The good, the bad, the native

Gregorio Palamà

Gregorio Palamà

DevOps & Cloud Engineer **@Finwave**

Google Cloud Innovator Champion

Community Manager **@GDG Pescara**

What
is
Cloud
Native

“Cloud-native technology is when engineers and software people utilize *cloud computing* to build tech that’s **faster** and more **resilient**, and they do that to meet customer demand really **quickly**.”

Priyanka Sharma, CNCF’s General Manager

https://learn.microsoft.com/en-us/dotnet/architecture/cloud-native/definition

https://aws.amazon.com/what-is/cloud-native/

https://cloud.google.com/learn/what-is-cloud-native

Microservices

Containers

Orchestration

Scalability

Cloud-native architectures employ infrastructure automation, helping to eliminate downtime due to human error. You can balance load based on demand, allowing you to **optimize cost** and **performance better**.

Lower costs

A streamlined software delivery process reduces the costs of delivering new updates and features. Cloud-native applications also allow for **sharing resources** and **on-demand consumption**, significantly lowering your operating costs.

Higher availability

Cloud-native architectures provide high availability and reliability as they reduce operational complexity, simplify configuration changes, and offer **autoscaling** and **self-healing**.

Cloud-native applications make the most of modern infrastructure's dynamic, distributed nature to achieve **greater speed**, agility, **scalability**, reliability, and **cost efficiency**.

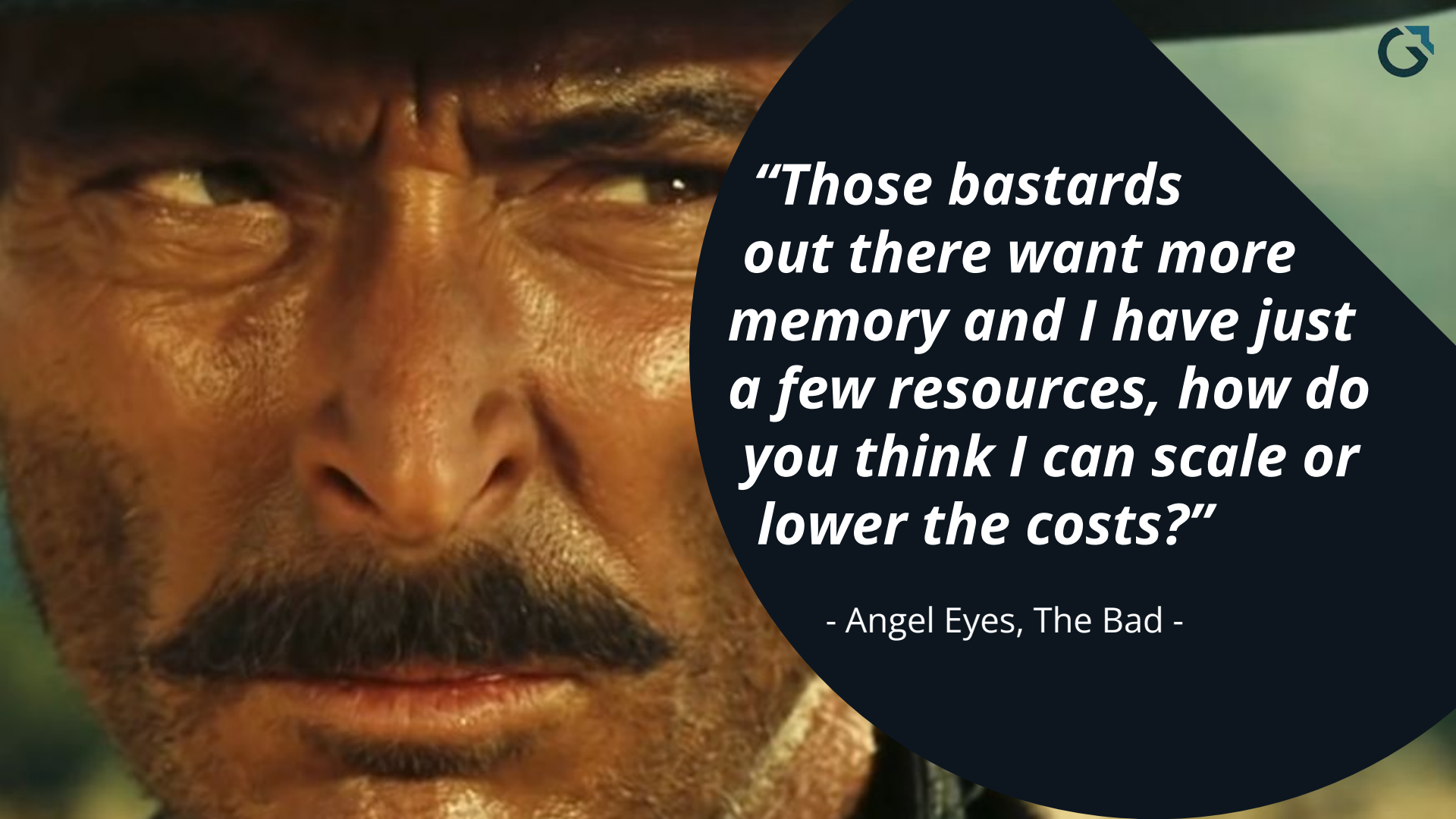
Smaller memory footprint

Less CPU consumption

Lower startup time

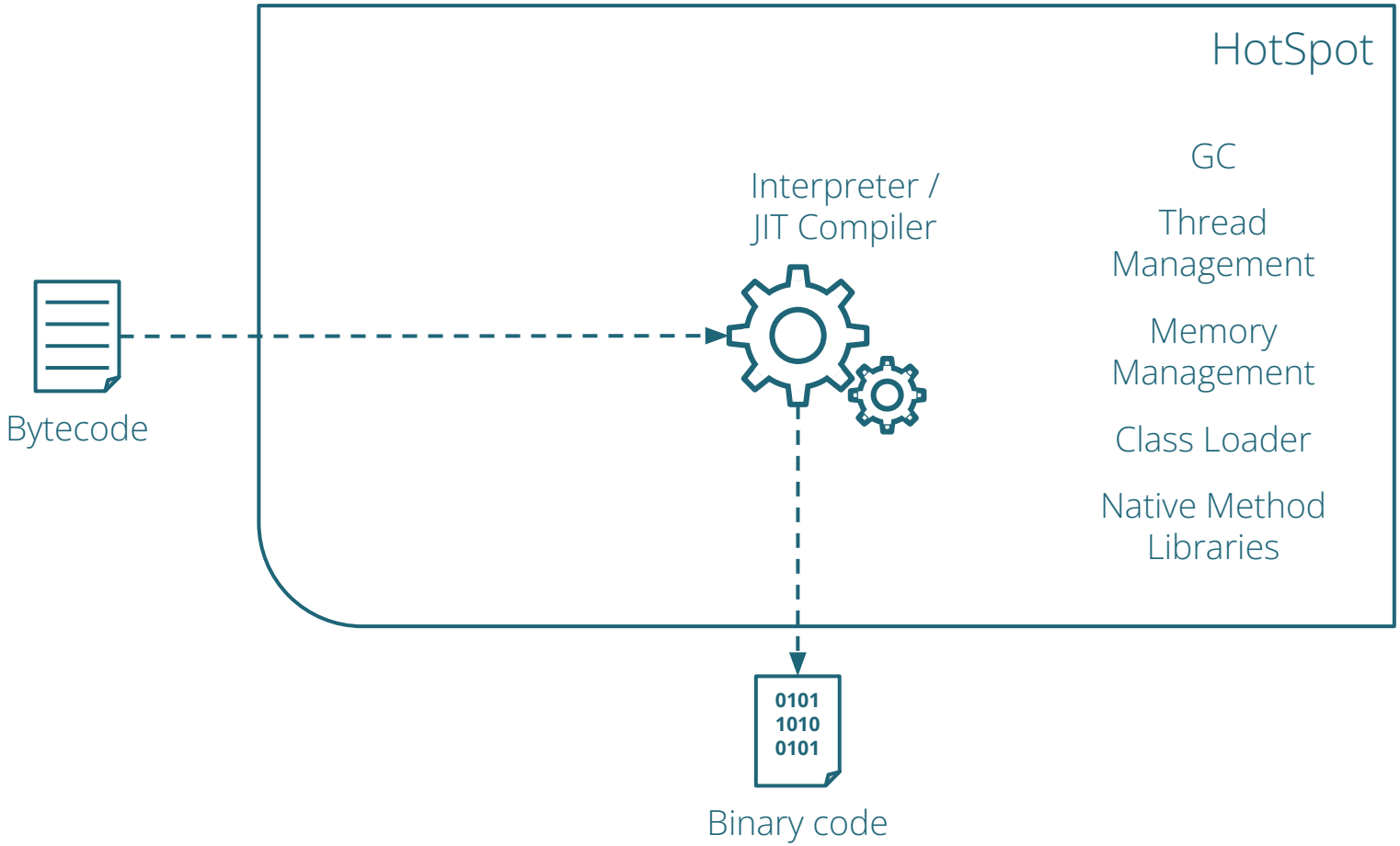
The JVM microservices frameworks ecosystem

Demo project


“Those bastards out there want more memory and I have just a few resources, how do you think I can scale or lower the costs?”
- Angel Eyes, The Bad -

GraalVM

The JVM is an **abstraction** of an underlying actual machine that **interprets** the *bytecode* generated by the compilation of a code supported by the JVM itself.


HotSpot
GC
Thread Management
Memory Management
Class Loader
Native Method Libraries
Interpreter / JIT Compiler
Bytecode
0101
1010
0101
Binary code

Interpreter

- Slow execution
- Interprets bytecode and collects profiling informations
- Fast startup

C1

- JIT
- Compiles code when it gets frequently executed
- Continue collecting profiling information
- Fast warmup

C2

- JIT
- Compiles and optimizes code when it's executed often enough and reaches certain thresholds
- Uses profiling informations
- High peak performance

Polyglot VM

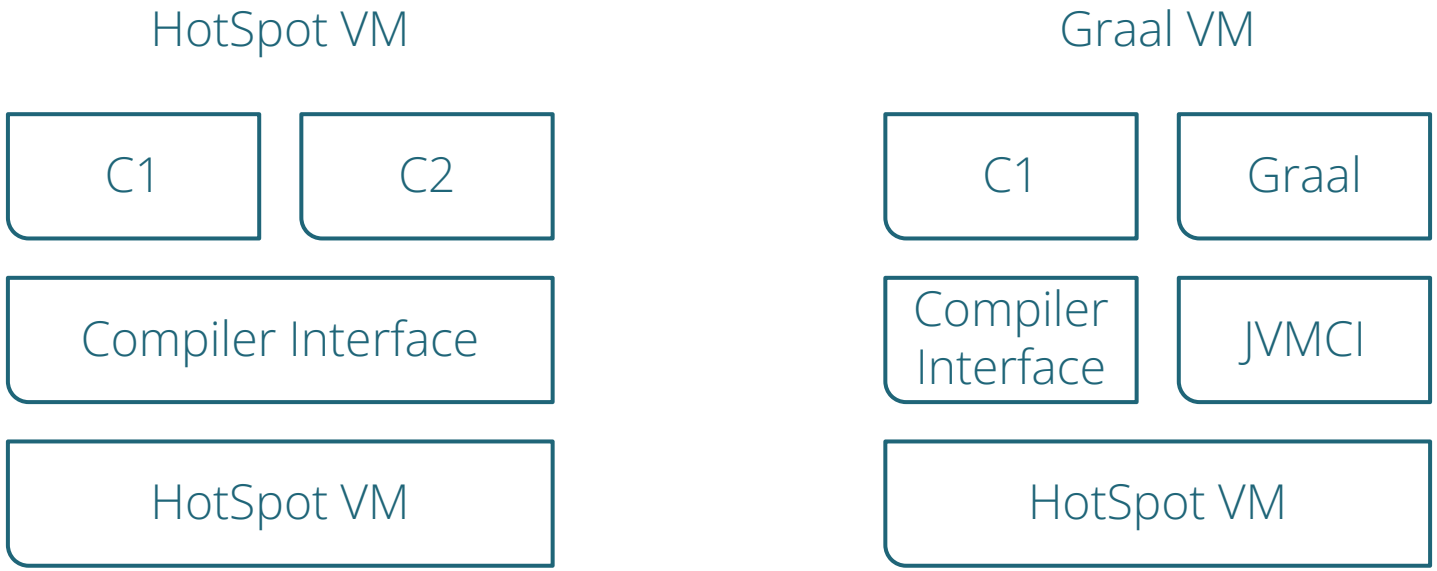
- Runtimes for many languages: Python, Ruby, Javascript, etc.

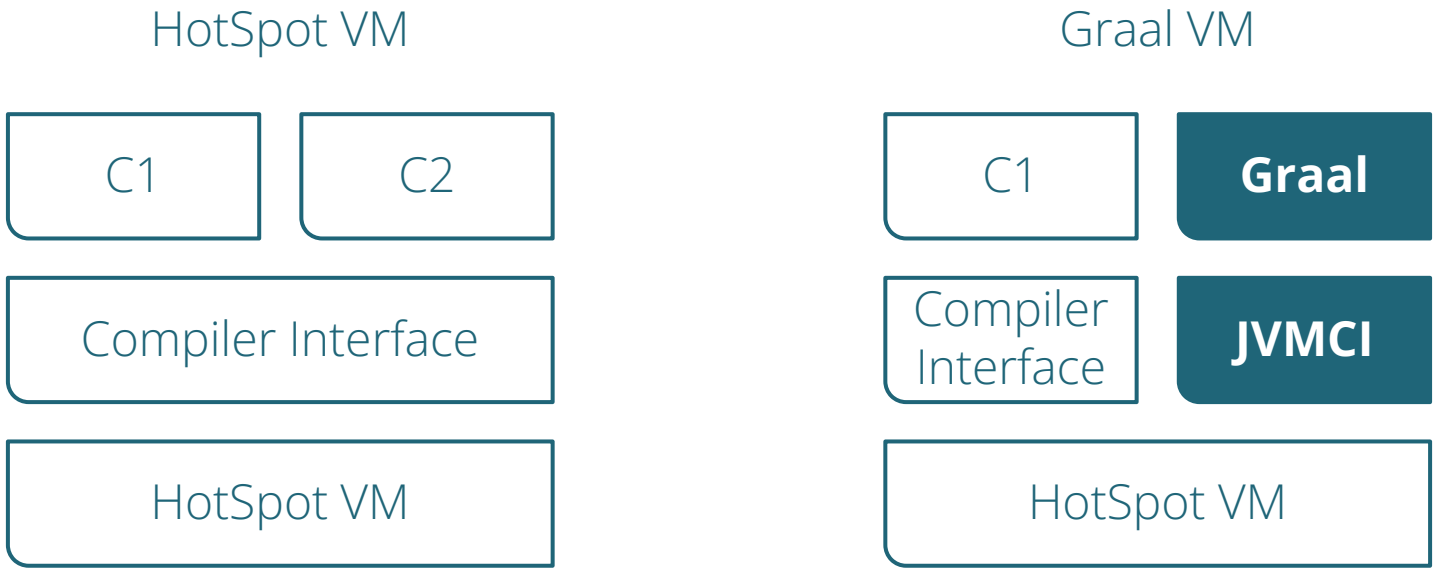
Graal compiler

- C2 implementation
- Various optimizations
- Removes unnecessary object allocations on the heap

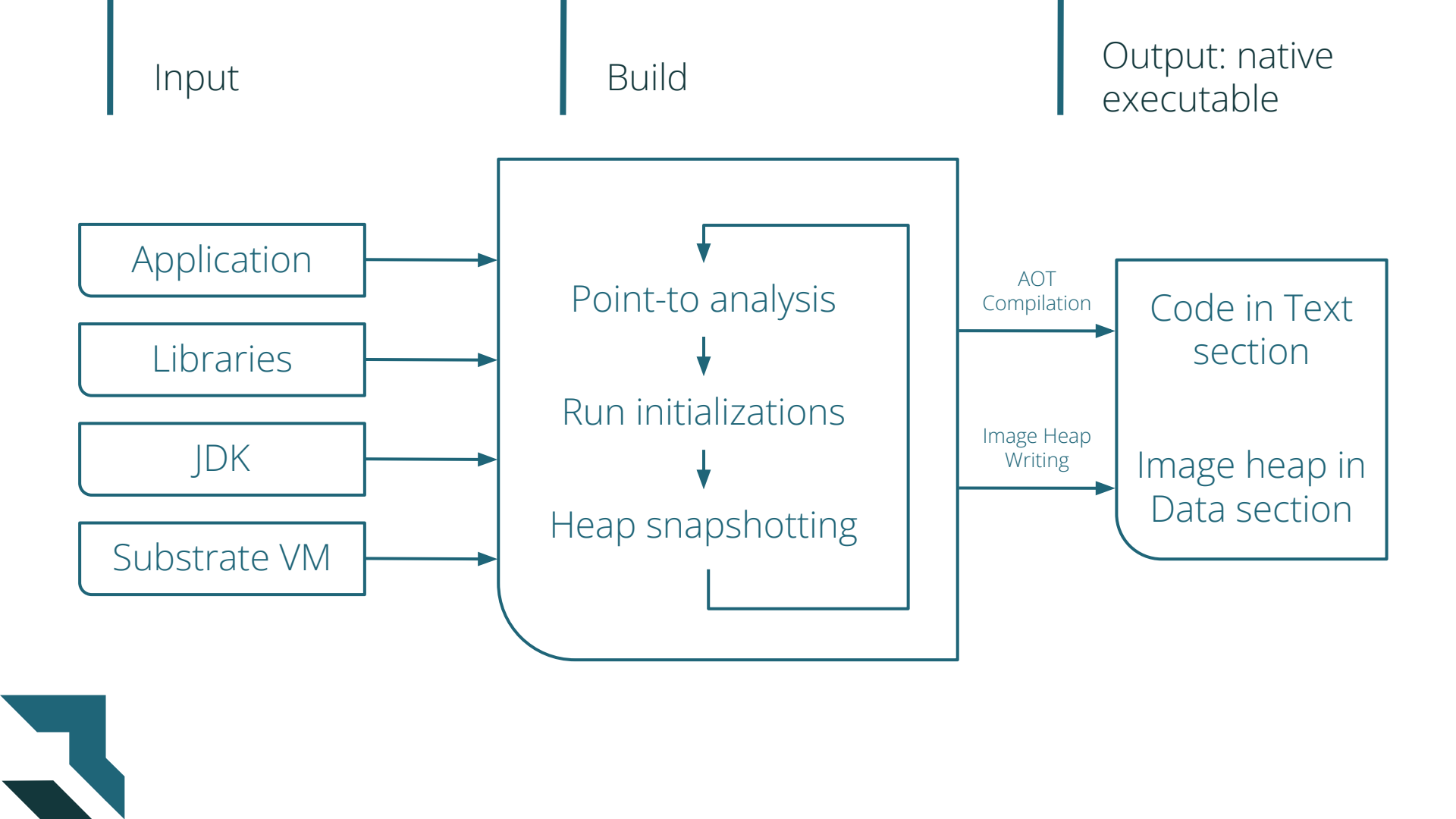
Native image

- AOT
- Compiles to native platform executables


HotSpot VM
C1
C2
Compiler Interface
HotSpot VM
Graal VM
C1
Graal
Compiler Interface
JVMCI
HotSpot VM


HotSpot VM
C1
C2
Compiler Interface
HotSpot VM
Graal VM
C1
Graal
Compiler Interface
JVMCI
HotSpot VM

Ahead
Of
Time


Input
Build
Output: native executable
Application
Libraries
JDK
Substrate VM
Point-to analysis
Run initializations
Heap snapshotting
AOT Compilation
Image Heap Writing
Code in Text section
Image heap in Data section

Demo project - Native

Smaller memory footprint

Less CPU consumption

Lower startup time

Scalability

Lower costs

Higher availability

“I… I will sleep peacefully… because I know that the native… is watching over me”
- Blondie, The Good -

Native building drawbacks

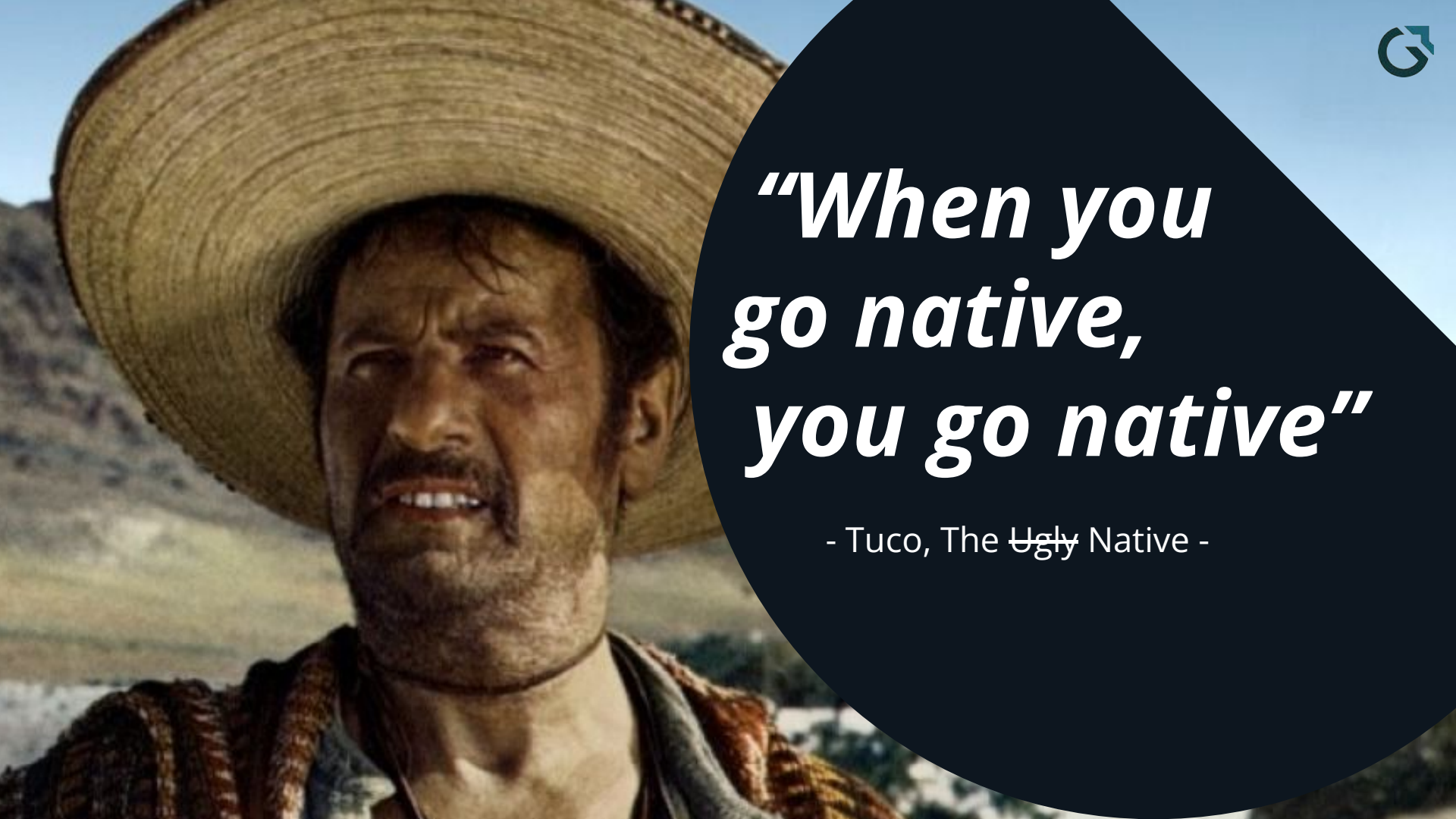
Native image takes **a lot** of time and needs **more resources** than bytecode generation.

Native image generates metadata performing static analysis under a **closed-world** assumption. Thus, some **dynamic** features require additional configuration: *reflection*, *dynamic proxying*, etc.

The **Tracing Agent** can be used to easily gather metadata and prepare configuration files.

Some libraries does not provide good enough **reachability metadatas**. Some include **dependencies** that we may need to manually exclude.

It is better to **avoid** *shaded* libraries.


“When you go native, you go native”
- Tuco, The ~~Ugly~~ Native -

“Cloud-native technology is when engineers and software people utilize *cloud computing* to build tech that’s **faster** and more **resilient**, and they do that to meet customer demand really **quickly**.”

Priyanka Sharma, CNCF’s General Manager

Start using GraalVM native image

Use the Tracing Agent

Test the native executable or perform native tests

https://www.graalvm.org/22.0/reference-manual/native-image/

https://www.graalvm.org/latest/reference-manual/native-image/metadata/AutomaticMetadataCollection/

https://docs.spring.io/spring-boot/docs/current/reference/html/native-image.html

https://quarkus.io/guides/building-native-image

Thank you