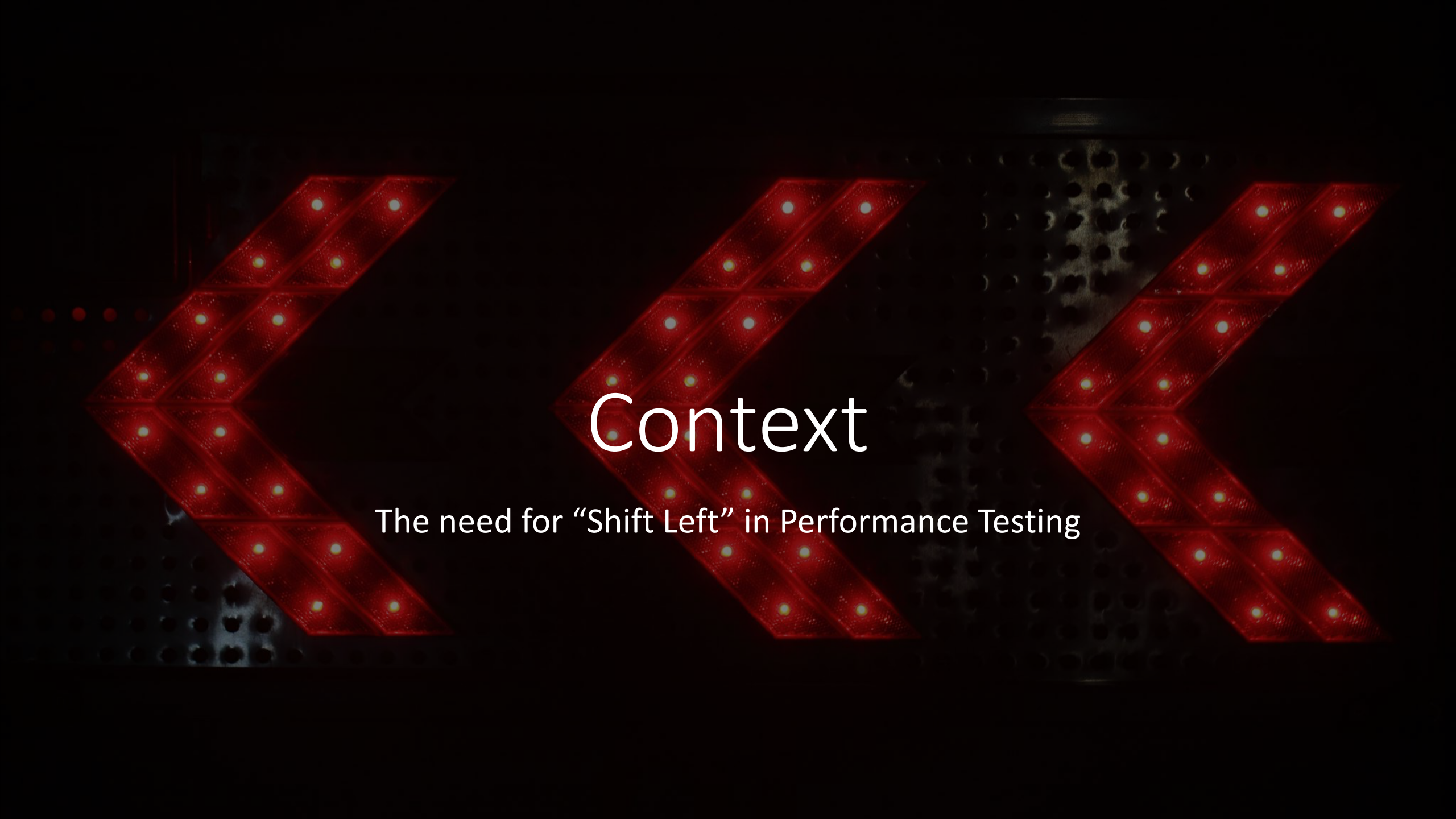
Shift Left your Performance Testing

HARI KRISHNAN

@harikrishnan83



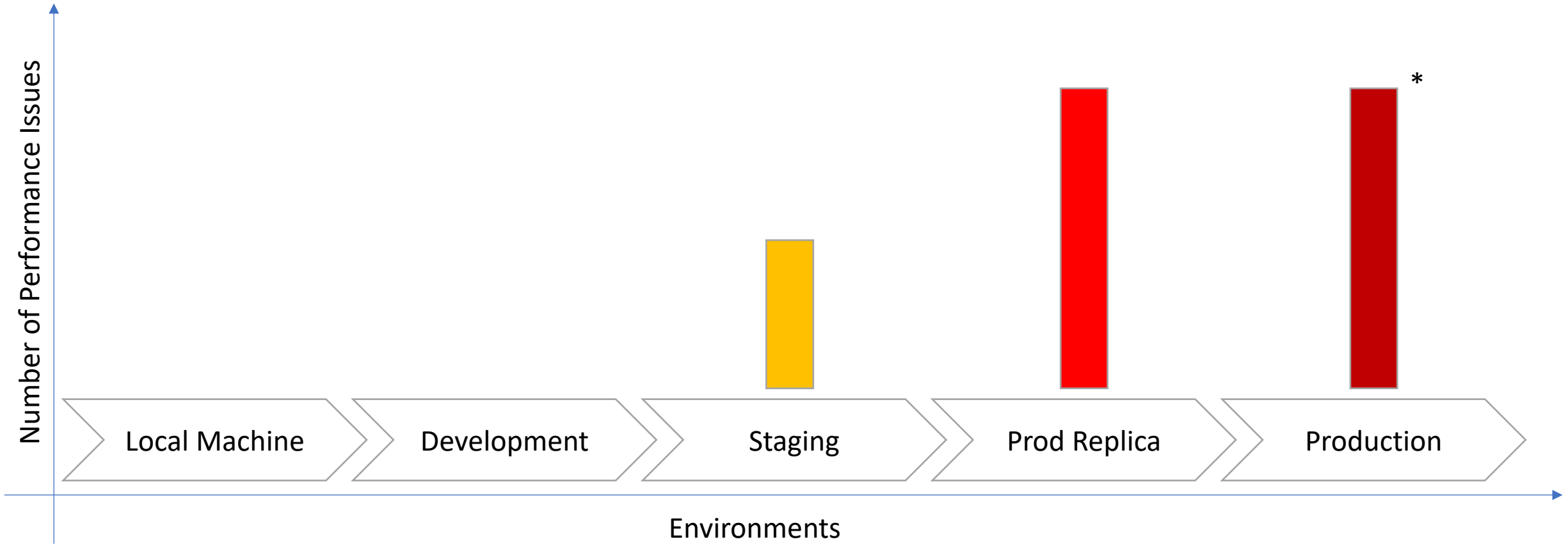
Context

The need for “Shift Left” in Performance Testing

Show of hands

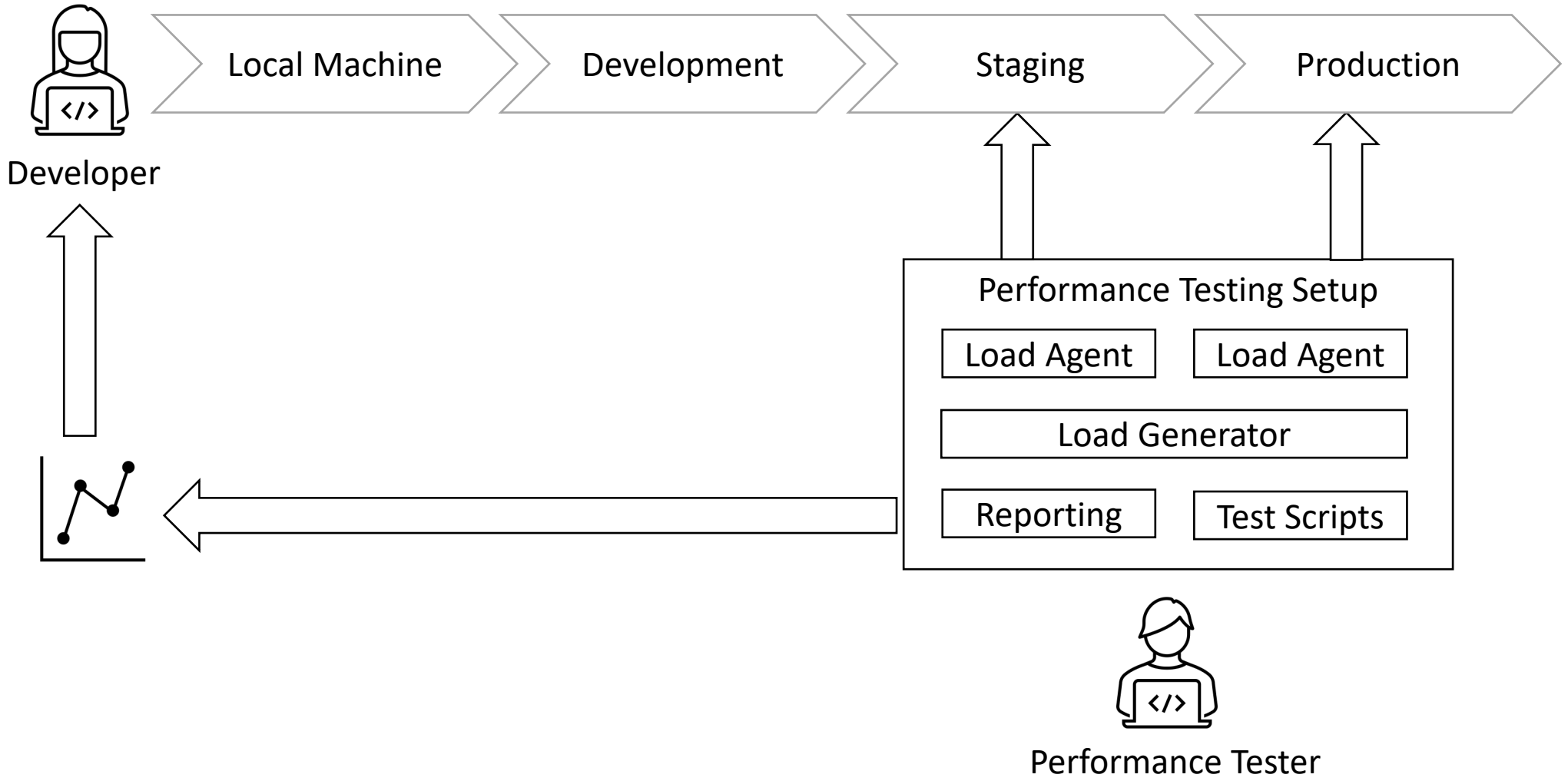
In which Environment do you identify Performance Issues?

In which Environment do youidentify Performance Issues?

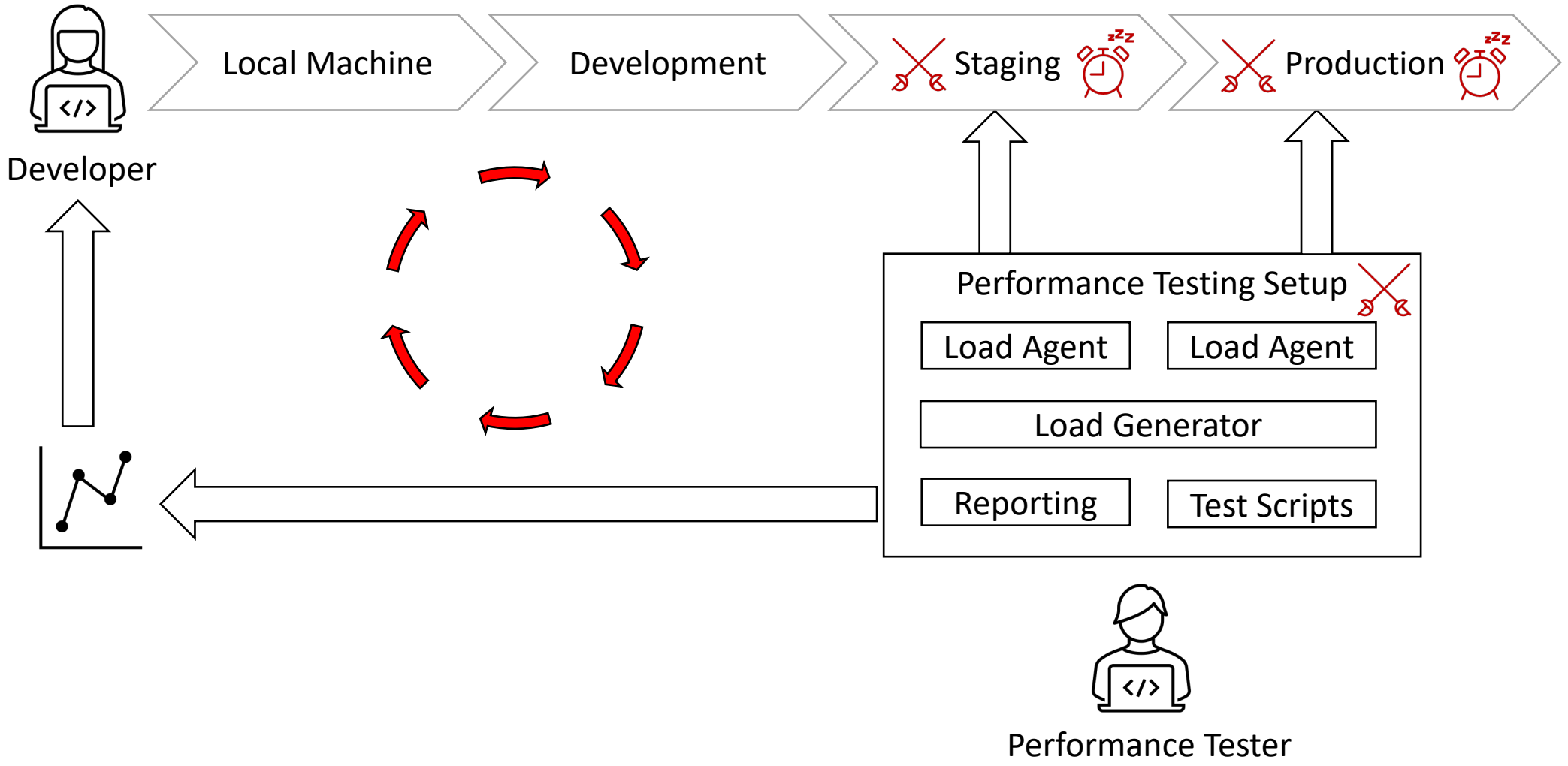


* **Legend:** Color represents Mean Time To Resolution, Darker the shade of Red, longer it takes it fix

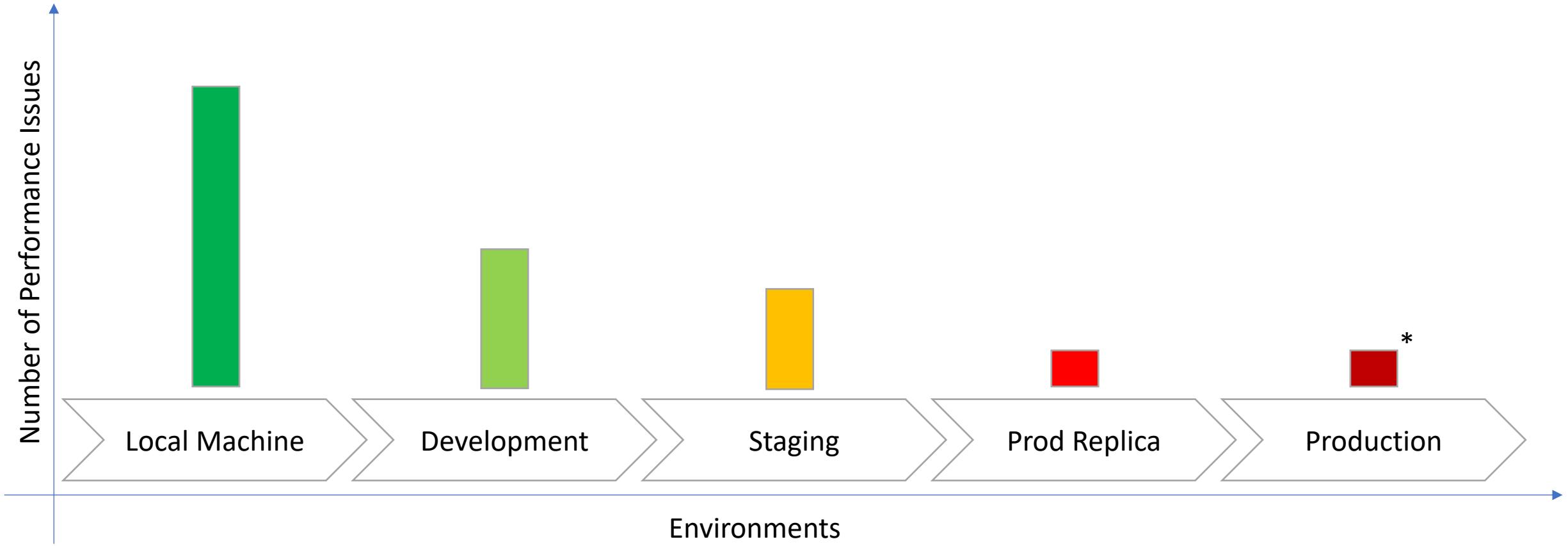
Perf Testing – The Usual Setup



Perf Testing – Usual Setup Issues

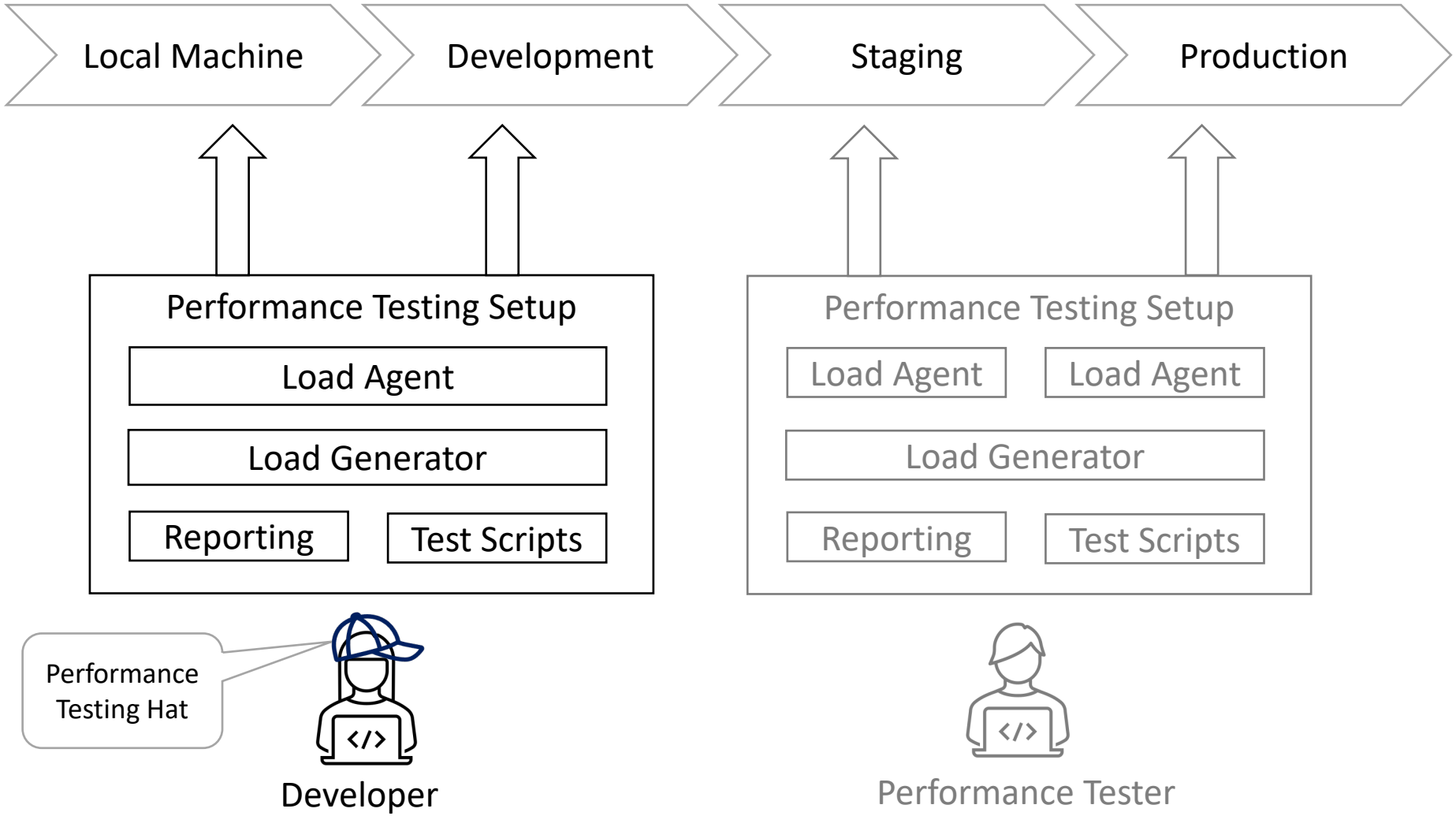


What it should look like instead



* **Legend:** Color represents Mean Time To Resolution. Green represents quickest resolution. Darker the shade of Red, longer it takes it fix.

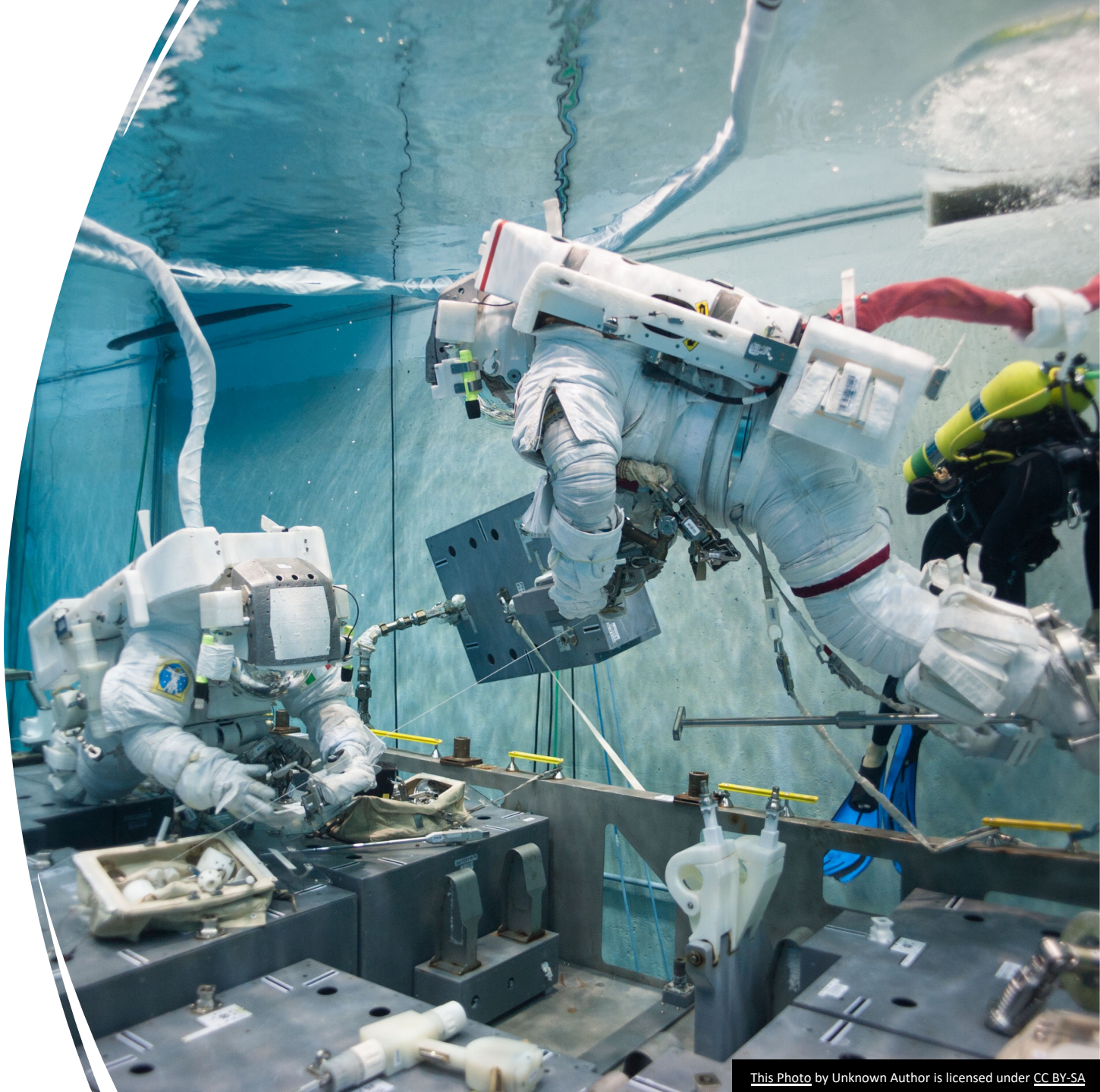
Running Perf Test Early in your Development Cycle



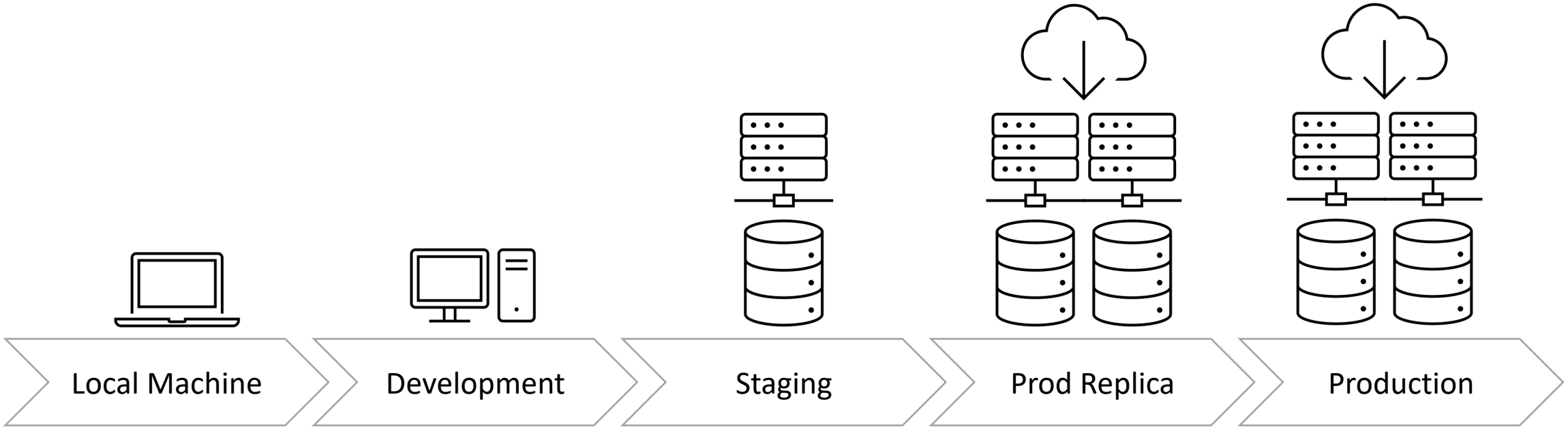

“Shift Left” Challenges

The Scale Challenge

Creating truly representative performance test setups on local machines



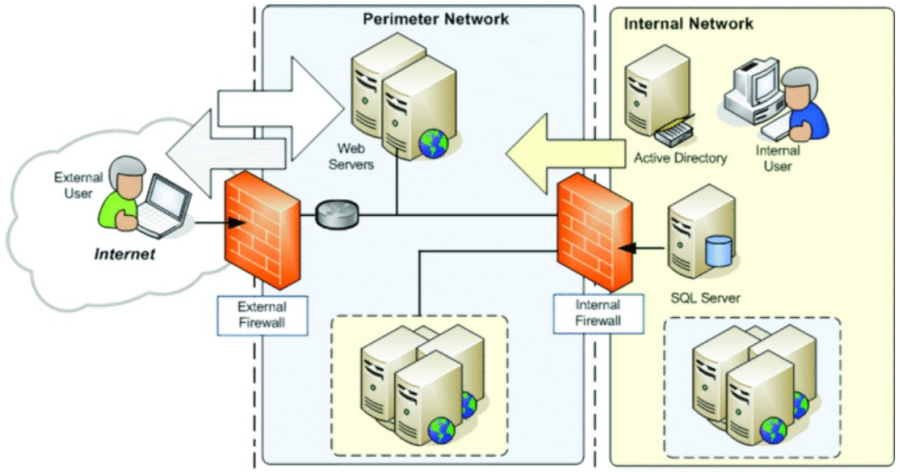
Local Machines are not representative of **Production Architecture**

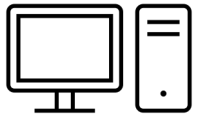


Local Machines are not representative of **Network Topology**

Lower environments do not have production like sophisticated network topology which involves

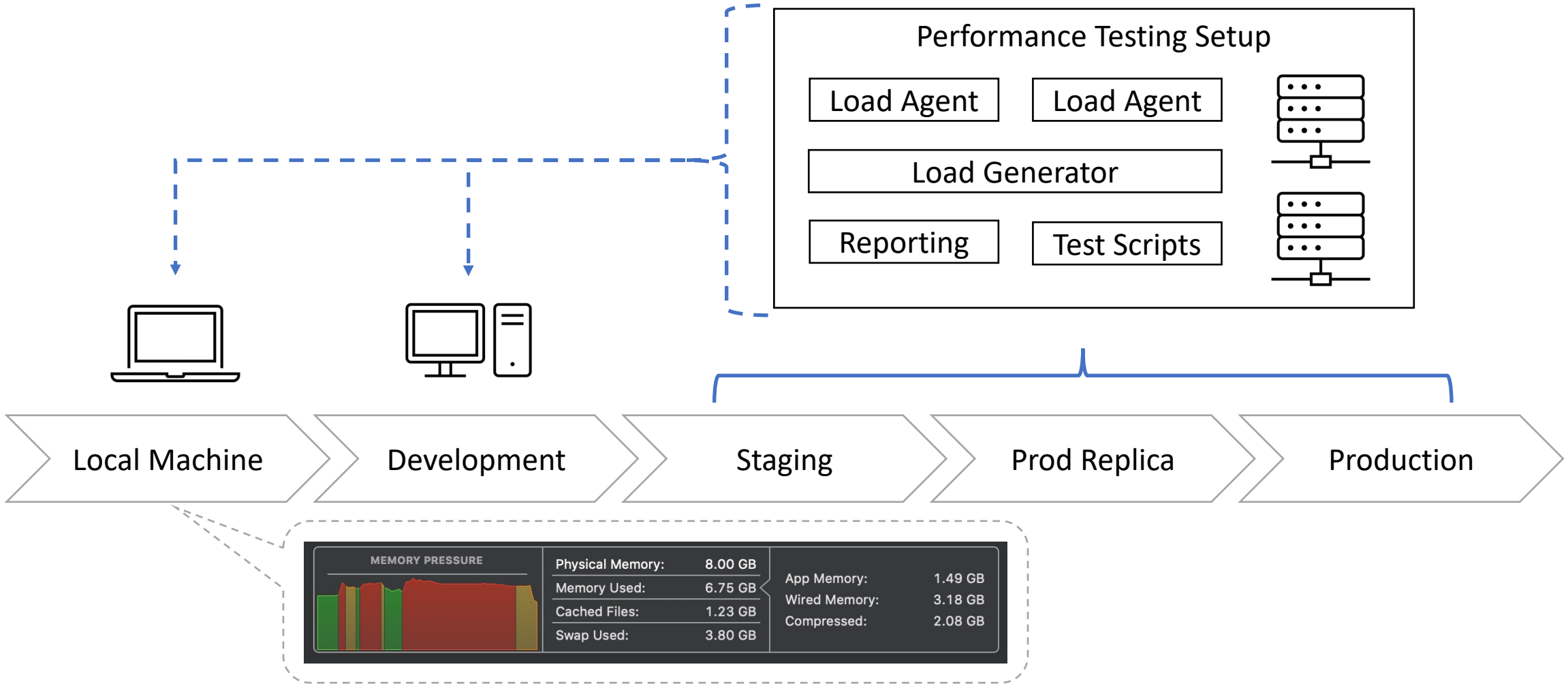
- Firewalls
- Proxies
- Latency levels etc.





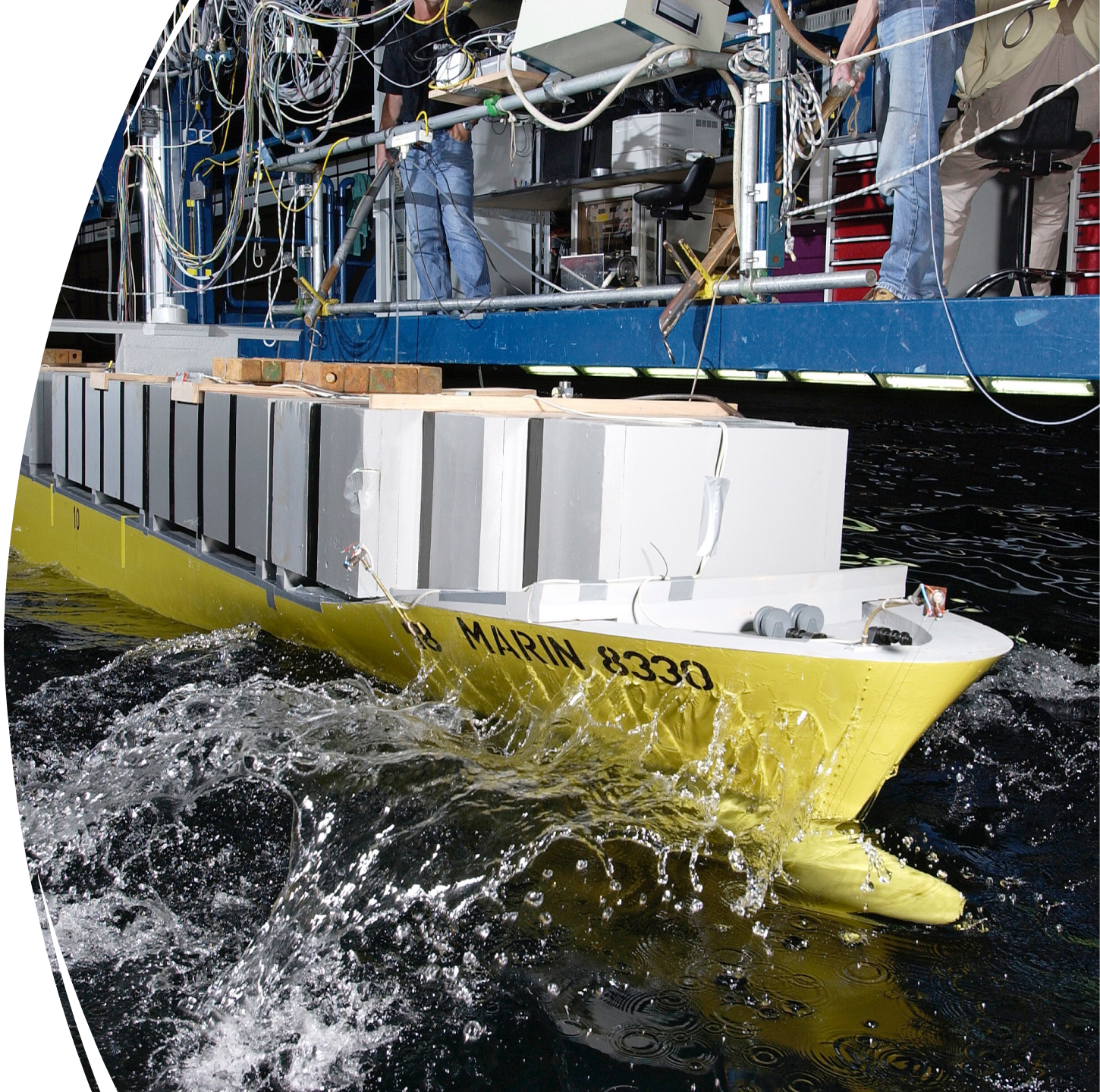
Local Machine → Development → Staging → Prod Replica → Production

Developer Machines are not capable of **Generating Adequate Load**

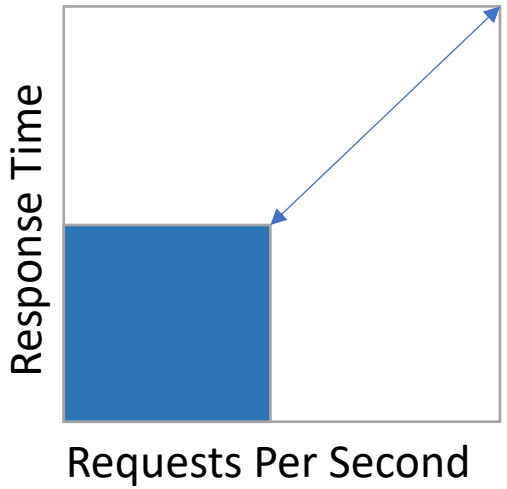


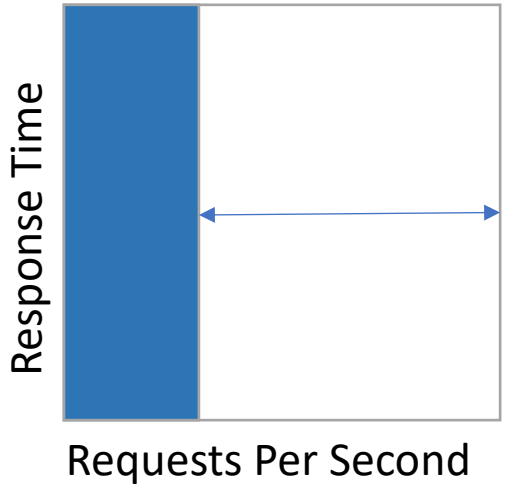
Shift Left = **Scale Down**

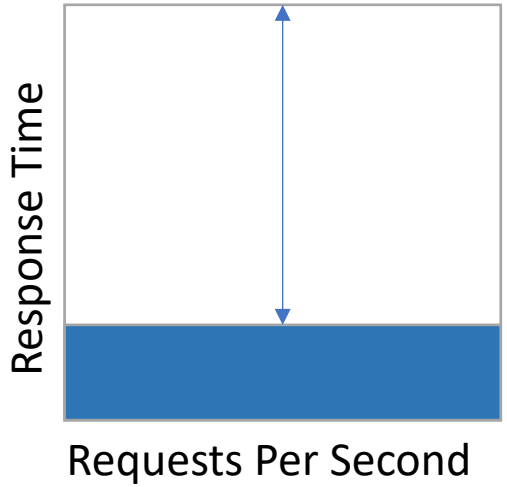
It is not possible to conduct sea trials on a full-scale ship with every design change.



Basis to Scale Down







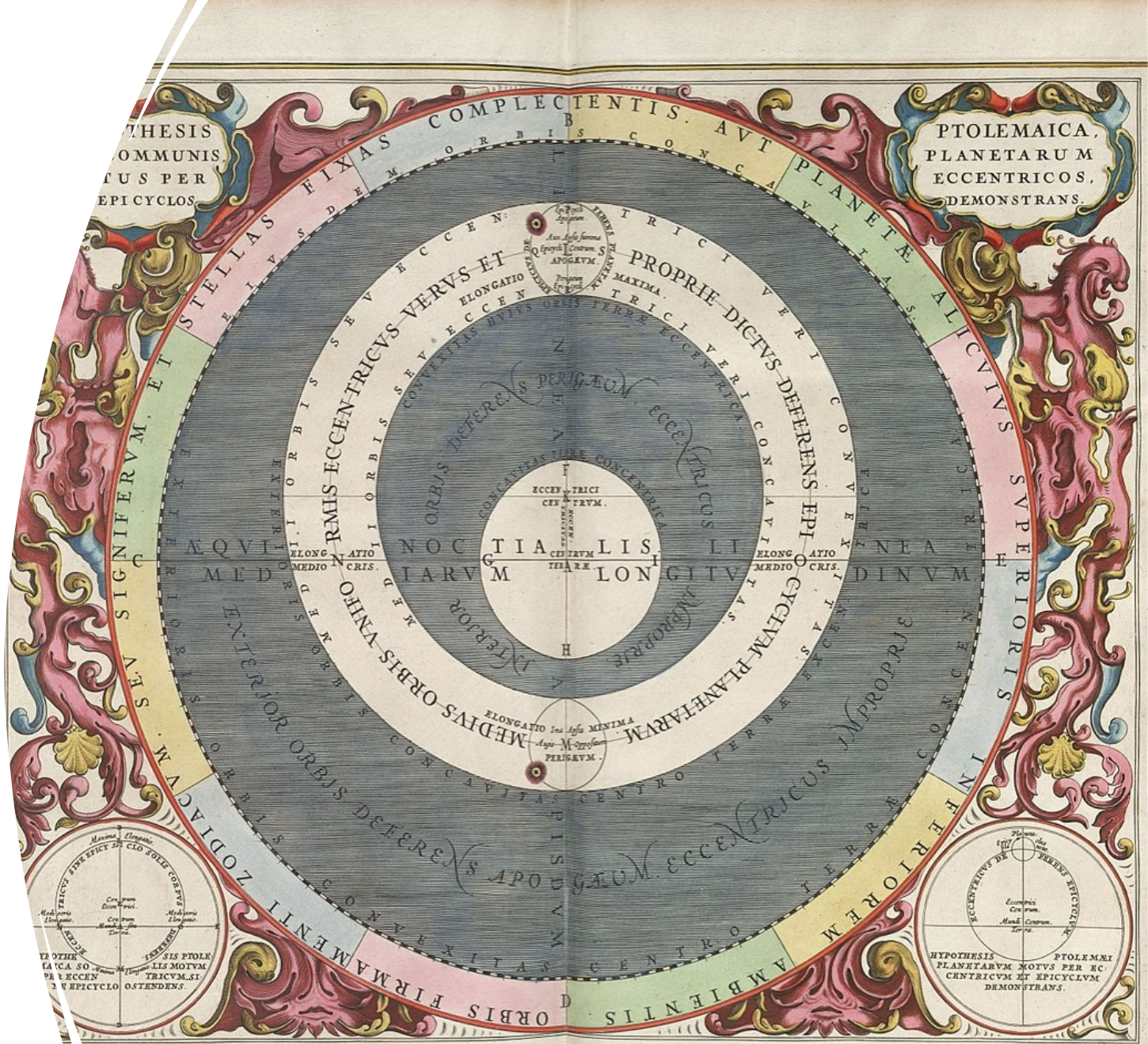
How do we scale-down load **accurately** to **validate** performance KPIs?

We cannot.

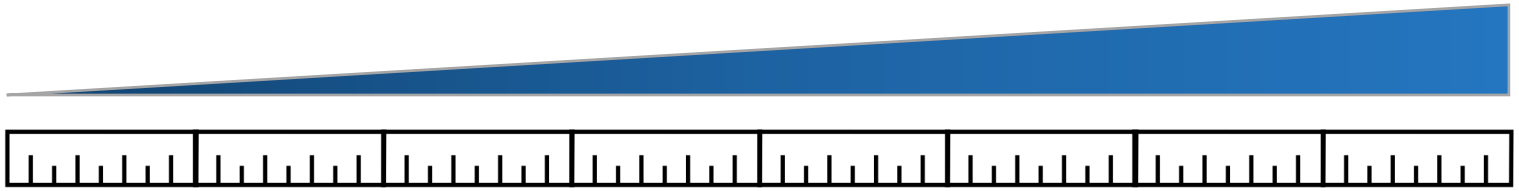
However, we can **scale down the trend** and
invalidate hypotheses

Hypothesis Invalidation

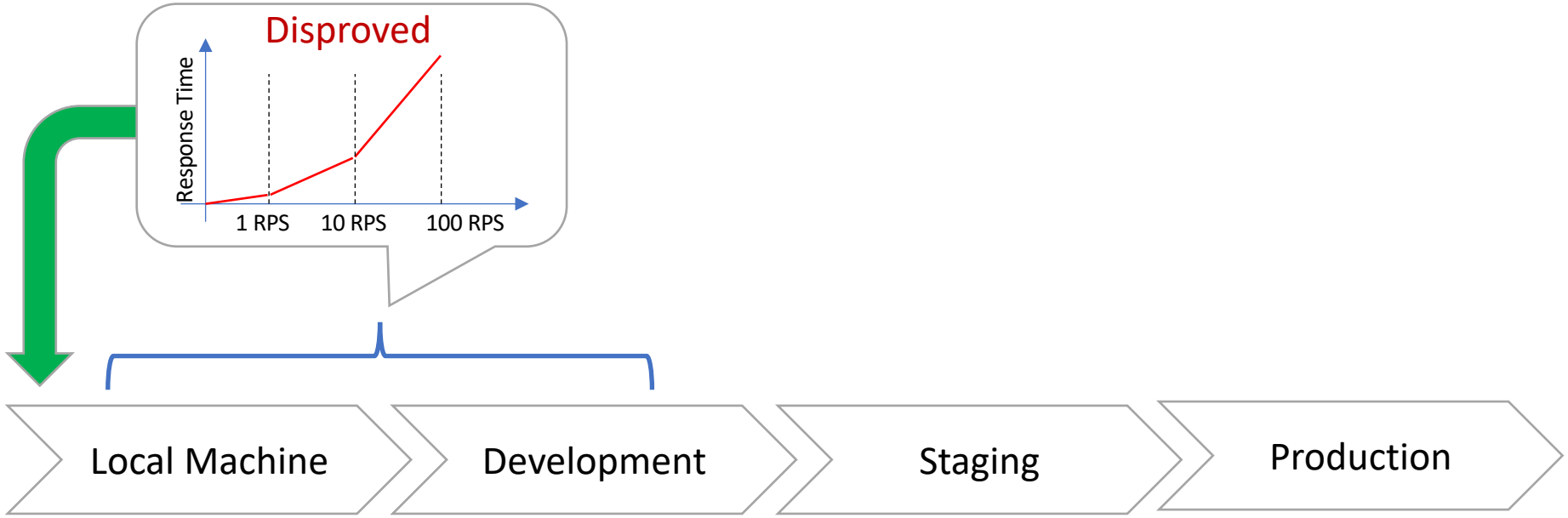
*a **hypothesis** is used to formulate provisional ideas… the hypothesis is proven to be **"true"** or **"false"** through a **verifiability**- or **falsifiability**- oriented experiment* – Wikipedia



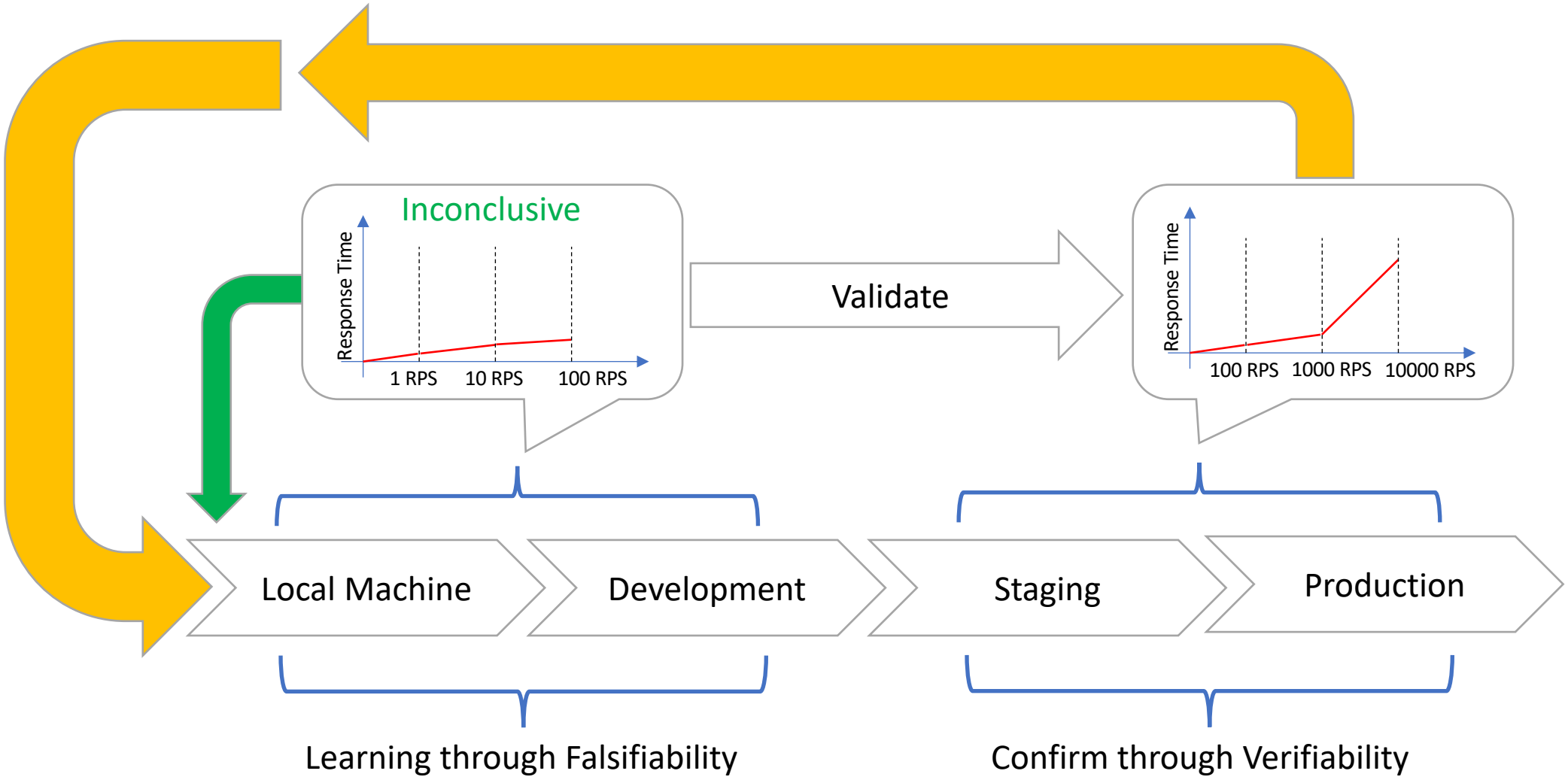
Invalidating the Scaled Down Problem



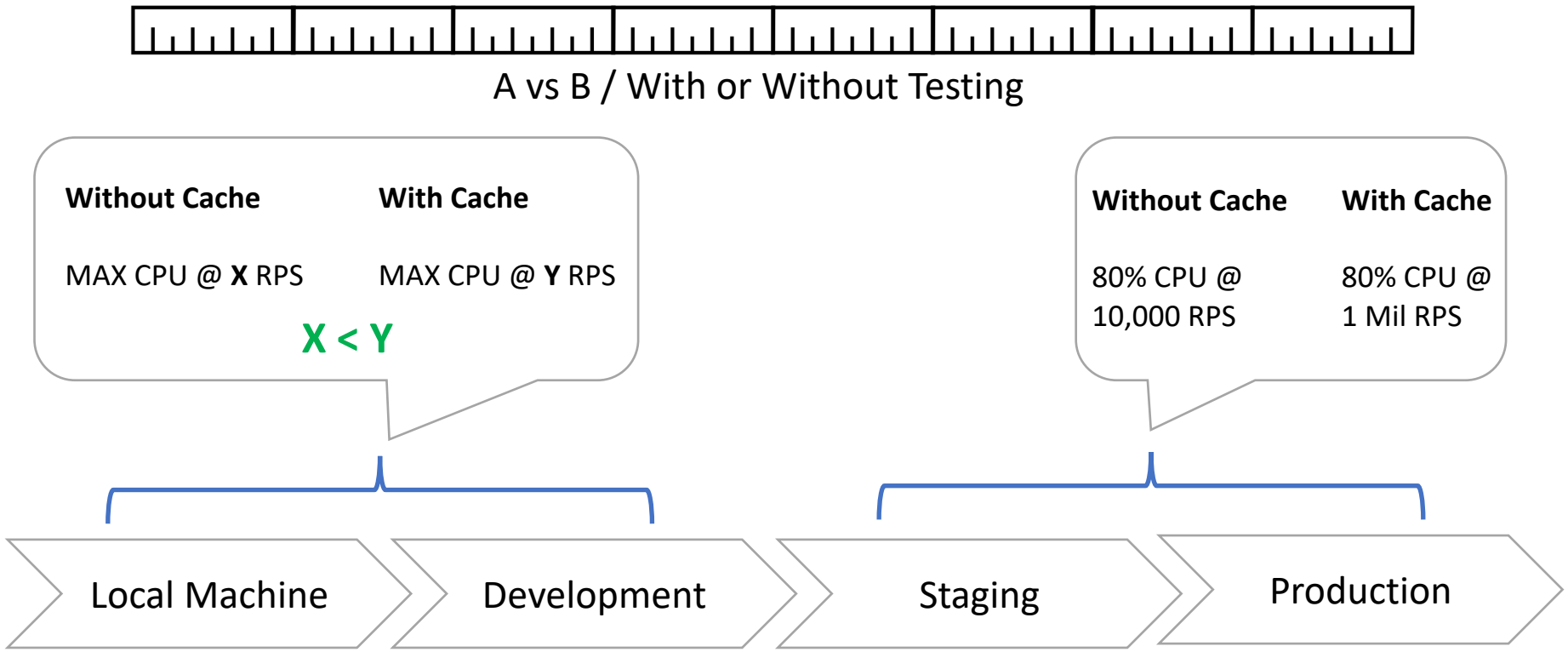
RPS and Response Time Trend



Invalidating the Scaled Down Problem



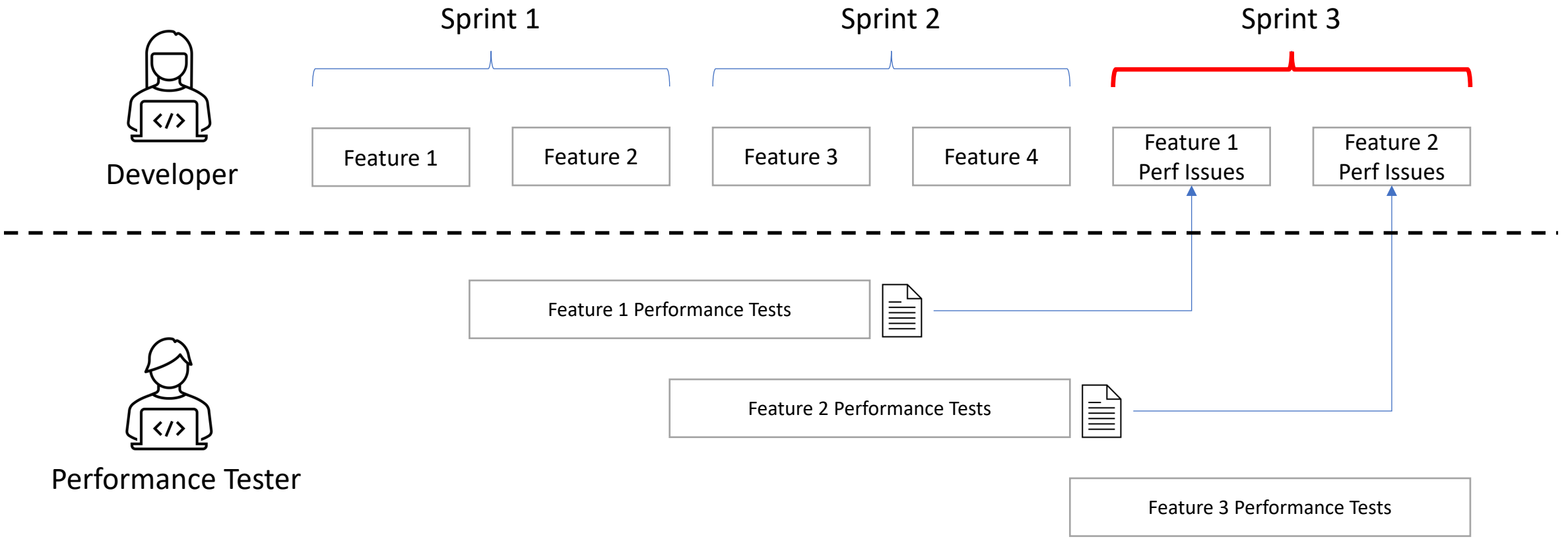
Invalidating the Scaled Down Problem



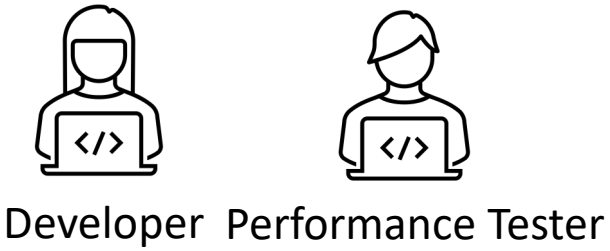
The Capacity Challenge

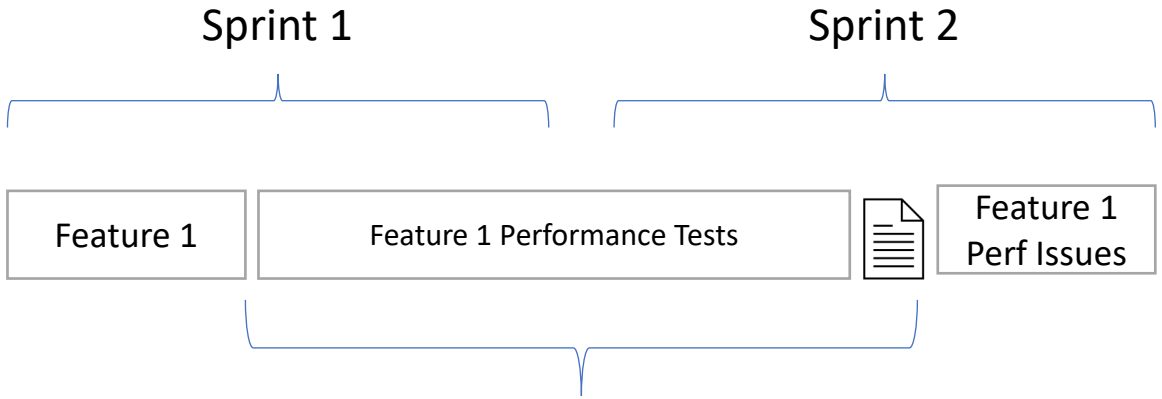
The problem with running Performance Tests within your Sprint Cycle

Perf Tests and Sprint Cadence



Perf Tests and Sprint Cadence

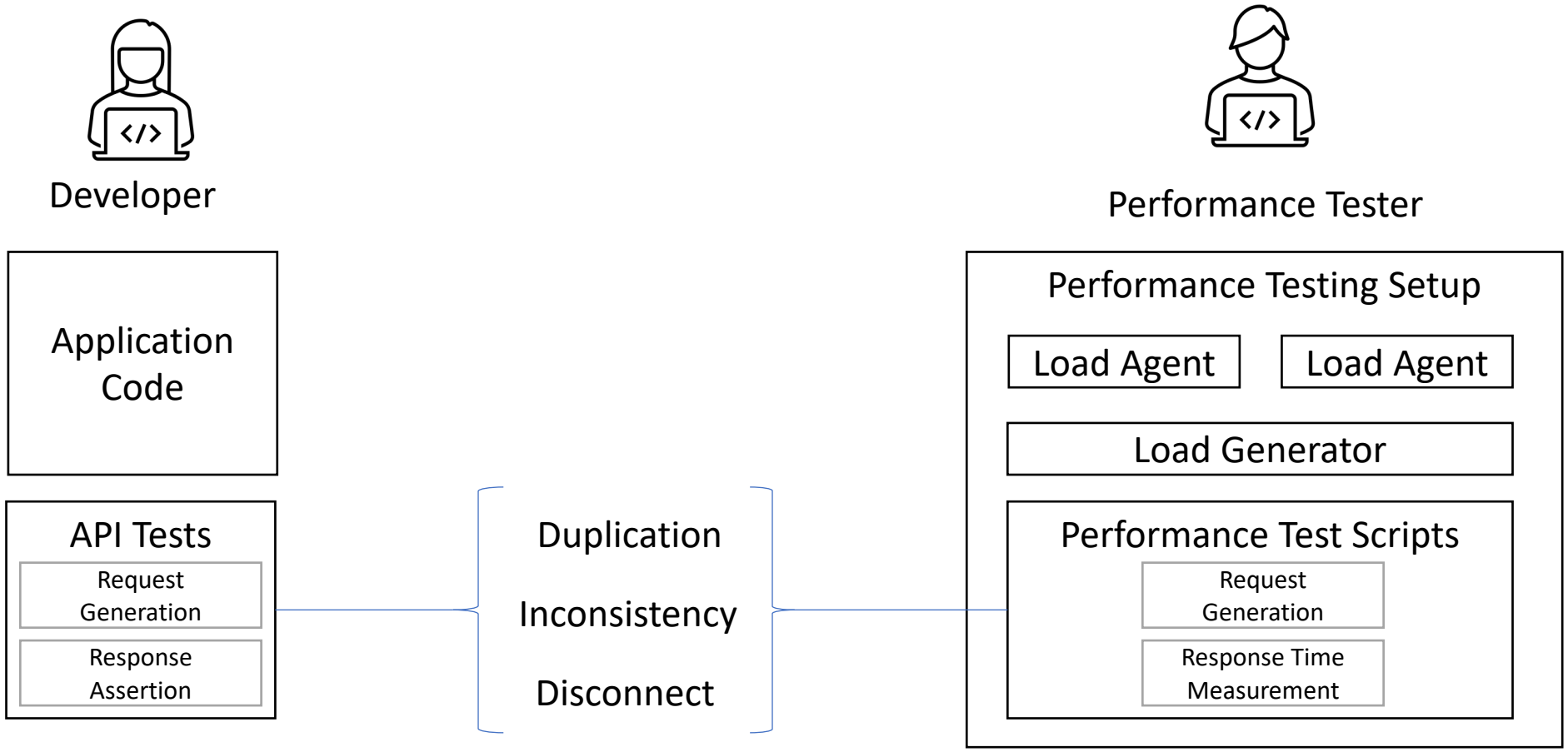




Reduce this Feedback Cycle

1. **Reduce** Effort
2. **Reduce** Complexity
3. **Reduce** Repetition
4. **Automate**
5. **Automate**
6. **Automate**…

Reduce Repetition: Performance Test Scripts

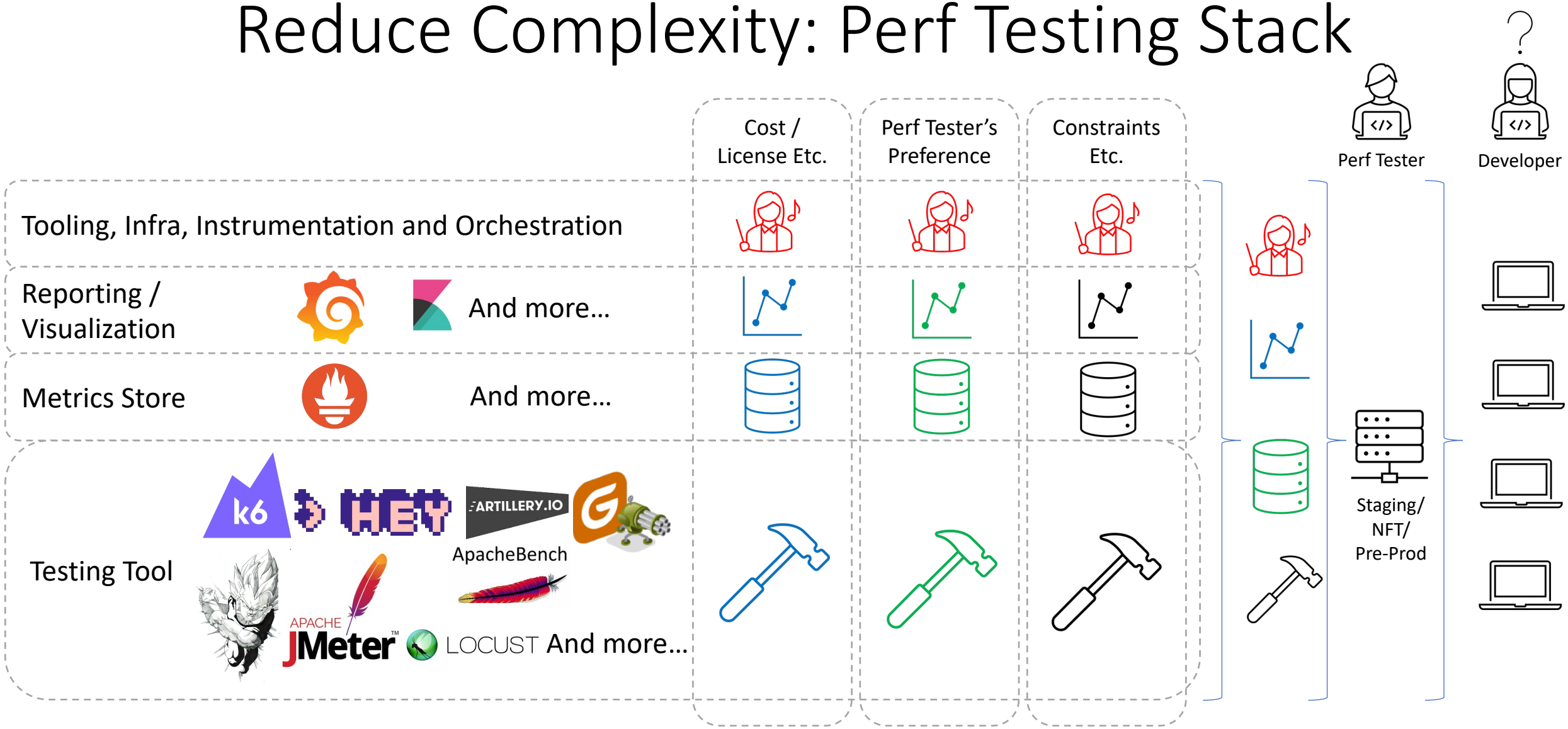


Shift Left = Repurpose

Leverage API Tests / Specs as Perf Test Scripts

- **Reduced Maintenance** – No need to update two separate scripts
- **Consistency** – While Perf tests may not require result assertion, they will still be consistent with API specs
- **Co-ordinated Effort** – Performance Testers and Developers work more closely and can help each other

Reduce Complexity: Perf Testing Stack



Shift Left = Containerize

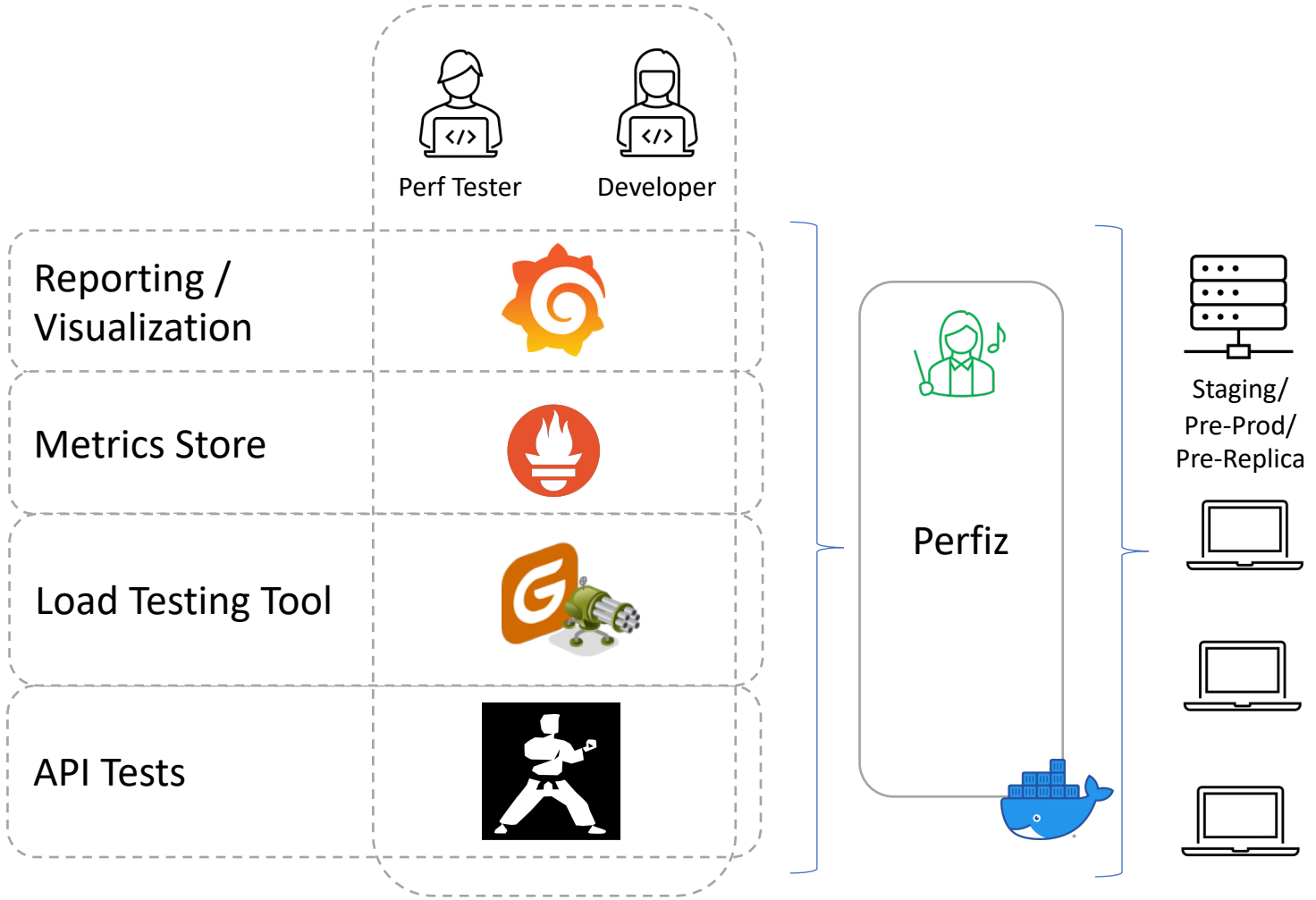
- Containerize the performance test setup so that there are no manual steps to bring it up in a local machine or in a higher env
- Develop your Perf Test Setup just like code on Dev Machine and promote it to higher environments

Live Demo

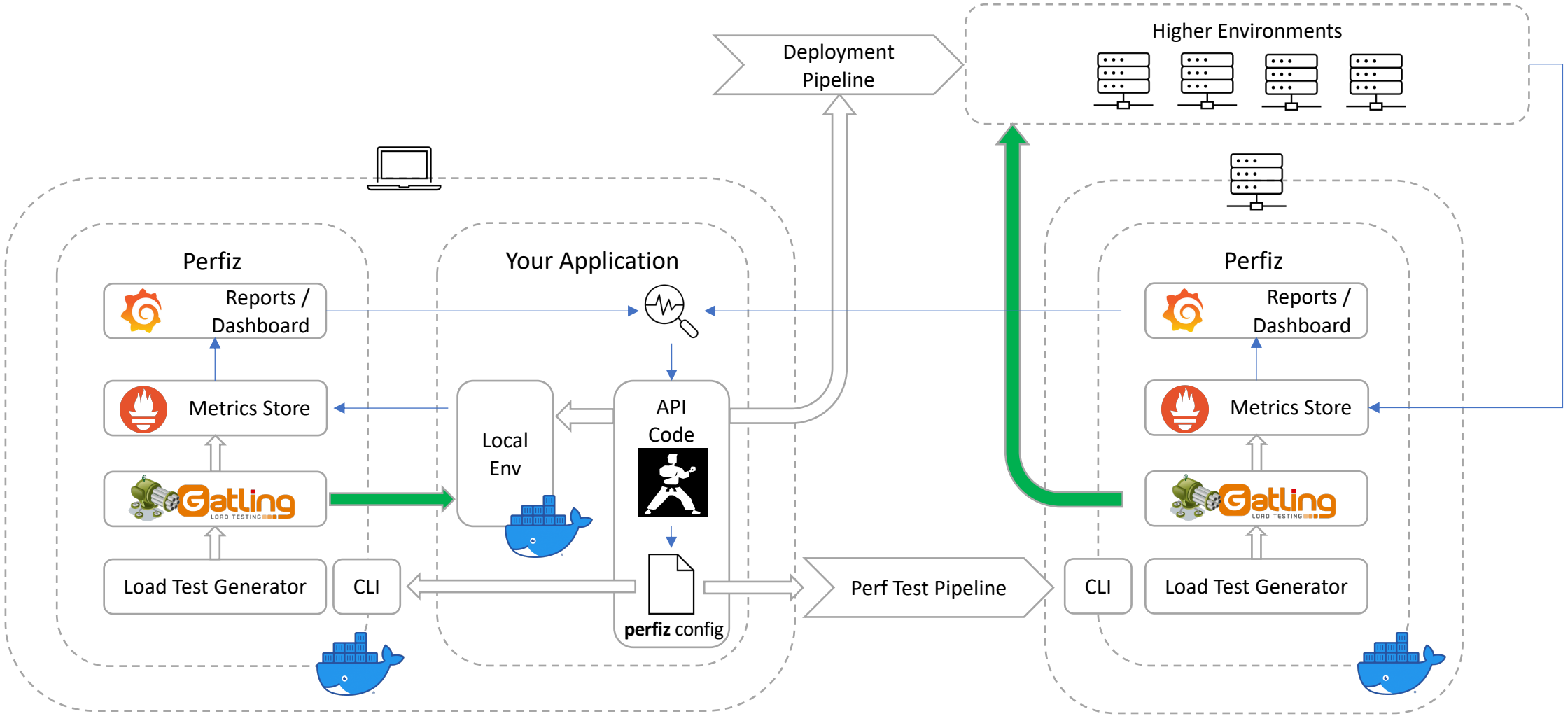
Perfiz - An Open-Source Containerized Perf Test Setup

https://github.com/znsio/perfiz

The Perfiz Stack



Perfiz – How it works?

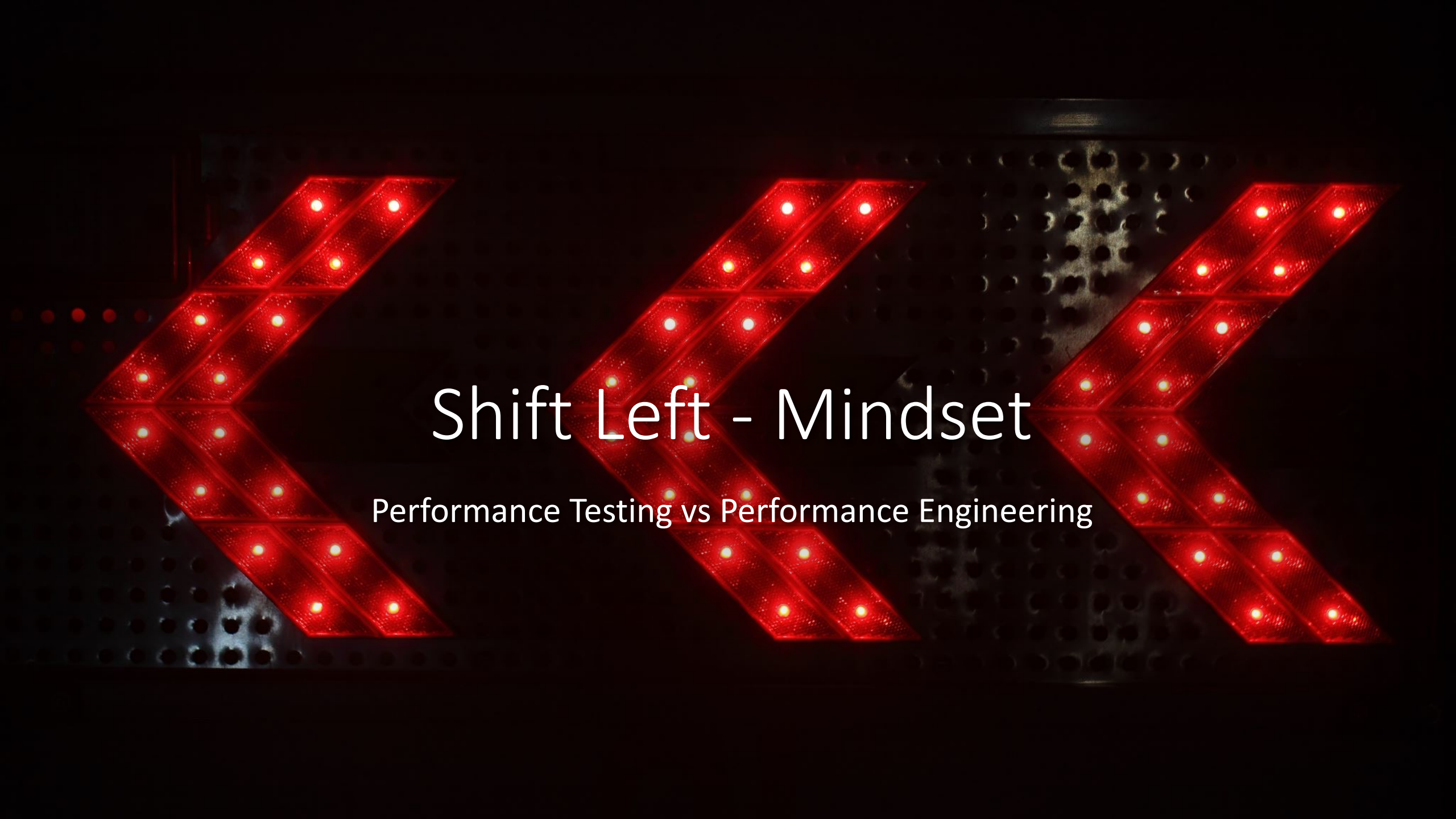


Perfiz – A Codified Approach

- Embody “**Shift Left**” strategy by enabling Developers to run Performance Tests on their local machines
- **Containerize** the setup so that it is equally easy to run on local machine, lower environments and perf test environments
- **Live Dashboard** that can visualize Load Test metrics and Application metrics together to help correlate load with Application behavior
- **Re-usable** setup that is agnostic to application tech stack
- Requires **no / low code** (config only) to leverage API tests or Specs as Perf Test Suite
- **Pre-configured** setup with Canned Dashboards, Data Source setup, monitoring hooks etc. to help Developers get started quickly

Perfiz - What it is not?

- Not a replacement for Perf Test Tools. It depends on them.
- Does not replace perf testing skills
  - It just makes your life easy in getting your test runs.
  - You still need to understand load profiles and best practices for running perf tests (Topic for another talk?)
  - You still need to design the tests and what you want to learn from it.
- While it is opinionated about what makes a good test setup
  - It does not push any specific tools.
  - Examples:
    - We started InfluxDB and then moved to Prometheus, we can support other time series DBs also.
    - We are using Gatling now; however, we may support other tools as we see fit



Shift Left - Mindset

Performance Testing vs Performance Engineering

Shift Left = Performance Testing as a Learning Exercise

- **“Testing”** – This word may be leading us to believe Perf Testing is a Verification Activity.
- Instead, a large part of Perf Testing should be a series of **Spikes** which help us learn and think scientifically about our solutions.
- Avoid **“Guesswork”** in Architecting Applications by Validating your Hypothesis (OR invalidating them early)

Scientific Method

Avoiding Guesswork

Continuous Evolution Template

Problem Statement	Baseline KPI	Target KPI	Hypothesis	Experiments		Validated Learning
				Falsifiability	**Verifiability**	
Improve Max Throughput	10 K RPS @ 80% CPU	100 K RPS @ 80% CPU	Adding an In-Memory Cache will reduce Repeat Computations	On **Local Machine** there must be a change CPU usage after adding Cache	NA	Miss Rate is 100 %
			TTL on Cache Keys is too small and thereby keys are expiring before they are accessed a second time	Same as Above	On **Staging**, should be able to achieve 10x improvement in RPS	Staging achieved only 4x Improvement Miss Rate is 40% Eviction Rate is High
			Eviction Rate is High because we are running out of space. Increase Cache Size to X.	On **Local Machine**, decreasing Cache Size should cause Eviction Rate (To establish cause)	Same as Above	Application is stable at 80% CPU and X GB Memory
			Based on Previous Learnings, below configuration should achieve Target KPI • TTL: X • Cache Size: Y	On **Prod-Replica**, under baseline load • CPU Should Drop • Cache Hit Rate 100%	On **Prod-Replica**, should able to achieve Target KPI	

Left

Right

Thank you!

Twitter: @harikrishnan83

LinkedIn: https://www.linkedin.com/in/harikrishnan83/

Medium: https://medium.com/@harikrishnan

Q & A