

Performance Testing





Performance testing

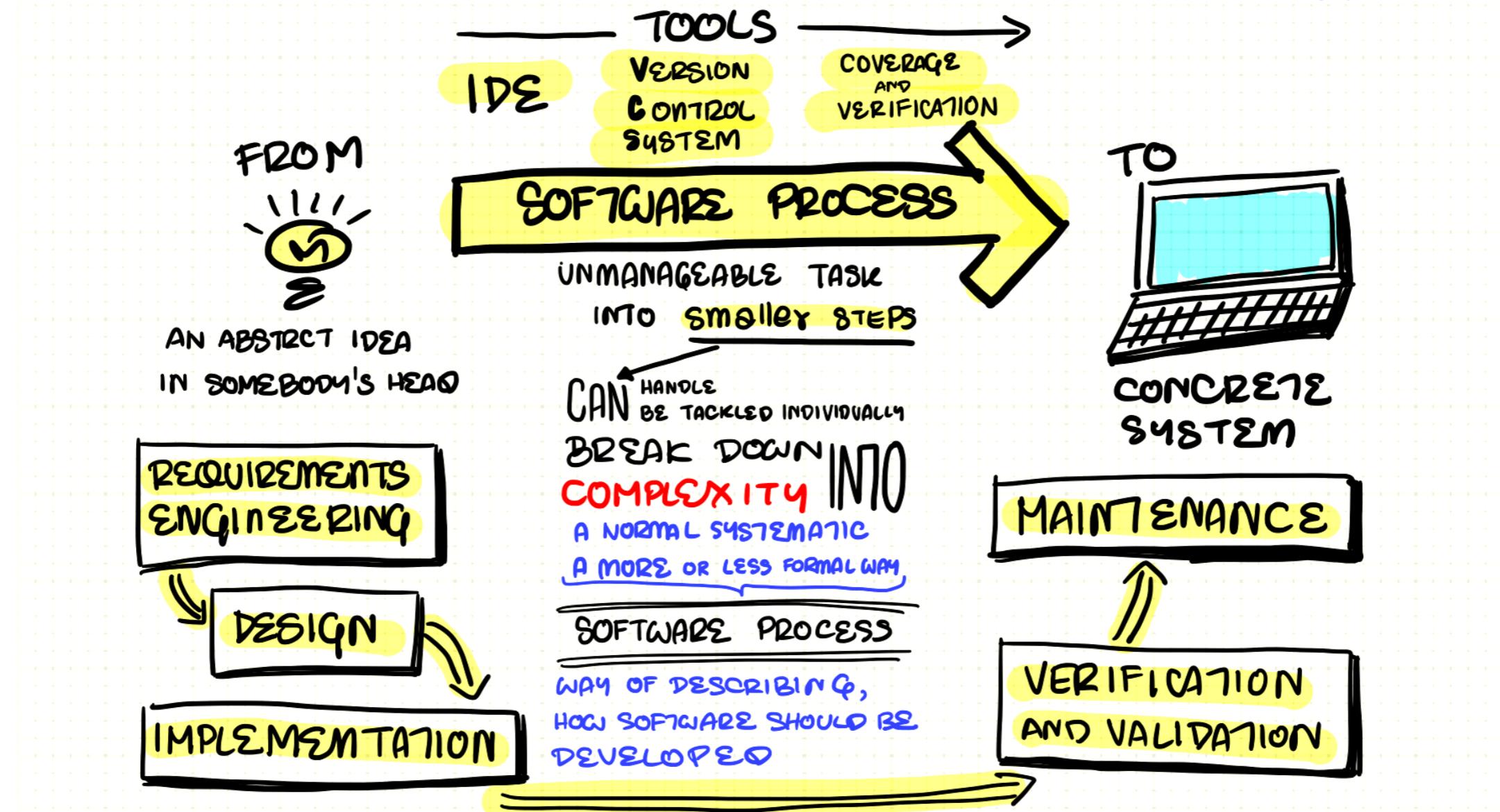


Software Development Life Cycle



SOFTWARE DEVELOPMENT

on 11th
029 RACHIZE
APR 13, 2018



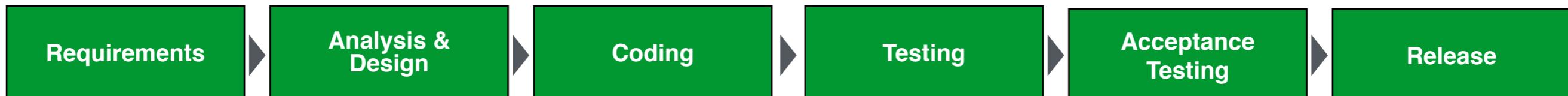
FROM : SOFTWARE DEVELOPMENT PROCESS - UDACITY



Silo with poor communication and collaboration

Requirements change

Don't understand business domain and requirements



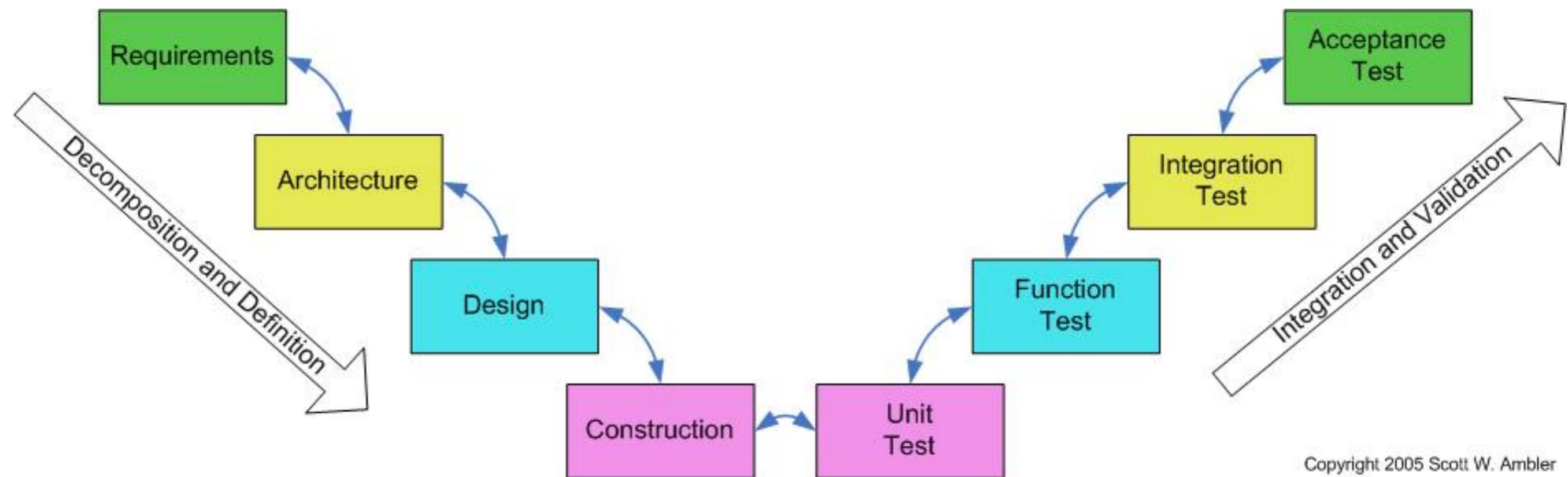
Testing at the end

Tester don't involved early

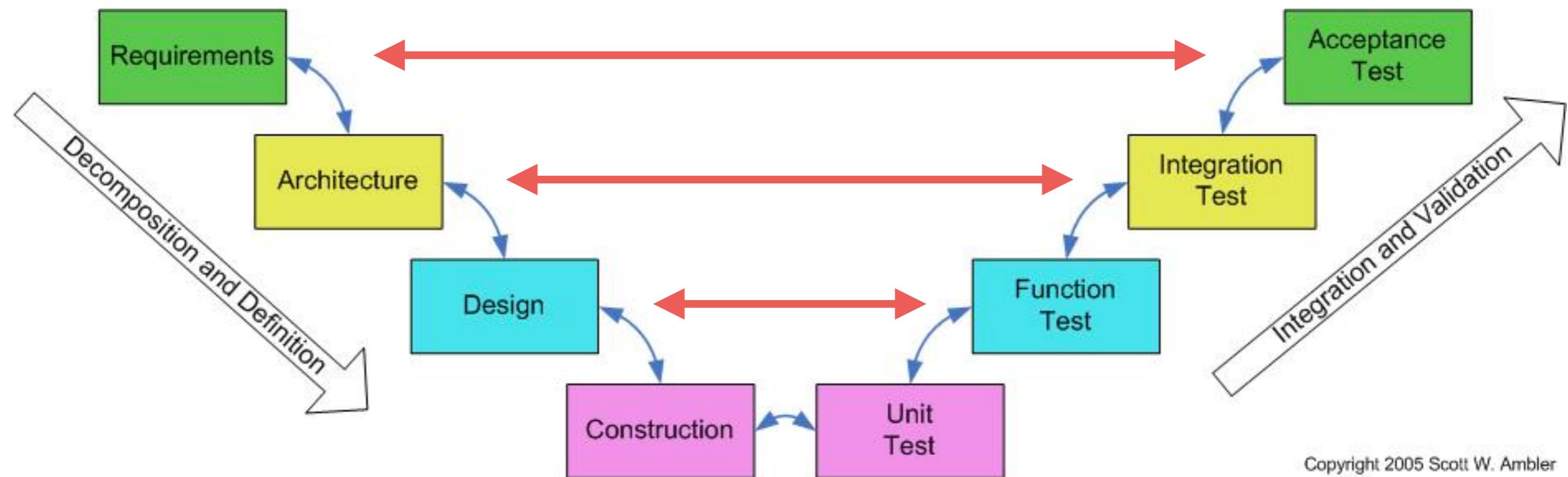
Only tester take responsibility



V Model



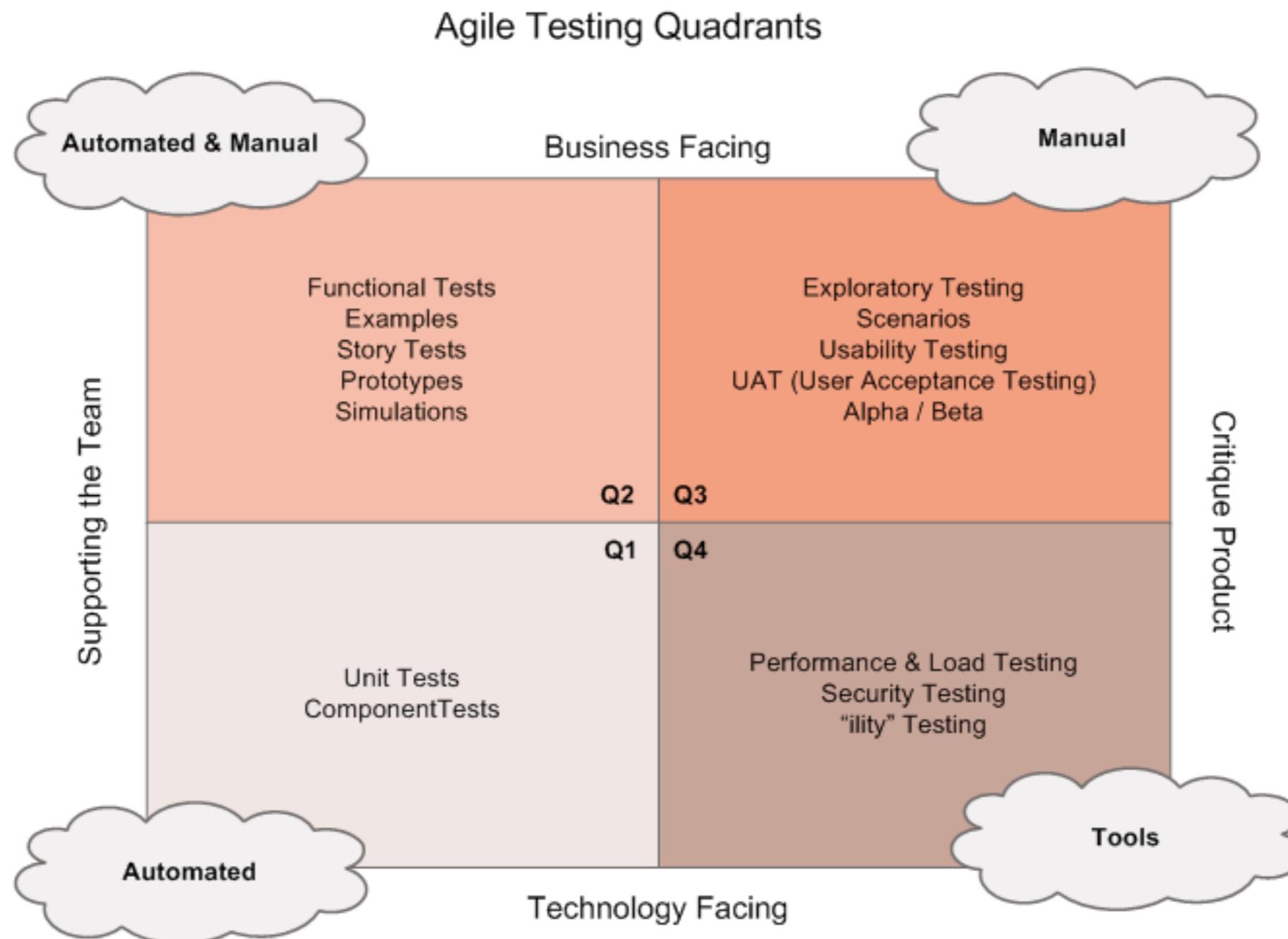
V Model



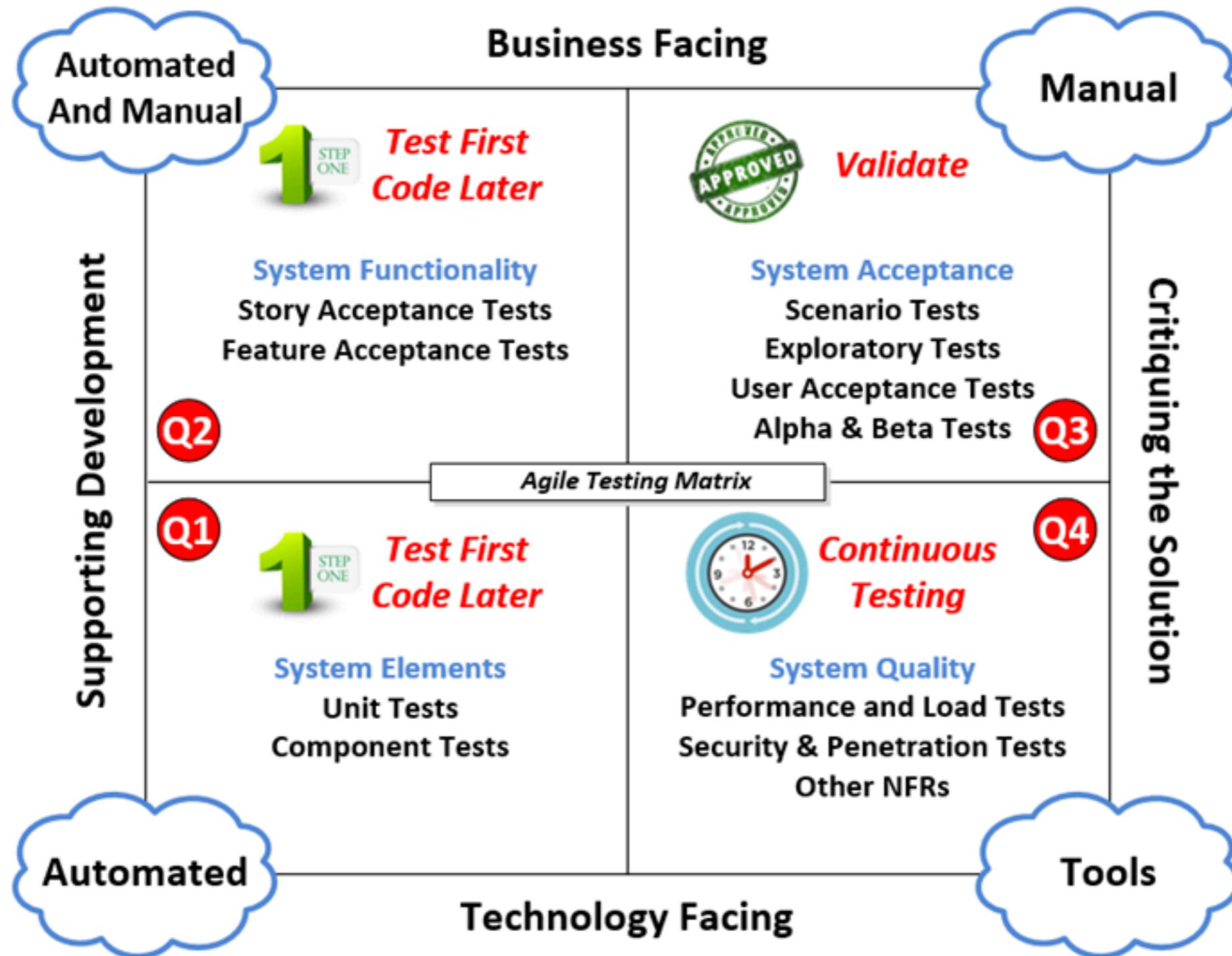
Types of testing



Agile Testing Quadrant



Agile Testing Quadrant



When performance testing ?



At the end !!

Before release !!



Simplify testing and start early



When performance testing ?

Design phase

Development phase

Deployment phase



Why do performance testing ?



Questions ?



Questions ?

How many end users will the application need
to support at release ?

After 6 months, 12 months, 2 years ?



Questions ?

Where will these users be located, and how will they connect to the application?



Questions ?

How many of these users will be concurrent at release?

After 6 months, 12 months, 2 years ?



Lead to more questions



Questions ?

How many and what specification of servers will I need for each application tier ?



Questions ?

Where should these servers be hosted ?



Questions ?

What sort of network infrastructure do I need to provide ?



Think about Capacity and scalability



Impact on Performance and Availability



Non Functional Requirements (NFRs)



Essential for NFRs (1)

Designing an appropriate performance test environment

Setting realistic and appropriate performance targets

Identifying and scripting the business-critical use cases



Essential for NFRs (2)

Providing test data

Creating a load model

Ensuring accurate performance test design

Identifying the KPIs

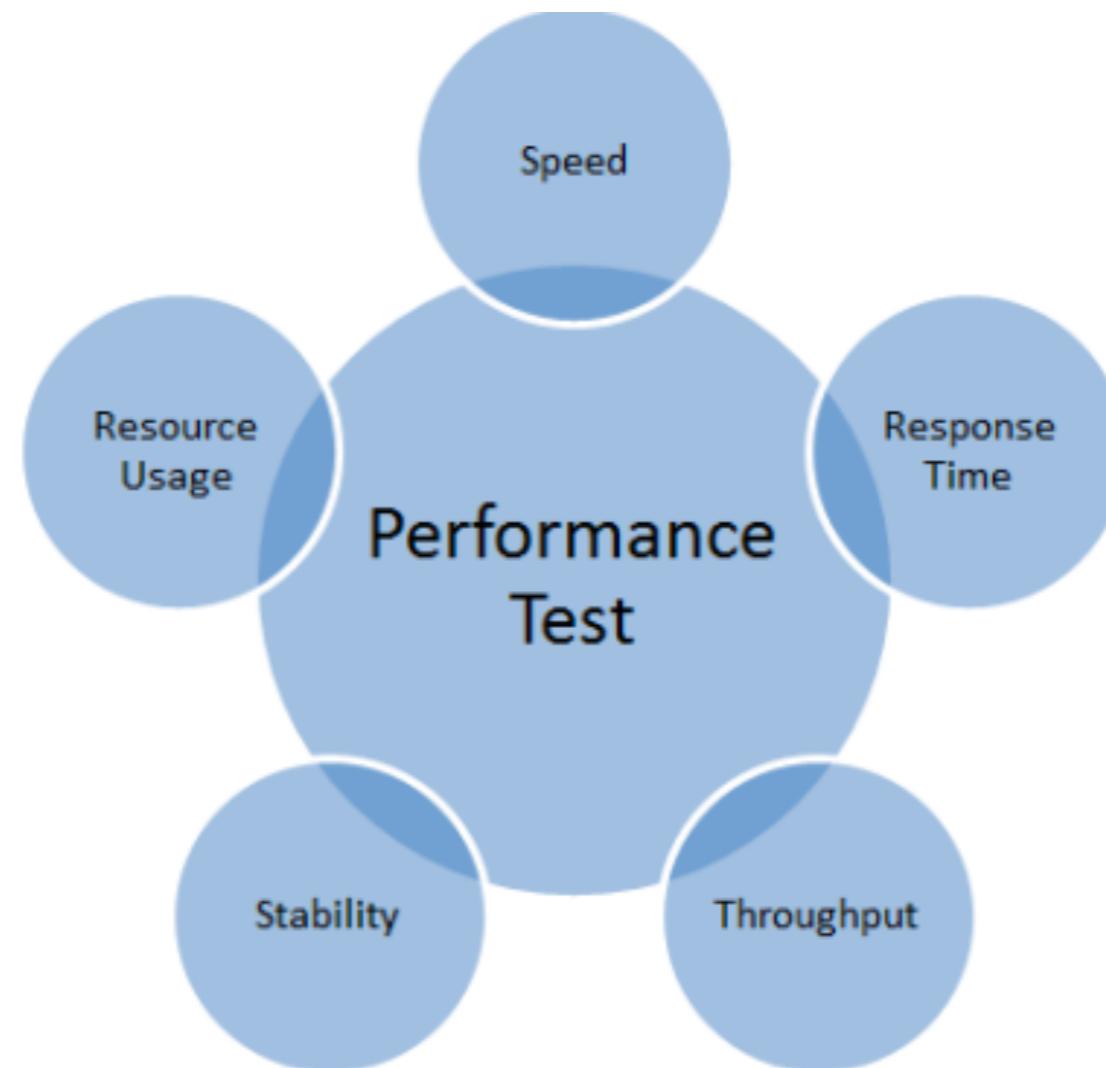


Fast is better



Why do performance testing ?

Stability
Responsiveness



Stability ?

No response

Incomplete response

Inconsistent experience

User may never come back !!



Responsiveness ?

User get frustrated
Negative feelings about system
Users may leave



Summary

Identify problems early
Reduce development cycle
Produce better quality system
Produce better scalable system
Prevent revenue and creditability loss
Enable planning for the future



Summary

Ensure the system meets performance expectation

Expose more bugs (memory leak, CPU)



Cost and benefits of Performance testing



Costs ?

Test creation time
Resources to run tests
Users may leave



Benefits ?

Found issues early (not in production)
Impact of issues



Make sure app is ready

- High data presentation
- Poorly performing SQL
- Large number of app network round-trips
- Undetected app errors



Example of the bad SQL

Database and Connection Pool	Contribution	Executions
app@amazona-rp6g1da:easyTravelBusiness (Tomcat)	99.0%	15
select journey0_.id as id1_, journey0_.amount as amount1_, journey0_.description as descript3_1_, journey0_.destination_name as destinat8_1_, journey0_.fromDate as fromDate1_, journey0_.name as name1_, journey0_.tenant_name as tenant10_1_, journey0_.toDate as toDate1_ from easyTravelBusiness.journey ijourney0_ where ?=journey0_.destination_name and ? >= ijourney0_.toDate and (normalize_location(? , ?) is not null)	52.0%	14,127
select location0_.name as name6_0_ from Location location0_ where tenant0_.name = ?	31.0%	8,418
select user0_.name as name3_0_, user0_.email as email3_0_ from User user0_ where tenant0_.name = ?		1,166
call verify_location(?)		1,138
select tenant0_.name as name0_0_, tenant0_.description as description0_0_ from Tenant tenant0_ where tenant0_.name = ?		734
select location0_.name as name6_ from Location location0_ where tenant0_.name = ?		690
update LoginUser set email=?, fullName=?, lastLogin=? where id=?		324
insert into Booking (bookingDate, journey_id, user_name, amount) values (?, ?, ?, ?)		21
select sum(journey1_.amount) as col_0_0_ from Booking booking0_ inner join Journey journey1_ on booking0_.journey_id = journey1_.id where journey1_.tenant_name = ?		20
select location2_.name as col_0_0_, count(booking0_.id) as col_0_1_ from Booking booking0_ inner join Journey journey1_ on booking0_.journey_id = journey1_.id inner join Location location2_ on journey1_.location_id = location2_.id where journey1_.tenant_name = ? group by location2_.name		20
select location2_.name as col_0_0_, count(booking0_.id) as col_0_1_ from Booking booking0_ inner join Journey journey1_ on booking0_.journey_id = journey1_.id inner join Location location2_ on journey1_.location_id = location2_.id where journey1_.tenant_name = ? group by location2_.name		20
select journey0_.id as id1_ , journey0_.amount as amount1_ , journey0_.description as descript3_1_ , journey0_.destination_name as destinat8_1_ , journey0_.fromDate as fromDate1_ , journey0_.name as name1_ , journey0_.tenant_name as tenant10_1_ , journey0_.toDate as toDate1_ from easyTravelBusiness.journey ijourney0_ where ?=journey0_.destination_name and ? >= ijourney0_.toDate and (normalize_location(? , ?) is not null)		20



Must allocate enough time to performance test



Performance test time ?

Prepare test environment

Create load injectors

Identify and script of use cases

Identify and create test data

Unexpected problems



What Performance testing ?



What ?

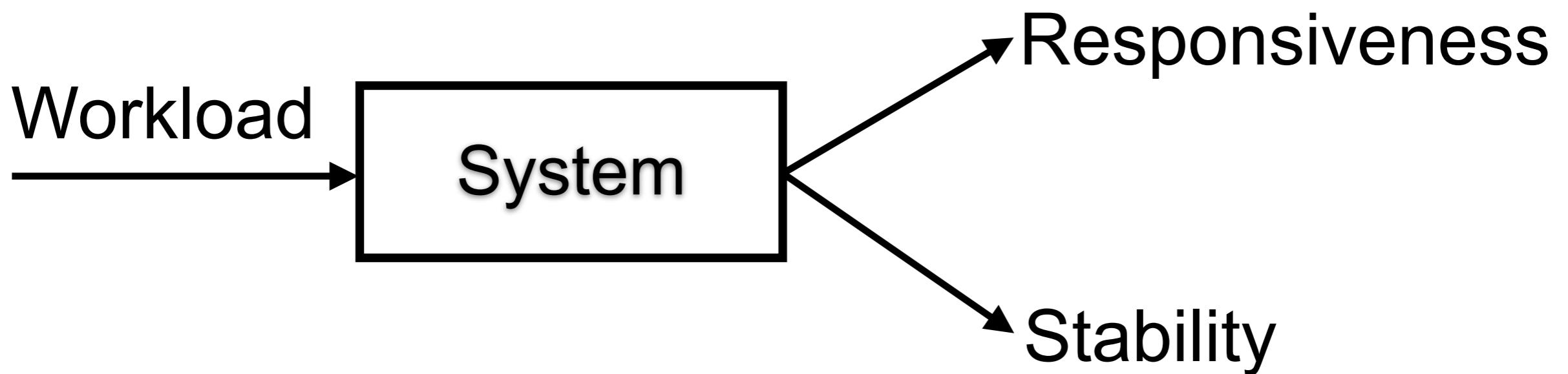
“Performance testing is in general a testing practice performed to determine how a system performs in terms of responsiveness and stability under a particular workload.”

— Wikipedia —



Performance testing

The way we load or stress the system



Responsiveness ?

How quickly an application **responds** when asked to do something



Stability ?

How consistent or reliable the application is



Response metrics

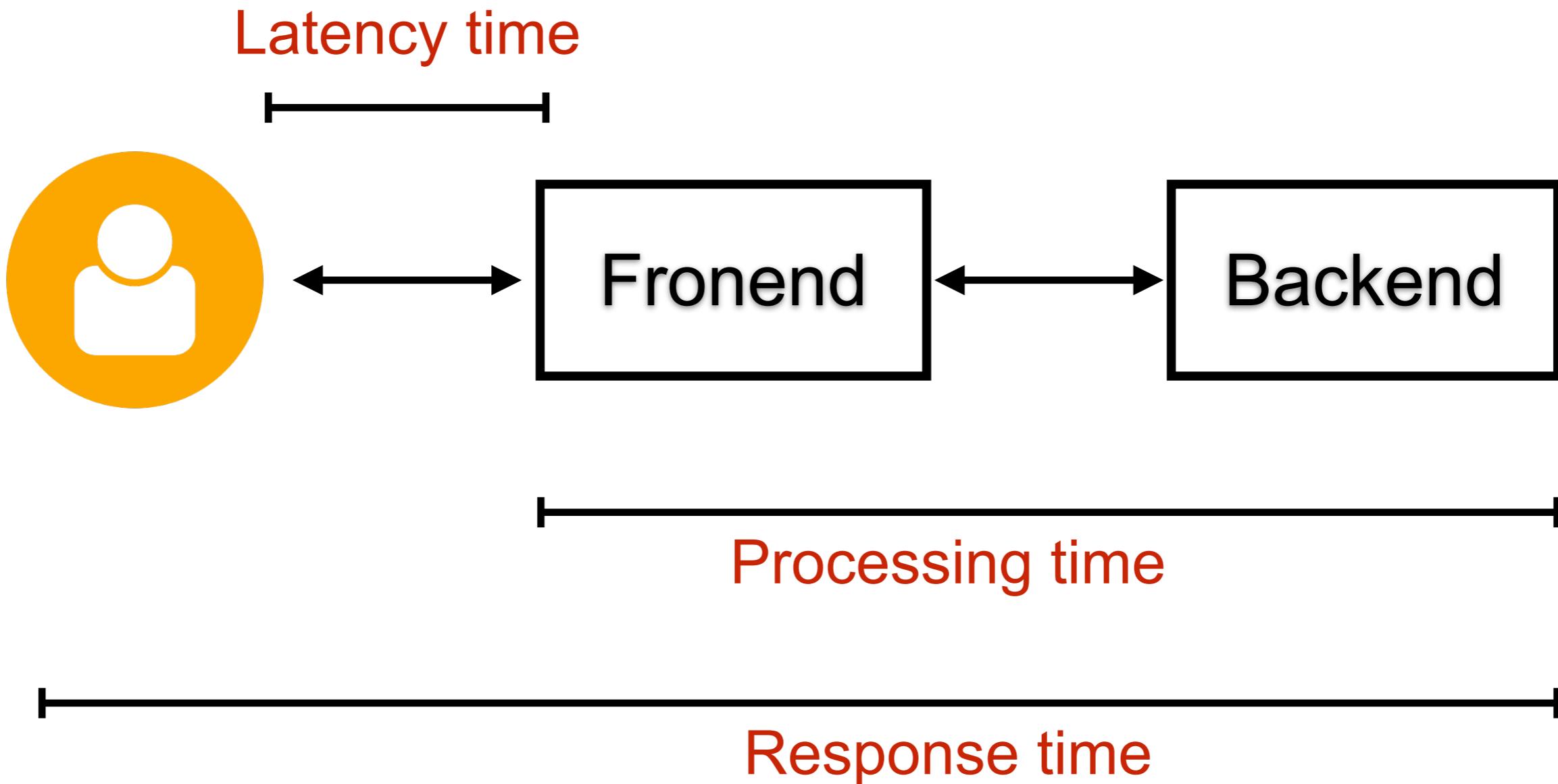
Average response time

Peak response time

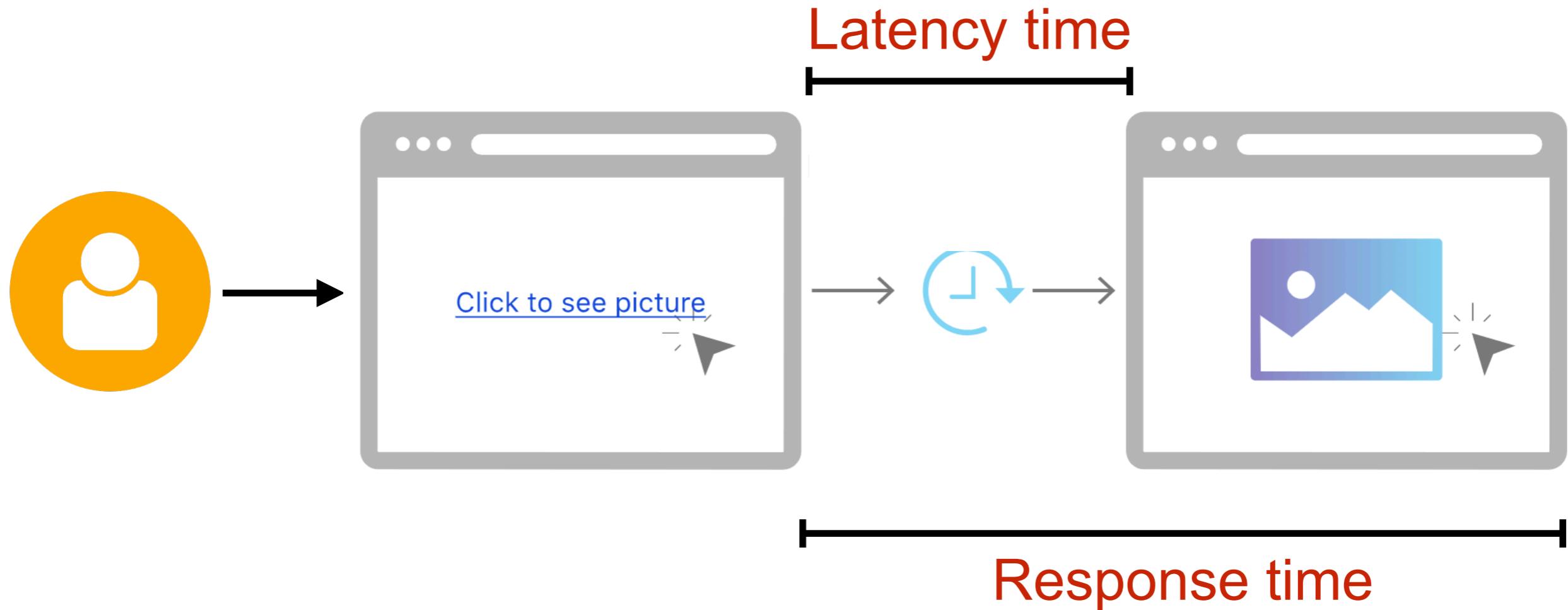
Error rate



Latency vs Response time



Latency vs Response time



Response time = Latency time + Processing time



Response time

Time to First byte

Time to Last byte



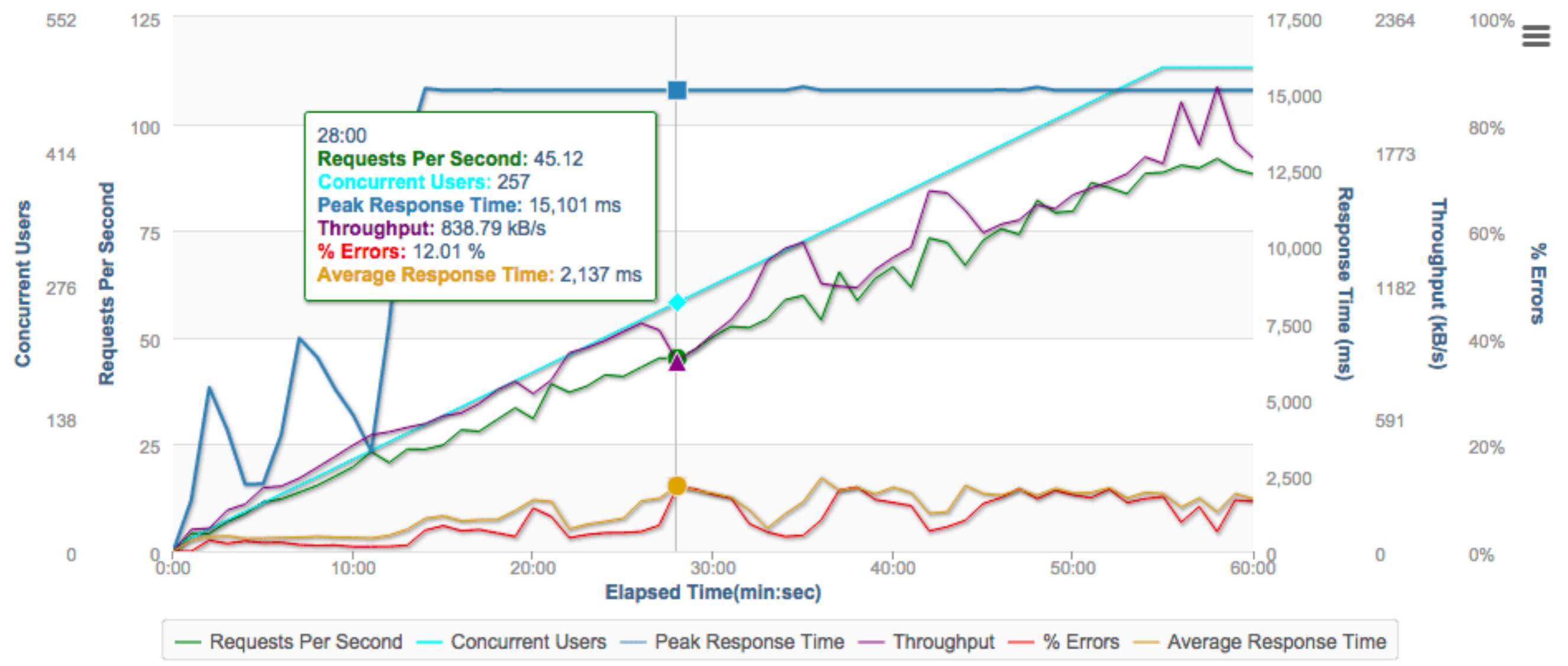
Volume measurements

Concurrent users
Requests per second
Throughput



Example

Summary - Example Test Run

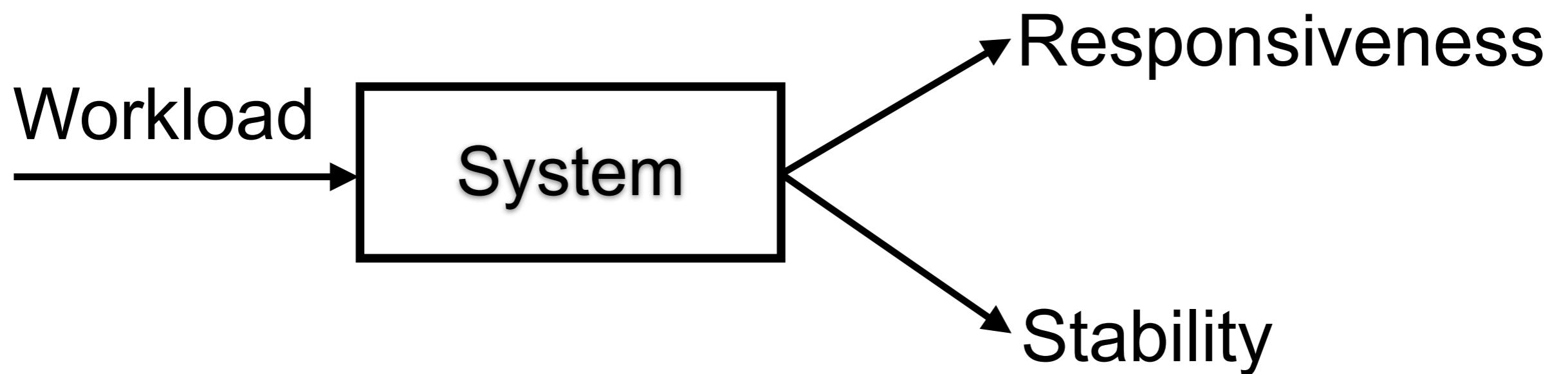


Workload



Workload ?

The way we load or stress the system



Workloads ?

Load testing

Stress testing

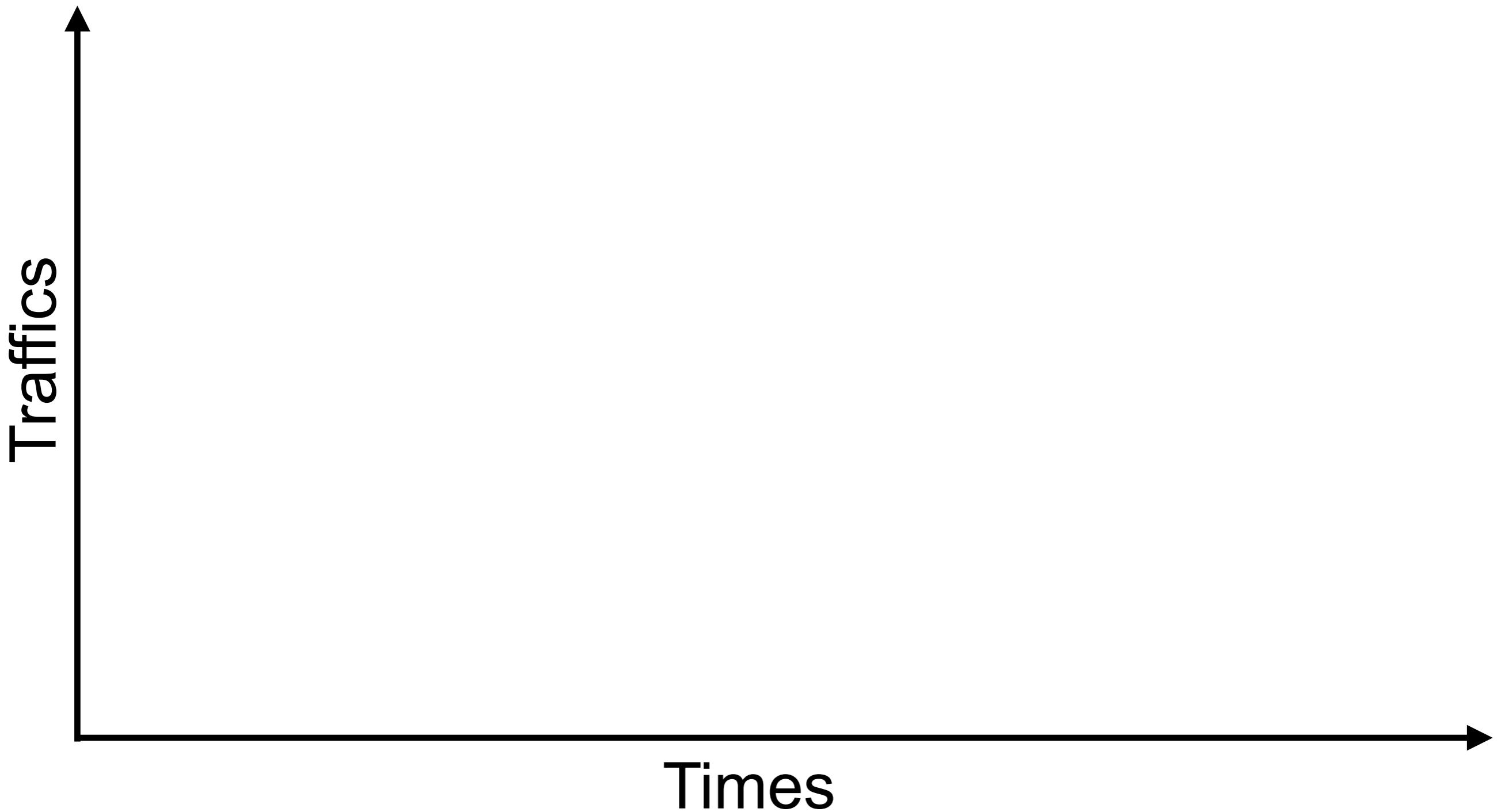
Endurance/soak testing

Spike testing

Scalability testing



Workloads ?



Load testing ?

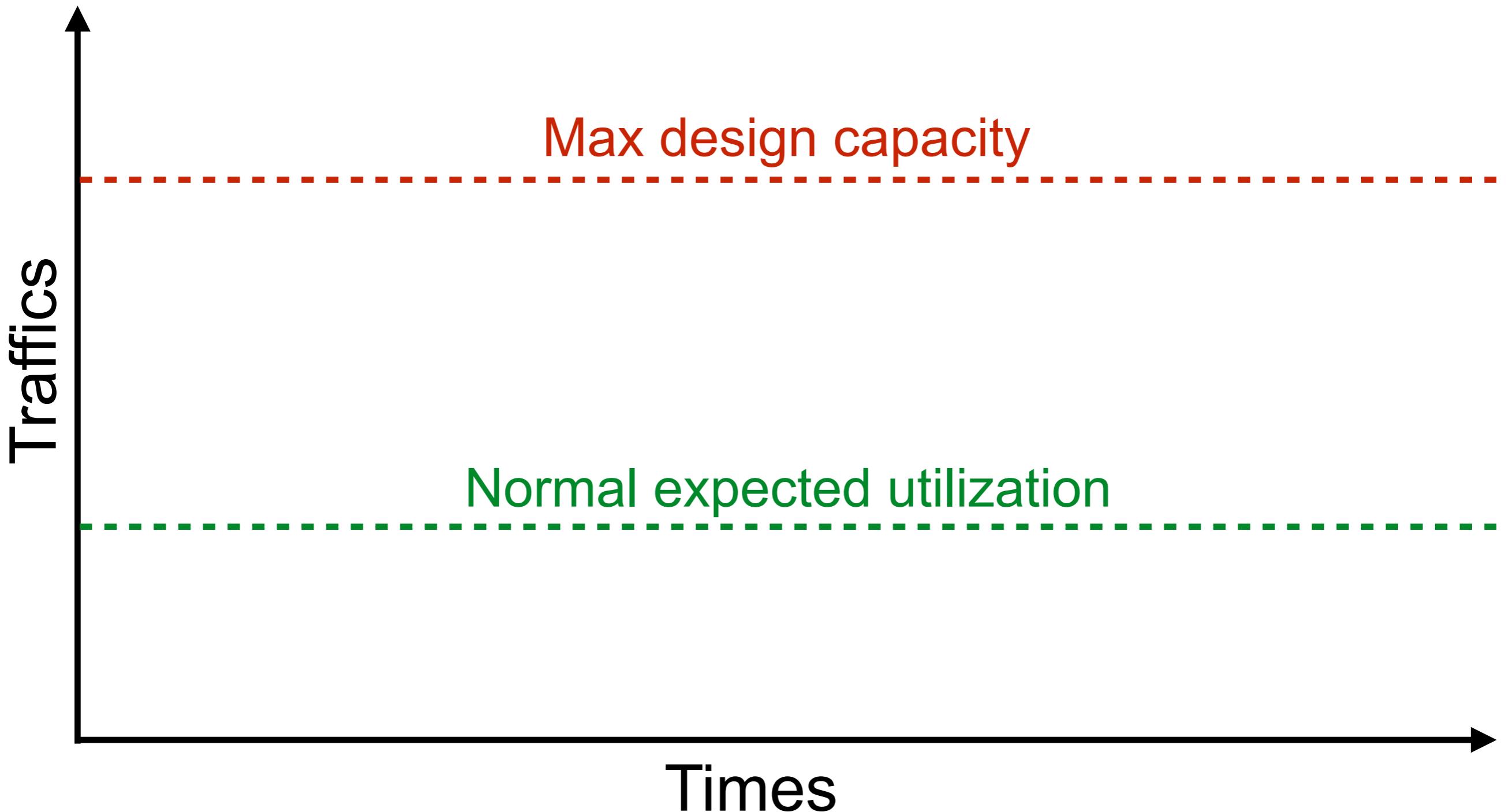
Understand the behaviour of system under
specific expected load

Concurrent, transaction, duration

Identify **bottleneck** in application and hardware



Load testing

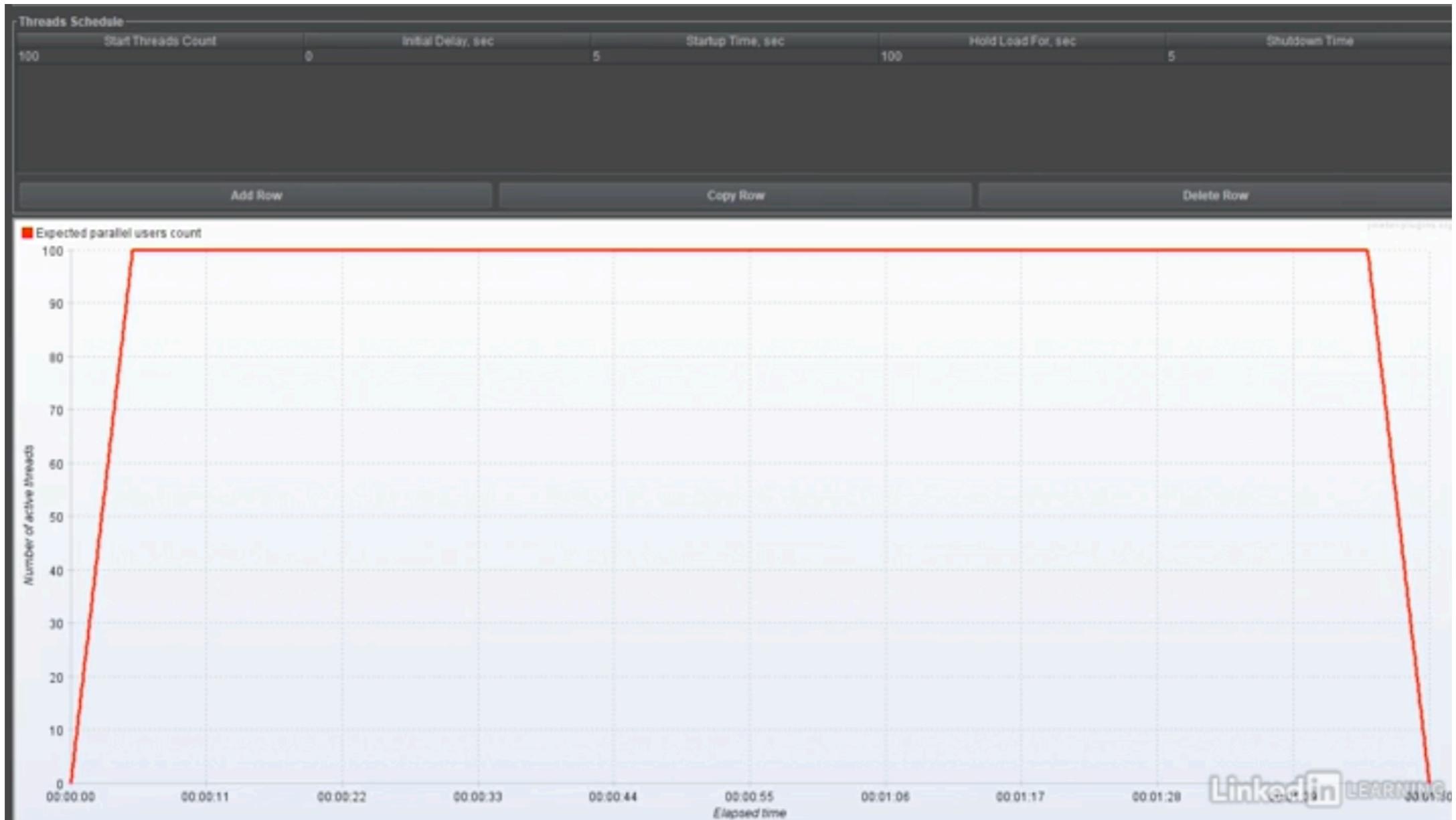


Types of load testing



Basic load

Certain number of users

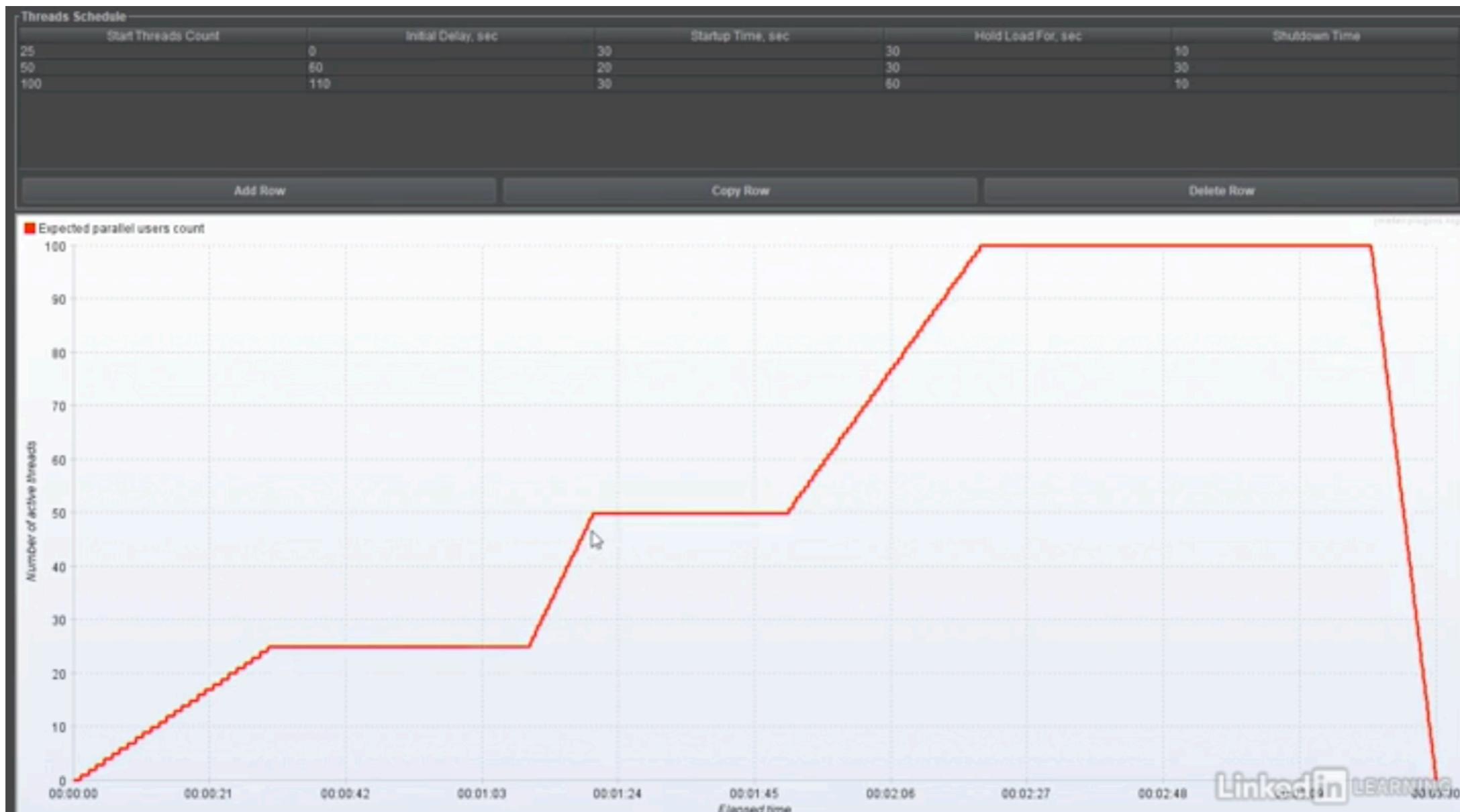


Ramp up

Understand how system response to increased loads overtime



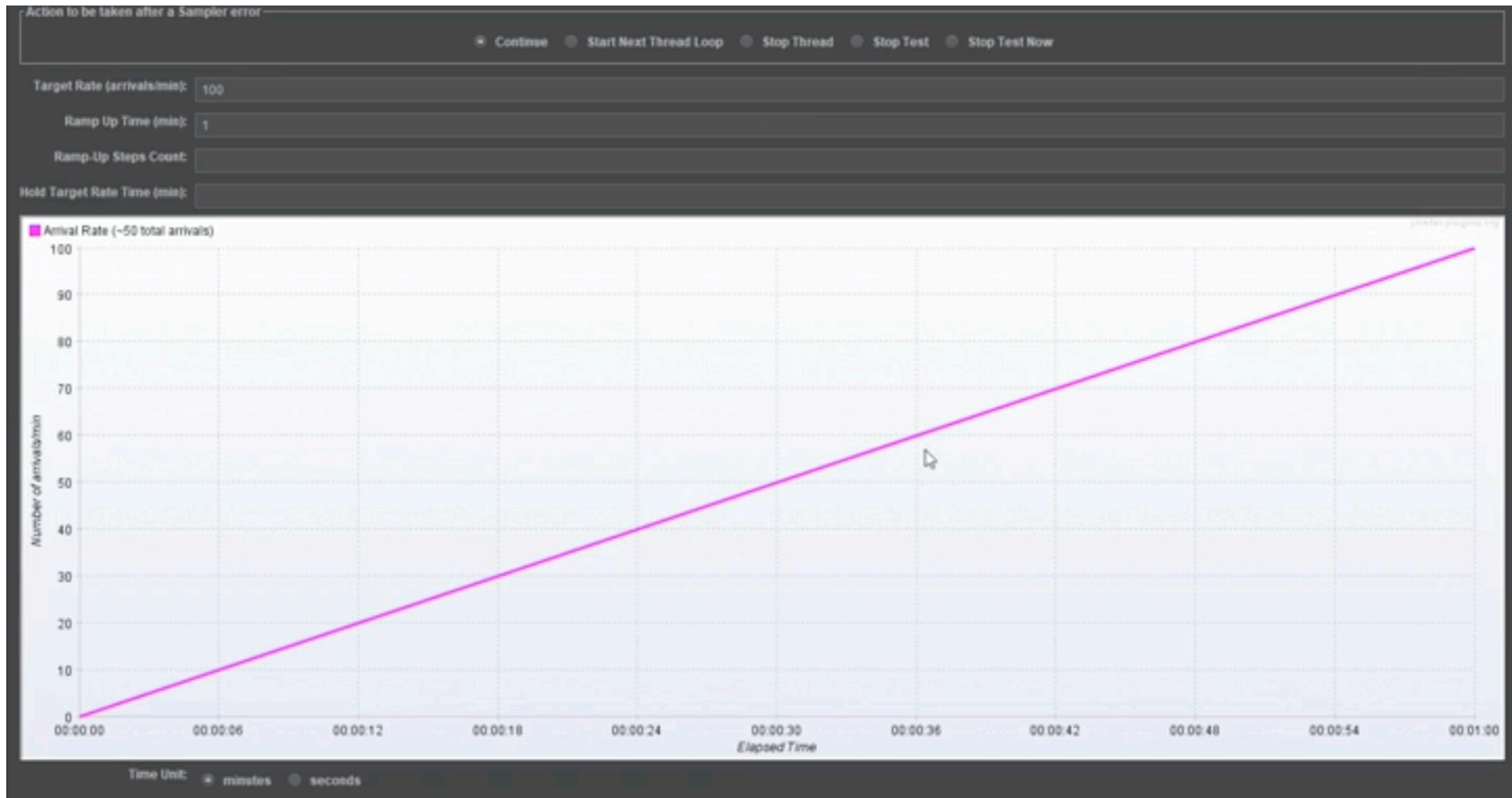
Multi-step ramp up



Random load



Linear load

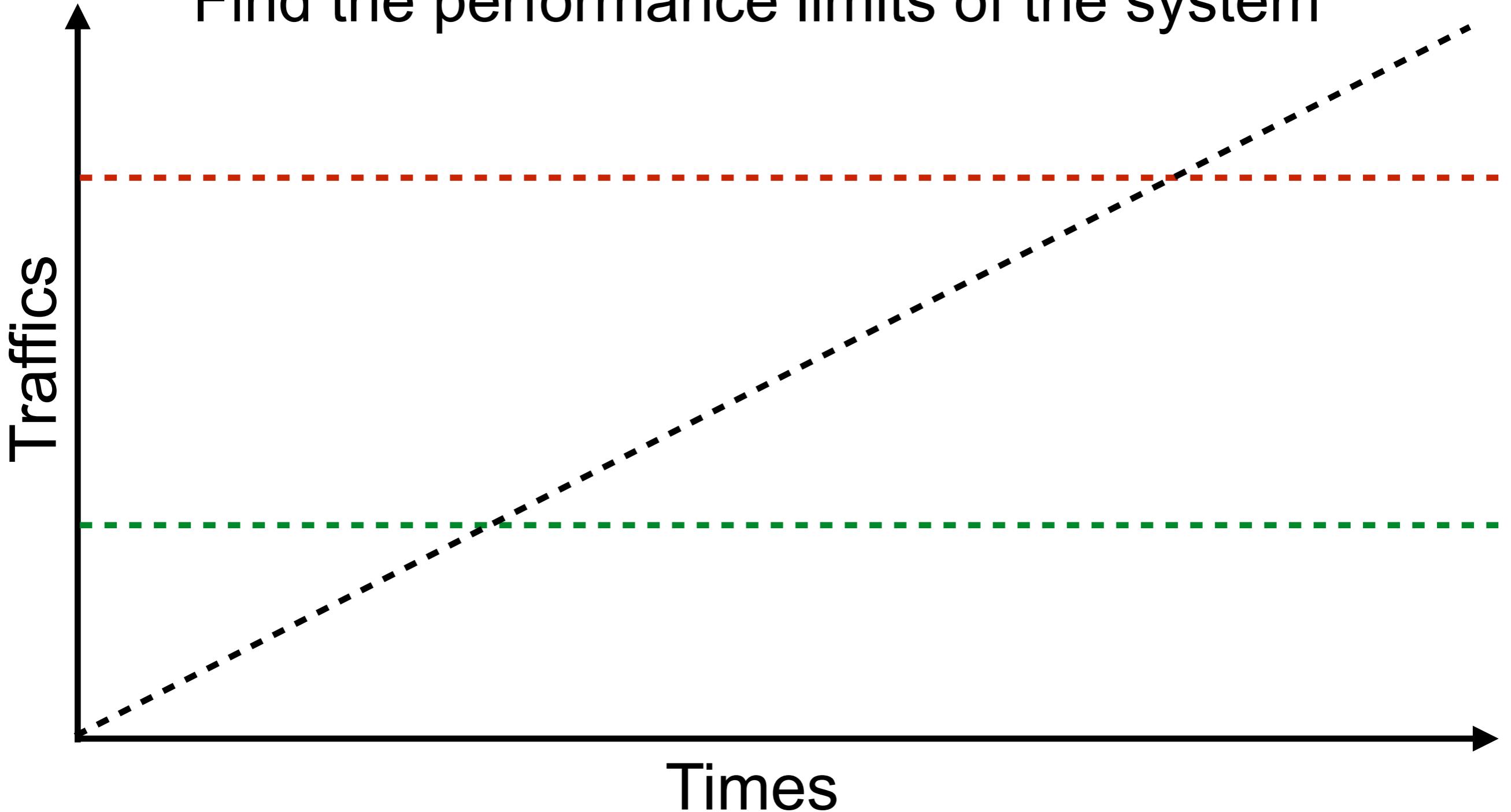


Exponential load

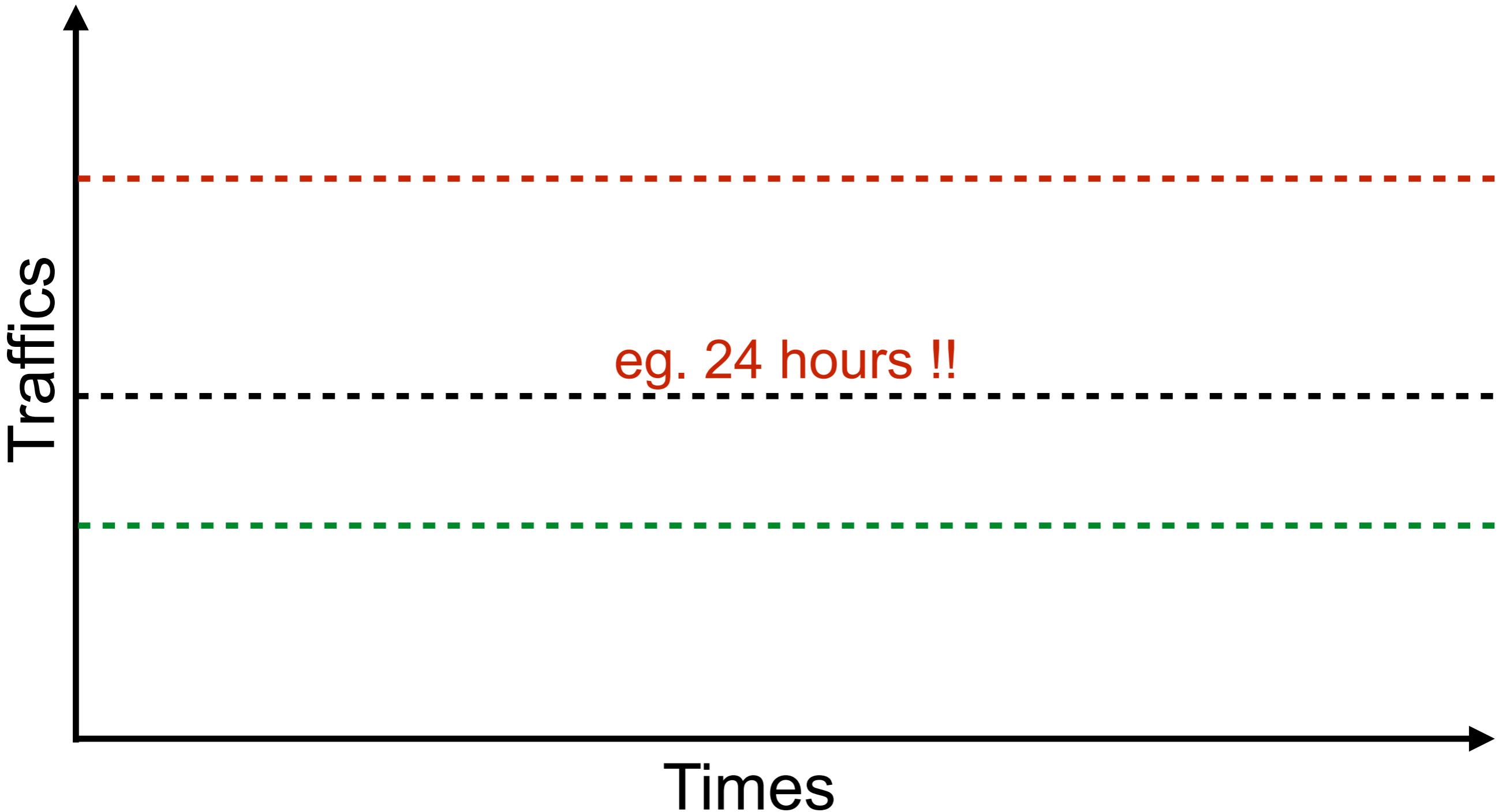


Stress testing

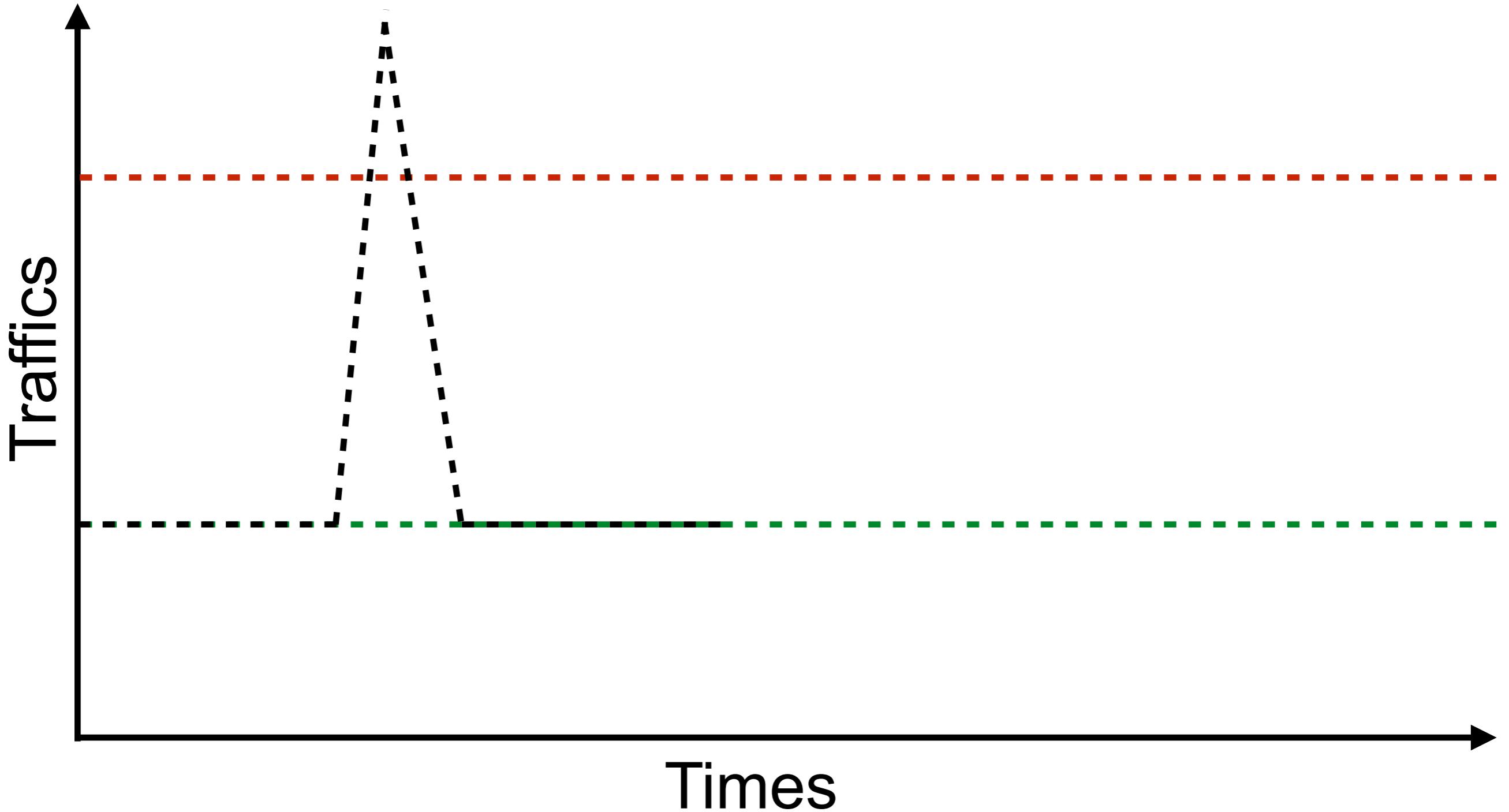
Find the performance limits of the system



Endurance/soak testing

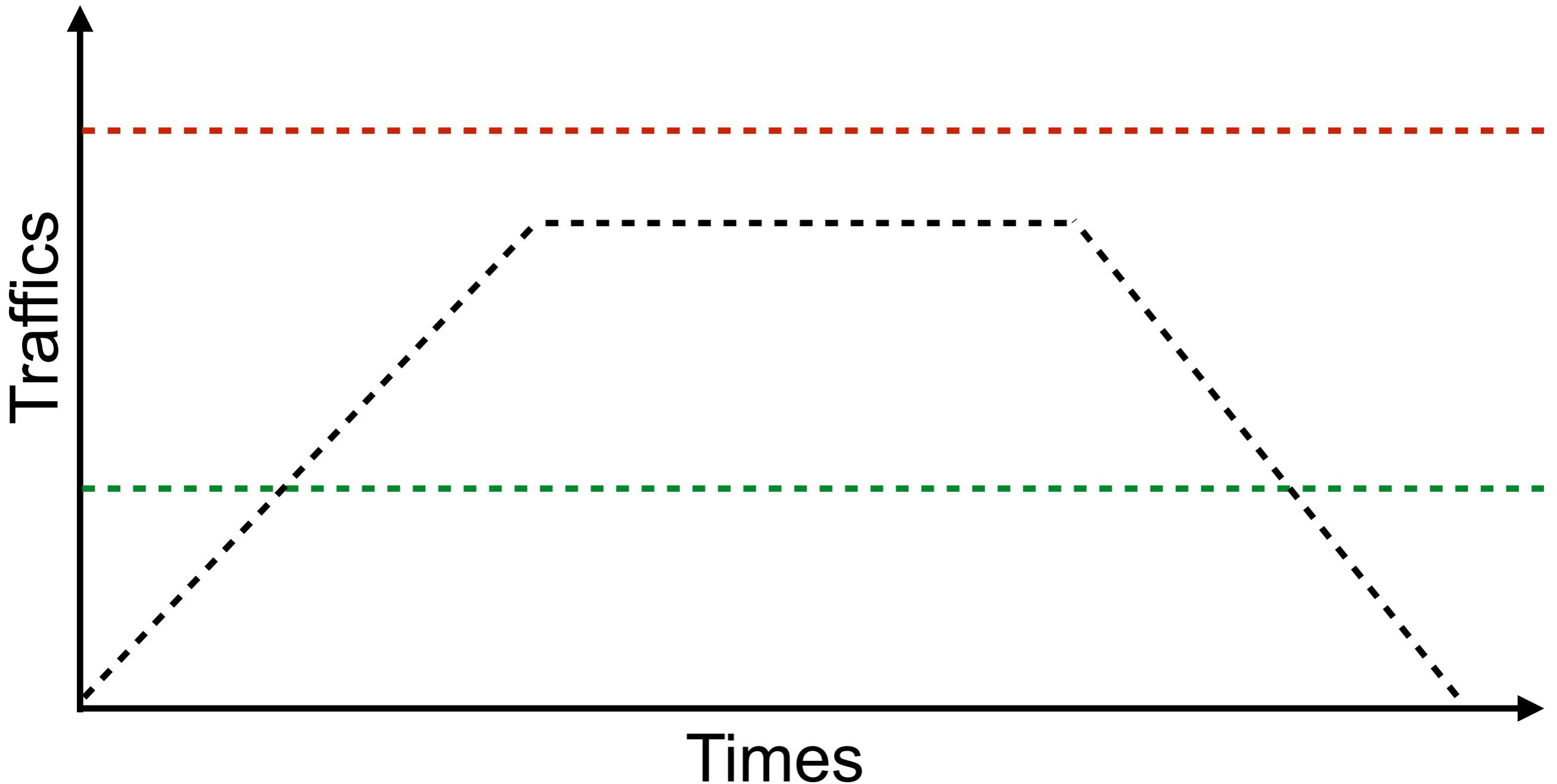


Spike testing



Peak load testing

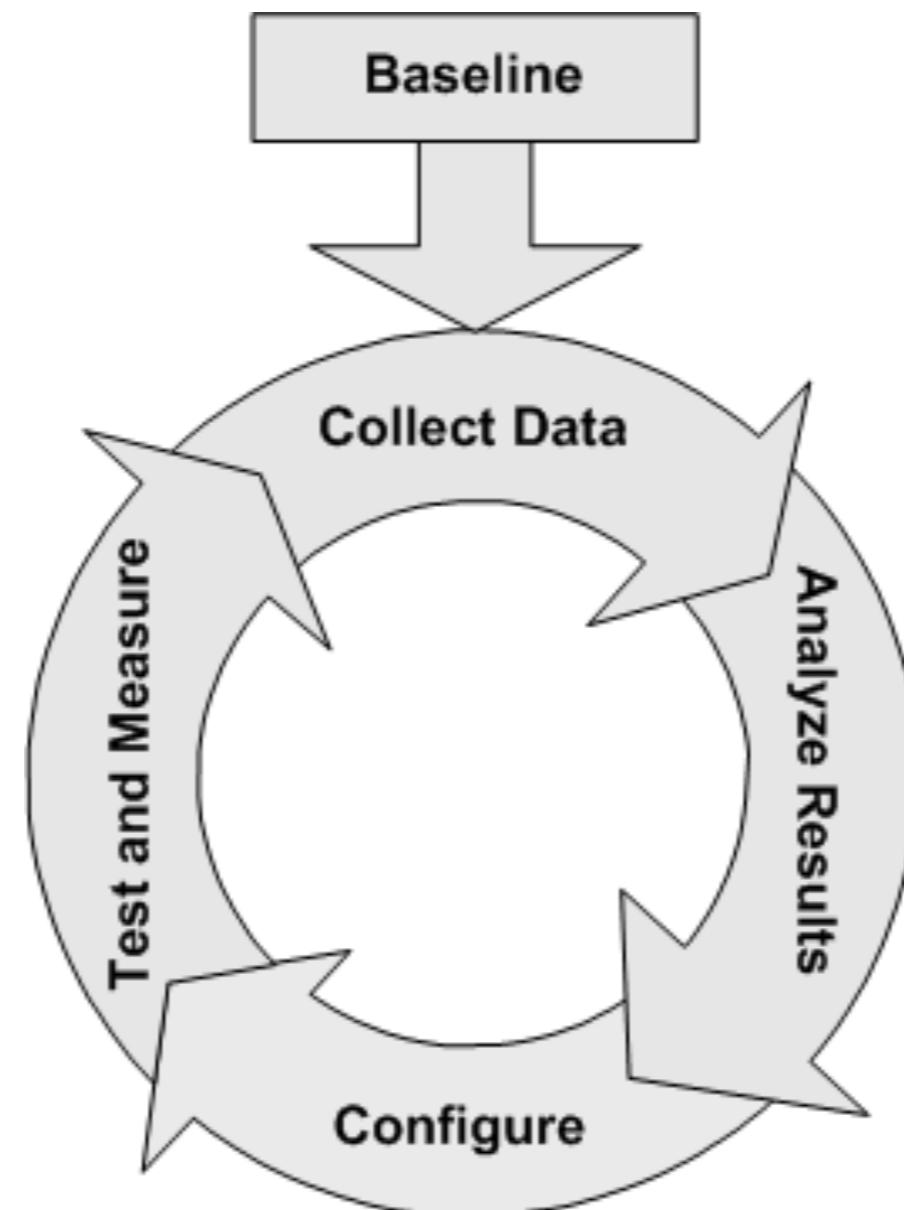
Simulate busy period (End of month payroll cut-off)



Tuning section



Tuning



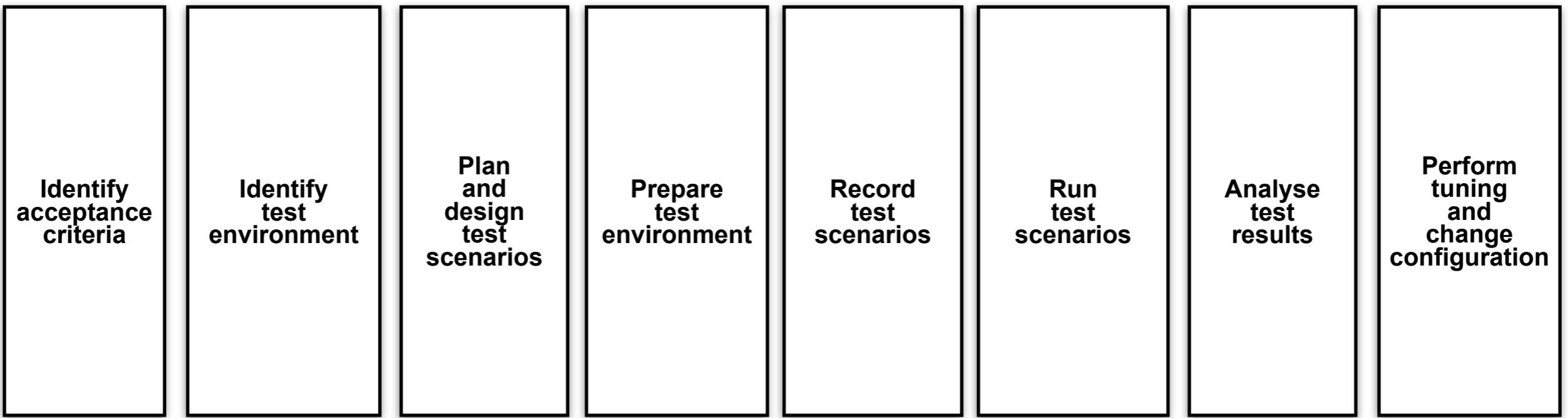
Tuning process for Database

Identify
Gather
Tune
Monitor
Repeat



Performance testing process





Approach

Non-functional requirements captures

Performance test environment build

 Use cases scripting

 Performance test scenario build

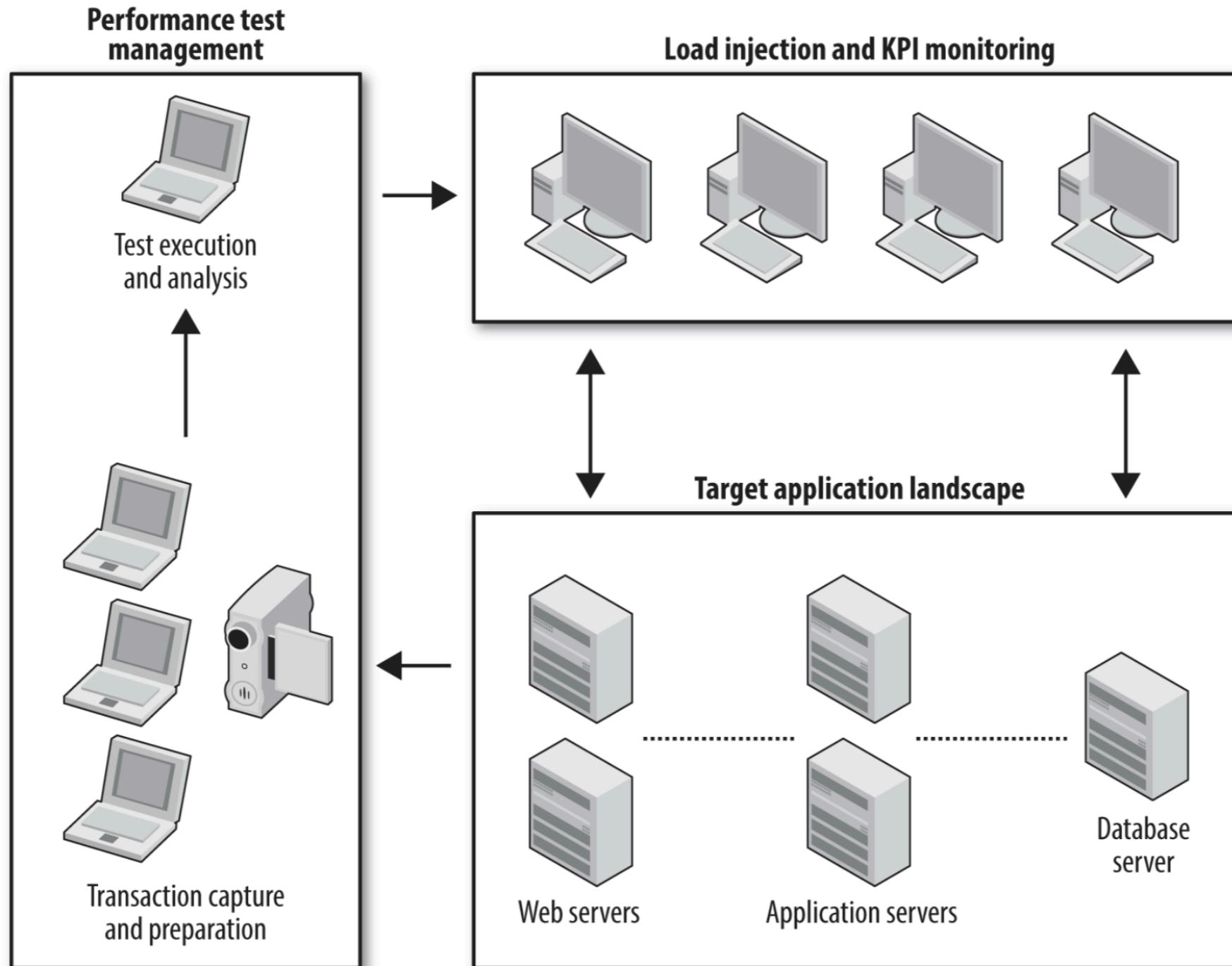
 Performance test execution

 Collect data

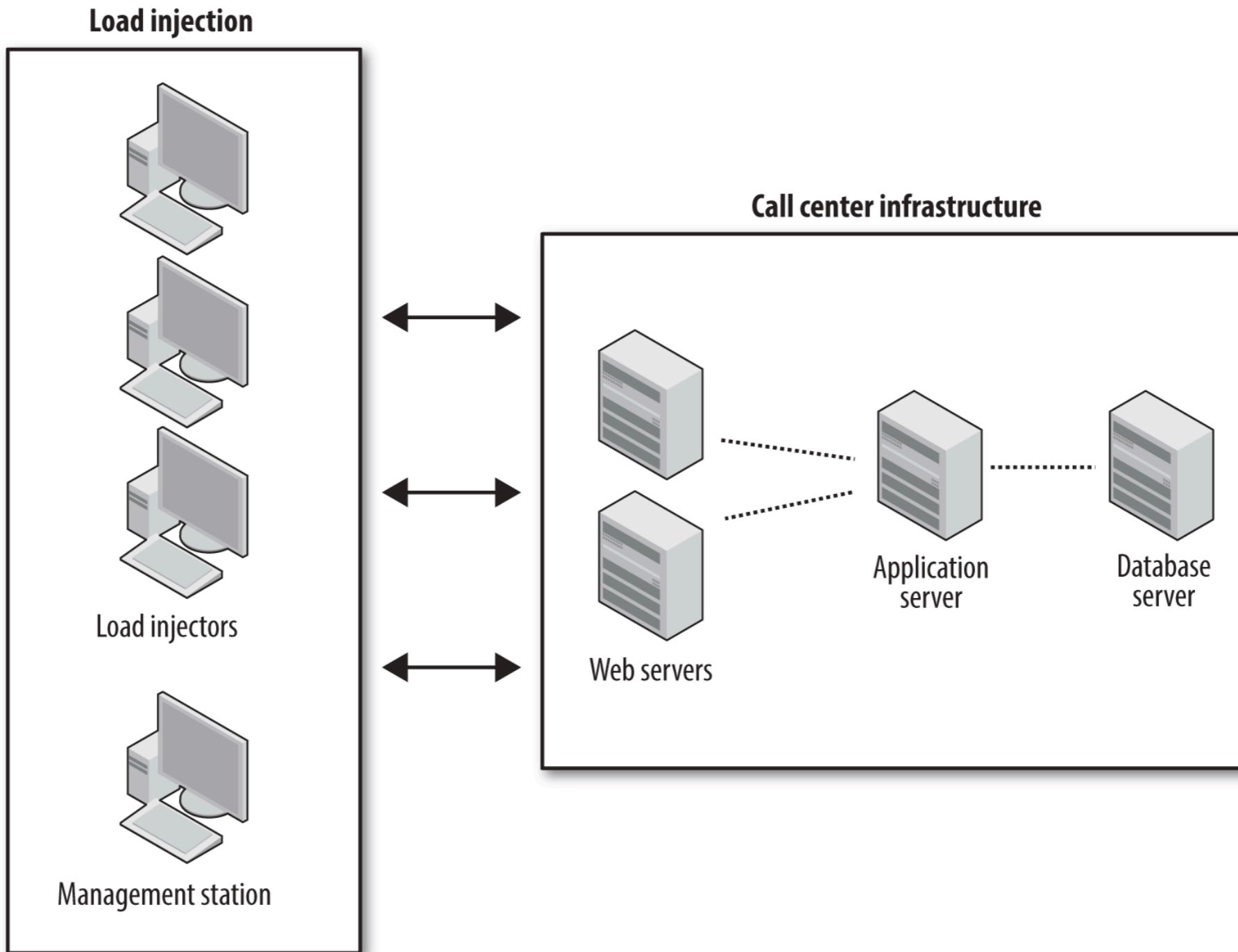
 Analyse and report



Performance testing architecture



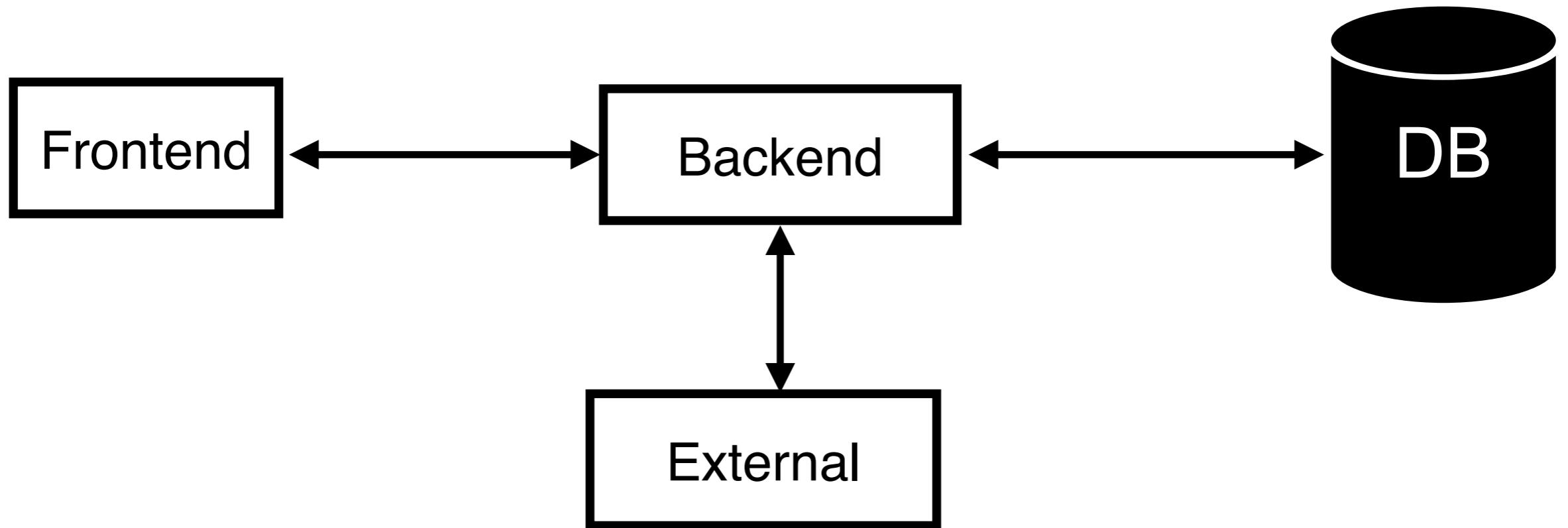
Example



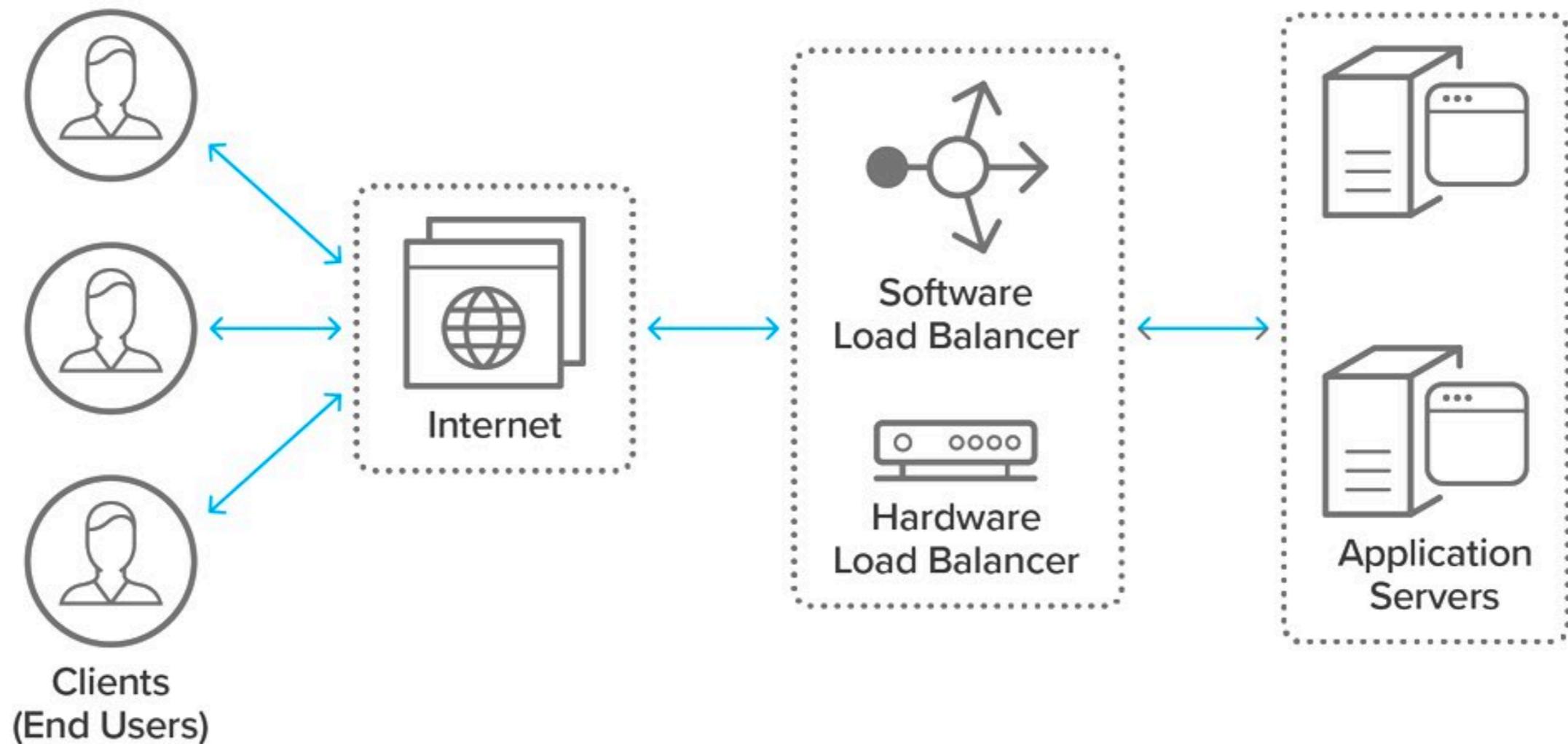
Architecture of system ?



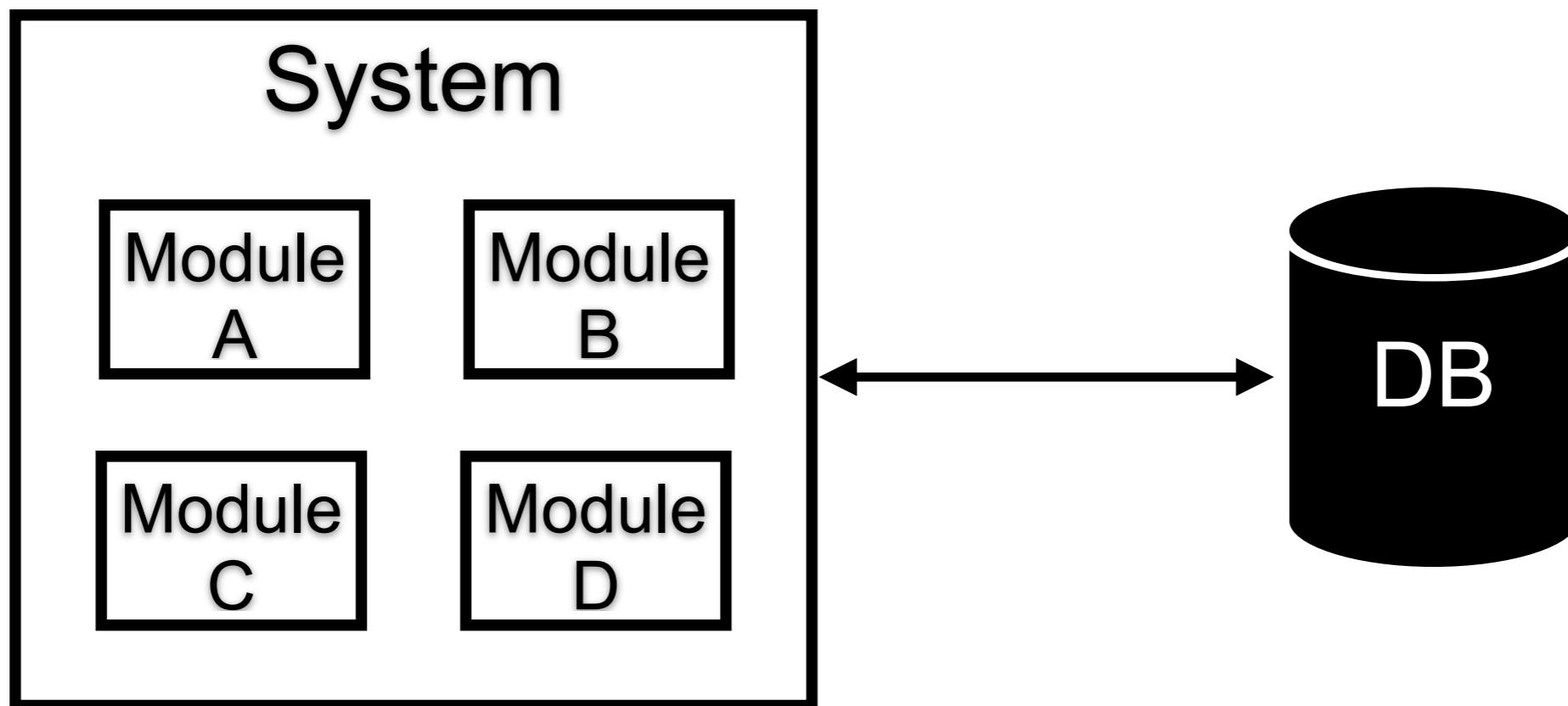
Architecture



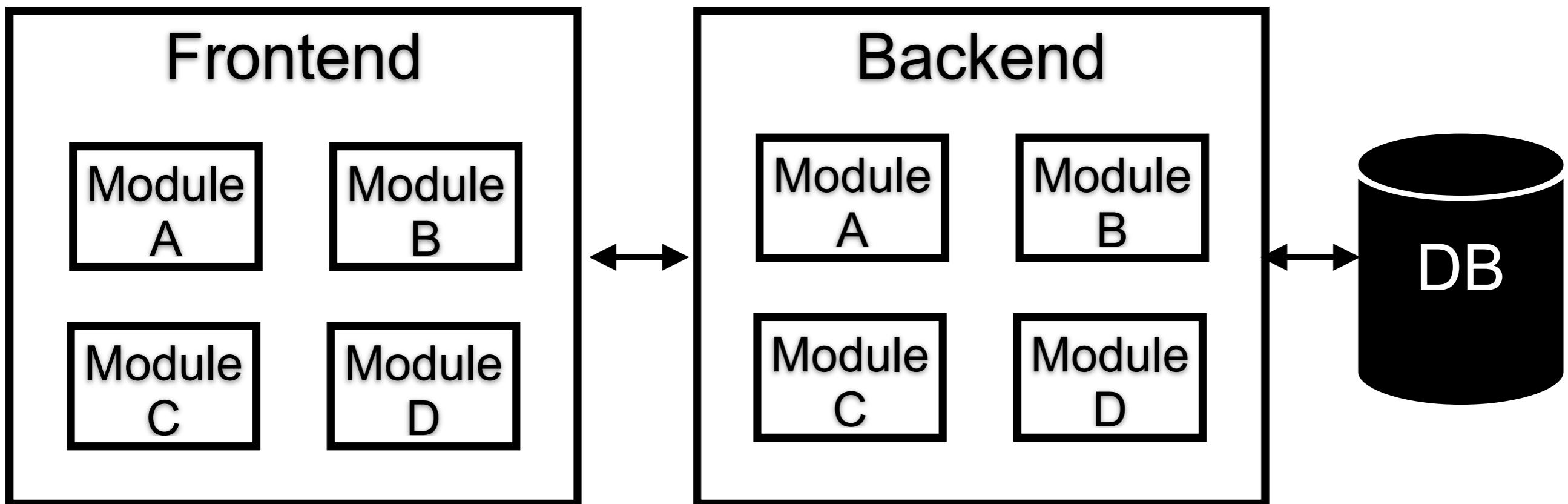
Architecture



Monolith



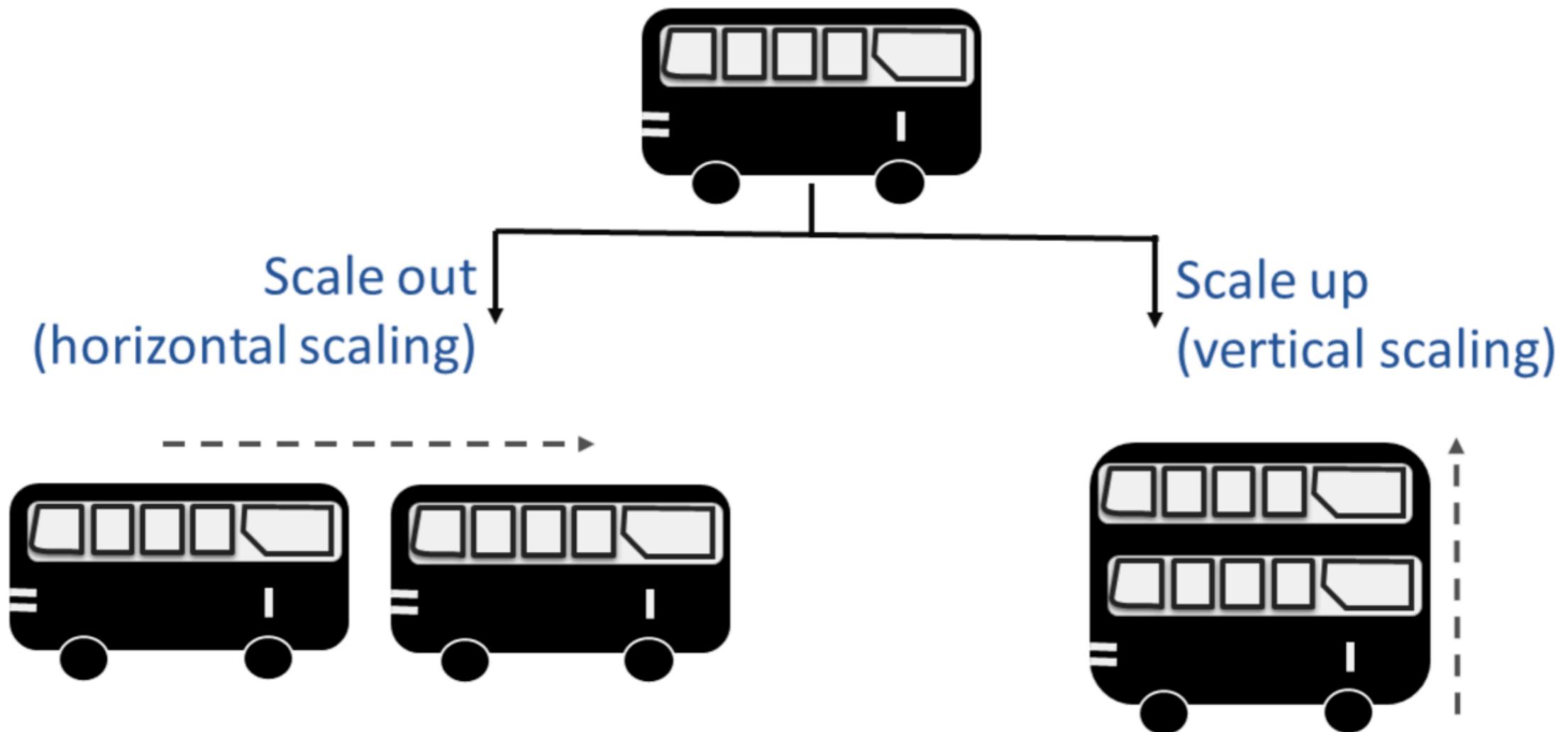
N-Tier



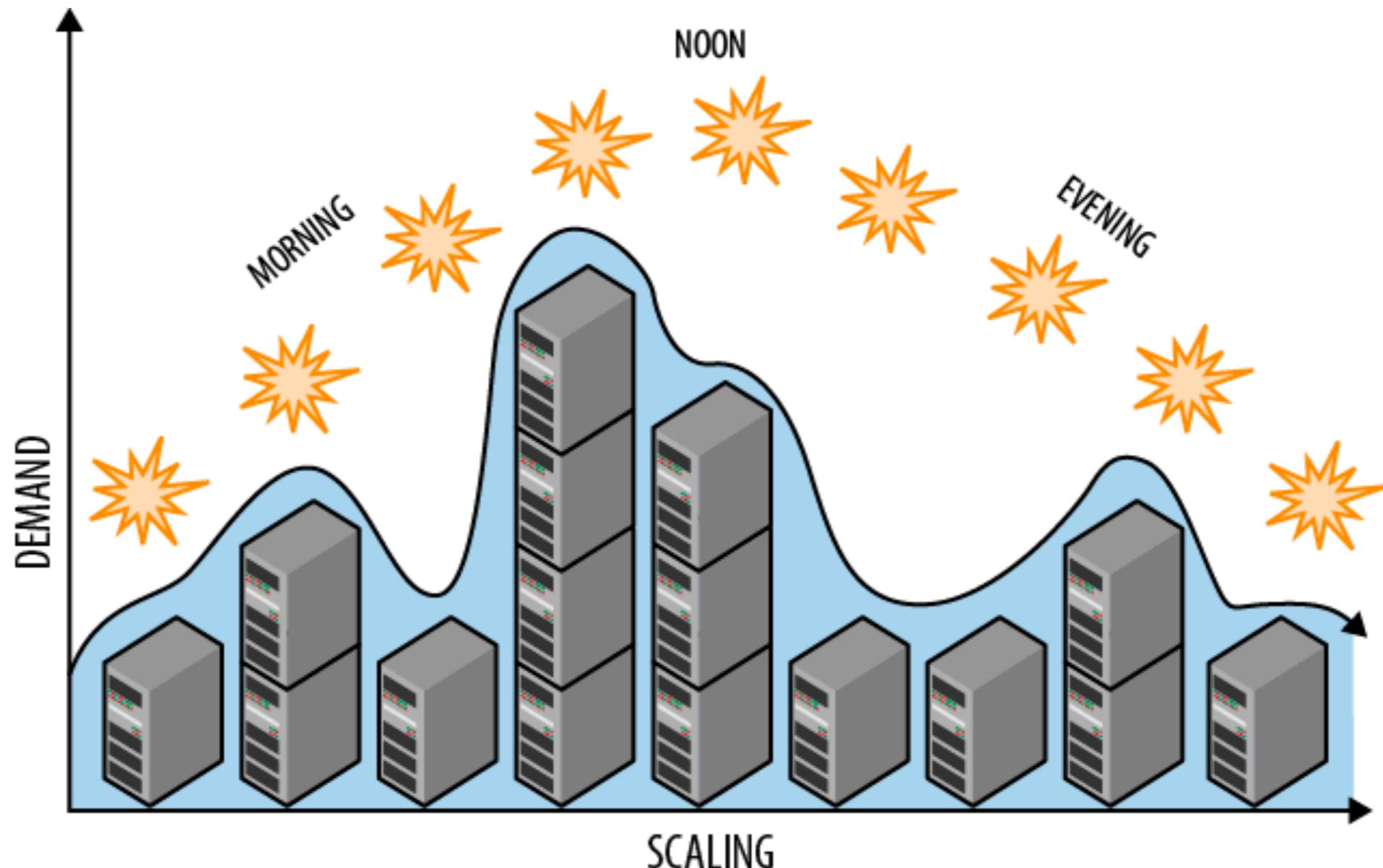
How to scale ?



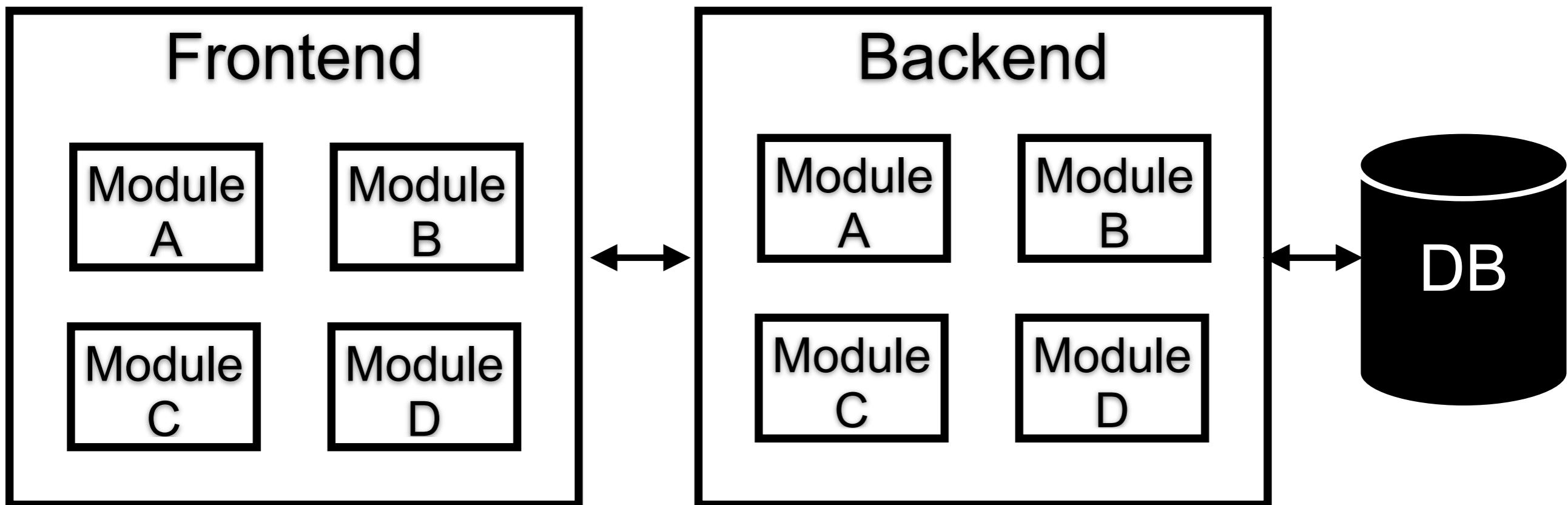
Scaling



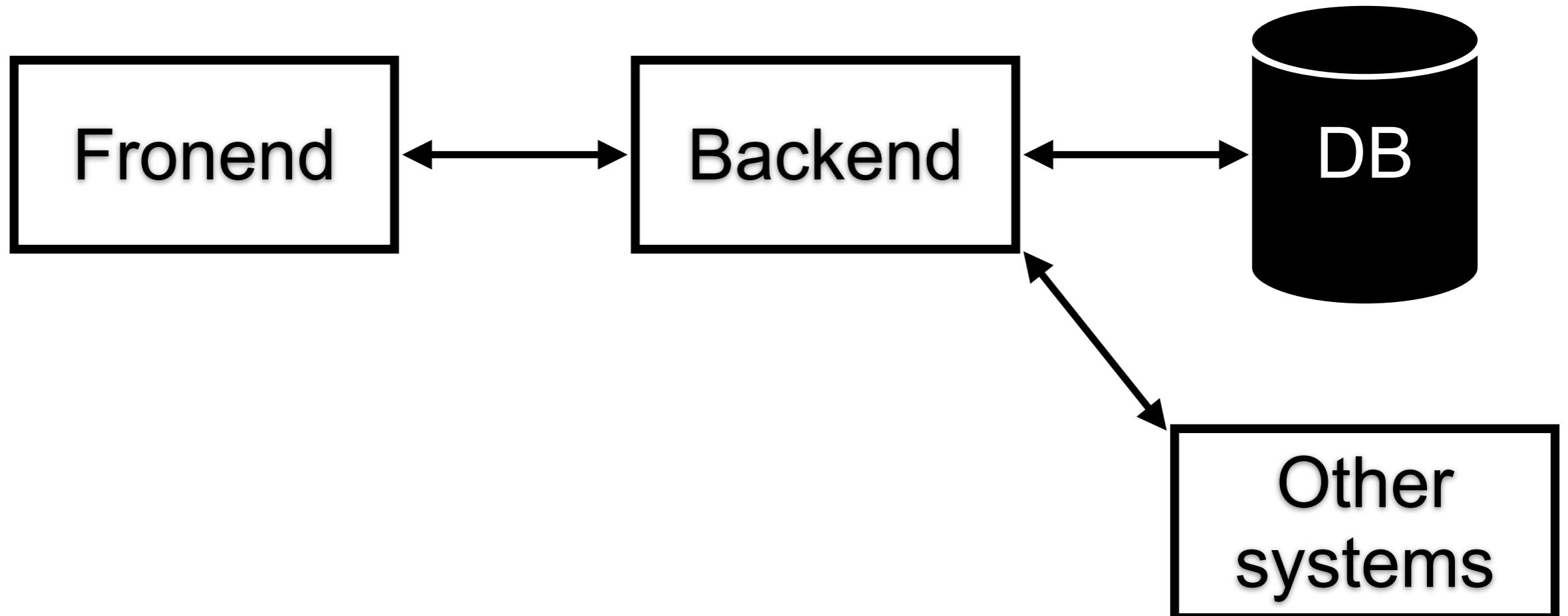
Elastic Scaling



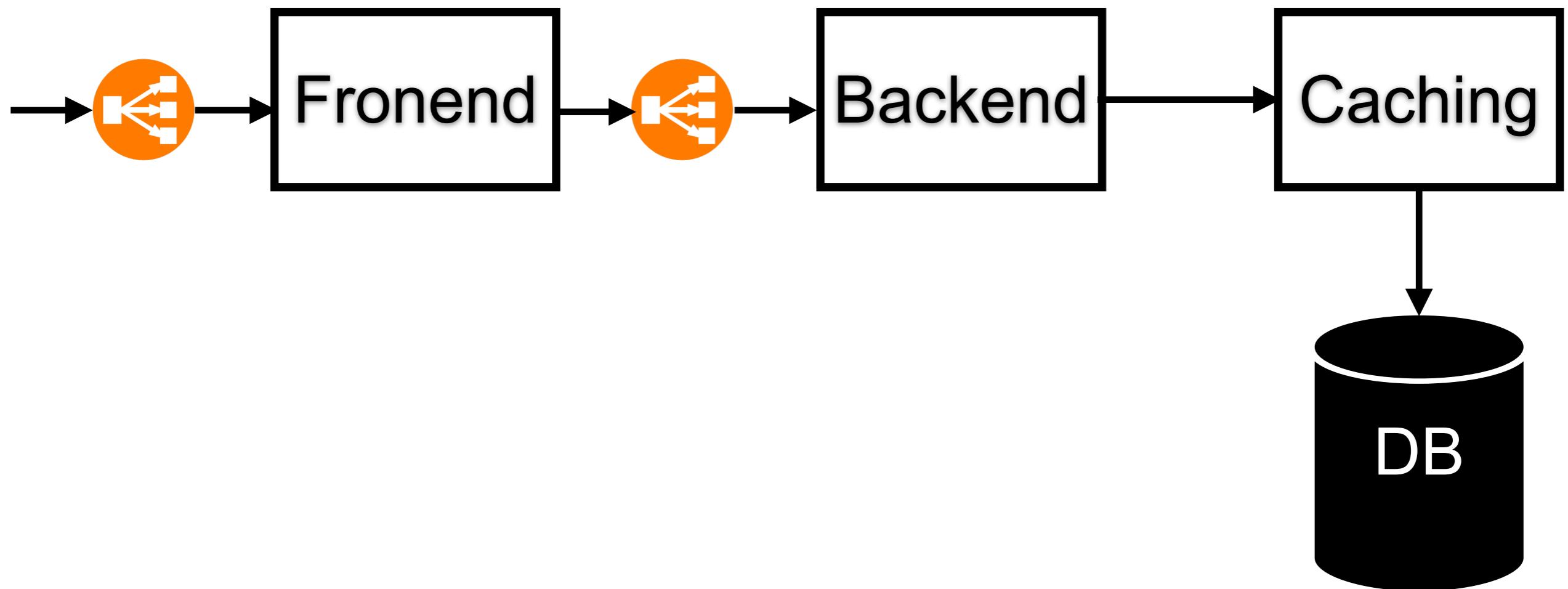
How to scale ?



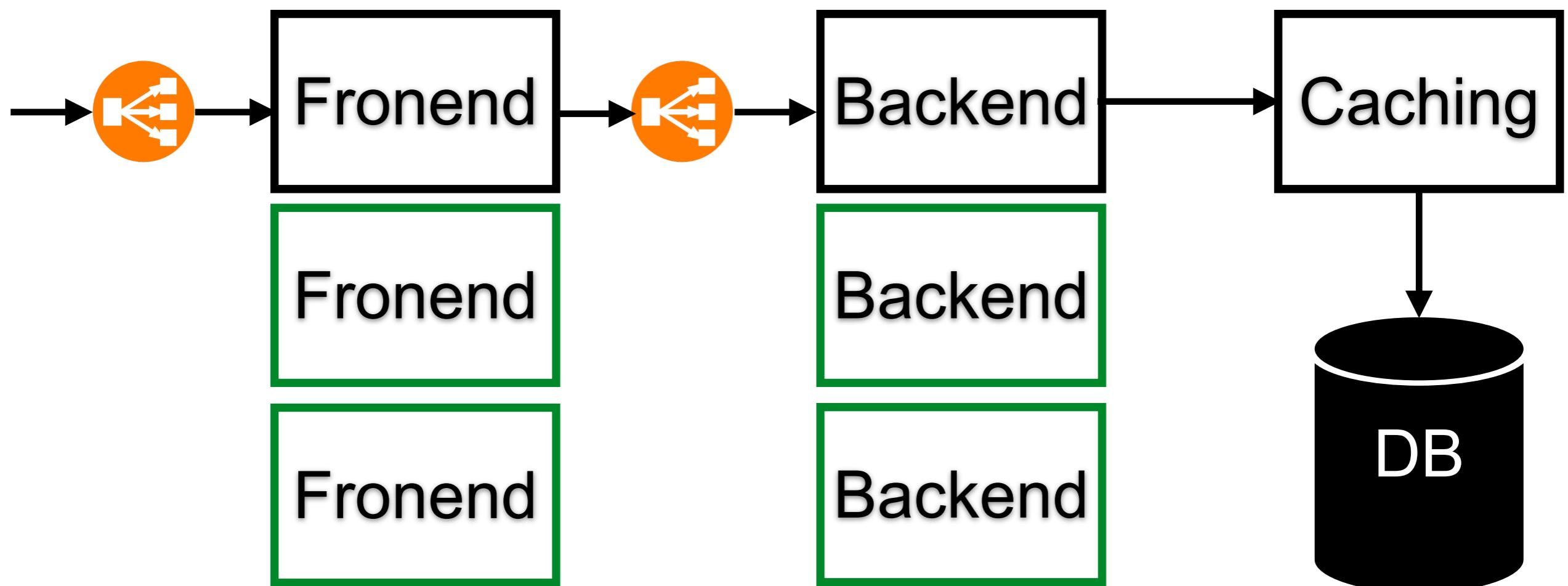
Redesign architecture ?



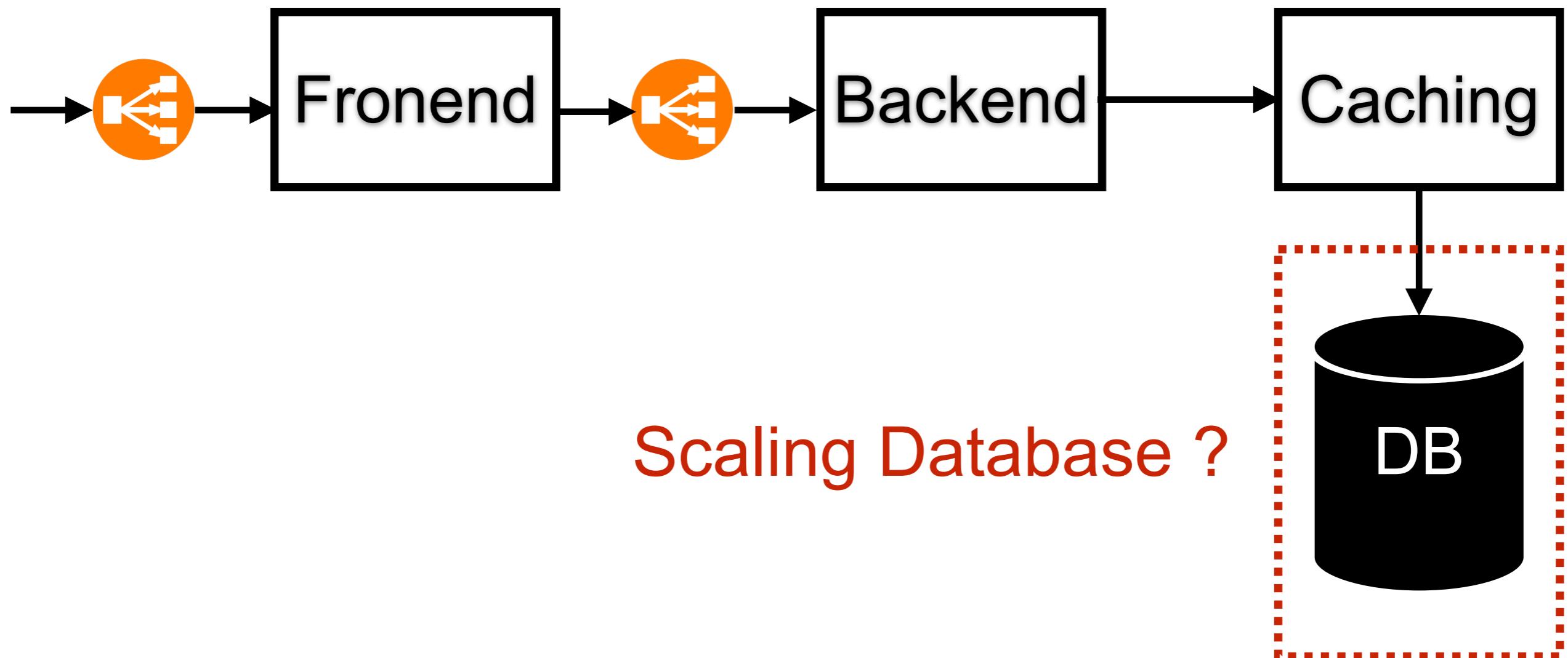
Redesign architecture ?



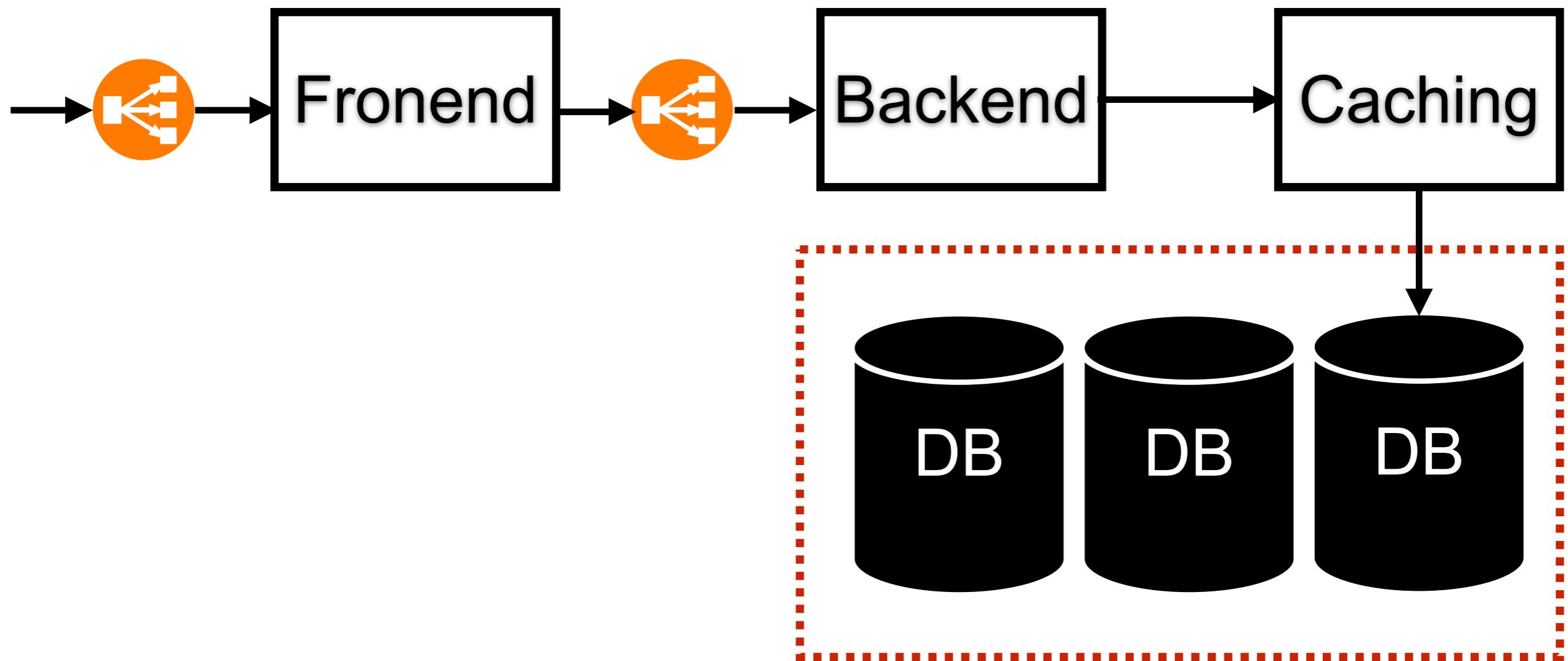
Redesign architecture ?



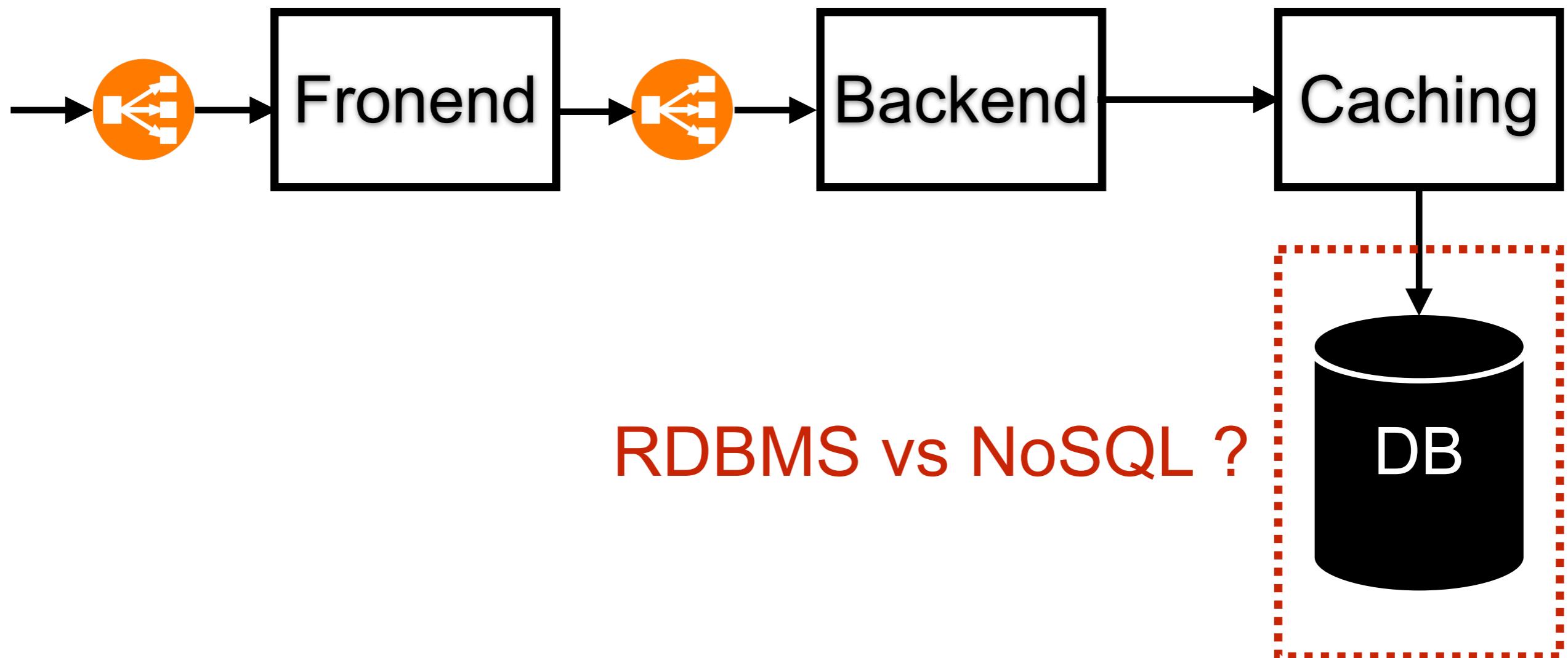
Redesign architecture ?



Redesign architecture ?



Redesign architecture ?



Database model ?

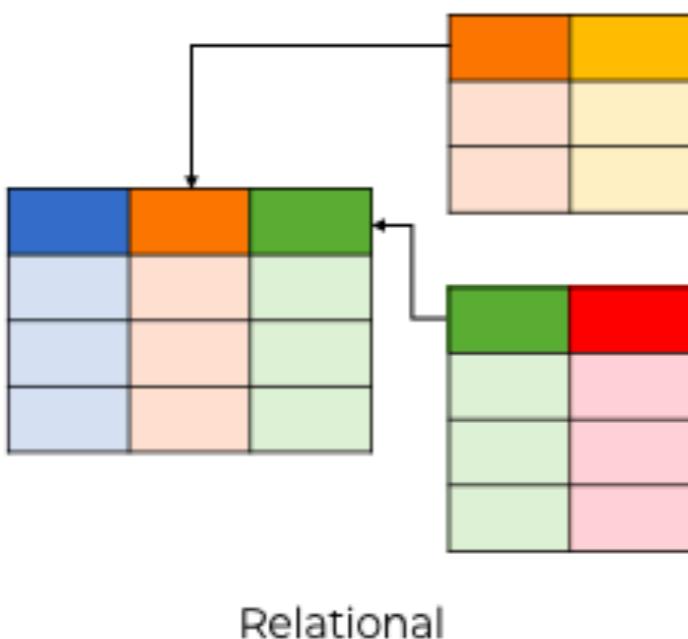
Rank			DBMS	Database Model
Apr 2020	Mar 2020	Apr 2019		
1.	1.	1.	Oracle 	Relational, Multi-model 
2.	2.	2.	MySQL 	Relational, Multi-model 
3.	3.	3.	Microsoft SQL Server 	Relational, Multi-model 
4.	4.	4.	PostgreSQL 	Relational, Multi-model 
5.	5.	5.	MongoDB 	Document, Multi-model 
6.	6.	6.	IBM Db2 	Relational, Multi-model 
7.	7.	8.	Elasticsearch 	Search engine, Multi-model 
8.	8.	7.	Redis 	Key-value, Multi-model 
9.	↑ 10.	↑ 10.	SQLite 	Relational
10.	↓ 9.	↓ 9.	Microsoft Access	Relational

<https://db-engines.com/en/ranking>



Database model

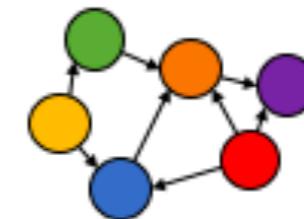
SQL DATABASES



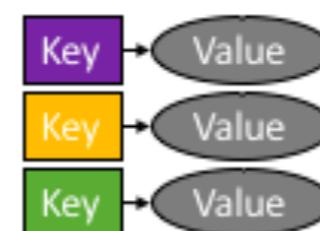
NoSQL DATABASES



Column



Graph



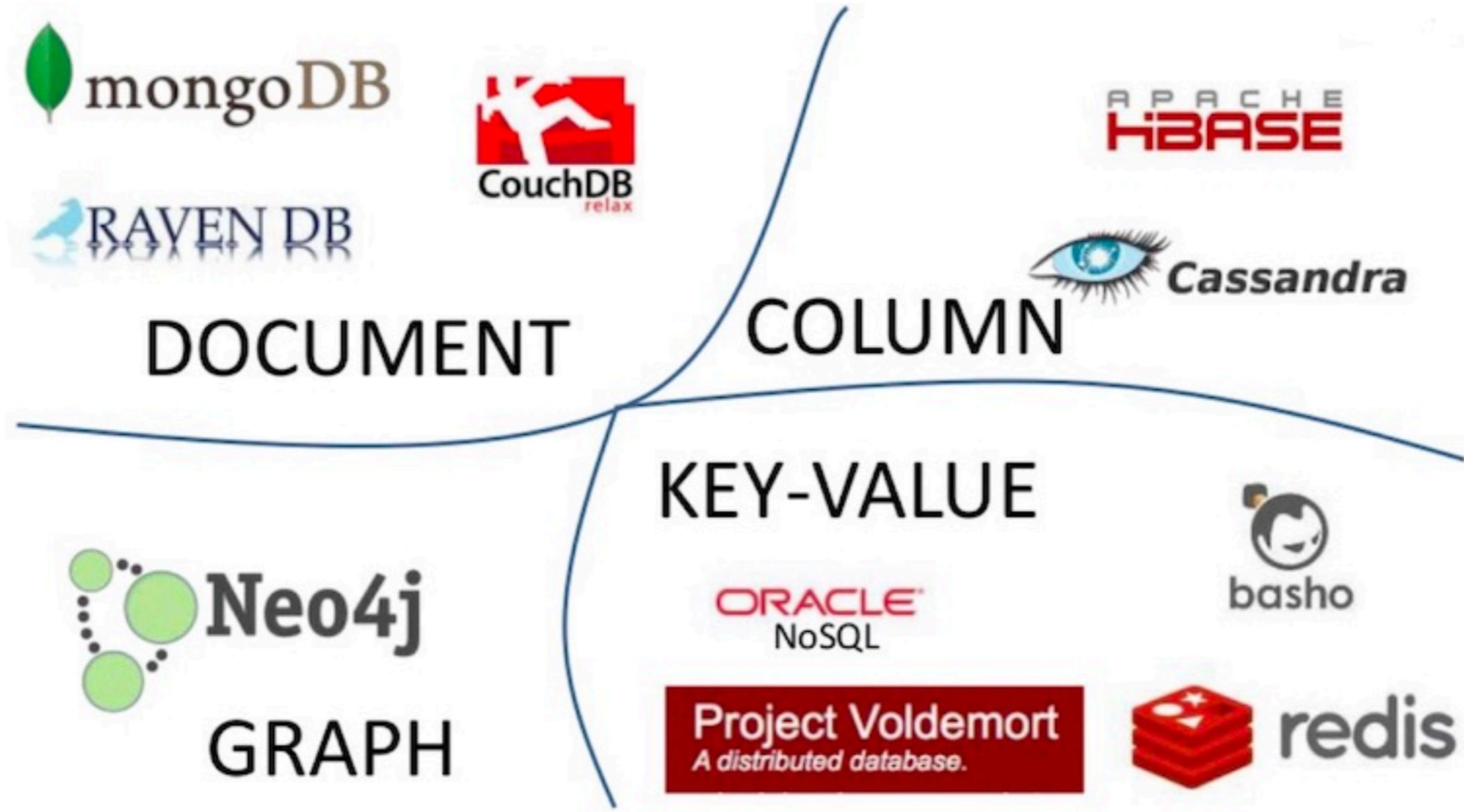
Key-Value



Document

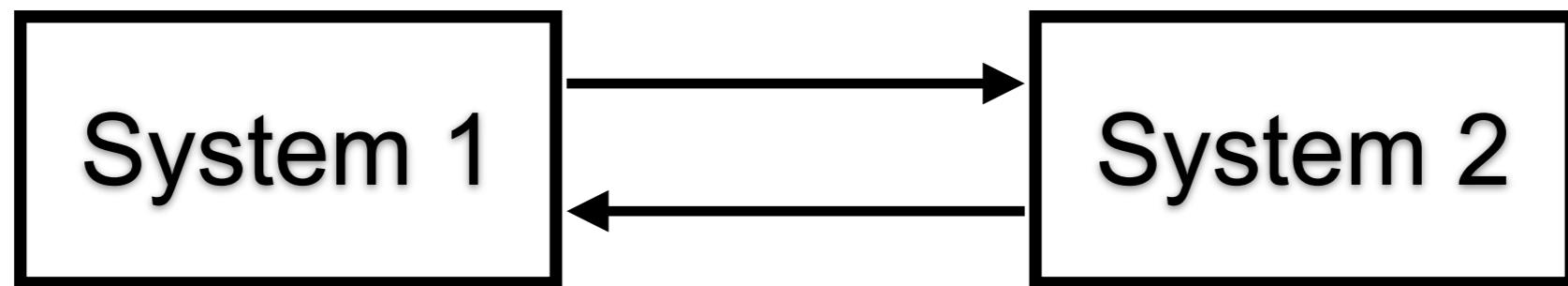


NoSQL



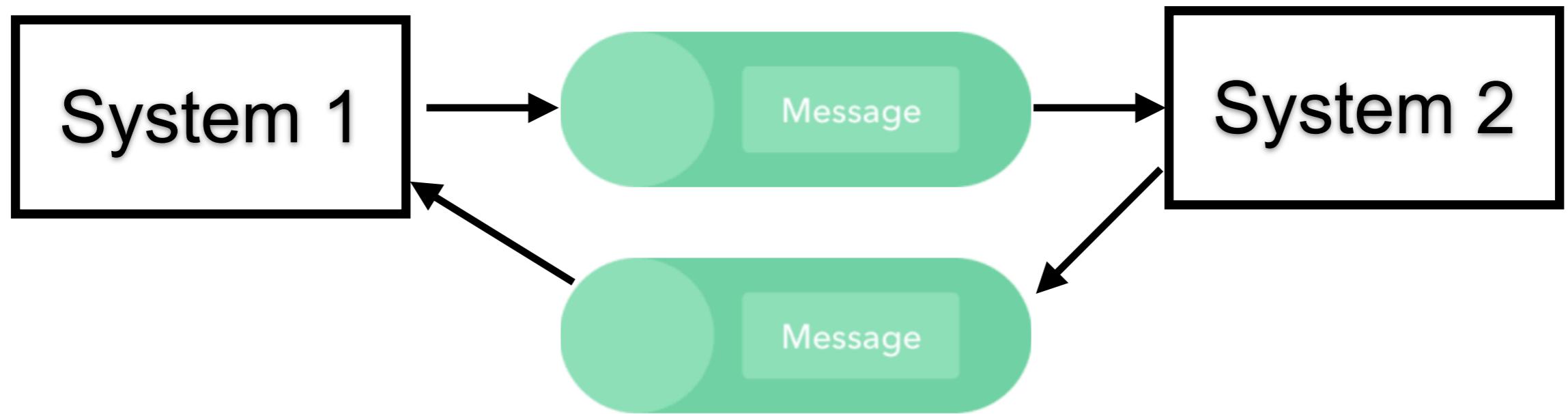
Redesign architecture ?

Synchronous communication

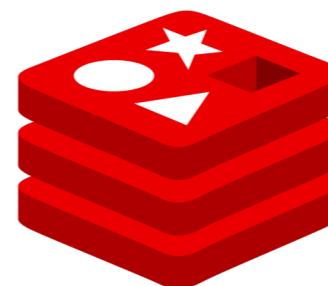


Redesign architecture ?

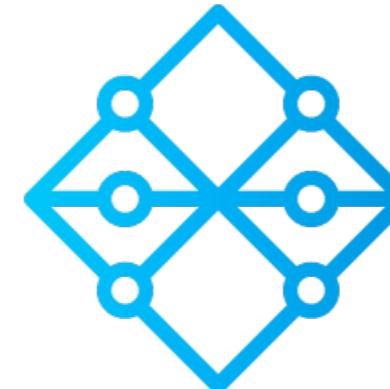
Asynchronous communication



Messaging system



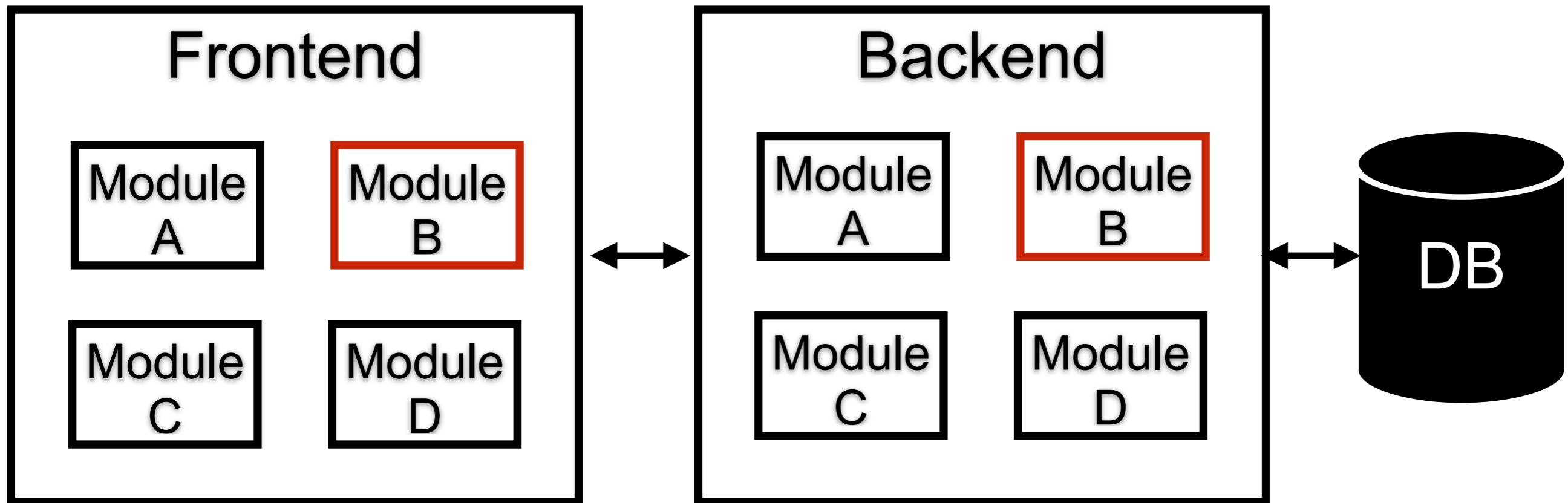
redis



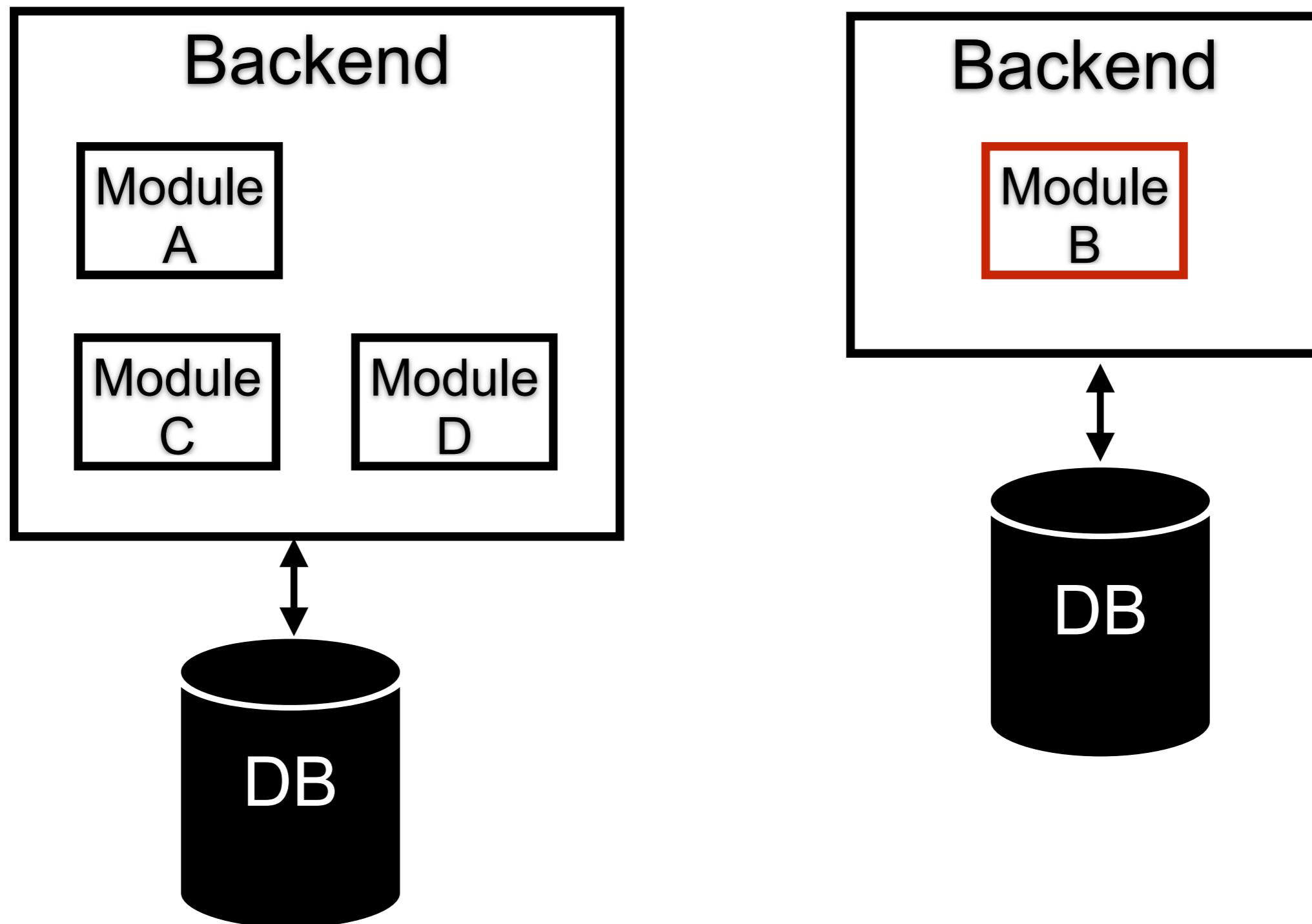
nqs



How to scale ?



Split to new service



12 factor app



THE TWELVE-FACTOR APP

INTRODUCTION

In the modern era, software is commonly delivered as a service: called *web apps*, or *software-as-a-service*. The twelve-factor app is a methodology for building software-as-a-service apps that:

- Use **declarative** formats for setup automation, to minimize time and cost for new developers joining the project;
- Have a **clean contract** with the underlying operating system, offering **maximum portability** between execution environments;
- Are suitable for deployment on modern **cloud platforms**, obviating the need for servers and systems administration;
- **Minimize divergence** between development and production, enabling **continuous deployment** for maximum agility;
- And can **scale up** without significant changes to tooling, architecture, or development practices.

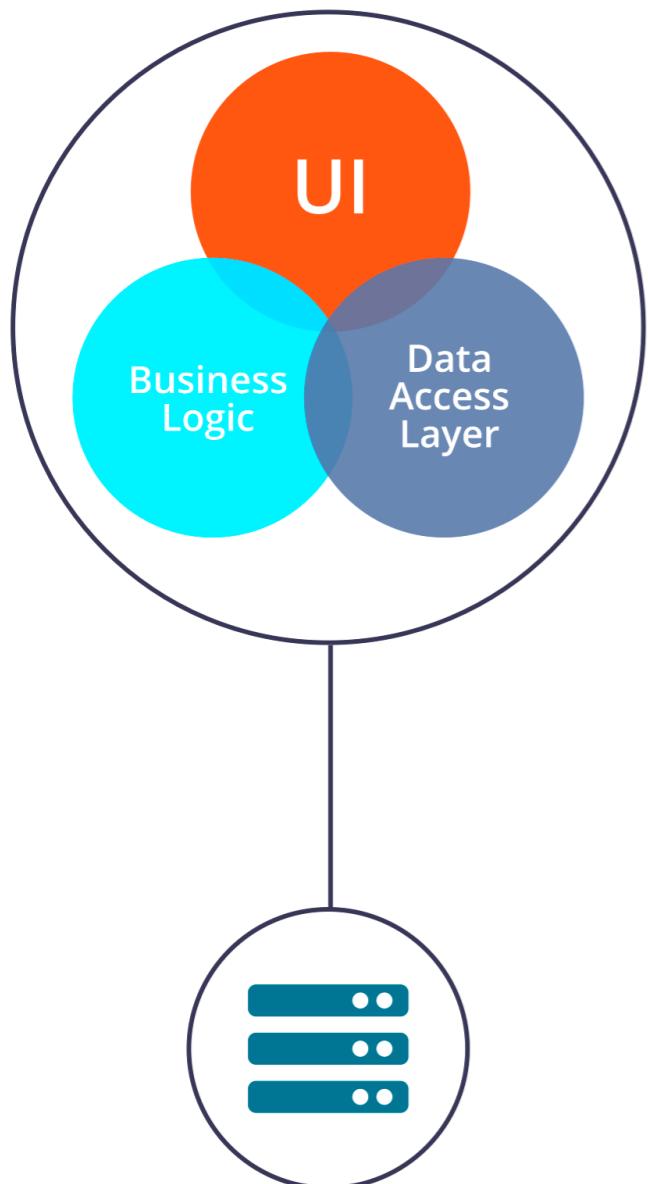
<https://12factor.net/>



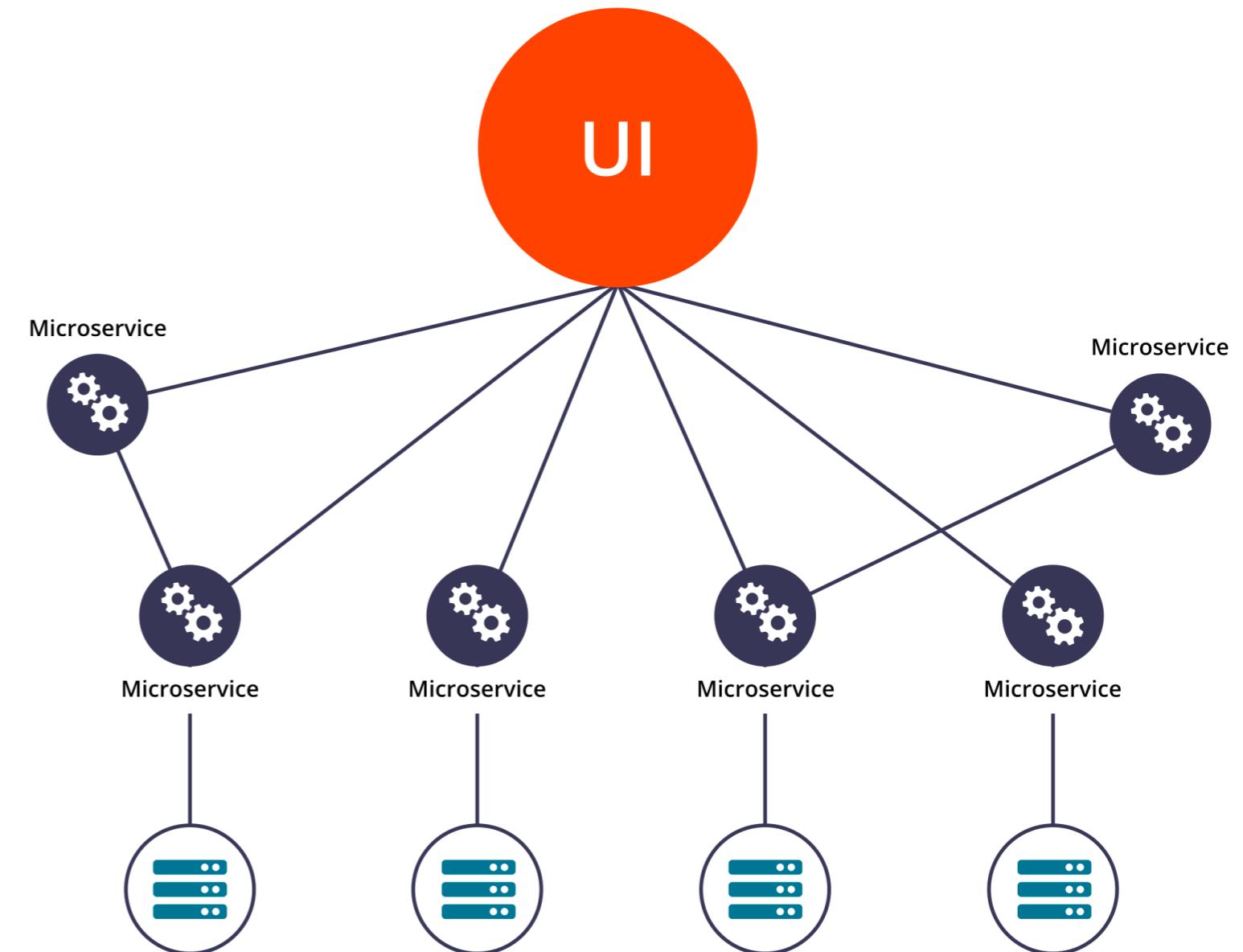
Performance testing

© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

Microservices



Monolithic Architecture



Microservice Architecture



Scalability testing for your architecture



Scalability testing ?

Compute resource allocated based on dynamic demand

Low usage => small server

Increases usage => growing server size

Test your architecture



What to check for in scalability testing ?

Too few resources

Too many resources

Frequently changing resource allocation



Use cases





มาตรการเยียวยา 5,000 บาท (3 เดือน)

ลงทะเบียนเข้าร่วมมาตรการได้ตั้งแต่วันเสาร์ที่ 28 มีนาคม 2563 เวลา 18.00 น. เป็นต้นไป



ลงทะเบียนมาตรการ >

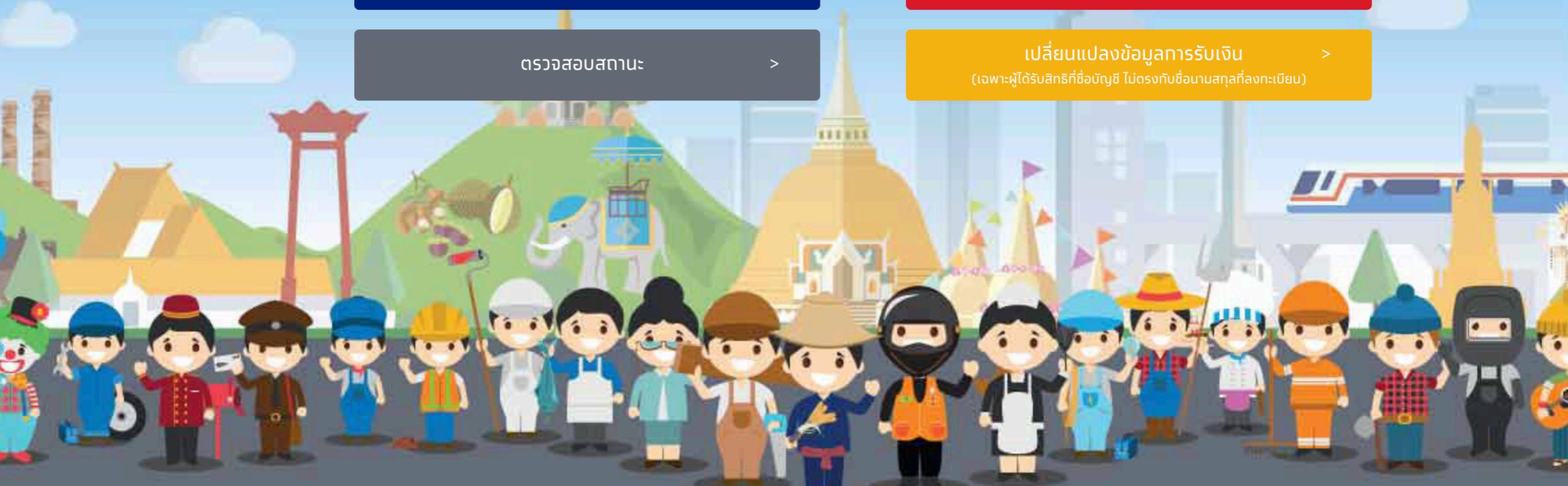
ตรวจสอบสถานะ >

ยกเลิกการลงทะเบียน

(ไม่สามารถลงทะเบียนได้อีก)

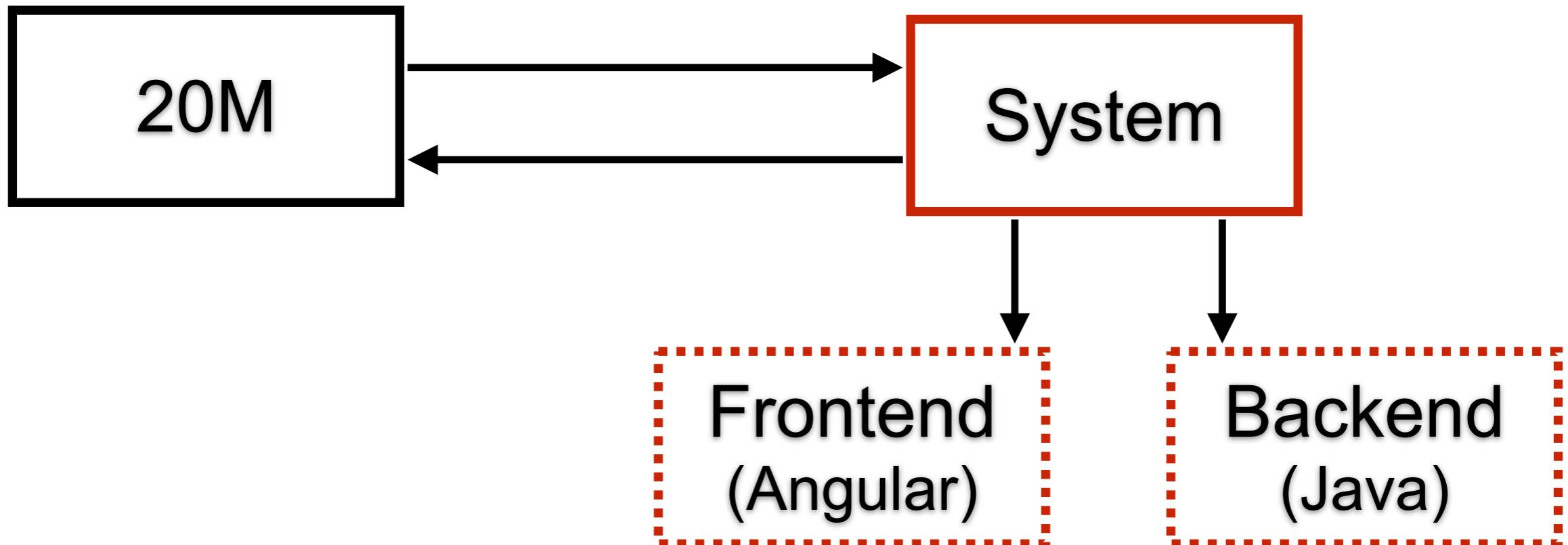
เปลี่ยนแปลงข้อมูลการรับเงิน

(เฉพาะผู้ได้รับสิทธิ์ซื้อบัญชี ไม่ตรงกับชื่อนามสกุลที่ลงทะเบียน)



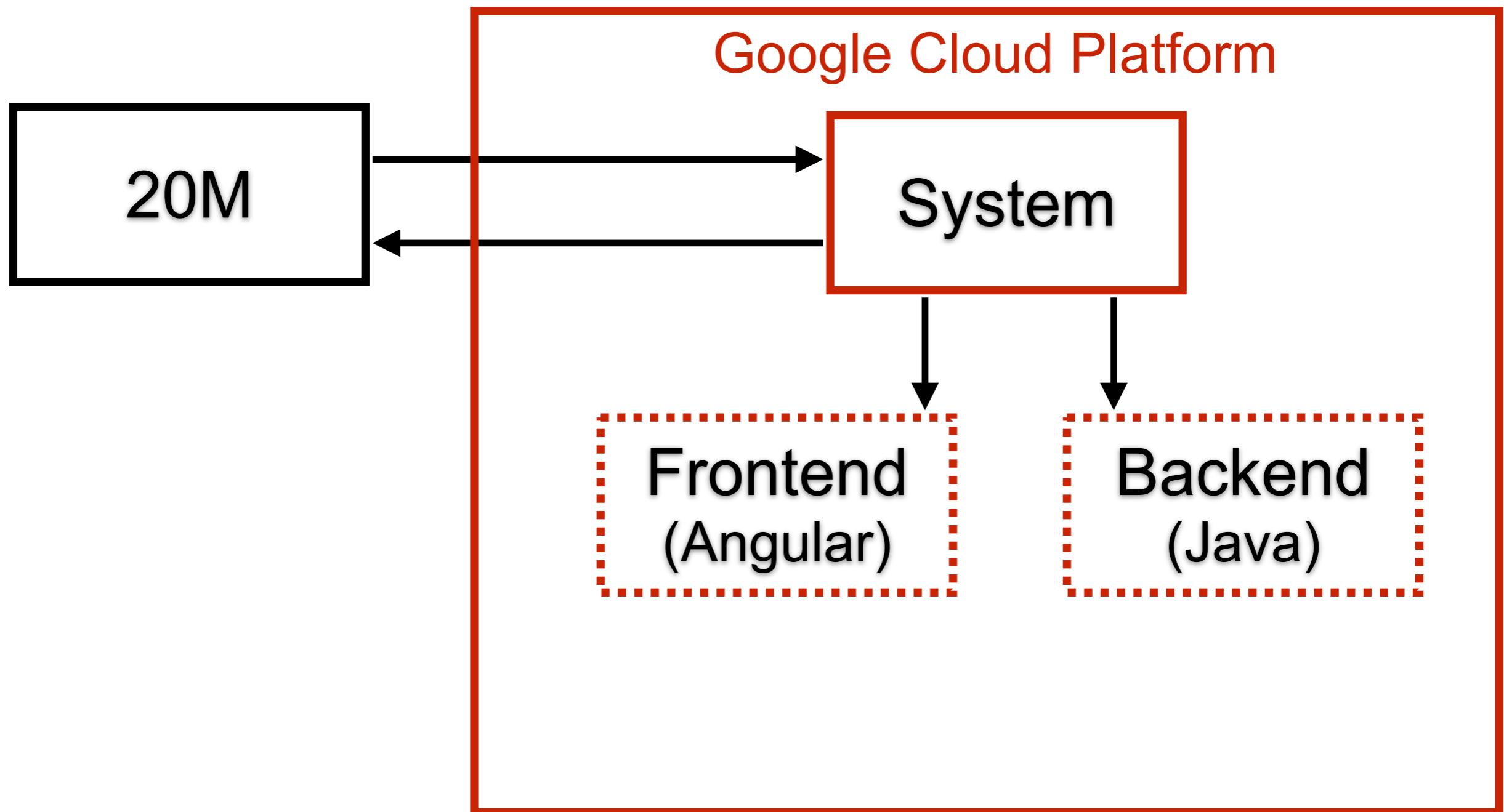
Architecture ?

20M connections waiting to register....



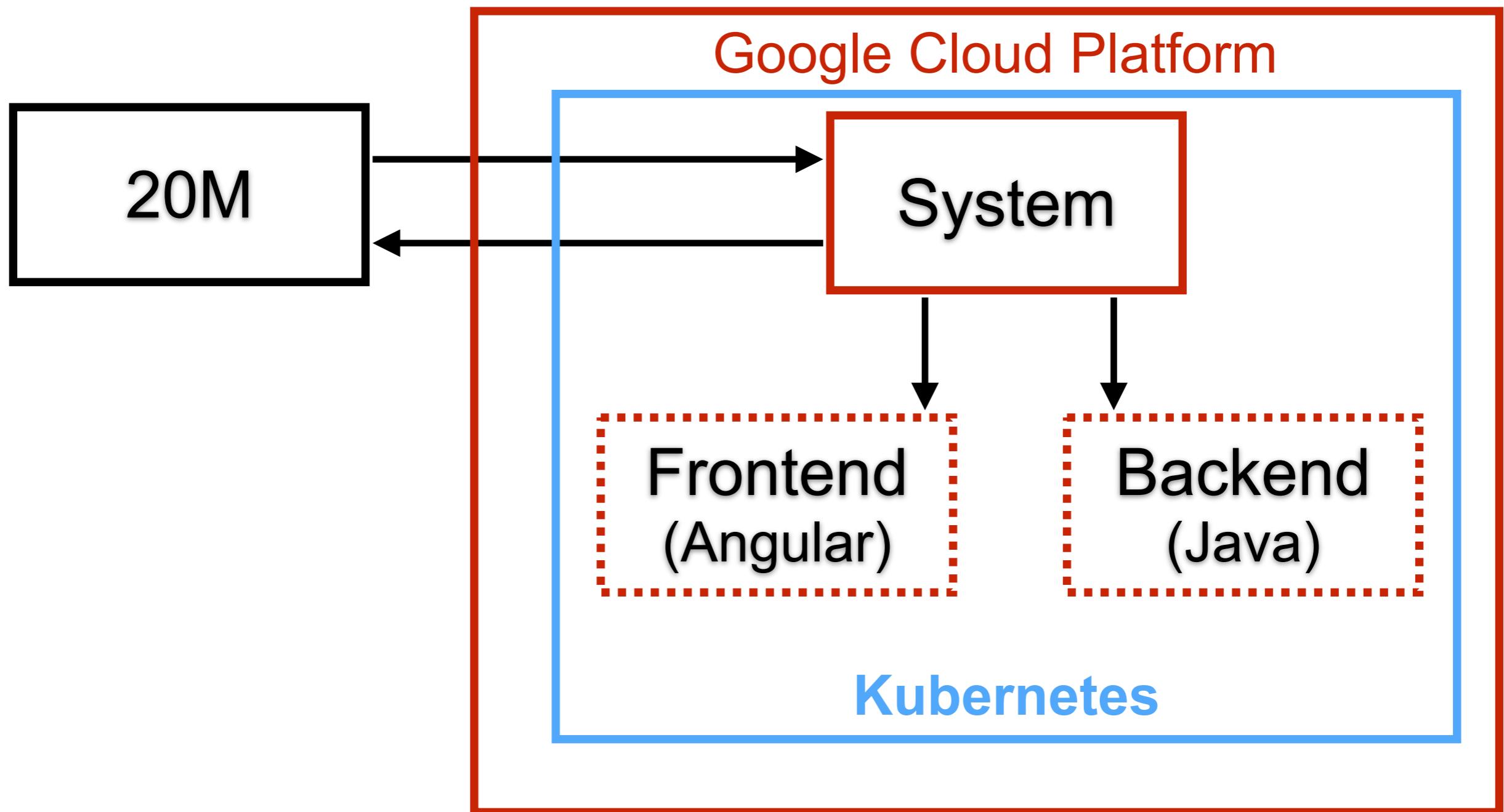
Architecture ?

Infrastructure at Google Cloud platform



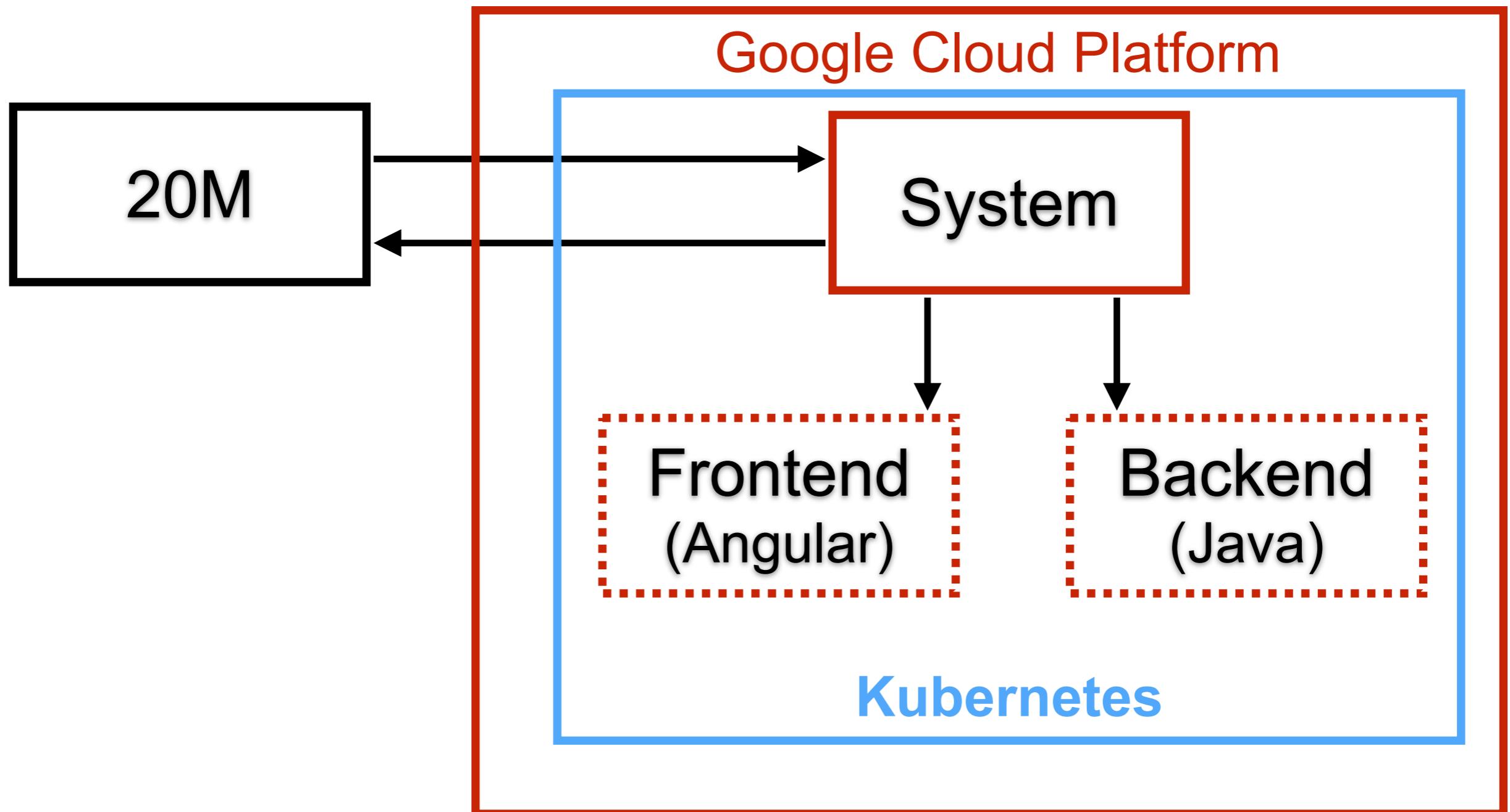
Architecture ?

Manage with Kubernetes (K8S)



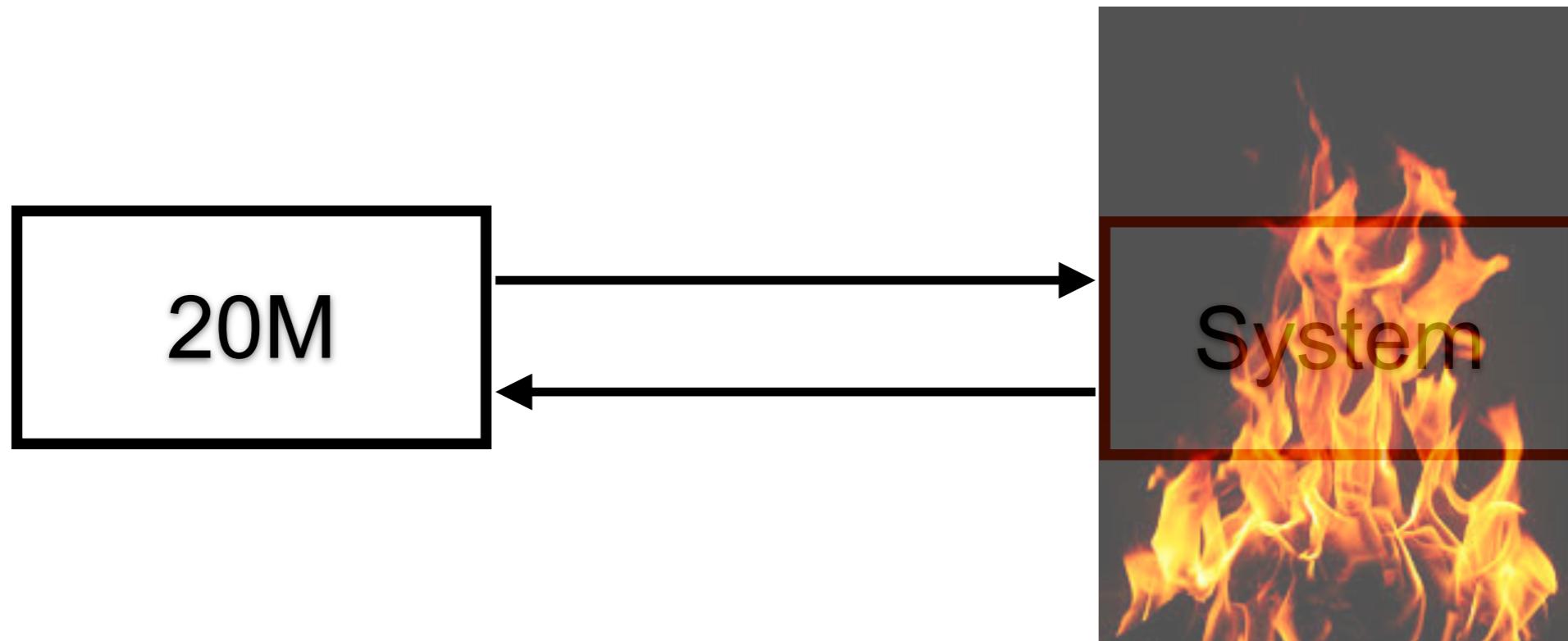
Architecture ?

Kubernetes and Horizontal Pods Autoscaling (HPA)

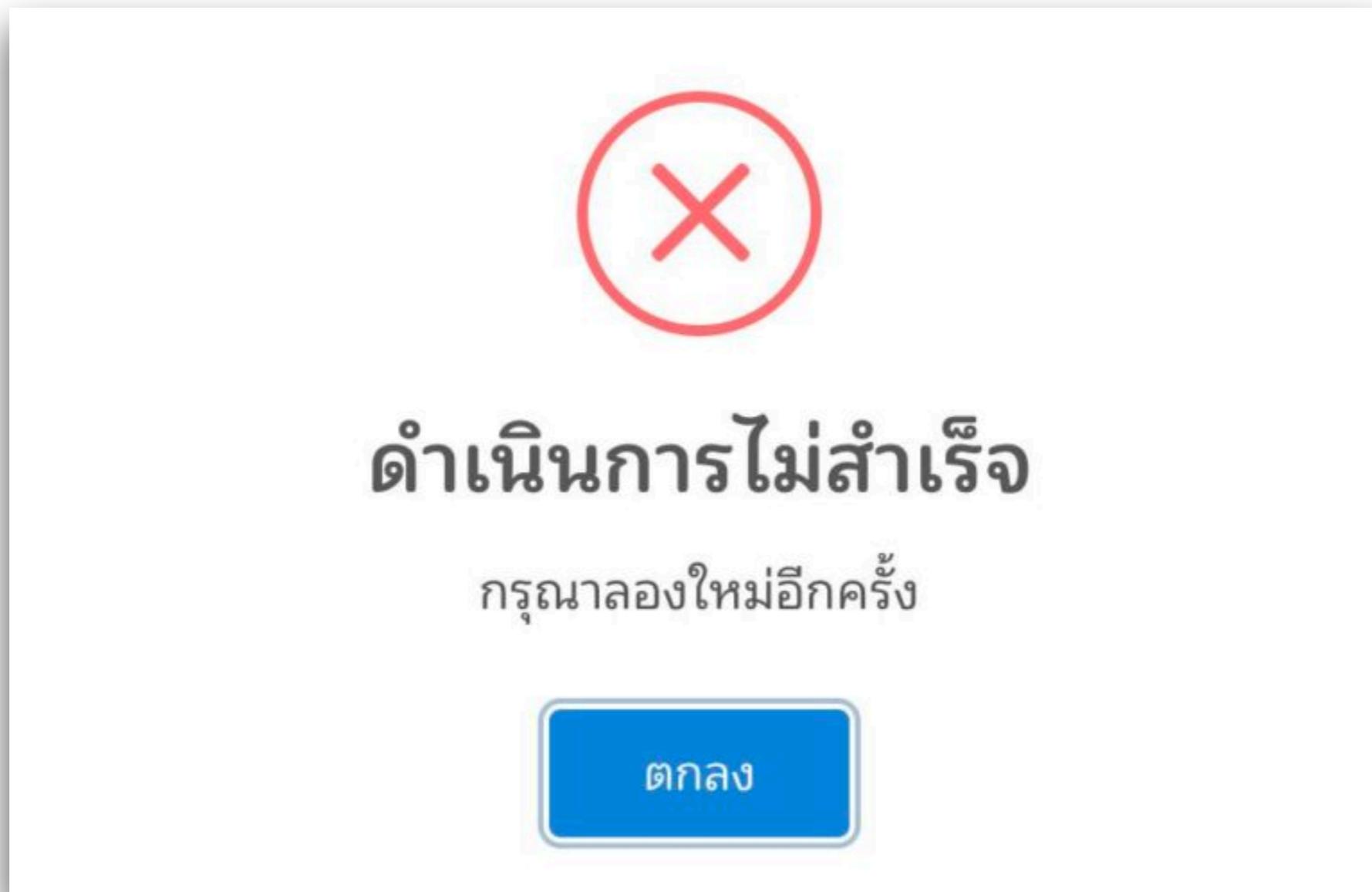


10 minutes after open system !!

20M connections come to system !!



10 minutes after open system !!



10 minutes after open system !!

This XML file does not appear to

```
<Error>
  <Code>NoSuchKey</Code>
  <Message>The specified key do
</Error>
```



10 minutes after open system !!

Error: Server Error

The server encountered a temporary error and could not complete your request.

Please try again in 30 seconds.



Restart ?



Reset all system



Discuss about business process !!



Fixed



Deploy service-by-service



Monitoring system





Manual scaling



Repeat ...



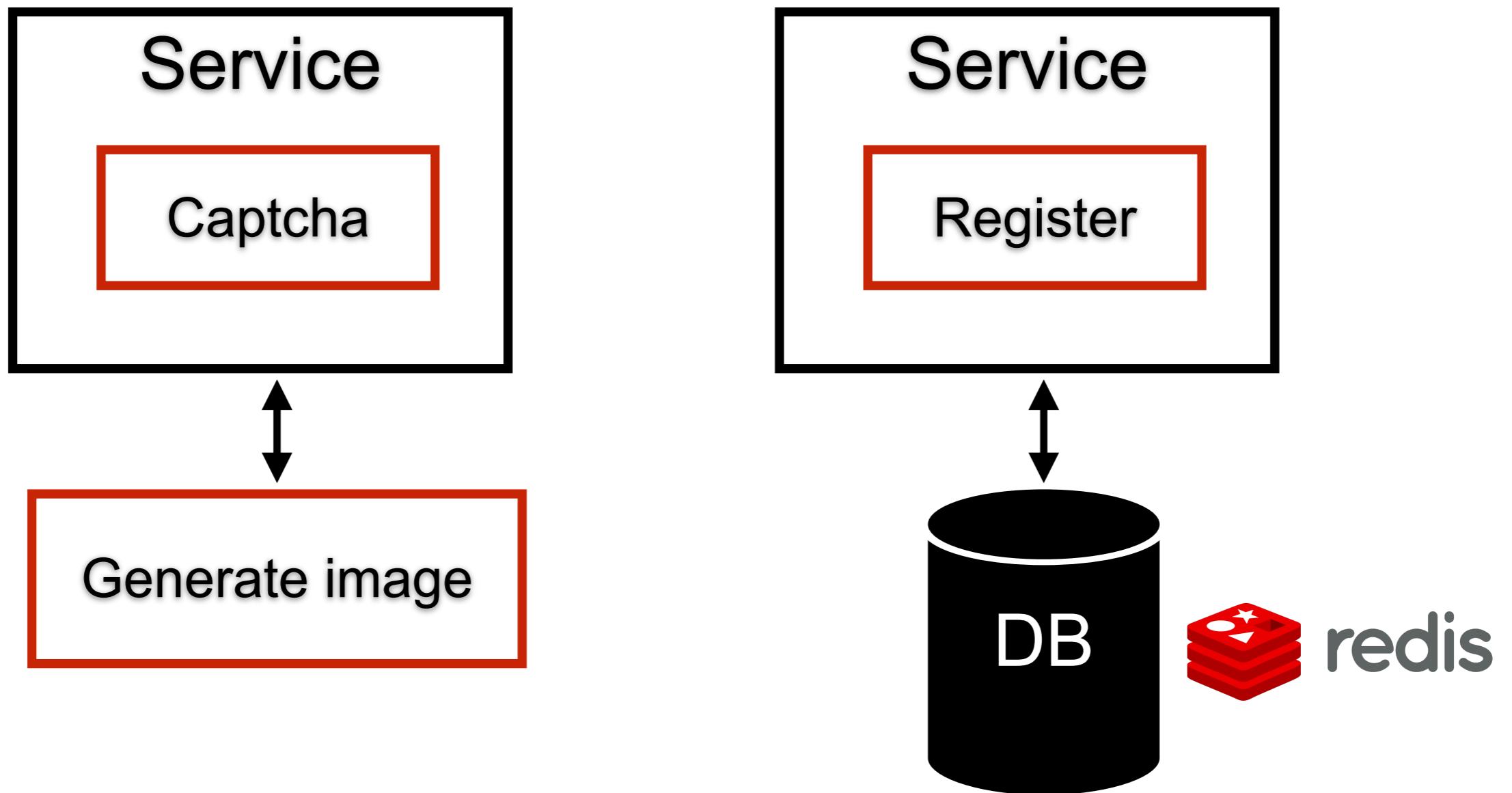
Monitoring system
Deploy service-by-service

Monitoring system
Manual scaling

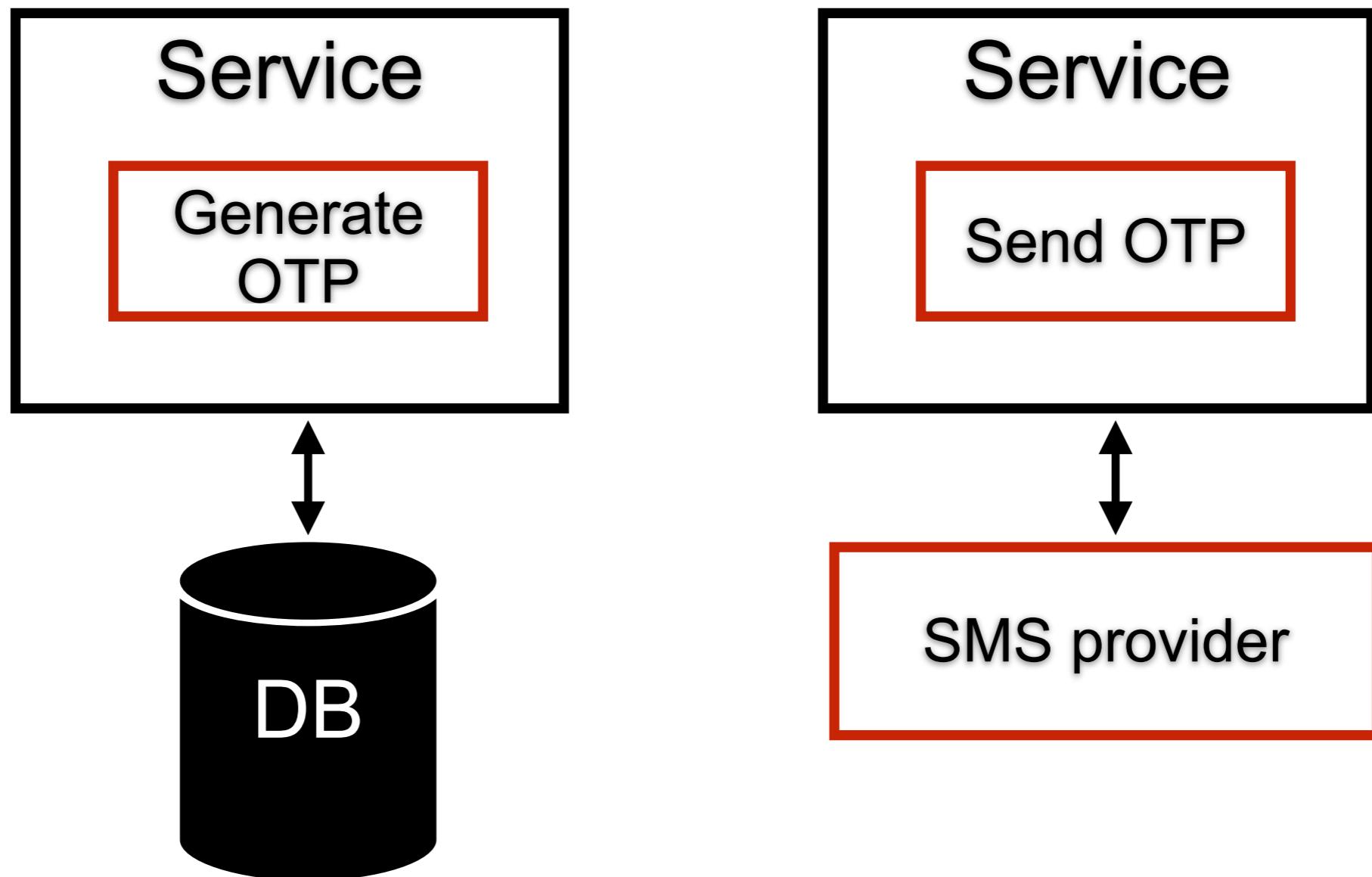
Repeat ...



Split to new service



Split to new service



Tools



How to choose the testing tool ?



Performance testing tool !!

Scripting module

Test management module

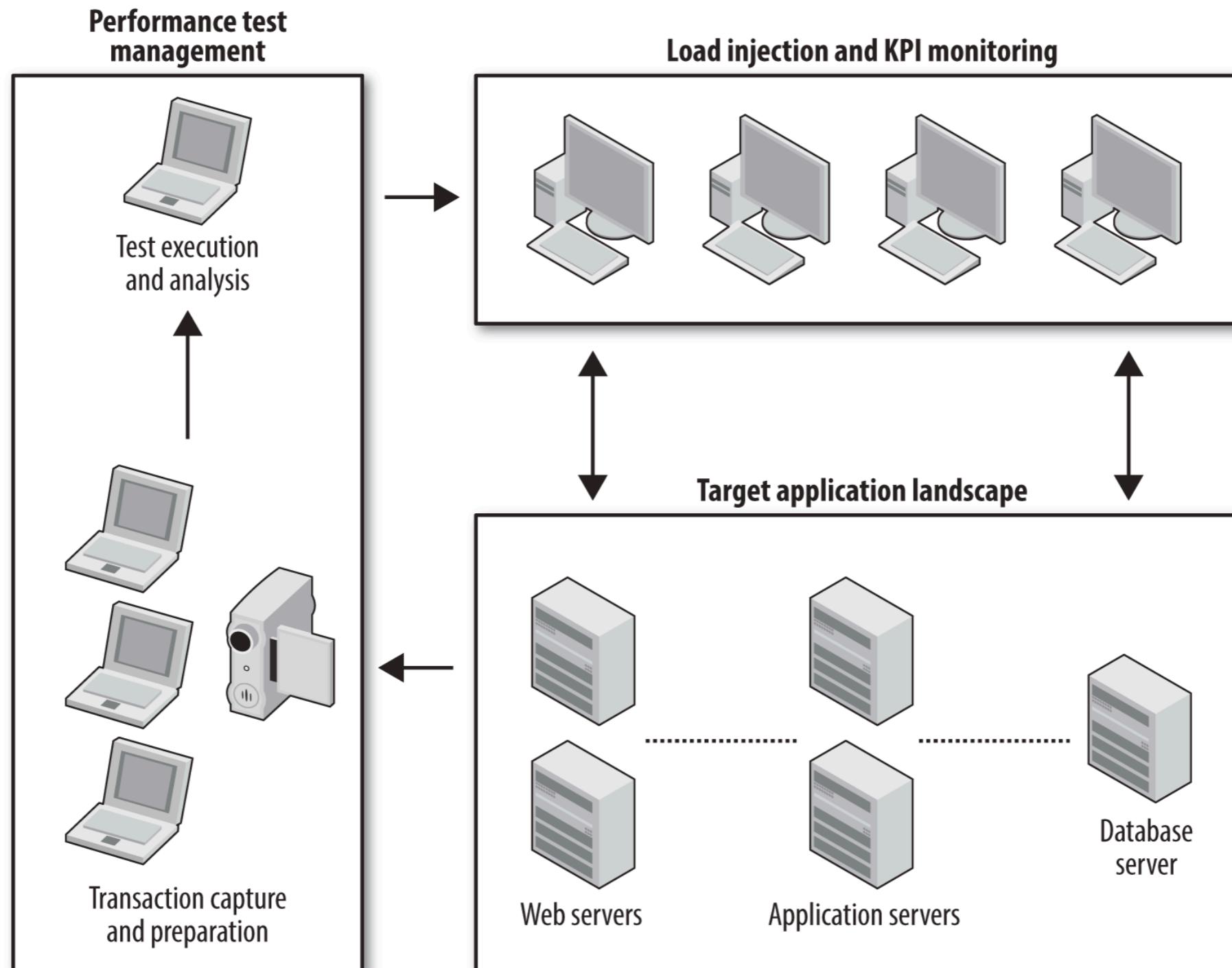
Load injectors

Analysis module

Optional module



Performance testing tool !!



How to choose ?

Protocol support

Licence model

Scripting effort

Solution vs load testing tool

In-house vs outsource

Software-as-a-Service (SaaS)



Try to PoC (Proof of Concept)

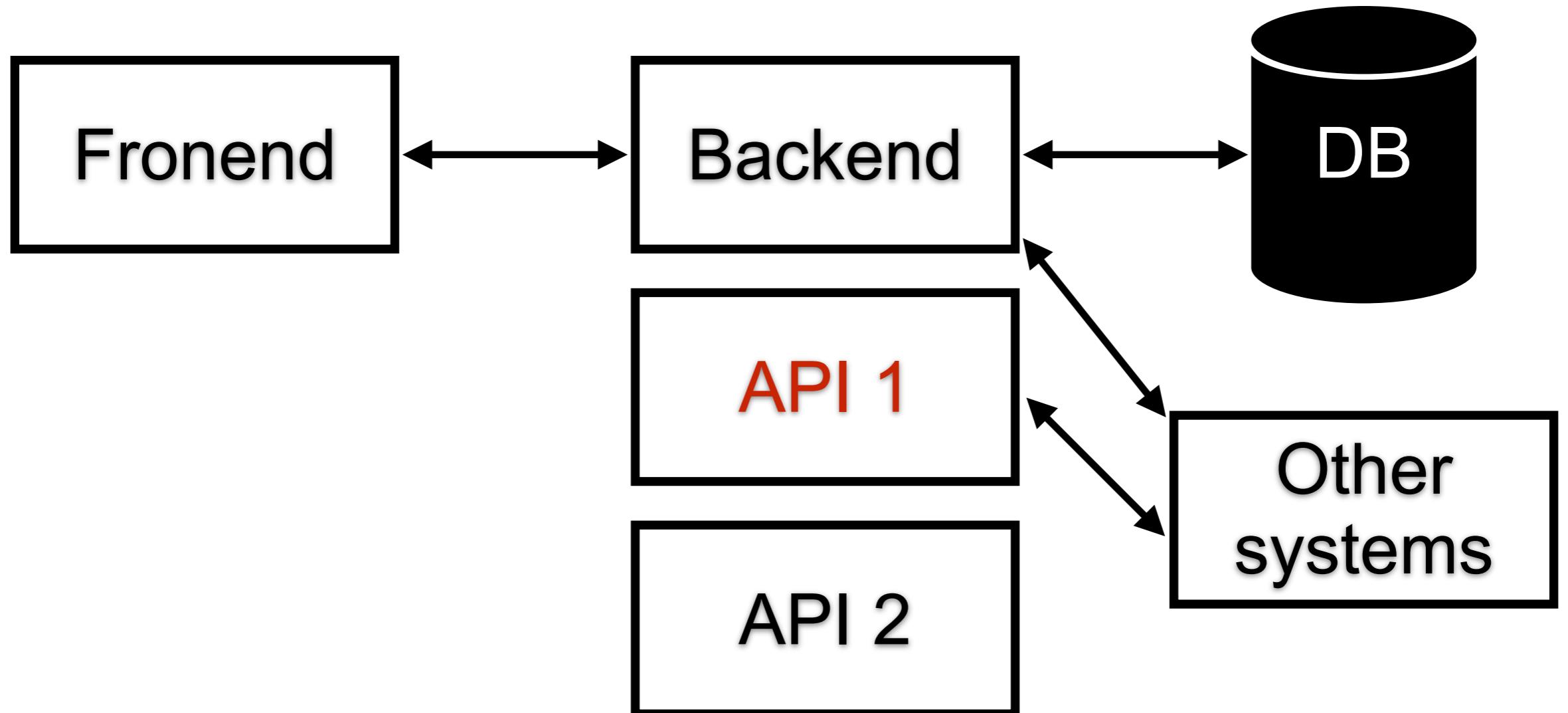


Effective performance testing

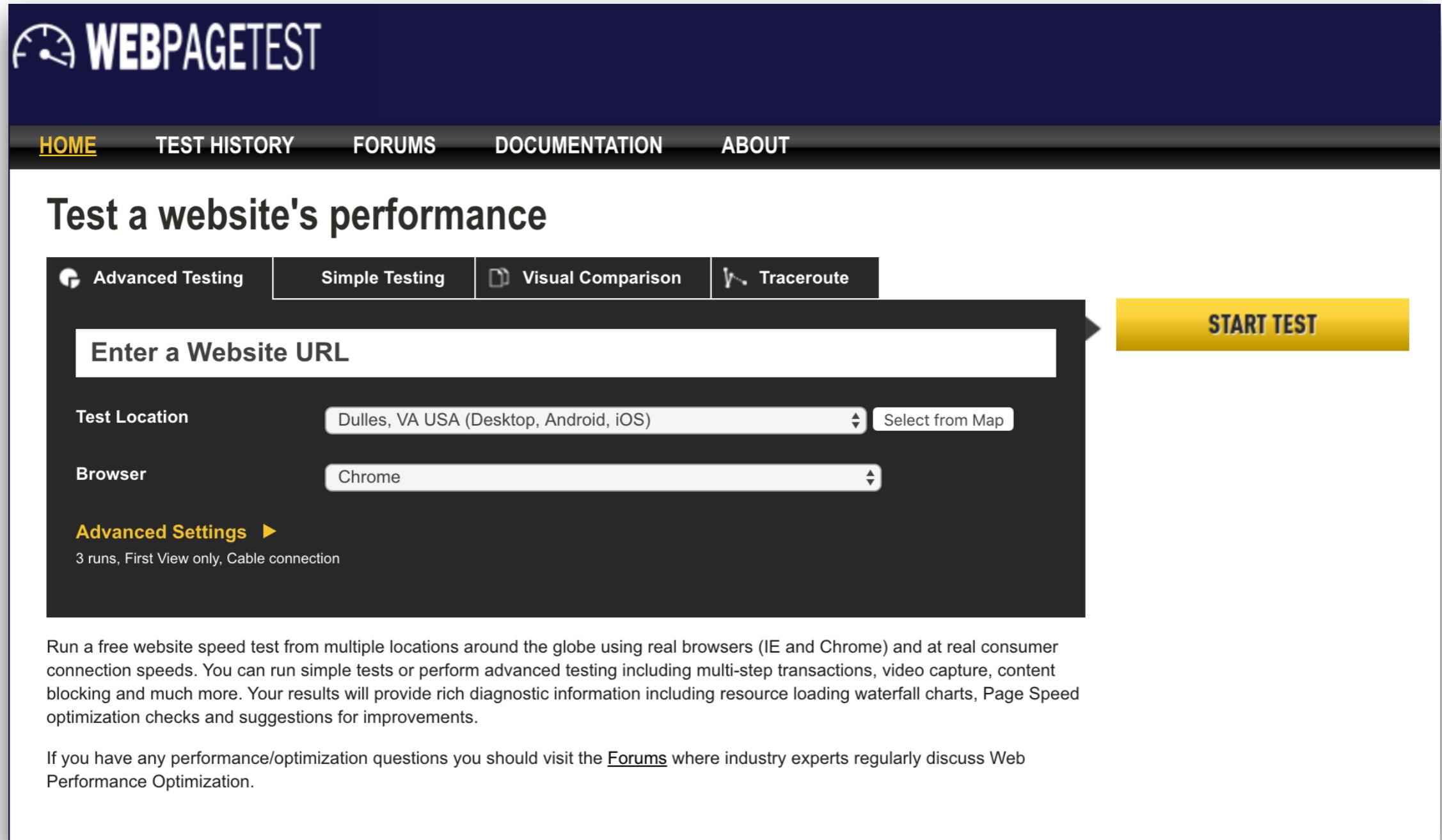
Must use Automation



Start with goals ?



How responsive a webpage ?



The screenshot shows the homepage of WebPageTest. At the top, there's a dark blue header with the "WEBPAGETEST" logo. Below it is a navigation bar with links for "HOME", "TEST HISTORY", "FORUMS", "DOCUMENTATION", and "ABOUT". The main content area has a dark background. It features a large input field labeled "Enter a Website URL" with a yellow "START TEST" button to its right. Above this input field are four tabs: "Advanced Testing" (selected), "Simple Testing", "Visual Comparison", and "Traceroute". Below the input field are dropdown menus for "Test Location" (set to "Dulles, VA USA (Desktop, Android, iOS)") and "Browser" (set to "Chrome"). There's also a "Select from Map" option next to the location dropdown. A "Advanced Settings" link with a dropdown arrow is present, showing "3 runs, First View only, Cable connection". Below the form, a descriptive text explains the service: "Run a free website speed test from multiple locations around the globe using real browsers (IE and Chrome) and at real consumer connection speeds. You can run simple tests or perform advanced testing including multi-step transactions, video capture, content blocking and much more. Your results will provide rich diagnostic information including resource loading waterfall charts, Page Speed optimization checks and suggestions for improvements." A note at the bottom encourages users to visit the "Forums" for performance optimization discussions.

<https://www.webpagetest.org/>



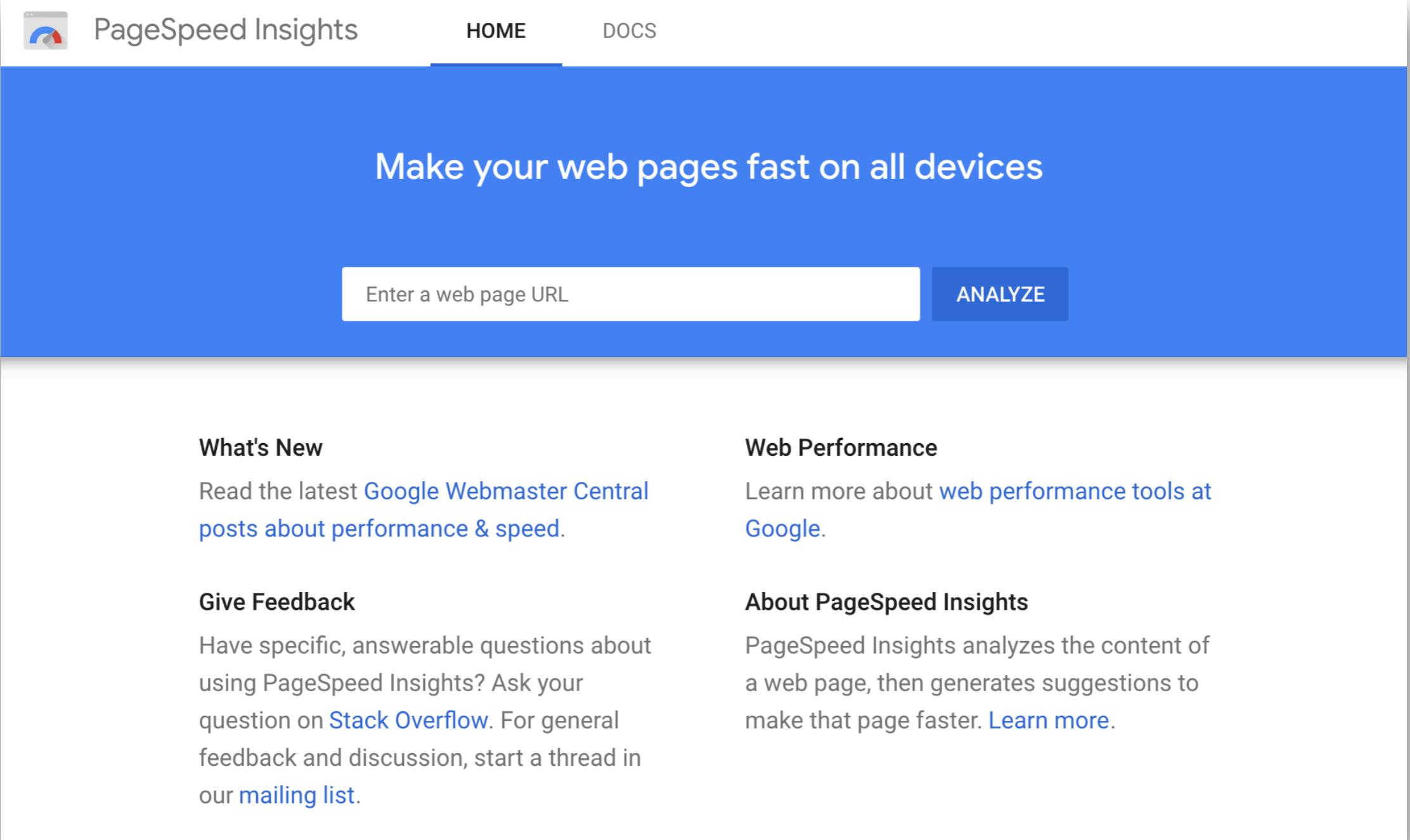
How responsive a webpage ?



<https://www.webpagetest.org/>



How responsive a webpage ?



The screenshot shows the homepage of Google PageSpeed Insights. At the top, there's a navigation bar with a logo, the text "PageSpeed Insights", and links for "HOME" and "DOCS". Below this is a large blue header with the text "Make your web pages fast on all devices". In the center of the header is a search bar with the placeholder "Enter a web page URL" and a blue "ANALYZE" button. The main content area is white and contains several sections: "What's New" (link to Google Webmaster Central), "Web Performance" (link to web performance tools at Google), "Give Feedback" (link to Stack Overflow and mailing list), and "About PageSpeed Insights" (description of the tool and link to learn more).

<https://developers.google.com/speed>



Non-code performance issue

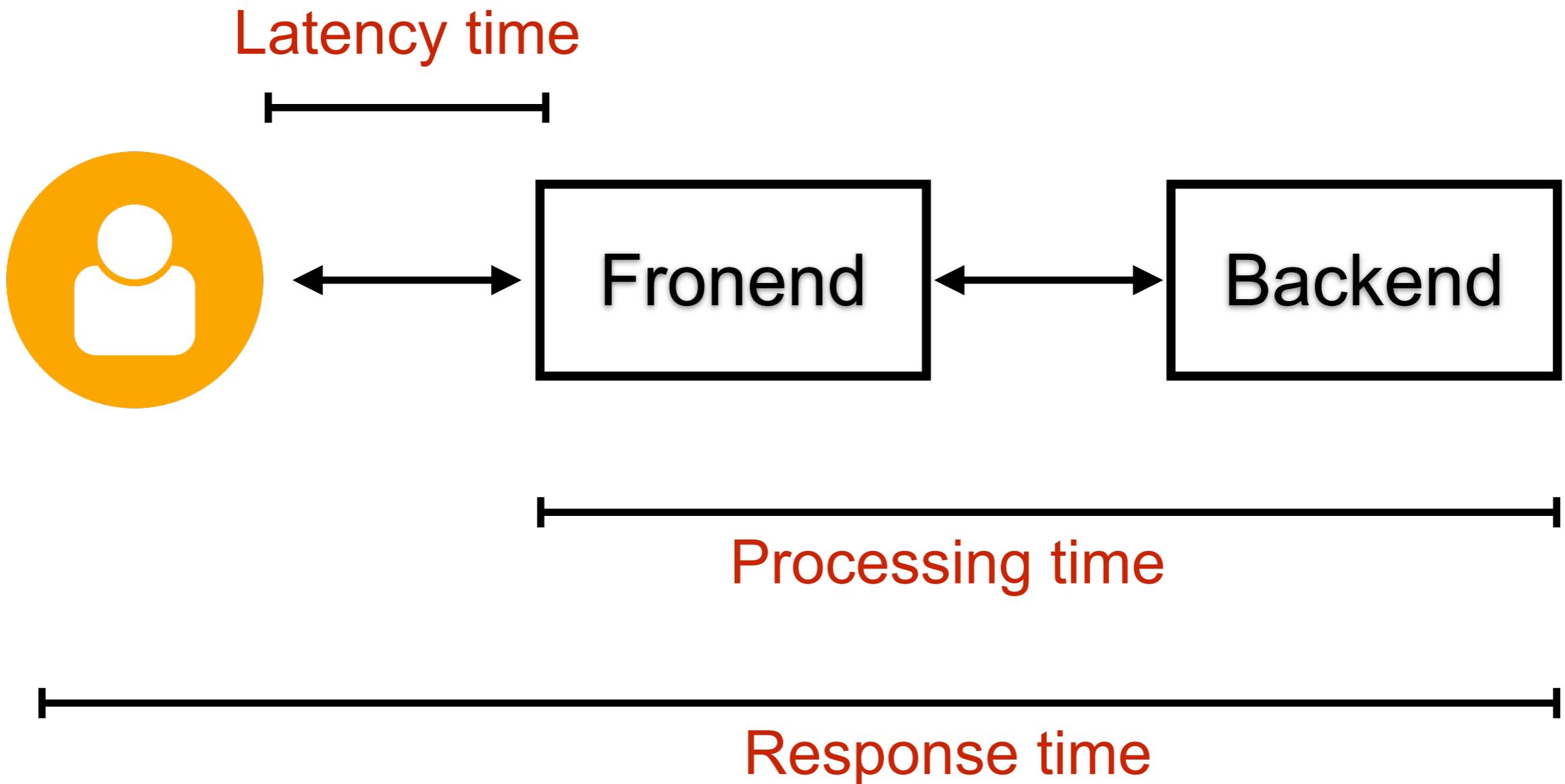
Latency

Network speed

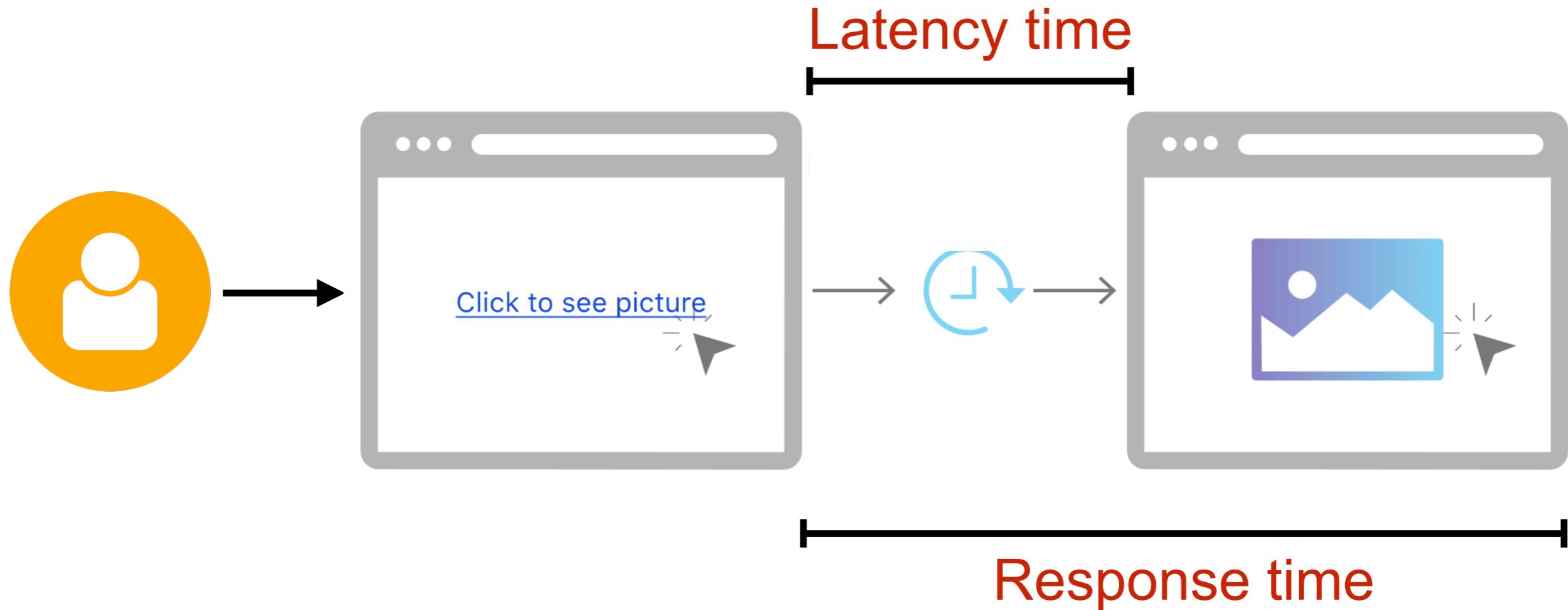
Human perception



Latency ?



Latency vs Response time



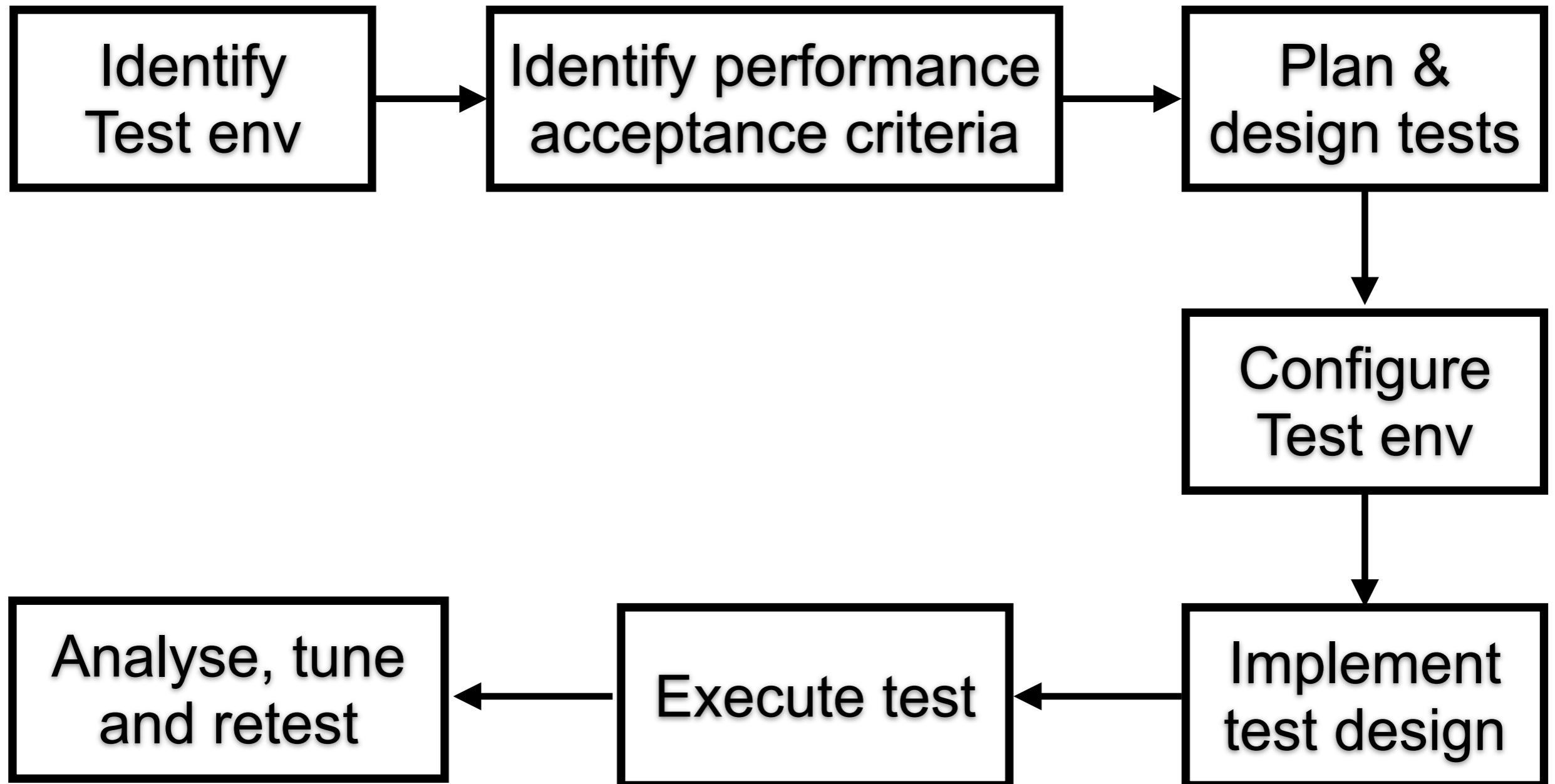
Response time = Latency time + Processing time



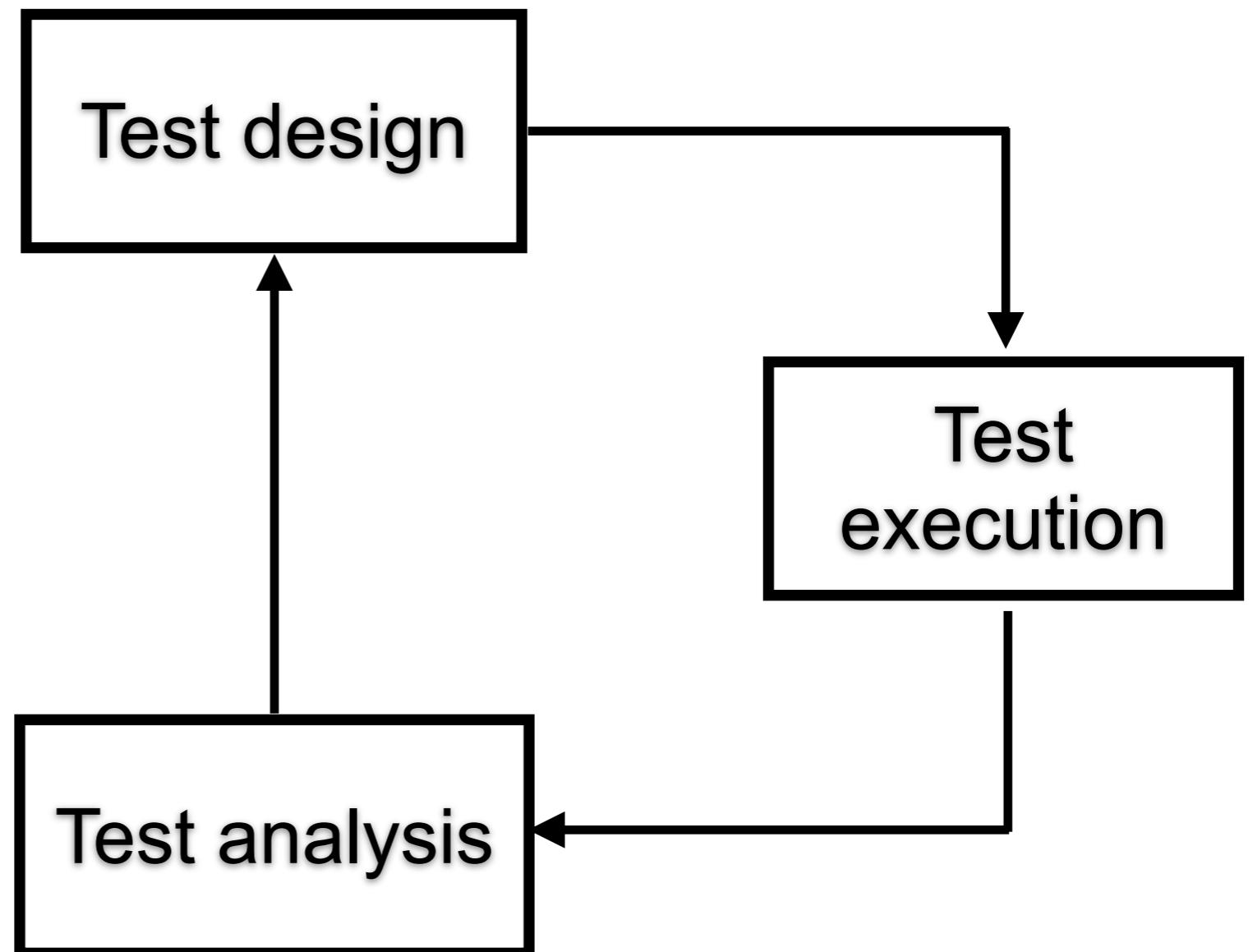
Performance testing process



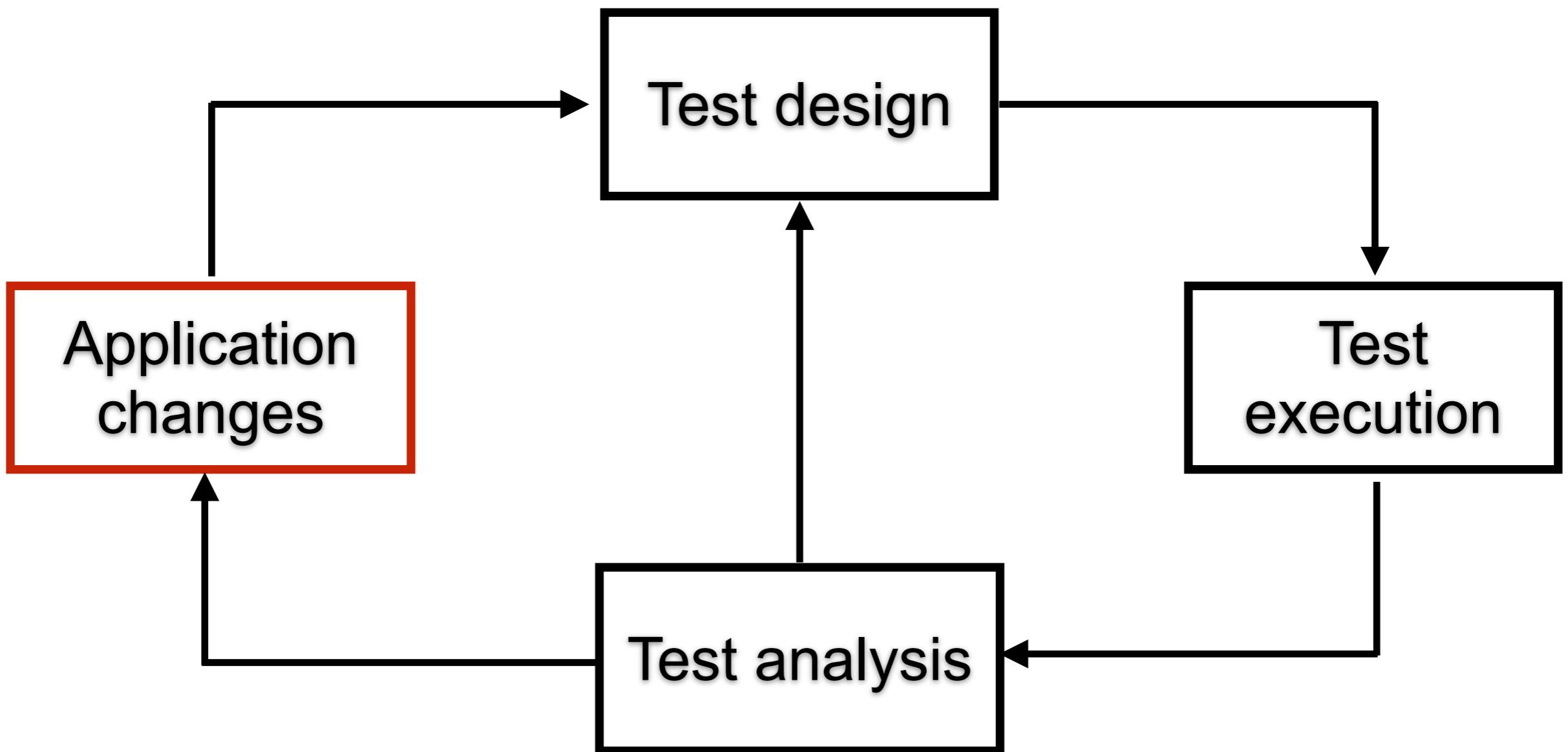
Process



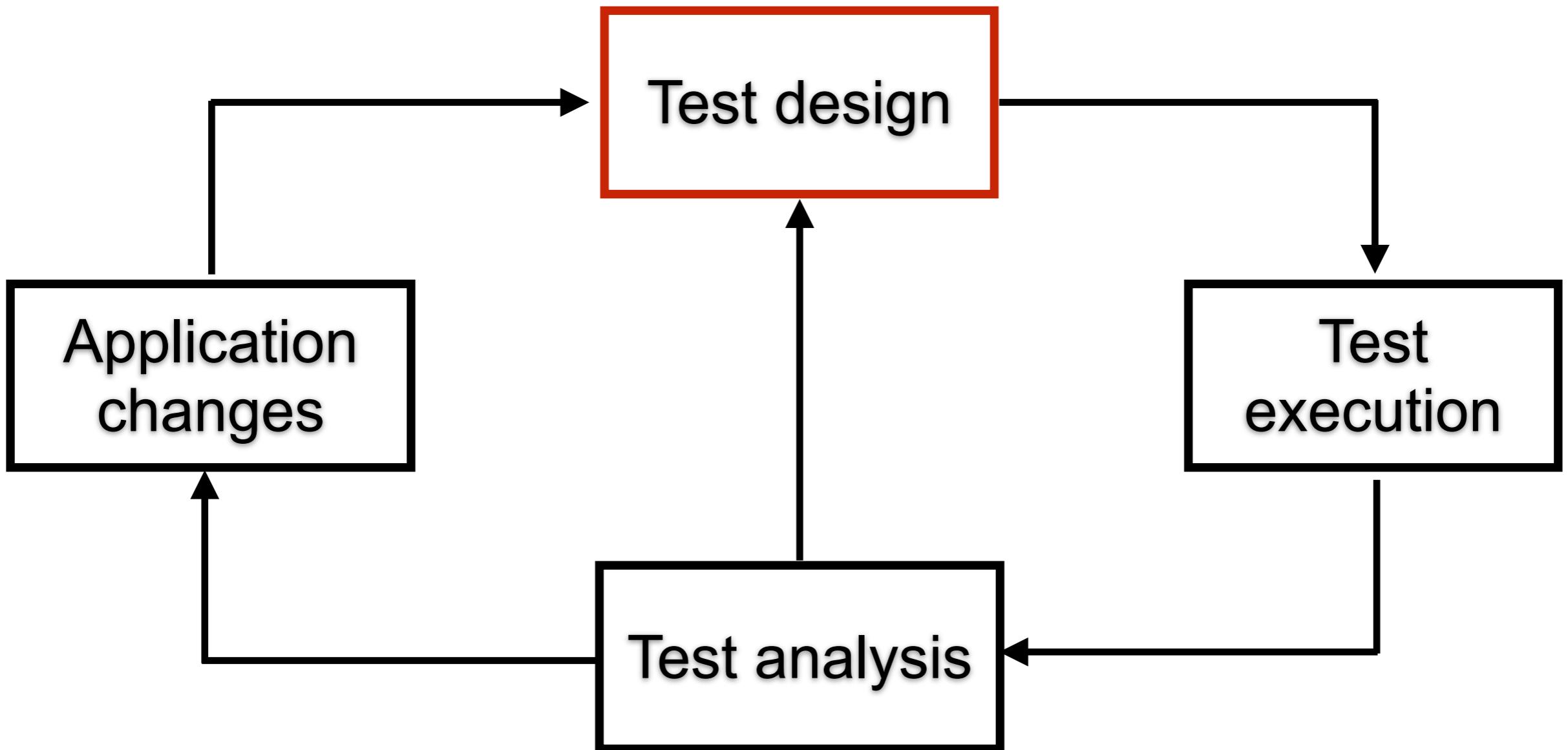
Testing process



Testing process



Start with design



What to measure ?



Measure timing

What times are you checking ?

Help you understand the user's experience ?

What questions you are trying to answer ?

Every time when you need ?



Measure stability

Errors ?

Understand and compare error rates
Find in logs (app/server logs)



Measure stability

Server-side ?

CPU, memory, I/O usage

Queued requests and thread utilization



Measure analytics

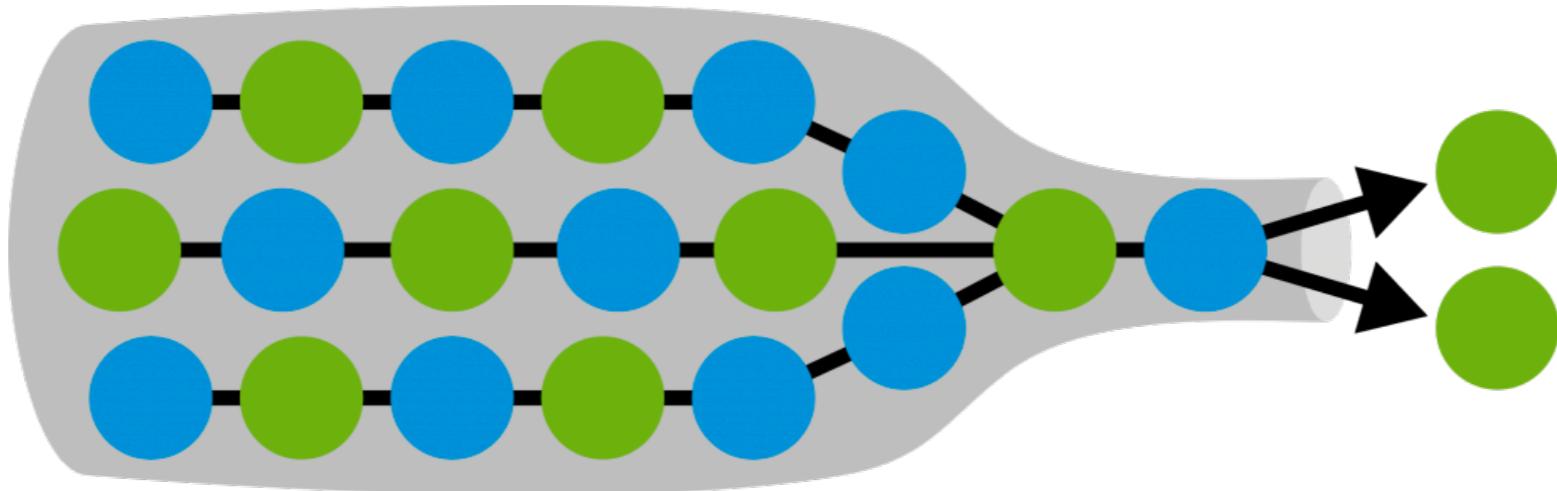
Instrumentation

Consider performance metrics on production
Discover popular actions

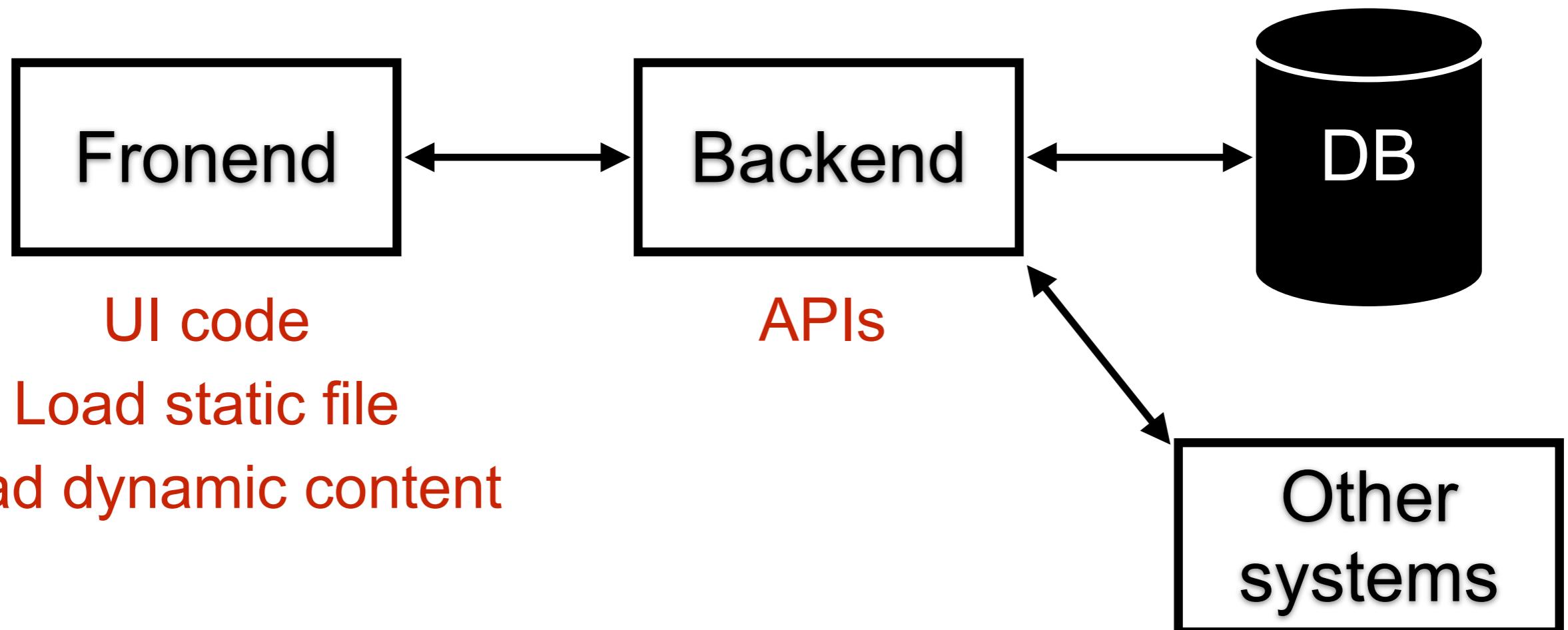
*“Understand performance impact
that user care about”*



Understand bottleneck



Find valuable problems



Potential bottlenecks

CPU, memory, file I/O
Third-party application
Network latency
Design issue



Monitoring results



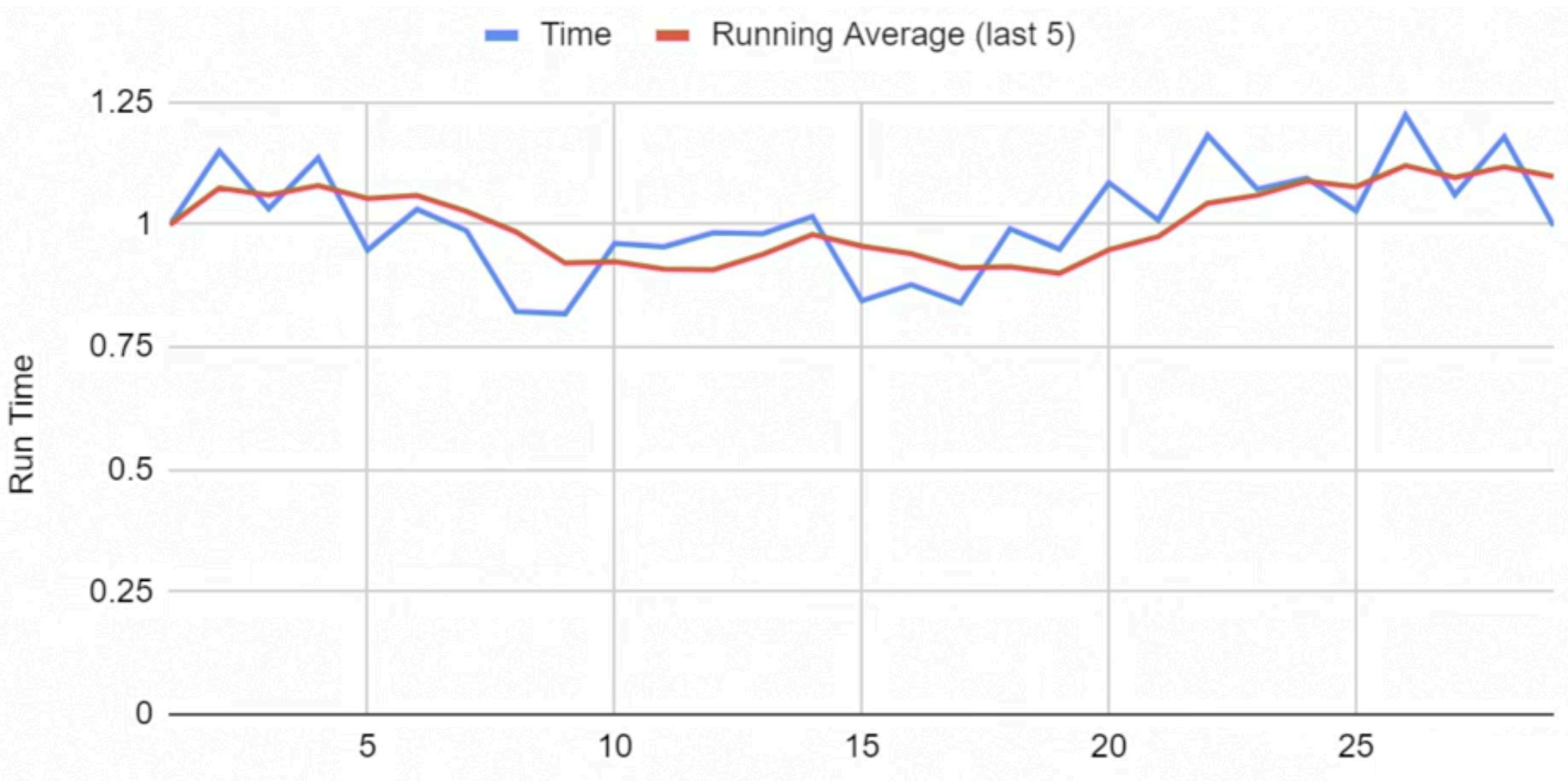
Challenge !!

How do we **know** when system have issues or slowdown ?

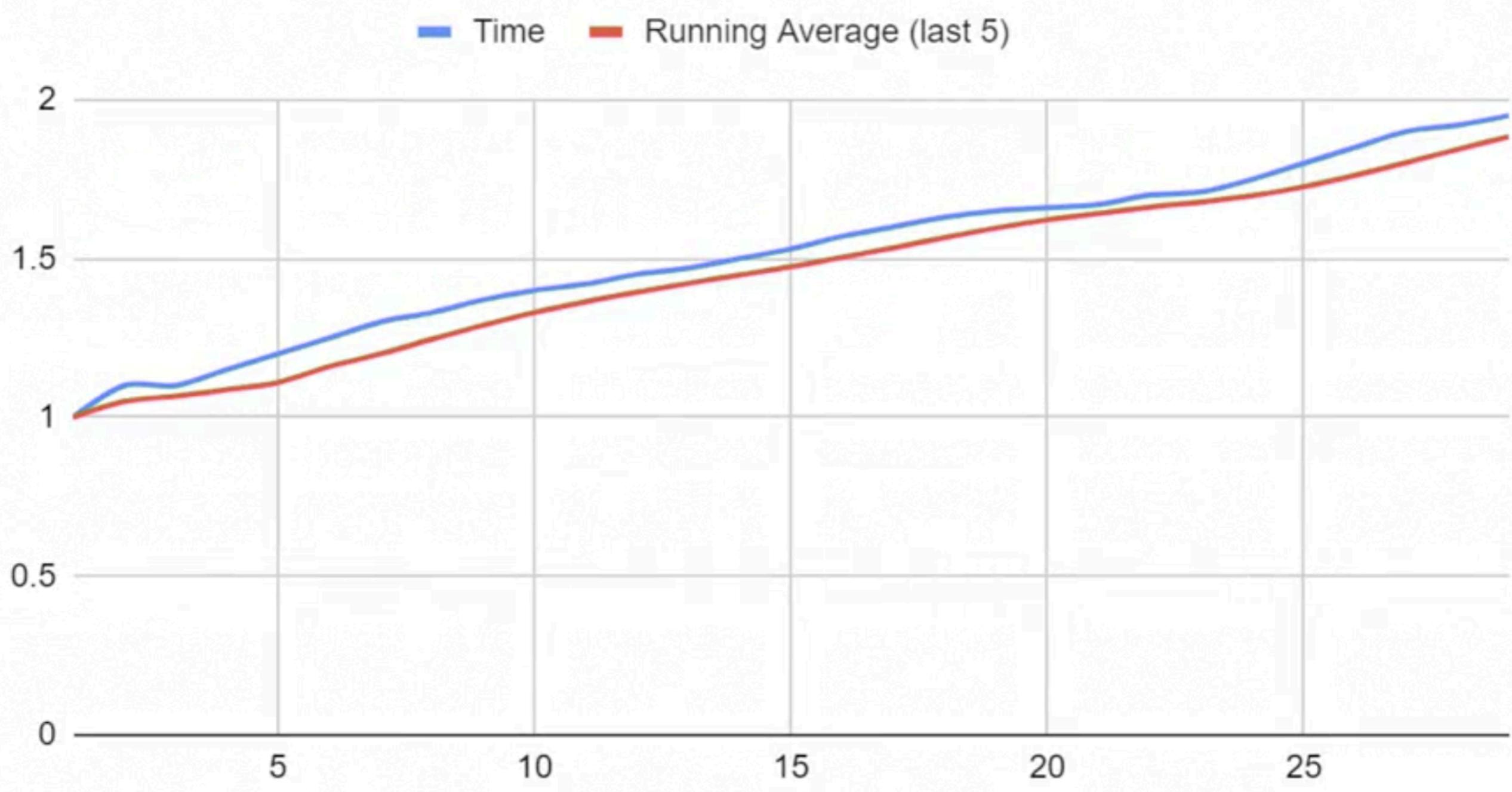
Performance results have a lot of **variation**



Good or bad ?



Good or bad ?



Good monitoring

Check more than one performance indicator

- Have alerts be actionable

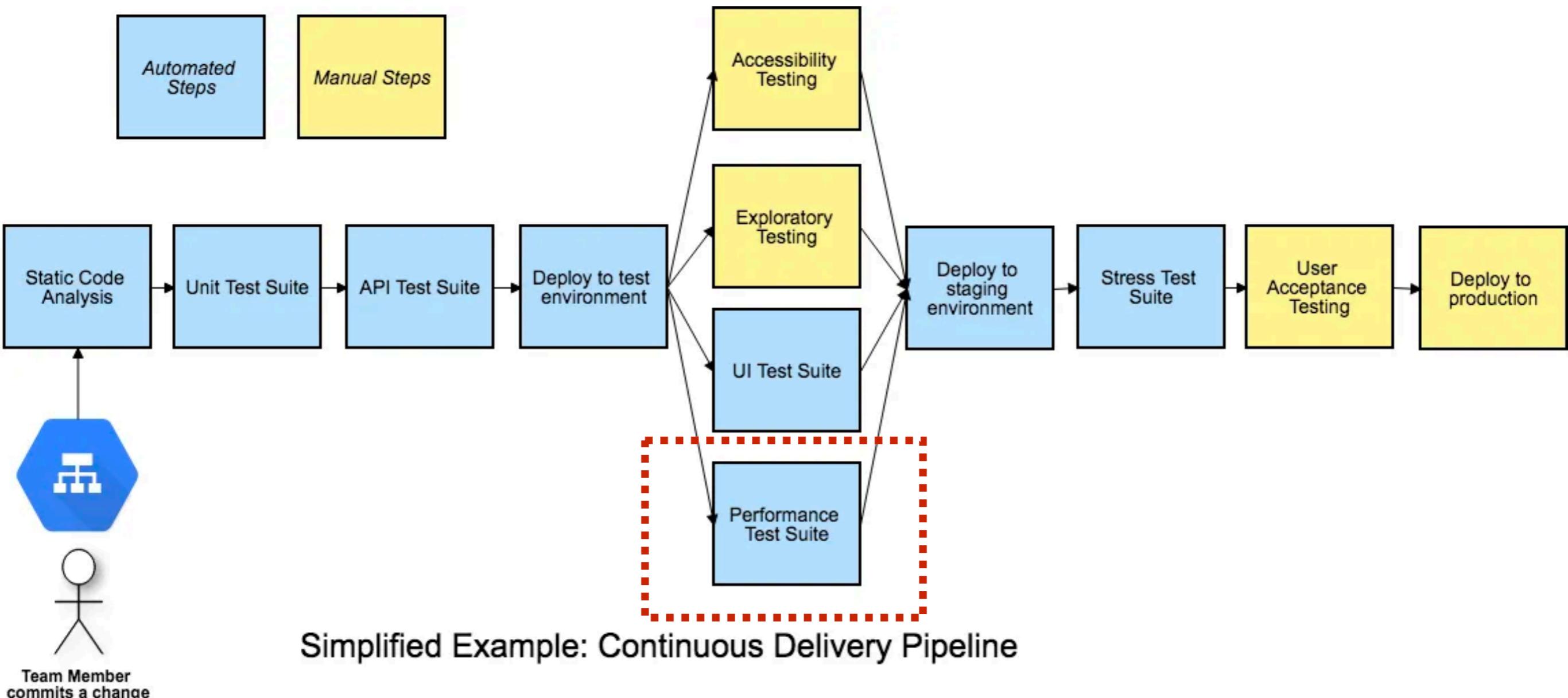
- Account for test variation



Pipeline



Pipeline



Tools



Performance test tools

ApacheBench



Drill



Siege



Wrk

<https://k6.io/blog/comparing-best-open-source-load-testing-tools>



Goad

Goad is an AWS Lambda powered, highly distributed, load testing tool built in Go for the [2016 Gopher Gala](#).

Go + Load ⇒ Goad

 Downloads for macOS, Linux, and Windows

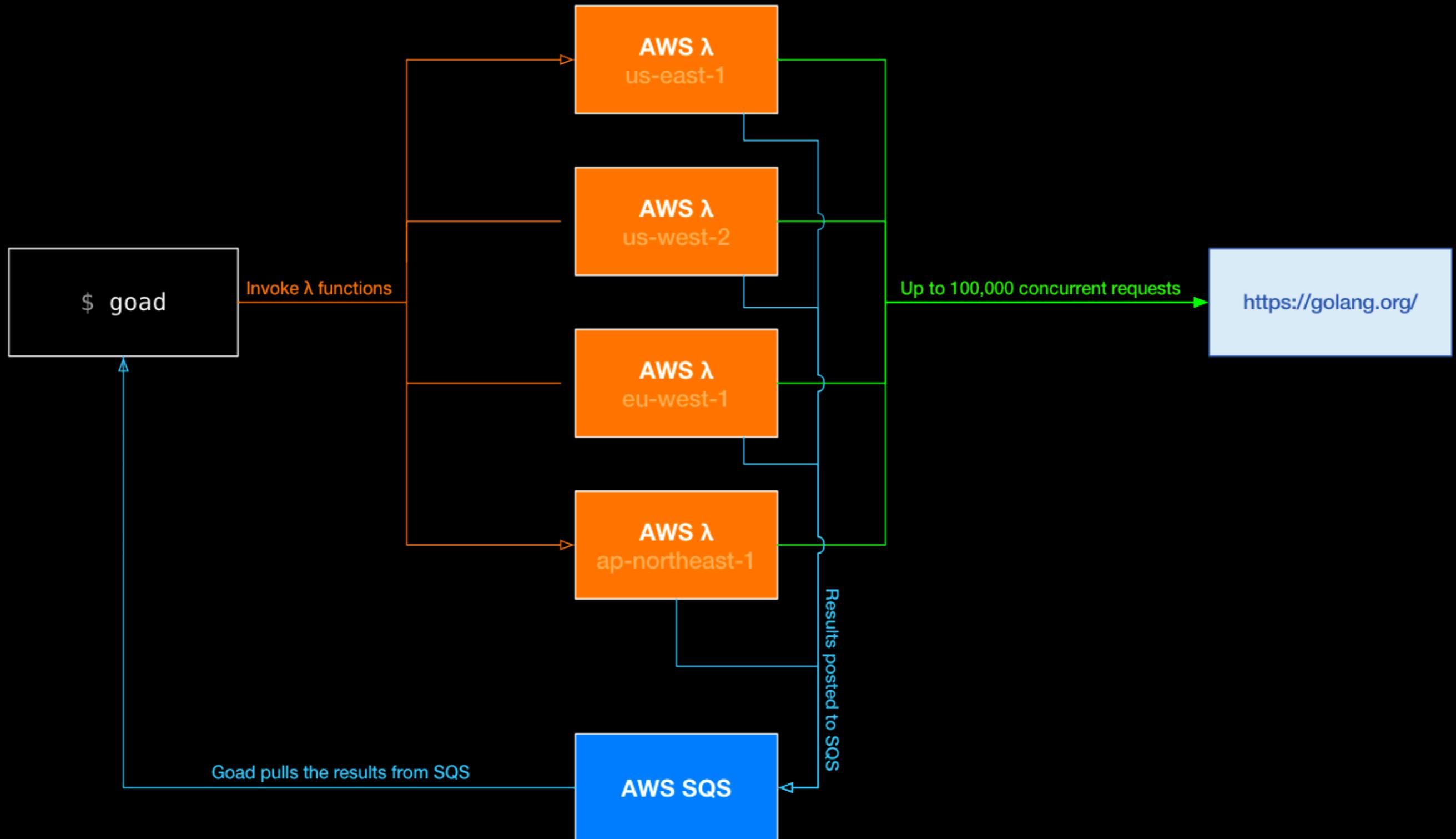
or get the source code from GitHub



G O A D

<https://goad.io/>

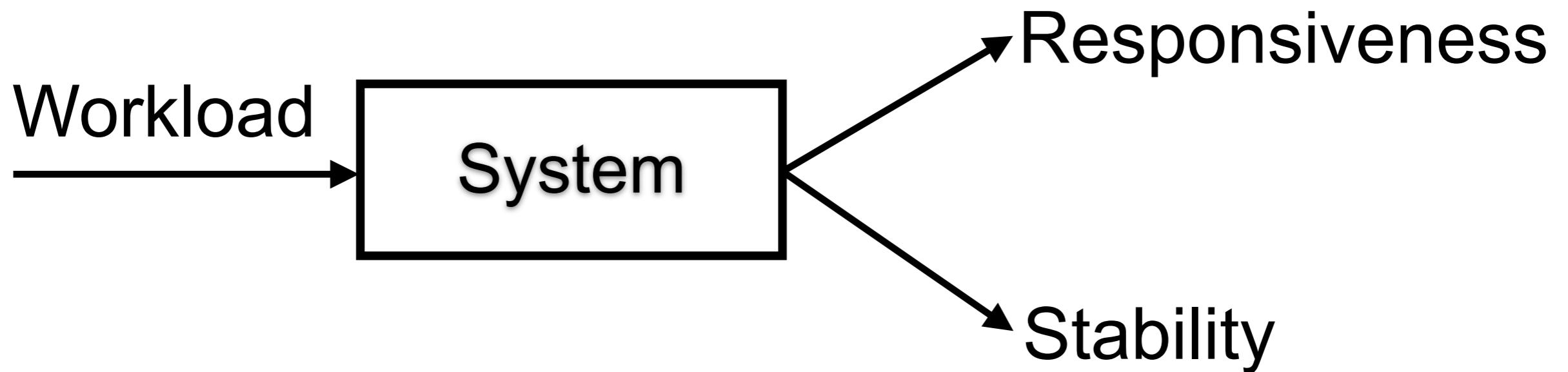




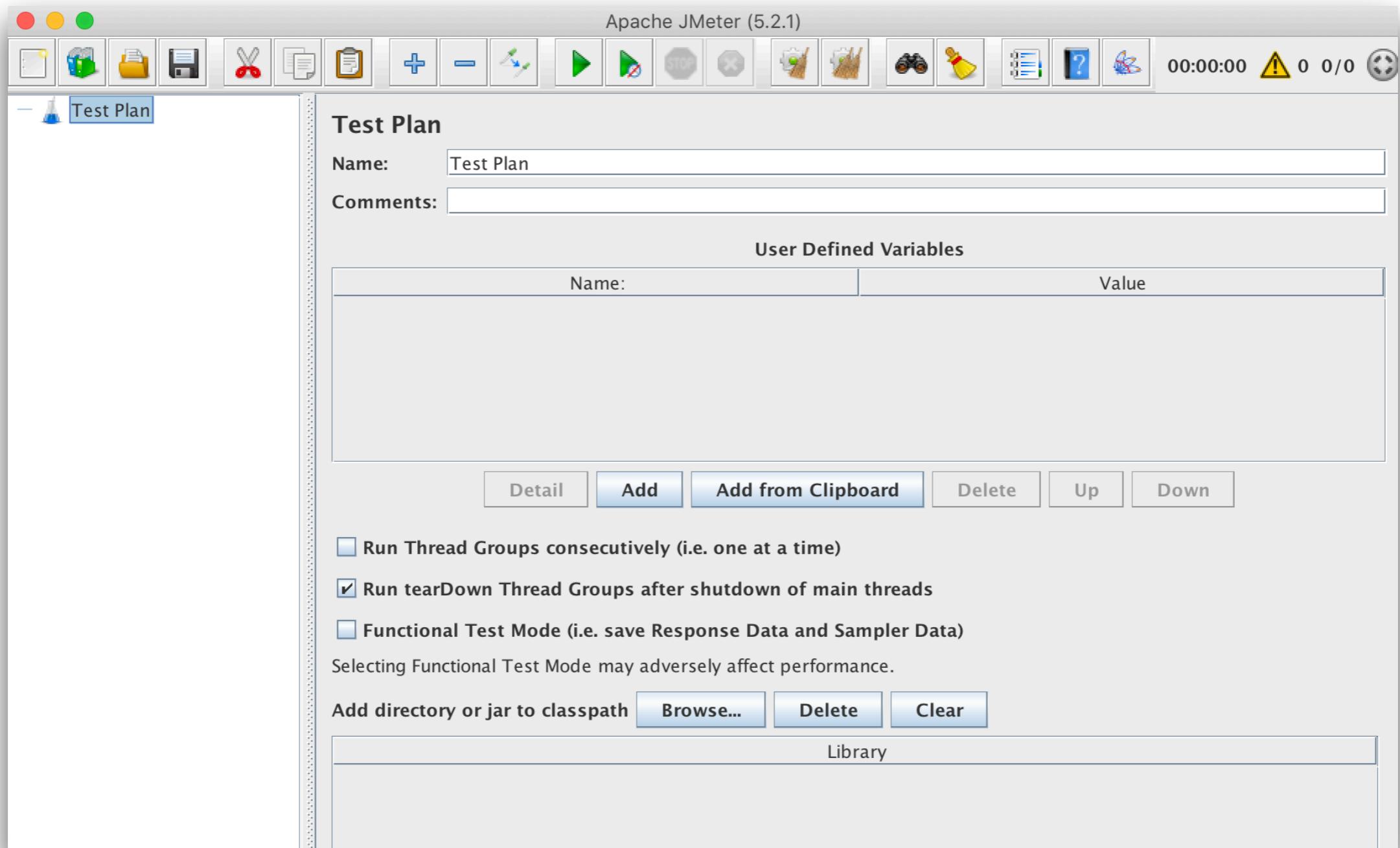
JMeter workshop



Create workload with JMeter



JMeter



<https://jmeter.apache.org/>



JDK (8, 11, 12)



OpenJDK

<https://openjdk.java.net/>



JMeter plugins

JMeter Plugins



jmeter-plugins.org

Every load test needs some sexy features!



Online
JMX
Editor

[Install](#)

[Browse Plugins](#)

[Documentation](#)

[Usage Statistics](#)

[Support Forums](#)

[JMX Editor](#)



[Star](#)

Custom Plugins for Apache JMeter™



This project is an independent set of plugins for [Apache JMeter](#), the popular Open-Source load and performance testing tool.

This catalogue lists plugins available for use with Plugins Manager. If you're first time here, consider installing [Plugins Manager](#) into your JMeter.

Items found: 82

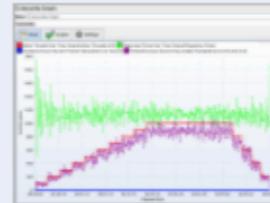
Search...

Composite Timeline Graph

Graph that is able to show several other graphs data in single place

[Download Versions: 2.0](#)

[Maven Artifact: kg.apc:jmeter-plugins-graphs-composite:2.0](#)



ID: jpgc-graphs-composite



Users: 10197

Dummy Sampler

Debugging sampler that just generates result as you specified. Very convenient to debug your complex scripts

[Download Versions: 0.1, 0.2, 0.3, 0.4](#)

[Maven Artifact: kg.apc:jmeter-plugins-dummy:0.4](#)



ID: jpgc-dummy



Users: 17628

<https://jmeter-plugins.org/>



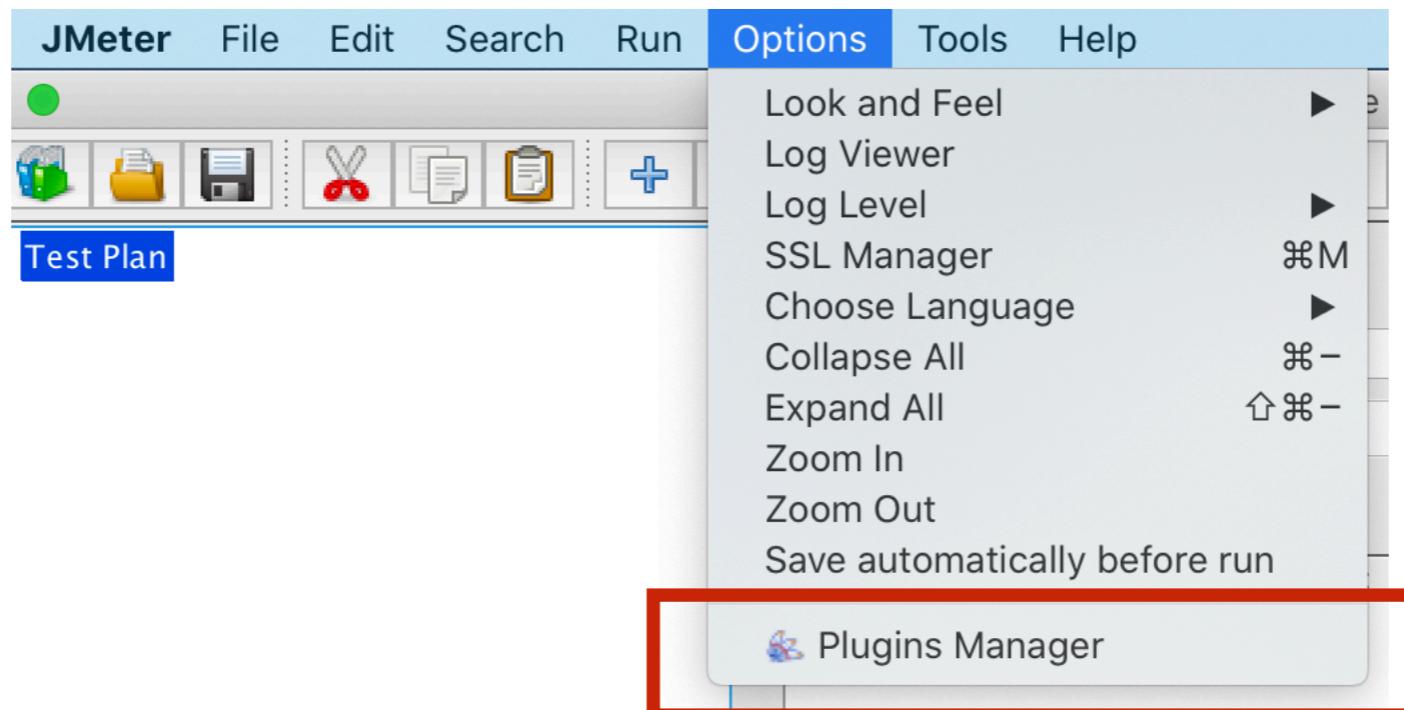
Performance testing

© 2017 - 2018 Siam Chamnkit Company Limited. All rights reserved.

JMeter plugins

Copy jar file to **\$JMETER_HOME/lib/ext**

Restart JMeter



<https://jmeter-plugins.org/>



Custom Thread Groups

JMeter Plugins Manager

Installed Plugins Available Plugins Upgrades

conc

Custom Thread Groups

Custom Thread Groups

Vendor: *JMeter-Plugins.org*

Adds new Thread Groups:

- [Stepping Thread Group](#)
- [Ultimate Thread Group](#)
- [Concurrency Thread Group](#)
- [Arrivals Thread Group](#)
- [Free-Form Arrivals Thread Group](#)

Documentation: <https://jmeter-plugins.org/wiki/ConcurrencyThreadGroup/>

What's new in version 2.9: Support JMeter's way of prefixing threads for distributed testing

Maven groupId: *kg.apc*, artifactId: *jmeter-plugins-casutg*, version: 2.9

Libraries: [jmeter-plugins-cmn-jmeter]

Version: 2.9

Review Changes

Install library: json-lib
Install library: jmeter-plugins-cmn-jmeter
Install library: cmdrunner
Install plugin: jpgc-casutg 2.9

Apply Changes and Restart JMeter



JMeter listener pack

JMeter Plugins Manager

Installed Plugins Available Plugins Upgrades

influ

JMeter Listener pack

JMeter Listener pack

Vendor: Alexander Babaev

This JMeter Plugin allows to write load test data on-the-fly to [ClickHouse](#) (as well as InfluxDB, ElasticSearch, with additional feature: aggregation of Samplers)
Explanations and usage examples on [project wiki](#).
[Dashboards for ClickHouse](#).

Documentation: <https://gitlab.com/testload/jmeter-listener/wikis/home>

What's new in version 1.7: New ClickHouse table scheme jmresults(buffer)->jmresults_data(ops storage with extended data request/response)->jmresults_statistic(mat view as archive storage). New field (hostname) in stats

Maven groupId: *cloud.testload*, artifactId: *jmeter.pack-listener*, version: *1.7*

Version: 1.7

Review Changes

Install plugin: jmeter.pack-listener 1.7



Demo with JMeter



Basic elements of JMeter

Test plan

Thread Group

**Sampler
Listener**

Config Element

Logic controller

Assertions

Pre/post processor



Let's start

The screenshot shows the Apache JMeter 5.2.1 interface. The title bar reads "Apache JMeter (5.2.1)". The toolbar contains various icons for file operations, test plan management, and monitoring. The main window is titled "Test Plan". It includes fields for "Name" (set to "Test Plan") and "Comments". A section for "User Defined Variables" is present, with a table for defining variables by Name and Value. Below this are buttons for "Detail", "Add", "Add from Clipboard" (which is highlighted), "Delete", "Up", and "Down". There are also checkboxes for run settings: "Run Thread Groups consecutively (i.e. one at a time)" (unchecked), "Run tearDown Thread Groups after shutdown of main threads" (checked), and "Functional Test Mode (i.e. save Response Data and Sampler Data)" (unchecked). A note states that selecting Functional Test Mode may adversely affect performance. At the bottom, there is a section for adding classpath entries with "Browse...", "Delete", and "Clear" buttons, and a "Library" table.

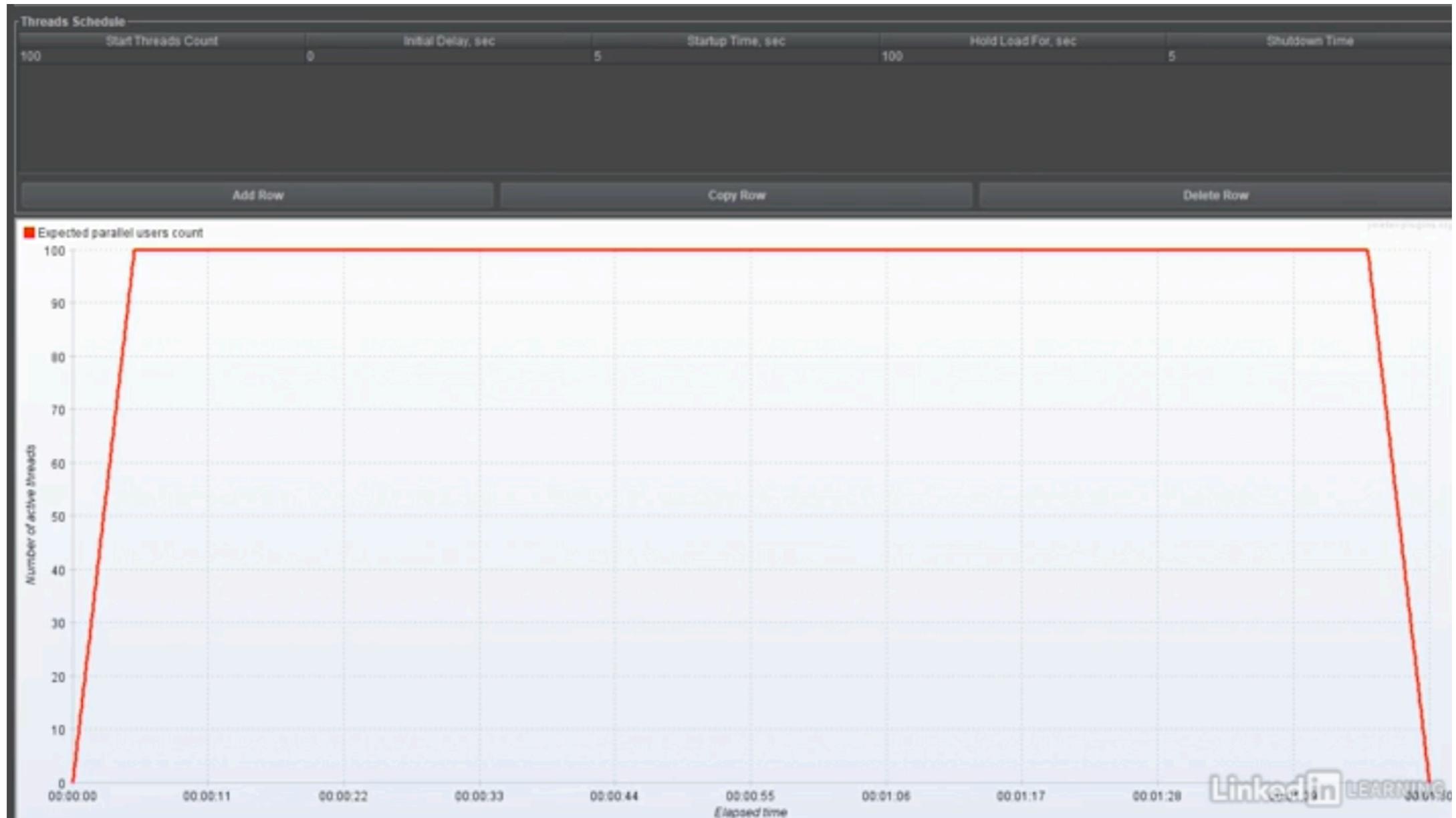


Workshop with Thread Group



Basic load

Certain number of users

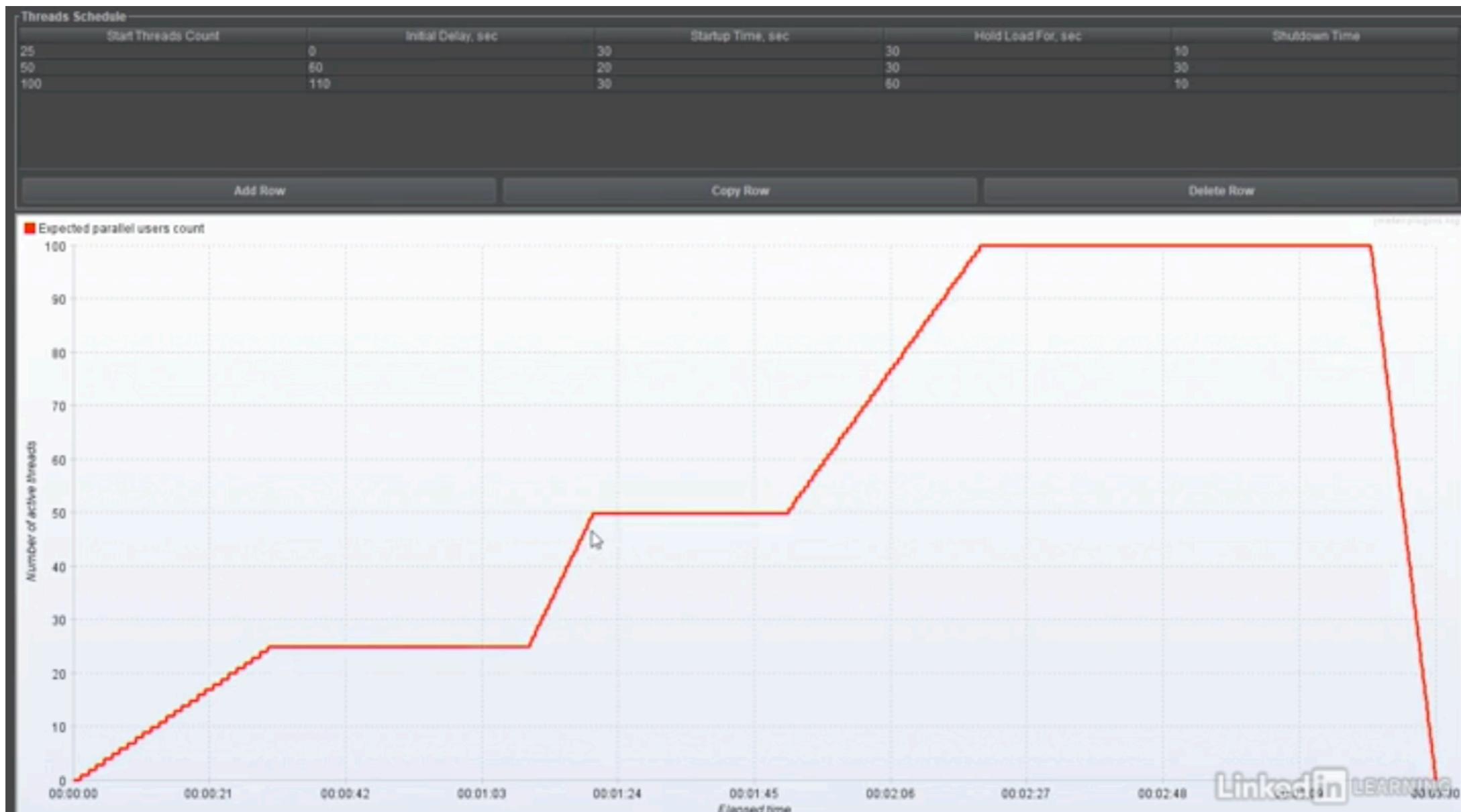


Ramp up

Understand how system response to increased loads overtime



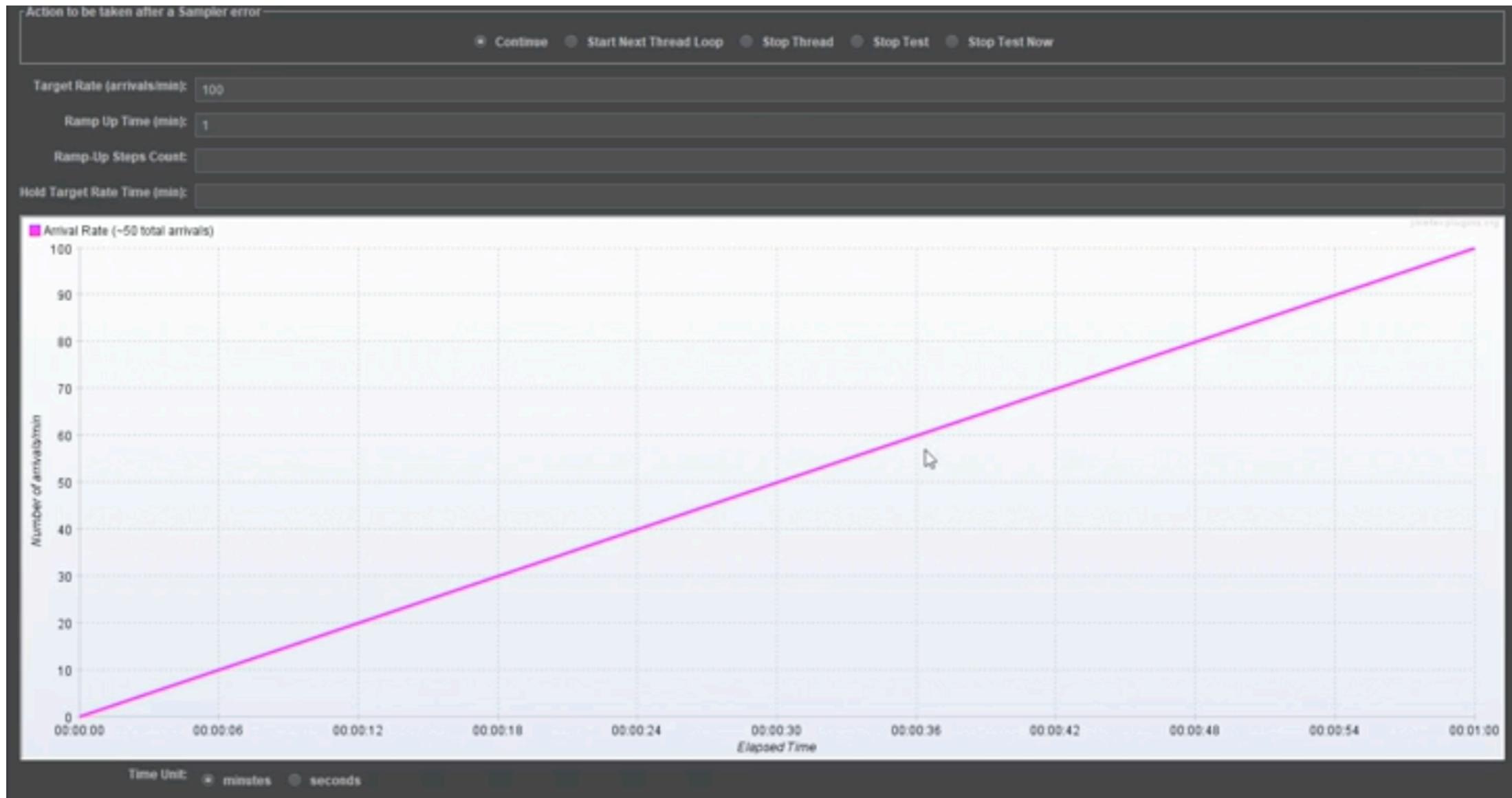
Multi-step ramp up



Random load



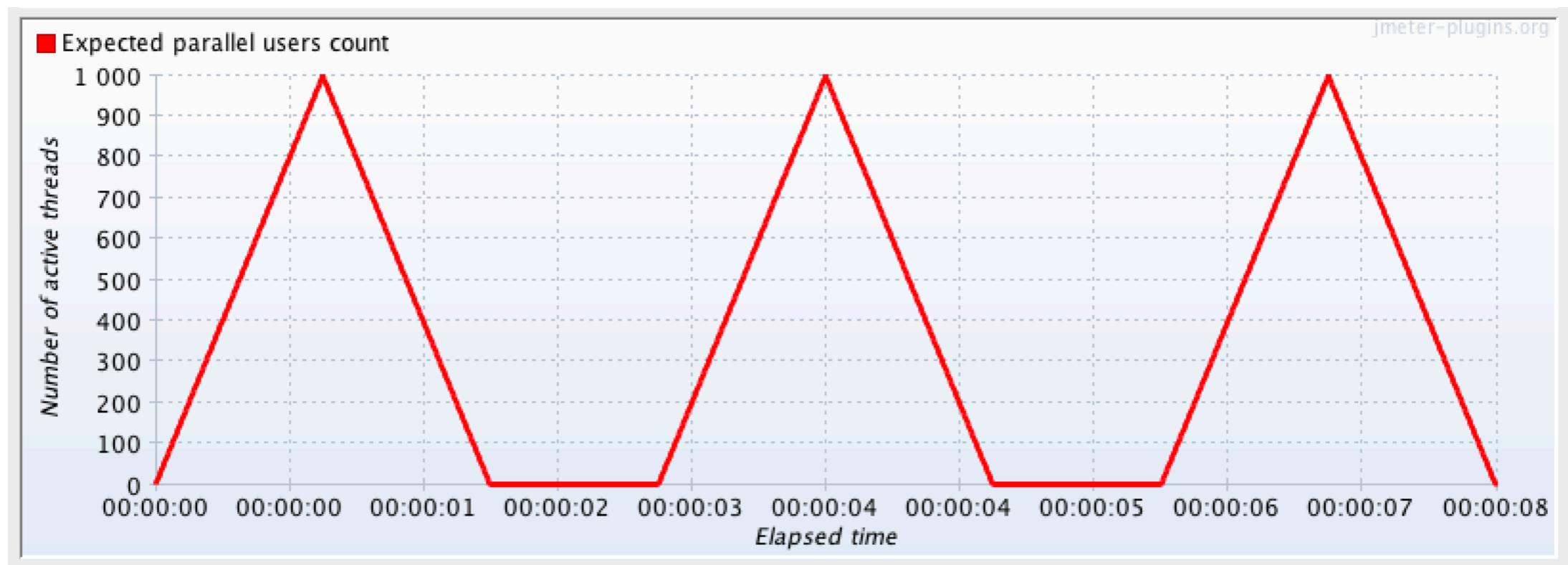
Linear load



Exponential load



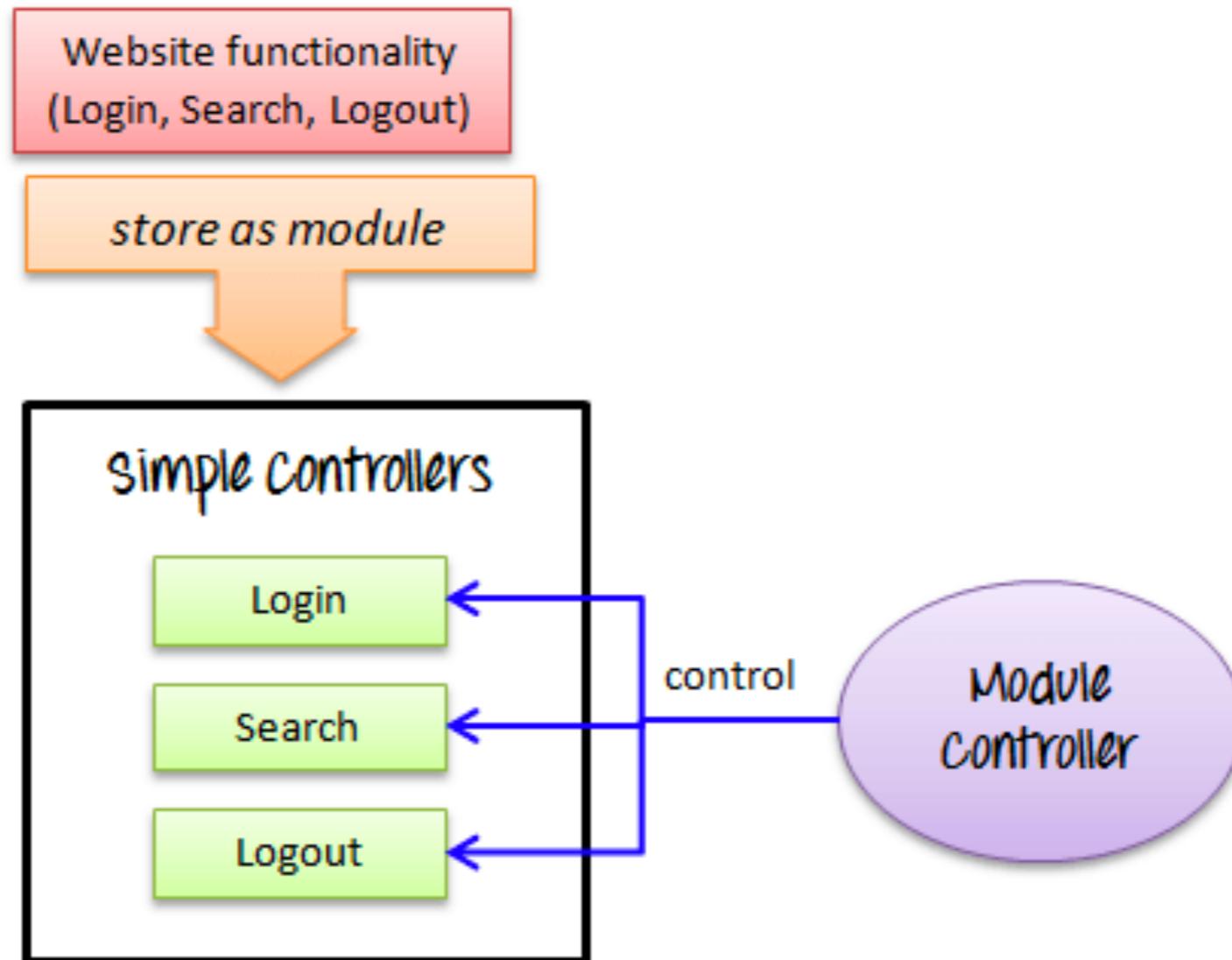
Quiz



Workshop with HTTP Request



Run in sequence with controller



Working with Data



Working with data (CSV)

```
user01,password01  
user02,password02  
user03,password03  
user04,password04  
user05,password05
```



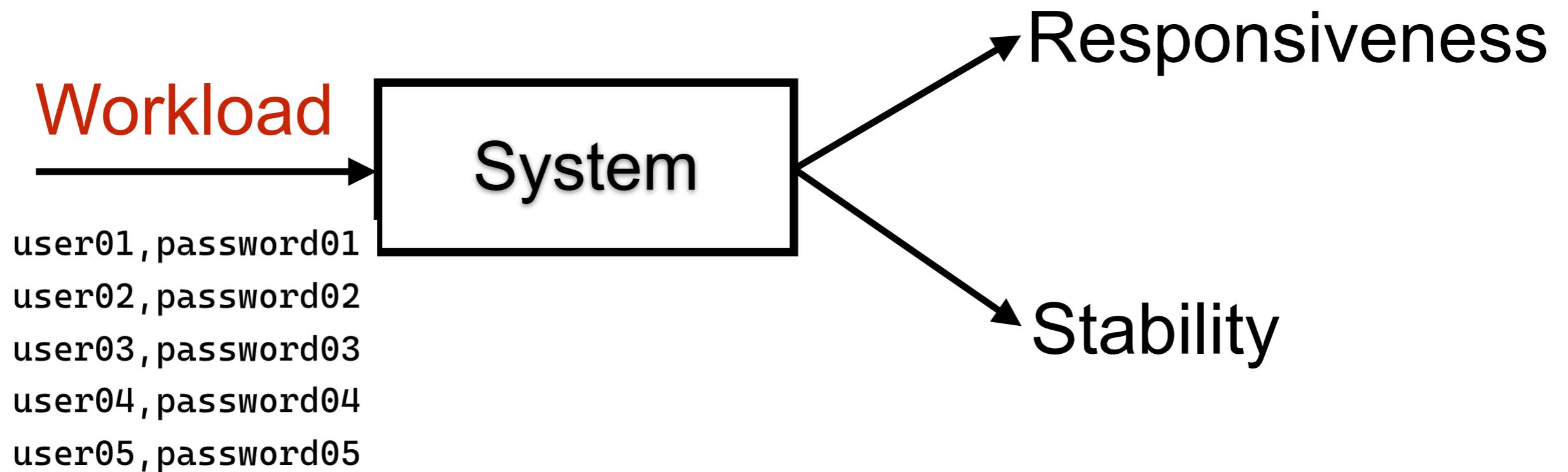
Working with data (CSV)

username password

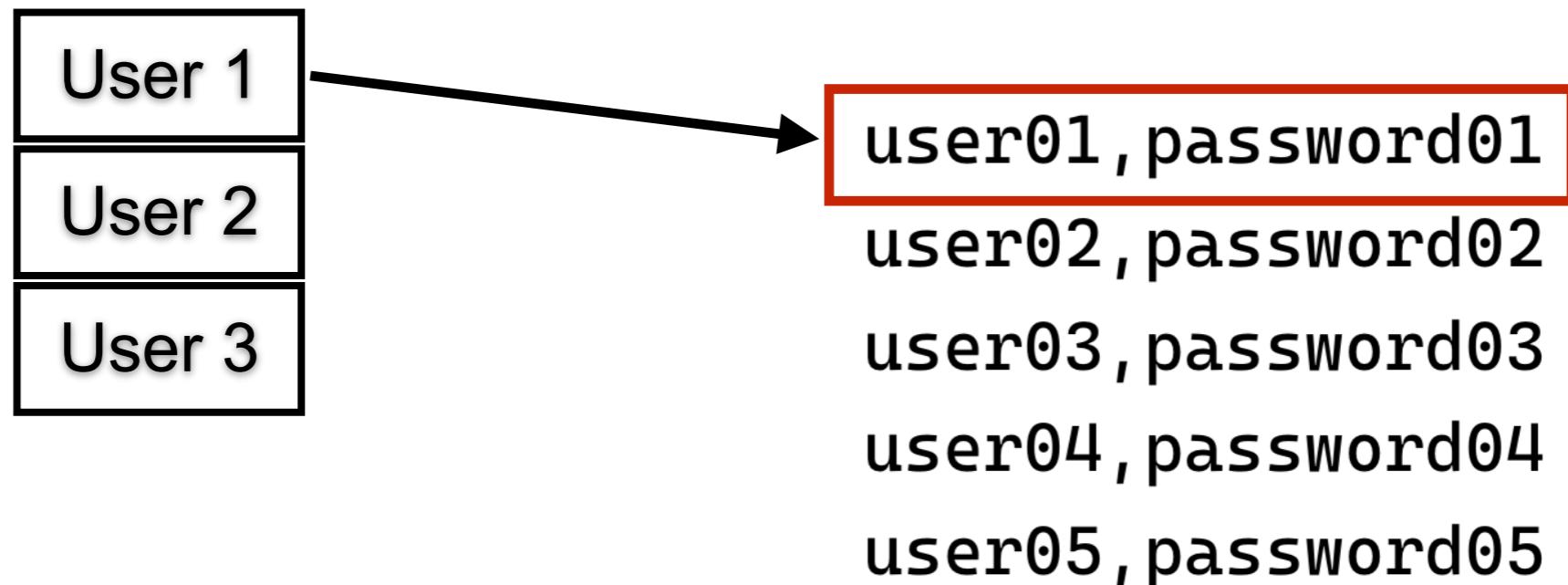
user01	password01
user02	password02
user03	password03
user04	password04
user05	password05



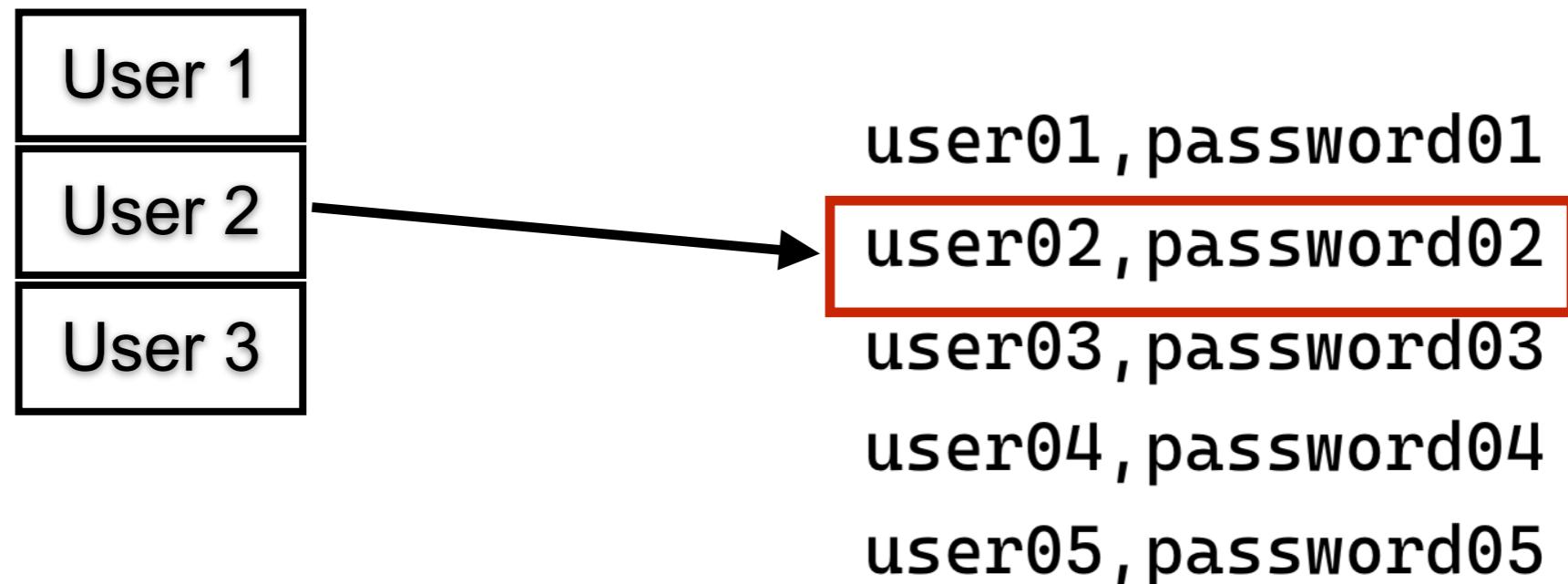
Create workload with JMeter



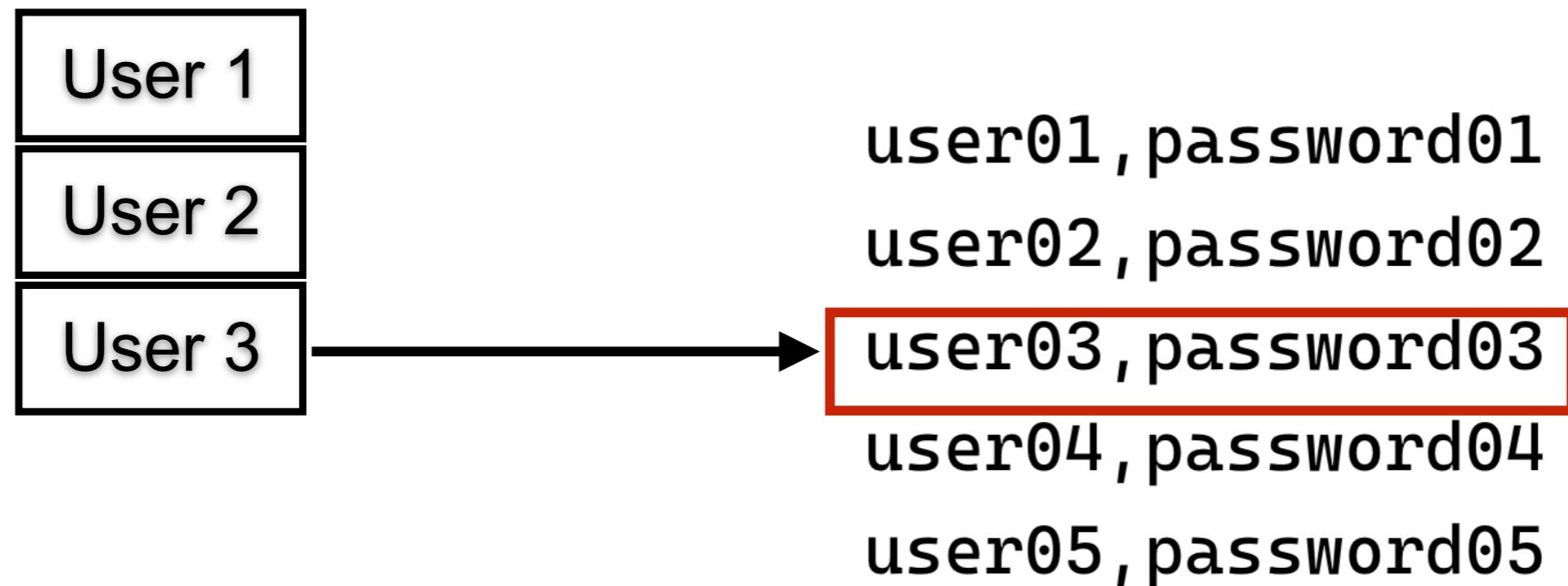
Working with data (CSV)



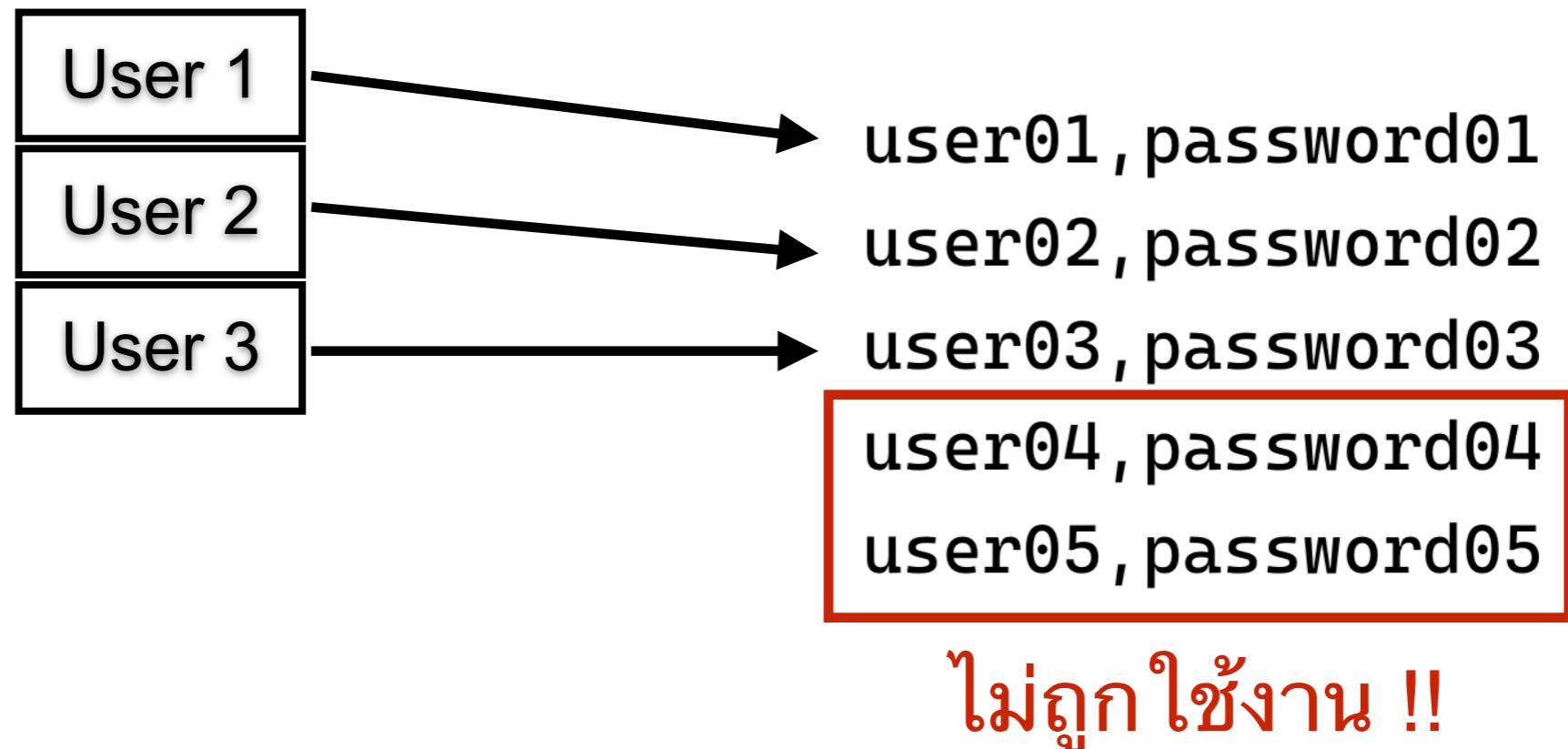
Working with data (CSV)



Working with data (CSV)

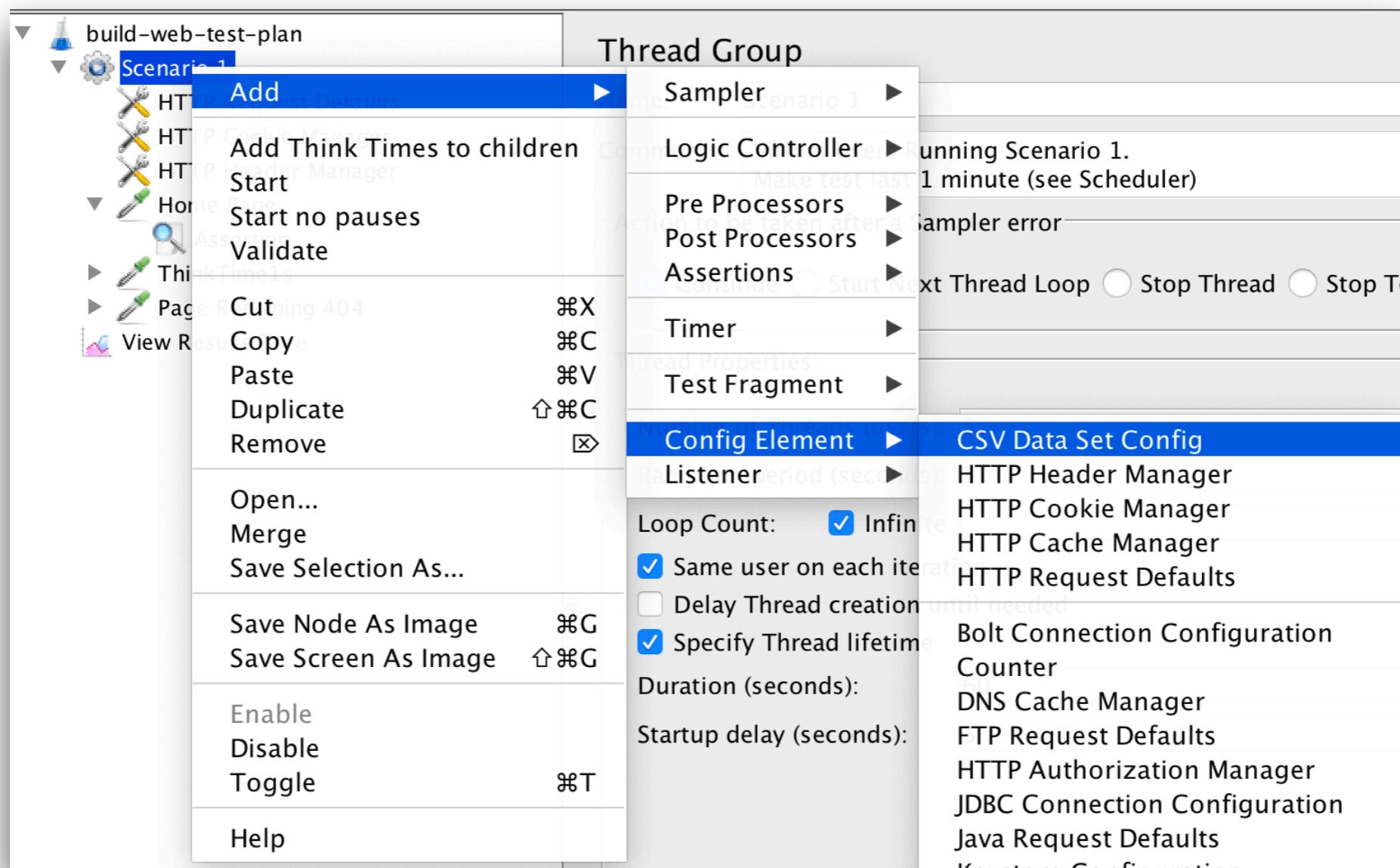


Working with data (CSV)

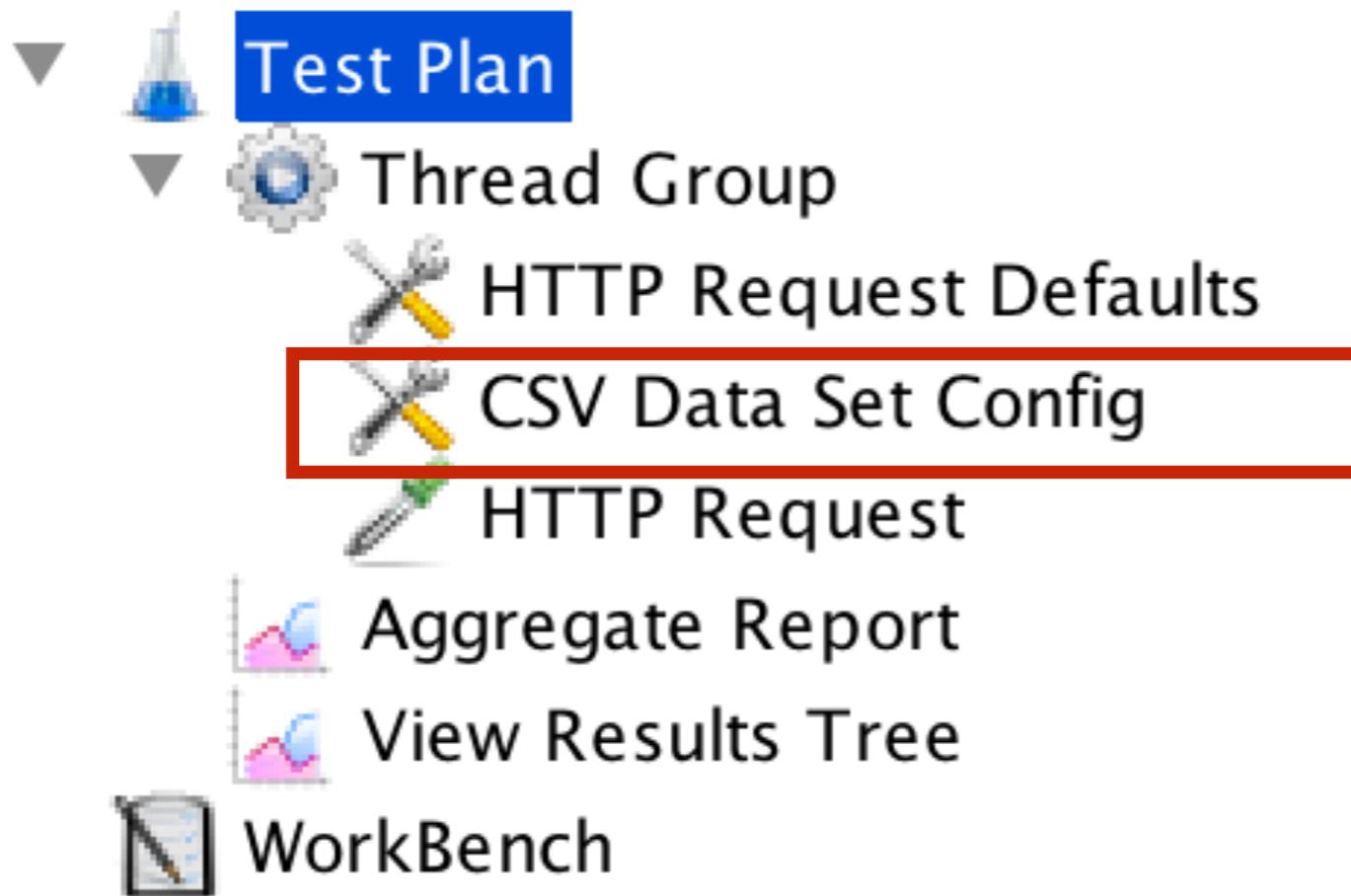


Working with data (CSV)

Add -> Config Element -> CSV Data Set Config



Working with data (CSV)



Working with data (CSV)

CSV Data Set Config

Name: CSV Data Set Config

Comments:

Configure the CSV Data Source

Data Source

Filename:	D:\Job\JMeter\csv_data.txt
File encoding:	
Variable Names (comma-delimited):	user,password,cookielength,cookieNeverExp

Ignore first line (only used if Variable Names is not empty): False

Delimiter (use '\t' for tab):	,
Allow quoted data?:	True

Recycle on EOF ?: True

Stop thread on EOF ?: False

Sharing mode: All threads



Working with data (CSV)

HTTP Request

Name: Login

Comments:

Web Server

Protocol [http]: Server Name or IP: Port Number:

HTTP Request

Method: POST Path: /index.php?board=;action=login2 Content encoding:

Redirect Automatically Follow Redirects Use KeepAlive Use multipart/form-data for POST Browser-compatible headers

Parameters Body Data Files Upload

Send Parameters With the Request:

Name:	Value	Encode?	Include Equals?
user	<code> \${user}</code>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
passwd	<code> \${passwd}</code>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
cookieLength	<code> \${cookieLength}</code>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
cookieNeverExp	<code> \${cookieNeverExp}</code>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>

The 'Value' column for the parameters 'user', 'passwd', 'cookieLength', and 'cookieNeverExp' contains the expression `${Variable Name}`, which is highlighted with a red box.

`${Variable Name}`



Run in command-line

```
$jmeter -n -t [jmx file]  
    -l [results file]  
    -e -o [Path to web report folder]
```

[results file] = CSV file

[path to web report folder] = Name of folder



Run in command-line

1.4.4 CLI Mode (Command Line mode was called NON GUI mode)

For load testing, you must run JMeter in this mode (Without the GUI) to get the optimal results from it. To do so, use the following command options:

- n** This specifies JMeter is to run in cli mode
- t** [name of JMX file that contains the Test Plan].
- l** [name of JTL file to log sample results to].
- j** [name of JMeter run log file].
- r** Run the test in the servers specified by the JMeter property "**remote_hosts**"
- R** [list of remote servers] Run the test in the specified remote servers
- g** [path to CSV file] generate report dashboard only
- e** generate report dashboard after load test
- o** output folder where to generate the report dashboard after load test. Folder must not exist or be empty

The script also lets you specify the optional firewall/proxy server information:

- H** [proxy server **hostname** or ip address]
- P** [proxy server port]

https://jmeter.apache.org/usermanual/get-started.html#non_gui



HTML Report

Apache JMeter Dashboard

- Dashboard
- Charts
- Customs Graphs

Test and Report information

Source file	"result.csv"
Start Time	"4/16/20, 9:37 AM"
End Time	"4/16/20, 9:38 AM"
Filter for display	==

APDEX (Application Performance Index)

Apdex	T (Toleration threshold)	F (Frustration threshold)	Label
0.000	500 ms	1 sec 500 ms	Total
0.000	500 ms	1 sec 500 ms	Login Page

Requests Summary

KO
100%

Legend: KO (Red), OK (Green)

Statistics

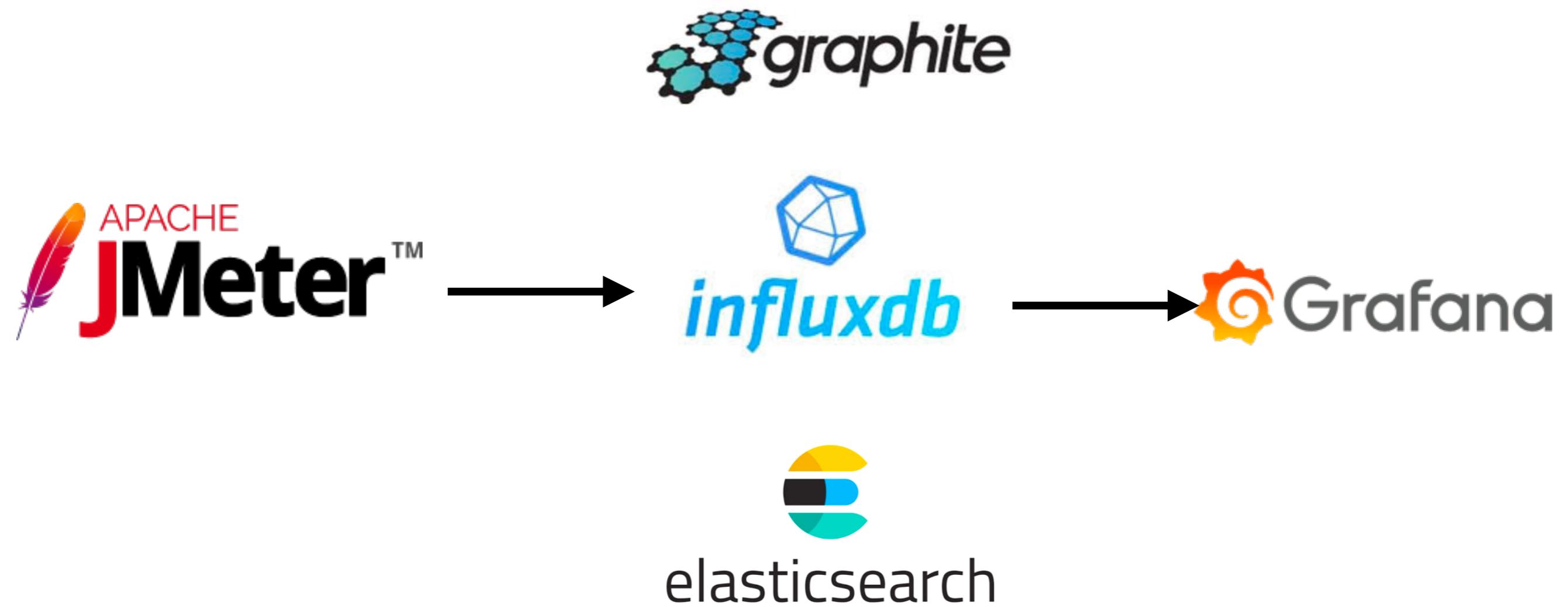


Realtime result with JMeter

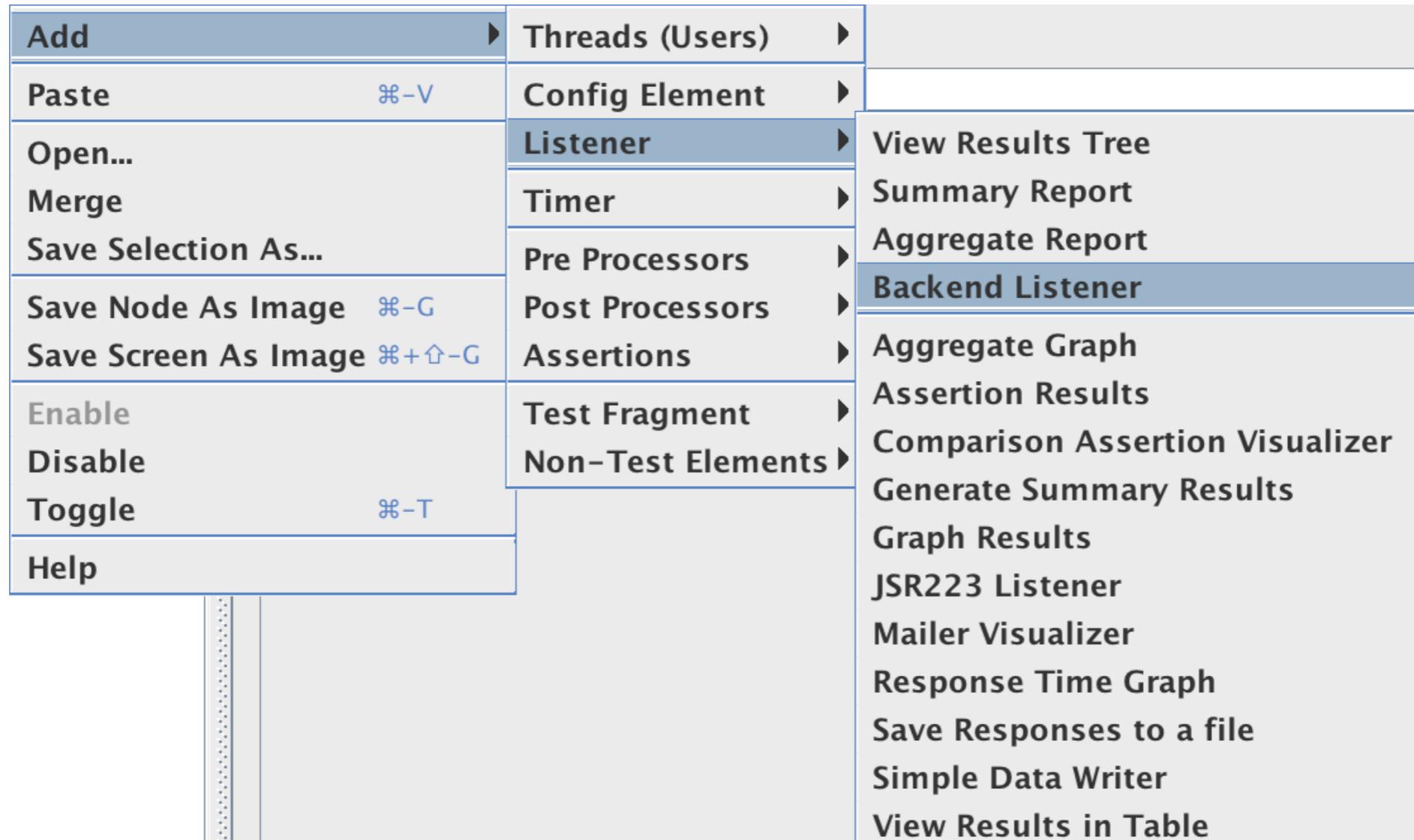
<https://jmeter.apache.org/usermanual/realtime-results.html>



Realtime result with JMeter



Backend listener



Backend listener

Backend Listener

Name:	Backend Listener
Comments:	
Backend Listener implementation	<code>org.apache.jmeter.visualizers.backend.influxdb.InfluxdbBackendListenerClient</code>
Async Queue size	5000
influxdbMetricsSender	<code>org.apache.jmeter.visualizers.backend.graphite.GraphiteBackendListenerClient</code>
influxdbUrl	<code>org.apache.jmeter.visualizers.backend.influxdb.InfluxdbBackendListenerClient</code>
application	application name
measurement	jmeter
summaryOnly	false
samplersRegex	.*
percentiles	99;95;90
testTitle	Test name
eventTags	



Backend listener

Backend Listener

Name: Backend Listener

Comments:

Backend Listener implementation `org.apache.jmeter.visualizers.backend.influxdb.InfluxdbBackendListenerClient`

Async Queue size 5000

Parameters

Name:	Value
influxdbMetricsSender	<code>org.apache.jmeter.visualizers.backend.influxdb.HttpMetricsSender</code>
influxdbUrl	<code>http://localhost:8086/write?db=jmeter</code>
application	application name
measurement	jmeter
summaryOnly	false
samplersRegex	.*
percentiles	99;95;90
testTitle	Test name
eventTags	

`influxdbUrl = http://localhost:8086/write?db=jmeter`

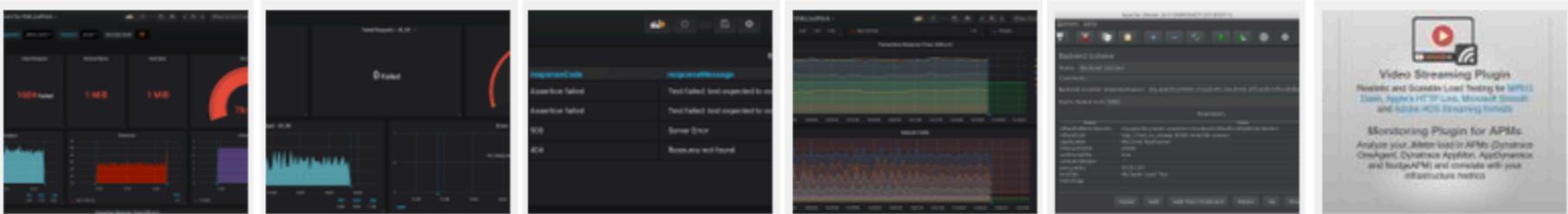


Grafana report

 Apache JMeter Dashboard using Core InfluxdbBackendListenerClient
by Philippe M.

DASHBOARD
Monitor your Apache JMeter load test in real time with InfluxDB and Grafana. Get overall summary, errors details and particular transaction response times.
Last updated: 2 years ago

Overview Revisions Reviews



This dashboard requires Apache JMeter 5 and upper versions. It shows overall statistics and you can zoom on one particular transaction. In order to use it you need to use JMeter Backend Listener and select InfluxdbBackendListenerClient.

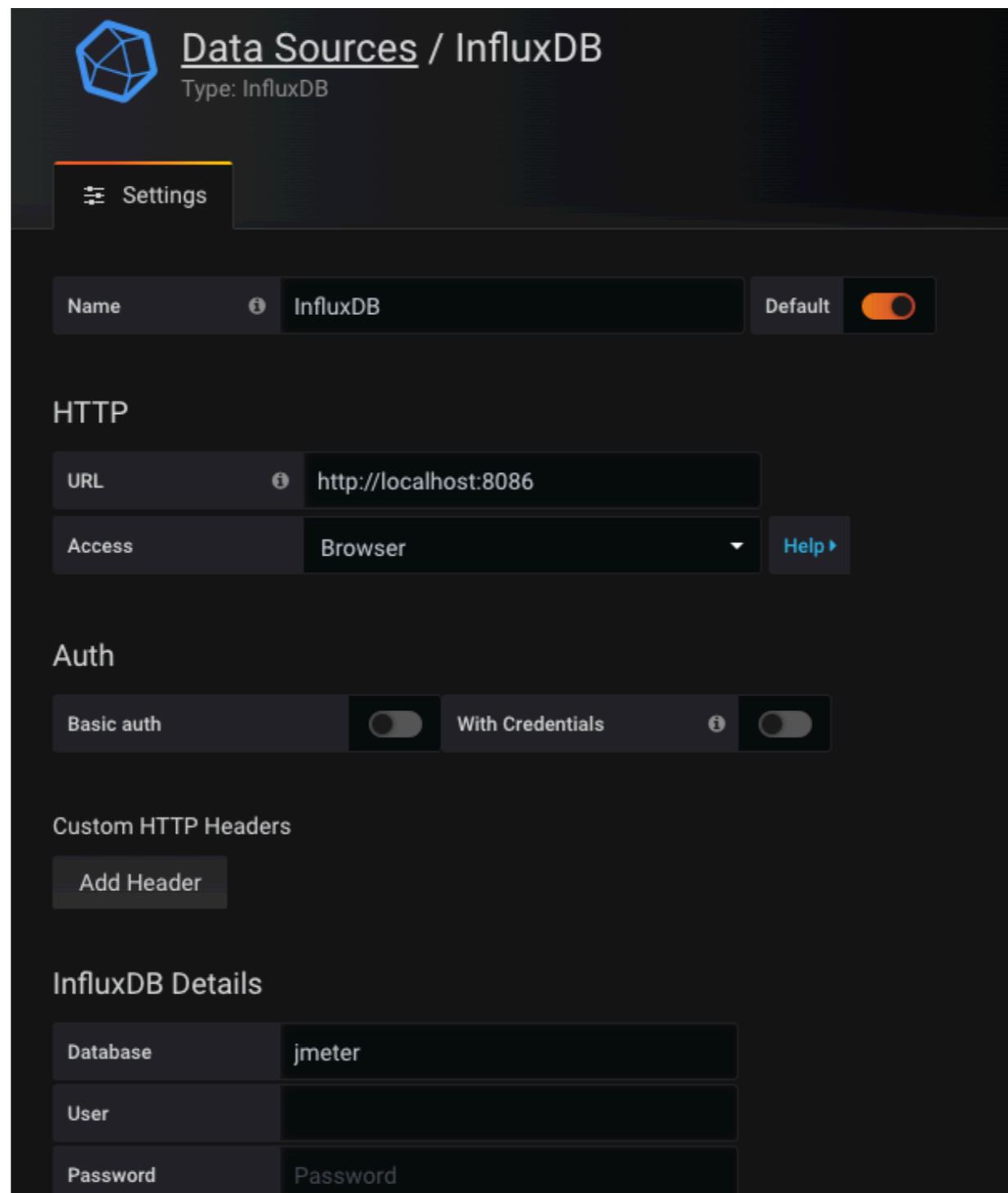
Setup:

- Add Backend Listener to your test plan (Add → Listener → Backend Listener) and select org.apache.jmeter.visualizers.backend.influxdb.HttpMetricsSender
- Provide in the Parameters table the InfluxDB settings, provide a name for the test, and specify which samplers to record.

<https://grafana.com/grafana/dashboards/5496>



Grafana report

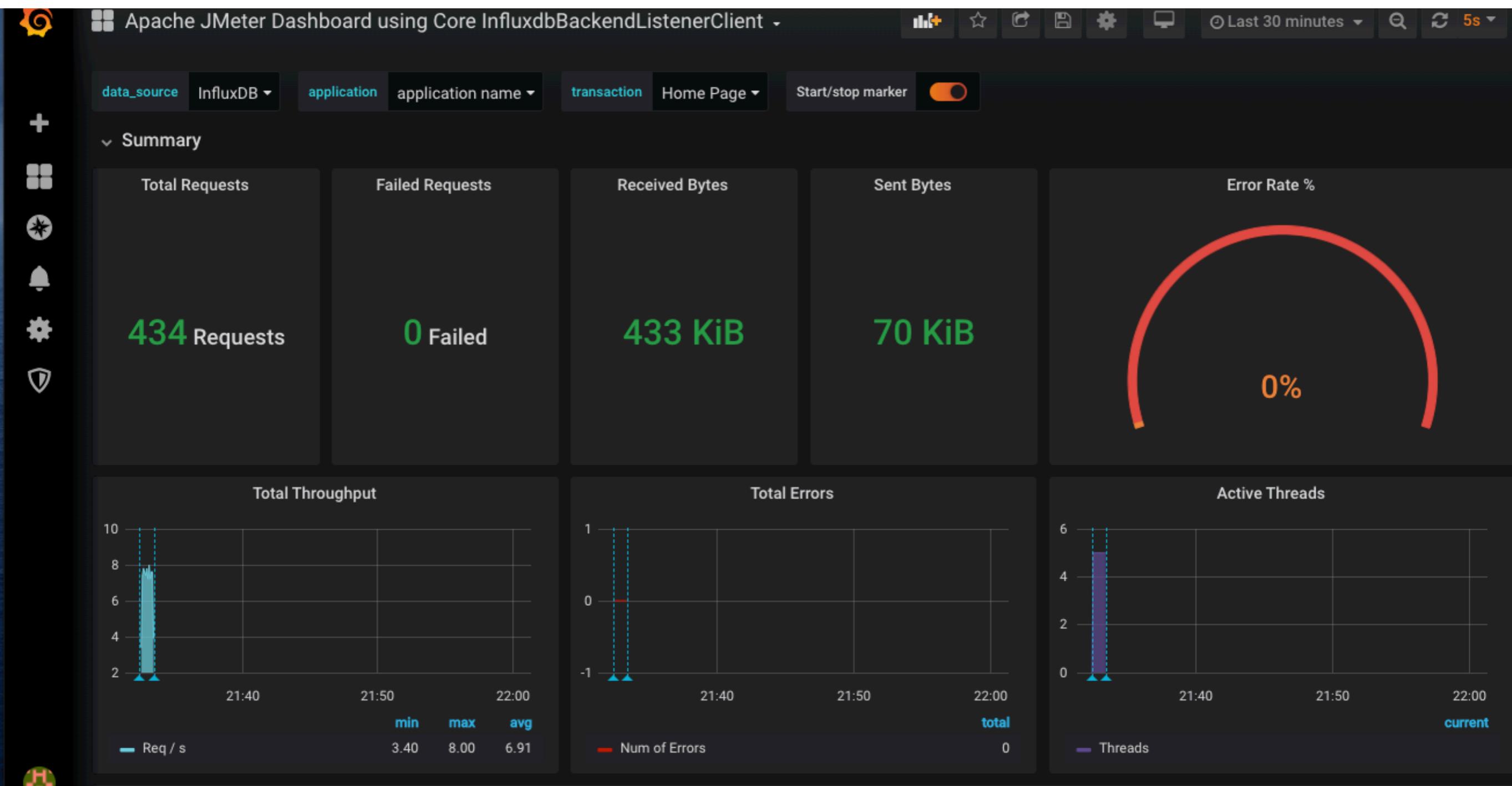


The screenshot shows the 'Data Sources / InfluxDB' configuration page in Grafana. The 'InfluxDB' data source is selected, indicated by the blue icon and the text 'Type: InfluxDB'. The 'Settings' tab is active. The 'Name' field is set to 'InfluxDB' and has a 'Default' toggle switch turned on. Under the 'HTTP' section, the 'URL' is set to 'http://localhost:8086' and the 'Access' dropdown is set to 'Browser'. There is a 'Help' button next to the access dropdown. Under the 'Auth' section, both 'Basic auth' and 'With Credentials' toggles are turned off. The 'Custom HTTP Headers' section contains a single button labeled 'Add Header'. The 'InfluxDB Details' section shows the 'Database' set to 'jmeter', and both 'User' and 'Password' fields are empty.

Database	jmeter
User	
Password	Password



Grafana report

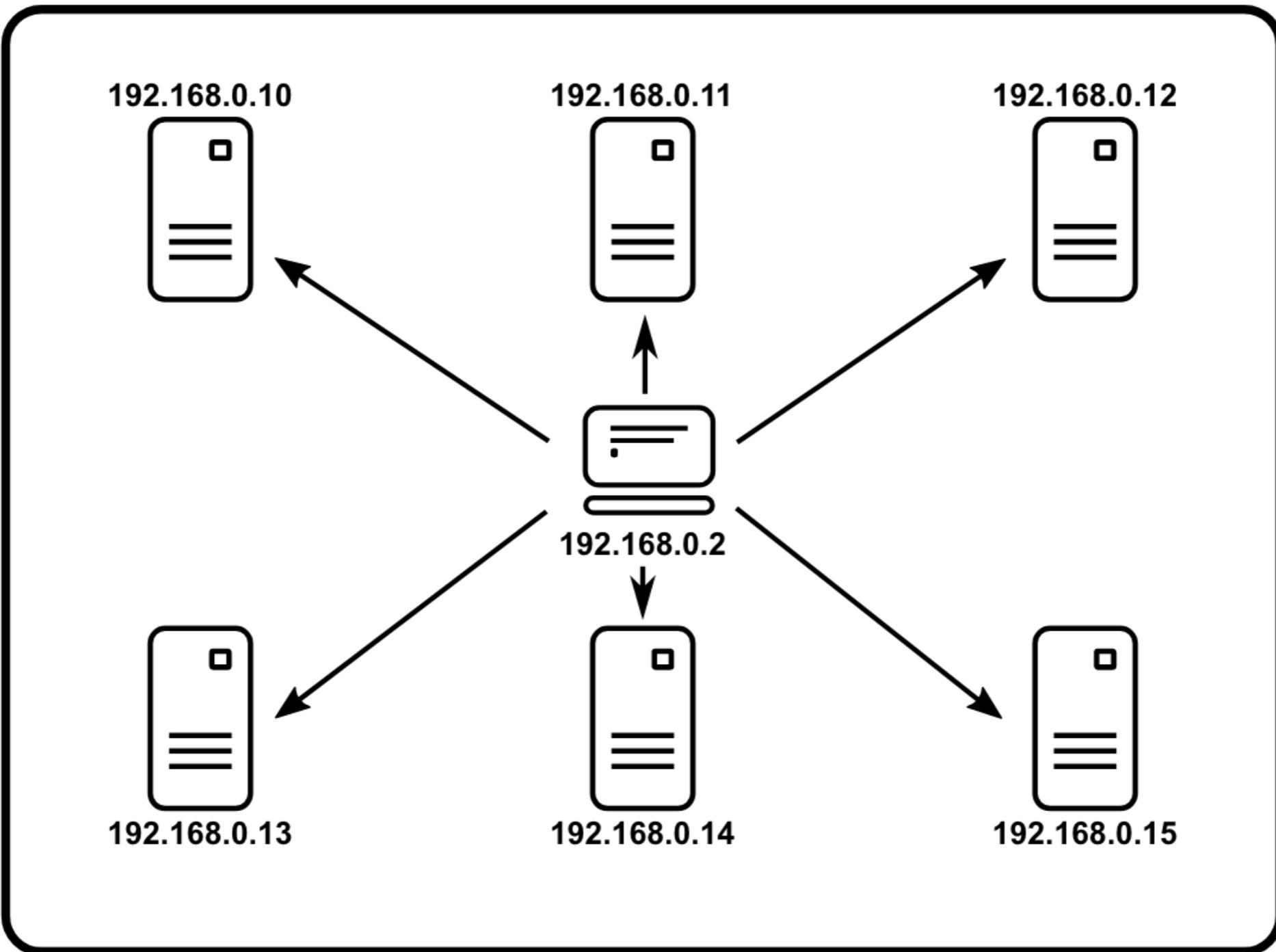


Distributed testing

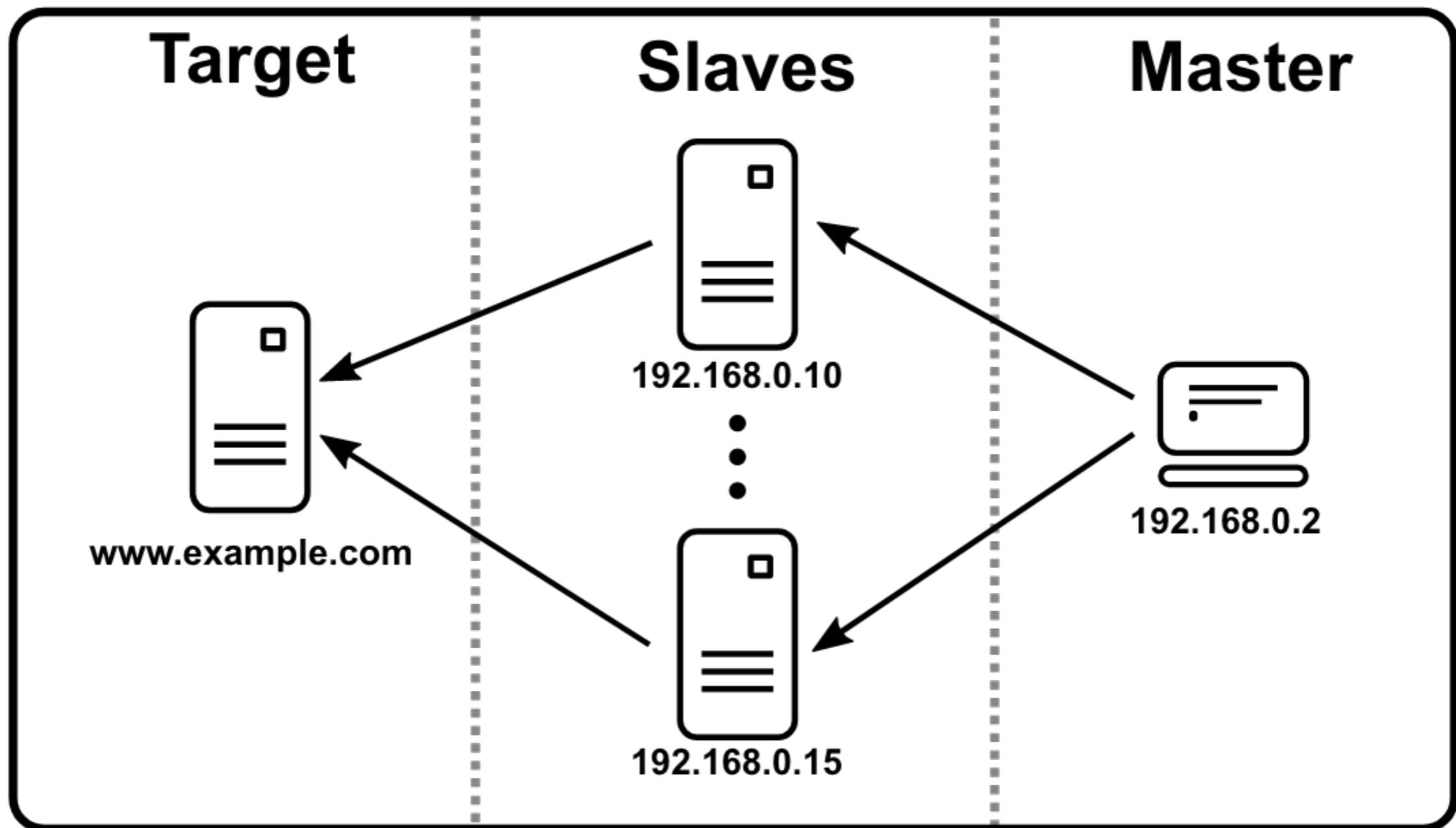
https://jmeter.apache.org/usermanual/jmeter_distributed_testing_step_by_step.html



Distributed testing



Architecture



Recording testing

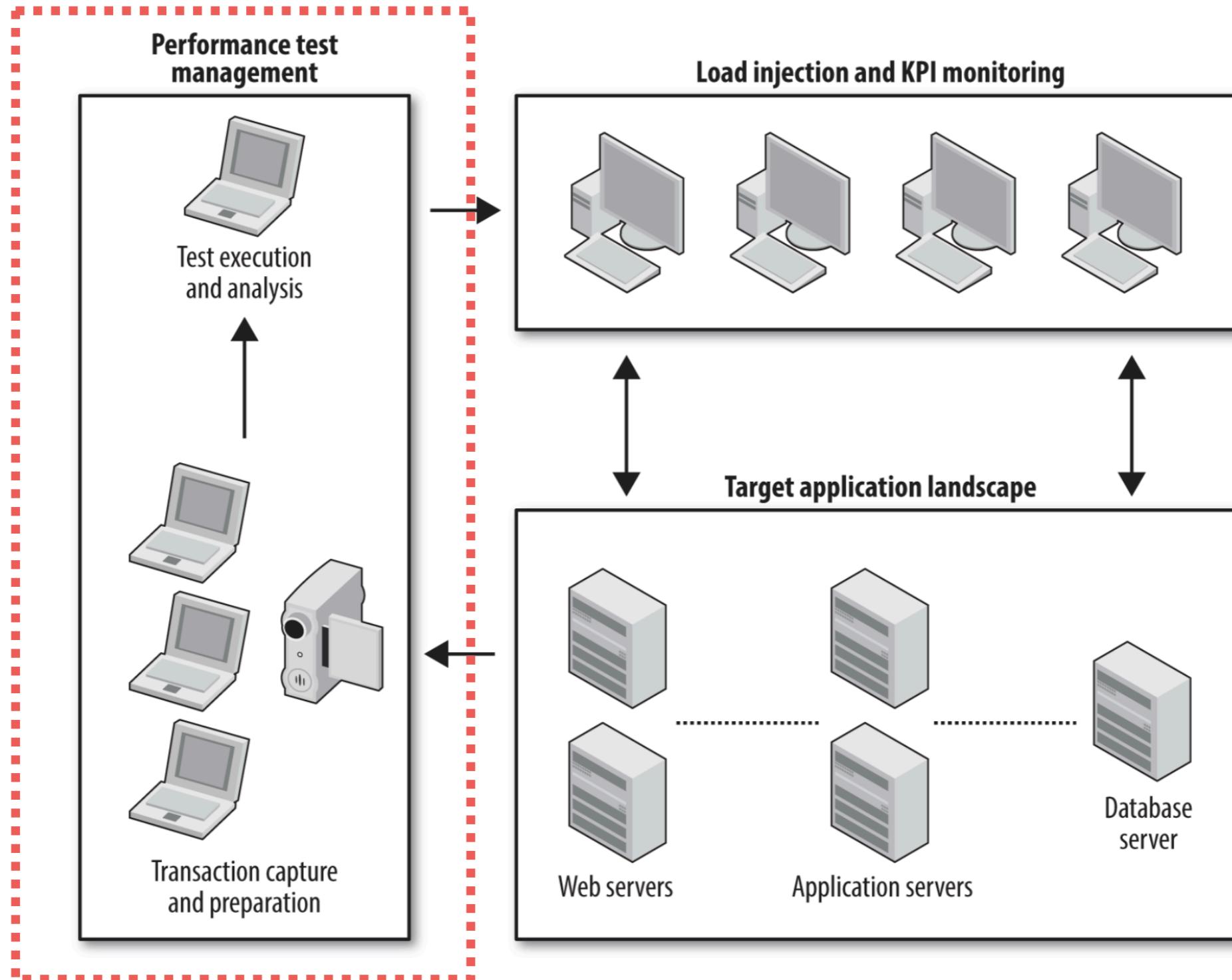
https://jmeter.apache.org/usermanual/jmeter_distributed_testing_step_by_step.html



Interpret result of testing



Performance testing architecture

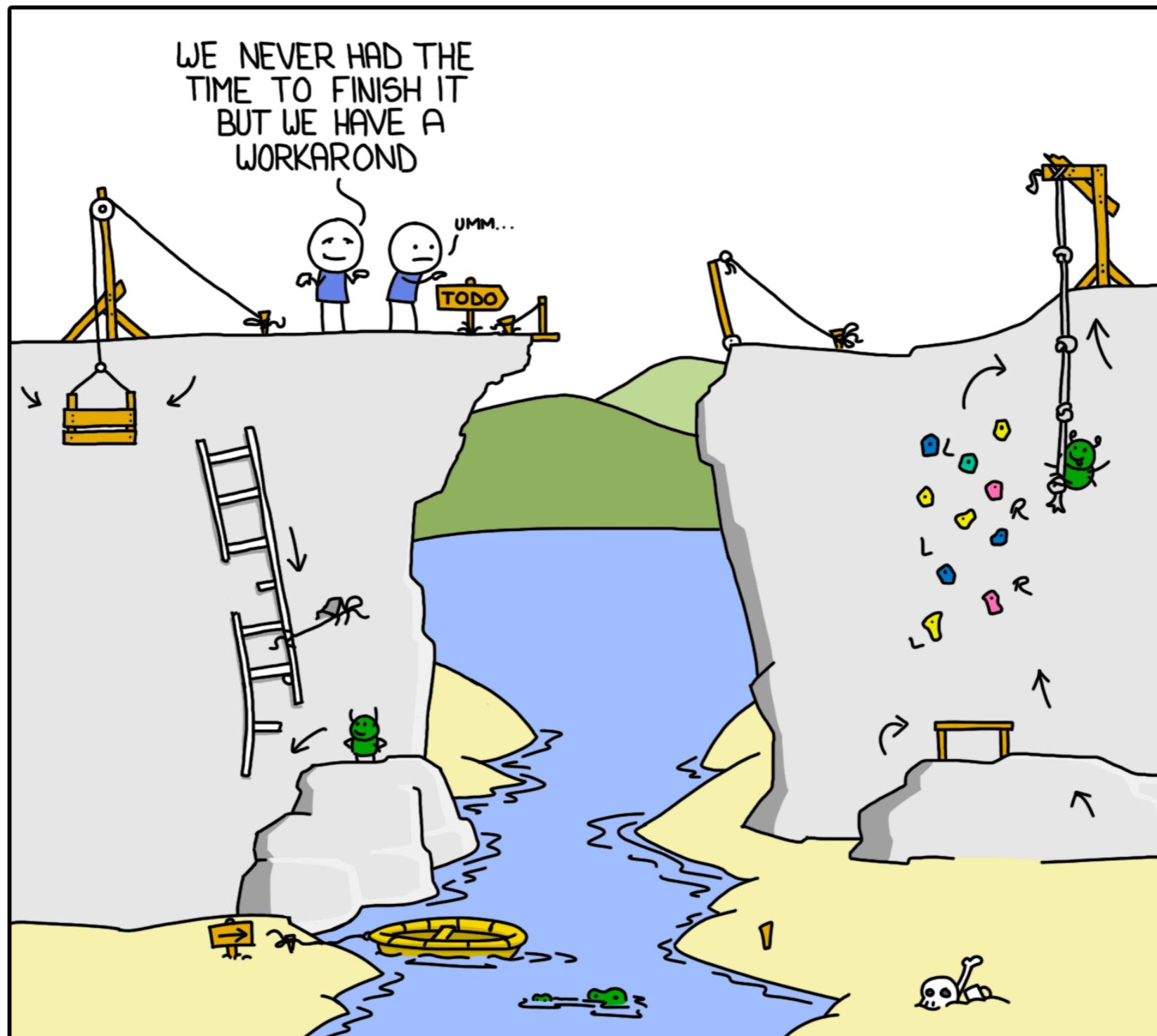


Goals

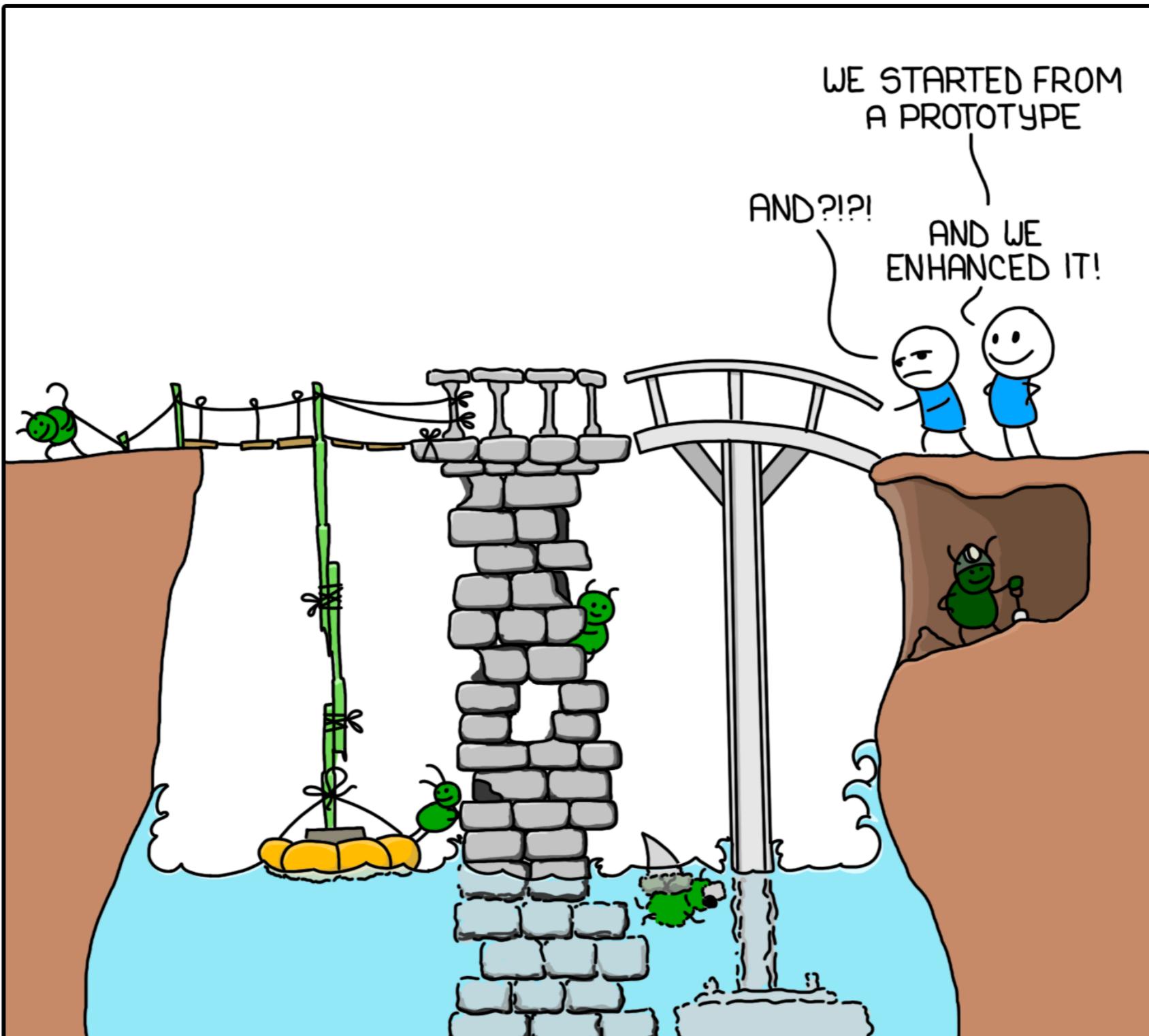
Root-cause analysis
Method of problem solving



WORKAROUND



PRODUCTION READY



MONKEYUSER.COM



Analysis process

Real-time analysis
Post-test analysis



Types of result/outputs

Statistic

Response time measurement

Throughput and capacity

Server KPI performance

Network KPI performance



Statistic

Mean and median
Standard deviation
Normal distribution
Percentile
Response time distribution



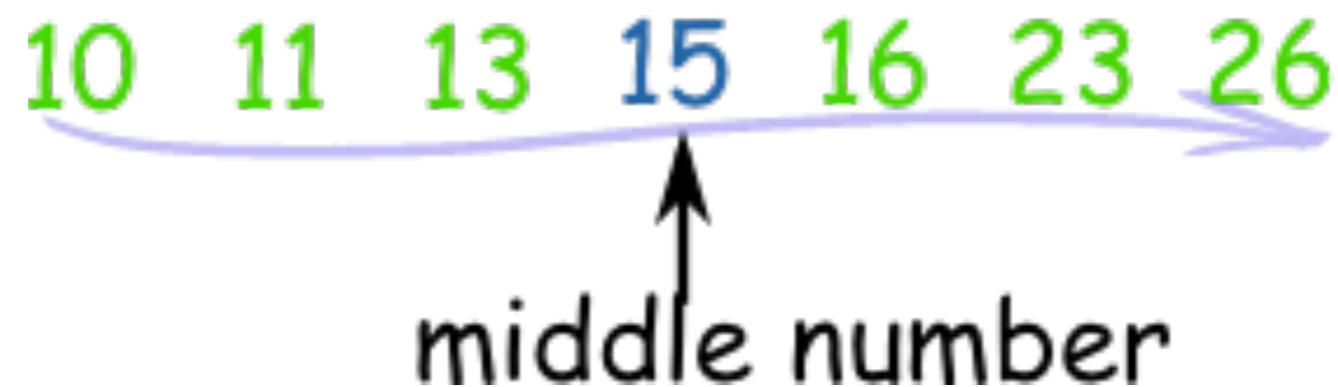
Mean

Average of the numbers



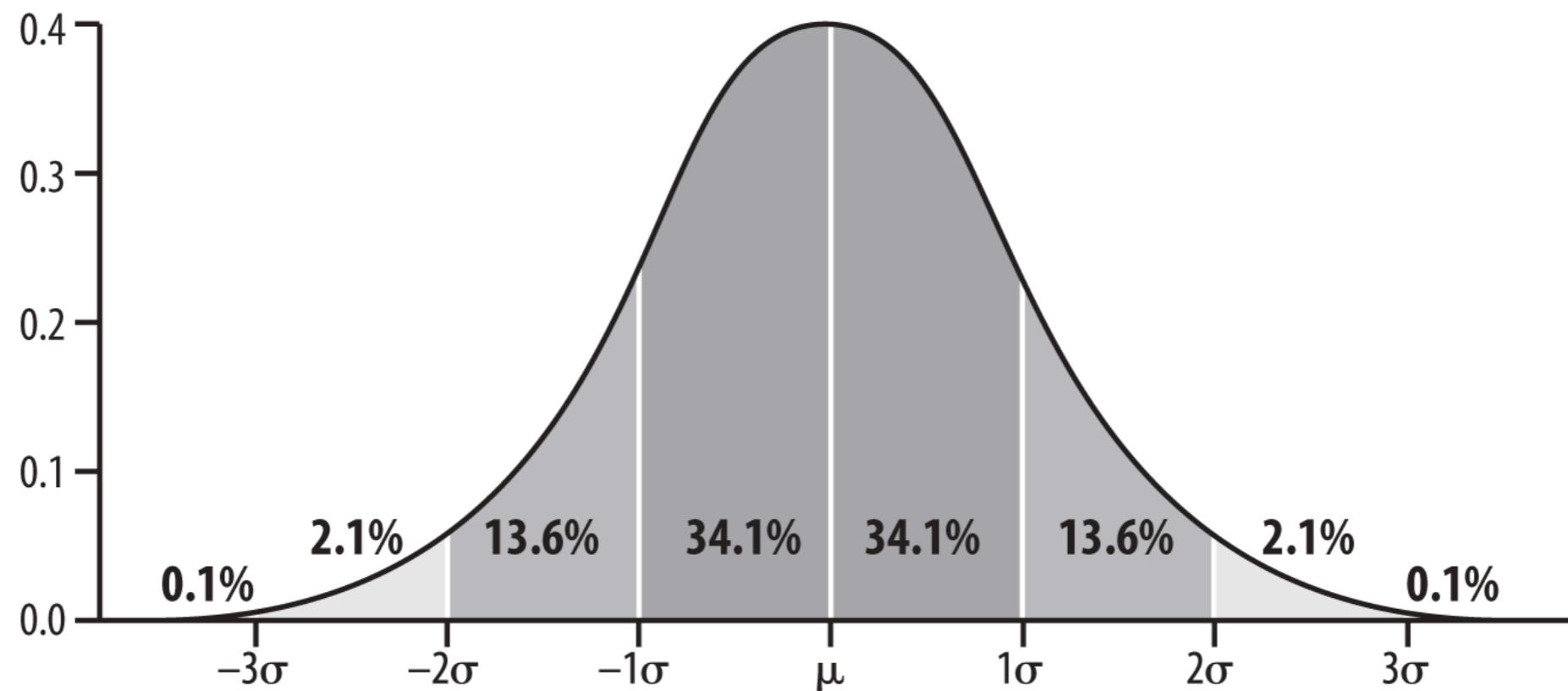
Median

the "middle" of a sorted list of numbers.



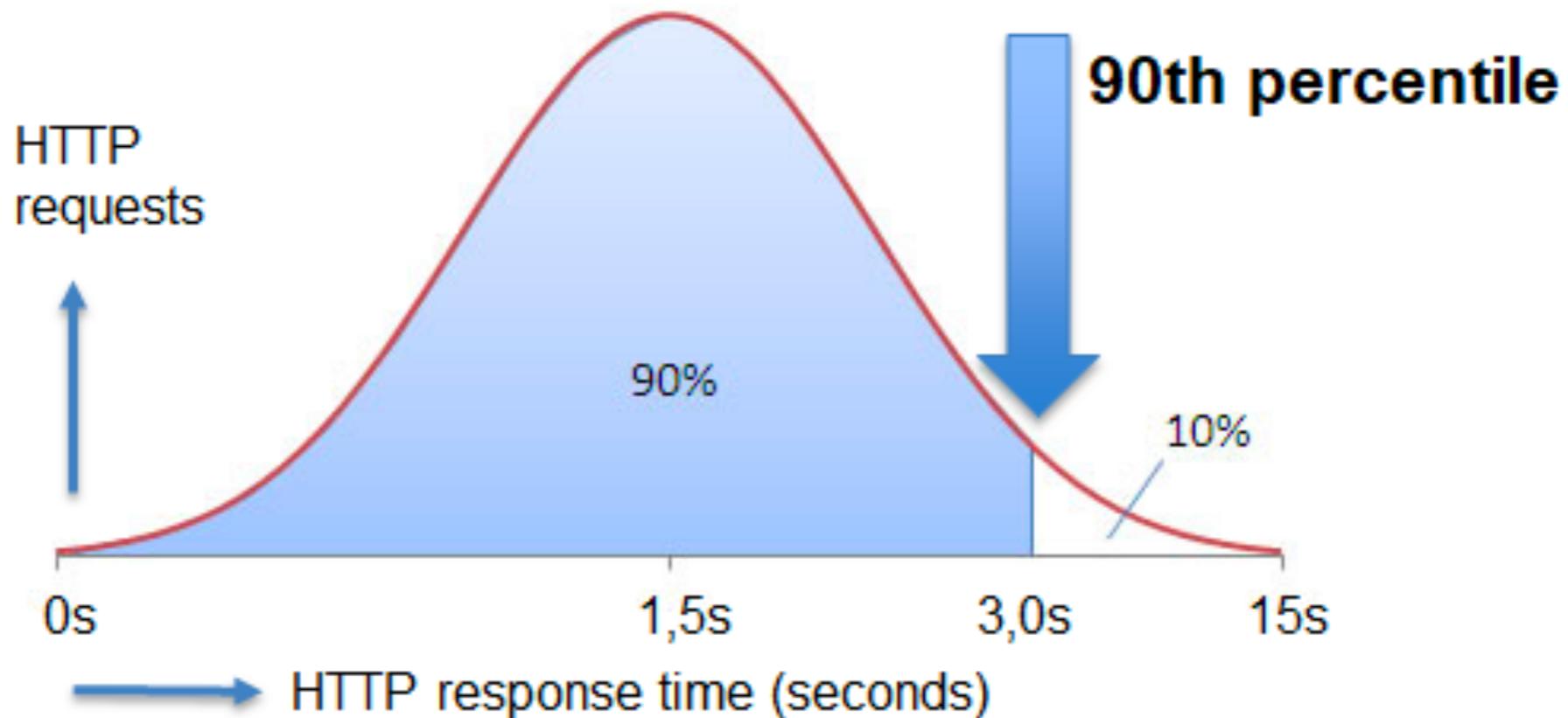
Standard deviation

Average variance from the calculated mean value



Percentile

50, 90, 95, 99 percentile ?



Percentile as a SLA

The value below which a percentage of data falls

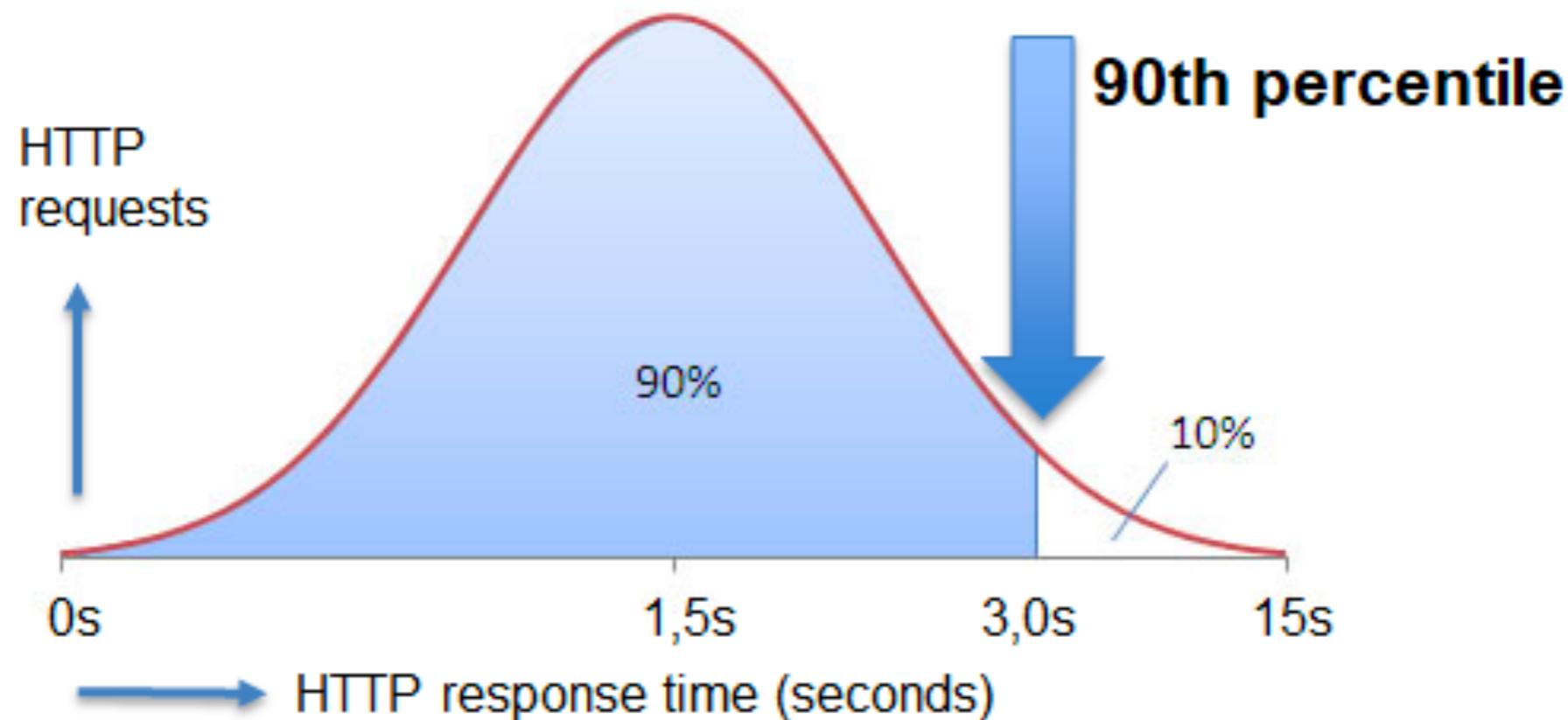


80% of people are shorter than you



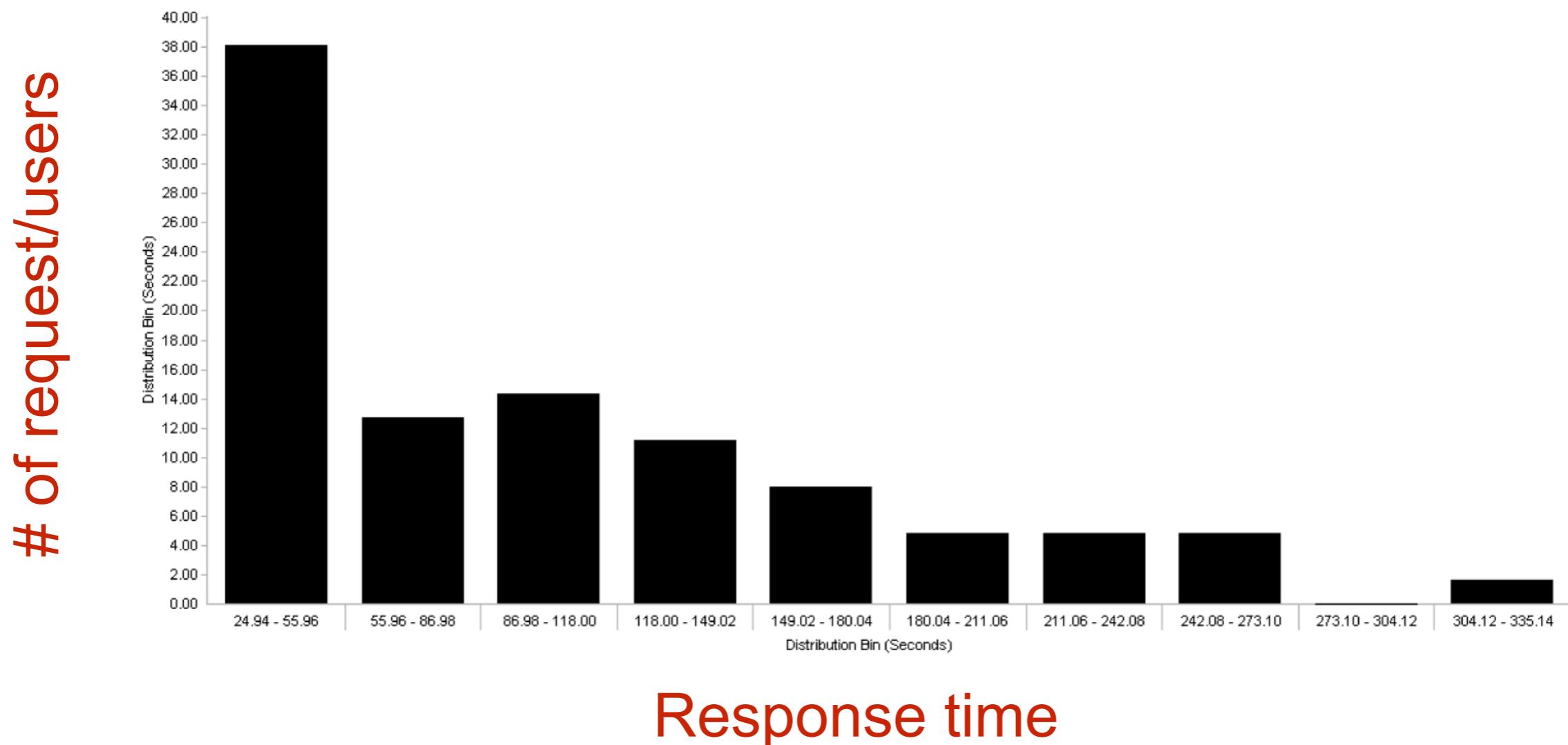
90th Percentile

90% of the requests is processed in 3.0 seconds or less



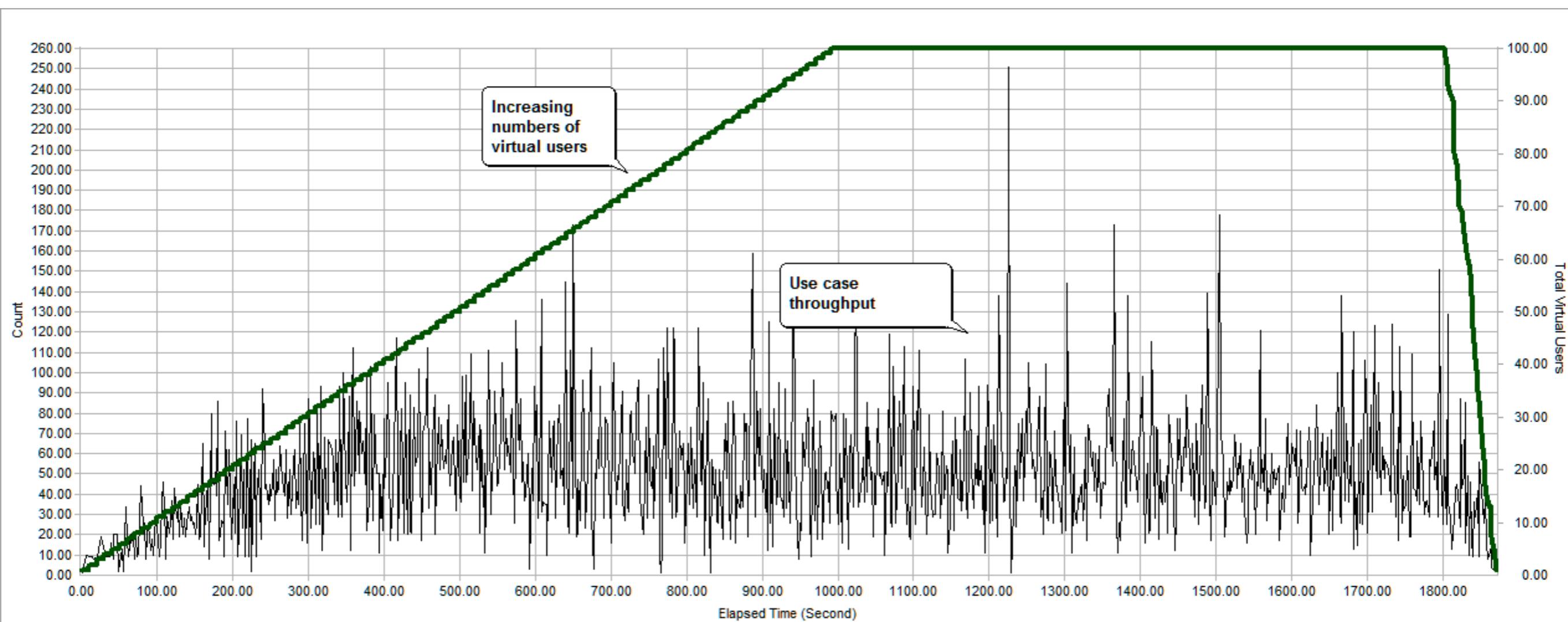
Response time distribution

Aggregate all the response times collected during a performance test into a series of groups/buckets

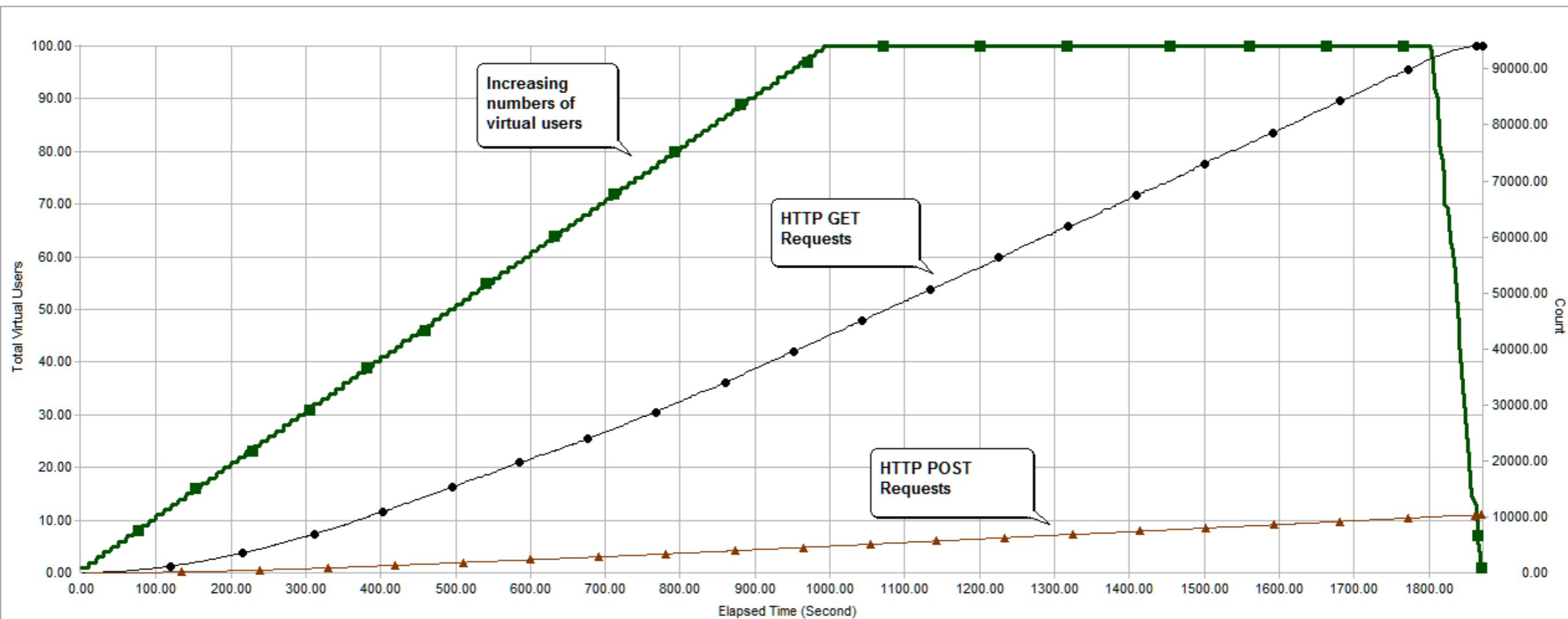


Throughput

How much data/work/use cases can handled simultaneously ?



Working with Web request

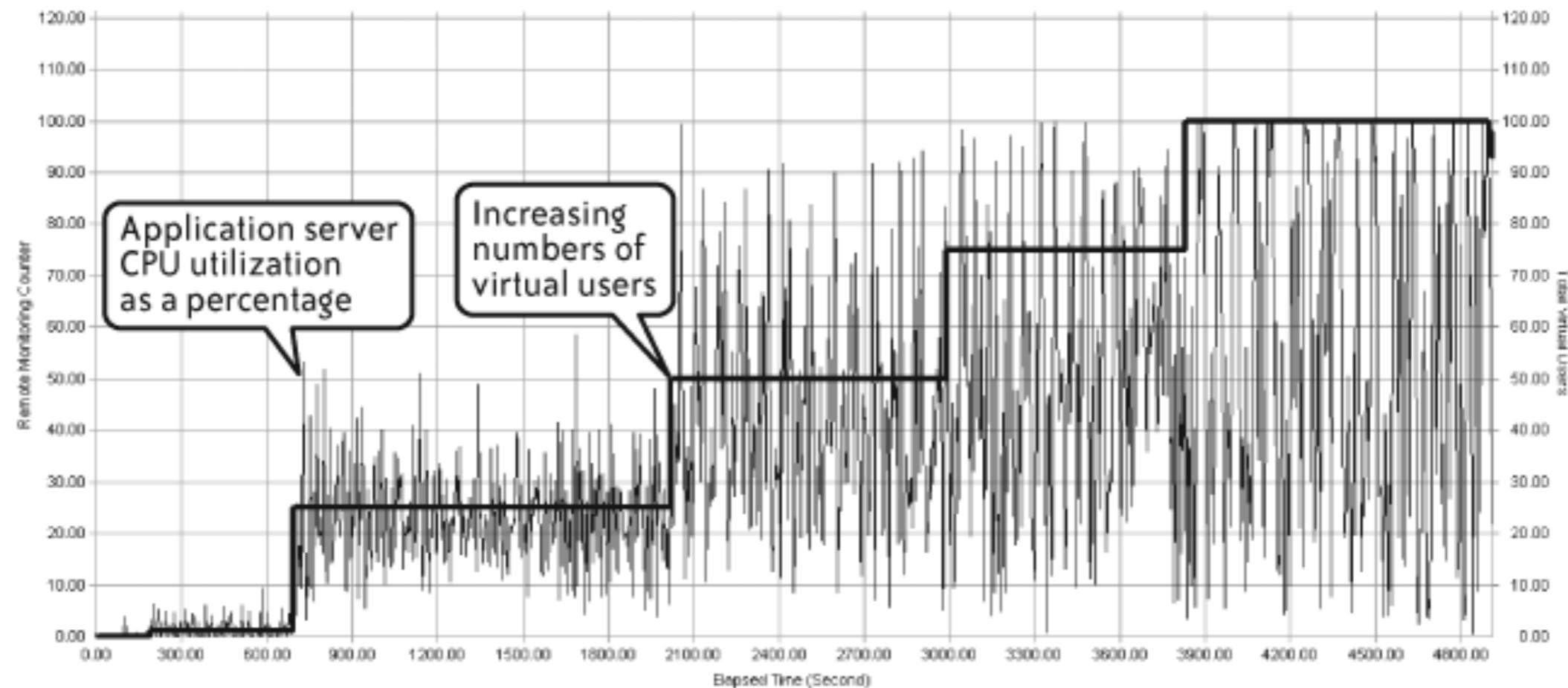


Server KPI performance

CPU usage
Memory usage



Concurrent virtual users correlated with database CPU utilization

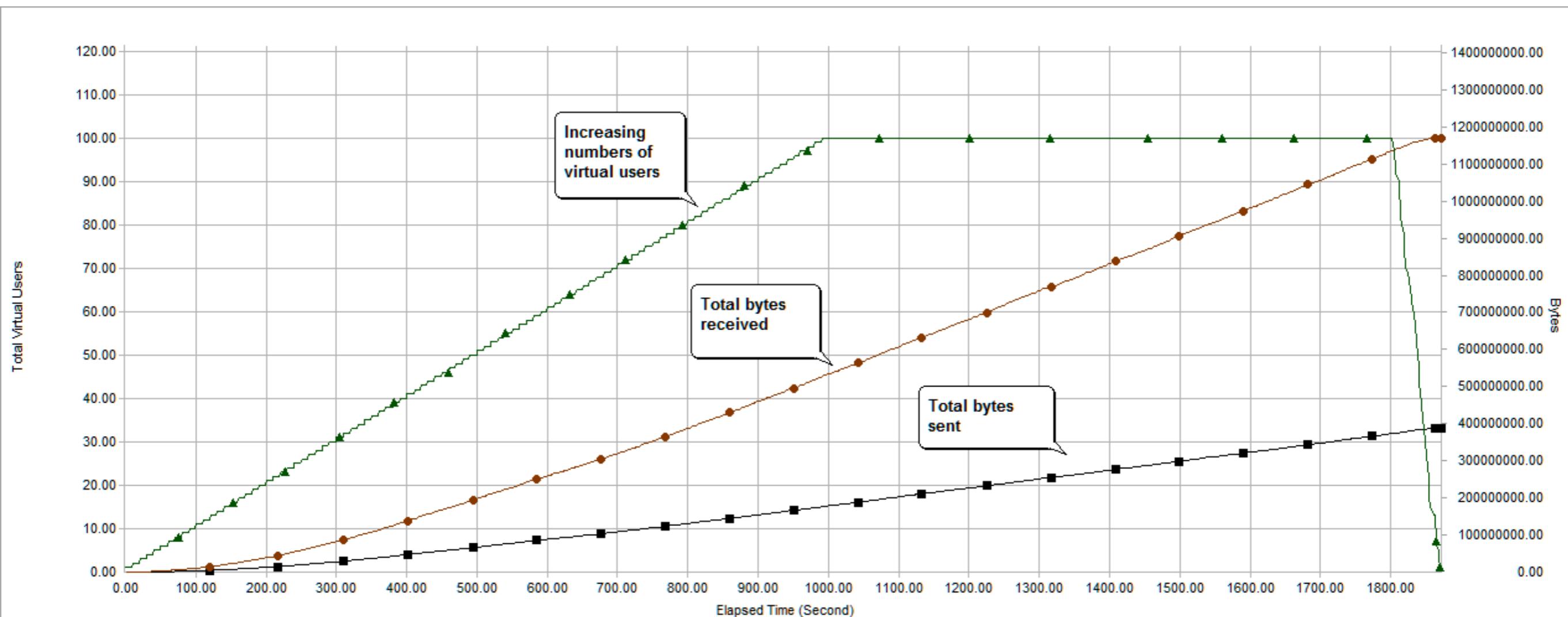


Network KPI performance

Size of data in request and response
Number of connection



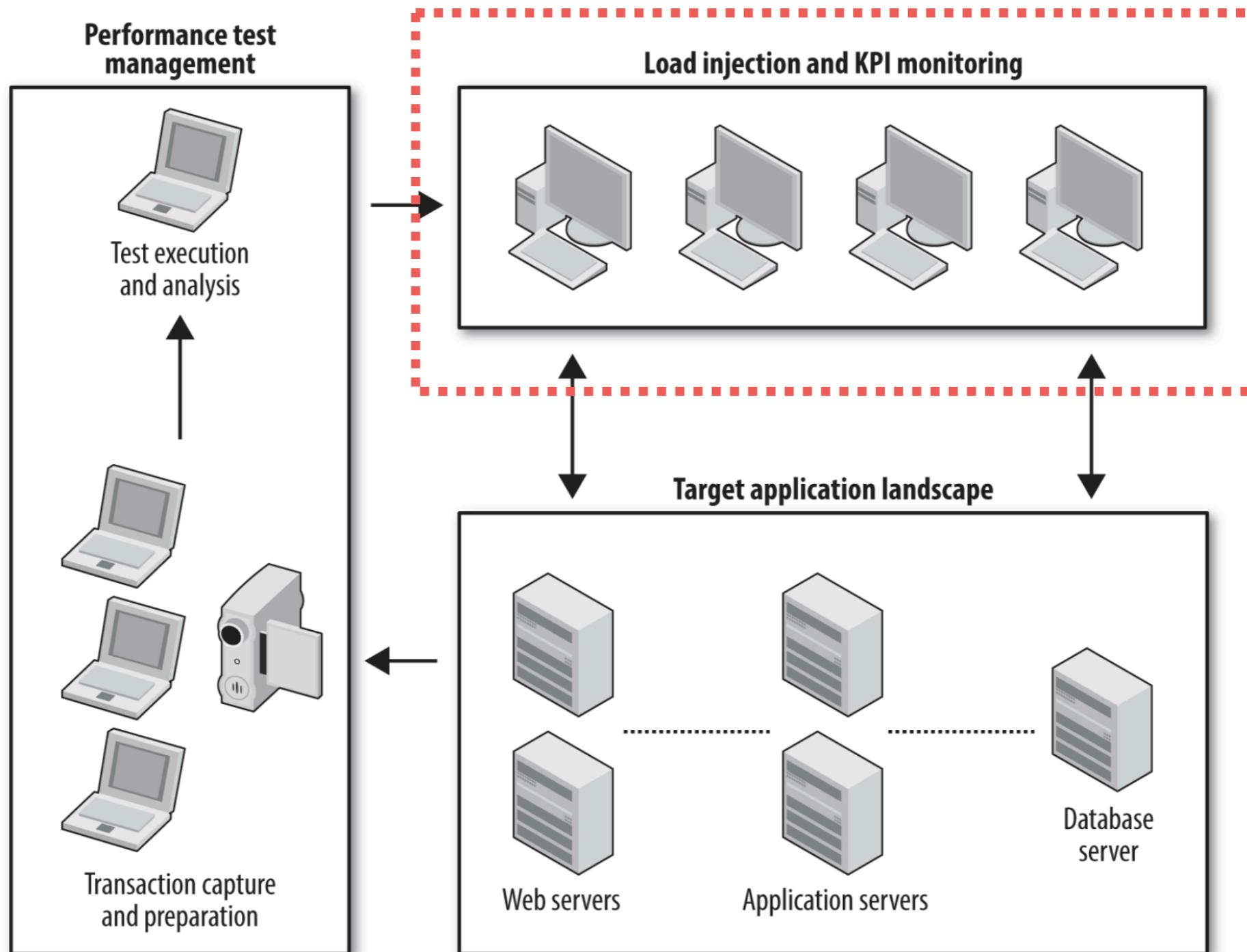
Network byte transfers correlated with concurrent virtual users



Server and Network KPI



Performance testing architecture



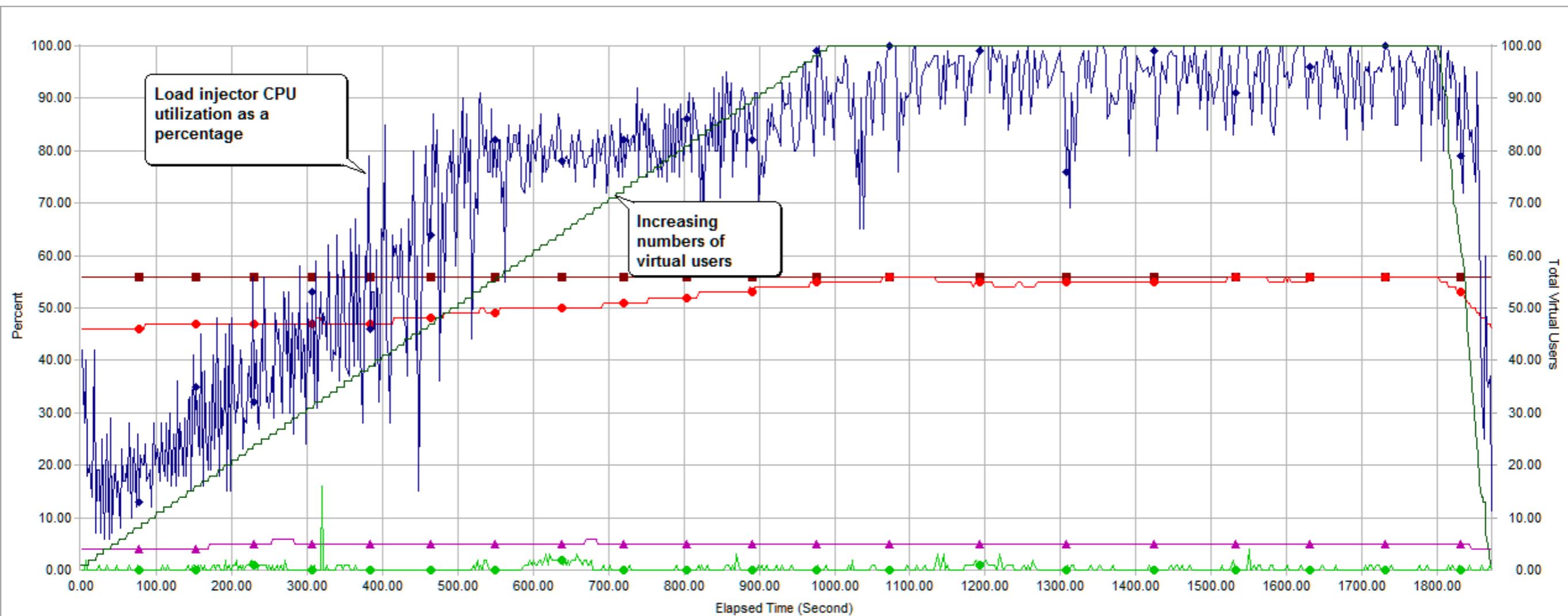
Load injector performance

- CPU usage
- Memory usage
- Disk usage
- File system

Better load injector make better virtual users



Load injector performance

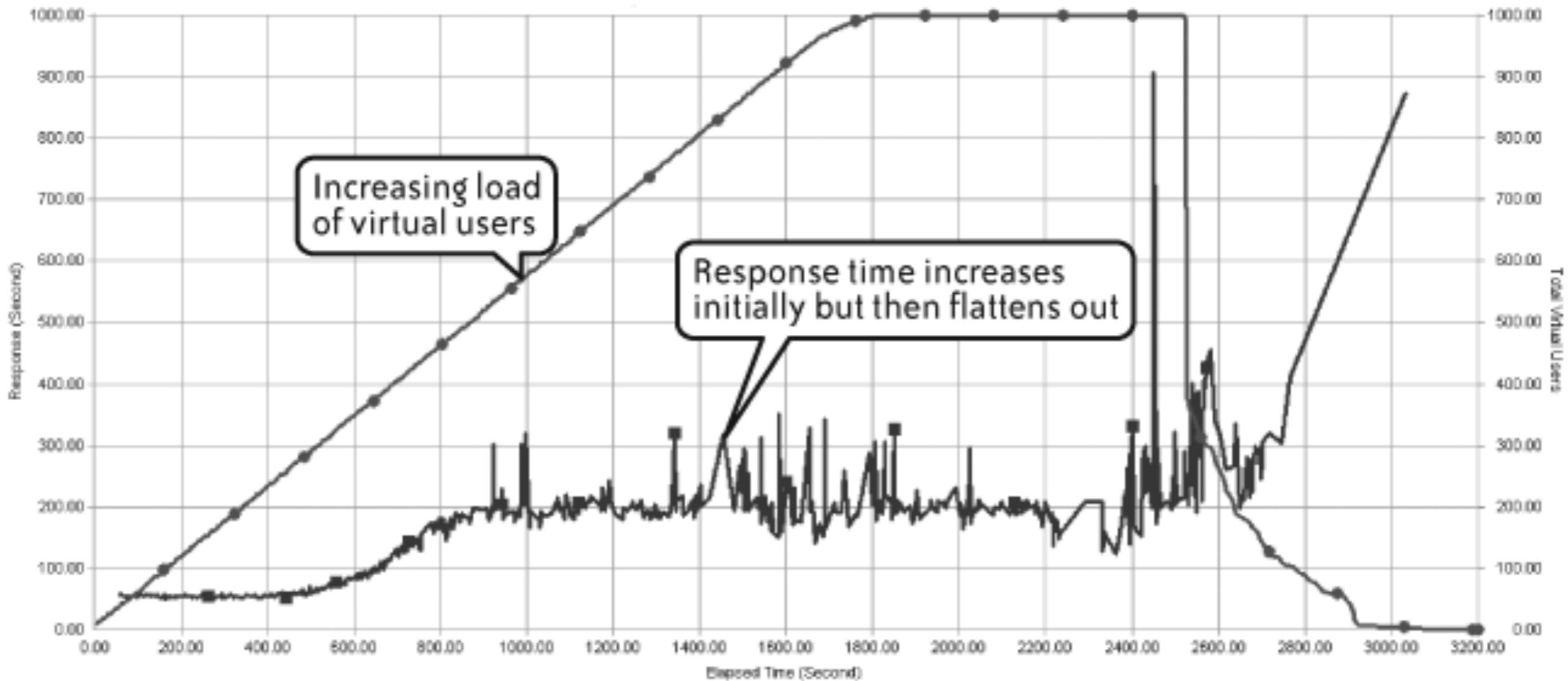


Root causes analysis



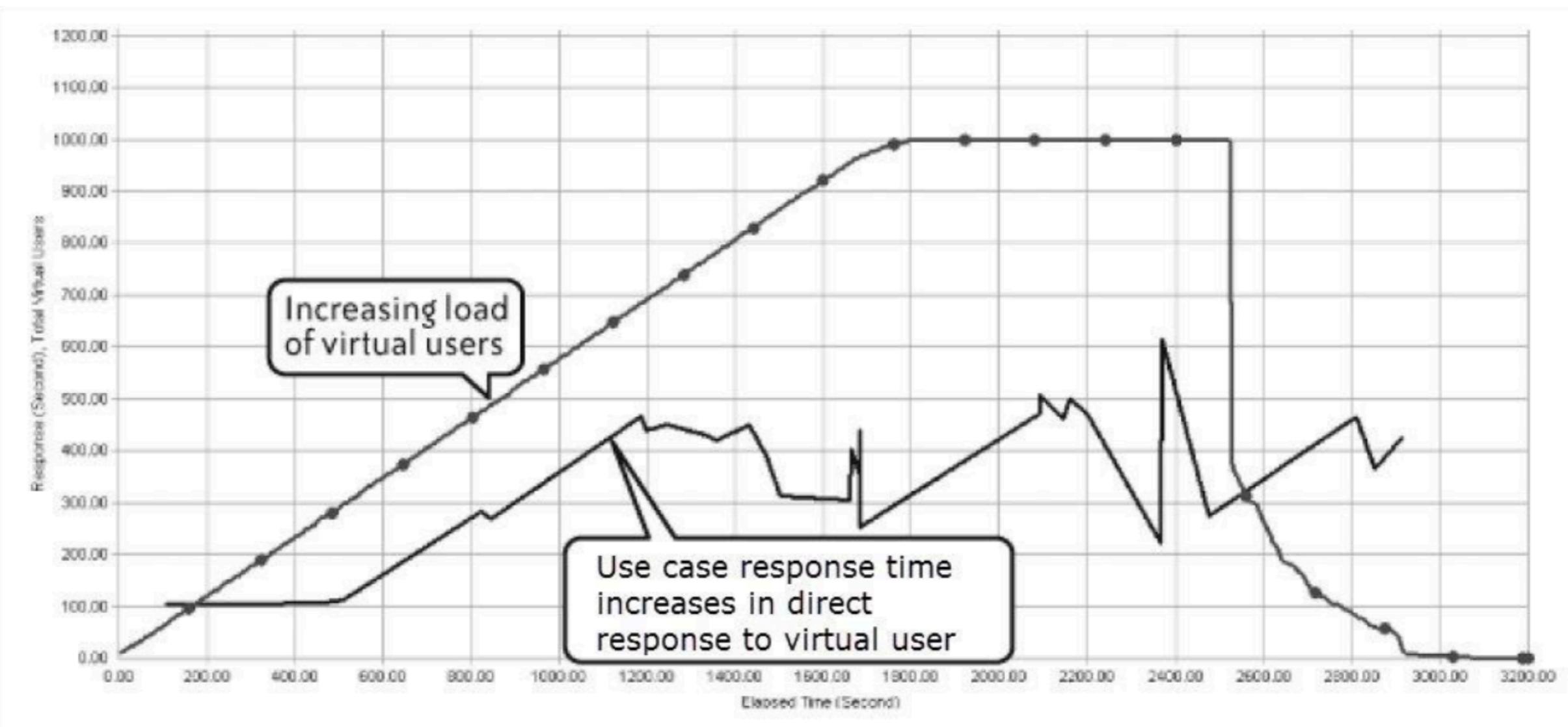
Scalability and response time

Good or bad ?

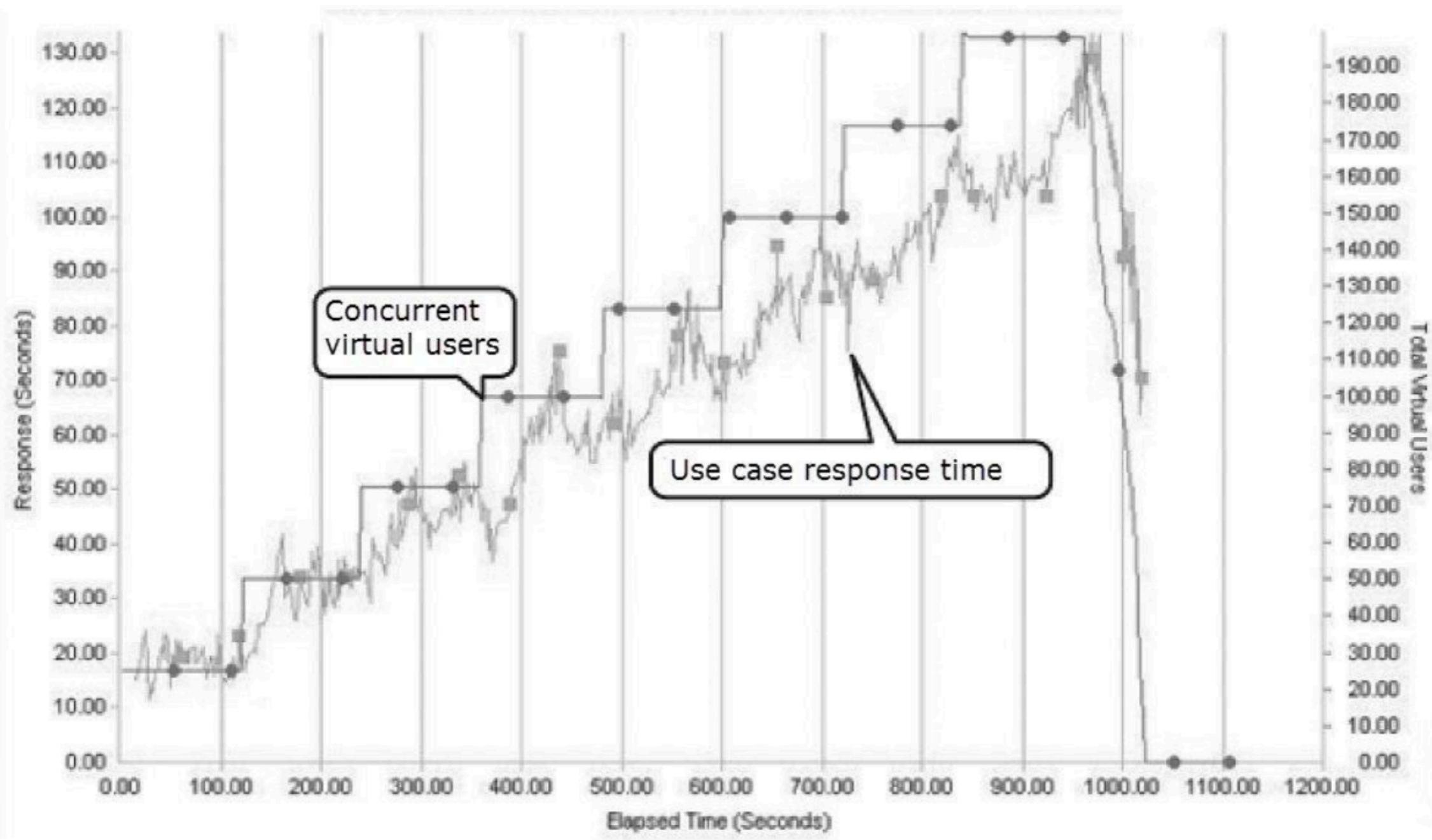


Scalability and response time

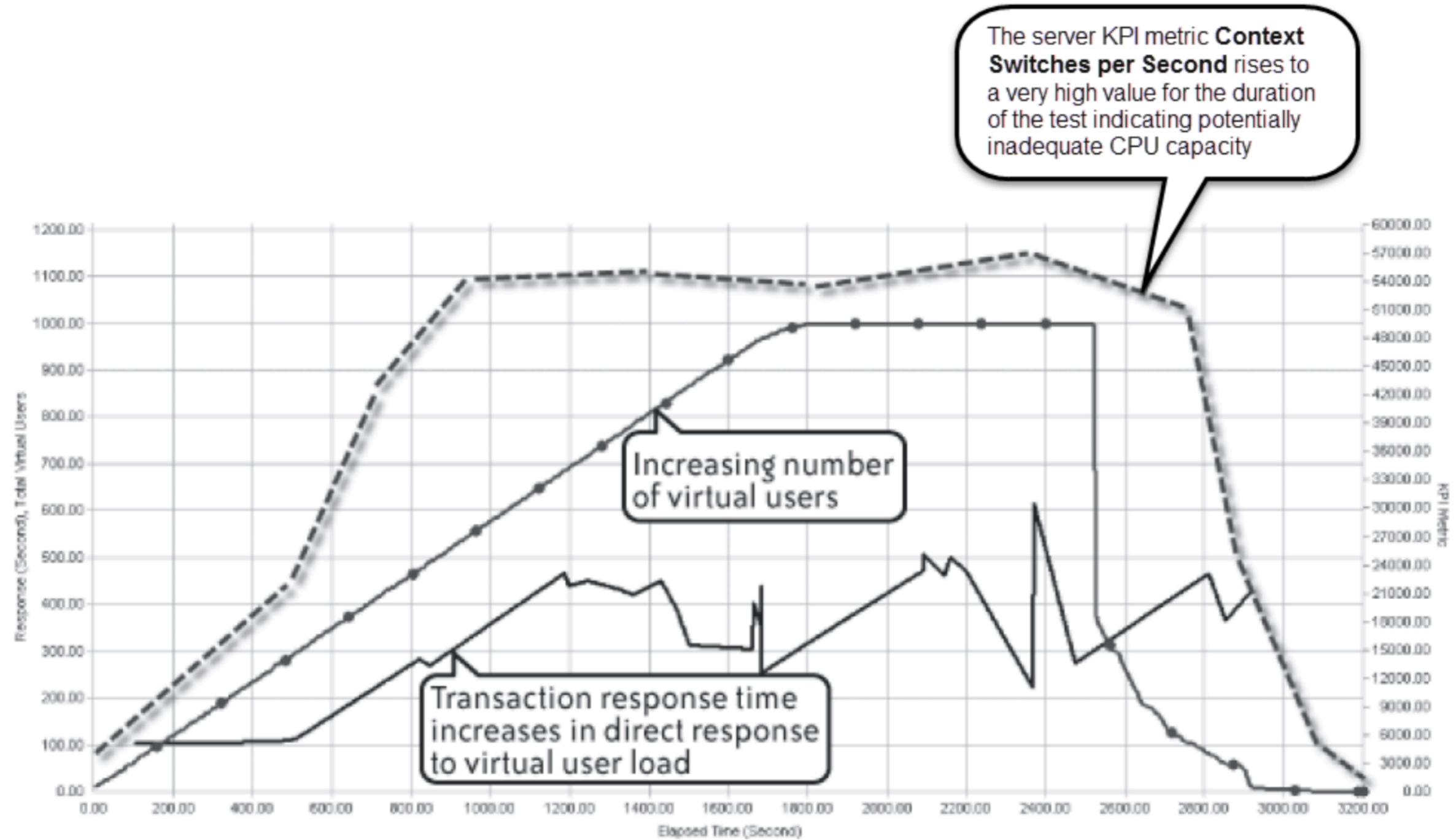
Good or bad ?



Good or bad ?

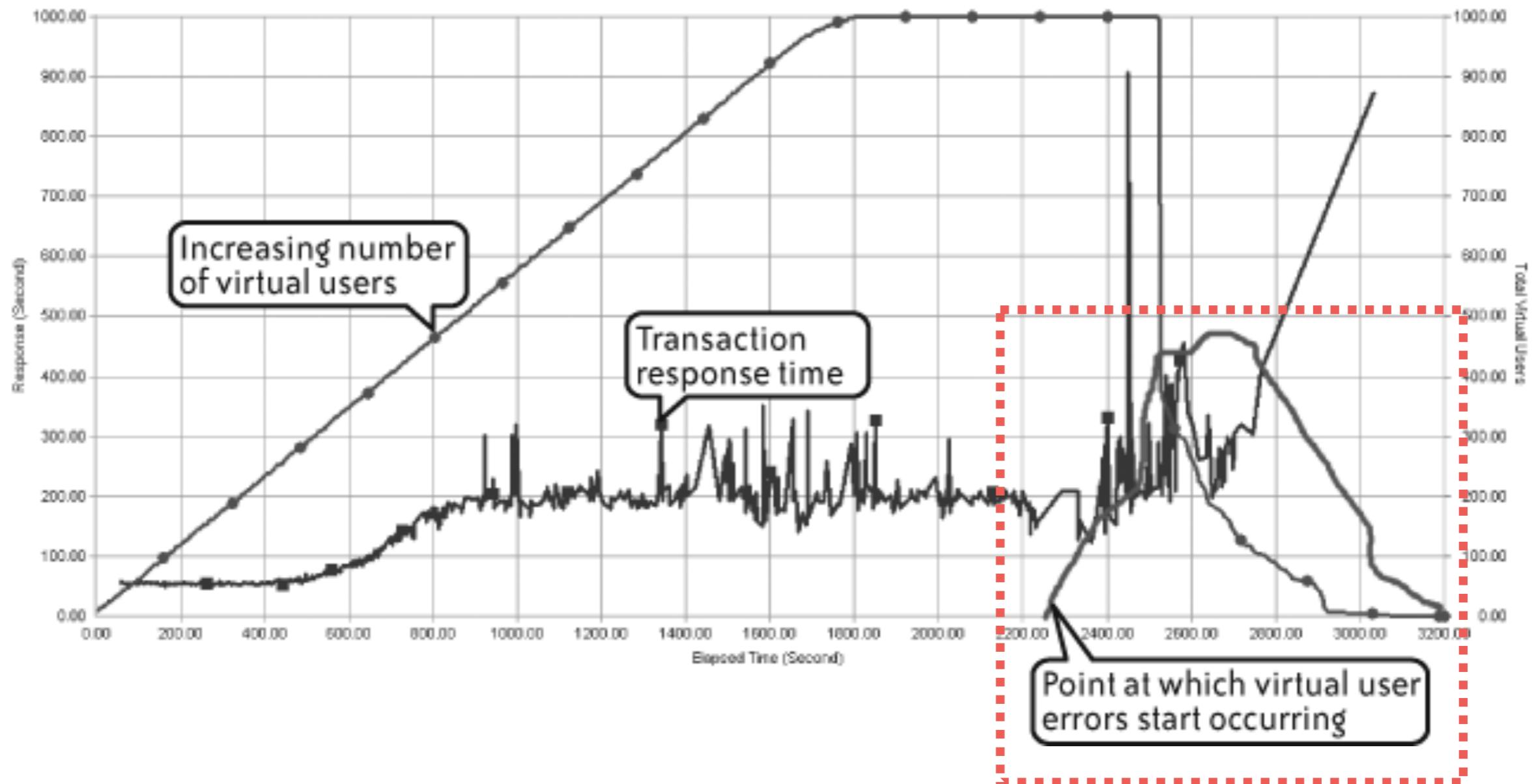


Good or bad ?



Error rate ?

Errors that occur during a performance test



Analysis checklists

Help to adopt a consistent approach to analysing the results



Analysis checklists

Pre-test tasks

During test execution tasks

Post-test tasks



Pre-test tasks

Configure the appropriate server/app/network KPI

How to test ?

Make sure that you can access the app from the
load injector

Configure of the load injector

Configure of 3-party tools and data



During test execution tasks

Make sure every **tests** are **executed**

Go or no go when **errors** are occurred

Go or no go when **throughputs** are dropped

CPU, memory, disk, network usage !!



Post-test tasks

Make sure you collect all relevant data

Backup all testing resources (script, input, result)

When producing report, make sure you map results
to the performance targets

