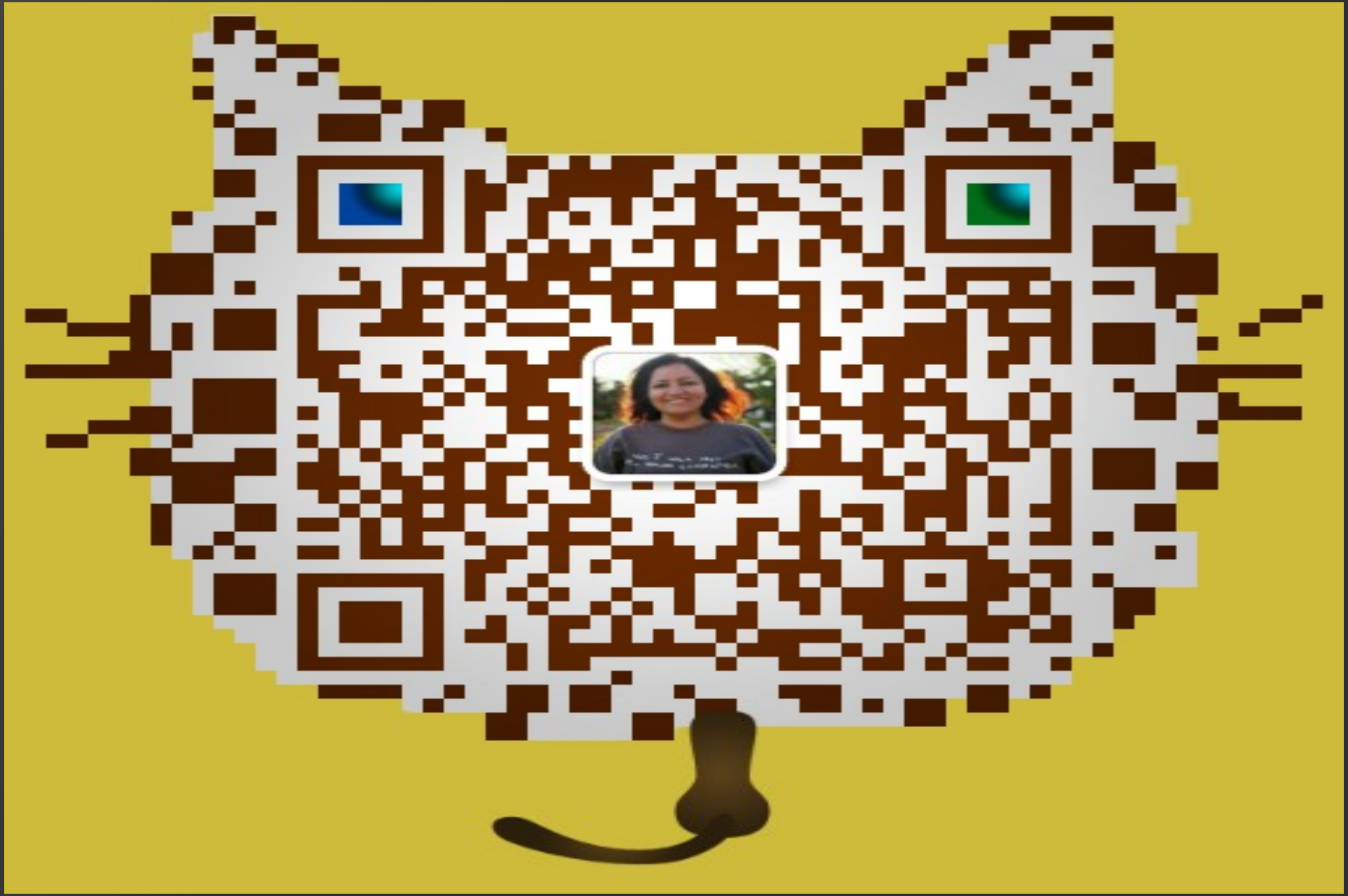
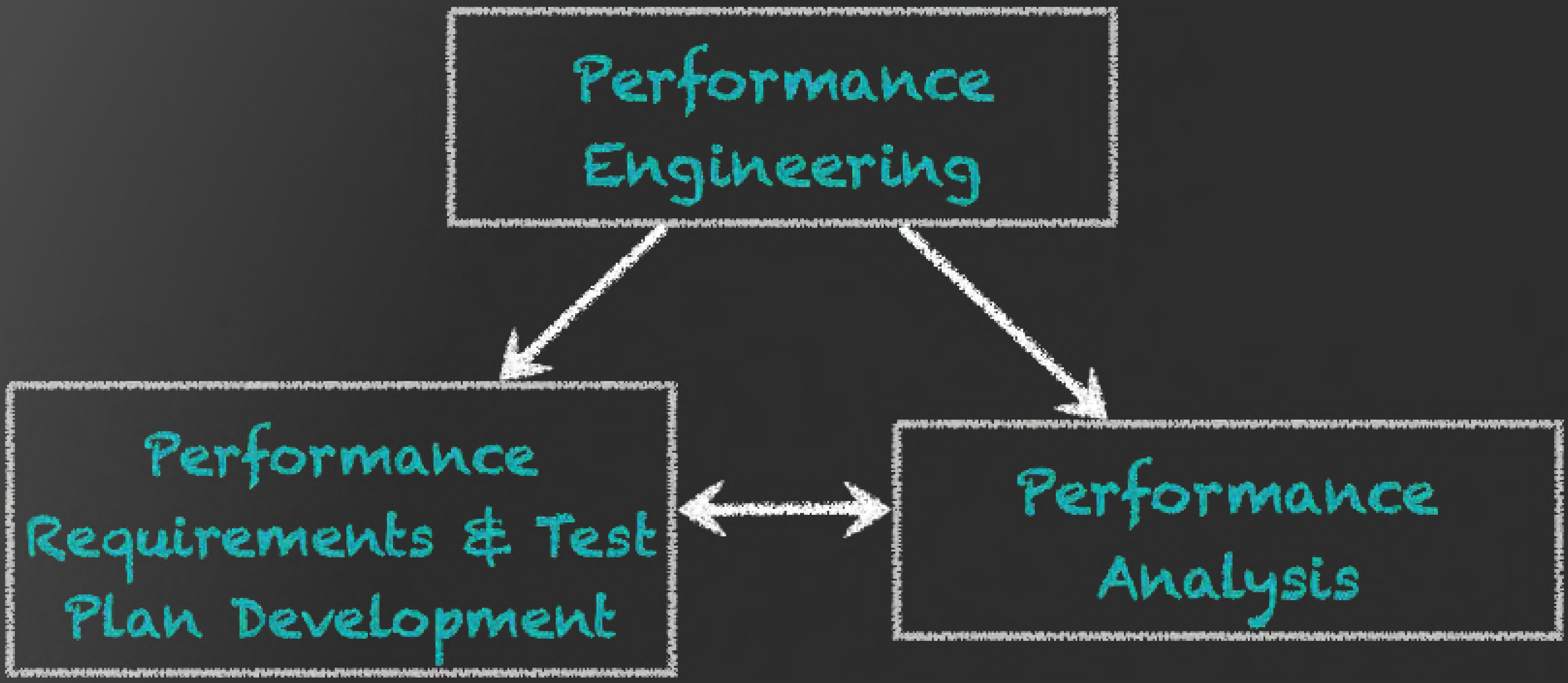
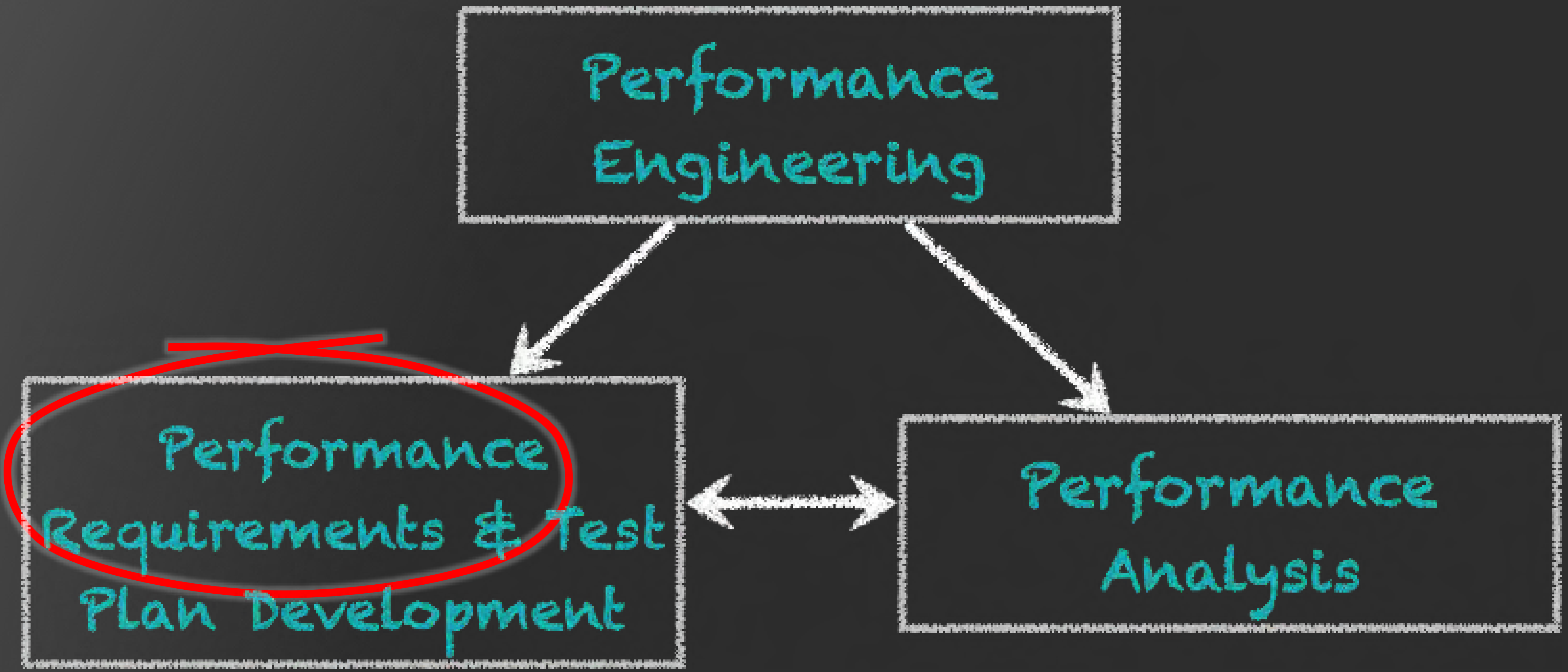
About Me



Performance Engineering



©2016 CodeKaram



©2016 CodeKaram

Performance Requirements

Ask Questions

- What makes the users happy?
- What makes them unhappy?
- Any current issues that need to be tackled?

©2016 CodeKaram

Define and Understand

- Understand QoS (Quality of Service) for end users
- Define QoS success metrics in measurable terms
  - Those are the service level agreements (SLAs)!

©2016 CodeKaram

Define Success!

- Document, understand and prioritize SLAs
    - Throughput
    - Response time
    - Capacity
    - Footprint
    - CPU utilization?
    - …

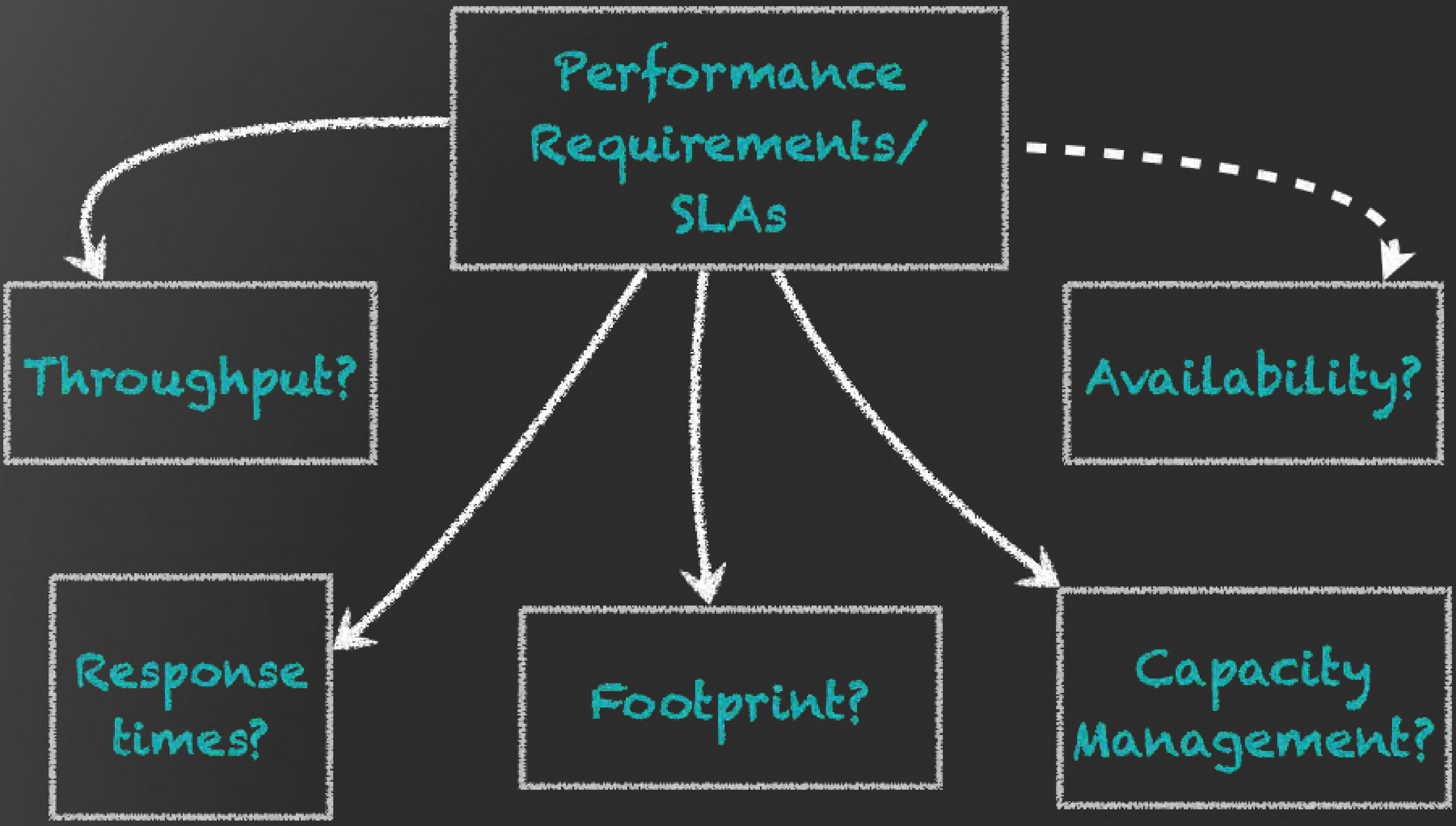
©2016 CodeKaram

Work For It!

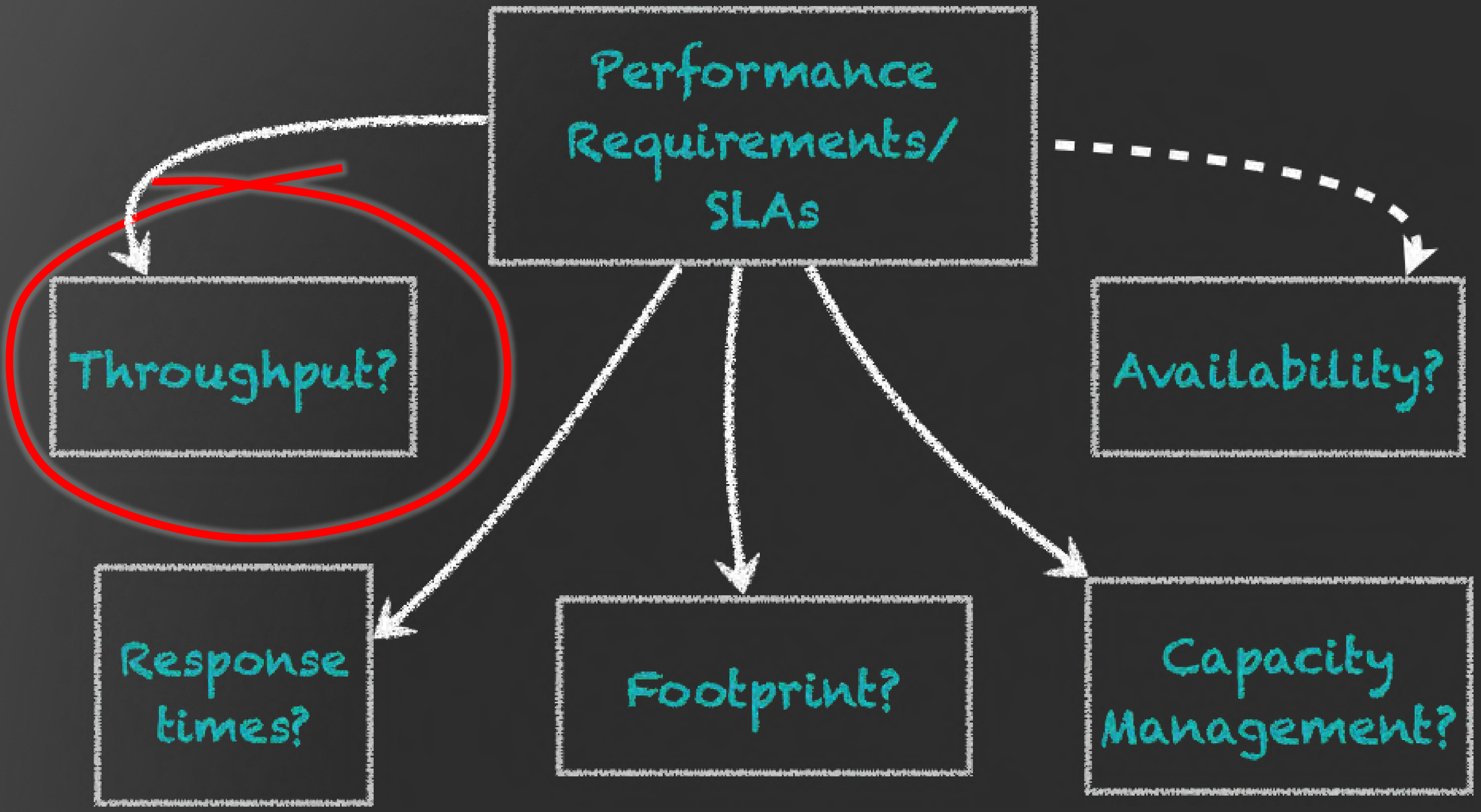
Monitor, measure and define performance in terms of throughput, latency, capacity, footprint, utilization ...

©2016 CodeKaram

Defining Success!



©2016 CodeKaram



©2016 CodeKaram

Throughput

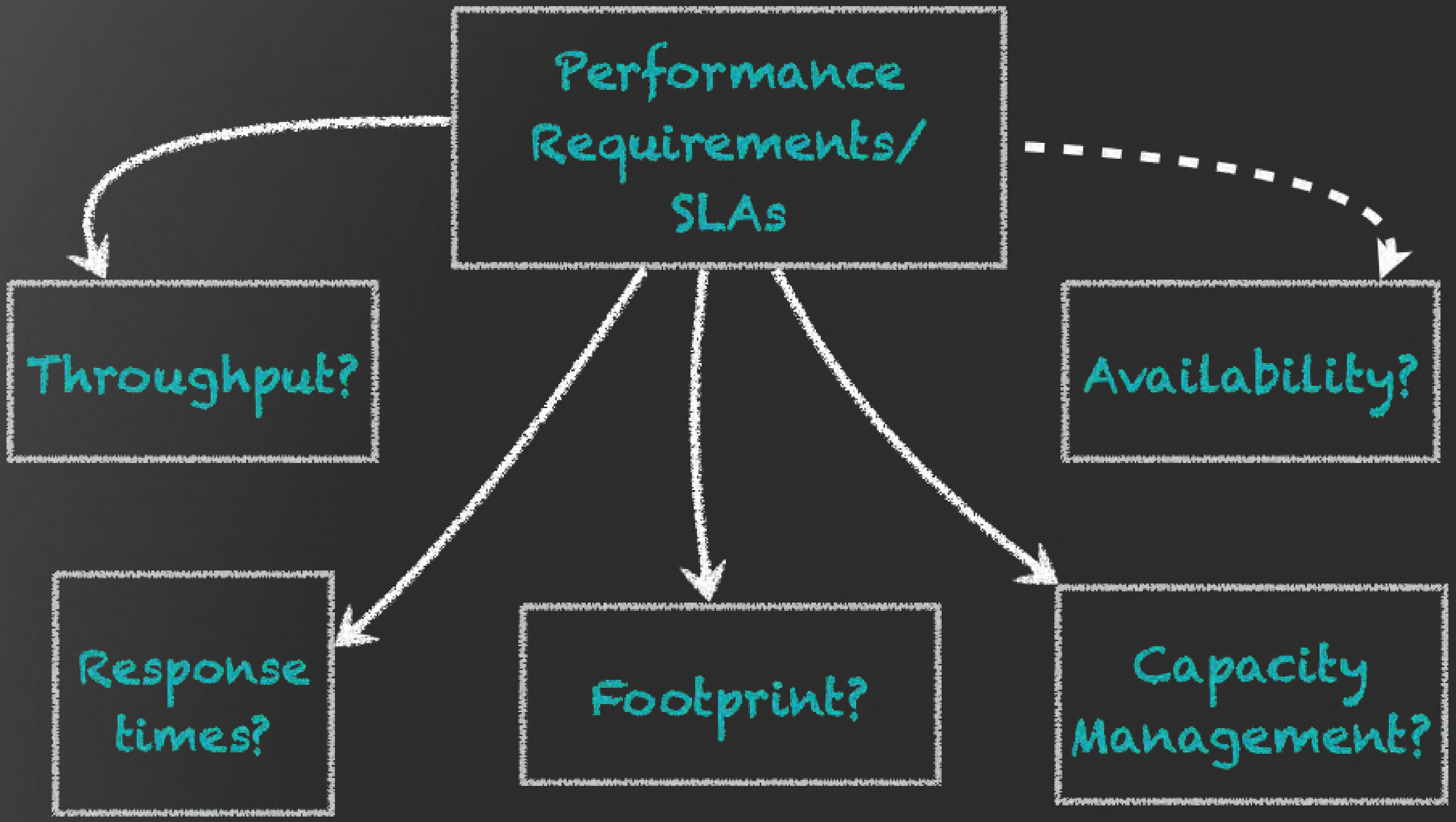
- Expected throughput?
  - Can you fall below the expected throughput?
  - How long can you stay below the expected throughput?
  - What is the lowest that you can go?

©2016 CodeKaram

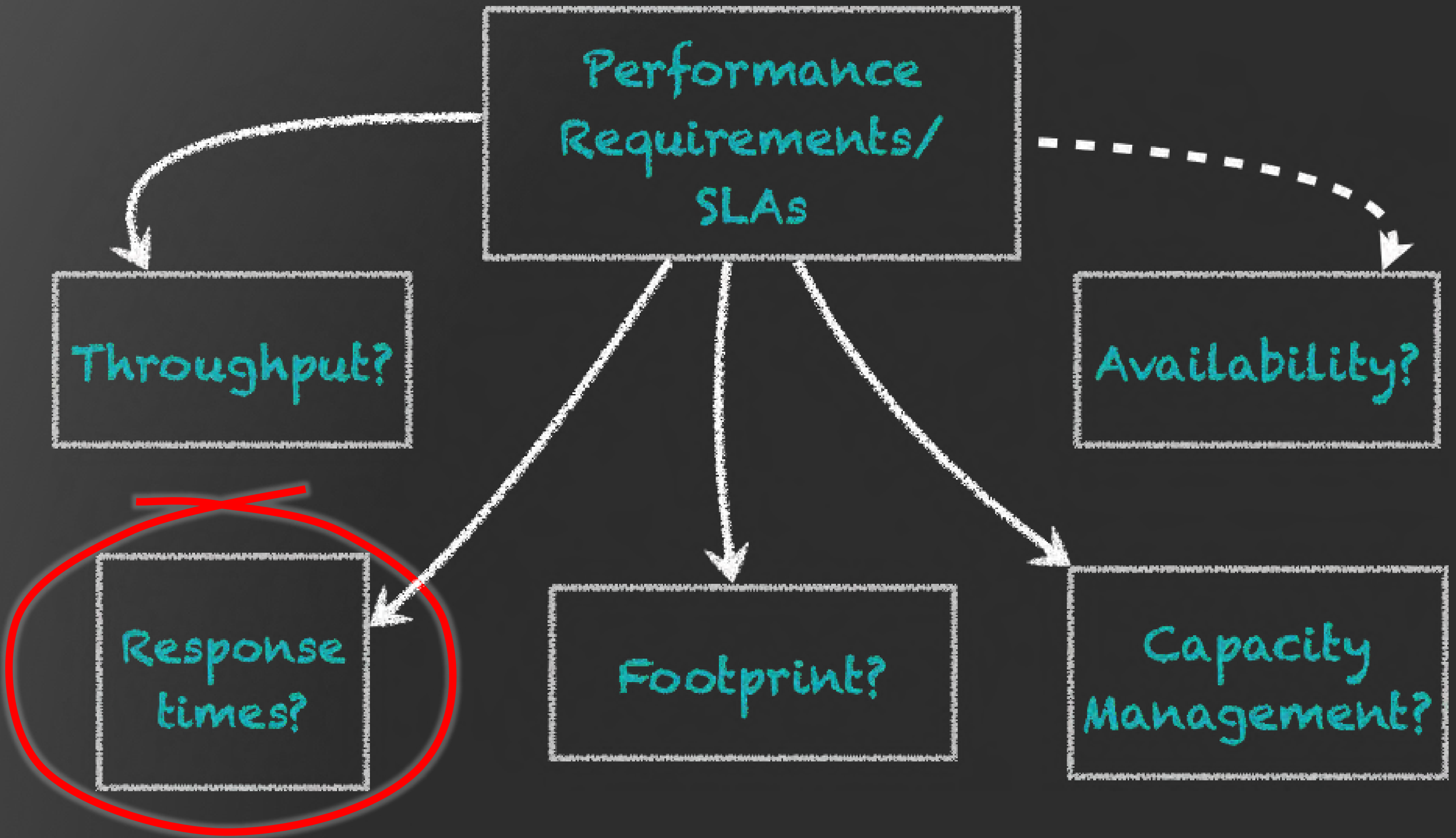
Throughput

- Throughput measurement
  - How is it measured?
    - Transactions/sec; messages/sec or all of them?
  - Where is it measured?
    - Client-side; server-side; browser?

©2016 CodeKaram



©2016 CodeKaram



©2016 CodeKaram

Response Time

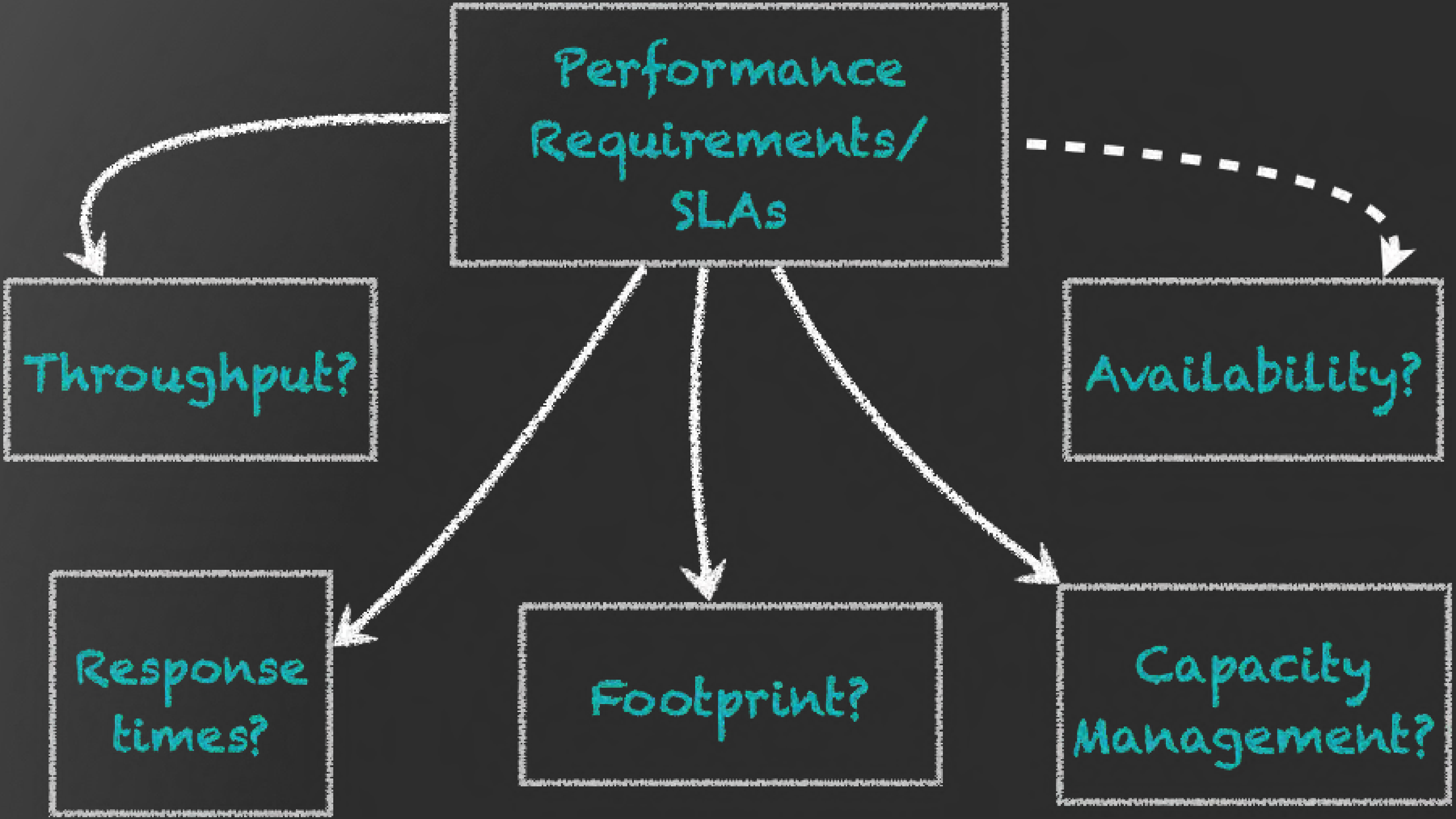
- Expected response time?
  - Can you go above the expected response time?
  - How long can you stay there?
  - How much can you exceed?

©2016 CodeKaram

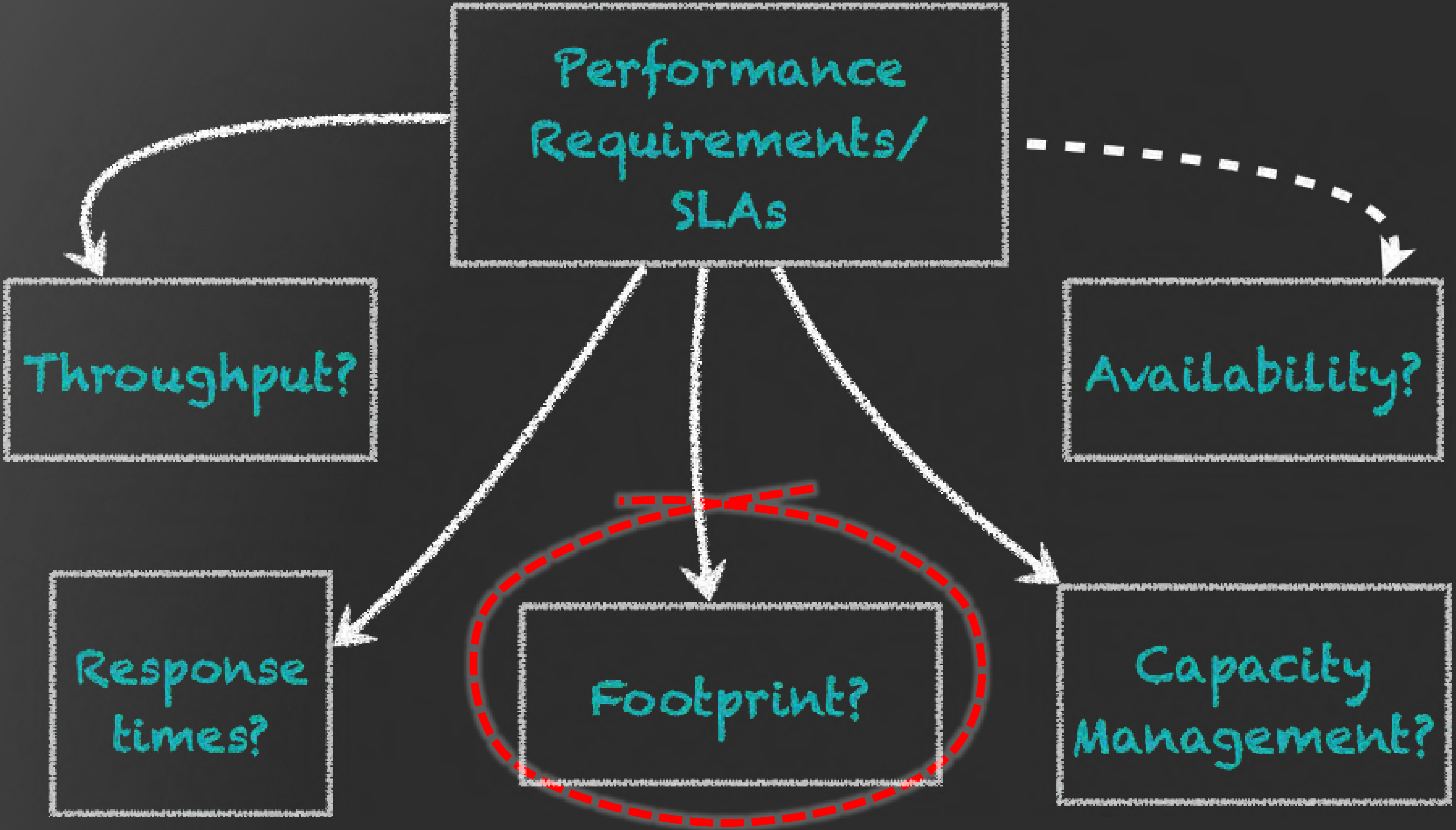
Response Time

- Response time measurement
  - How is it measured?
    - 99th percentile; 5-9s; worst-case; or all?
  - Where is it measured?
    - Client-side; server-side? Complete loop?

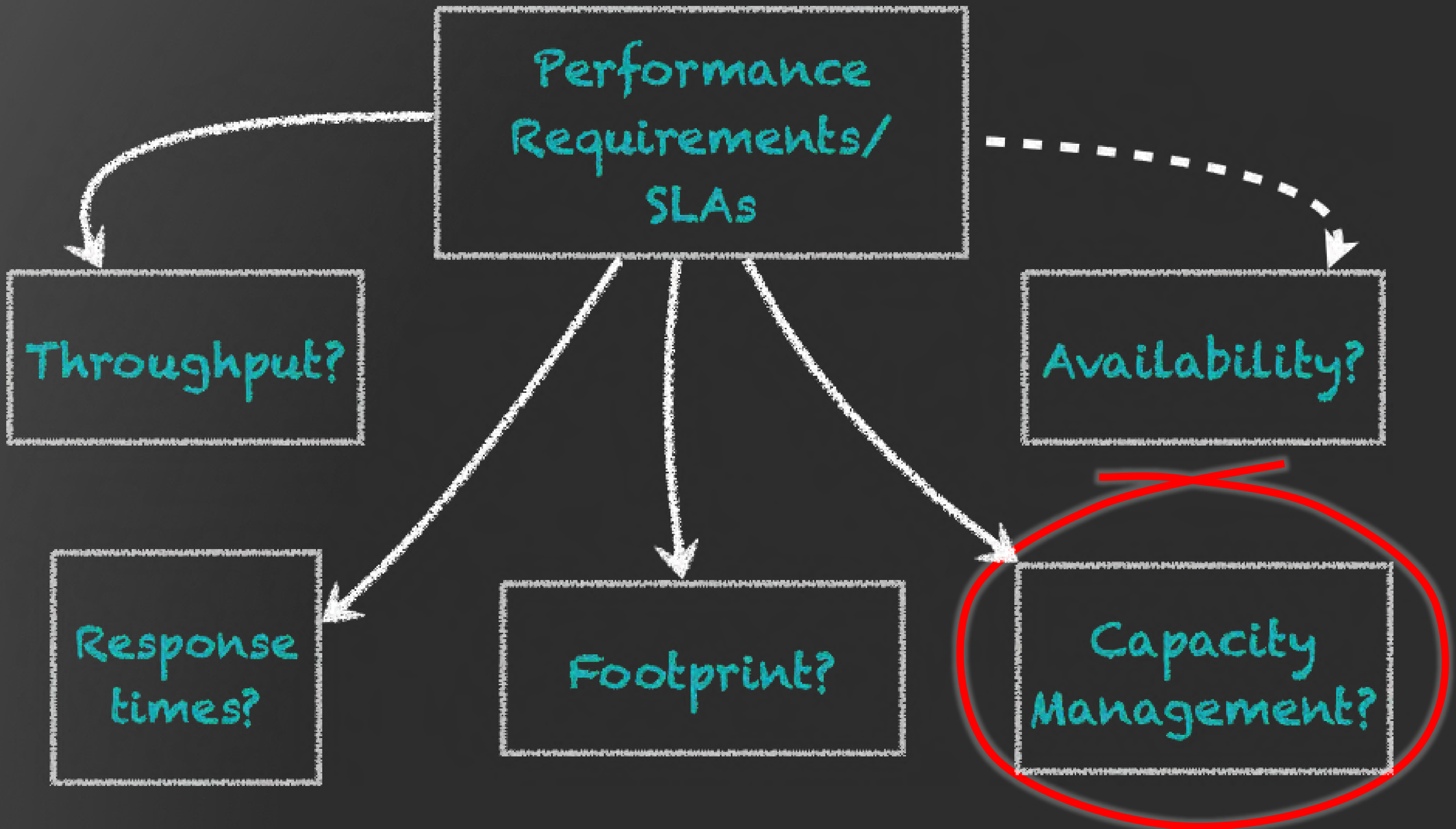
©2016 CodeKaram



©2016 CodeKaram



©2016 CodeKaram



©2016 CodeKaram

Capacity Management

- What is the expected load?
- What happens if one system gets loaded more than others?

  (load balancer issue)
- How is it measured?

©2016 CodeKaram

Capacity Management

- What’s the max load that a system and all systems can tolerate?
- How long?
- What metrics are being captured?

©2016 CodeKaram

Performance Analysis

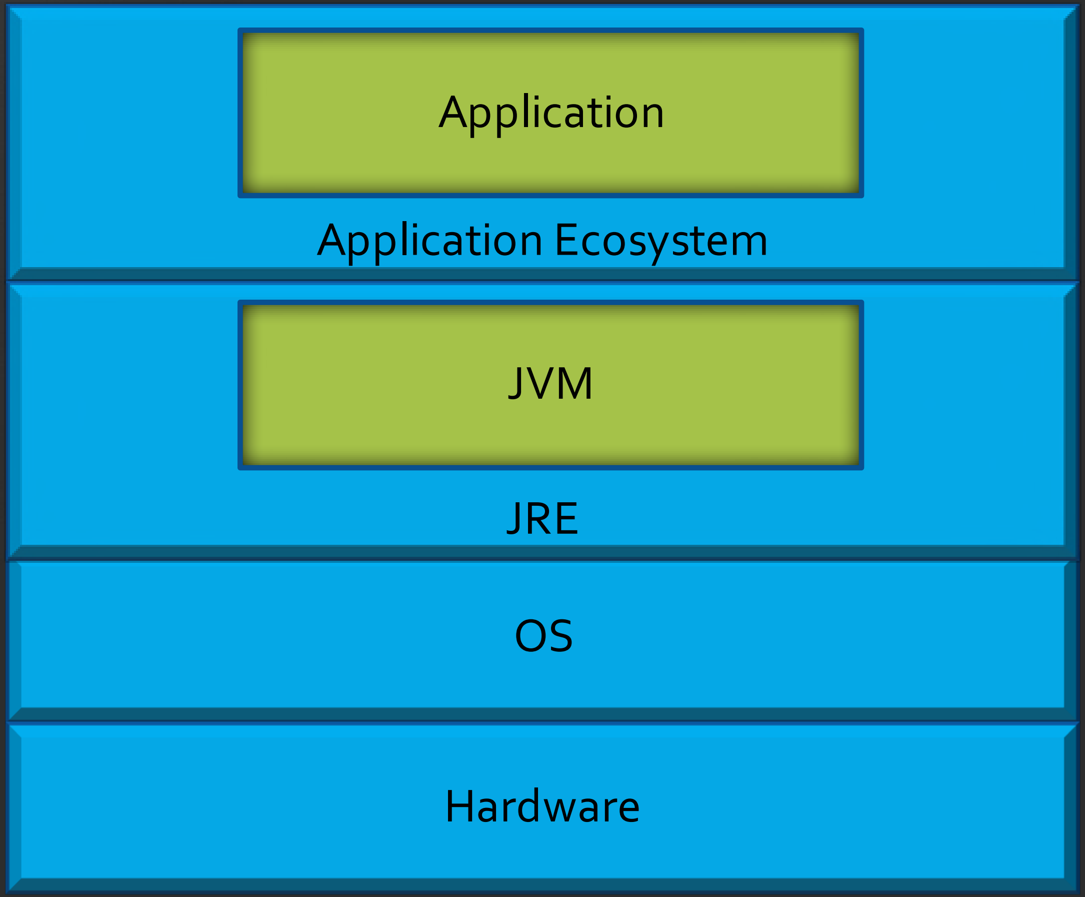
$\overset{\text{Java}}{\wedge}$ Performance Analysis

Performance Analysis

- Analyze what factors enable the end-user experience to meet or exceed the promised QoS
- Track your SLAs!

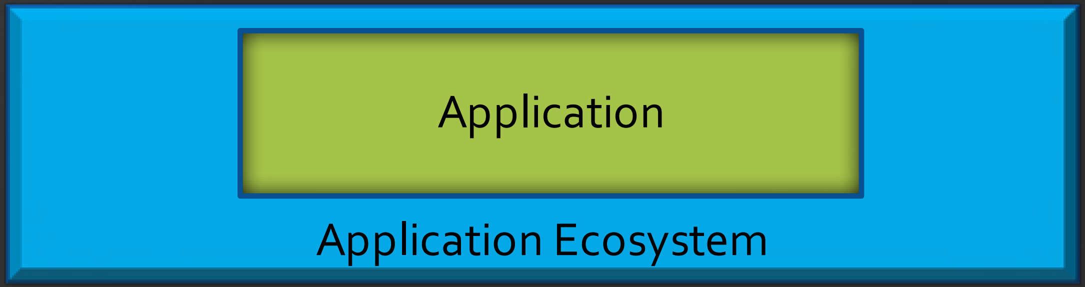
©2016 CodeKaram

Java Application Stack



©2016 CodeKaram

Application Performance Analysis



- Application services
- Application server
- Database
- Any other services in the ecosystem?

©2016 CodeKaram

JVM + Runtime Performance Analysis

- Classloading stats
- JIT Compilation stats
- Garbage Collection stats
- Threads stats

©2016 CodeKaram

OS Performance Analysis

- System/ Kernel stats
- Lock stats
- Threads stats

©2016 CodeKaram

Hardware Performance Analysis

- Memory bandwidth/ traffic/ consumption
- CPU/ core utilization
- CPU cache efficiency/ utilization/ levels
- Architectural specific?
- IO Stats

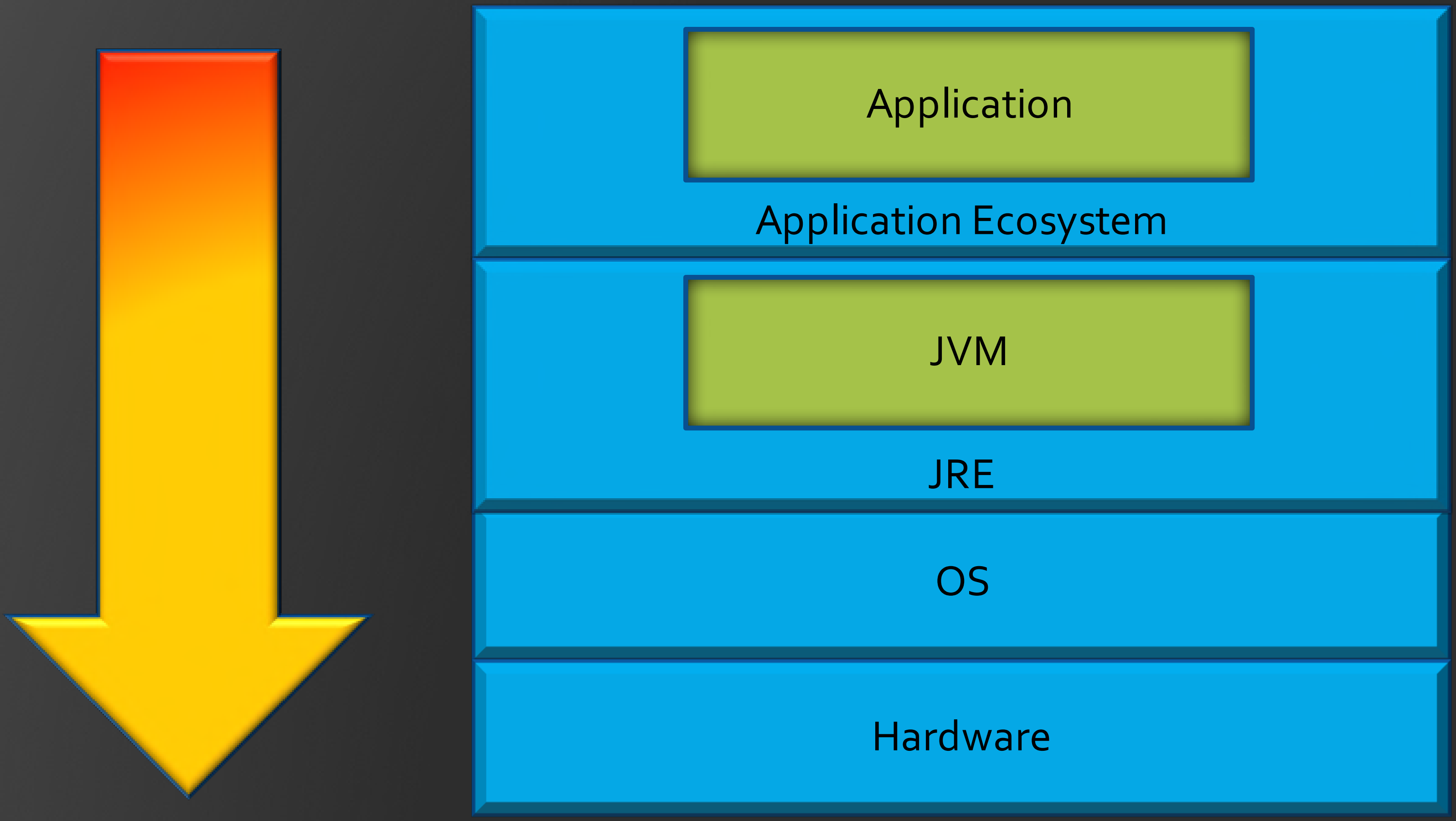
©2016 CodeKaram

What are You Trying to Achieve?

Improve application?

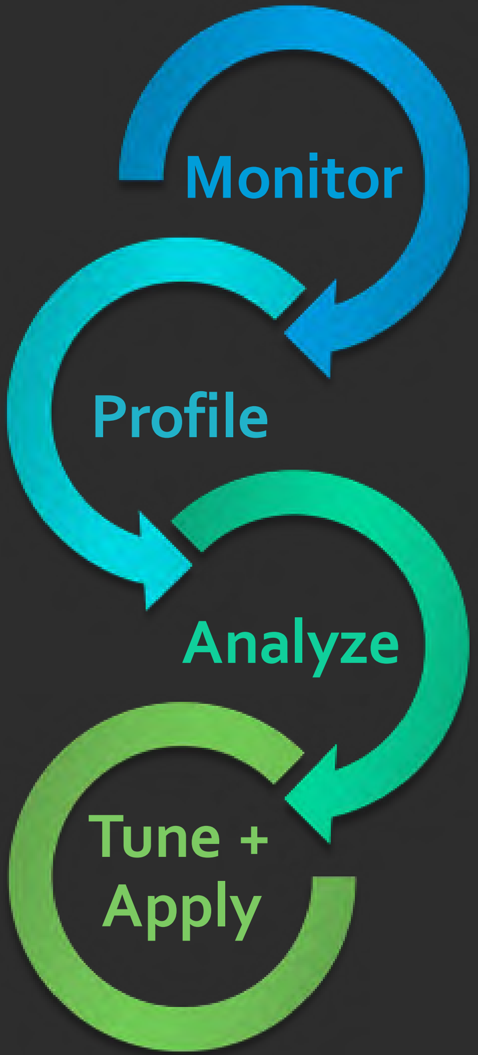
©2016 CodeKaram

Top-Down Approach



©2016 CodeKaram

Top Down Approach - Process



- Step 1
- Step 2
- Step 3
- Step 4

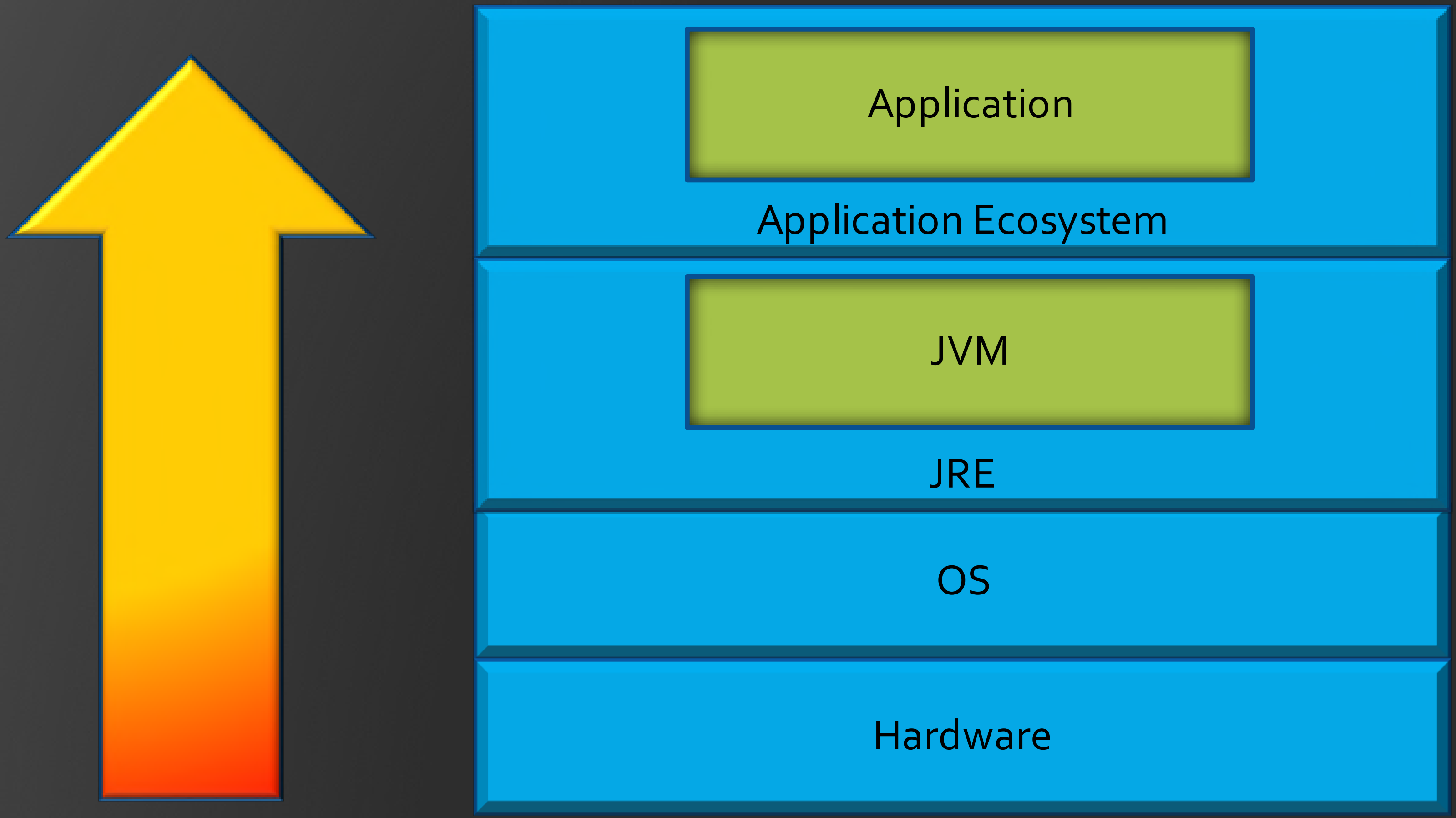
©2016 CodeKaram

What are You Trying to Achieve?

Improve the platform?

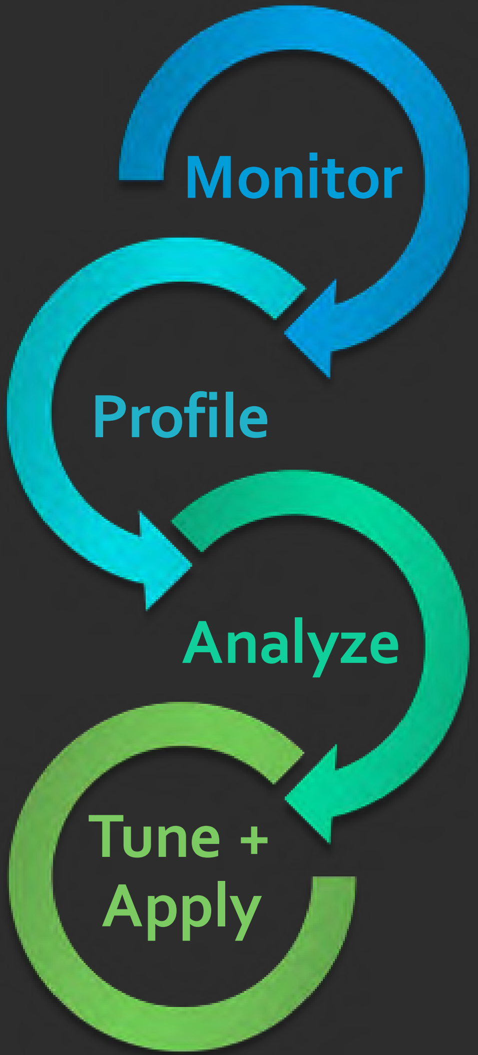
©2016 CodeKaram

Bottom-Up Approach



©2016 CodeKaram

Bottom Up Approach - Process



- Step 1
- Step 2
- Step 3
- Step 4

©2016 CodeKaram

Top-Down Approach

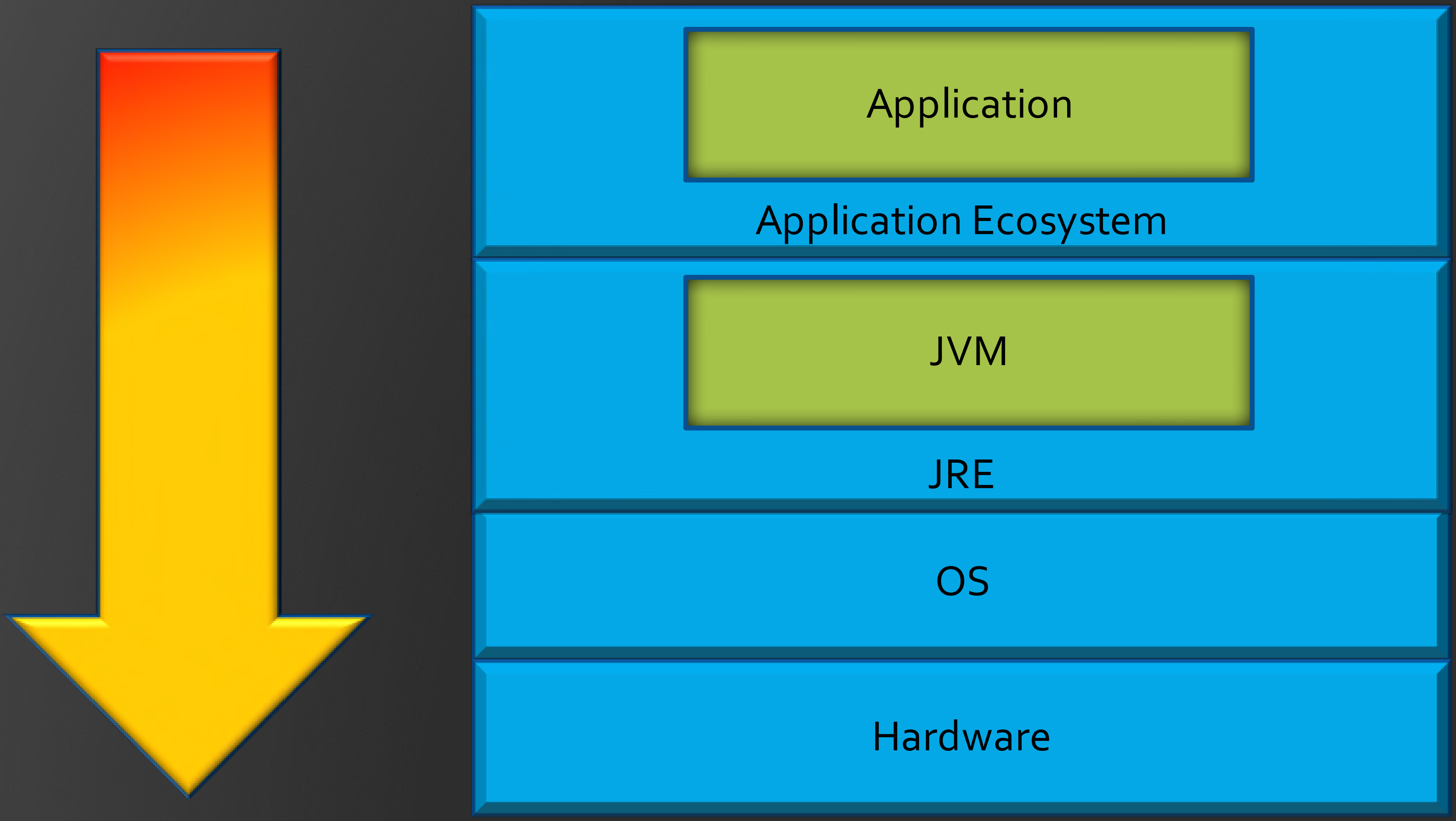
Top-Down Approach

I HAVE the power!!

... to modify the code

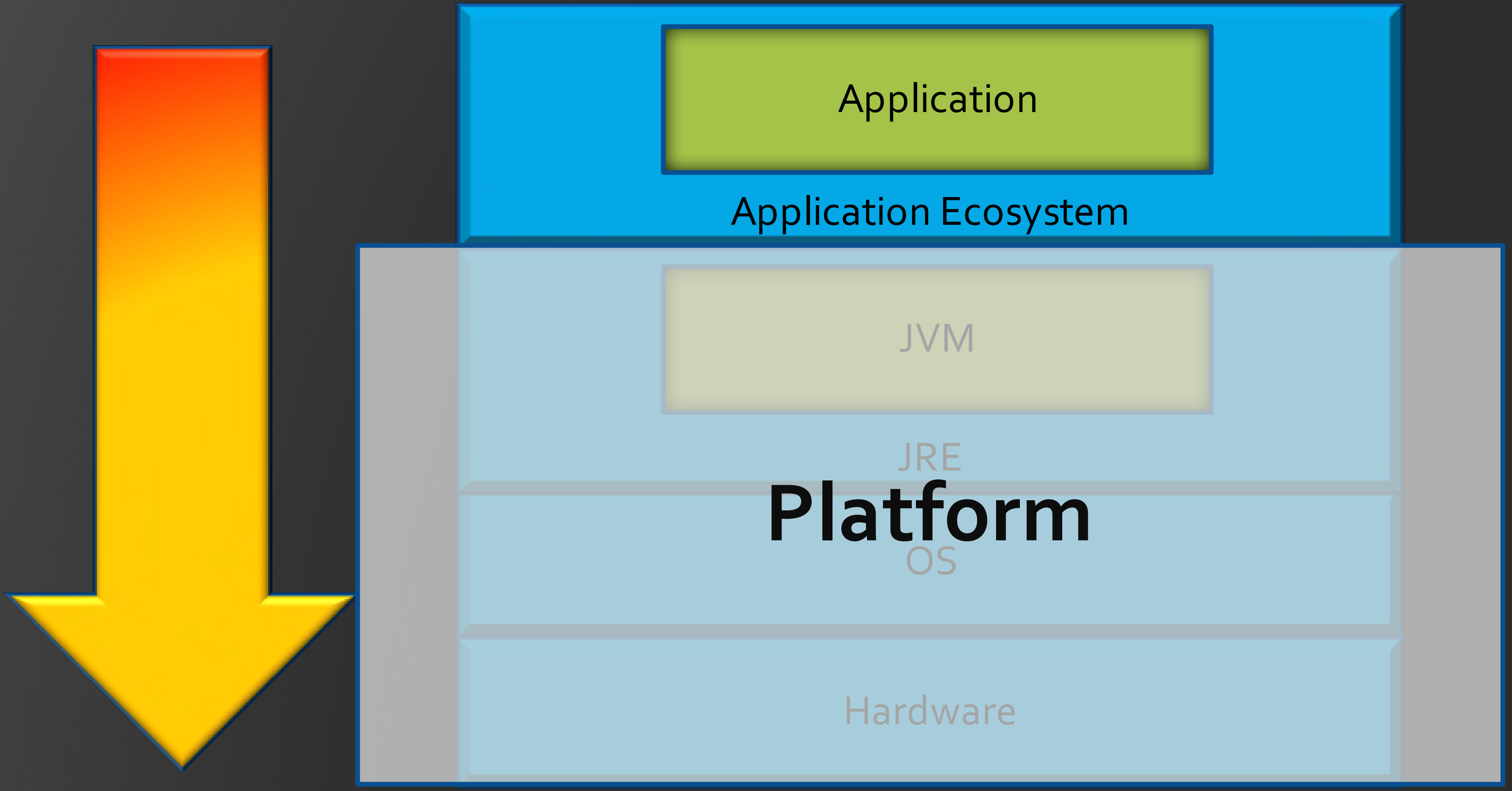
©2016 CodeKaram

Top-Down Approach



©2016 CodeKaram

Top-Down Approach



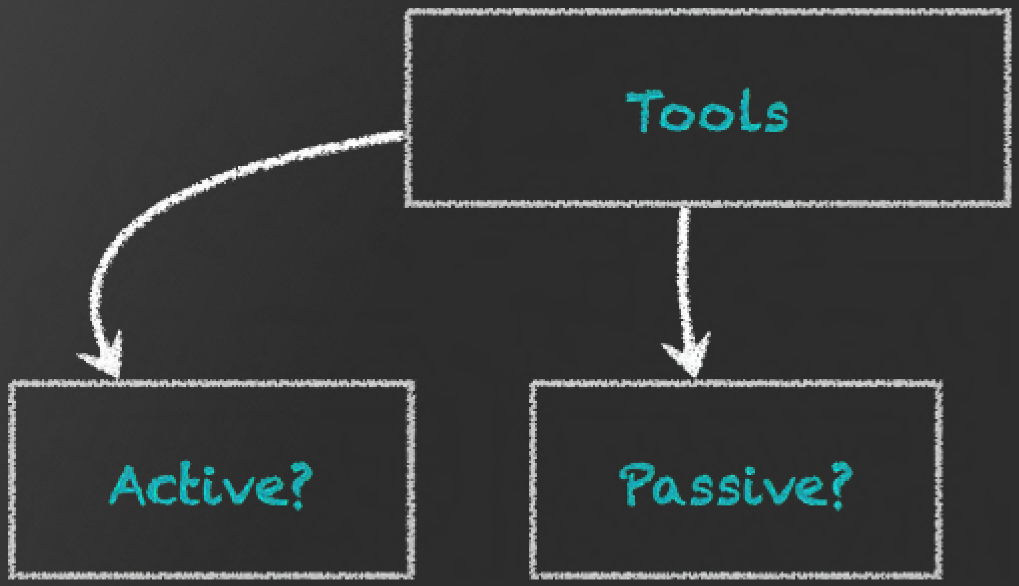
©2016 CodeKaram

Step 1: Monitor

Monitor and plot SUT (System Under Test) statistics

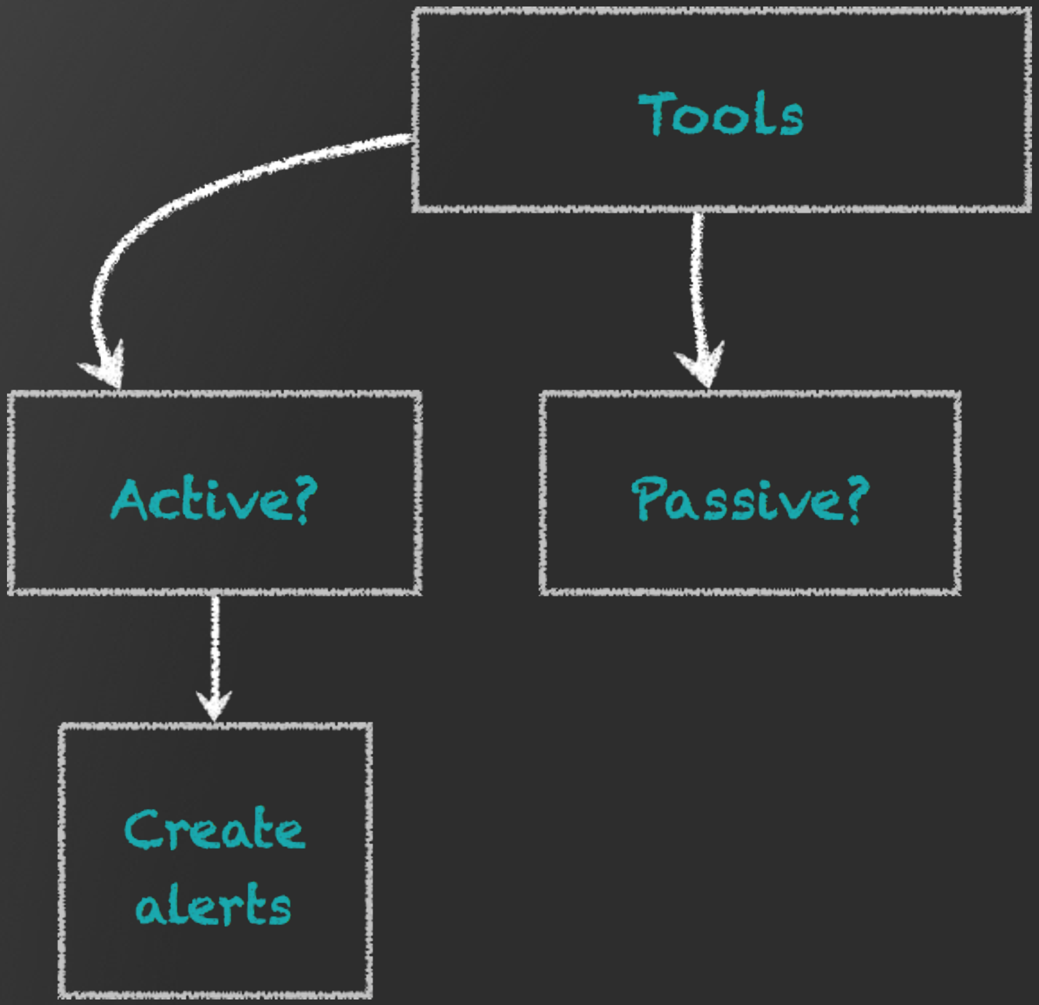
©2016 CodeKaram

Step 1: Monitor



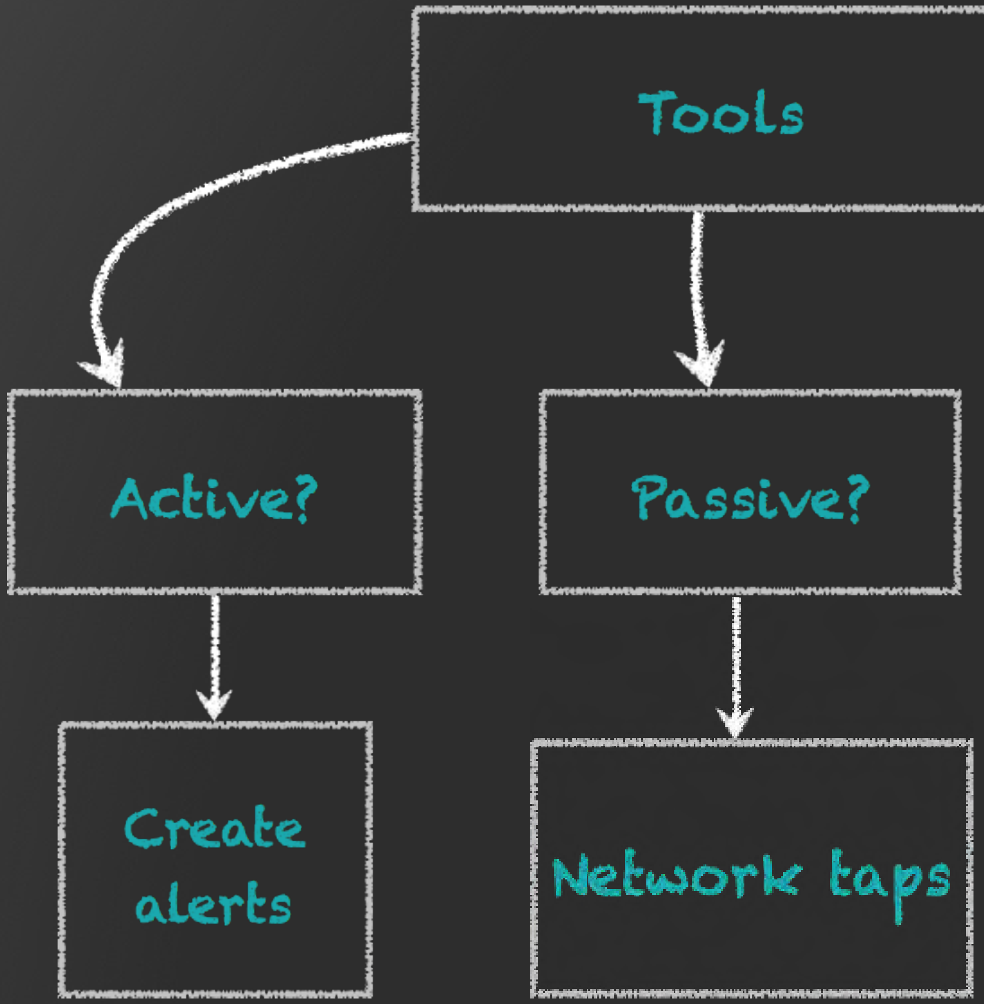
©2016 CodeKaram

Step 1: Monitor



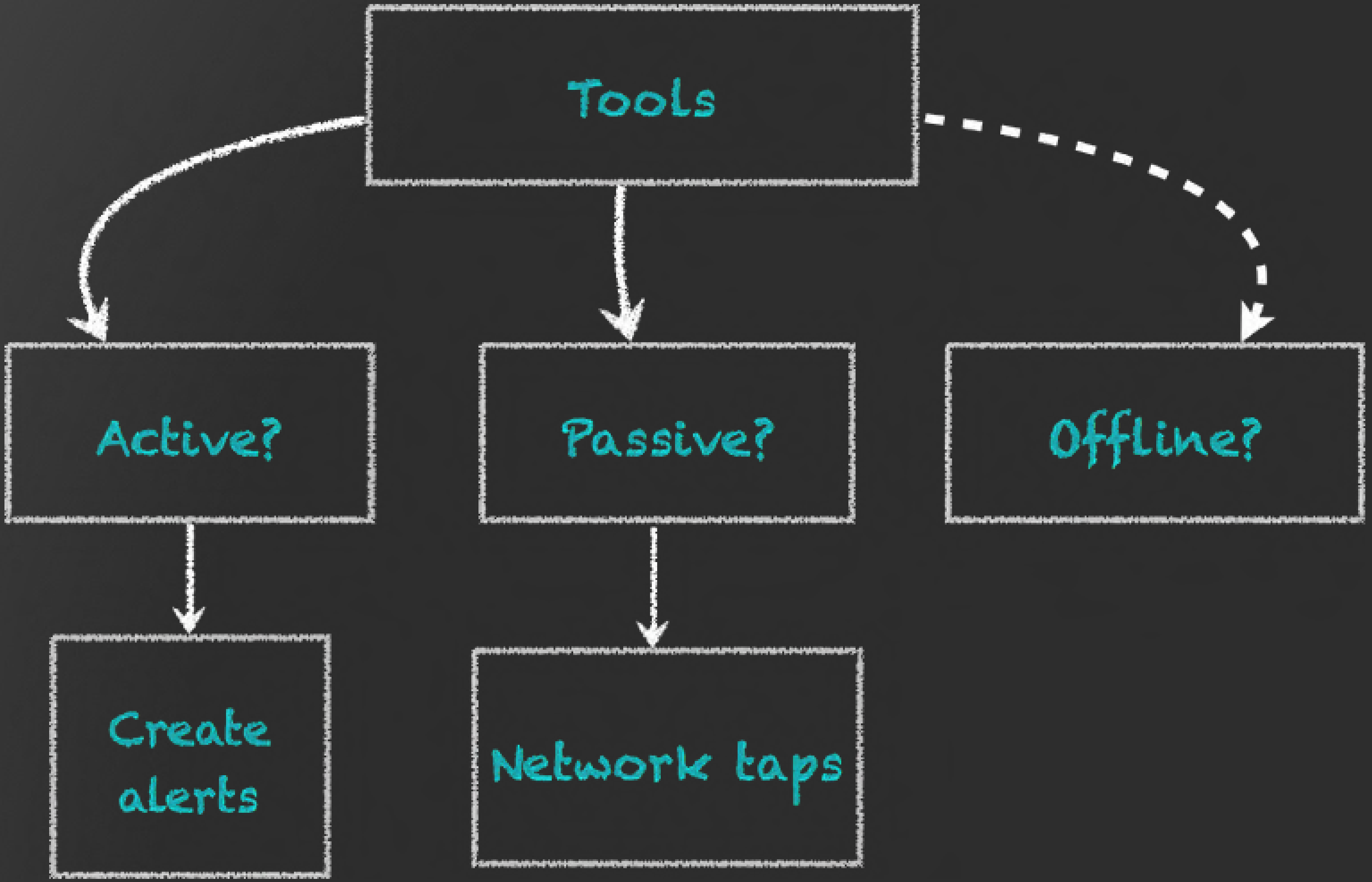
©2016 CodeKaram

Step 1: Monitor



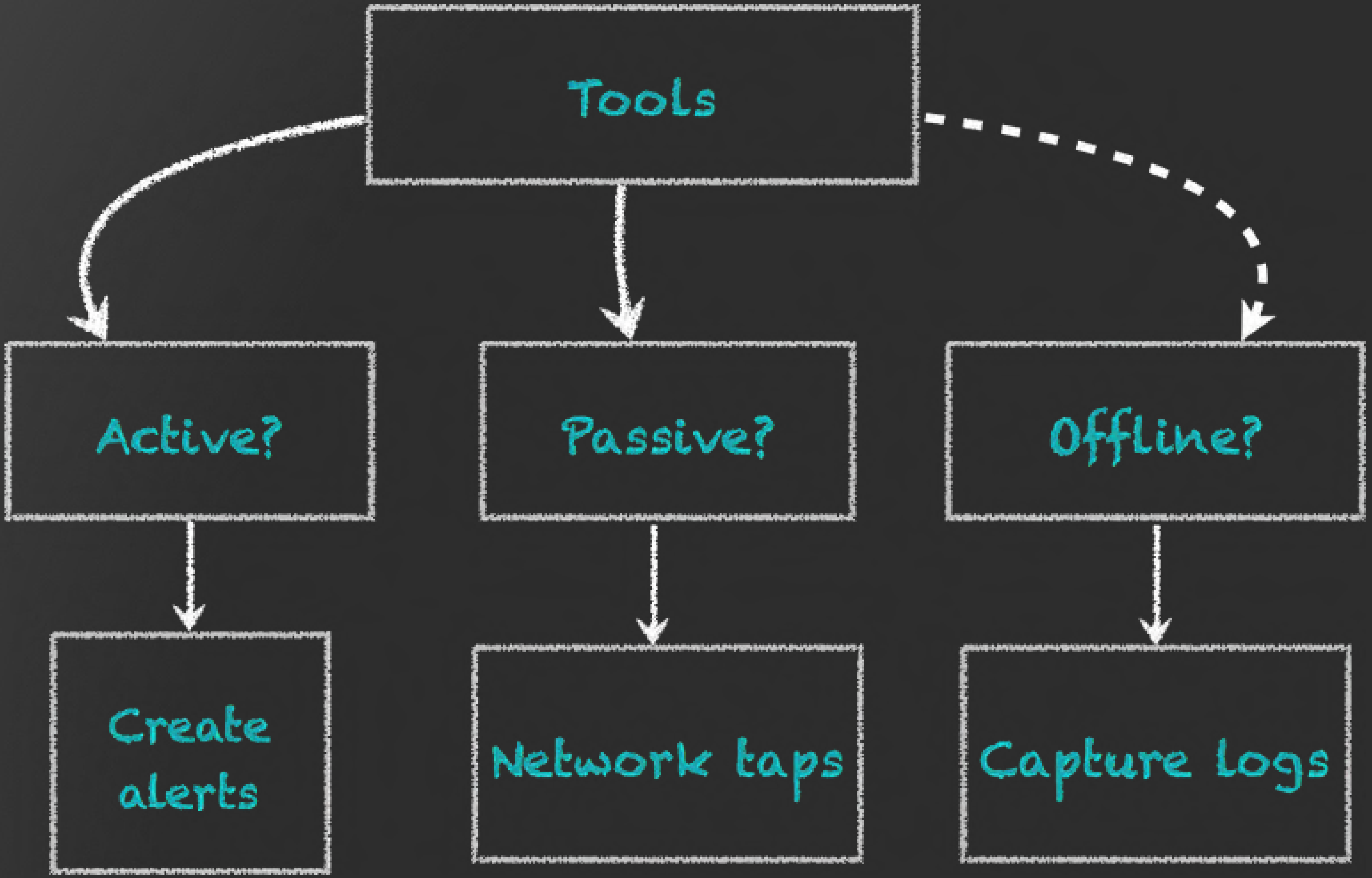
©2016 CodeKaram

Step 1: Monitor



©2016 CodeKaram

Step 1: Monitor



©2016 CodeKaram

Step 1: Monitor

- Tools?
  - VisualVM, Java Flight Recorder
  - PrintCompilation, PrintGCDetails (+PrintGCDateStamps), jmap –clstats, jcmd GC.class_stats

©2016 CodeKaram

Step 1: Monitor

- Tools?
  - Linux – mpstat, sysstat – iostat,pidstat…, prstat, vmstat, dash, CPU-Z, cacti …
  - Windows – Performance Monitor, Task Manager, Resource Monitor, CPU-Z, cacti …

©2016 CodeKaram

Step 2+3: Profile + Analyze

- You have all the data that you need!
- Identify areas of improvement
- Profile those potential hotspots
- Analyze those hotspots

©2016 CodeKaram

Step 2+3: Profile + Analyze

- Tools? (Free/Open source/GPL/BSD)
  - Oracle Solaris Studio Performance Analyzer, perf tools, PAPI, Code XL, Dtrace, Oprofile, gprof, LTT (linux trace toolkit)
  - Java Application – VisualVM, Netbeans profiler, jconsole …

©2016 CodeKaram

Step 4: Tune

- Tune the JVM/GC – select the right heap, the right GC algorithm
  - Age objects appropriately
  - Promote only long-lived objects
  - GC worker threads per VM (for stop-the-world GC events)
  - GC concurrent worker threads per VM

©2016 CodeKaram

Step 4: Tune

- Tune the JVM/GC – select the right heap, the right GC algorithm
  - See if compressed oops can be helpful
  - Larger heaps may need AlwaysPretouch to be enabled and also UseLargePages of appropriate size

©2016 CodeKaram

Step 4: Tune

- Tune your code to meet or exceed your SLAs
  - Appropriate ramp-ups and ramp-downs
  - Object age and retention strategies
    - Understand what forms your LDS (live data set)
  - Confirm you are measuring the right thing! ☺

©2016 CodeKaram

Bottom-Up Approach

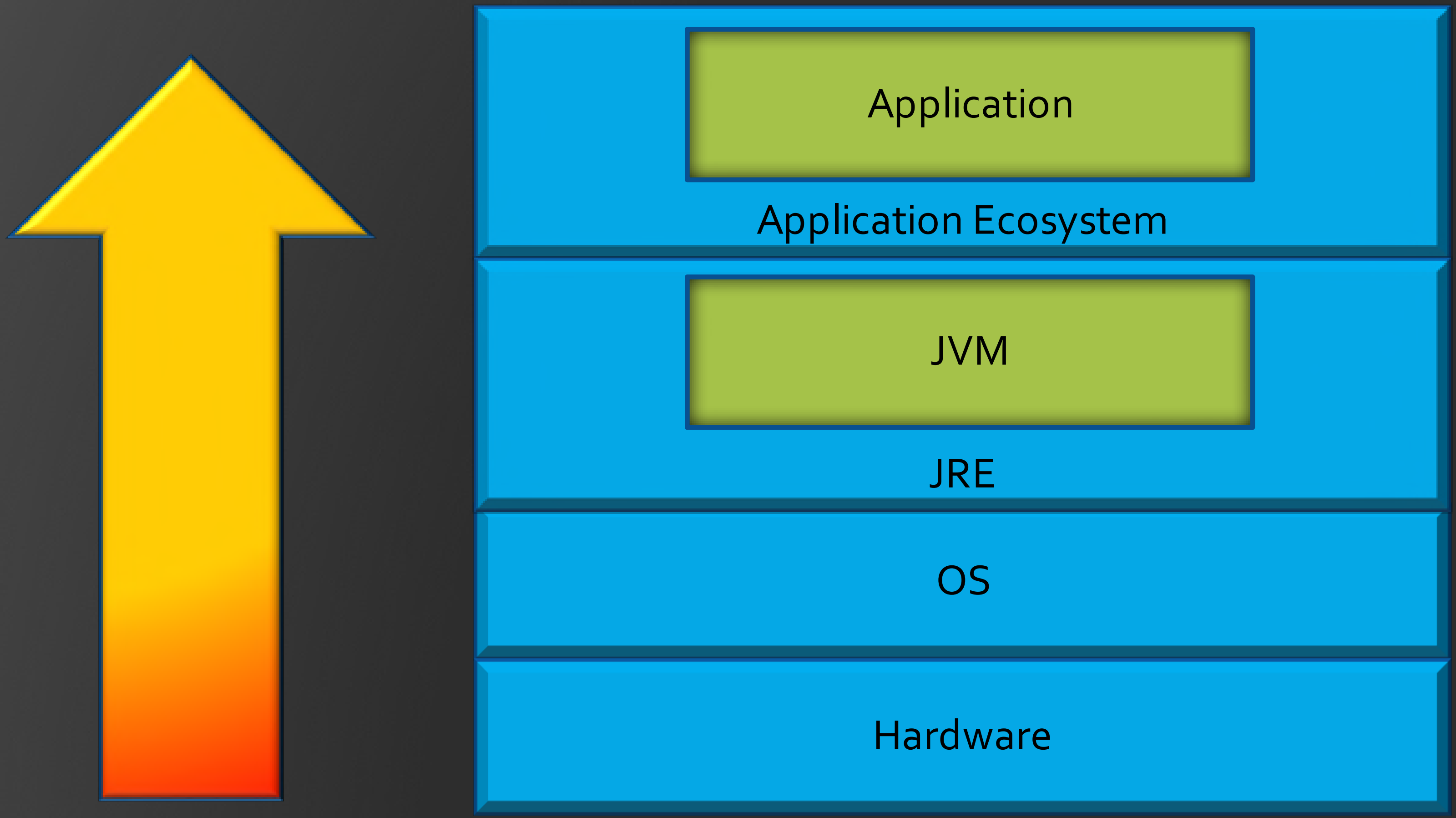
Bottom Up Approach

I NEED the power!!

... to stress the platform

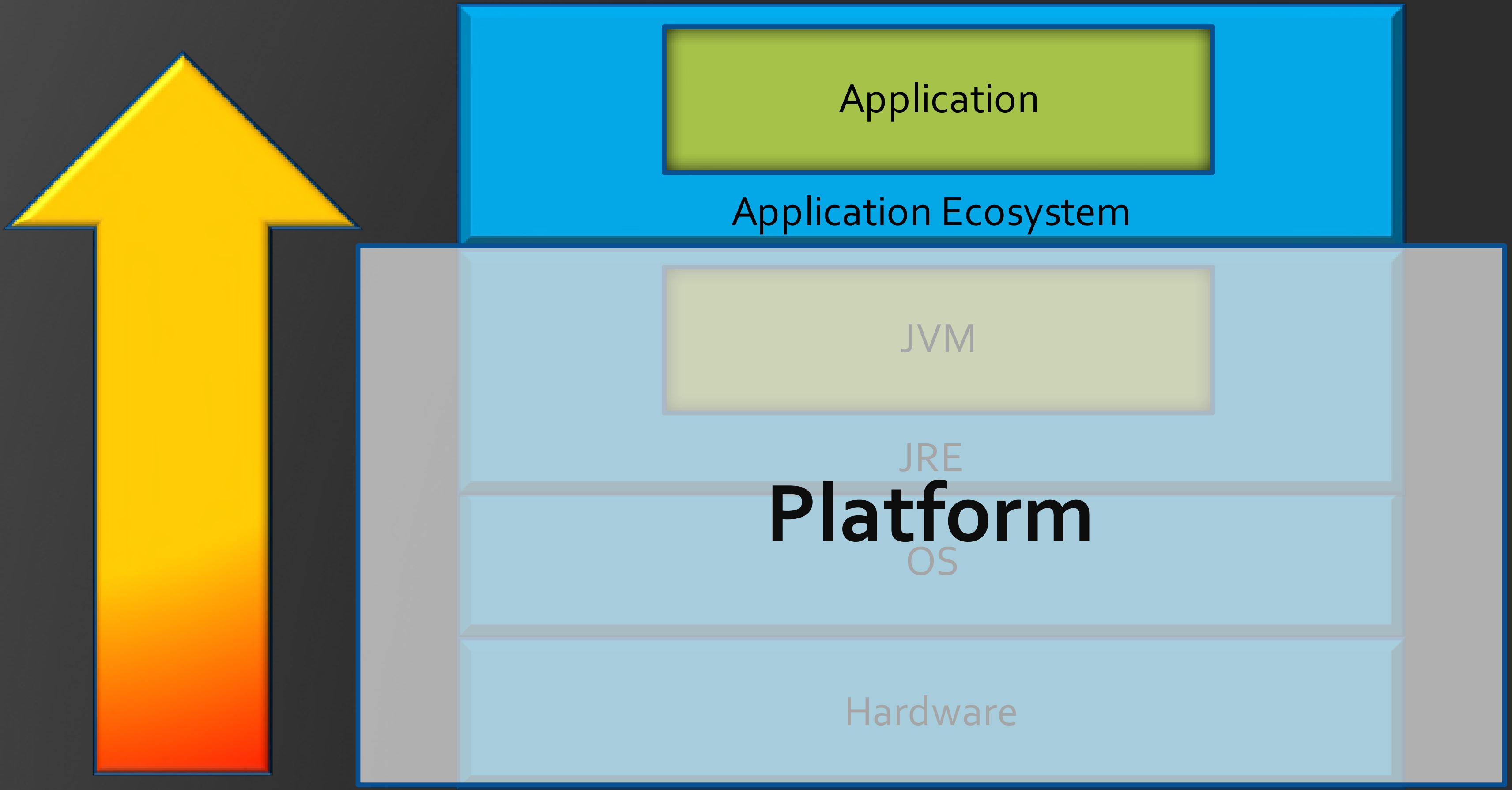
©2016 CodeKaram

Top-Down Approach



©2016 CodeKaram

Top-Down Approach



©2016 CodeKaram

Where to Start?

- Know what you are stressing
- Get/ write the appropriate workload/ application
- Get/write the appropriate tools

©2016 CodeKaram

Know What You Are Stressing!

CPU –
Gather performance counter information for your CPU stats, core stats, cache hits, misses and levels, branch predictions, pipeline information, order-of-execution, load-store unit load and queues, etc

©2016 CodeKaram

Know What You Are Stressing!

Memory –
Gather performance counter information for memory utilization, memory bandwidth, read-write stats, max read bandwidth, max write bandwidth, max cross traffic bandwidth, architectural related considerations, max capacity, etc

©2016 CodeKaram

Know What You Are Stressing!

JVM / GC–
Gather information related to the change – e.g. new GC!
Gather information on different GC phases, parallel work queues and work performance, concurrent work and pressure, internal queues and buffers, any GC work that’s staged?, etc

©2016 CodeKaram

Know What You Are Stressing!

JVM / GC–
Gather information related to the change – e.g. new GC!
Gather information on different GC phases, parallel work queues and work performance, concurrent work and pressure, internal queues and buffers, any GC work that's staged?, etc

©2016 CodeKaram

Where to Next?

- Know what you are stressing ✓
- Get/ write the appropriate workload/ application ✓
- Get/write the appropriate tools ✓

©2016 CodeKaram

Let's have some fun!

Further Reading

TURING 图灵程序设计丛书

PEARSON

Java性能优化圣经！ Java之父重磅推荐！

Java性能优化权威指南

Java Performance

[美] Charlie Hunt
Binu John ■ 著
柳飞 陆明刚 ■ 译

人民邮电出版社
POSTS & TELECOM PRESS

Java性能优化权威指南 [美] Charlie Hunt Binu John 著 人民邮电出版社

www.codekaram.com

www.slideshare.net/monicabeckwith