Building Pipeline Optimizations

George Smith
SOLUTION ARCHITECT & EDUCATOR
@GeorgeS11323298

Intro

Optimization topics:

Job Parallelization

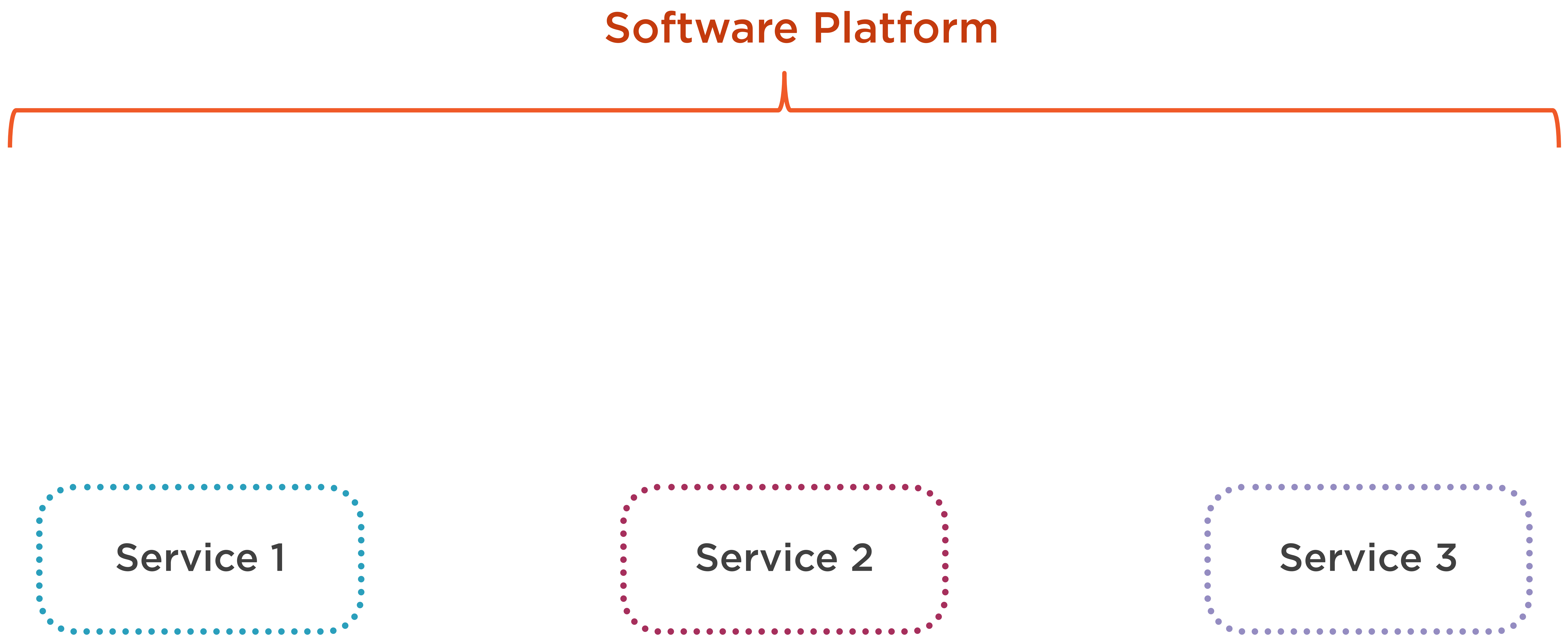
Pipeline Parallelization

Multiple Executors

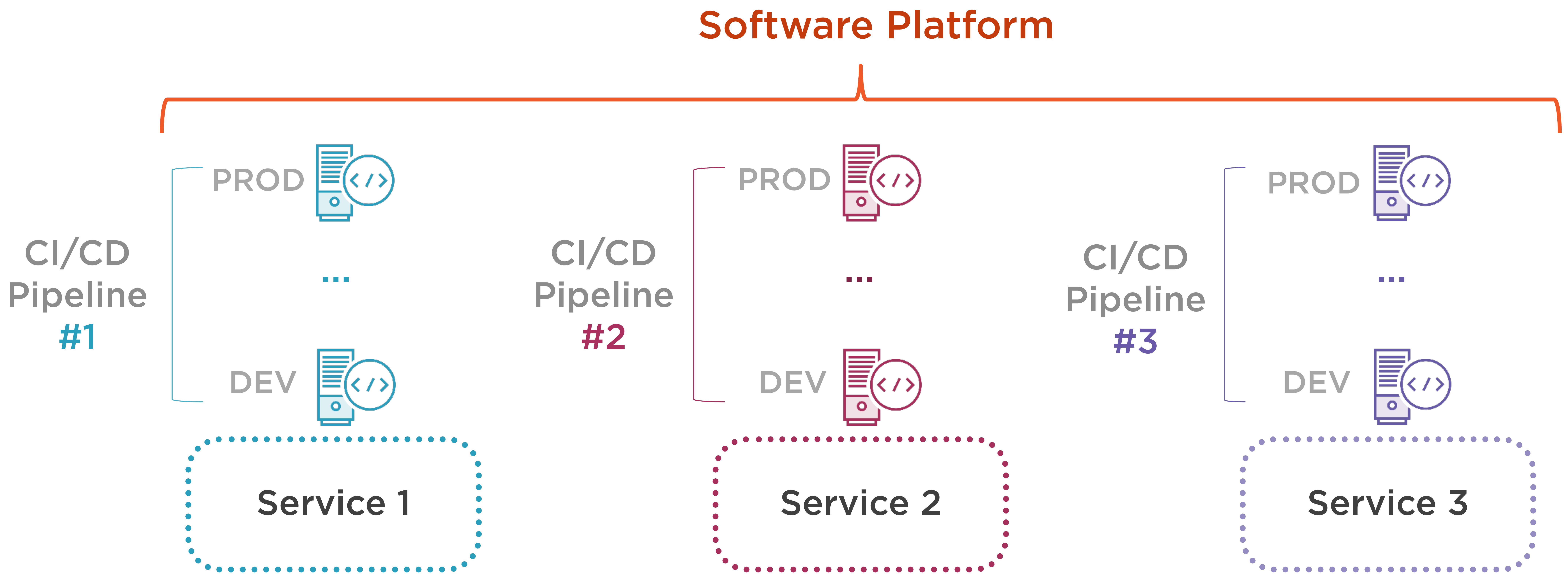
Java Application Performance tips

Business Case

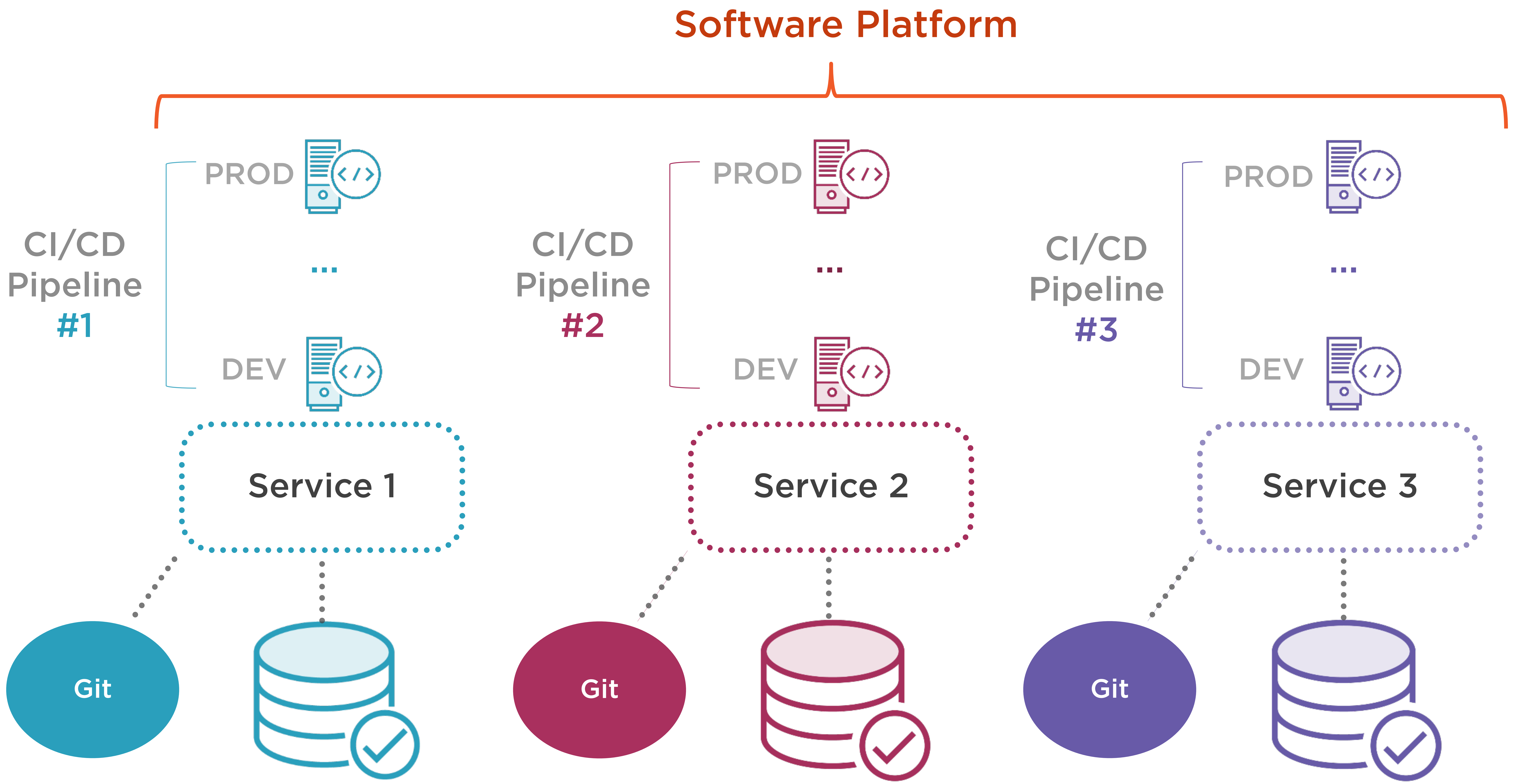
Micro Services Based Product



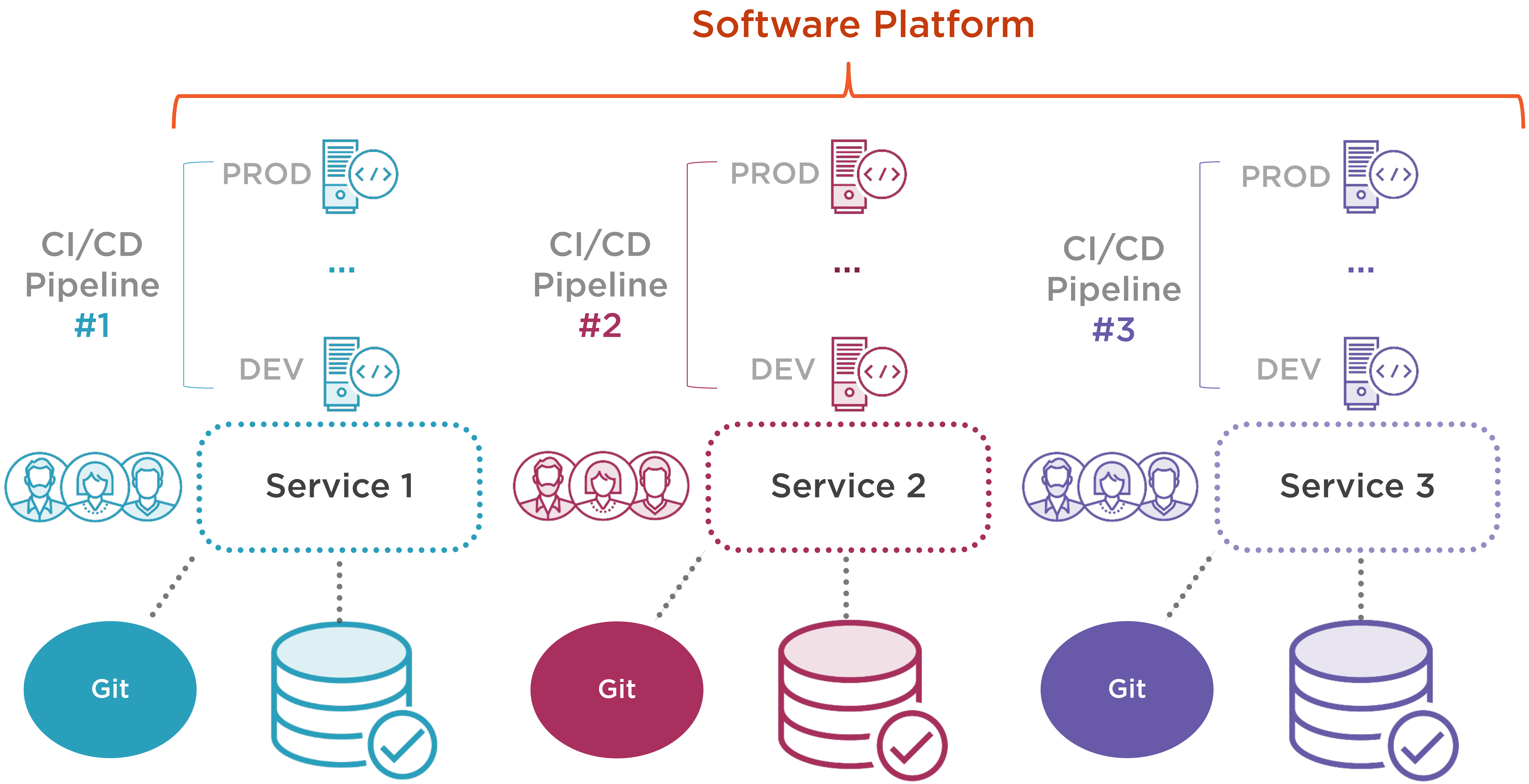
Micro Services Based Product



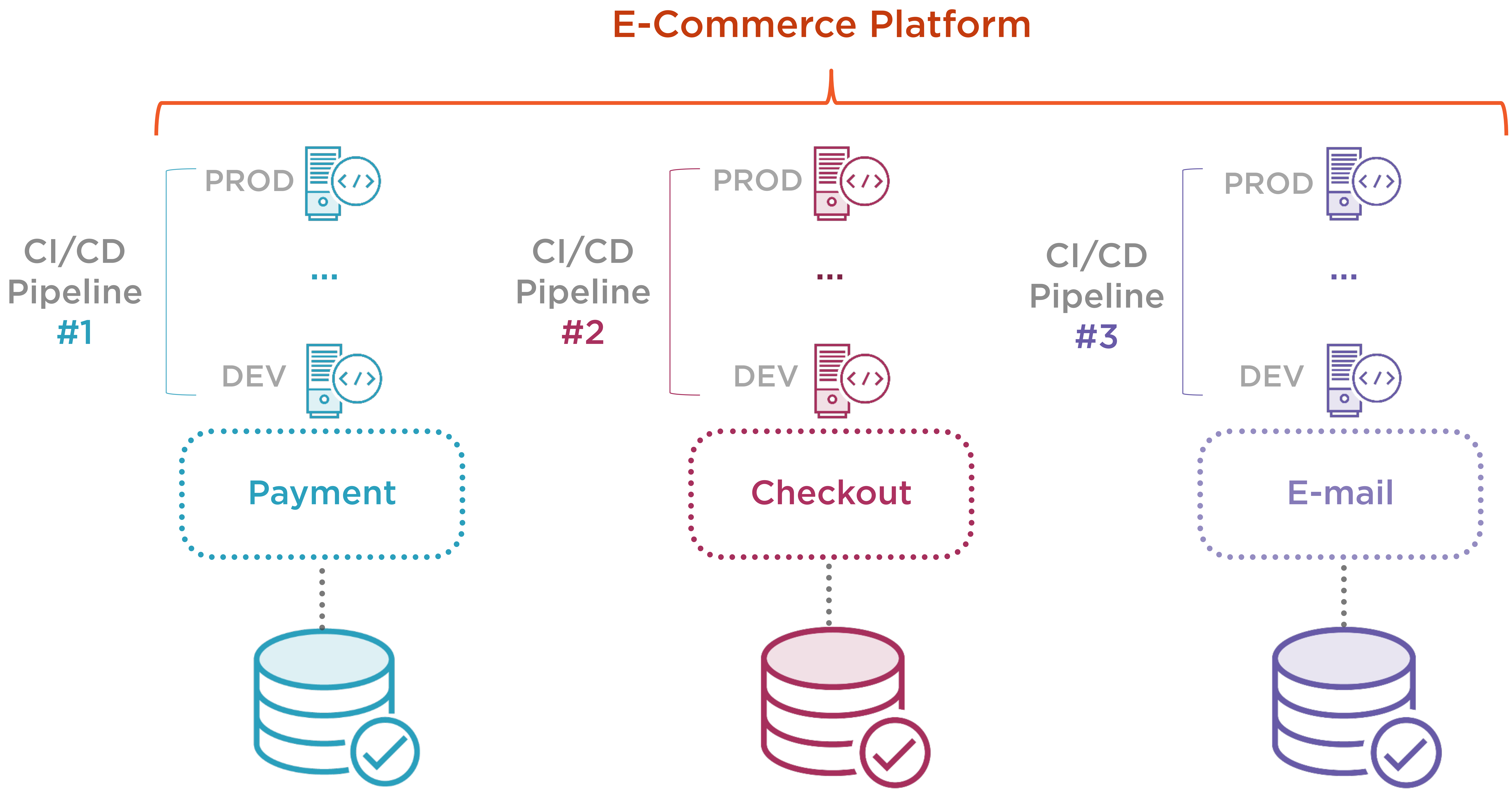
Micro Services Based Product



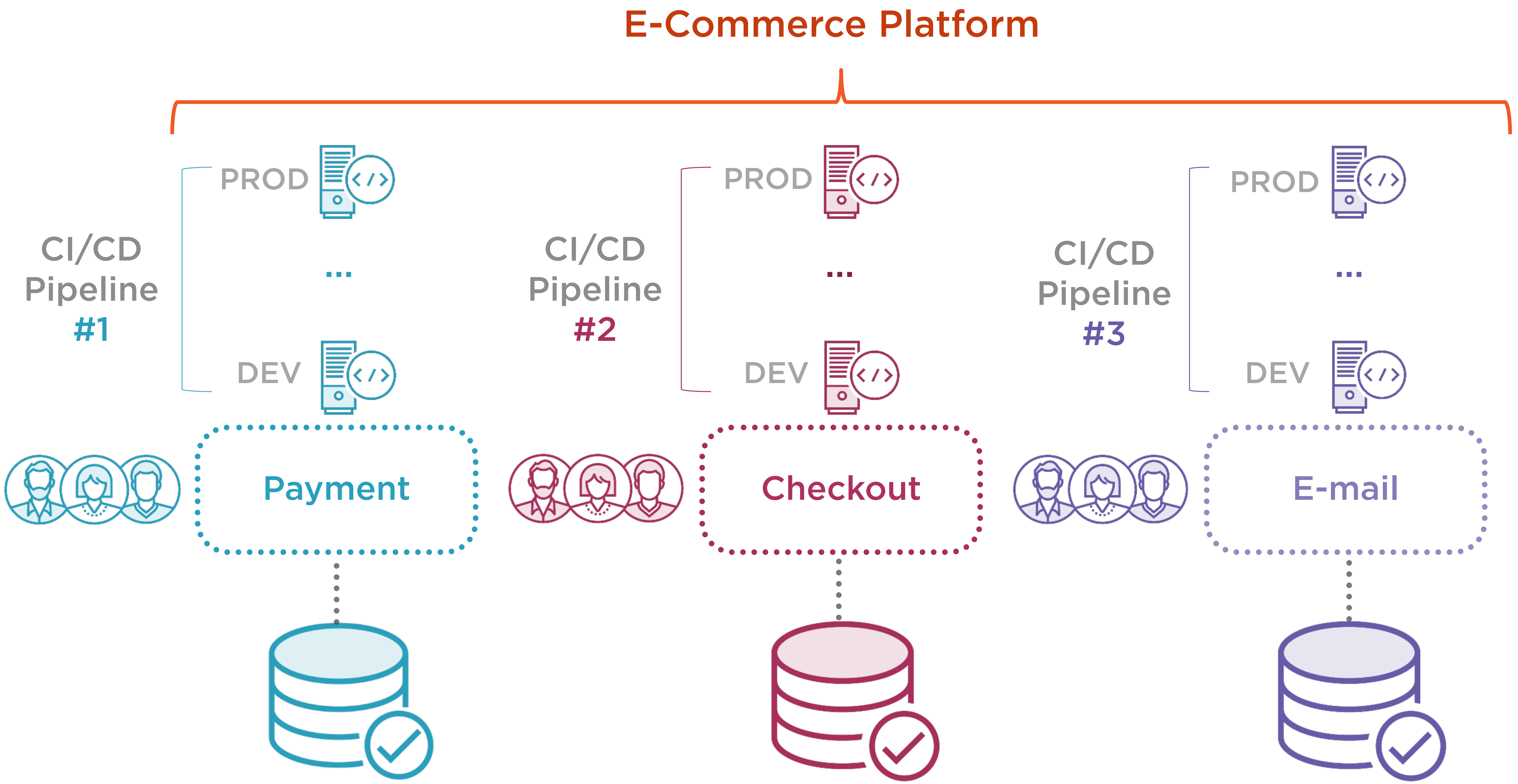
Micro Services Based Product



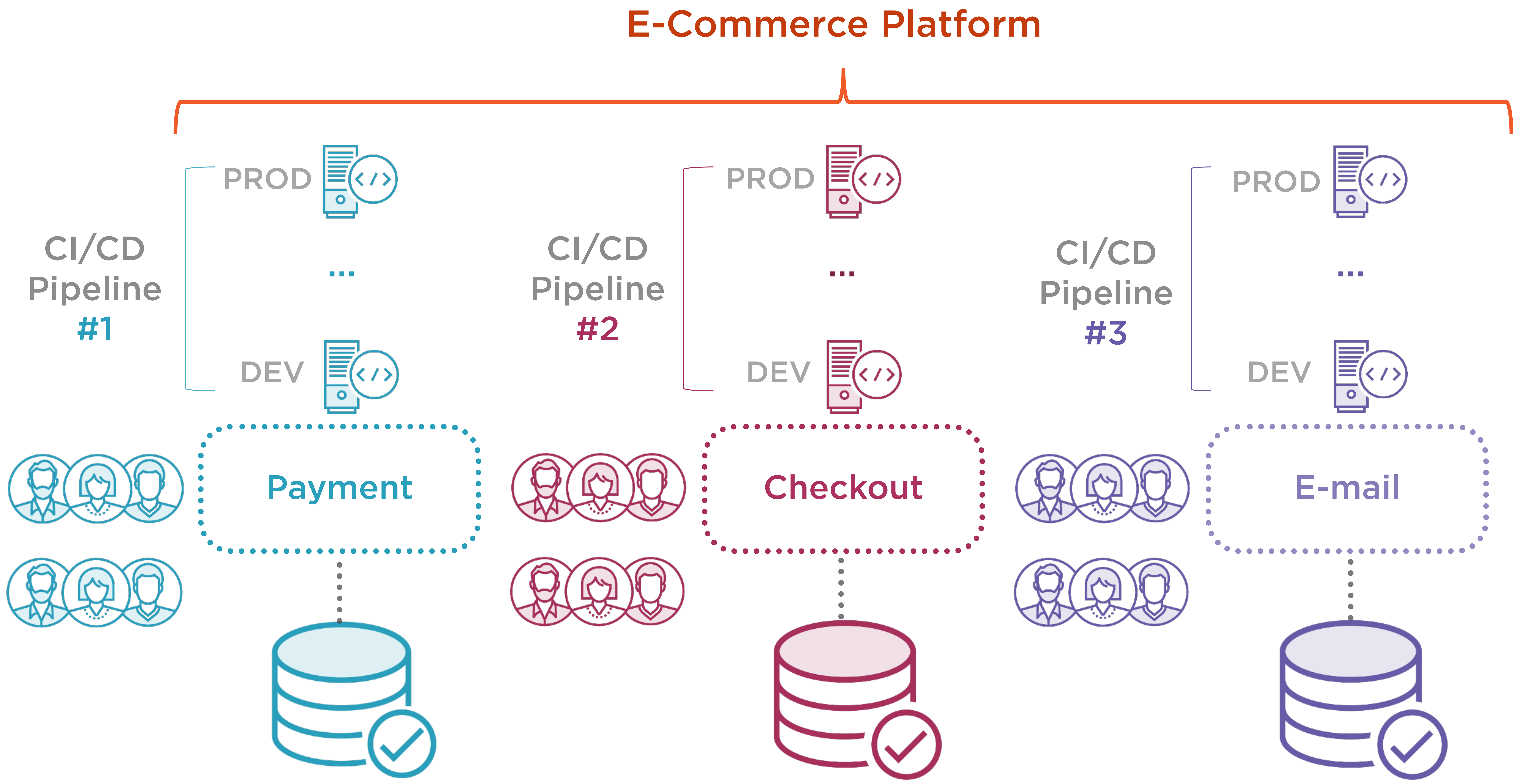
Micro Services Based Product



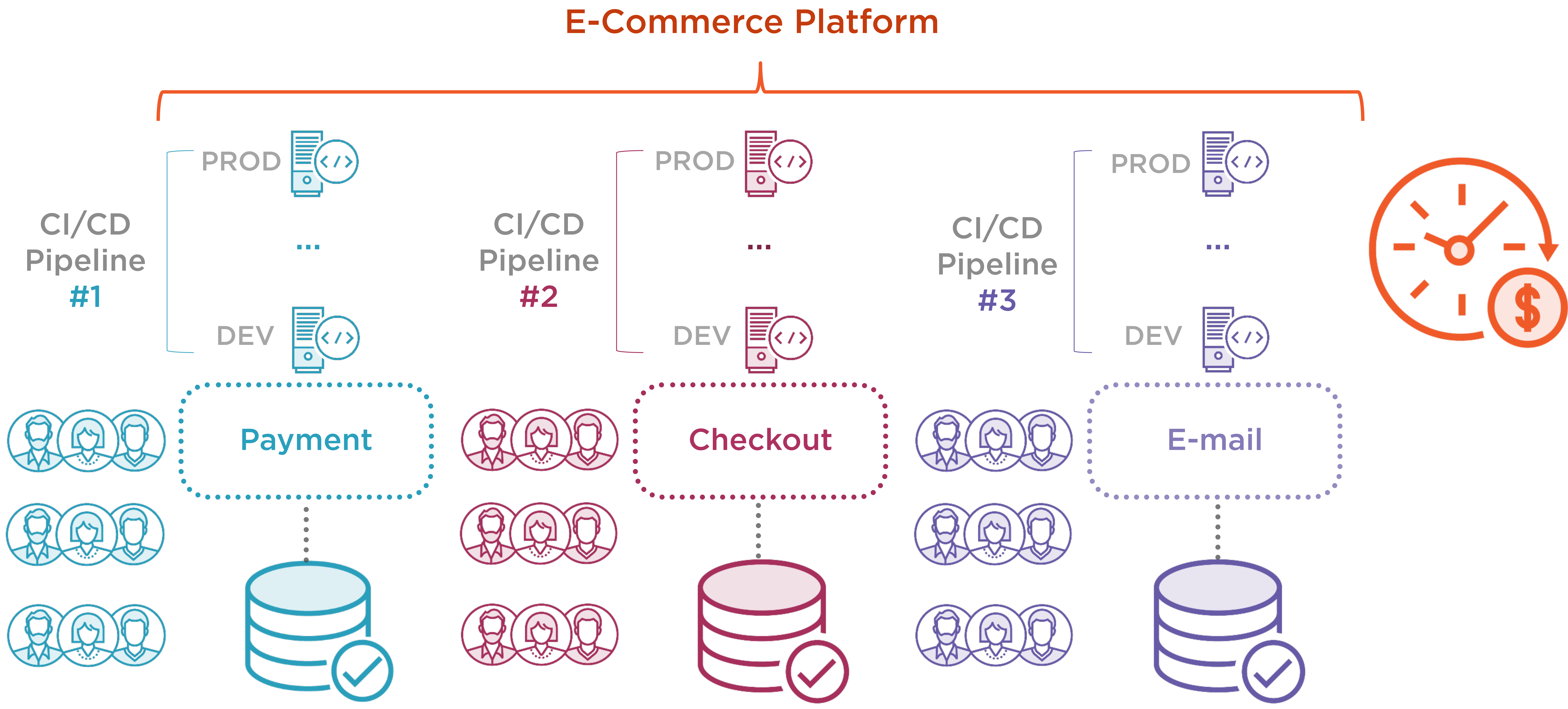
Micro Services Based Product



Micro Services Based Product



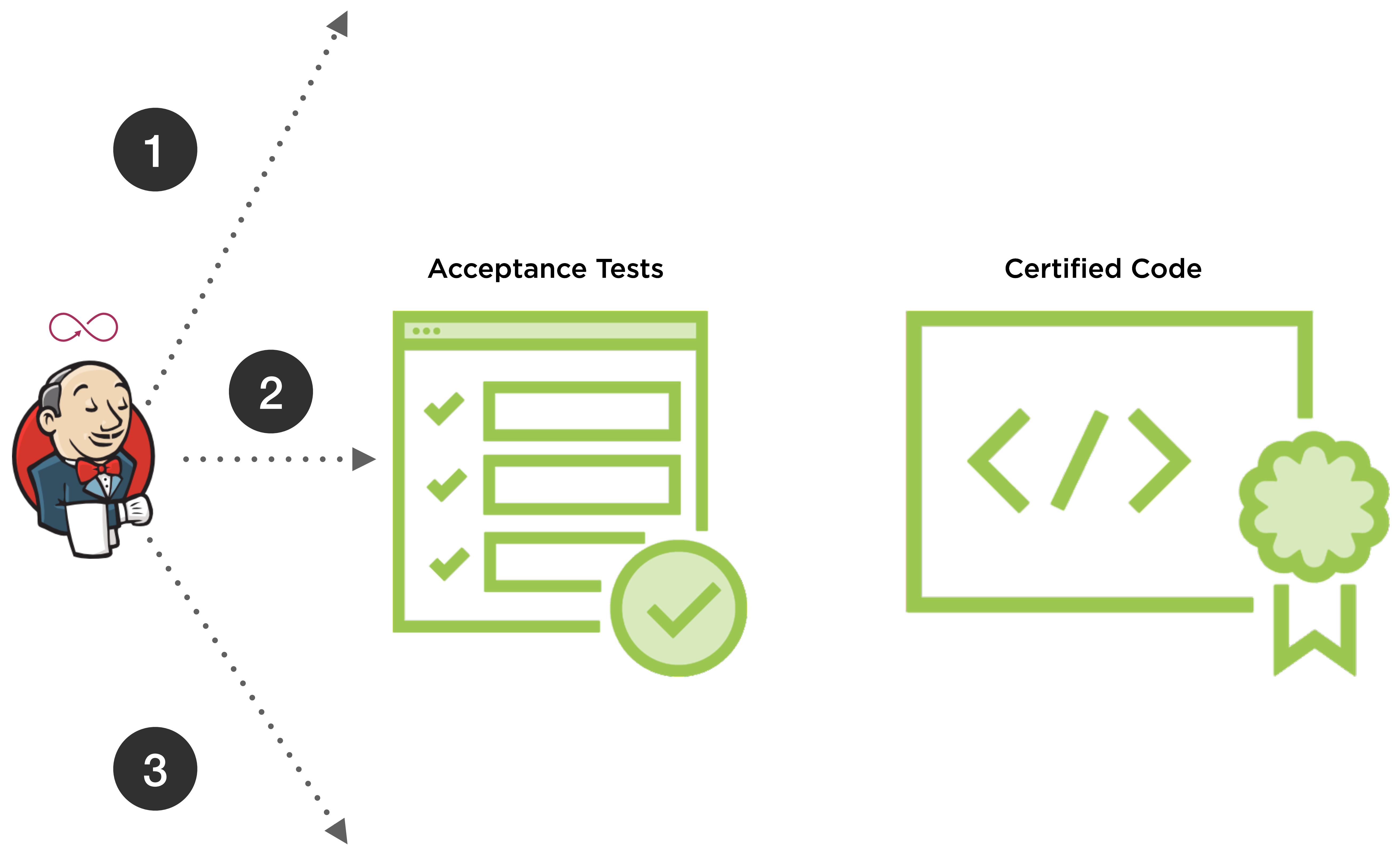
Micro Services Based Product



The Need for Speed

Job Parallelization

Acceptance Tests



The Number of tests grow over time.

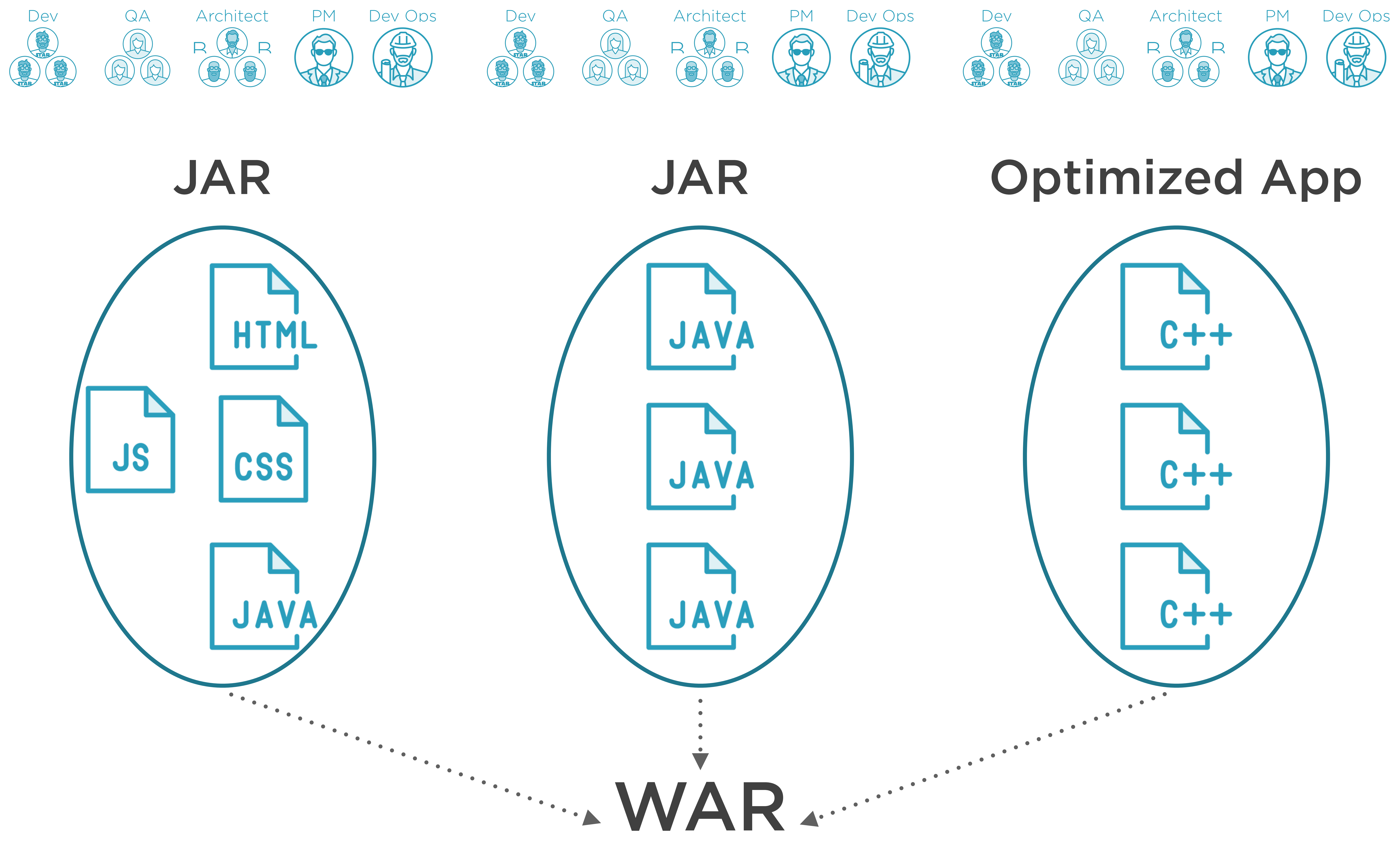
And so does the time needed to execute them

Up Next:

Pipeline Parallelization

Pipeline Parallelization

Independent Components



Pipeline Parallel Branch

A parallel unit of work in Jenkins. Not to be confused with branches in source control.

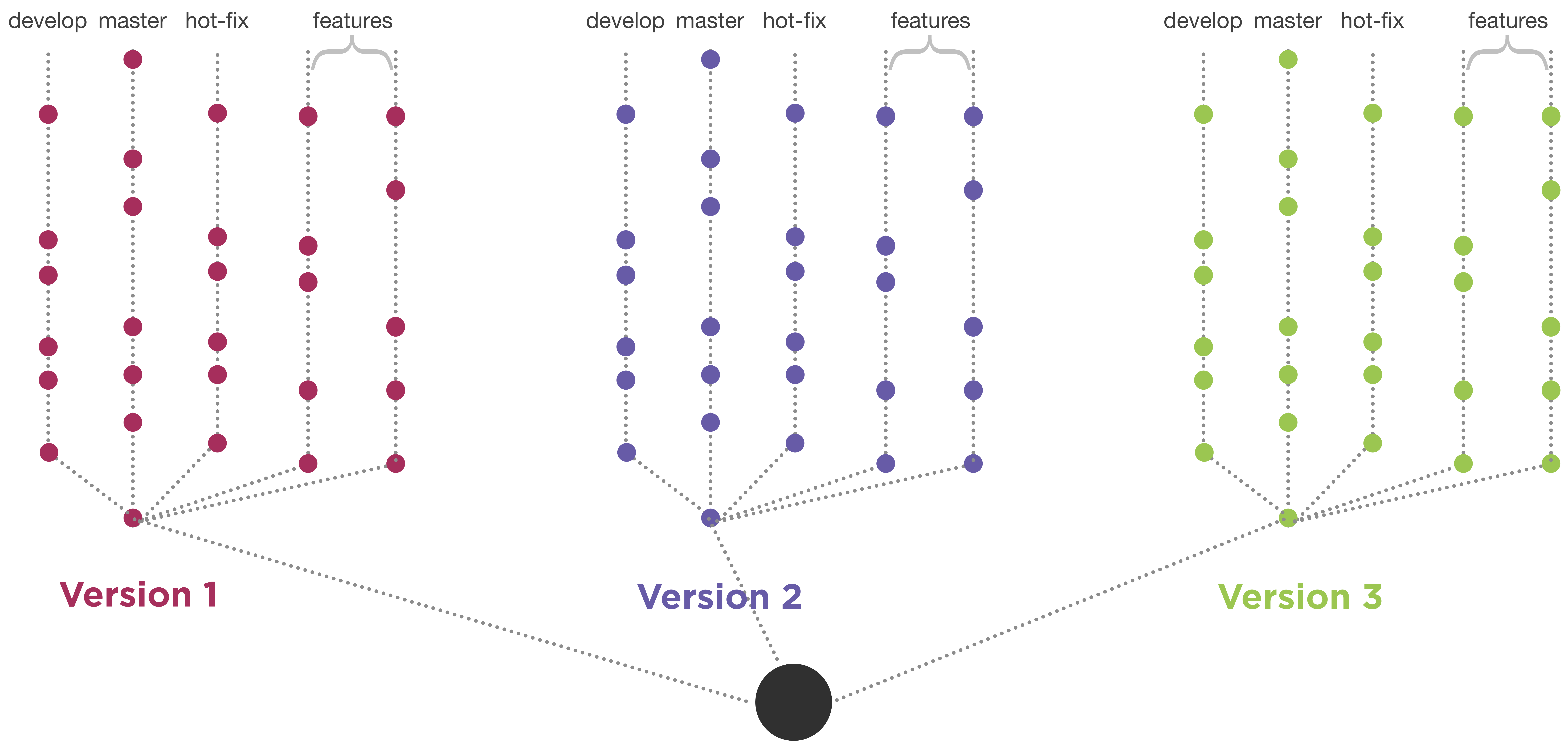
Customers on Different Software Versions

Version 1

Version 2

Version 3

Different Branches per Customer



Best Practices for Running Jenkins

Keep only the minimally required build history

Delegate builds to agents when you have large number of builds per day

Keep the number of 3rd party integrations to a minimum

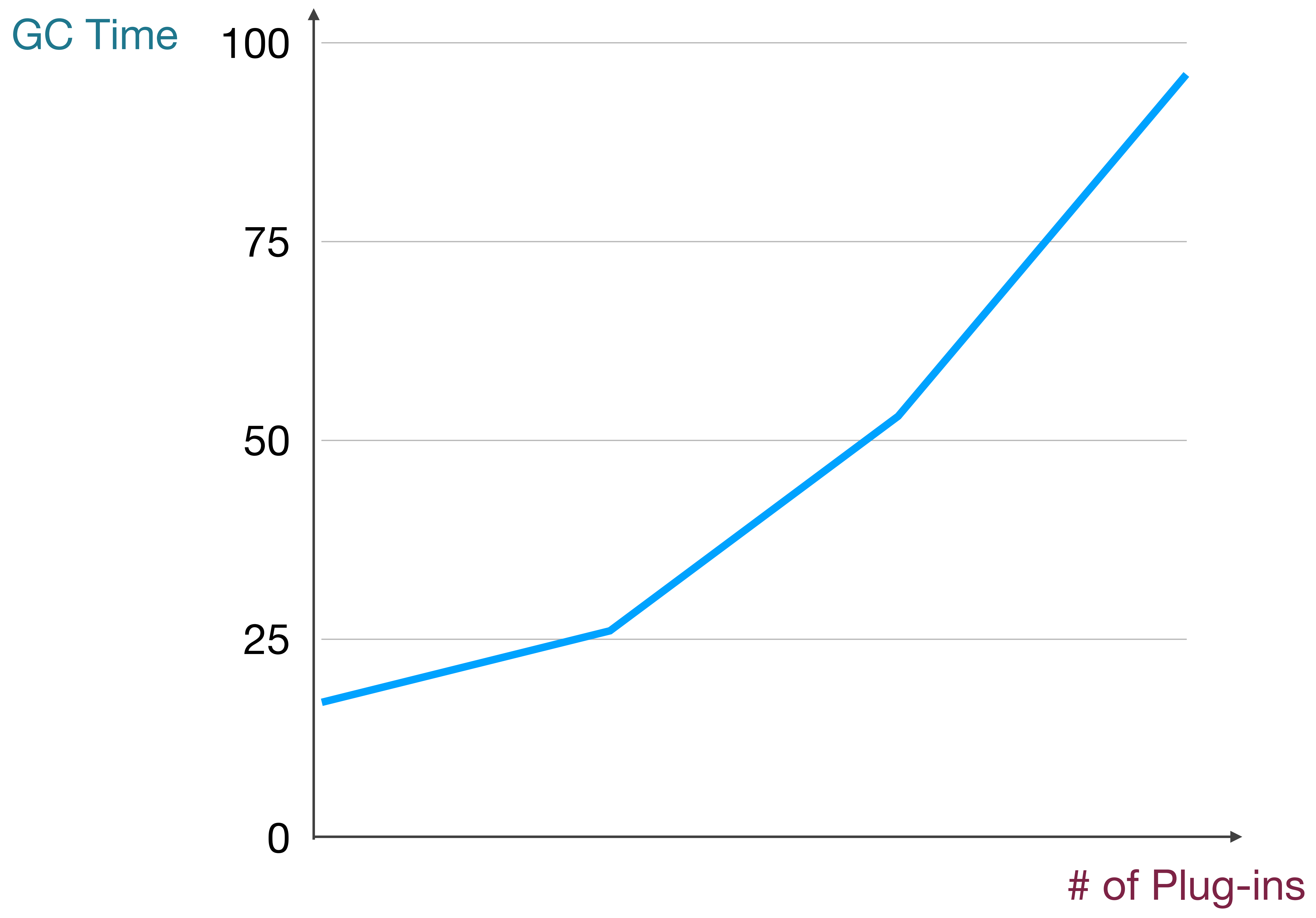
Disable or completely uninstall unused plugins

Java Application Performance Tips

Java Garbage Collection

Mechanism for releasing unused memory.

Number of Plug-ins & Garbage Collection



Tune the garbage collector

Make sure G1GC is enabled

- Default on JDK9+
- On JDK8 - enable manually with -XX:+UseG1GC

Garbage Collector Tuning Settings

```
Common: -server -XX:+AlwaysPreTouch

G1GC:

 -XX:+UseG1GC -XX:+ExplicitGCInvokesConcurrent -XX:+ParallelRefProcEnabled
 -XX:+UseStringDeduplication -XX:+UnlockDiagnosticVMOptions
 -XX:G1SummarizeRSetStatsPeriod=1

Garbage Collector Logging:

 -XX:+UseGCLogFileRotation-XX:NumberOfGCLogFiles=5 -XX:GCLogFileSize=20m

 -XX:+PrintGC -XX:+PrintGCDateStamps -Xloggc:$JENKINS_HOME/gc-%t.log

 -XX:+PrintGCDetails -XX:+PrintHeapAtGC -XX:+PrintGCCause

 -XX:+PrintAdaptiveSizePolicy -XX:+PrintTenuringDistribution

 -XX:+PrintReferenceGC
```

GC Tuning Resources

Responsiveness & Stability

How to tune Java settings to make the masters more responsive and stable

Large Heaps

How to tune Java settings when we deal with large heap sizes and resolve hangups

See the module resources for the links

Tune heap size

32 GB+ - enable manually with UseCompressedOops

Use plug-ins sparingly

- 1,700+ plug-ins available
- Disable/uninstall unused ones
- Ask a Java performance engineer for help

Trace Transactions

- With 3rd party systems
- Monitor and identify bottlenecks
- Resolve issues proactively

Throttle down logging

Other Tuning Optimizations

OS Level

- Services
- Logs
- Virtualization

Application Level

- Lazy loading
- Efficient patterns
- Caching

Network Level

- Traffic isolation
- Network speed
- Protocols

Summary

You learned:

- Solving a Complex Business Case
- Job & Pipeline Optimization Techniques
- General Java Application Tuning

Up Next:

Managing Distributed Build Farms