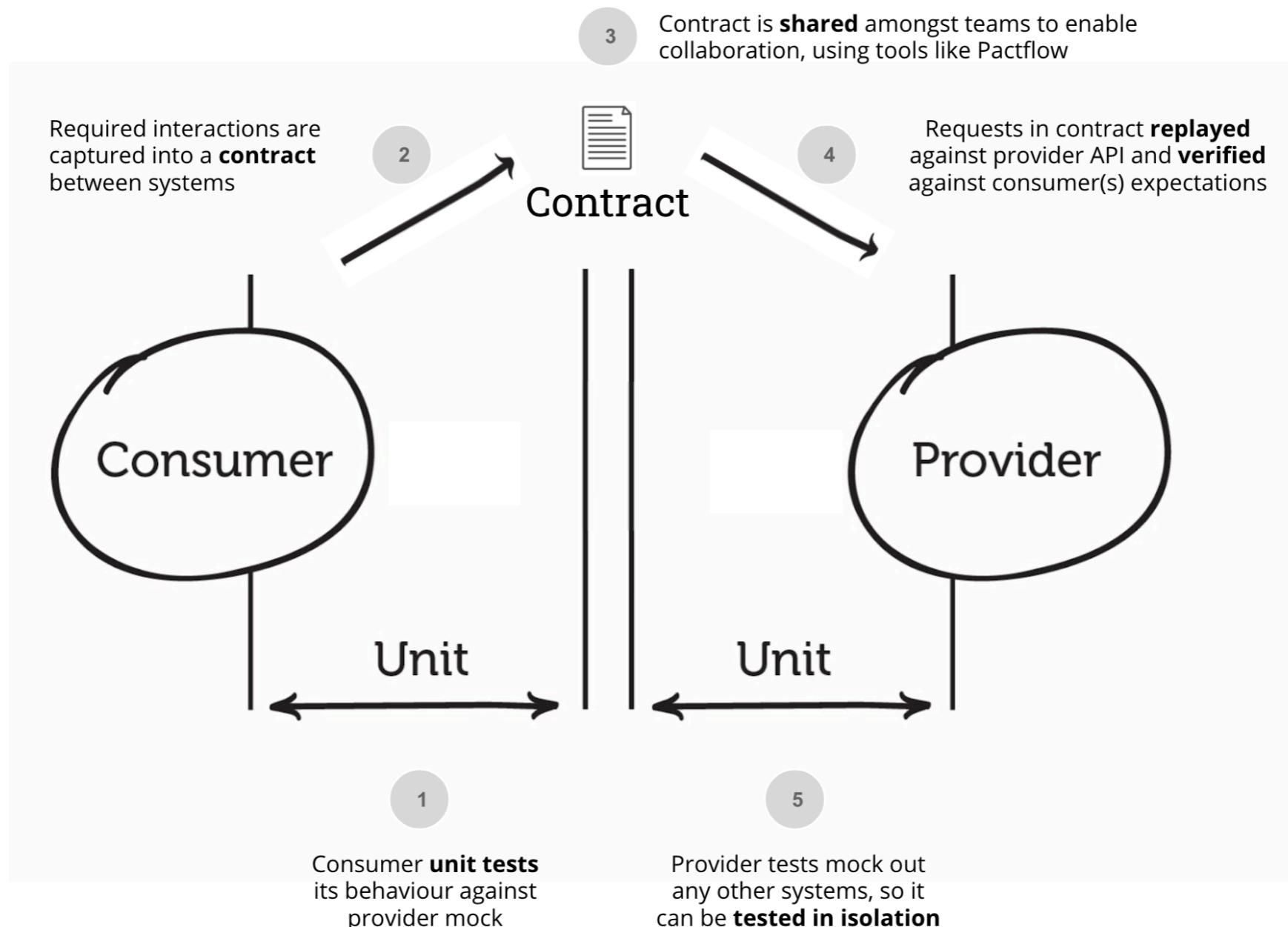
Component testing (out-process)



Contract testing



Contract testing with Pact



<https://docs.pact.io/>



Pact broker

Store and share all contracts and verification results

Consumer ↓↑	Provider ↓↑	Latest pact published	Last verified
Foo	Animals	2 minutes ago	2 days ago
Foo	Bar	7 days ago	15 days ago ▲
Foo	Hello World App	1 day ago	
Foo	Wiffles	less than a minute ago	7 days ago
Some other app	A service	26 days ago	less than a minute ago
The Android App	The back end	less than a minute ago	

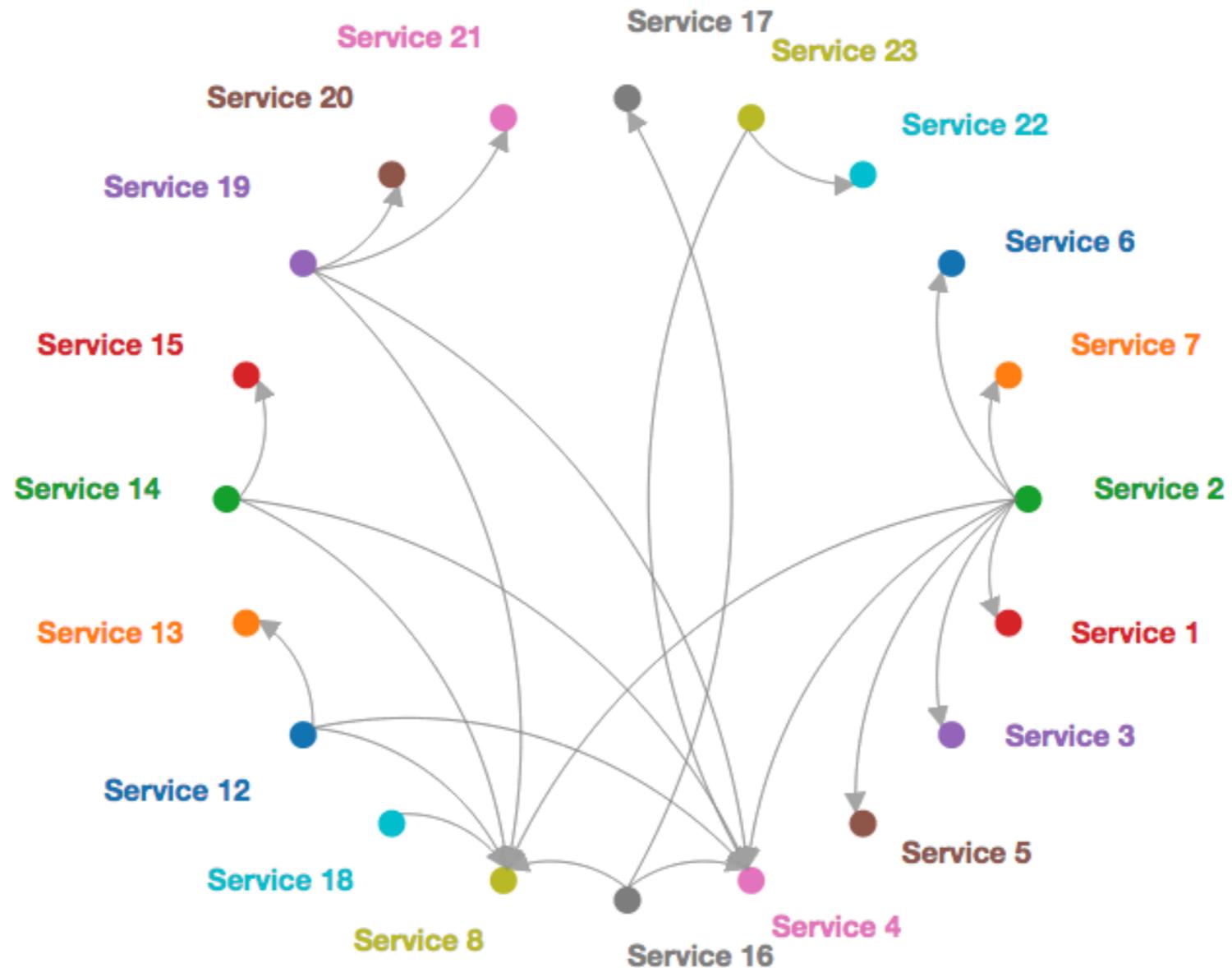
https://docs.pact.io/pact_broker



Microservices

© 2022 - 2023 Siam Chamnankit Company Limited. All rights reserved.

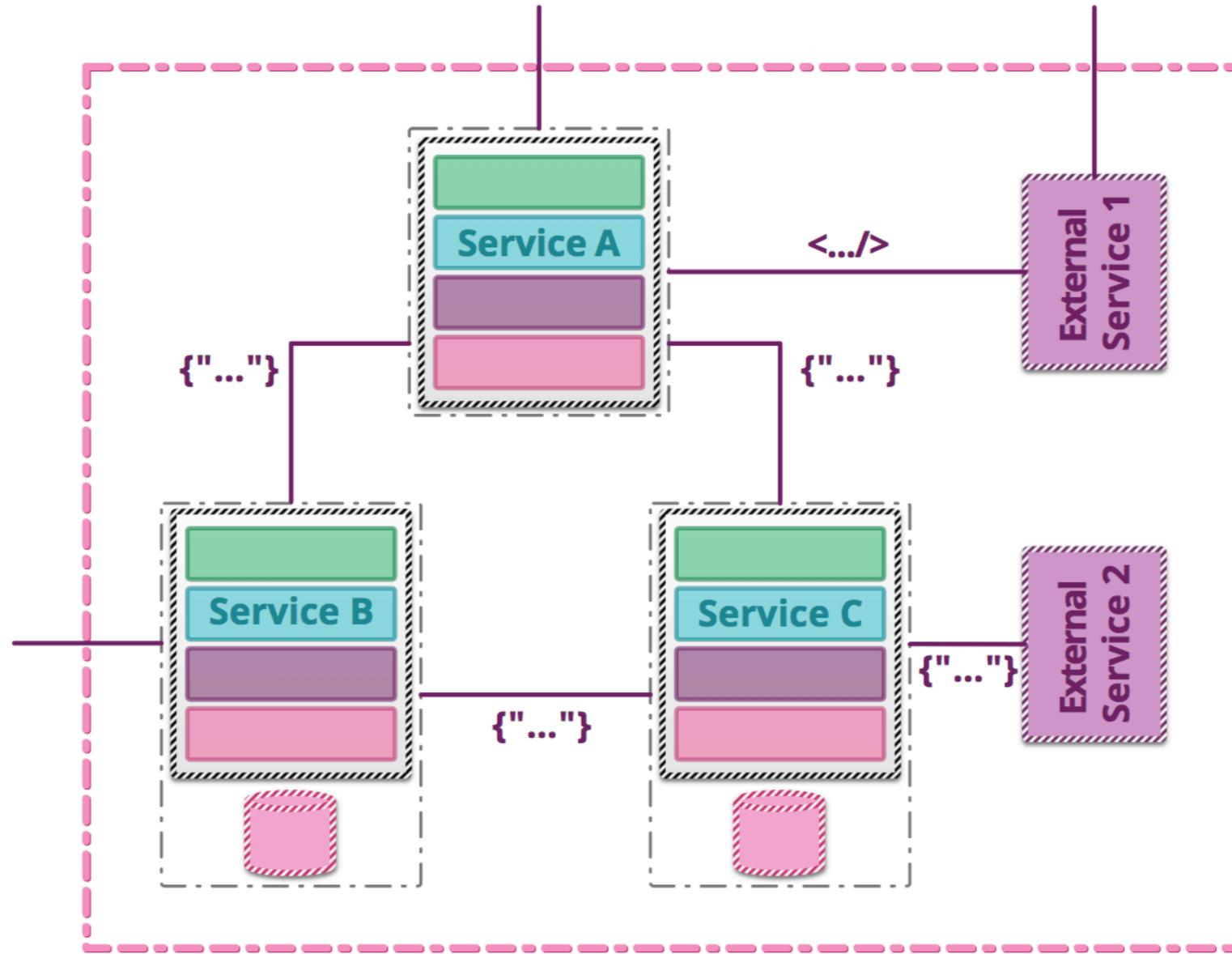
Pact broker



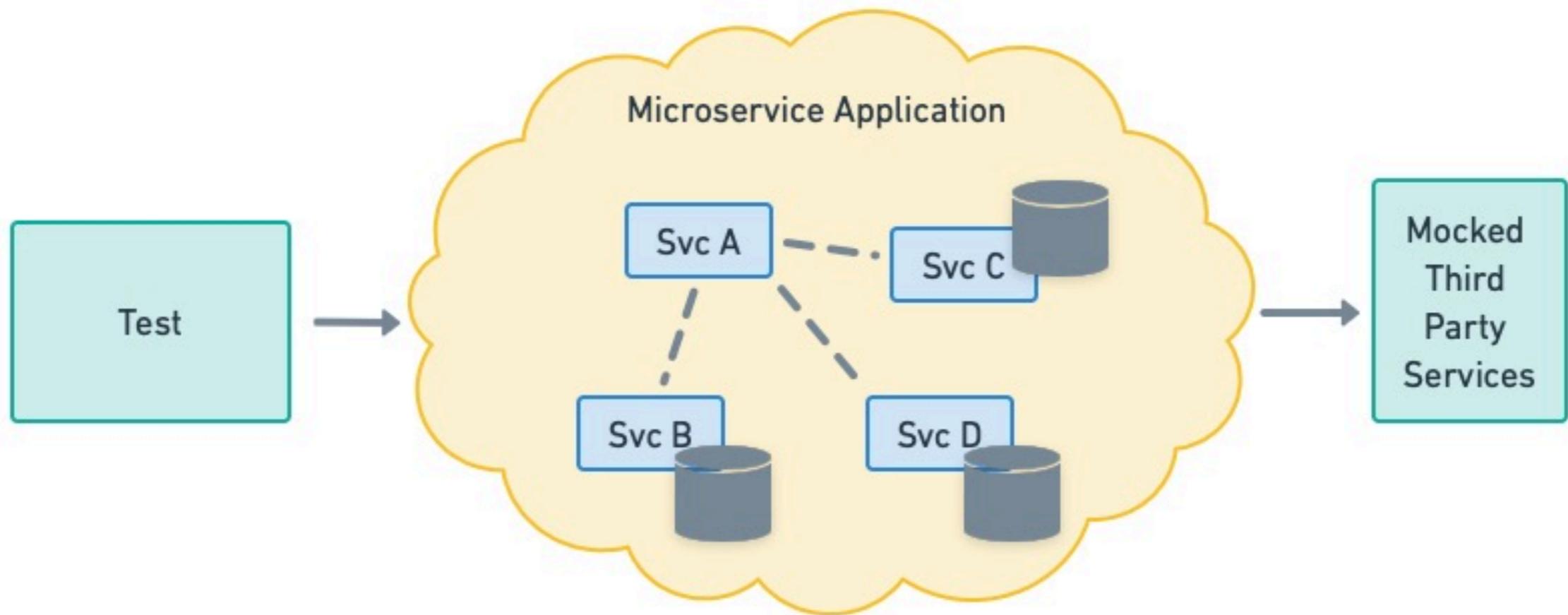
https://docs.pact.io/pact_broker



End-to-End testing



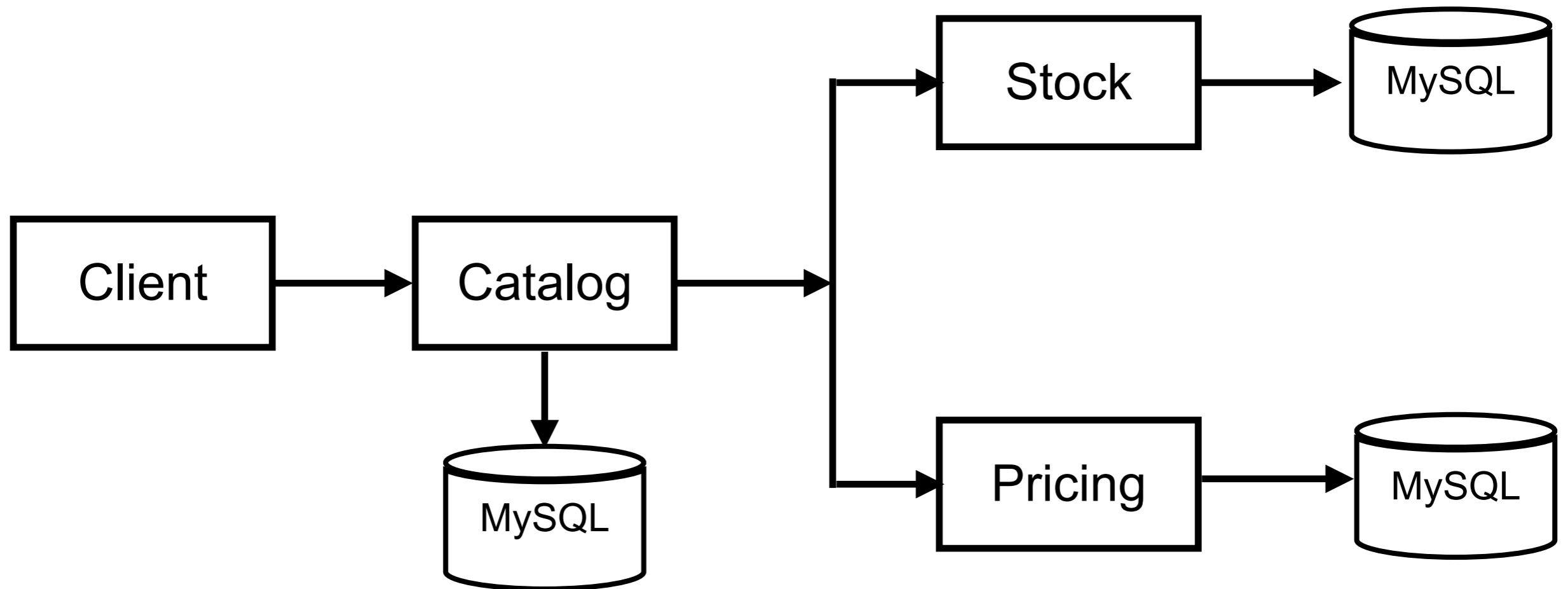
End-to-End testing



What is your testing strategy ?

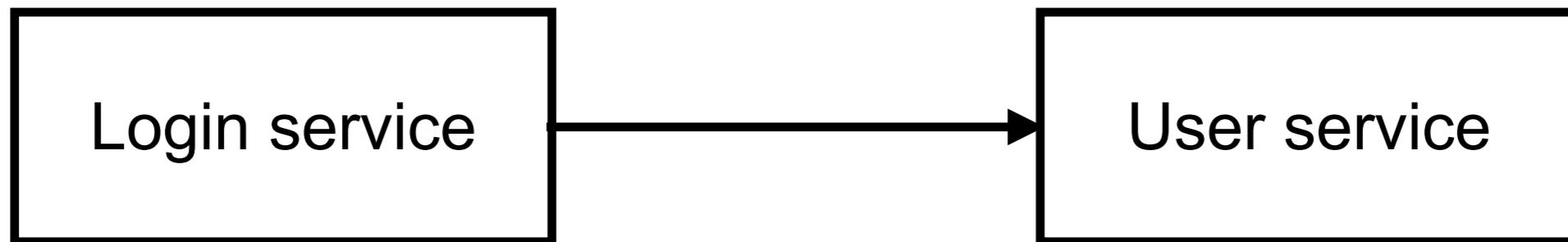


How to test ?



Testing workshop

Component testing
Contract testing



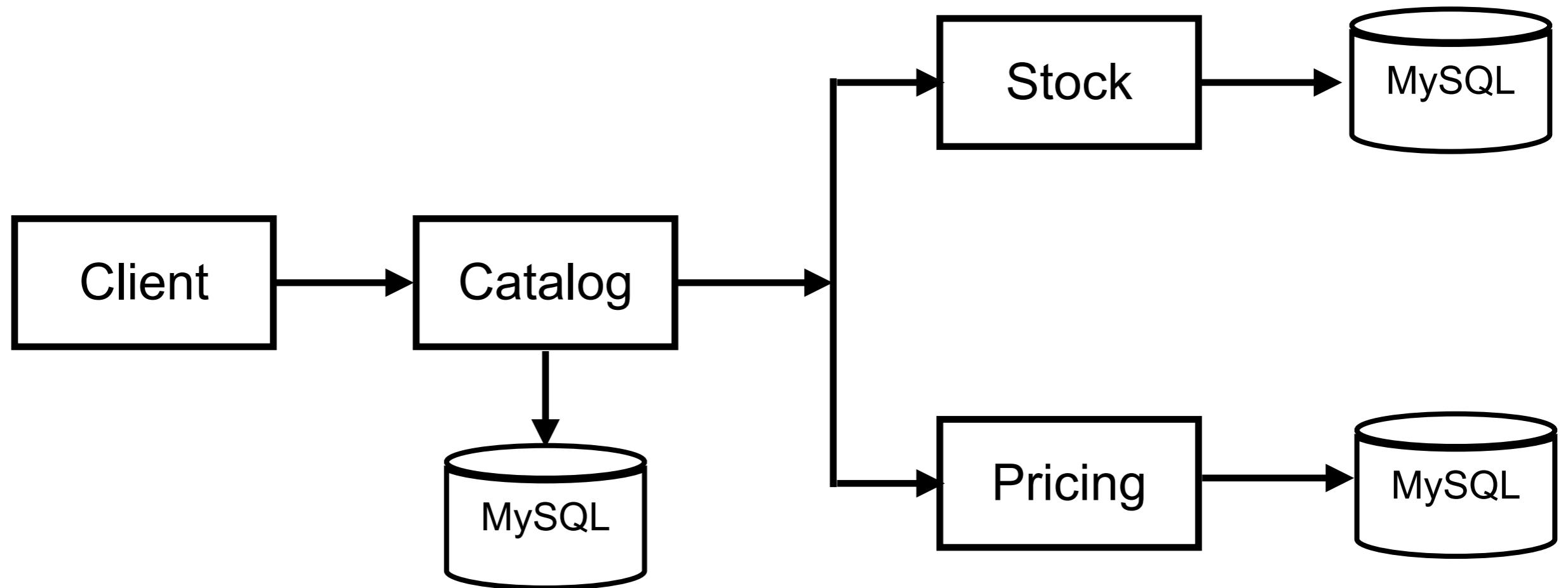
<https://github.com/up1/course-contract-testing>



Performance testing ?
Security testing ?



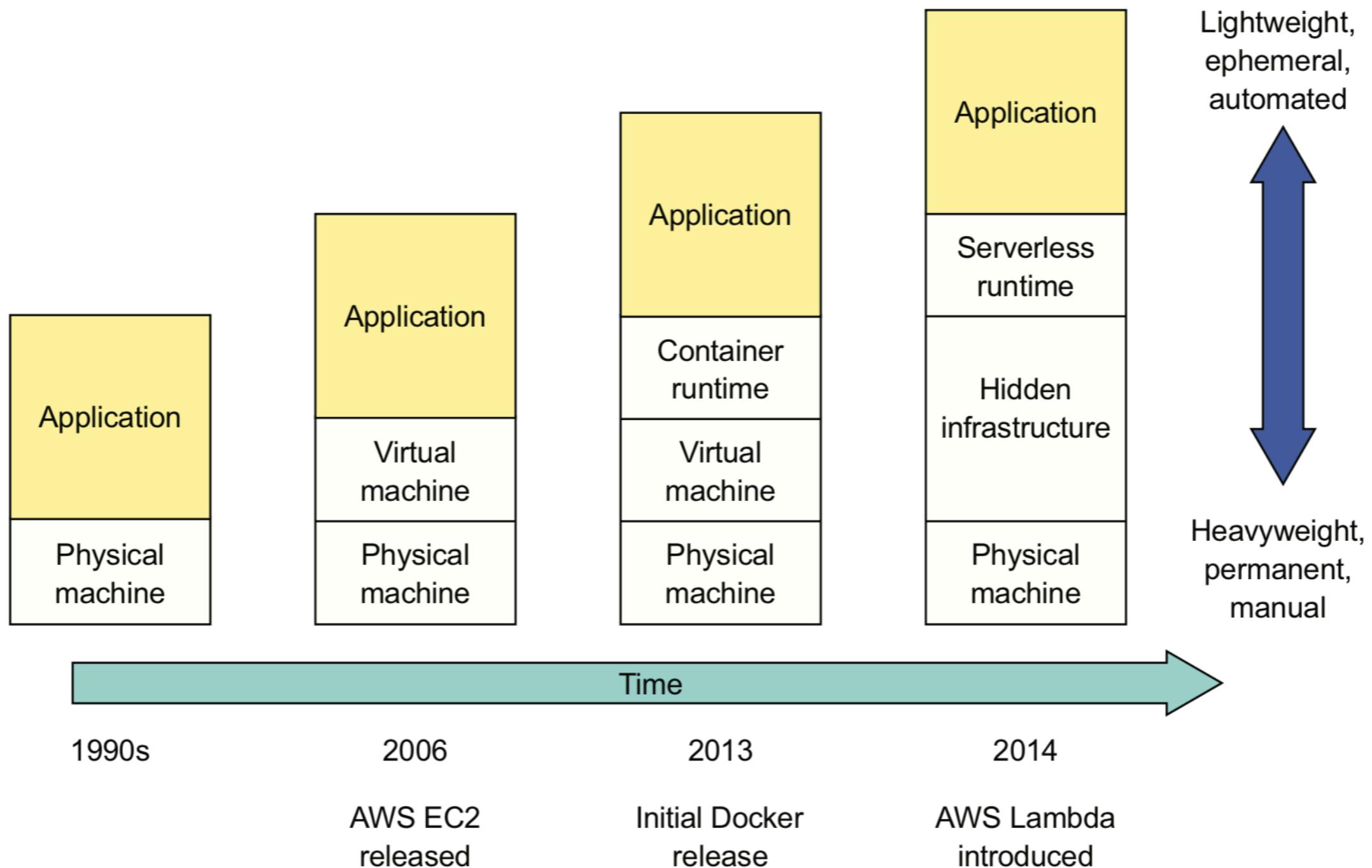
How to performance test ?



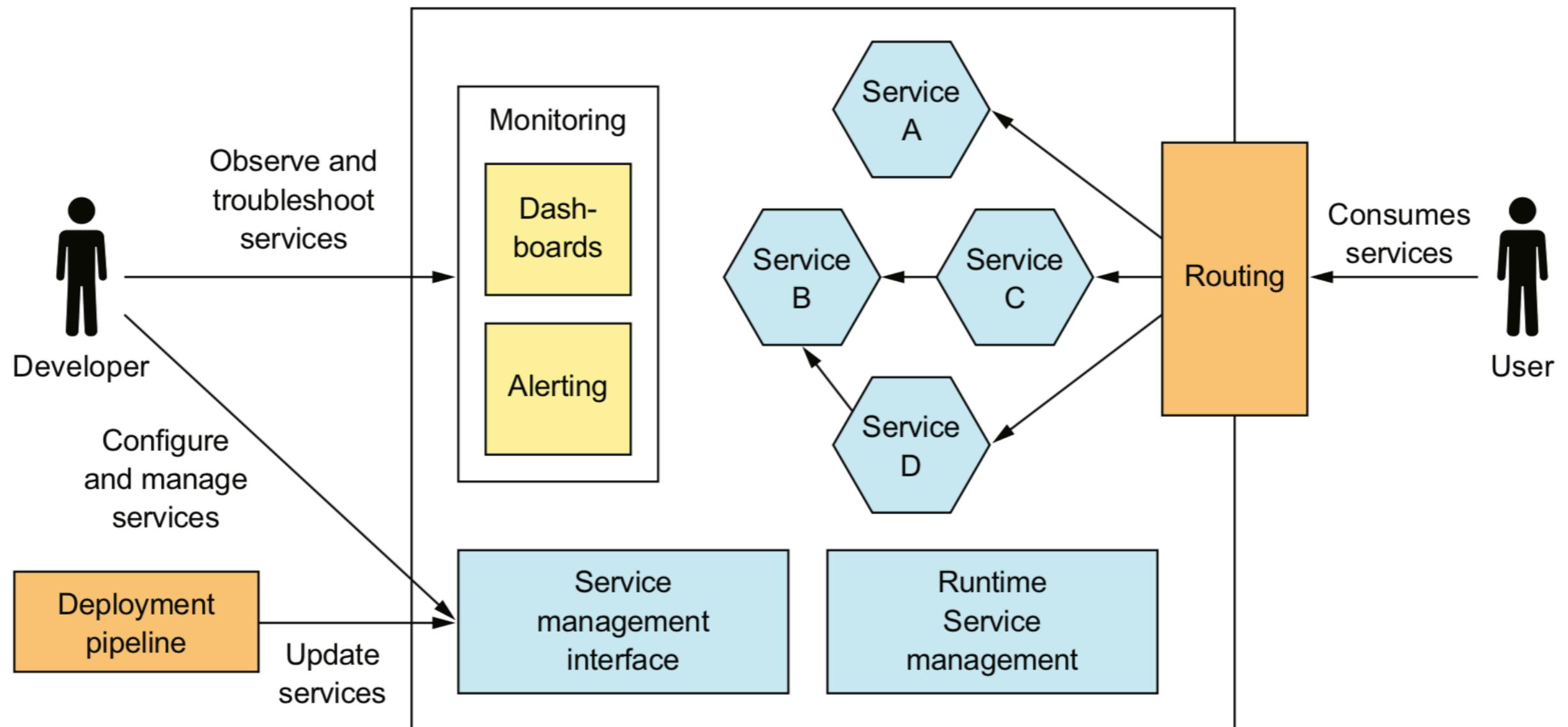
Module 3 : Deploy



Infrastructure



View of production environment



Deploy Microservice

Language-specific packaging

Virtual Machine (VM)

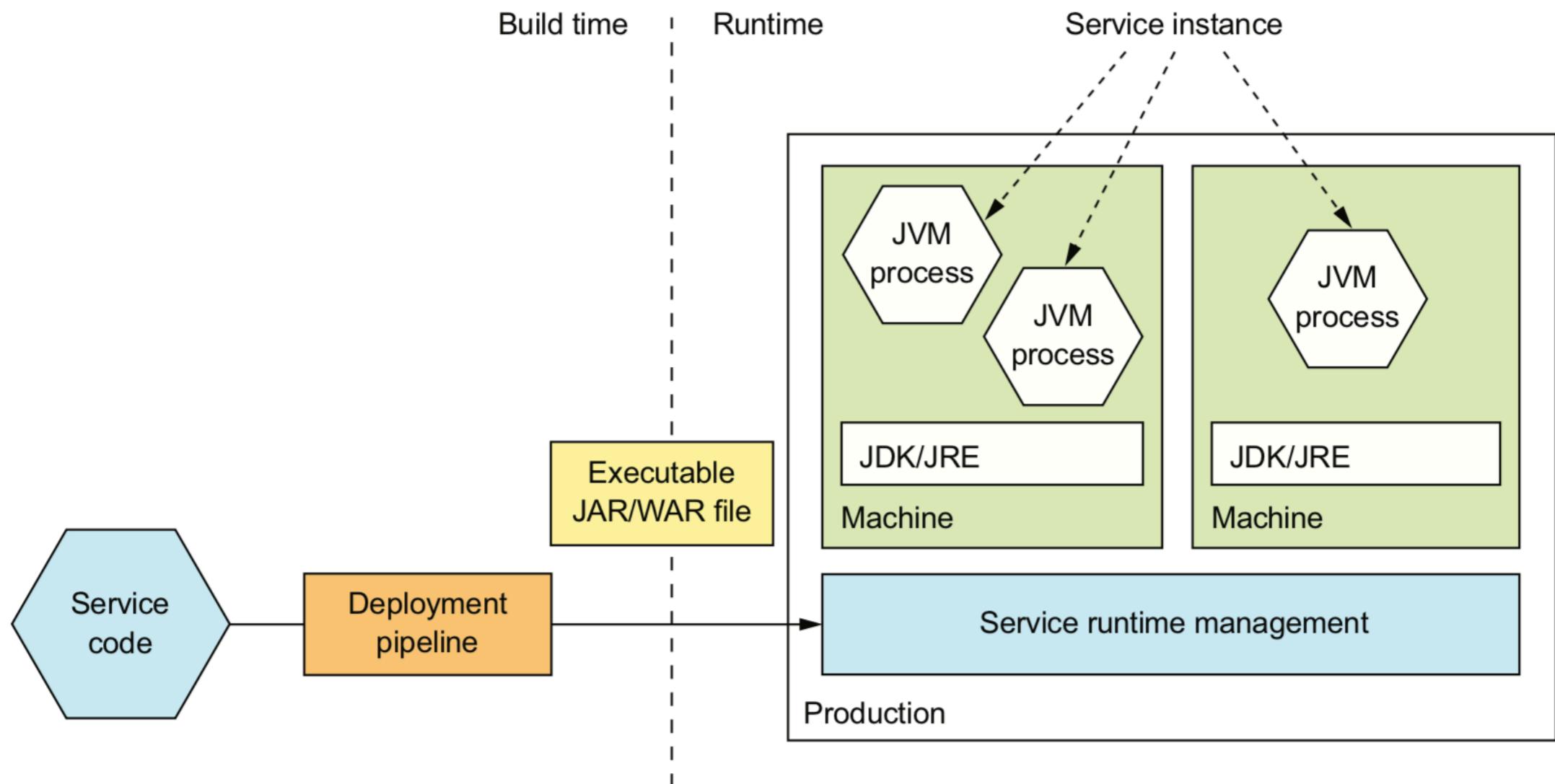
Container

Kubernetes

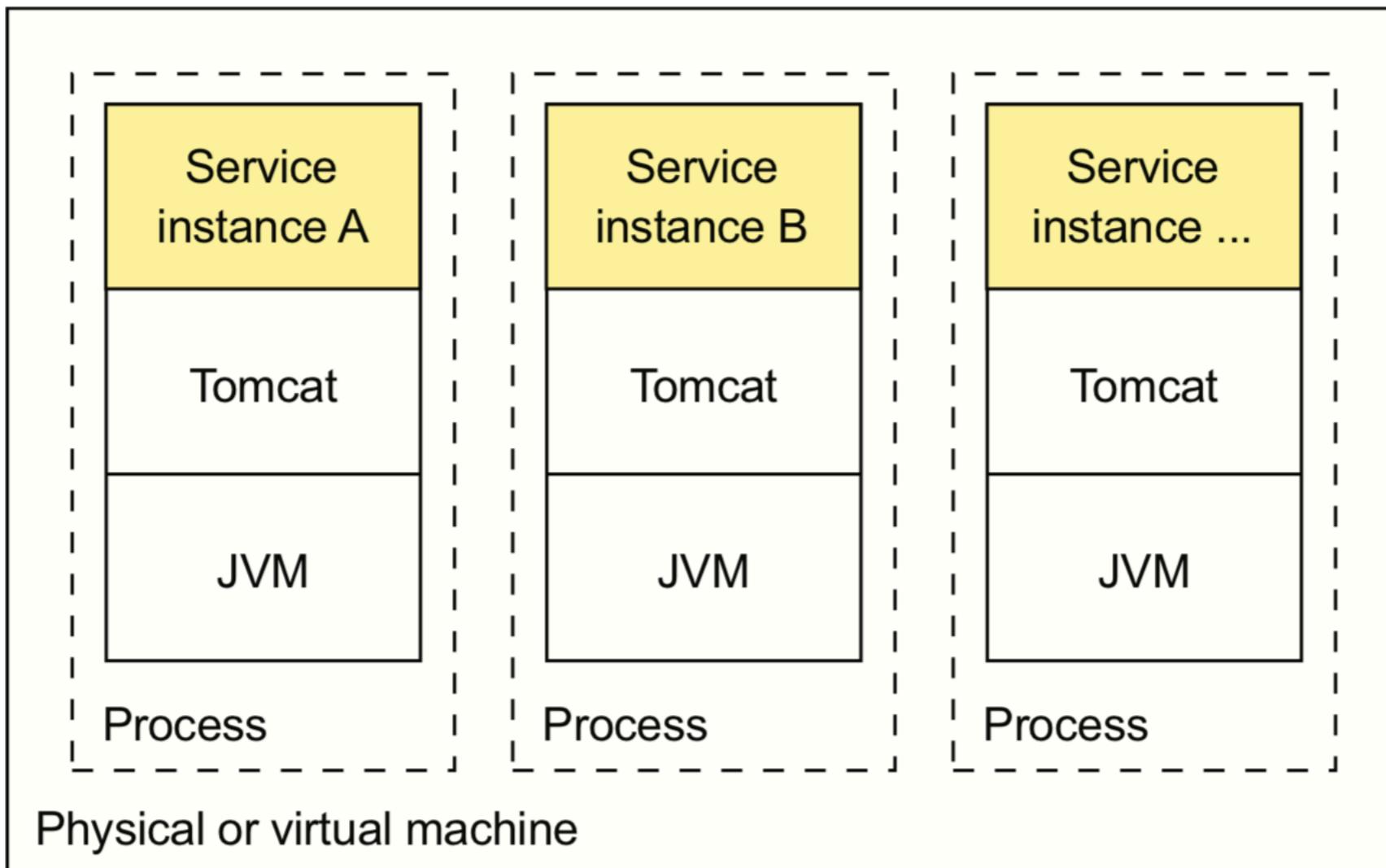
Serverless/FaaS



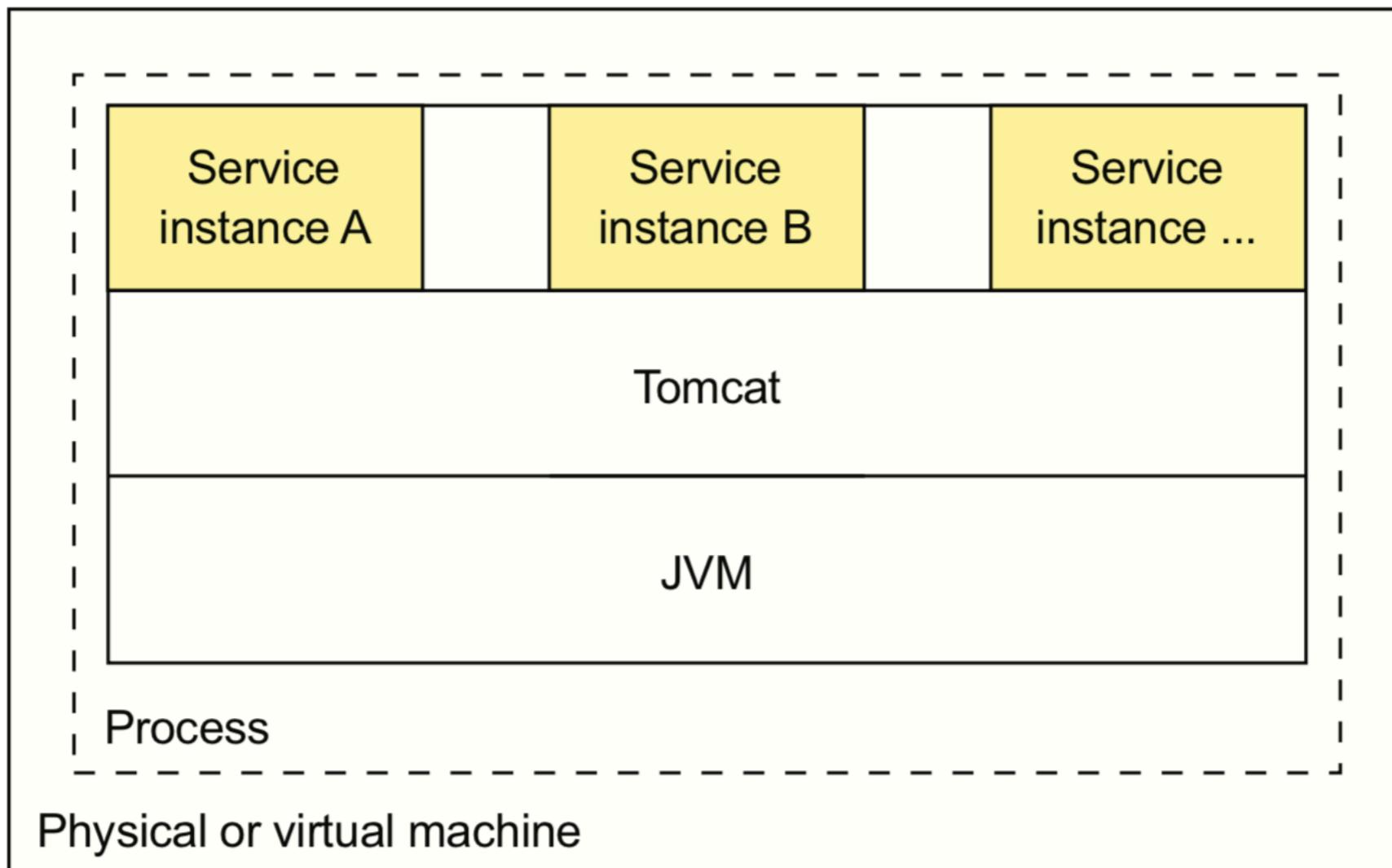
1. Language-specific packaging



Multiple services on same machine



Multiple services on same process



Benefits

Fast deployment

Efficient resource utilization

Service instances's resources are constrained



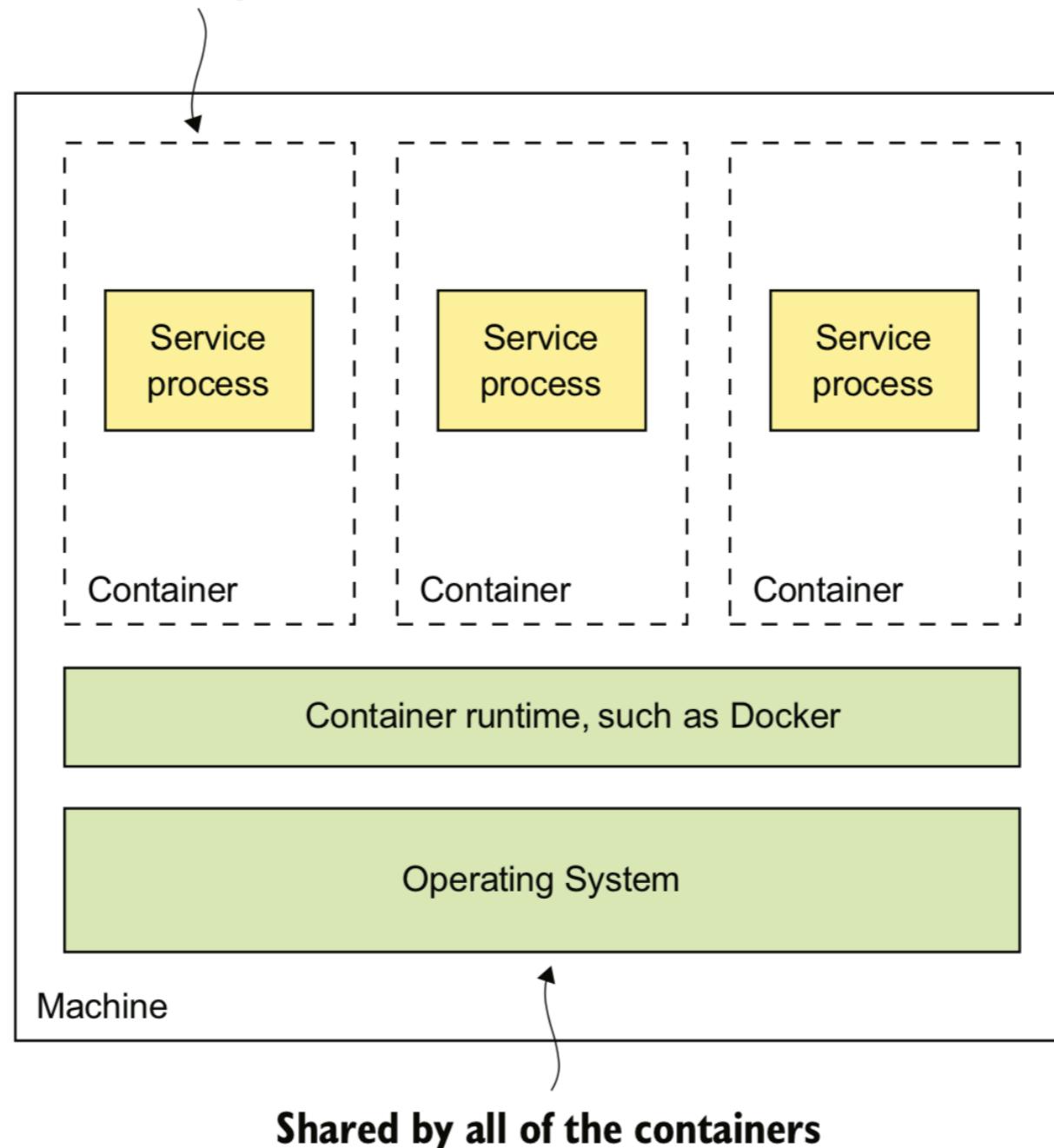
Drawbacks

Lack of encapsulation of technology stack
No ability to constrain resources of service
Lack of isolation

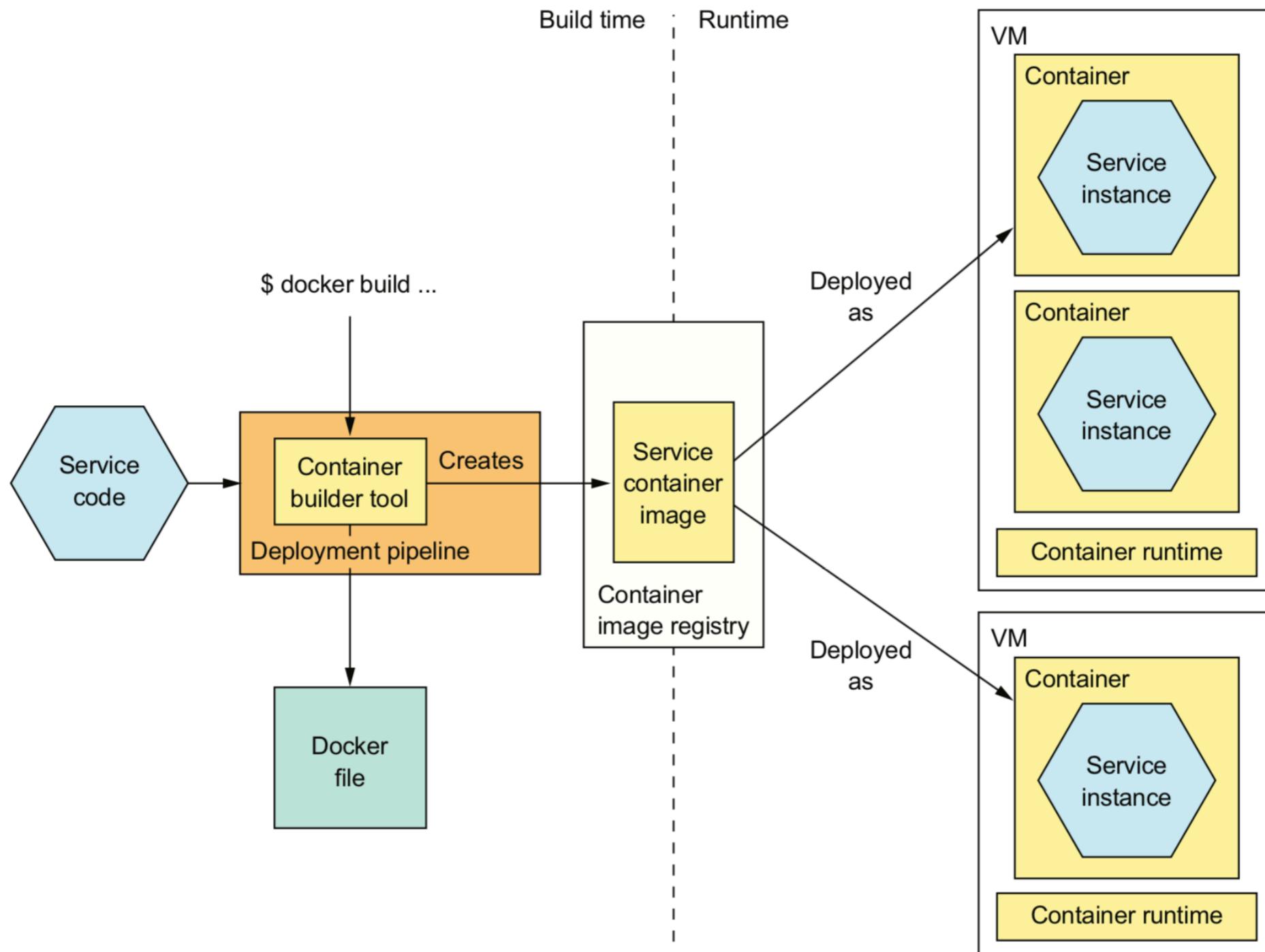


2. Working with container

**Each container is a sandbox
that isolates the processes.**



Deployment with container



Deploy services with Docker

Build a docker image

Push docker image to a registry

Run docker container

Working with docker-compose



Benefits

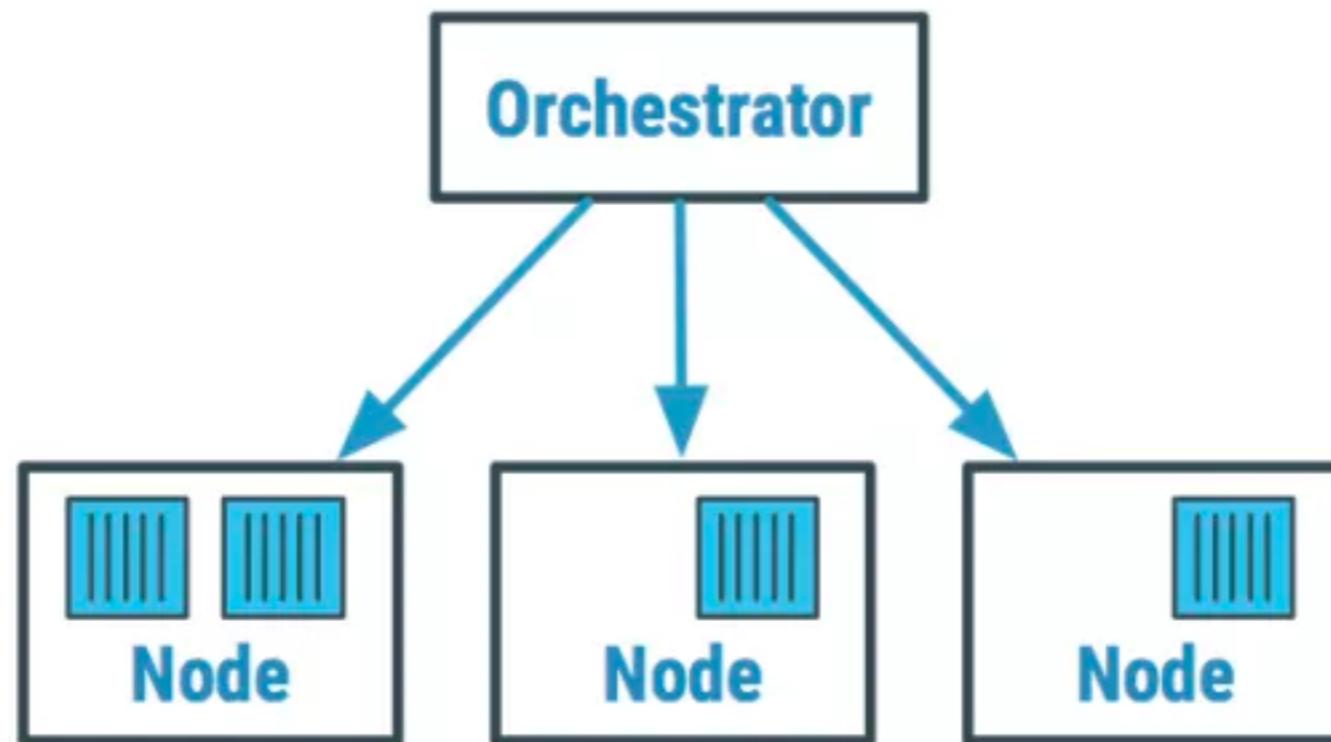
Encapsulate technology stack

Service instances are isolated

Service instances's resources are constrained



Container Orchestration ?



Orchestration tools

Configuration Management



CI/CD orchestration



Container orchestration



Cloud-specific orchestration



PaaS orchestration



Microservices

© 2022 - 2023 Siam Chamnankit Company Limited. All rights reserved.

Application Deployment Strategies



Strategies to deploy

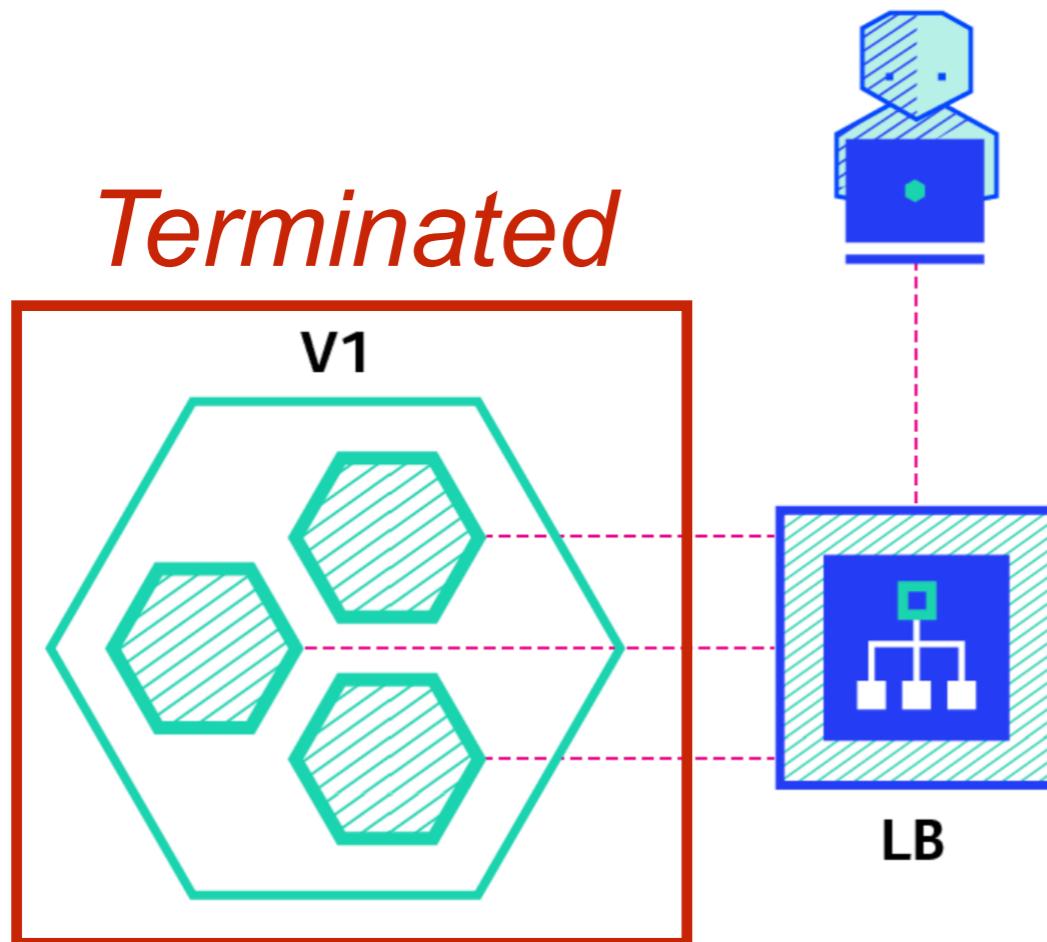
Recreate
Ramped
Blue/Green
Canary
A/B testing
Shadow

<https://thenewstack.io/deployment-strategies/>



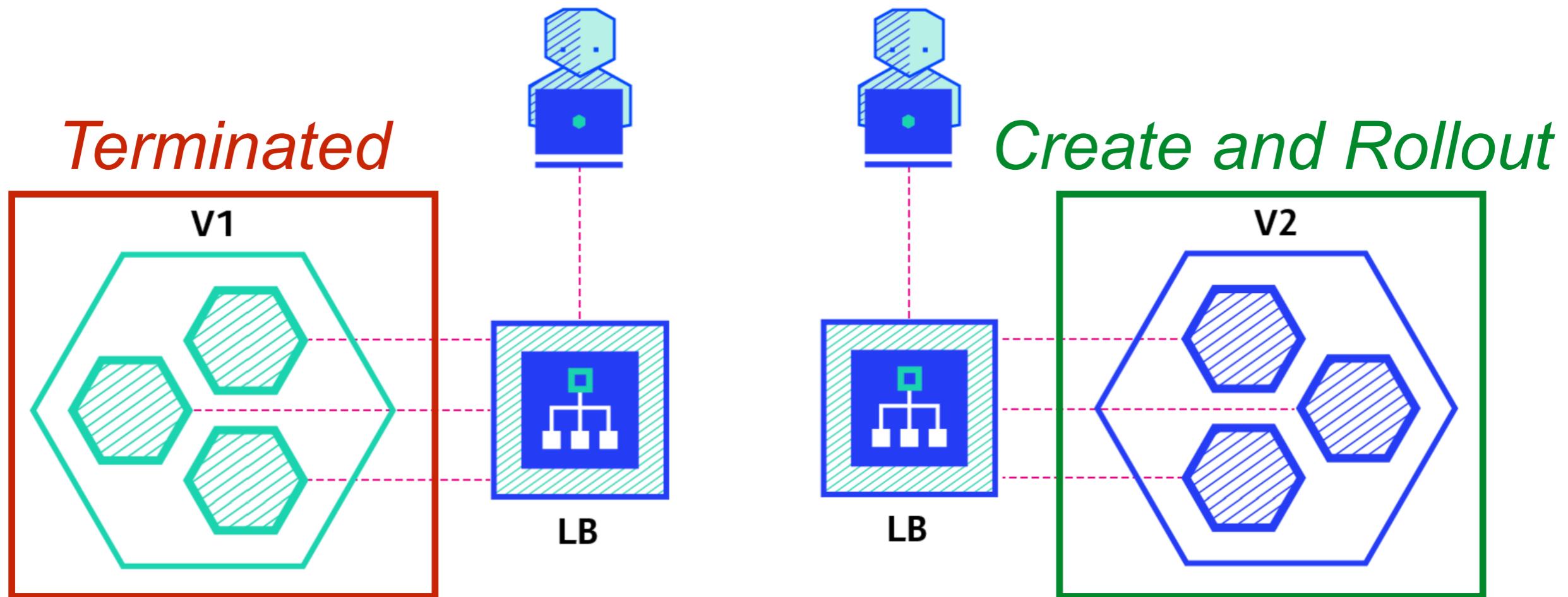
1. Recreate

Version A is terminated then version B is rollout



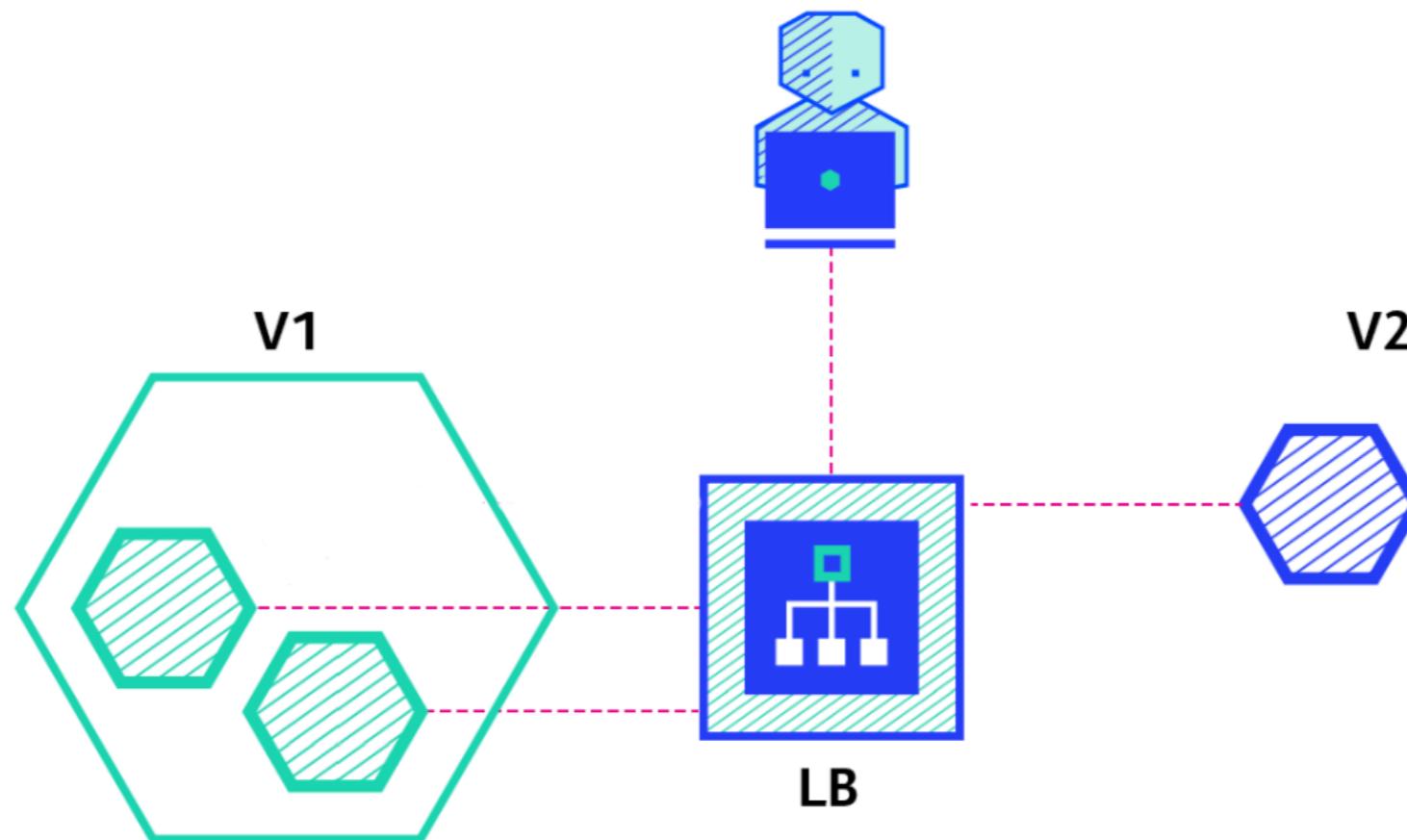
1. Recreate

Version A is terminated then version B is rollout



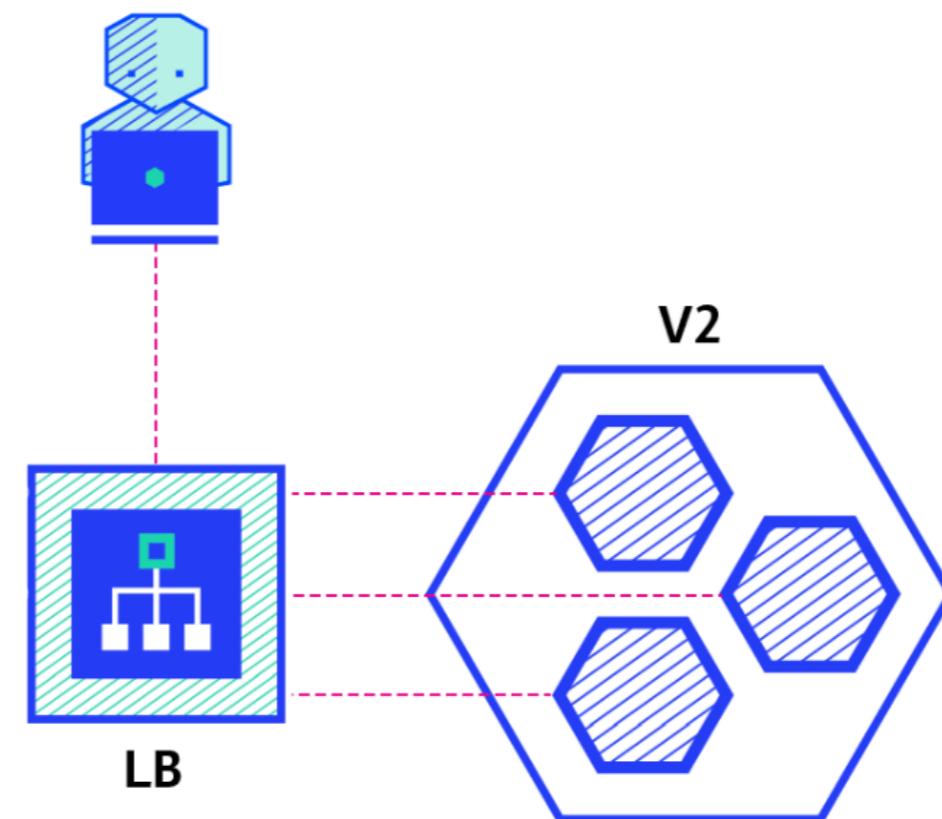
2. Ramped

Slow roll out by replace instance one-by-one



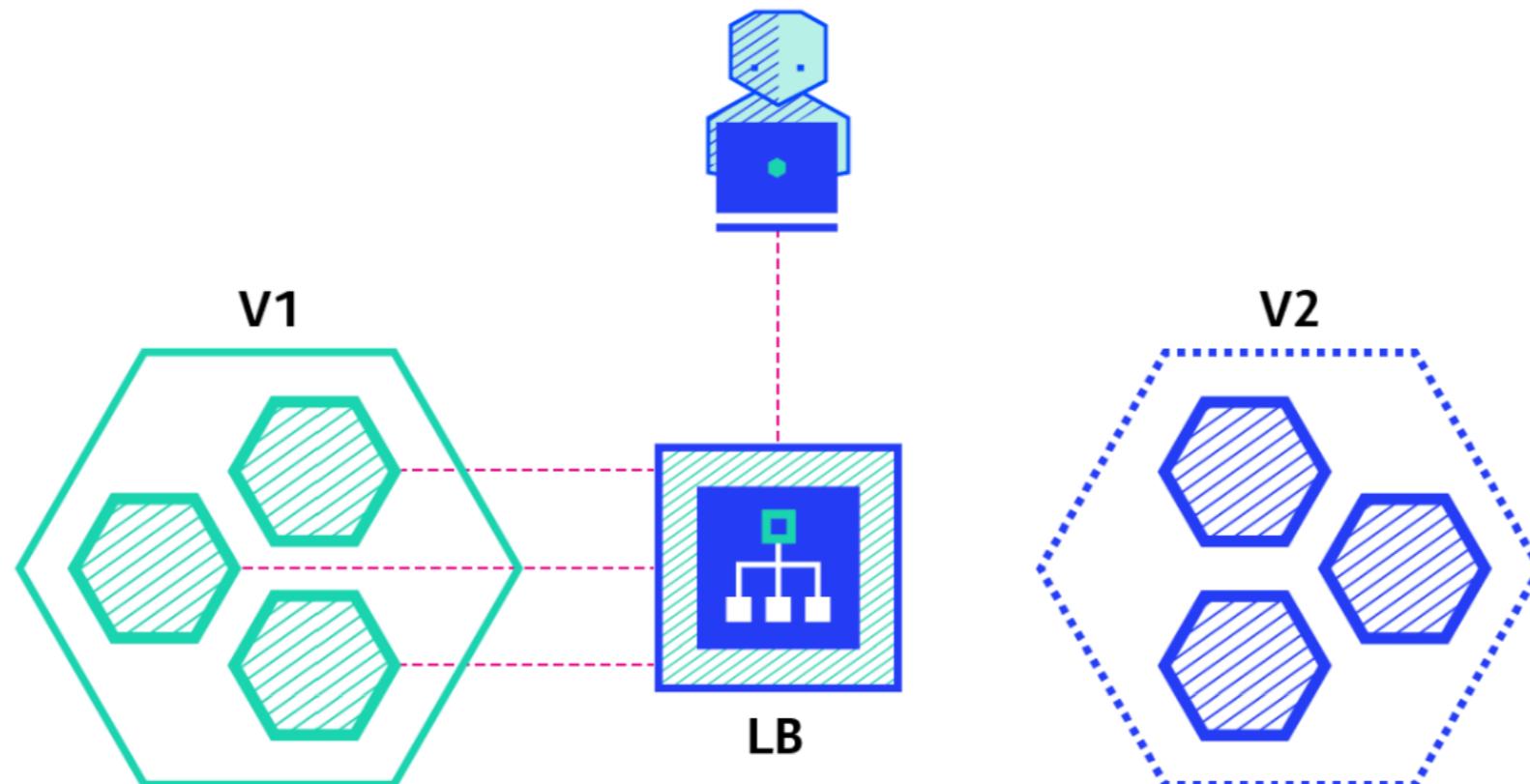
2. Ramped

Slow roll out by replace instance one-by-one

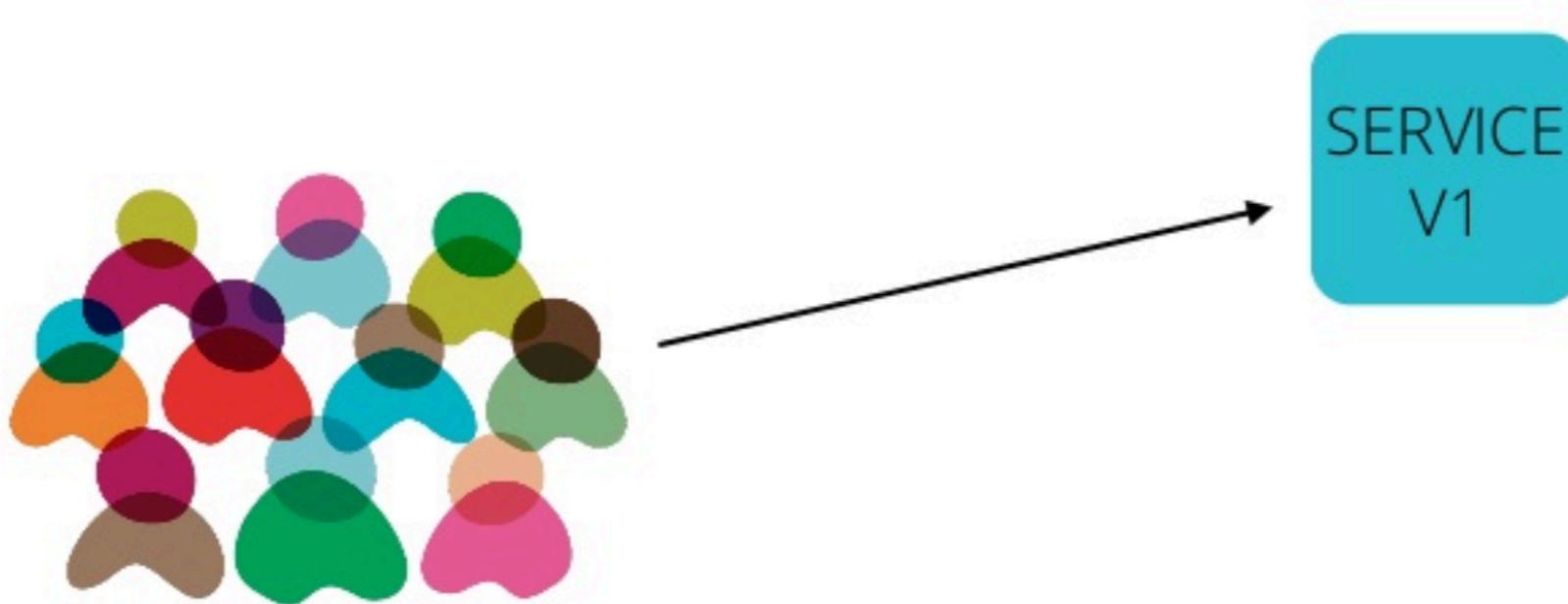


3. Blue/Green

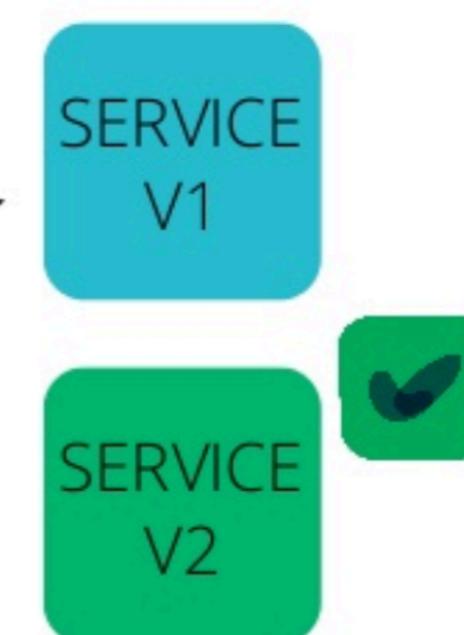
Current version is called **Blue**
New version is called **Green**



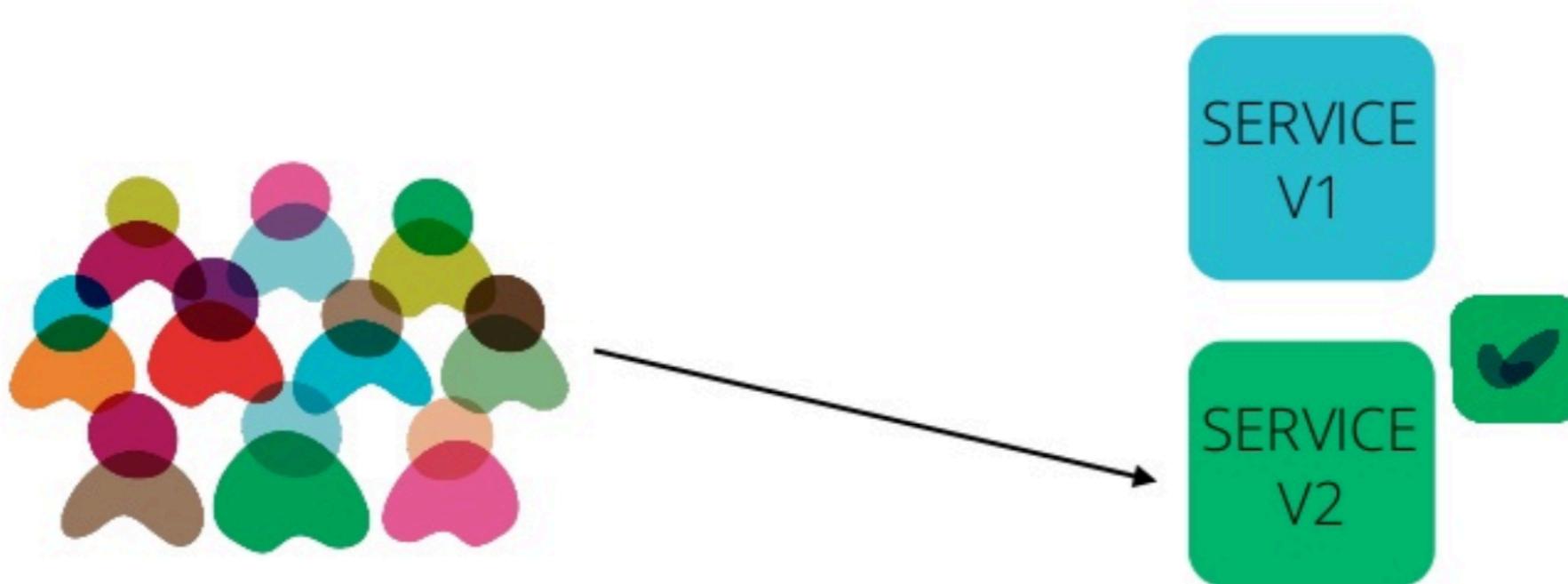
Blue Green Deployment



Blue Green Deployment



Blue Green Deployment

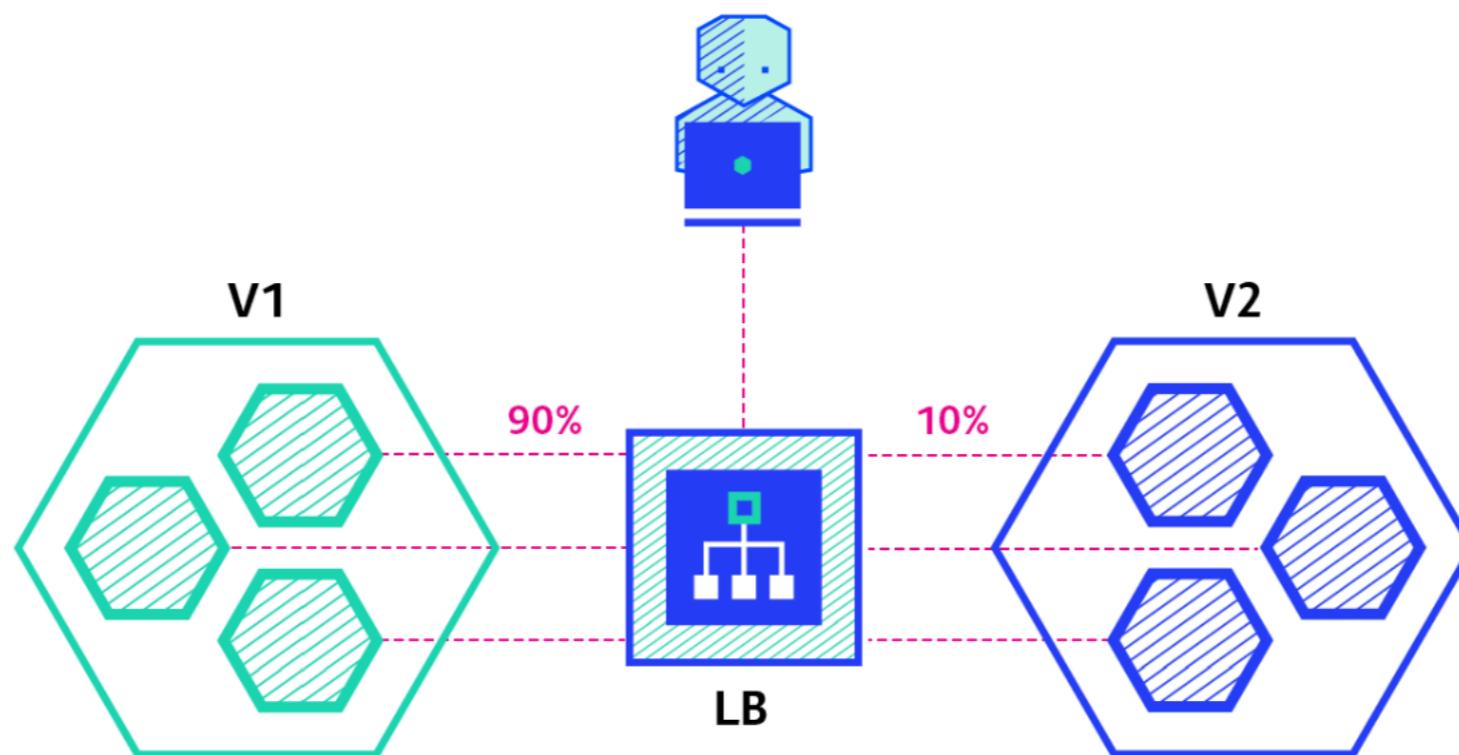


4. Canary

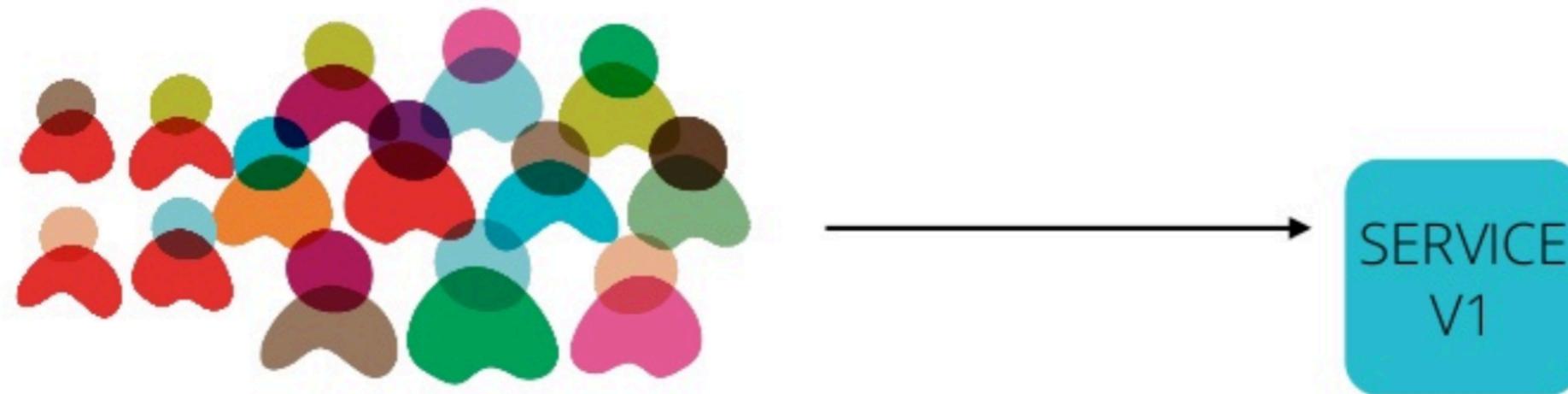
Shift production traffic from version A to B

Traffic is split based-on weight

Use when tests are lacking/not reliable and less confident in system



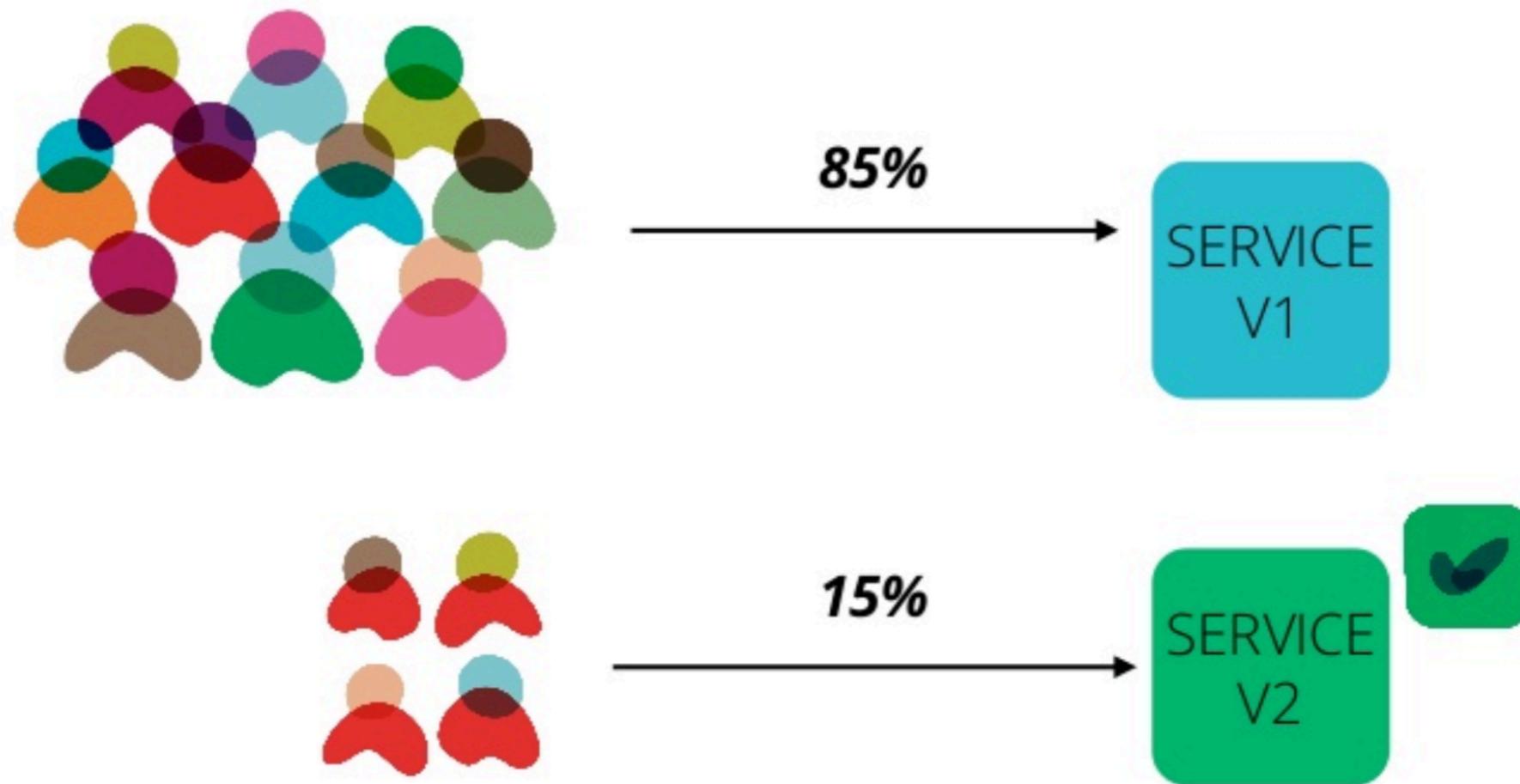
Canary Release



Canary Release

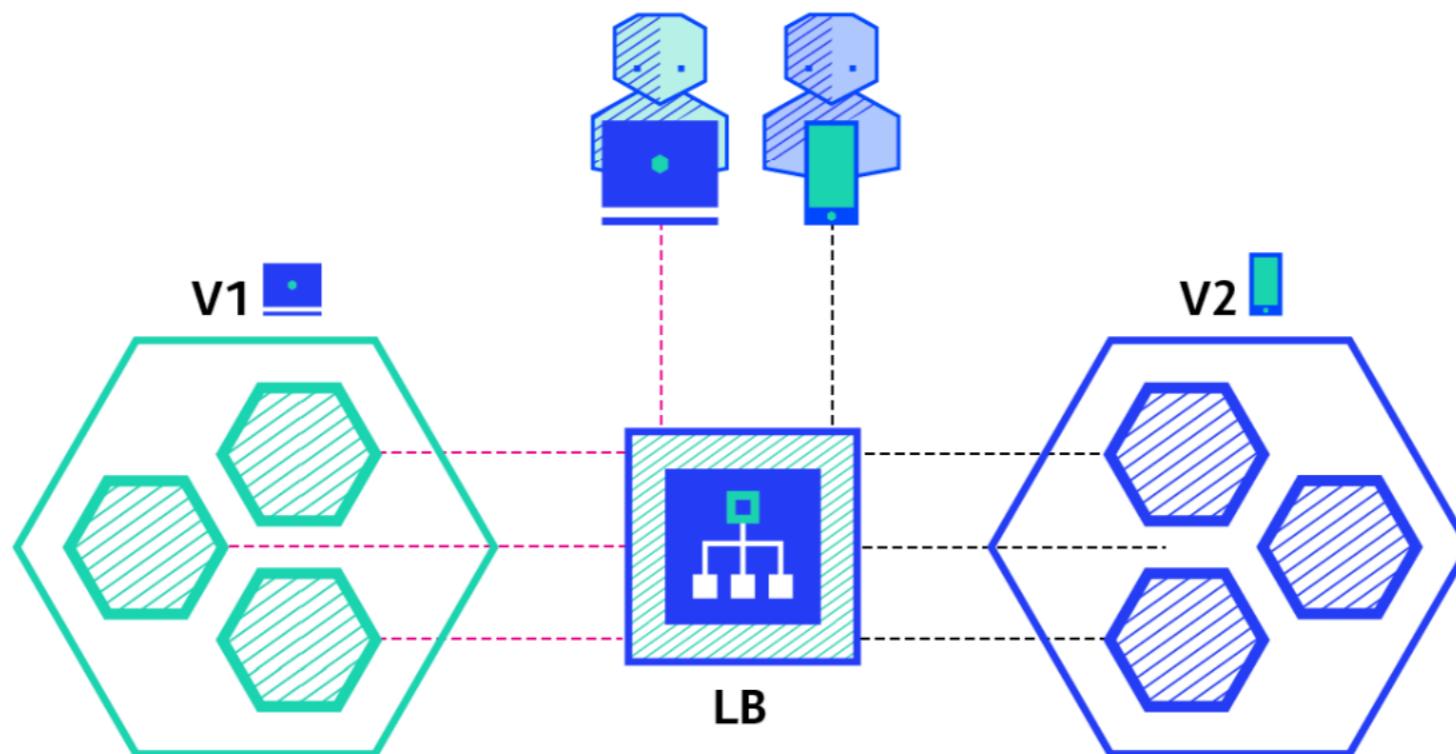


Canary Release



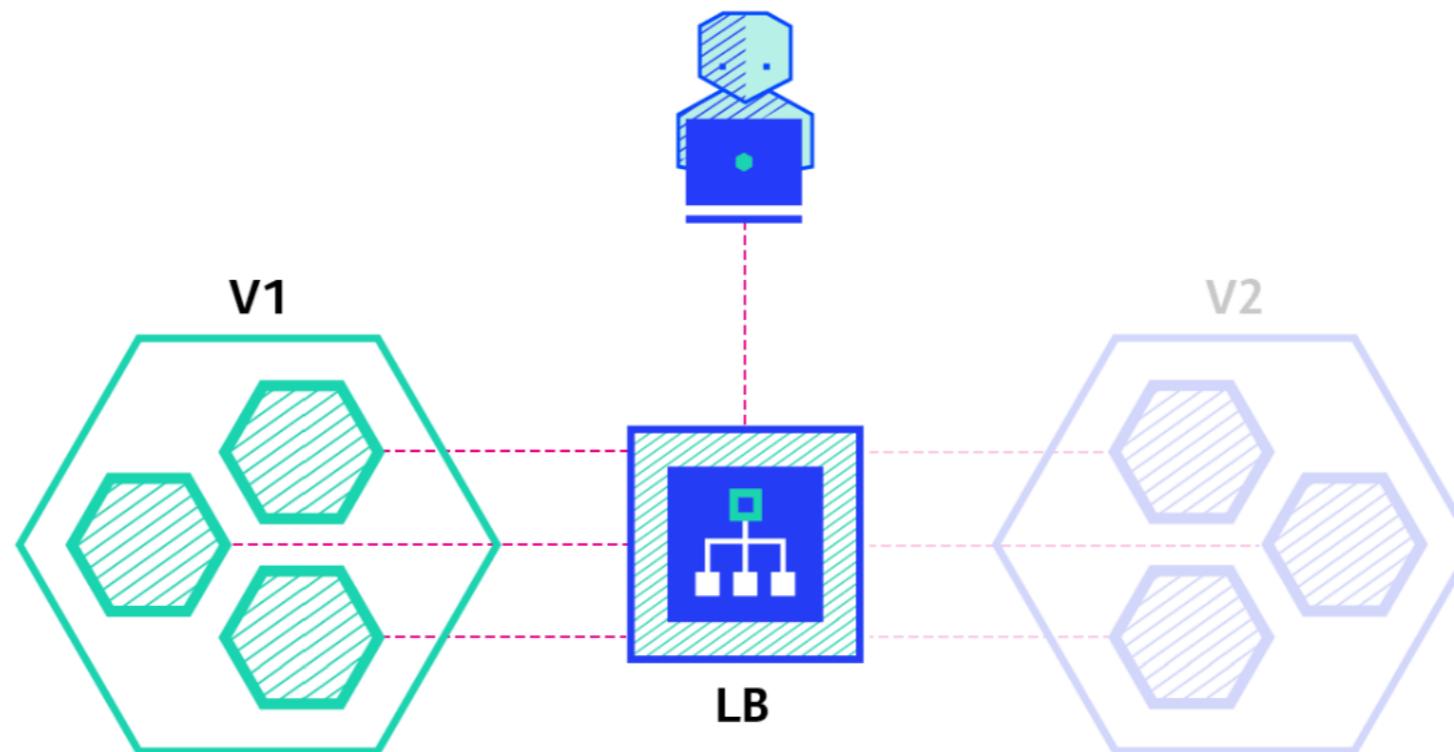
5. A/B testing

Routing the subset of users to new services under the specific condition



6. Shadow

Release version B alongside version A
Send request's A to B without production impact



DEPLOYMENT STRATEGIES

When it comes to production, a ramped or blue/green deployment is usually a good fit, but proper testing of the new platform is necessary.

Blue/green and shadow strategies have more impact on the budget as it requires double resource capacity. If the application lacks in tests or if there is little confidence about the impact/stability of the software, then a canary, a/b testing or shadow release can be used.

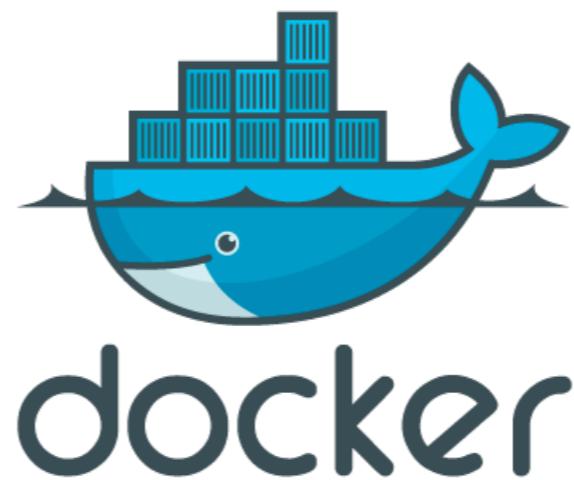
If your business requires testing of a new feature amongst a specific pool of users that can be filtered depending on some parameters like geolocation, language, operating system or browser features, then you may want to use the a/b testing technique.



Strategy	ZERO DOWNTIME	REAL TRAFFIC TESTING	TARGETED USERS	CLOUD COST	ROLLBACK DURATION	NEGATIVE IMPACT ON USER	COMPLEXITY OF SETUP
RECREATE version A is terminated then version B is rolled out	✗	✗	✗	■ ■ ■	■ ■ ■	■ ■ ■	□ □ □
RAMPED version B is slowly rolled out and replacing version A	✓	✗	✗	■ ■ ■	■ ■ ■	■ □ □	■ □ □
BLUE/GREEN version B is released alongside version A, then the traffic is switched to version B	✓	✗	✗	■ ■ ■	□ □ □	■ ■ □	■ ■ □
CANARY version B is released to a subset of users, then proceed to a full rollout	✓	✓	✗	■ ■ ■	■ □ □	■ □ □	■ ■ □
A/B TESTING version B is released to a subset of users under specific condition	✓	✓	✓	■ ■ ■	■ □ □	■ □ □	■ ■ ■
SHADOW version B receives real world traffic alongside version A and doesn't impact the response	✓	✓	✗	■ ■ ■	□ □ □	□ □ □	■ ■ ■



Deployment workshop with



<https://github.com/up1/workshop-develop-microservices-2023>



Design CI/CD process

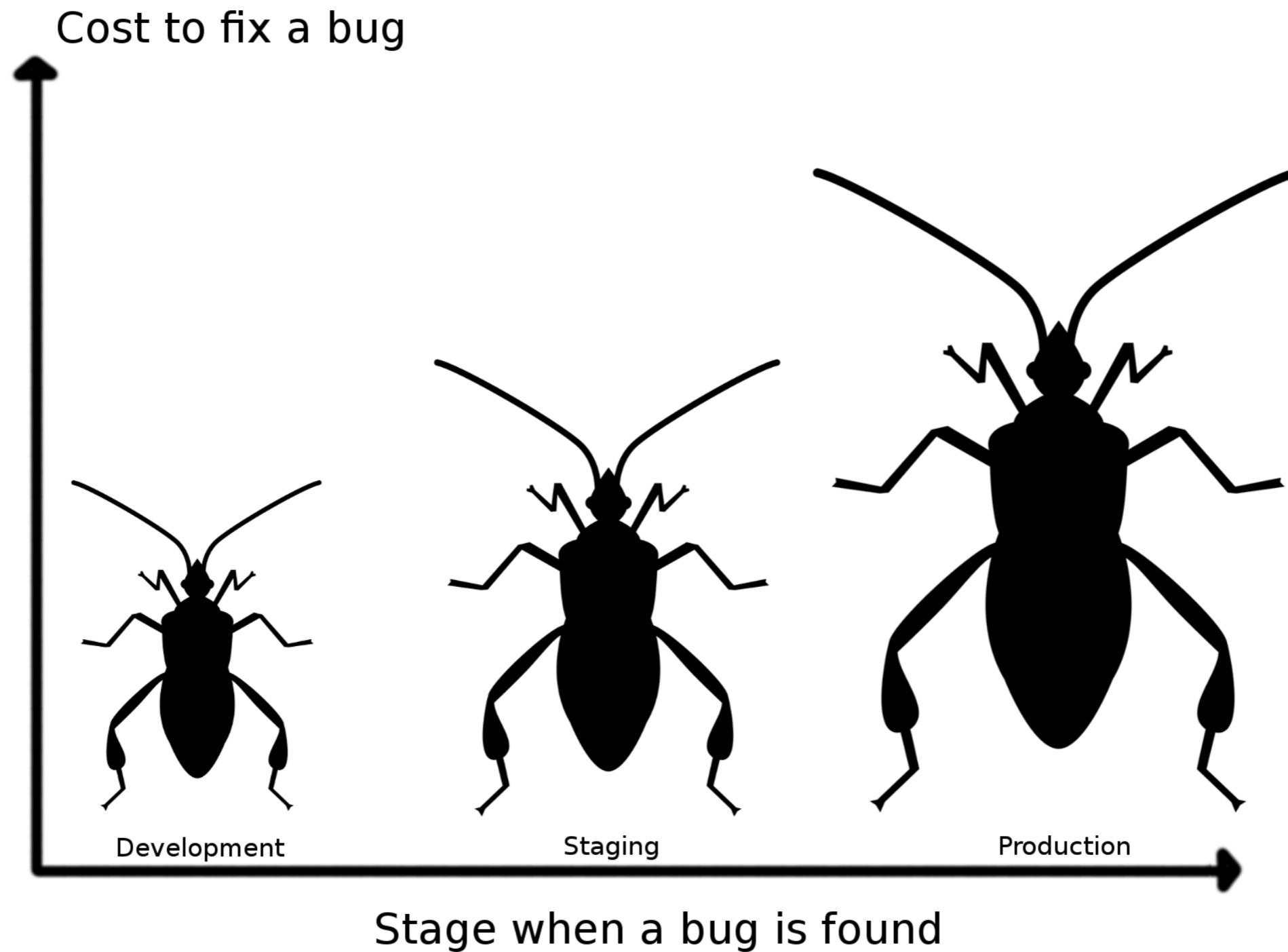


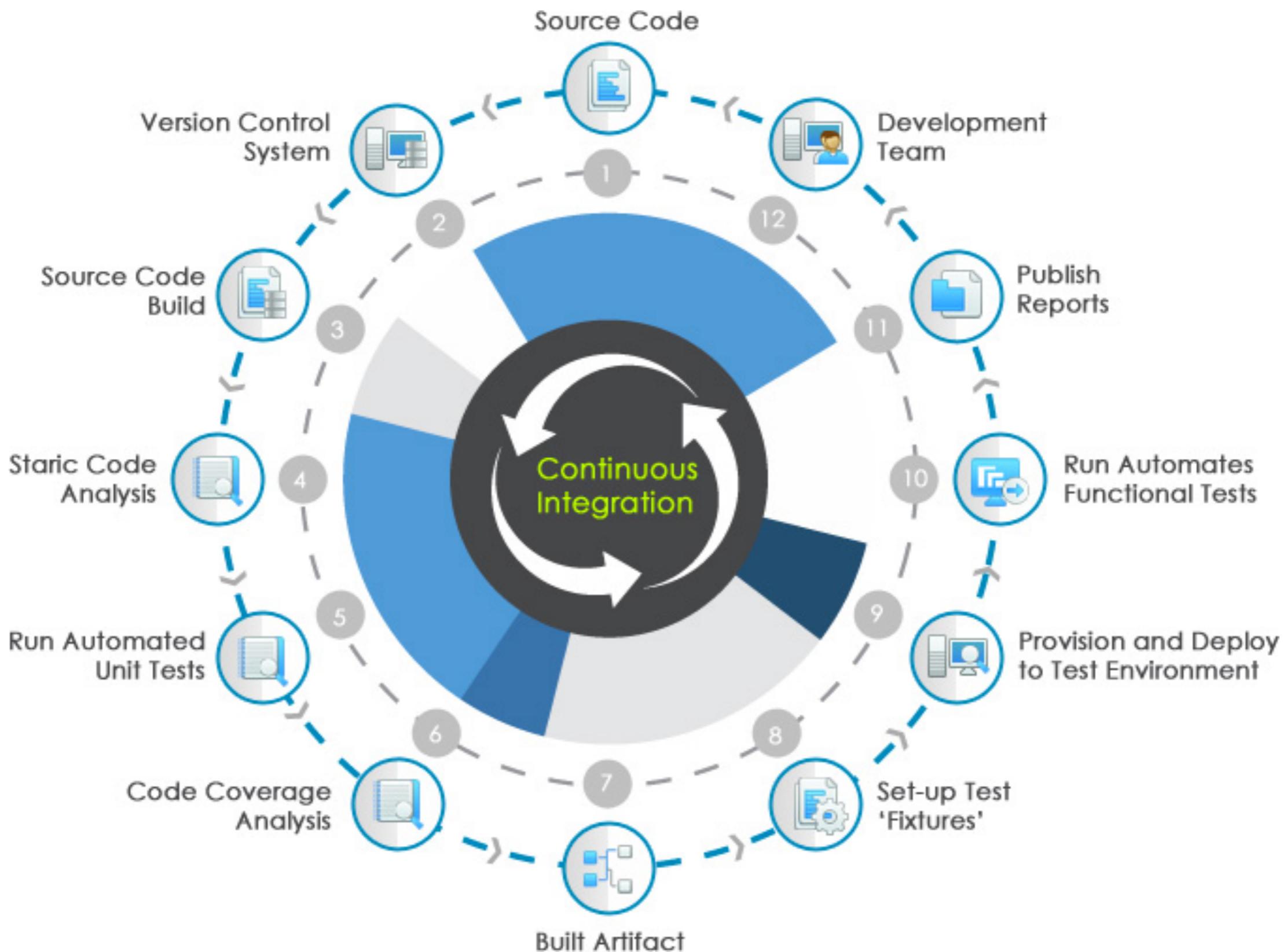
The cost of integration

1. Merging the code
2. Duplicate changes
3. Test again again !!
4. Fixing bugs
5. Impact on stability



The cost of integration







Jenkins



Bamboo

CI is about **what people do**
not about **what tools they use**



Visual Studio



Team Foundation Server

Hudson



Microservices

© 2022 - 2023 Siam Chamnankit Company Limited. All rights reserved.

Continuous Integration

Discipline to integrate frequently



Continuous Integration

Strive to make **small change**

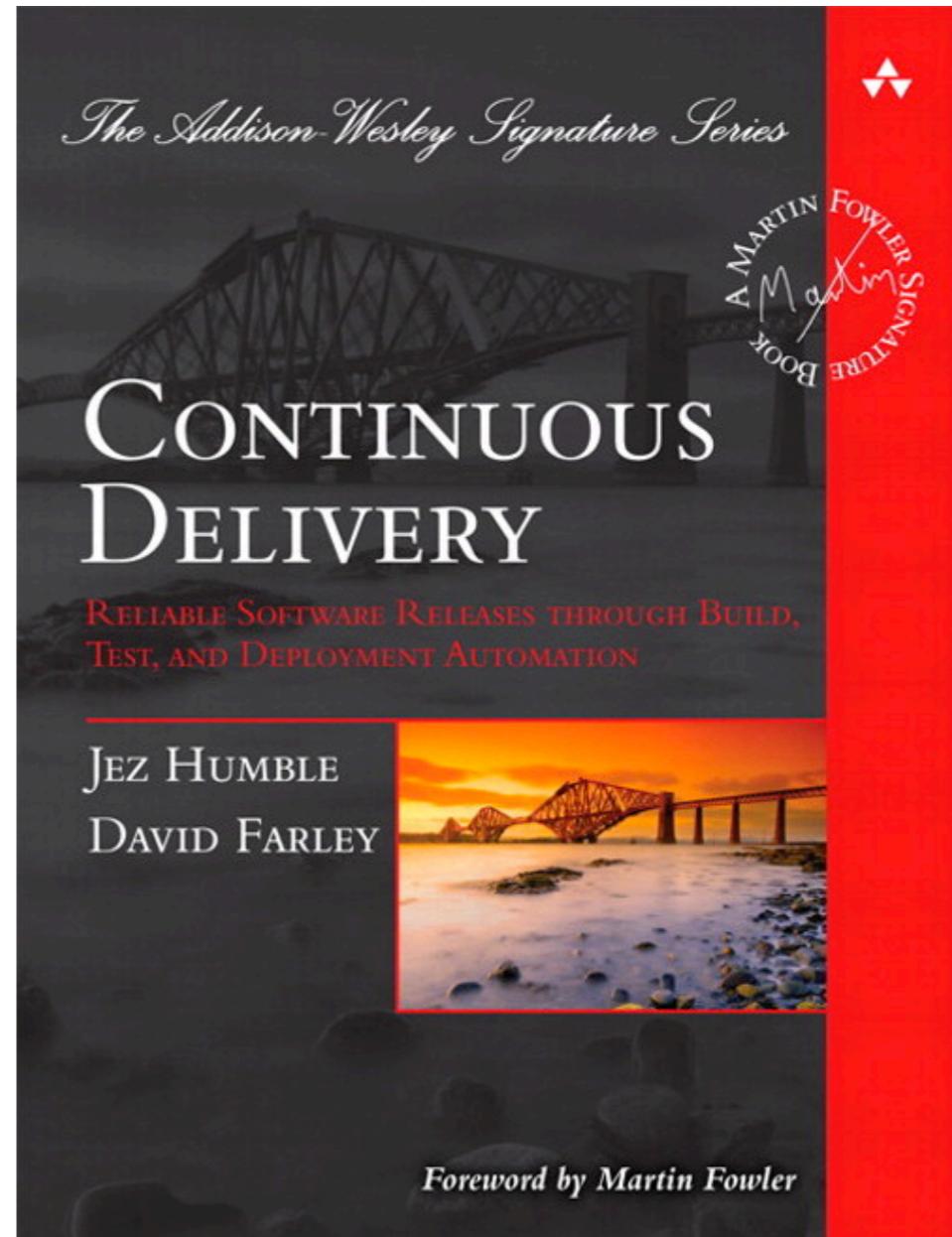
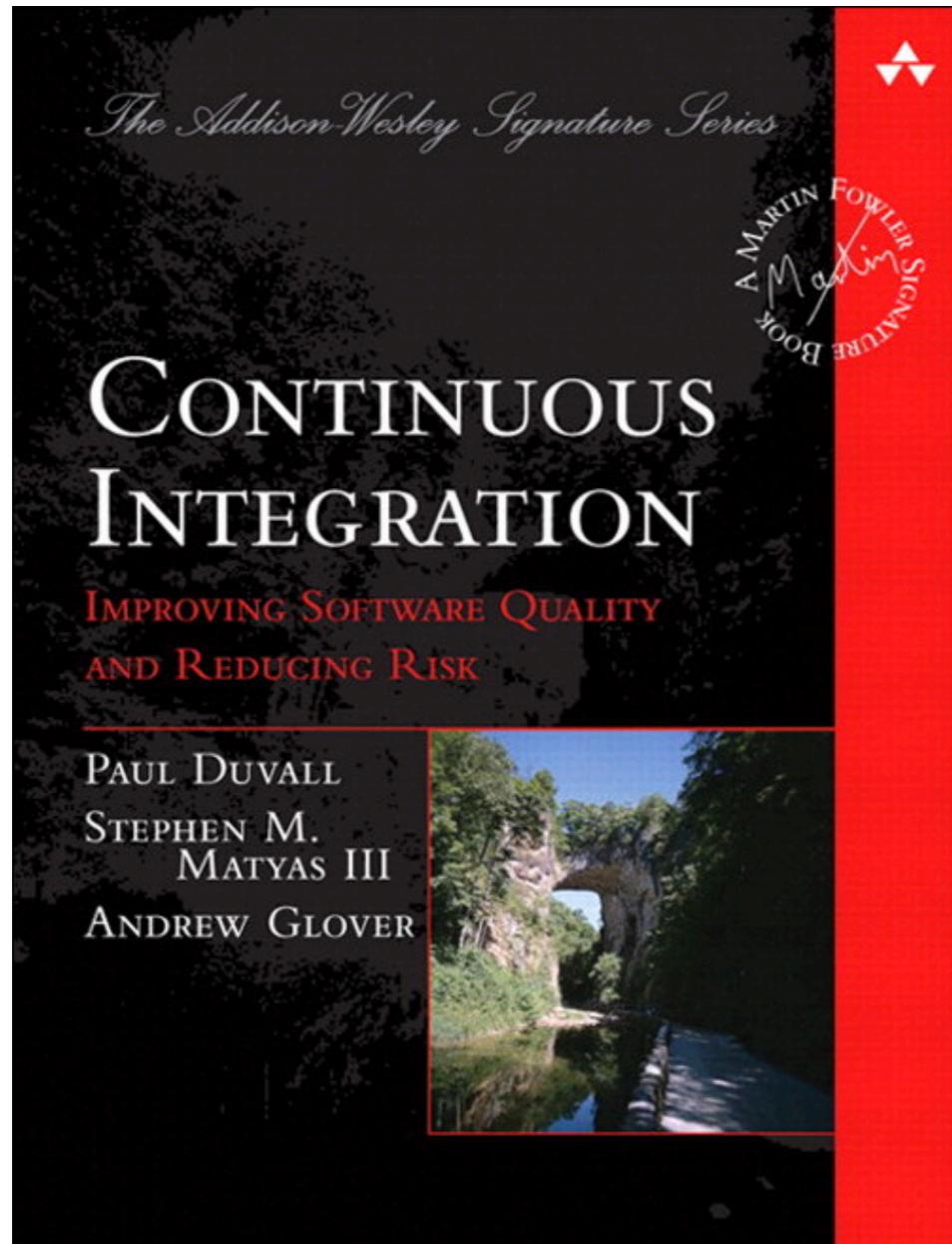


Continuous Integration

Strive for **fast feedback**



Improve quality and reduce risk



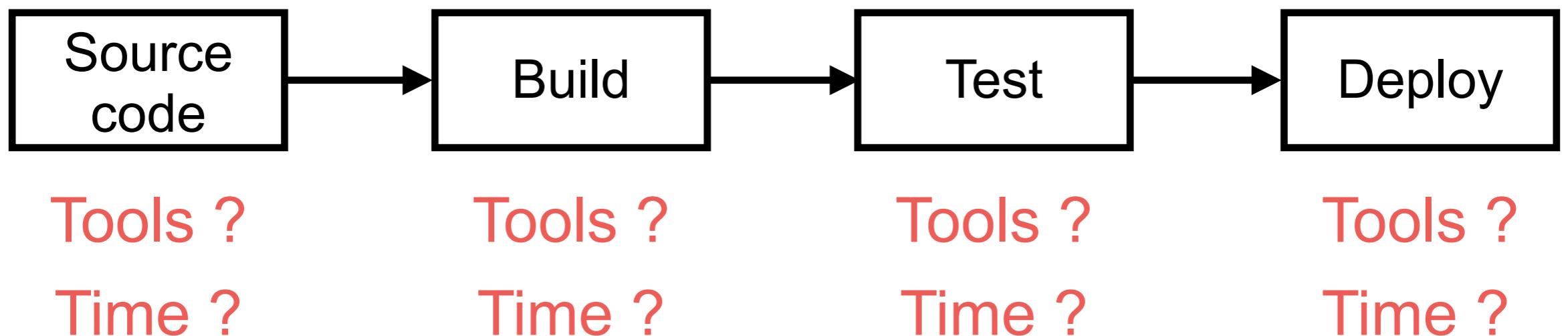
Design pipeline

Path to Production !!



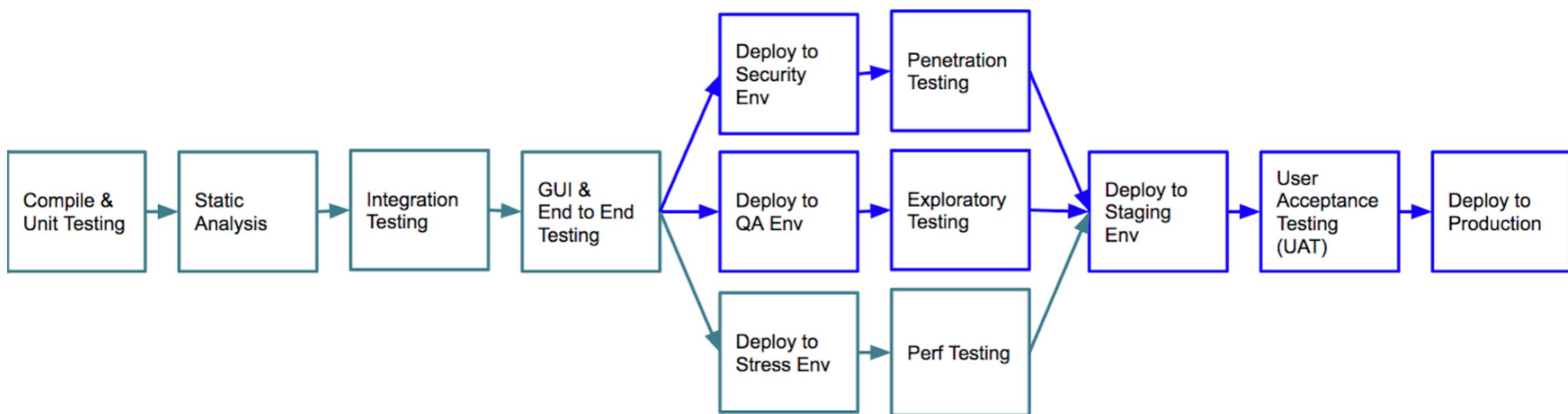
Pipeline of services

Fast feedback and high quality
Automation process



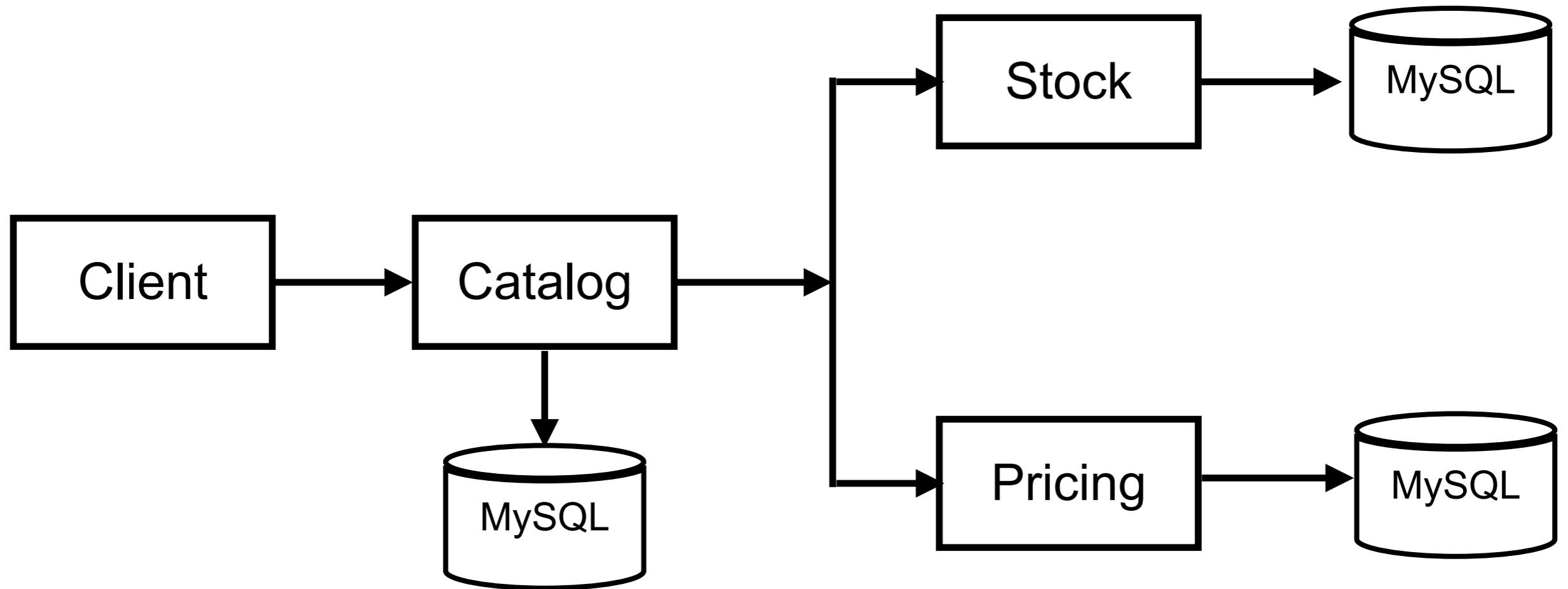
Path to Production ?

Improve your deployment/delivery pipeline





How to design pipeline ?



CI/CD pipeline



Create pipeline workshop



Jenkins

<https://github.com/up1/workshop-develop-microservices-2023>



Q/A

