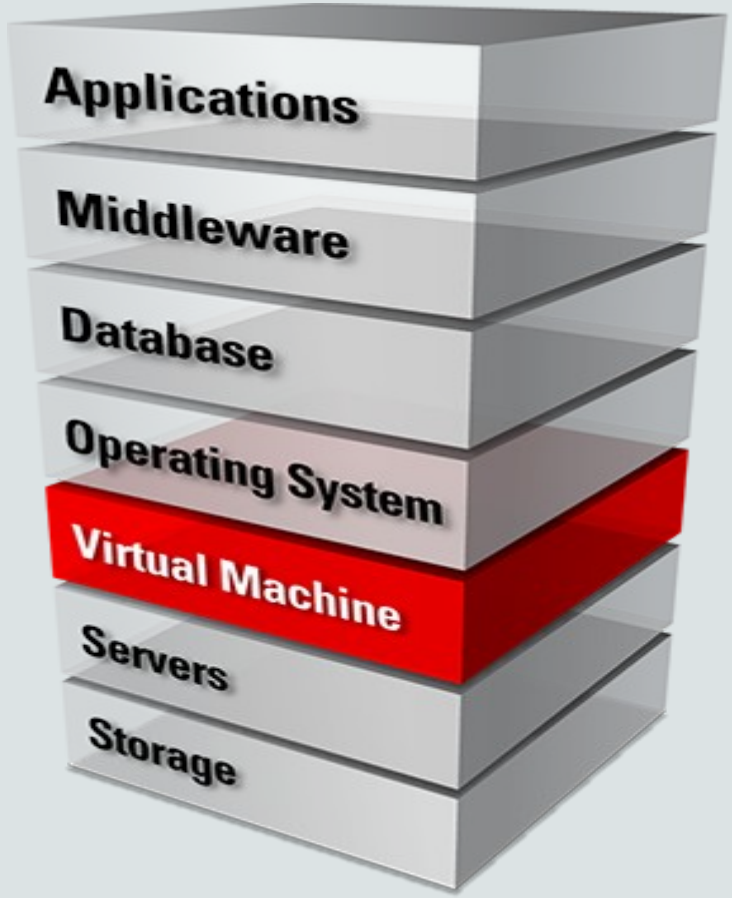

ORACLE

On the Quest Towards

Fastest (Java) Virtual Machine

on the Planet!



@JaroslavTulach
Oracle Labs

Jaroslav Tulach

1. NetBeans Founder
2. NetBeans Initial Architect
3. Practical API Design book
4. Oracle Labs: Graal/Truffle

Program Agenda

1. The Vision
2. Graal VM
3. Truffle Languages
4. References & Discussion

Oracle Labs Vision

- Vision without speculation would be today's reality
  - Radical innovation
  - New approach to existing technologies
  - Research + Engineering
- Hardware & Software
  - Engineered to work together
  - Java/bytecode/IR/assembly/processors/memory

Graal Compiler Vision Statement

"Create an **extensible**, **modular**,

dynamic, and **aggressive** compiler using

object-oriented and **reflective** Java programming,

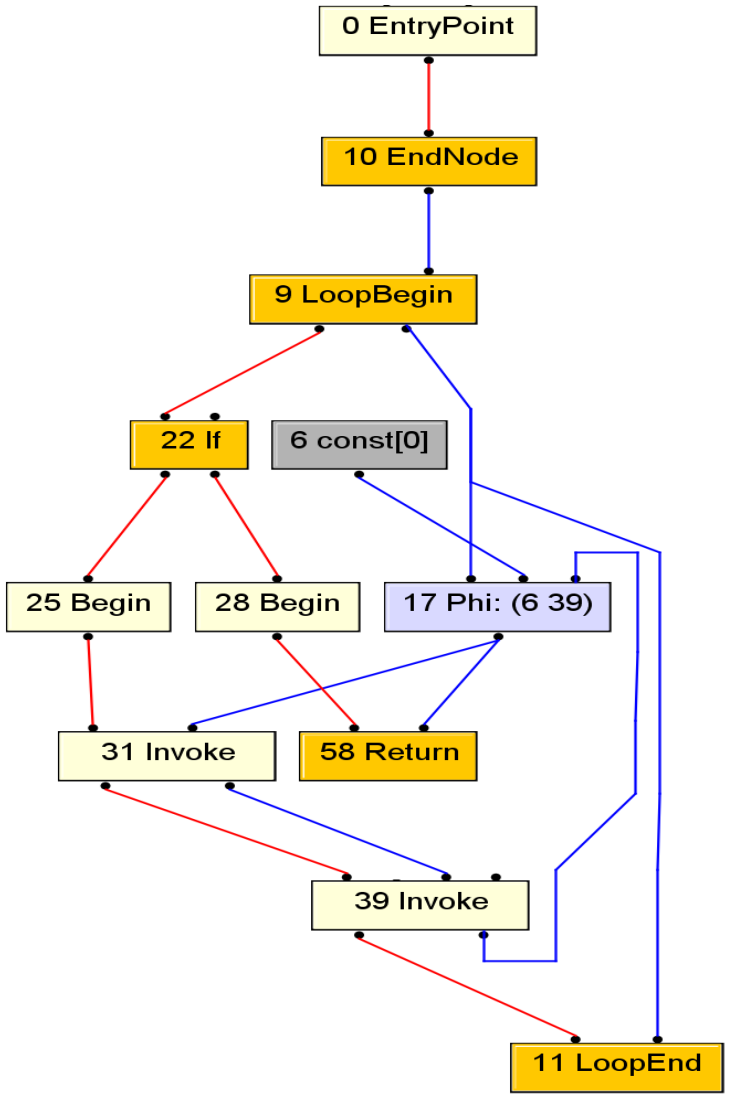
a **graph-based**

and **visualizable** intermediate representation,

and Java **snippets**."

Thomas Würthinger

Graal Virtual Machine



- Modern alternative to HotSpot C2
  - Maintainable code base
  - Toolable, approachable
  - Ready for today's code
  - JEP 243: Java Compiler Interface
- Partial evaluation
- Aggressive speculations
- Smooth de-optimizations

Truffle: “Write Your Own Language”

Current situation

Prototype a new language

Parser and language work to build syntax tree (AST), AST Interpreter

Write a “real” VM

In C/C++, still using AST interpreter, spend a lot of time implementing runtime system, GC, …

People start using it

People complain about performance

Define a bytecode format and write bytecode interpreter

Performance is still bad

Write a JIT compiler
Improve the garbage collector

How it should be

Prototype a new language in Java

Parser and language work to build syntax tree (AST)
Execute using AST interpreter

Integrate with VM-building framework

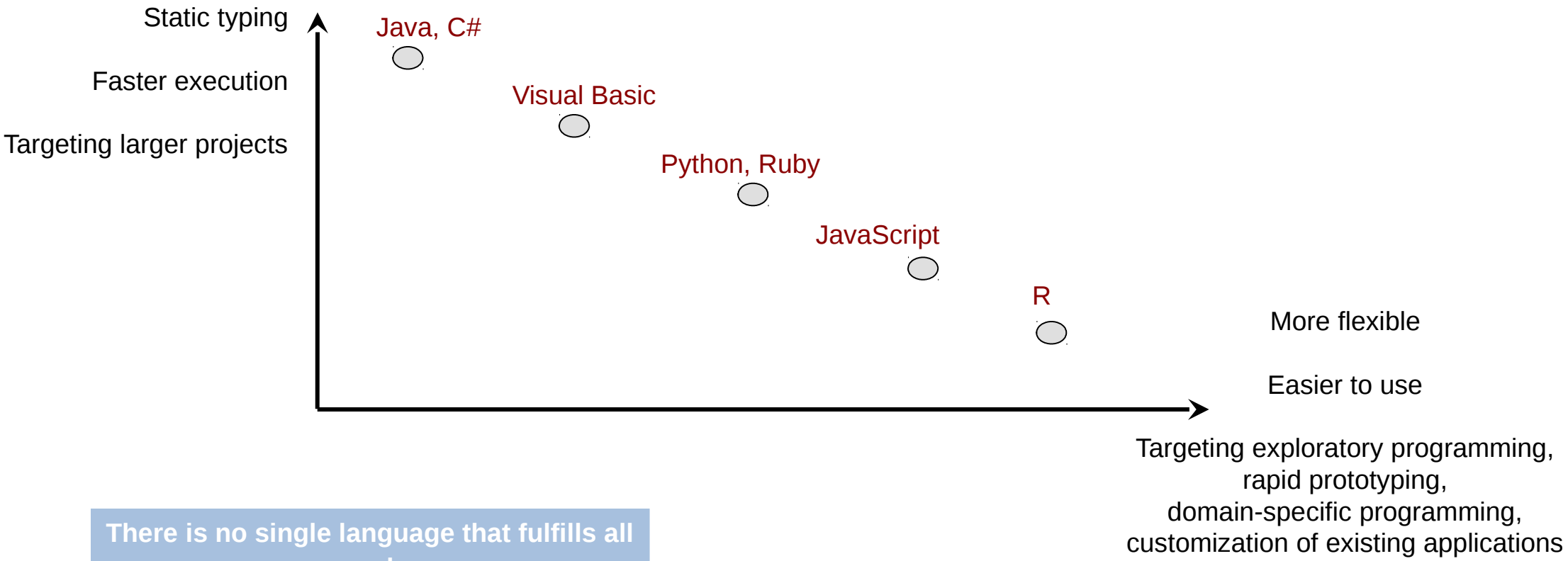
Integrate with Modular VM
Add small language-specific parts

People start using it

And it is already fast

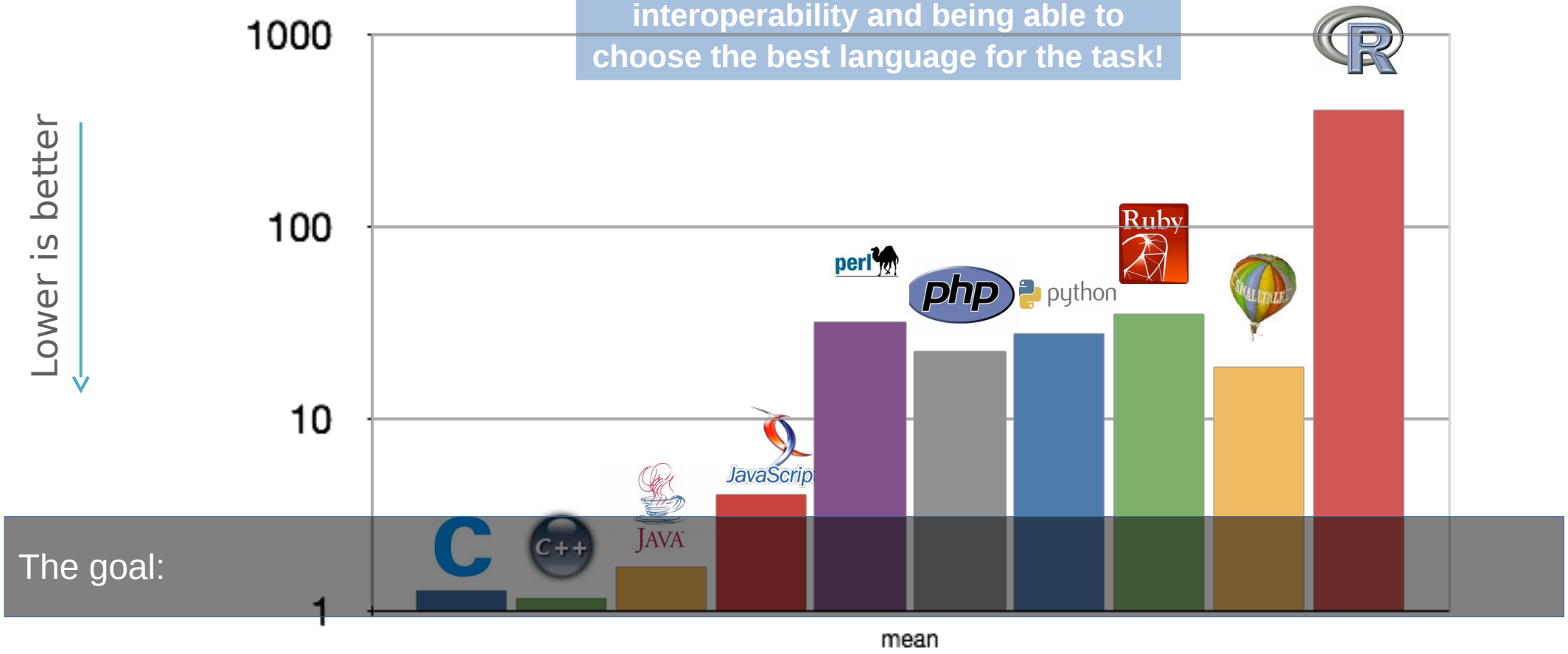
A Spectrum of Programming Languages

Do you care about your code or your data?



There is no single language that fulfills all needs

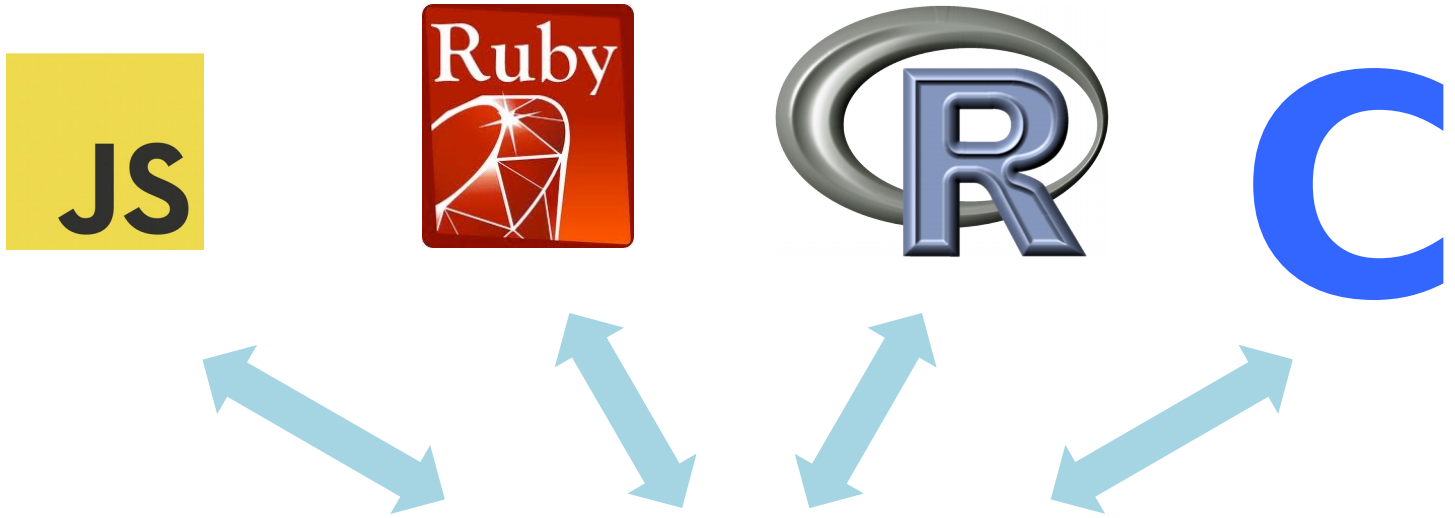
One VM for all languages means interoperability and being able to choose the best language for the task!



The goal:

You can execute any language on the JVM / CLR

- as long as it looks like Java / C#.



VM Expert

Truffle

Java

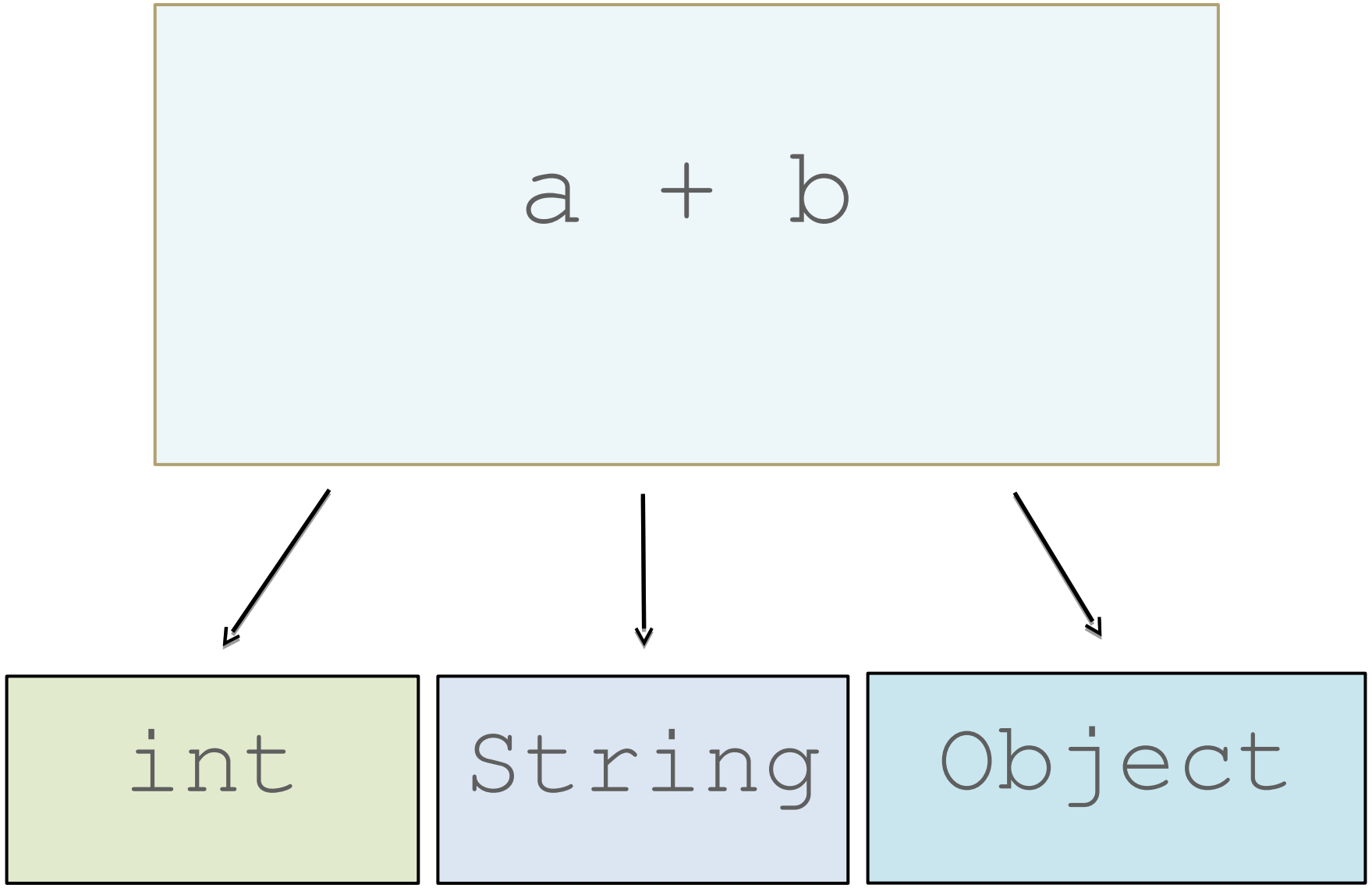
Graal VM

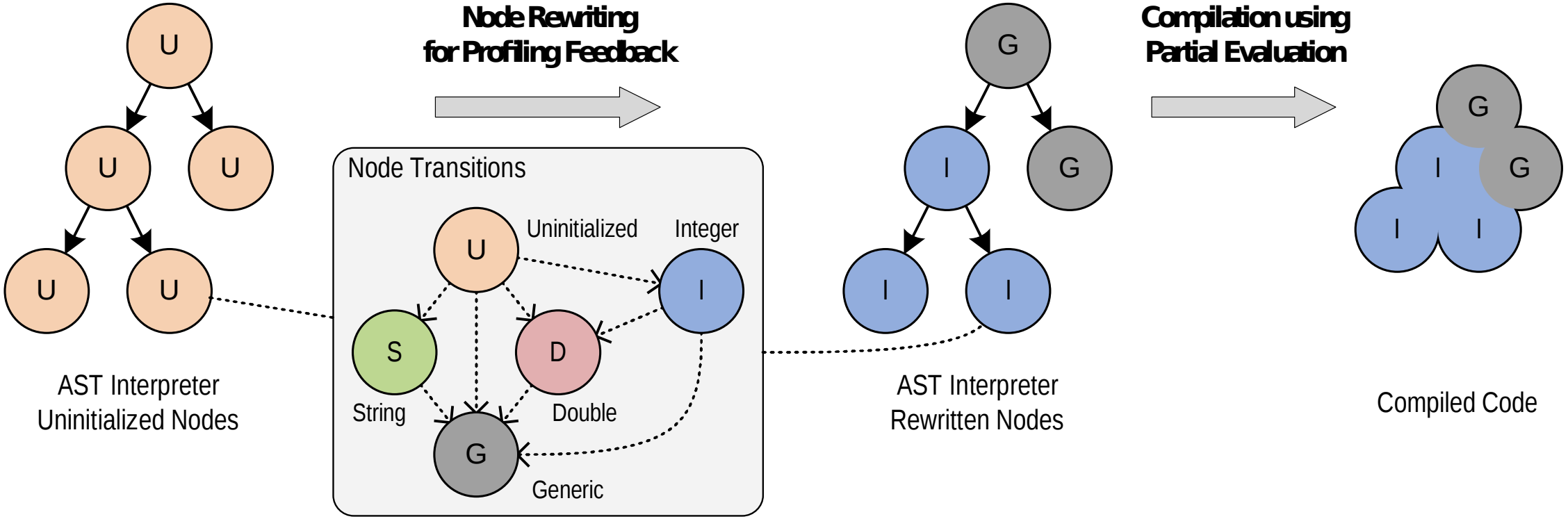
Java / C++

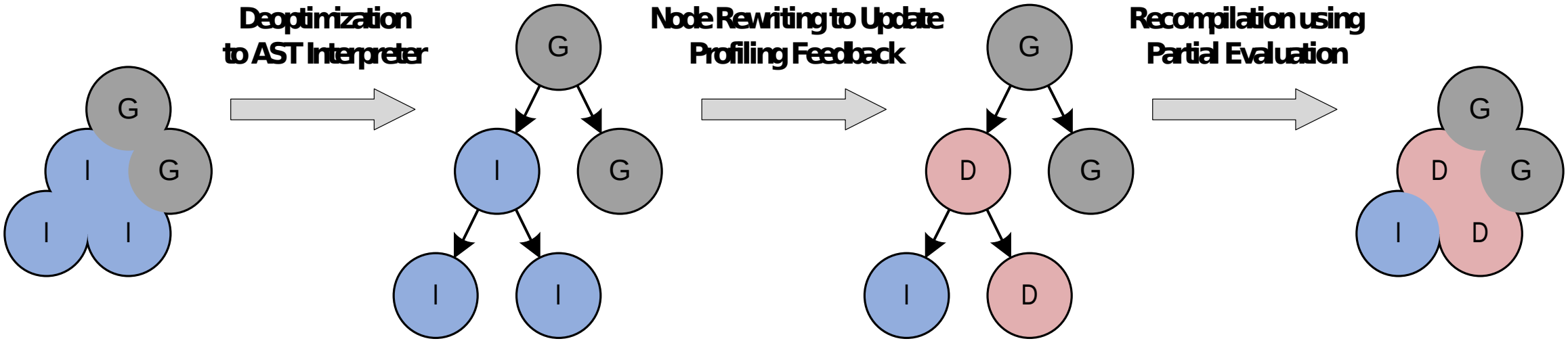
OS Expert

OS

Unmanaged Language
(typically C or C++)







Safe Harbor Statement

The preceding and following is intended to outline our general product direction. It is intended for information purposes only, and may not be incorporated into any contract. It is not a commitment to deliver any material, code, or functionality, and should not be relied upon in making purchasing decisions. The development, release, and timing of any features or functionality described for Oracle’s products remains at the sole discretion of Oracle.

Demo

The Speed(s) of Ruby

Multi, Multi, Multi, Multi, Multi VM

- Multi-language
- Multi-threaded
- Multi-tenant
- Multi-node
- Multi-tooling

Multi Language

- Single launcher for all

```
tenant = PolyglotEngine.buildNew().config(...).build();
fourtyTwo = tenant.eval(“text/x-javascript”, “32 + 10”);
twoAndFourty = tenant.eval(“application/x-ruby”, “10 + 32”);
```

- Languages can export and import symbols

```
window = tenant.findGlobalSymbol(“window”);
// and JavaScript provides it by overriding
@Override protected Object findGlobalSymbol(String name) {
 return “window”.equals(name) ? jsWindowGlobal : null;
}
```

Multi Tenant

- Create as many tenants as needed

```
tenant1 = PolyglotEngine.buildNew().config(...).build();
tenant2 = PolyglotEngine.buildNew().config(...).build();
fourtyTwo = tenant1.eval(“text/x-javascript”, “x = this.x ? x + 1 : 42”);
// sees previous value in x:
fourtyThree = tenant1.eval(“text/x-javascript”, “x = this.x ? x + 1 : 42”);
// no x defined, as this is different tenant:
twoAndFourty = tenant2.eval(“text/x-javascript”, “x = this.x ? x + 1 : 42”);
```

- Isolation enforced by TCK

Multi Threaded

- **PolyglotEngine** is inherently single-threaded API
  - Access from a single (creator) thread enforced
- Easy to dispatch execution to different thread

```
tenantAsync = PolyglotEngine.buildNew()
.executor(Executors.newSingleThreadedExecutor())
.build();
```

- Tenants can exchange messages
  - Orchestrate multiple tenants running in parallel from a single thread

Multi Node *(under consideration)*

- **PolyglotEngine** as a facade for remote object protocol

```
tenantAway = PolyglotEngine.buildNew()
.nodeIP(“192.168.34.65”) // the execution will happen on different node
.build();
result = tenantAway.eval(“text/javascript”, “'Remote and slow ' + 42”);
```

- Keeping the same API facade: **PolyglotEngine**
  - Get multi language, multi threaded, multi tenant
  - Multi tooling and multi node virtual machine!

Multi Tooling

- JPDA debugging always on
  - Attach Java IDE and all Truffle language details will be shown
- Turn on your debugger protocol

```
tenantWithDebugger = PolyglotEngine.buildNew()
.onEvent(new ExecutionEventHandler() {…})
.onEvent(new SuspendedEventHandler() {…})
.build();
```

- Other instrumentation based tools available as well

Demo

Polyglot Debugging

The Fastest (J)VM on the Planet

- Try it now!
  - Download jvm: http://www.oracle.com/technetwork/oracle-labs/program-languages/
  - Open source:
    - http://openjdk.java.net/projects/graal/
    - https://github.com/graalvm
  - JRuby, Python, JavaScript, R, C
- Speed is Great!
  - In need of completeness
- We are hiring!

Hardware and Software
Engineered to Work Together