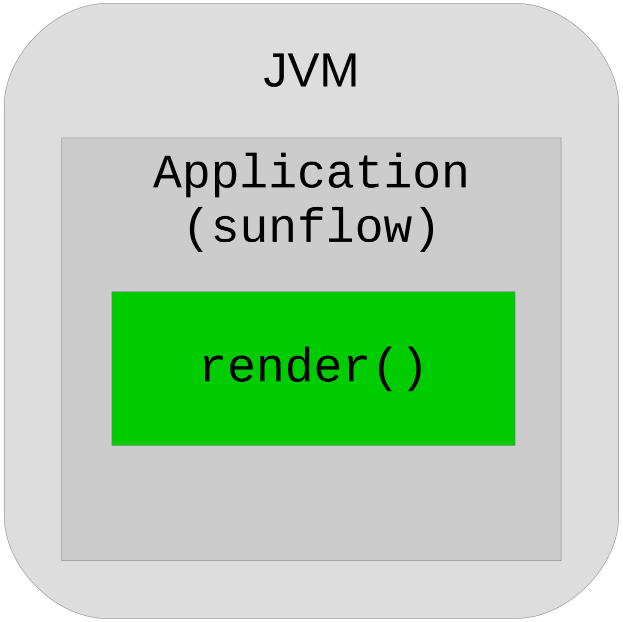
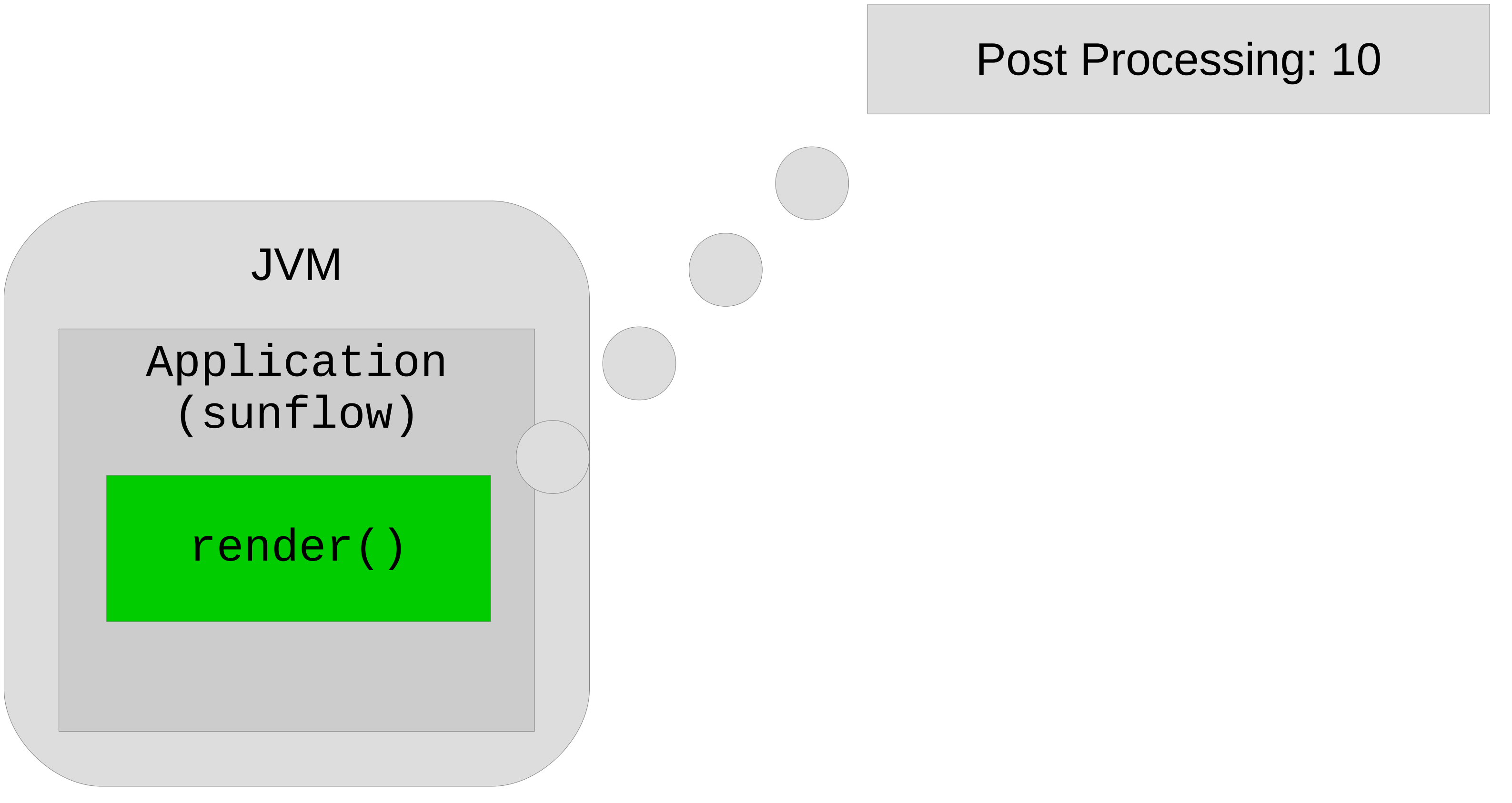
Application-Level Energy Accounting with Chappie

Timur Babakol, **Anthony Canino,** Khaled Mahmoud, Rachit Saxena, and Yu David Liu
Binghamton University,
State University of New York (SUNY),
{tabako1,acanino1,kmahou1,rsaxena3,davidl}@binghamton.edu

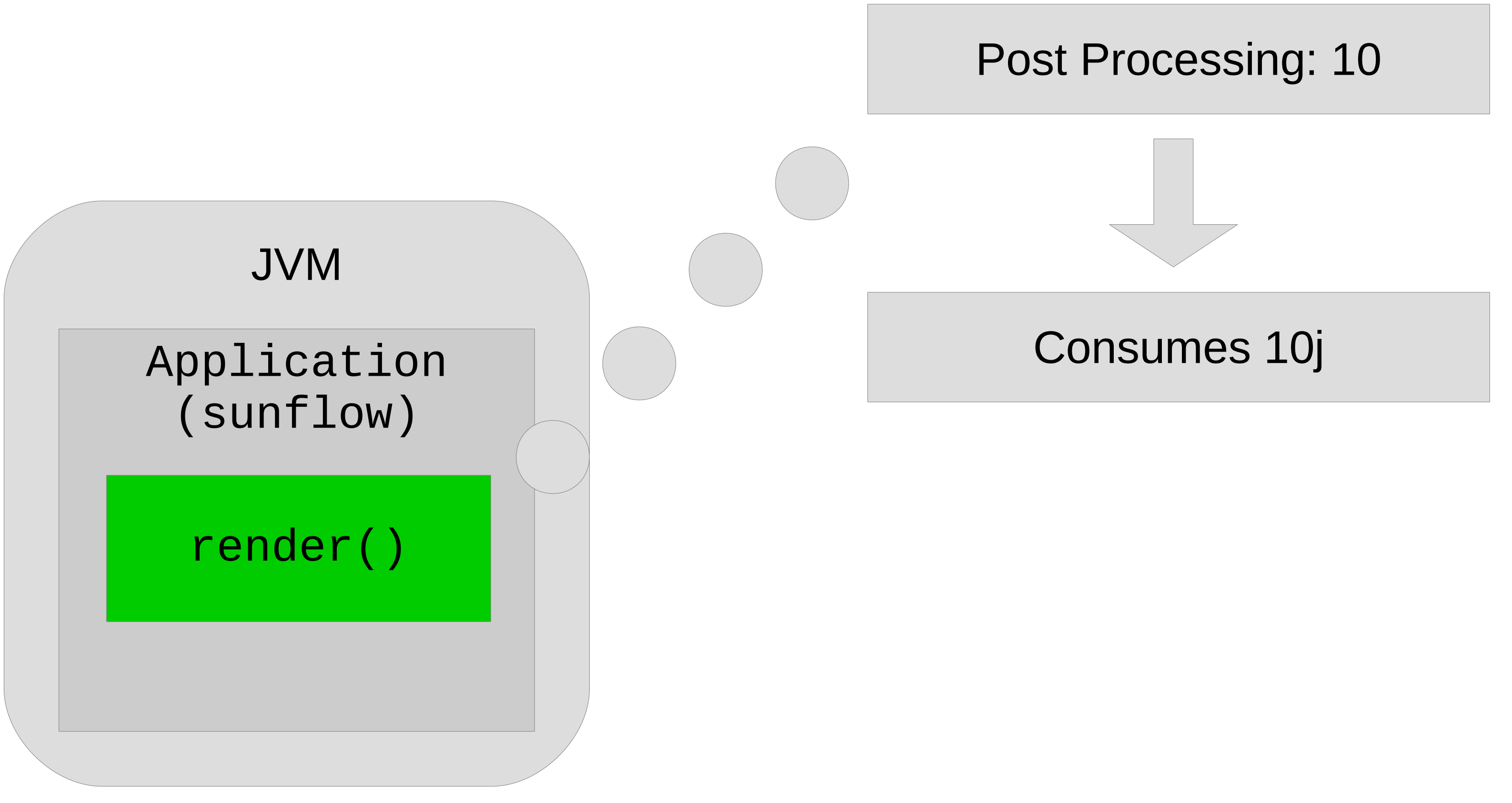
Application-Level Energy Management



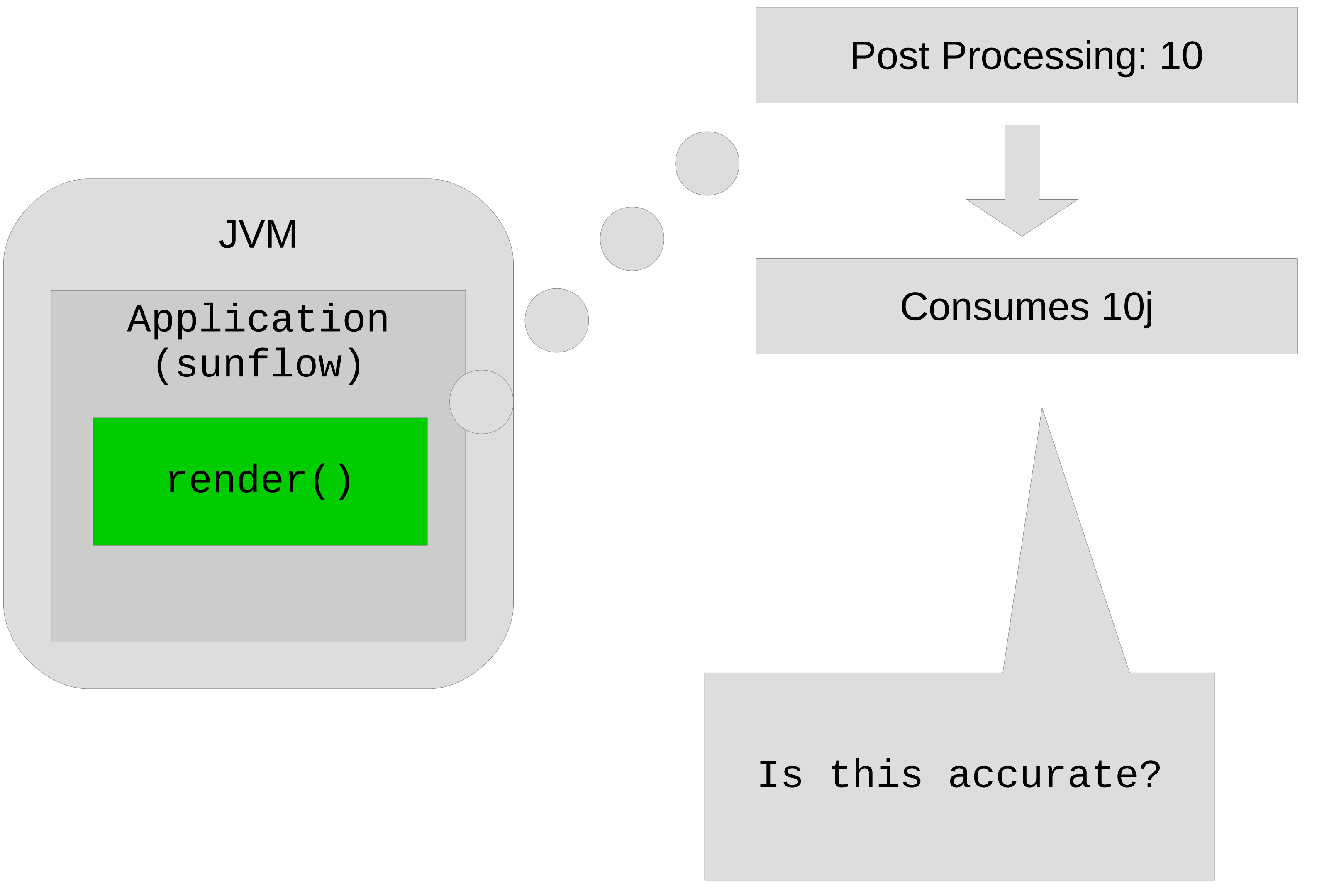
Application-Level Energy Management



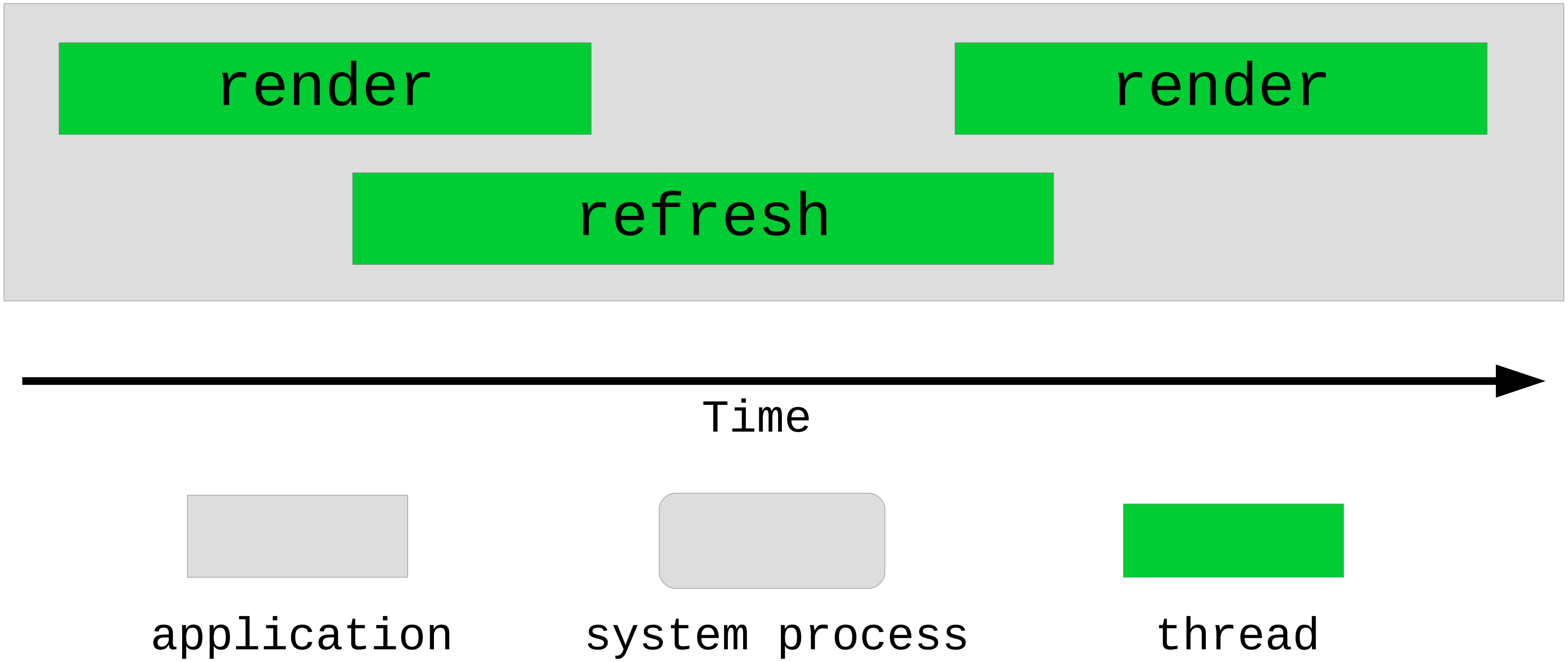
Application-Level Energy Management



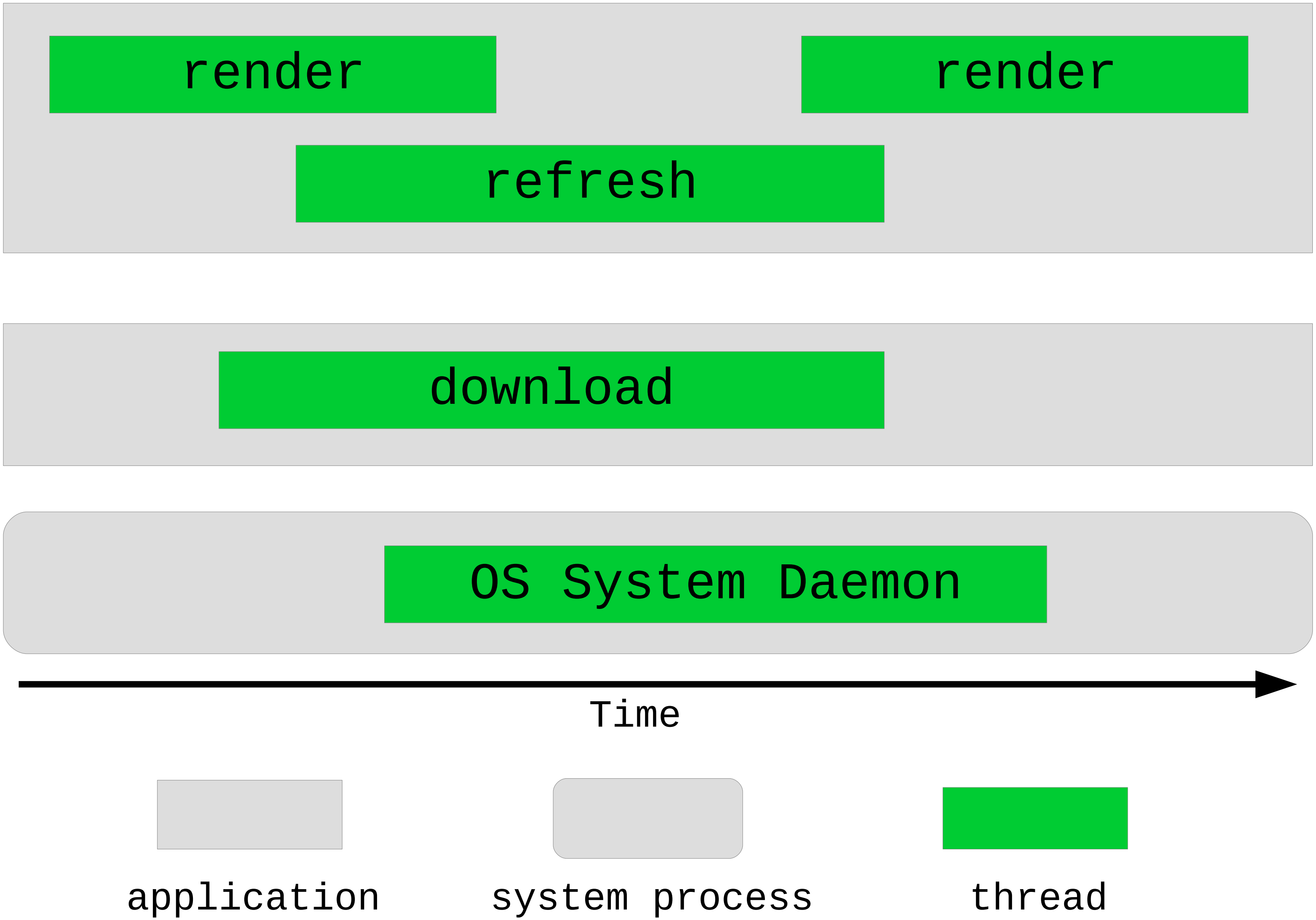
Application-Level Energy Management



Problem #1: Multi-threading



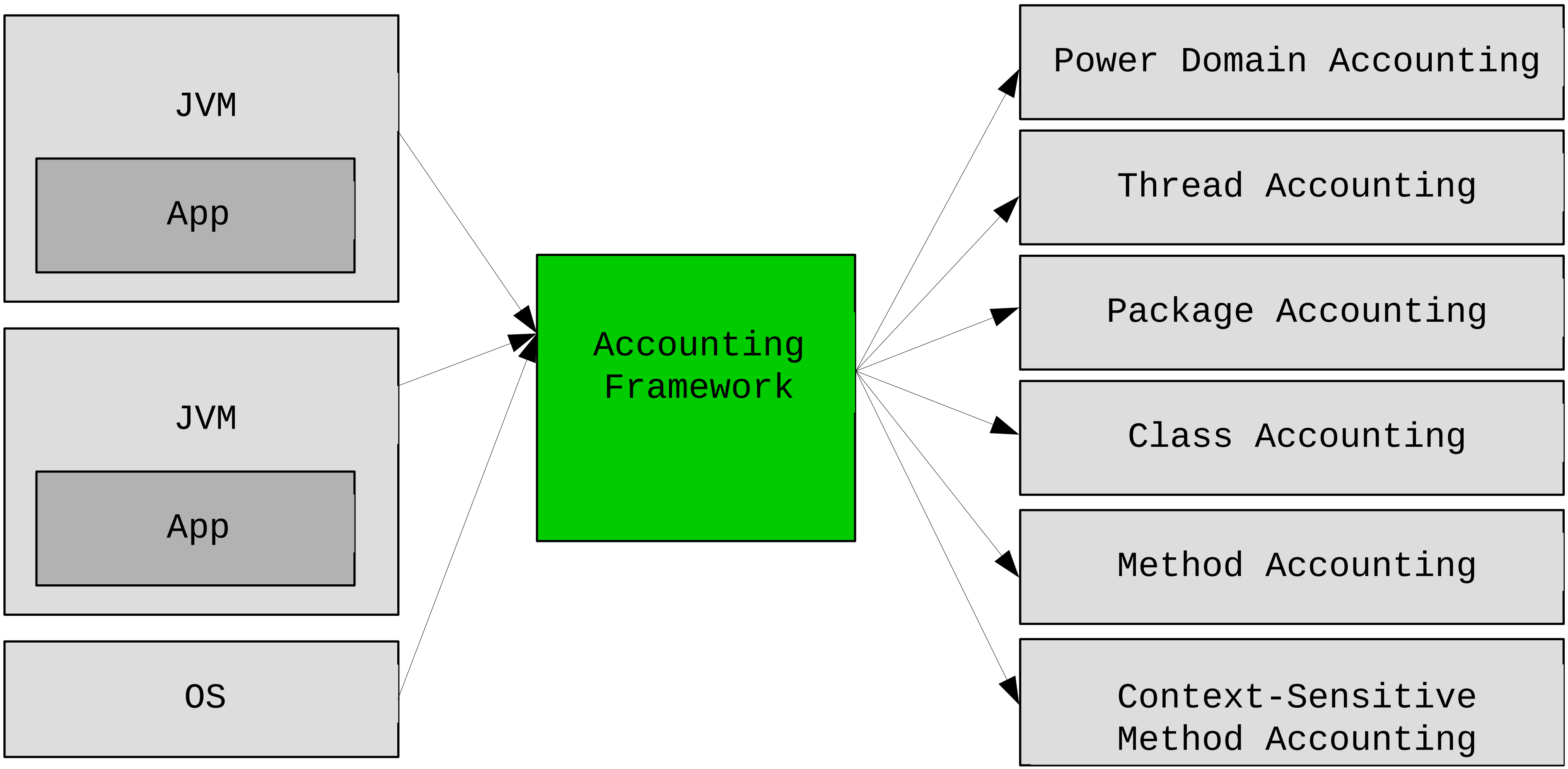
Problem #2: Co-Running



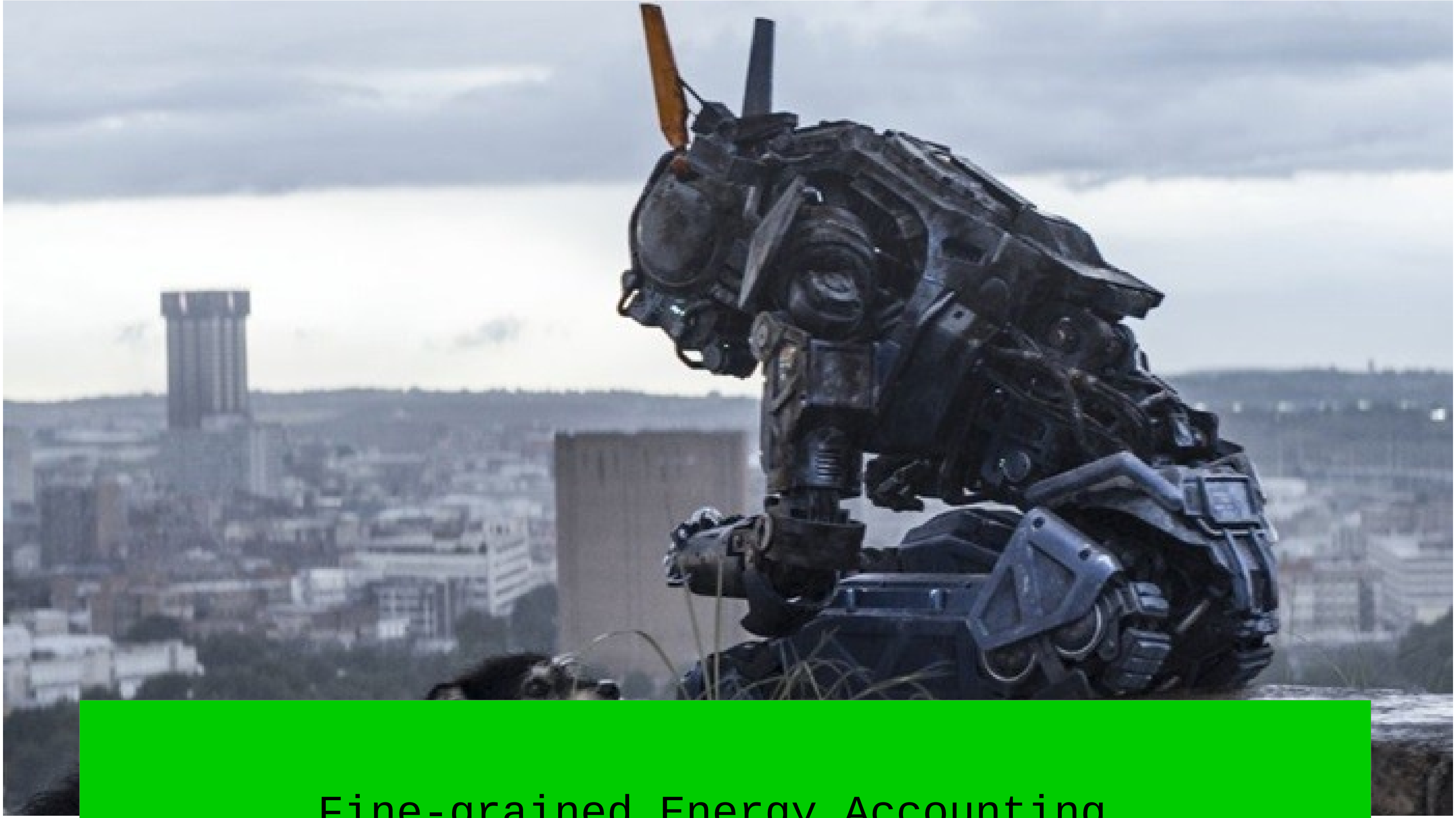
Problem #3: Overhead

	Instrumentation 100%	Instrumentation 5%
sunflow	189x	164x
batik	118x	117x
avrora	33x	30x
xalan	30x	29x
h2	25x	16x

What is Application-Level Accounting?



Chappie



Fine-grained Energy Accounting
of Java Applications

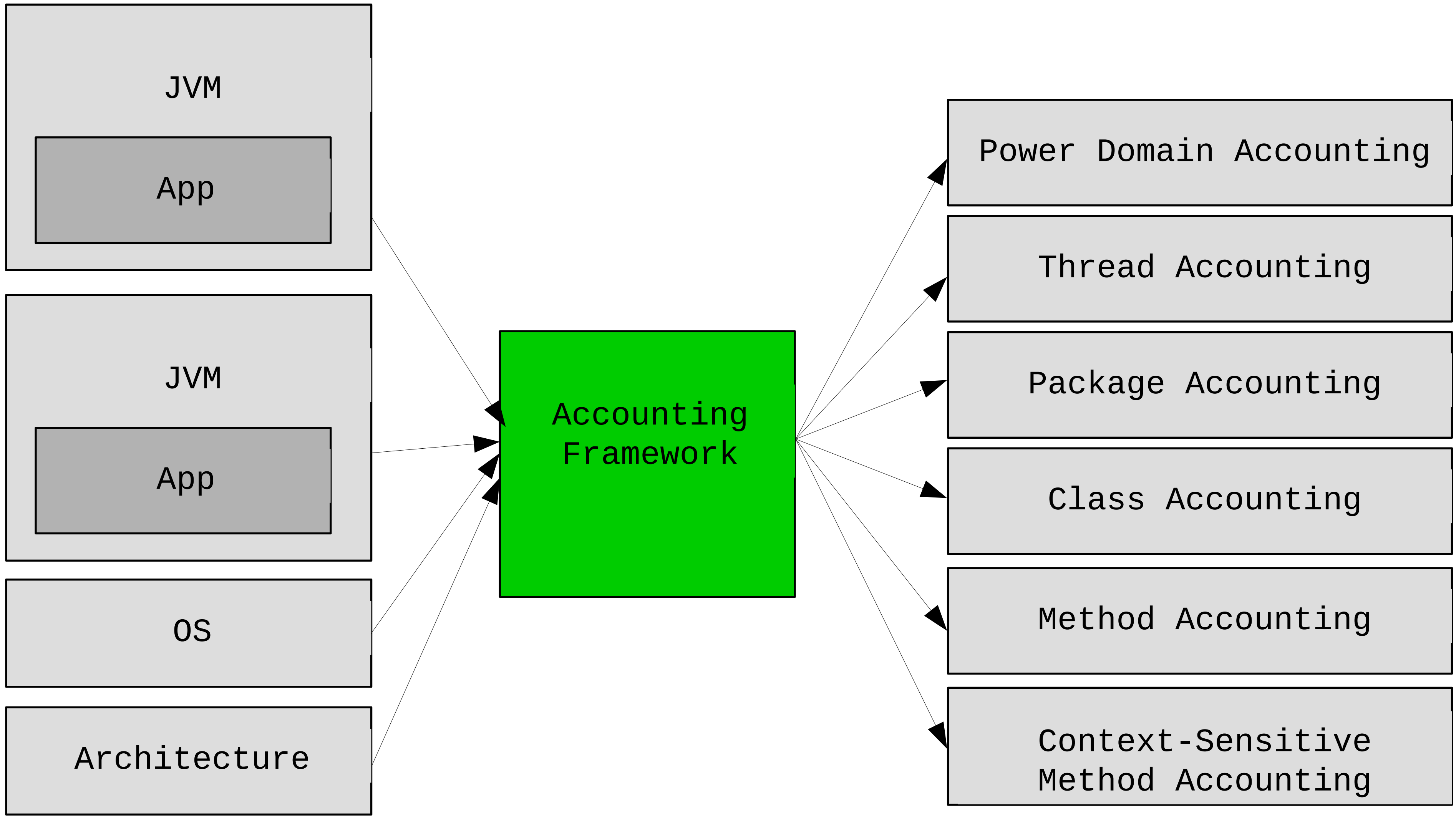
This Talk: **Chappie**

- A cross-layer, concurrency-aware design for fine-grained energy accounting of multi-threaded java applications
- A low-overhead sampling-based implementation
- An evaluation on 20 benchmark applications analyzing per-method, per-thread, and per-application energy accounting

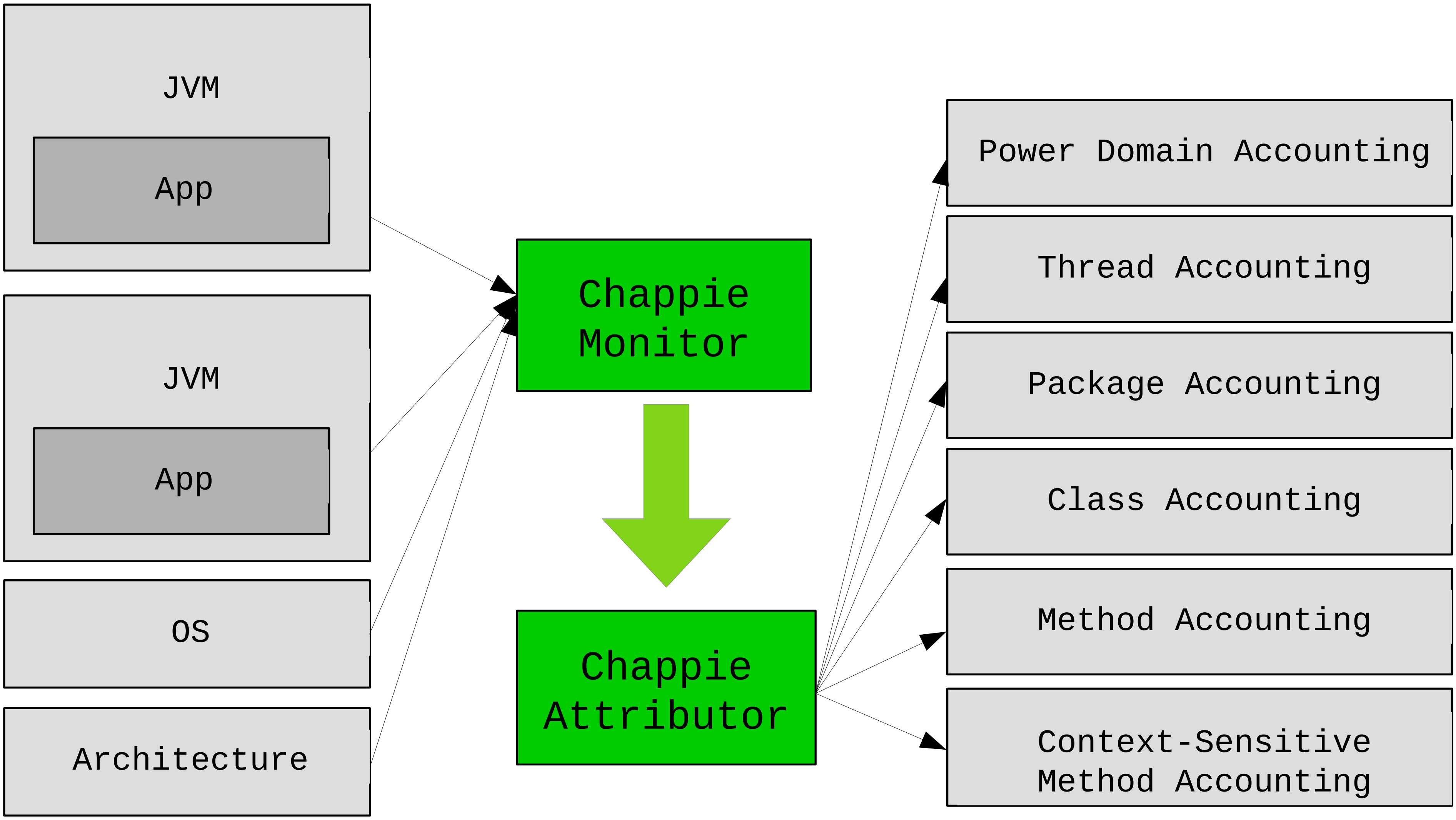
This Talk: **Chappie**

- A cross-layer, concurrency-aware design for fine-grained energy accounting of multi-threaded java applications
- A low-overhead sampling-based implementation
- An evaluation on 20 benchmark applications analyzing per-method, per-thread, and per-application energy accounting

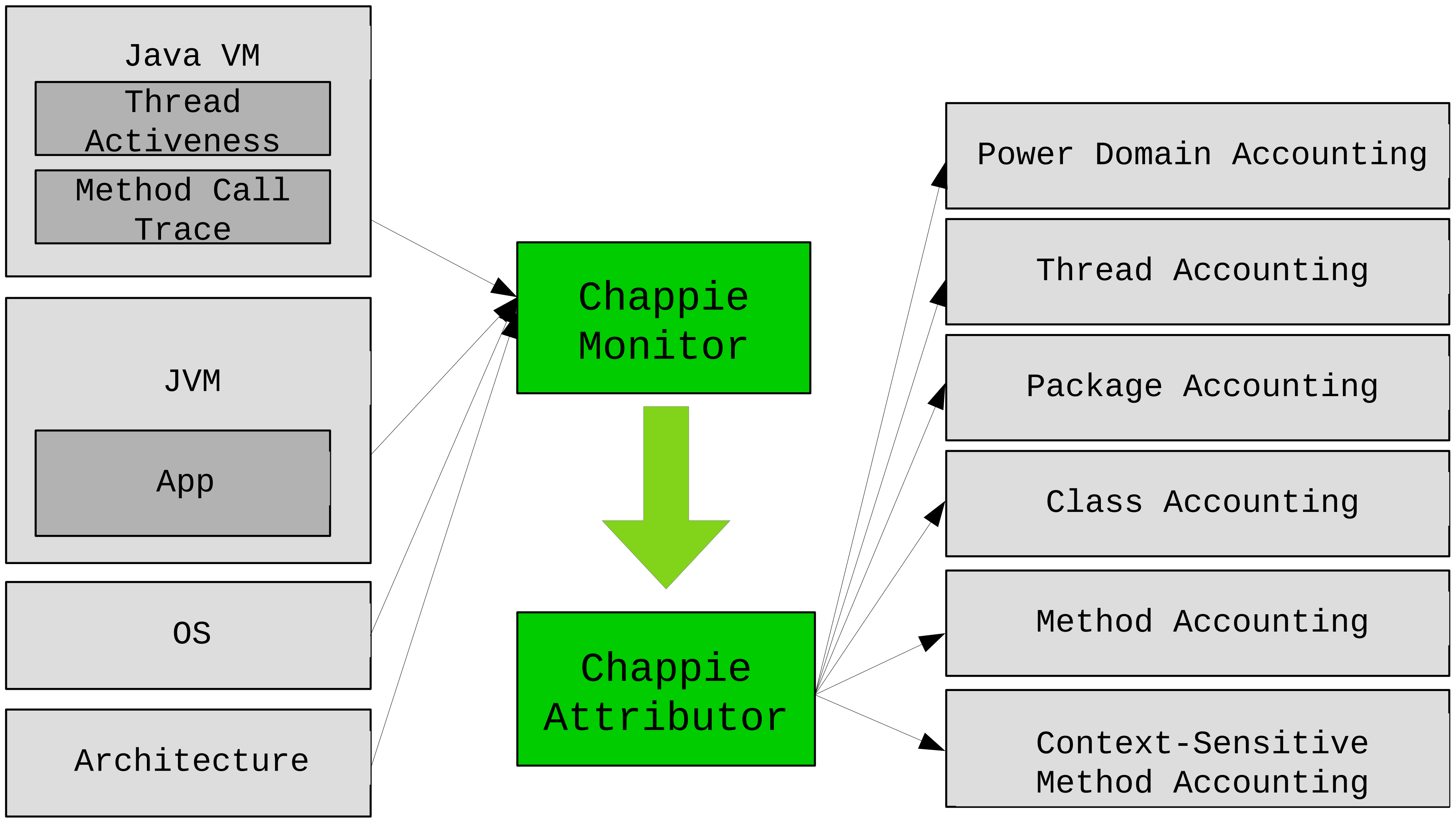
Cross Layer Design



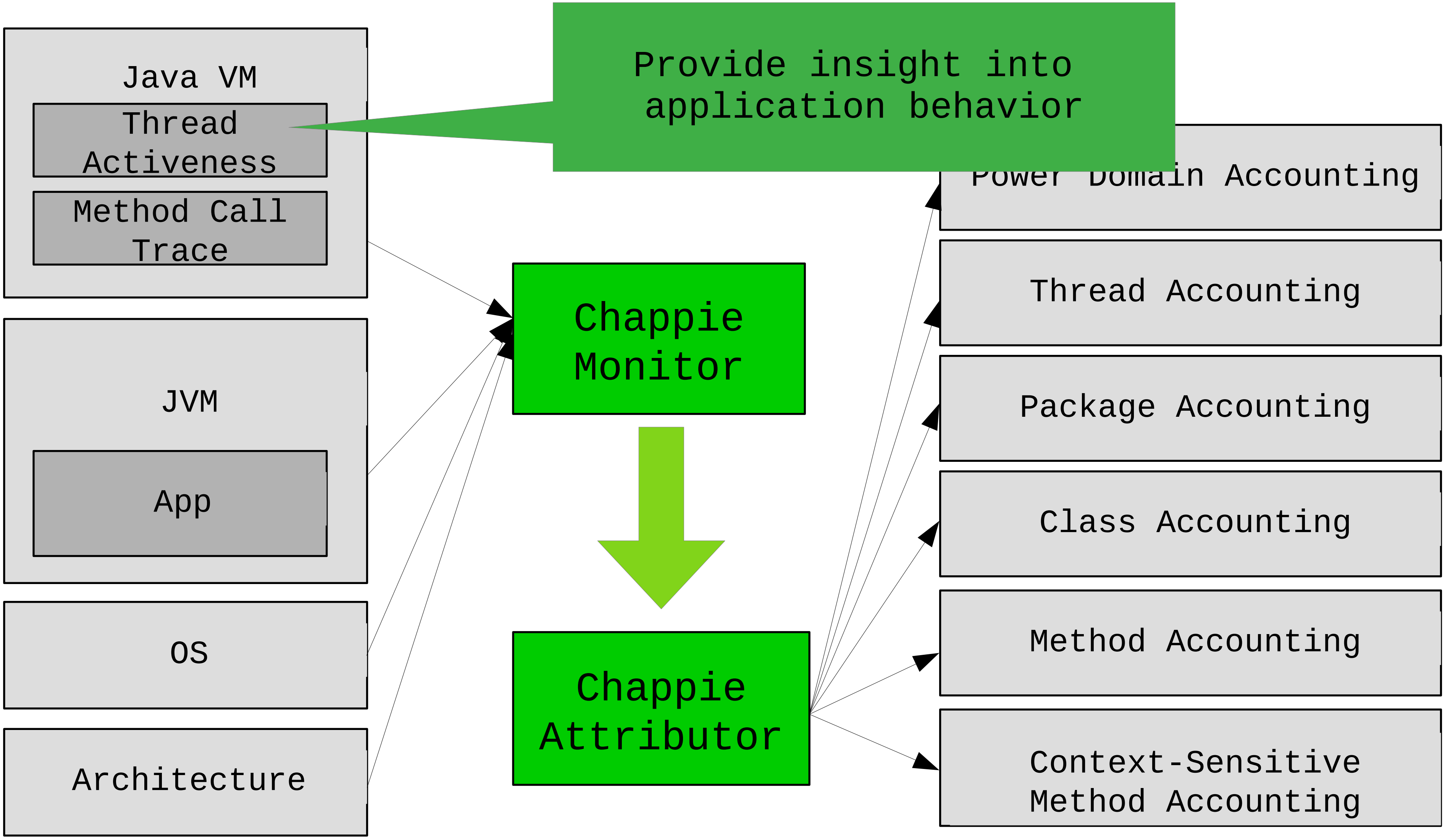
Cross Layer Design



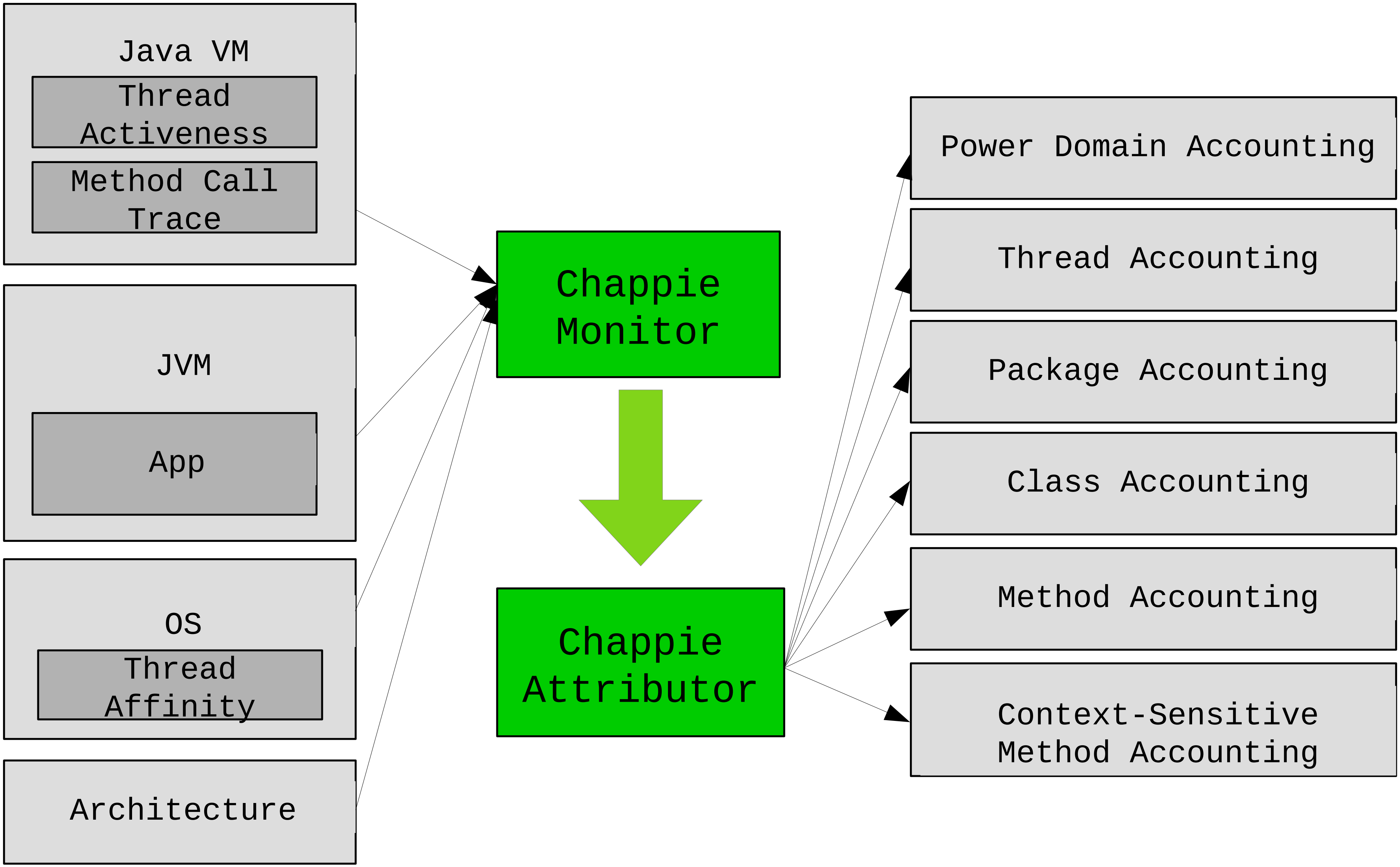
Cross Layer Design



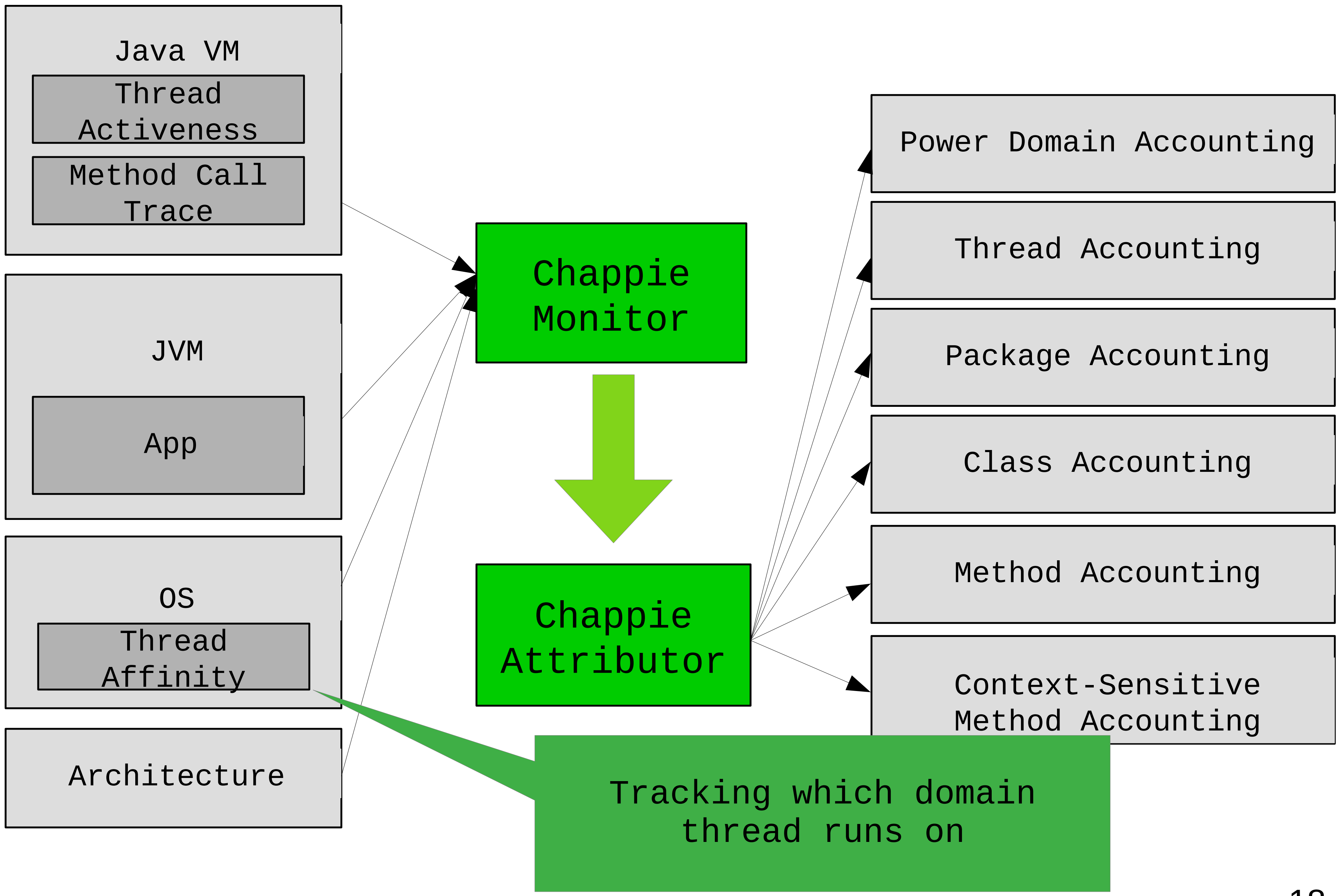
Cross Layer Design



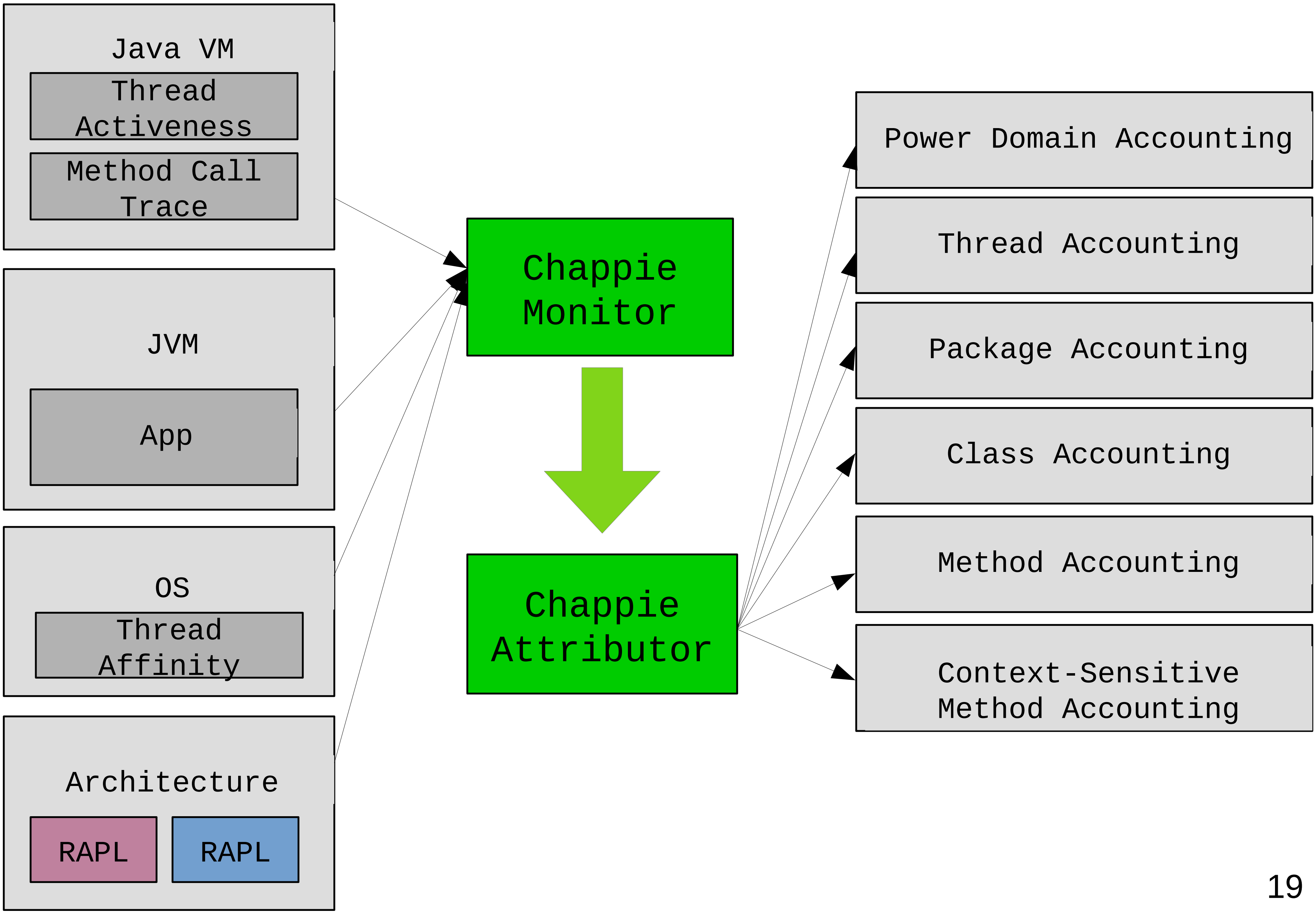
Cross Layer Design



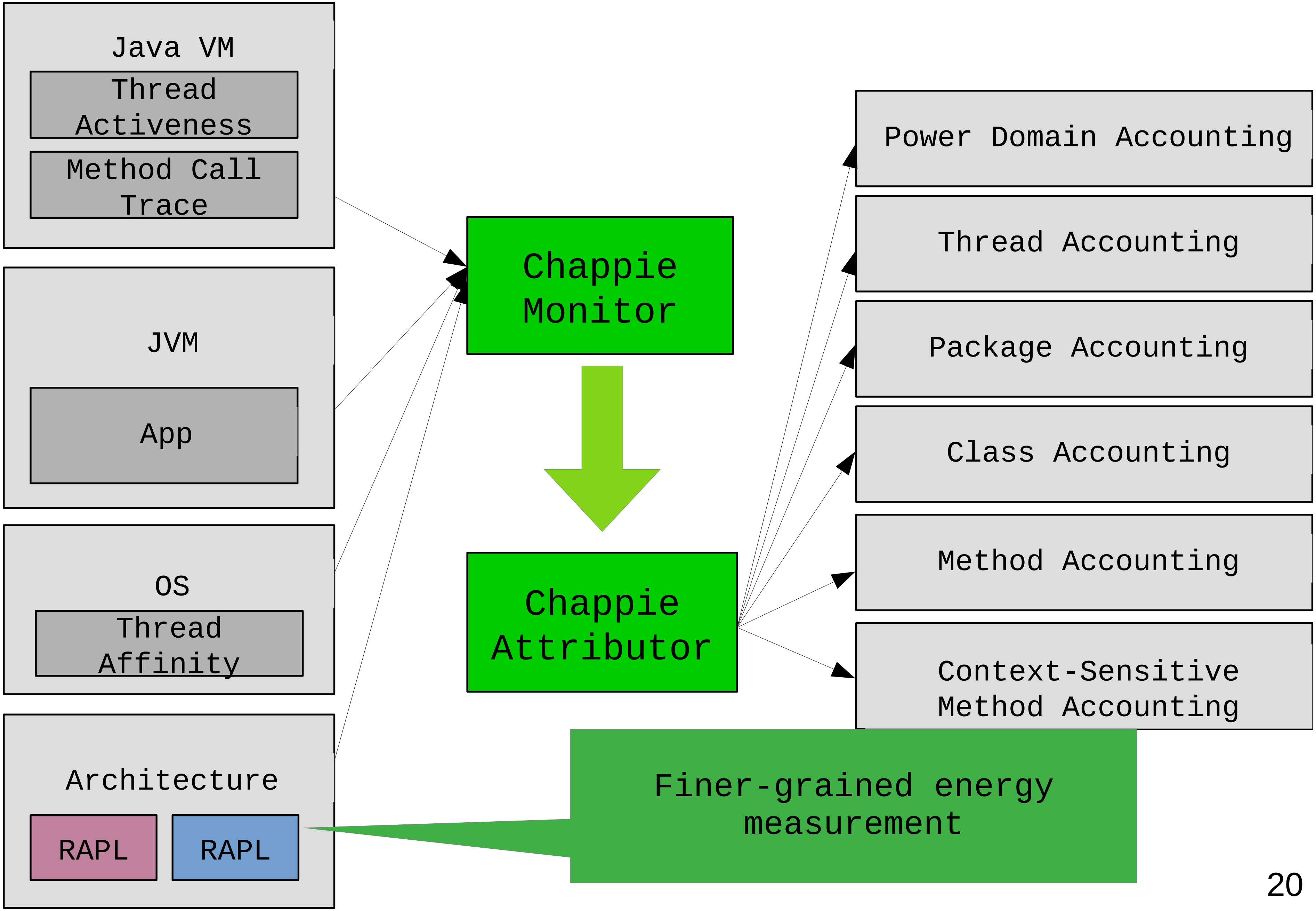
Cross Layer Design



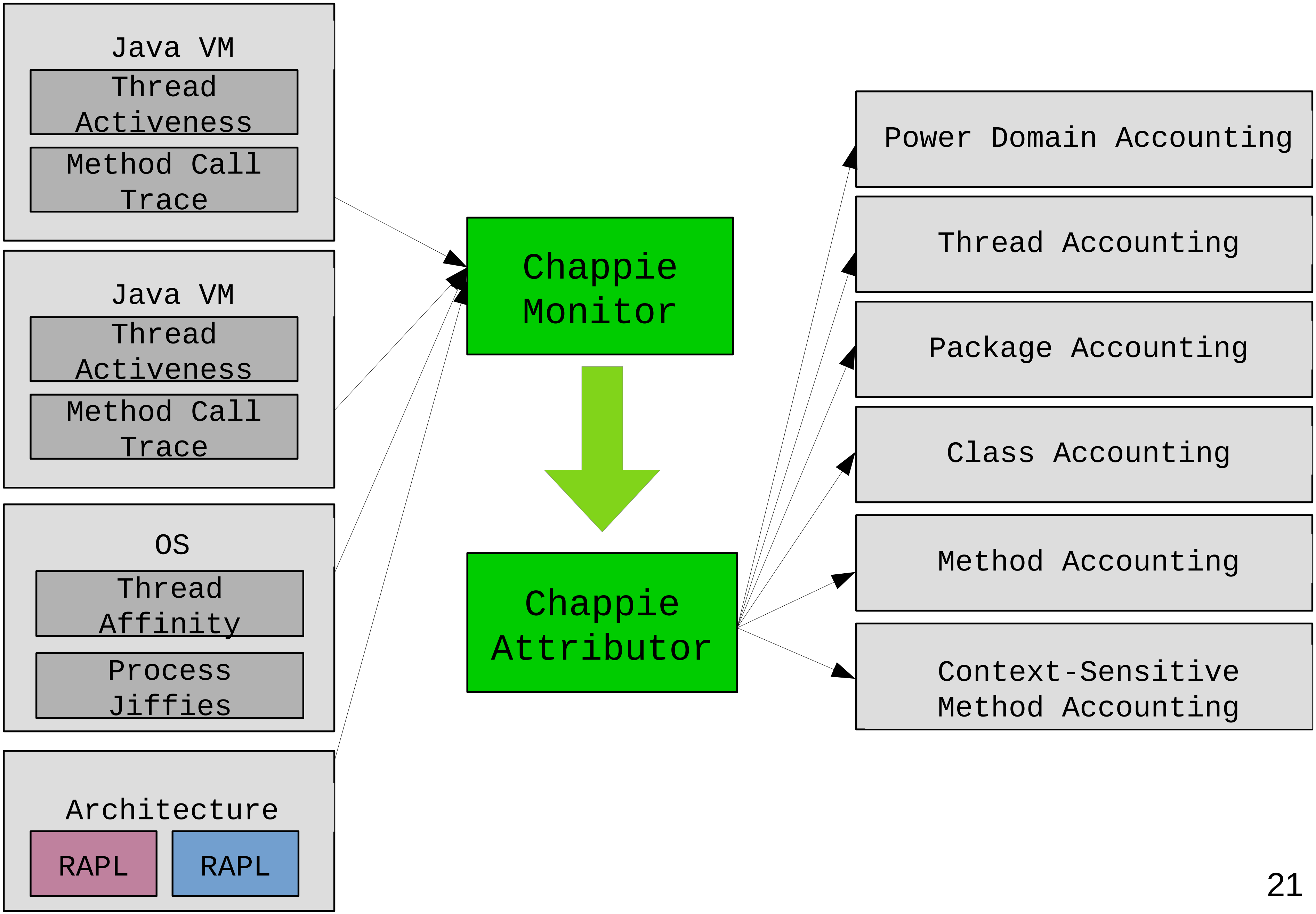
Cross Layer Design



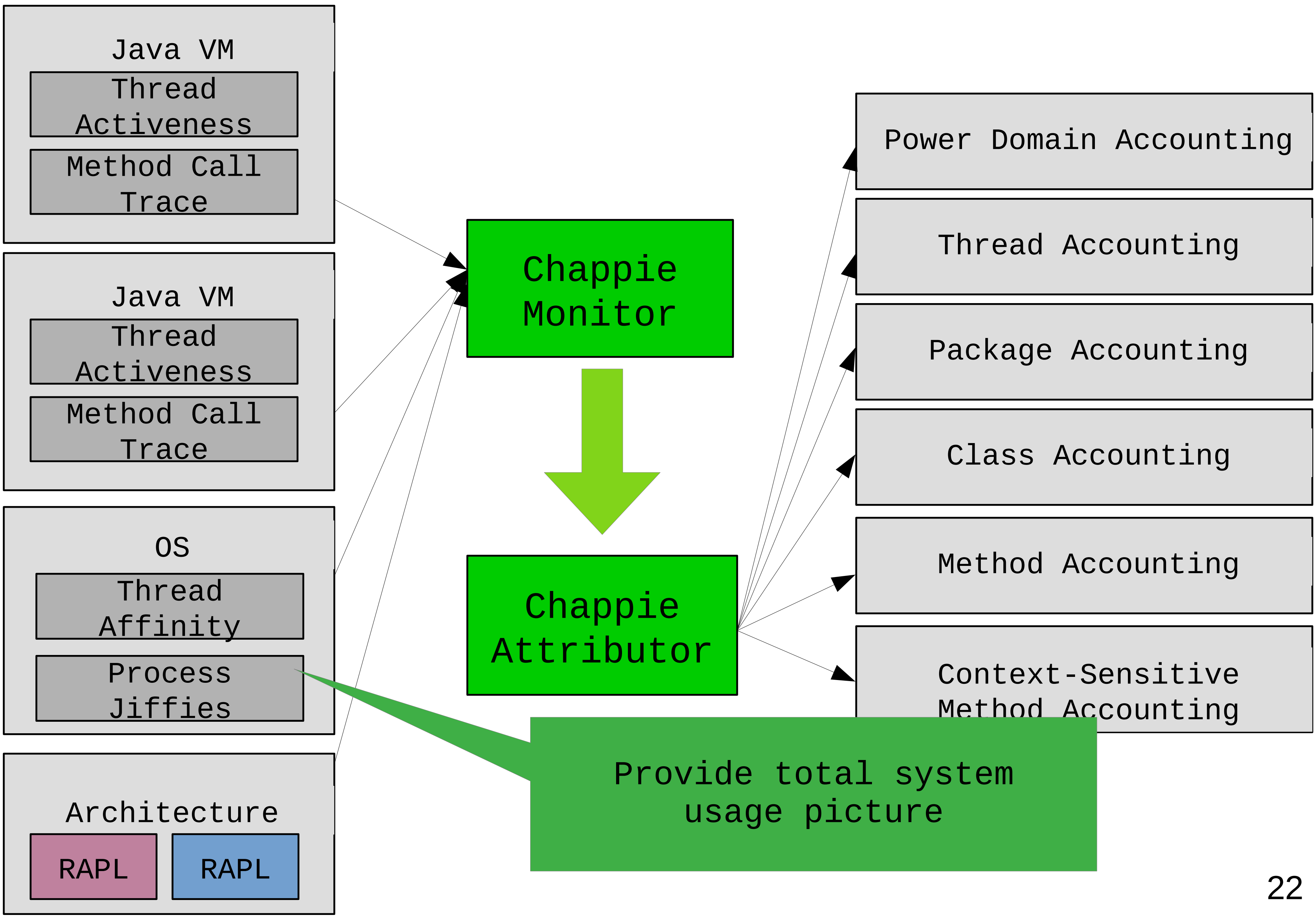
Cross Layer Design



Cross Layer Design



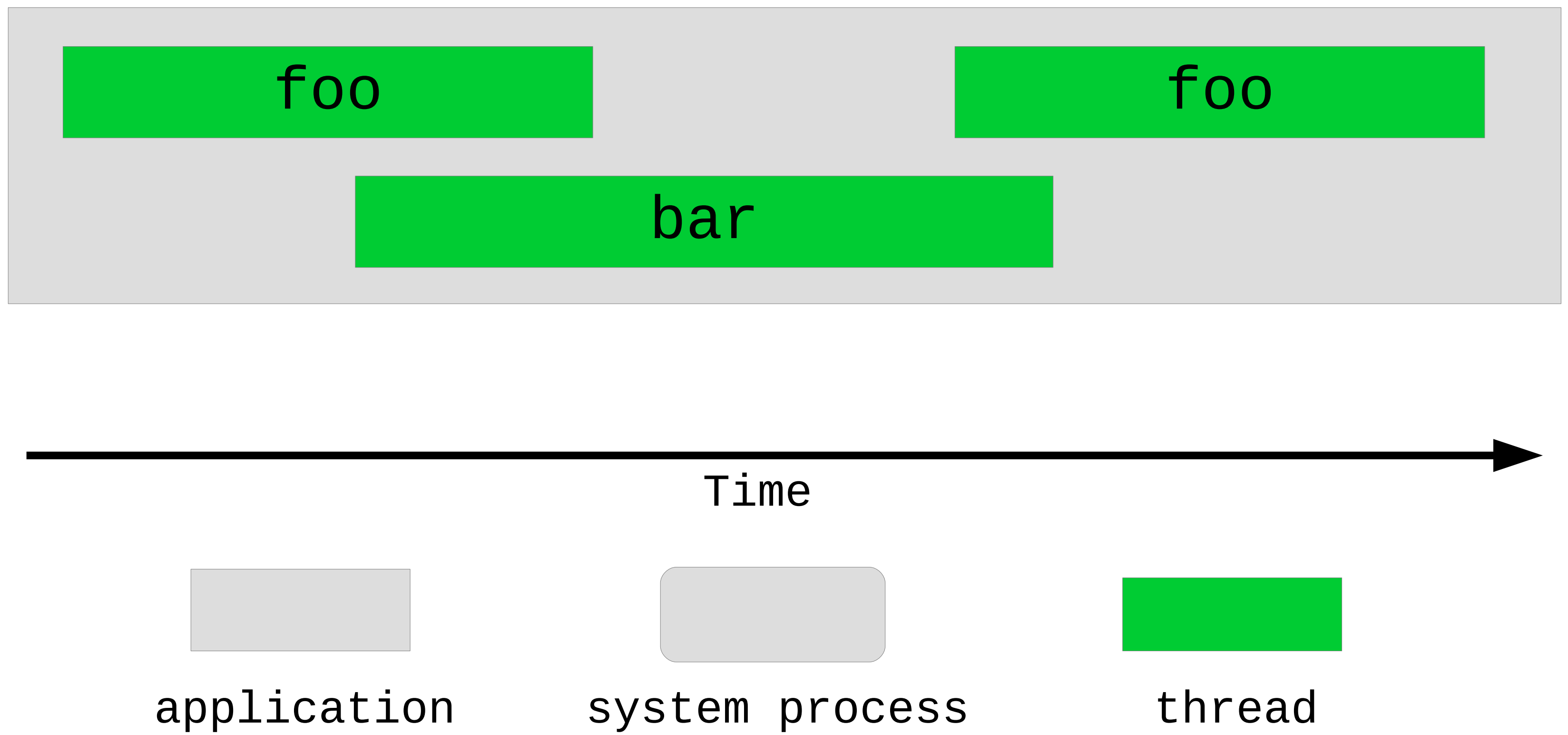
Cross Layer Design



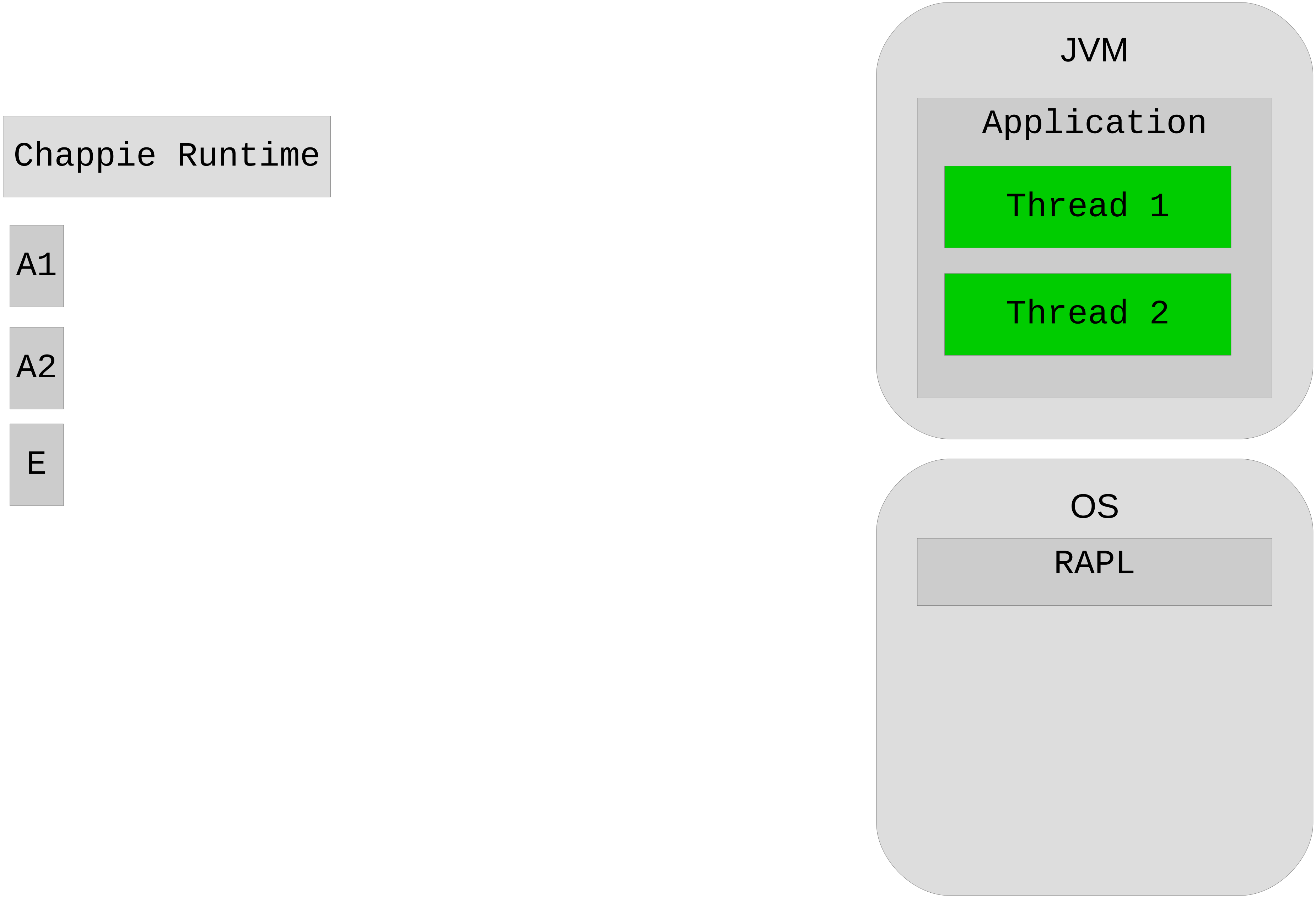
This Talk: **Chappie**

- A cross-layer, concurrency aware design for fine-grained energy accounting of multi-threaded java applications
- A low-overhead sampling-based implementation
- An evaluation on 20 benchmark applications analyzing per-method, per-thread, and per-application energy attribution

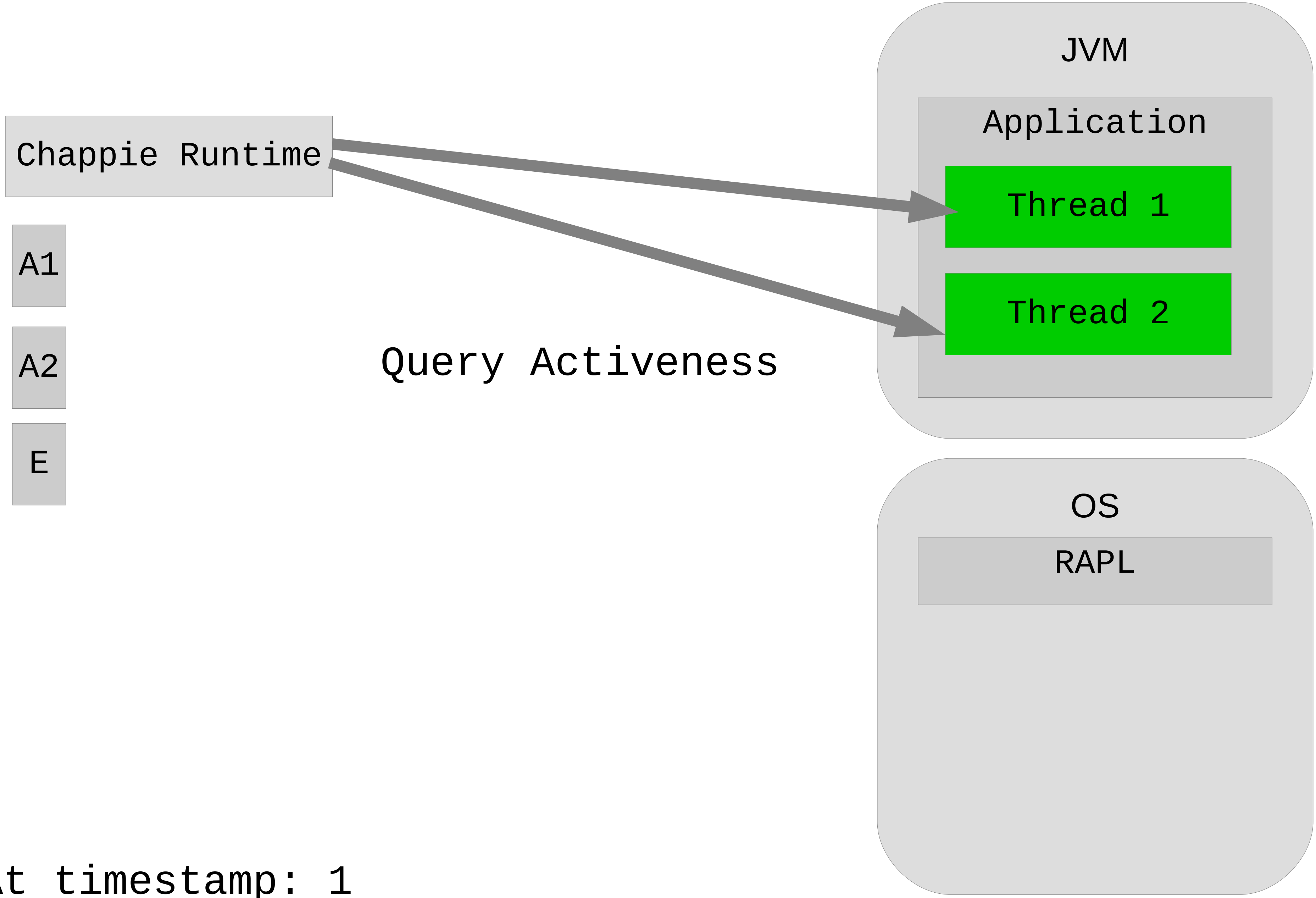
Addressing Problem #1: Multi-threading



Concurrency-Aware Attribution

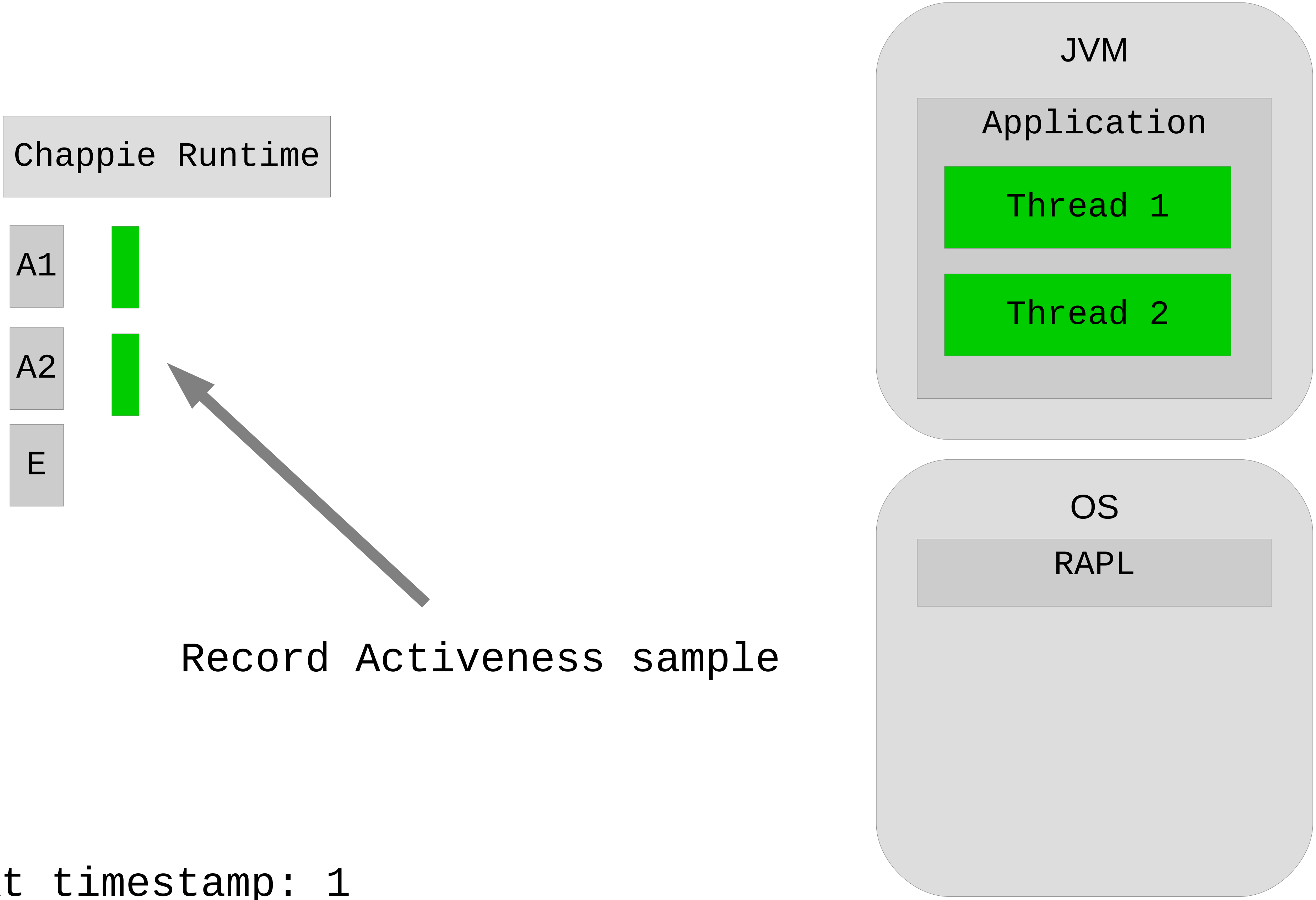


Concurrency-Aware Attribution



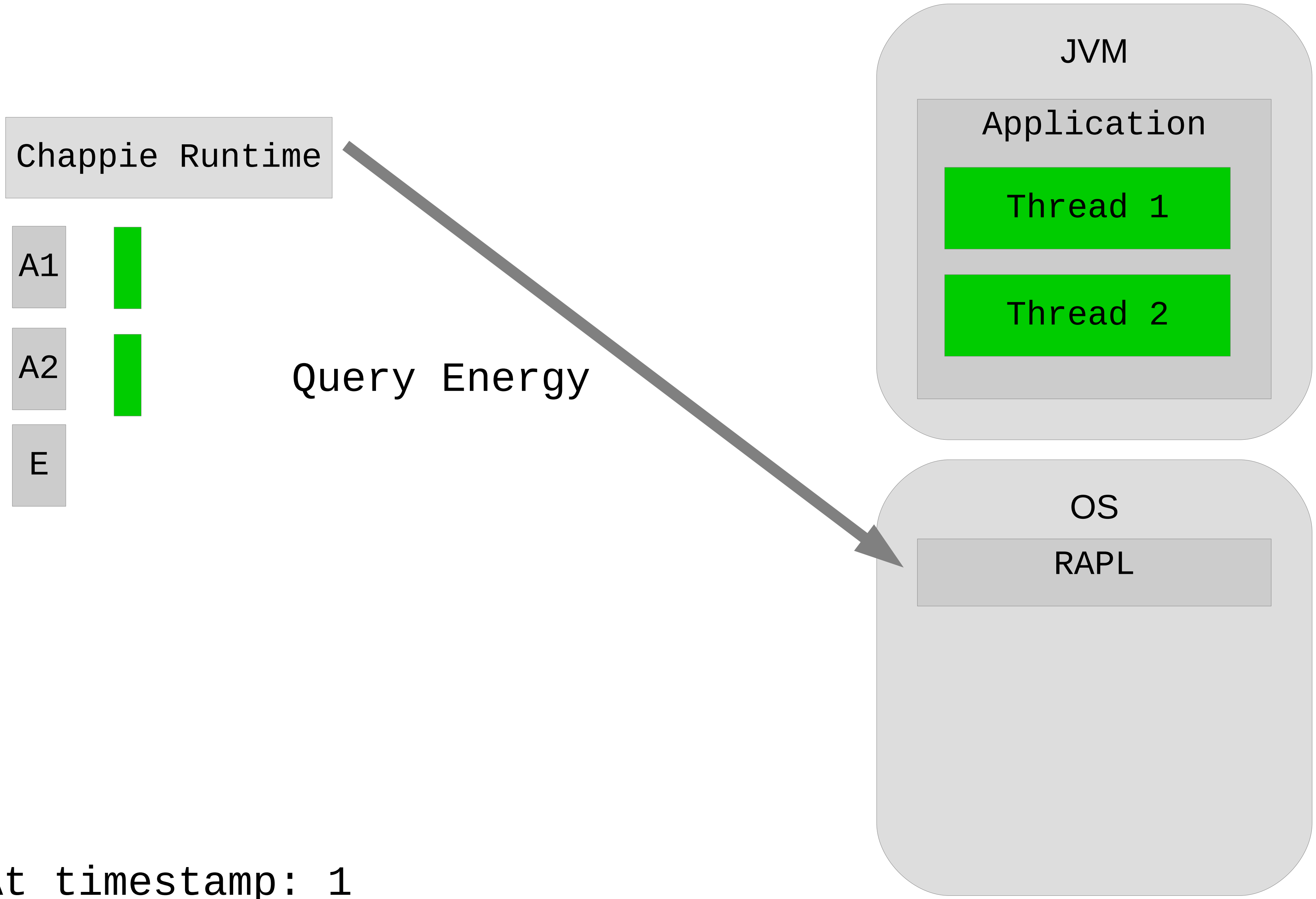
At timestamp: 1

Concurrency-Aware Attribution



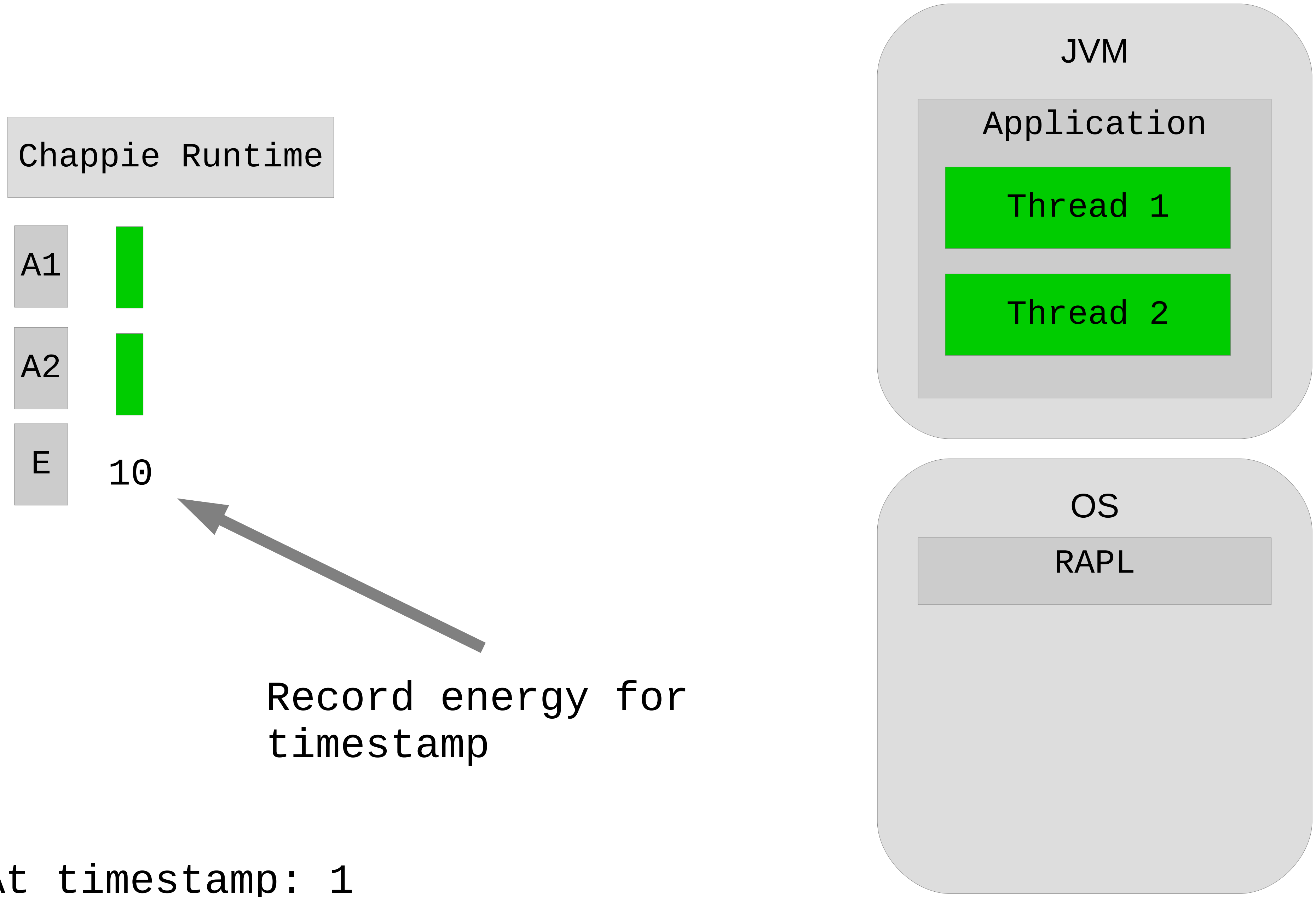
At timestamp: 1

Concurrency-Aware Attribution



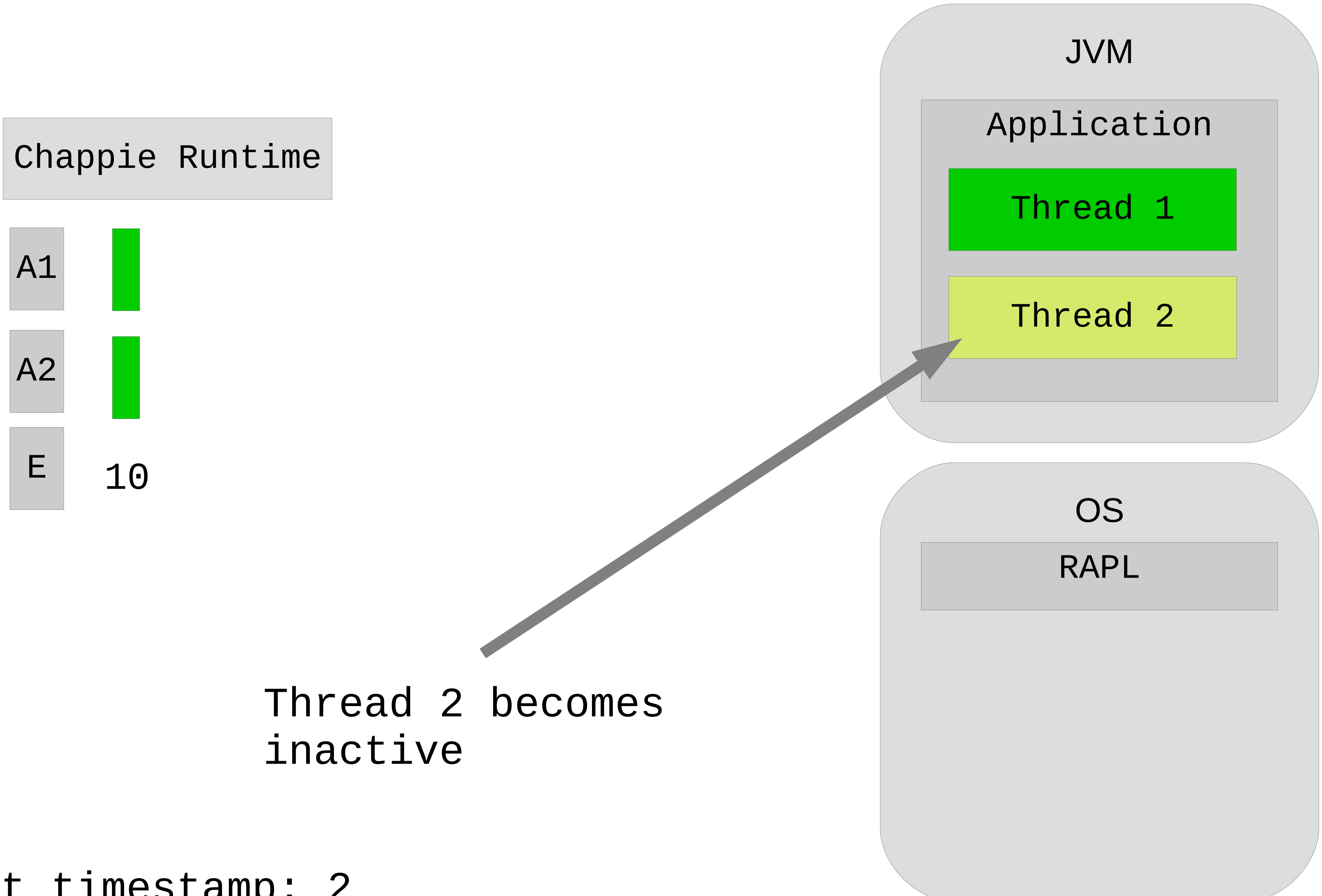
At timestamp: 1

Concurrency-Aware Attribution



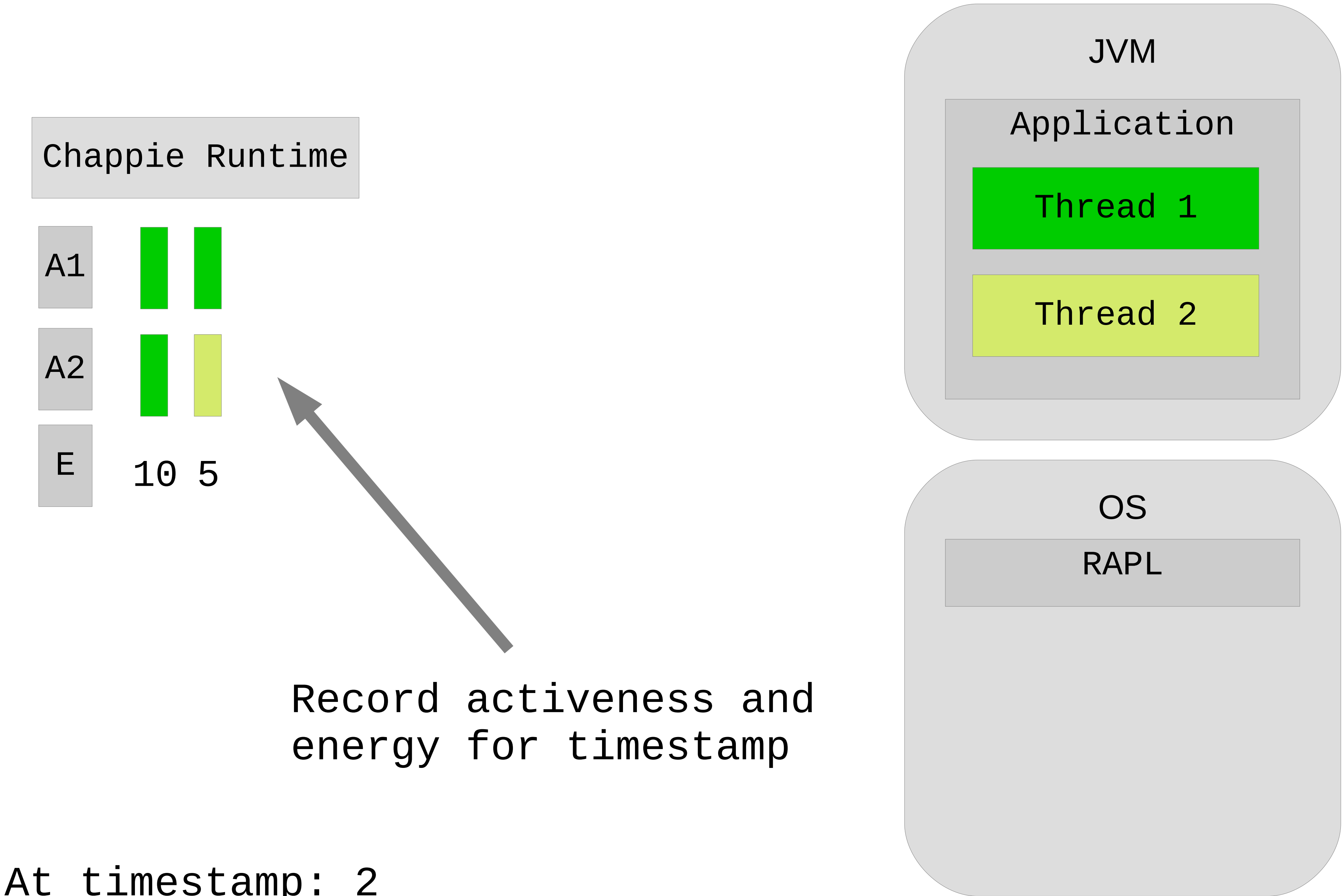
At timestamp: 1

Concurrency-Aware Attribution

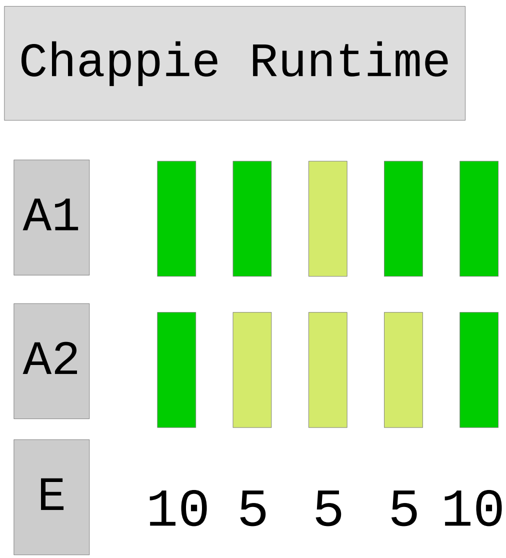


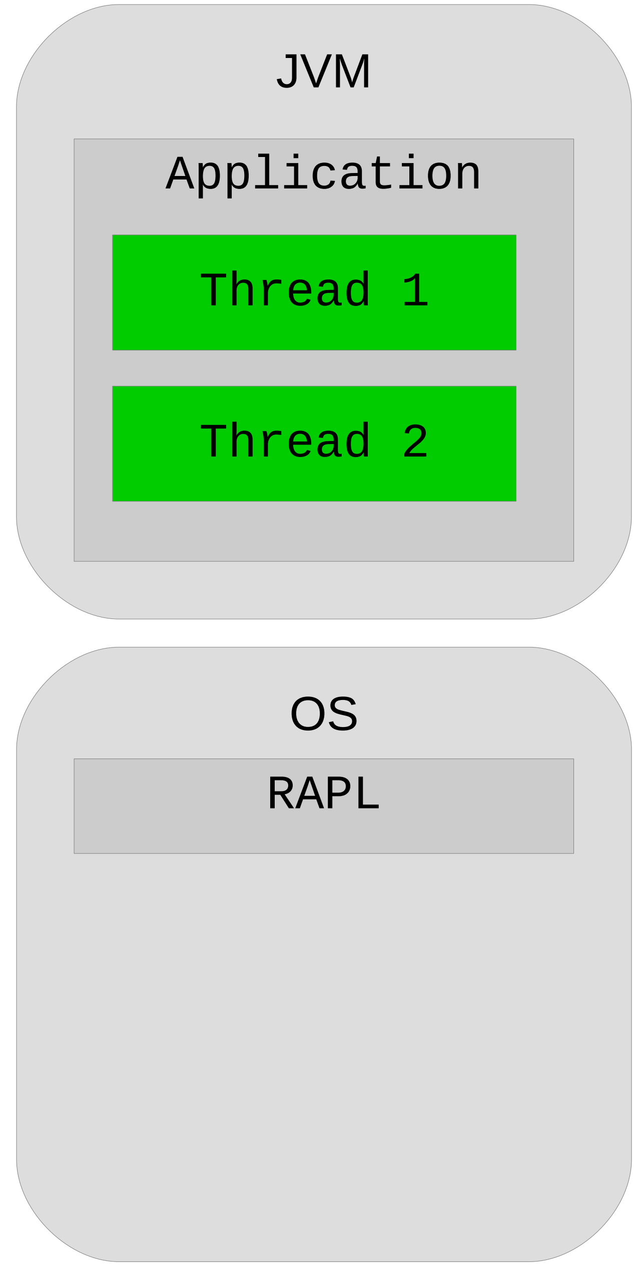
At timestamp: 2

Concurrency-Aware Attribution



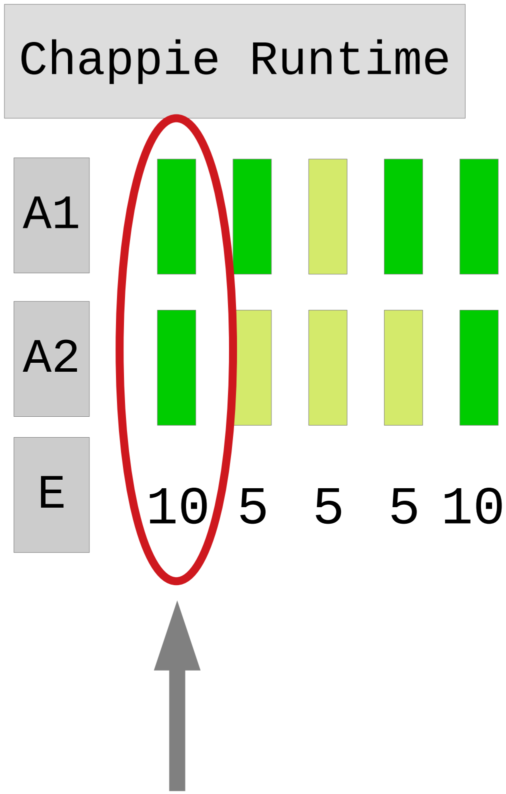
Concurrency-Aware Attribution





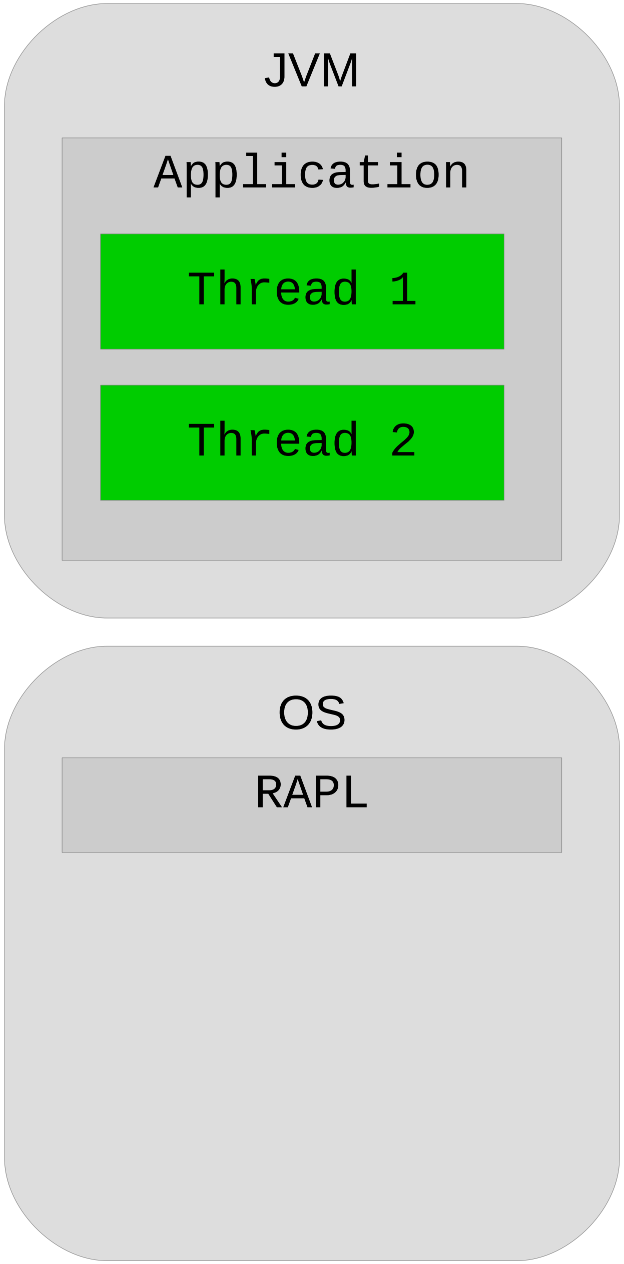
At timestamp: 5

Concurrency-Aware Attribution

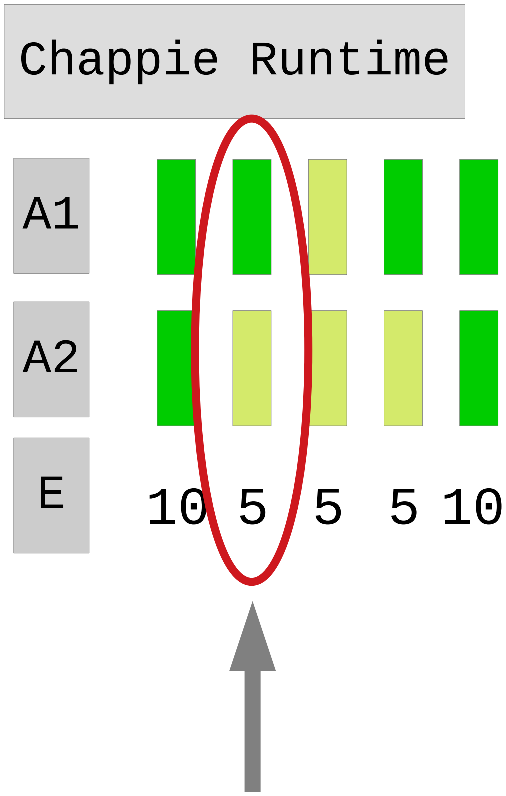


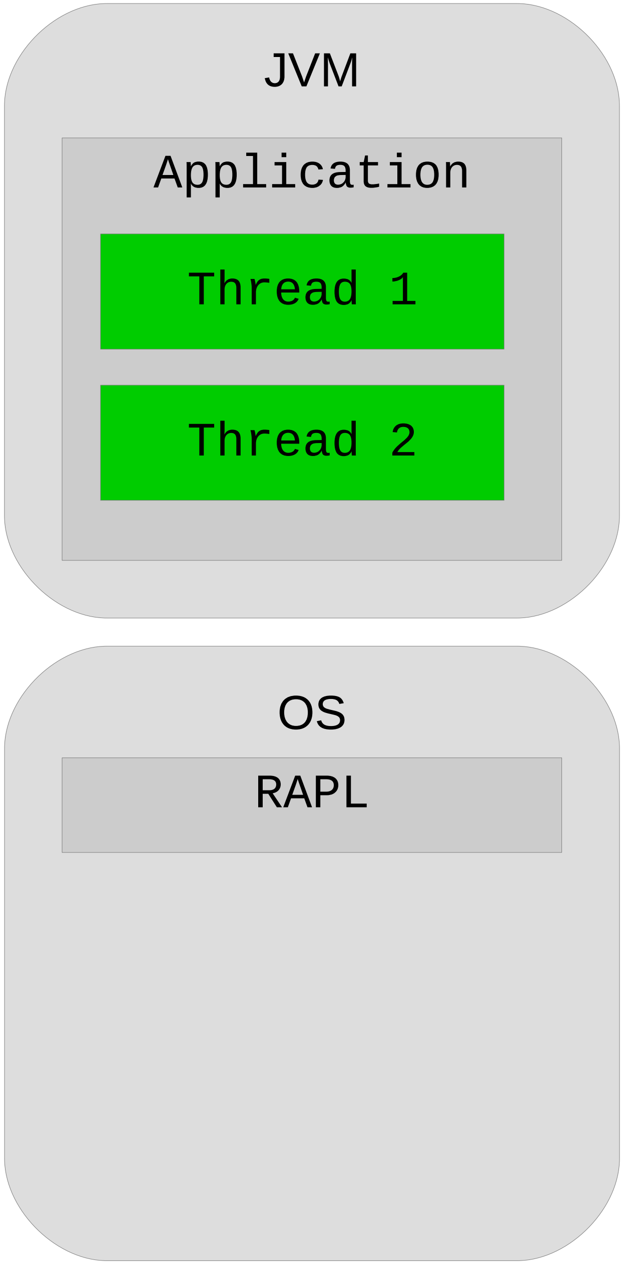
Attribute 5j to each Thread at T1

At timestamp: 5



Concurrency-Aware Attribution

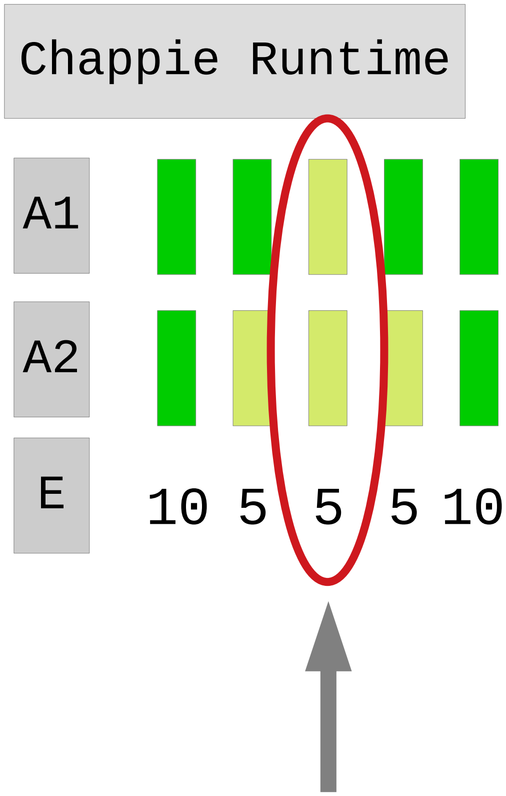




Attribute 5j to each
Thread 1 at T2

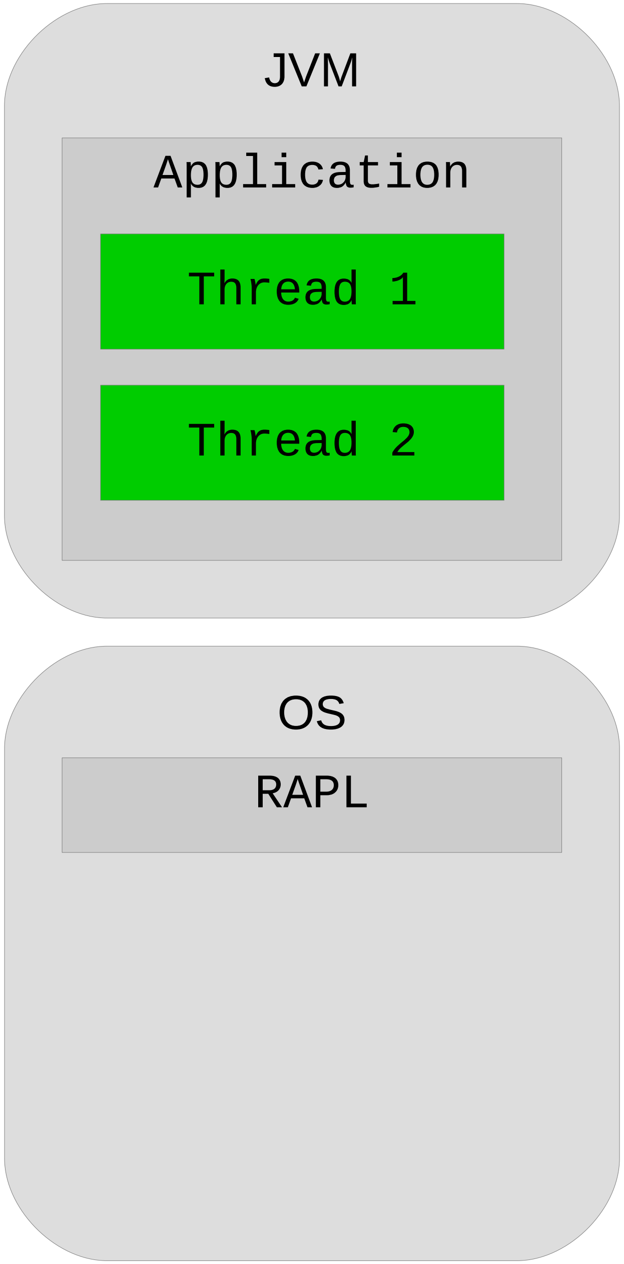
At timestamp: 5

Concurrency-Aware Attribution

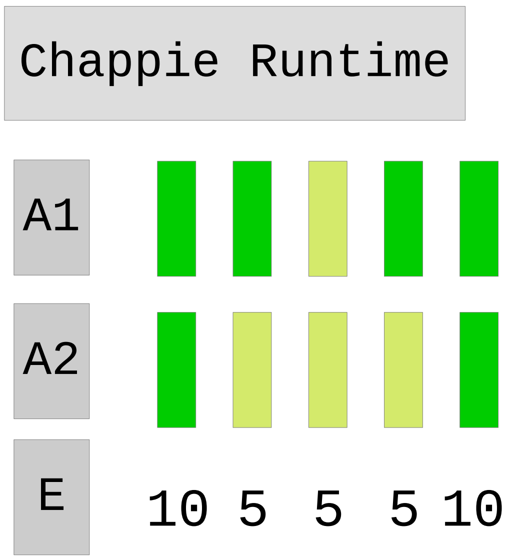


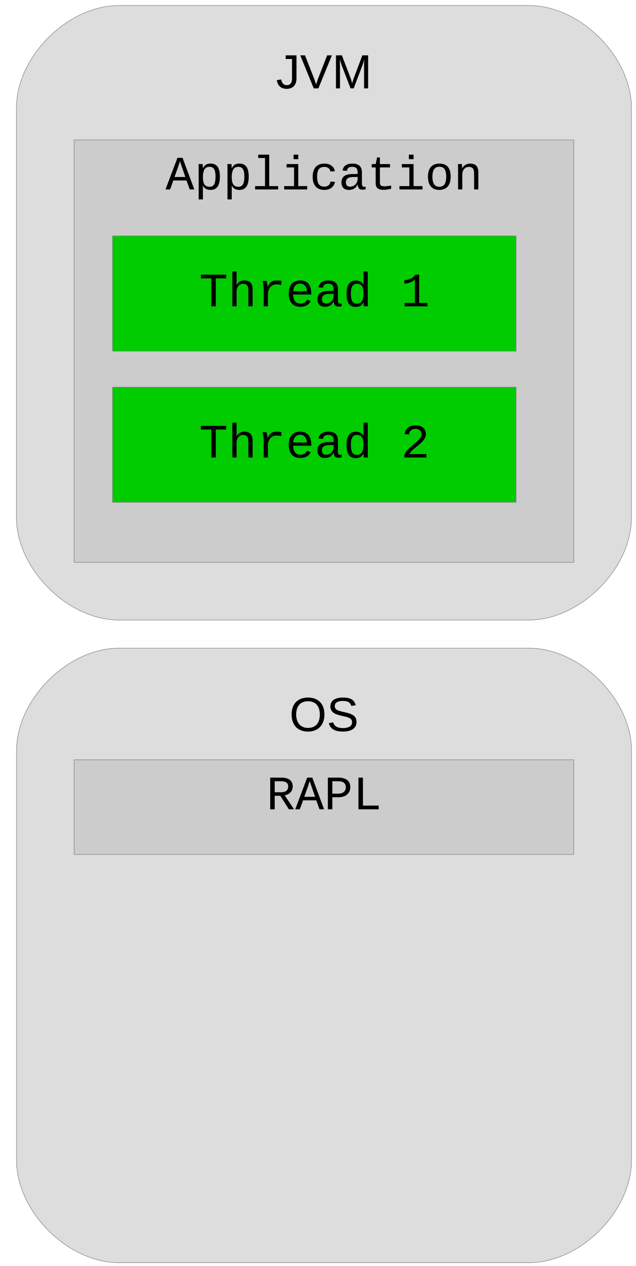
Attribute no joules to either Thread at T3

At timestamp: 5



Concurrency-Aware Attribution

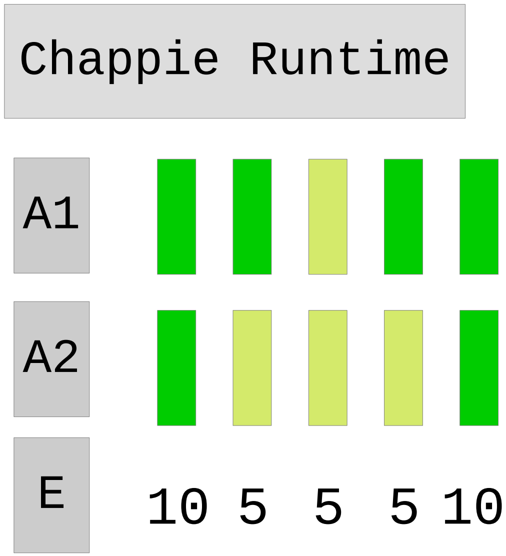


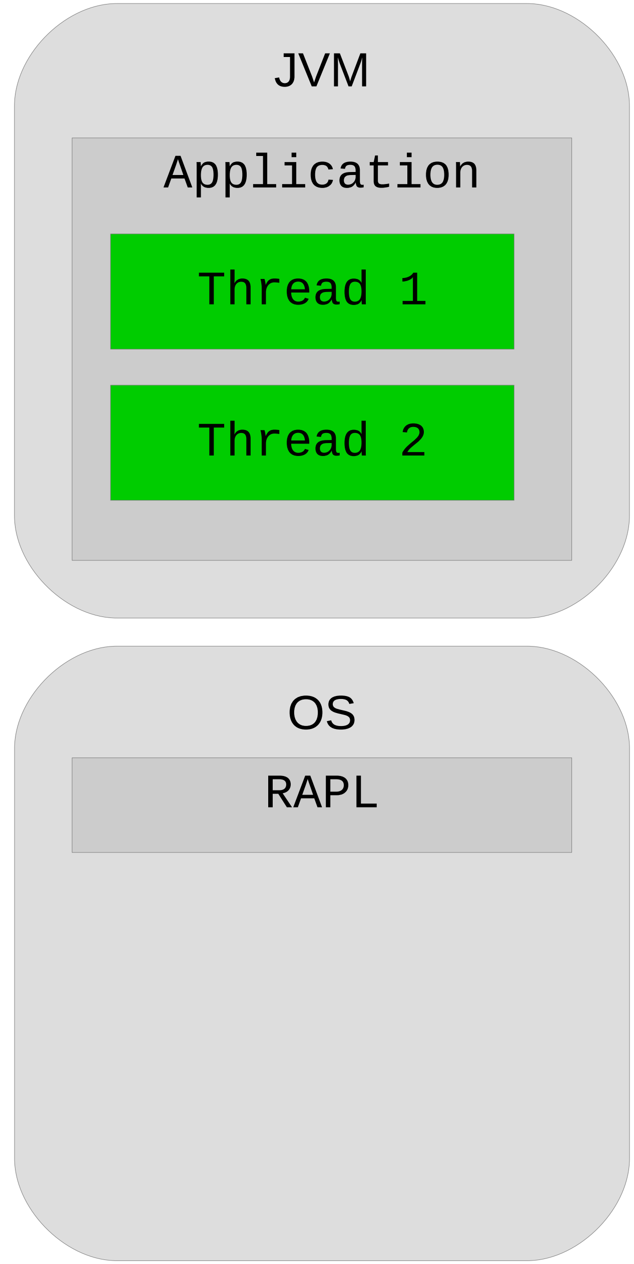


Thread 1: Attributed = 20j
Thread 2: Attributed = 10j

At timestamp: 5

Concurrency-Aware Attribution



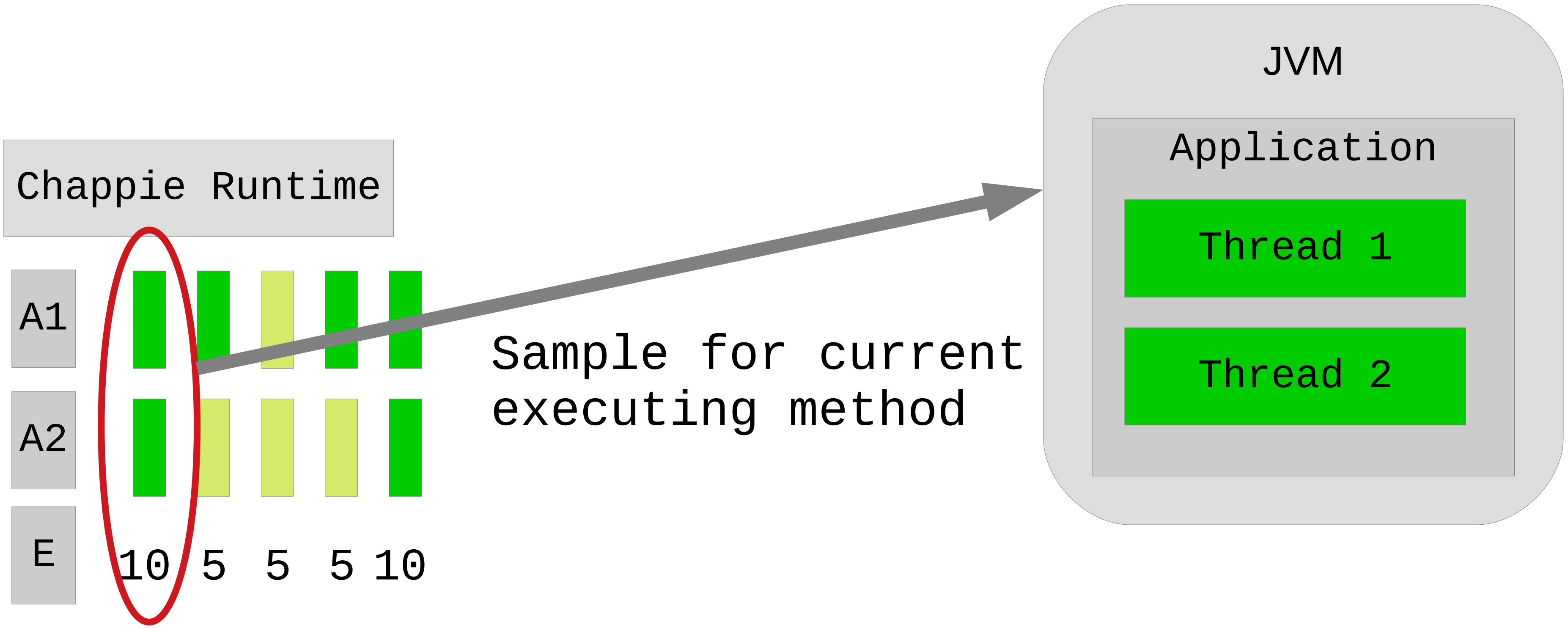


Thread 1: Attributed = 20j
Thread 2: Attributed = 10j

Thread 1: Unattributed = 17.5
Thread 2: Unattributed = 17.5

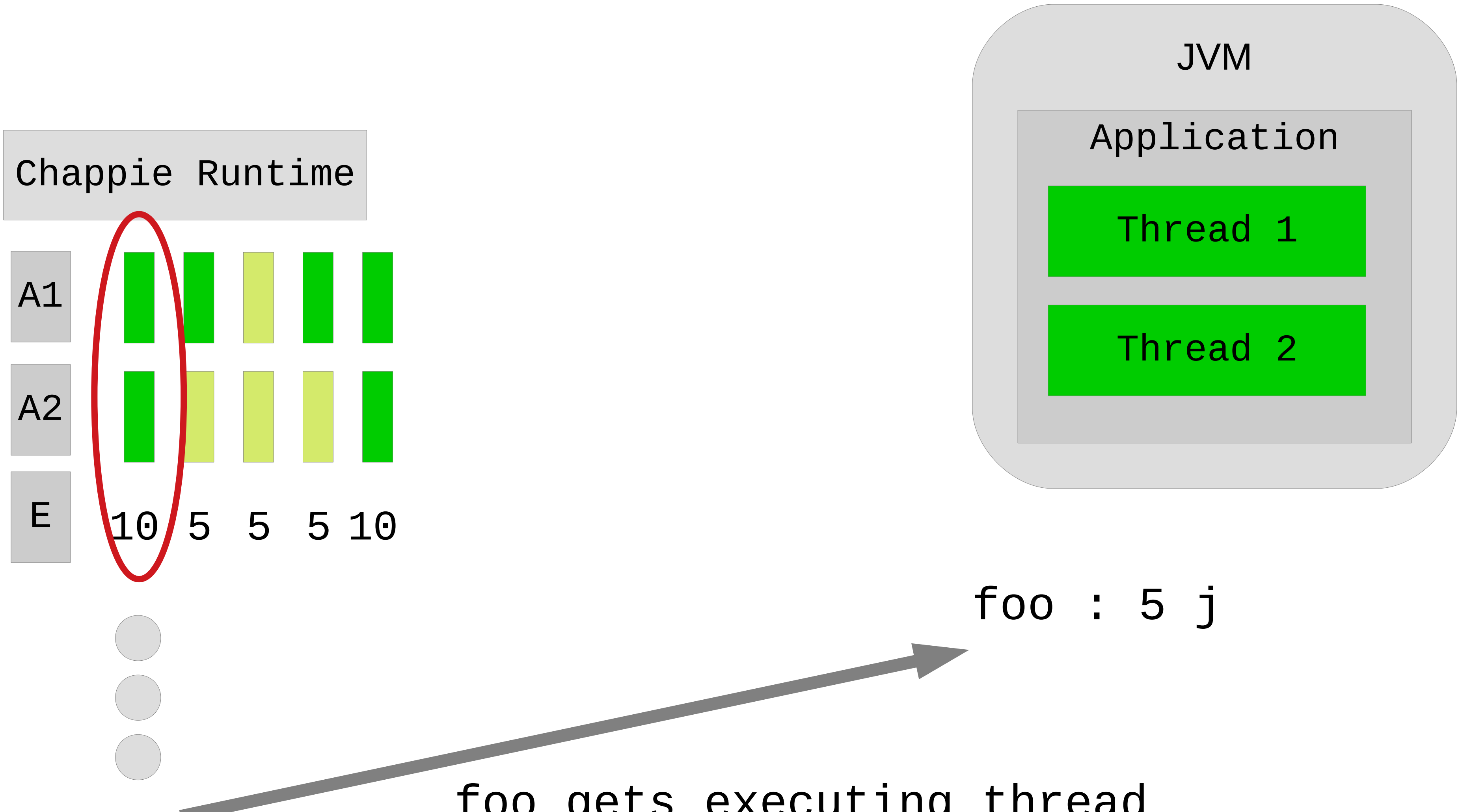
At timestamp: 5

Aligning Attribution to Methods



At timestamp: 1

Aligning Attribution to Methods

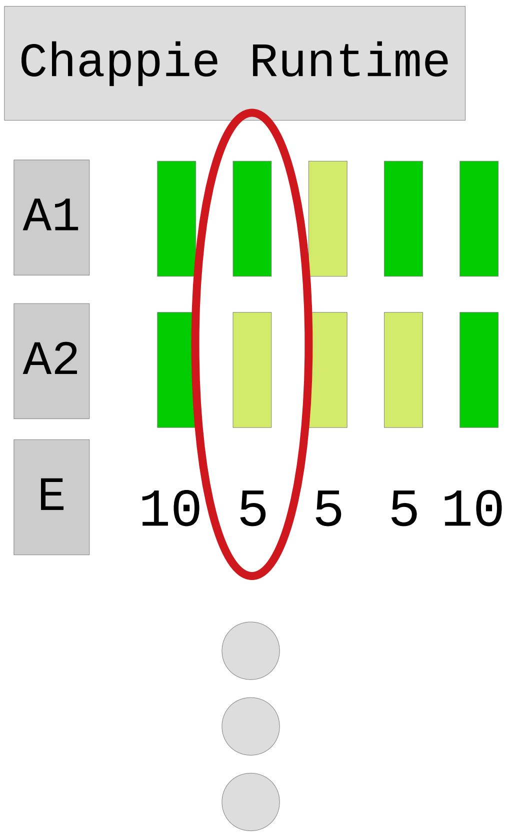


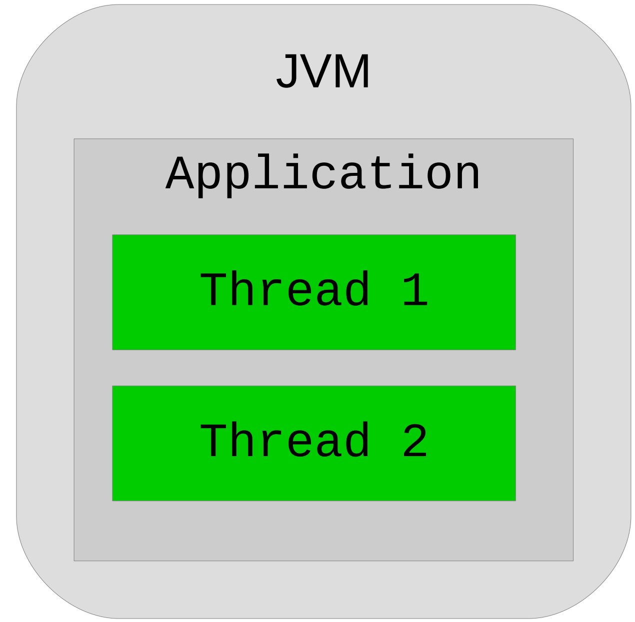
Thread 1 :: foo()

foo gets executing thread attributed energy

At timestamp: 1

Aligning Attribution to Methods





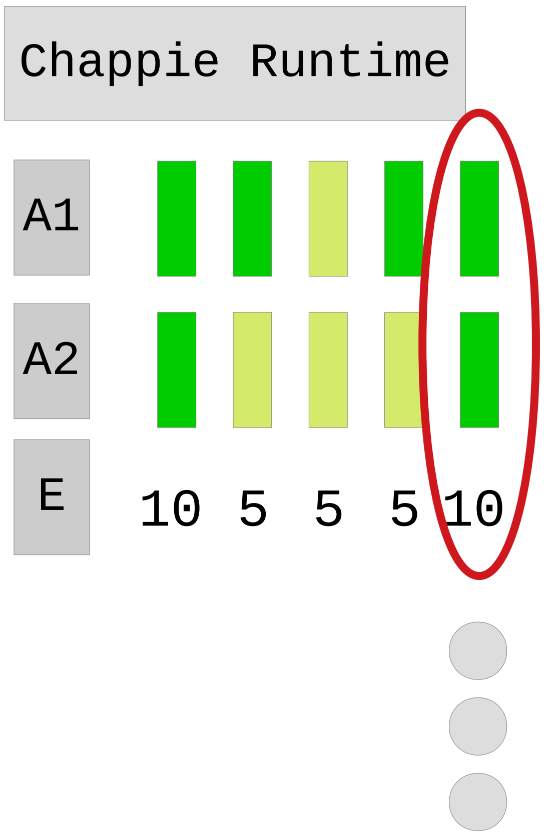
foo : 5 j

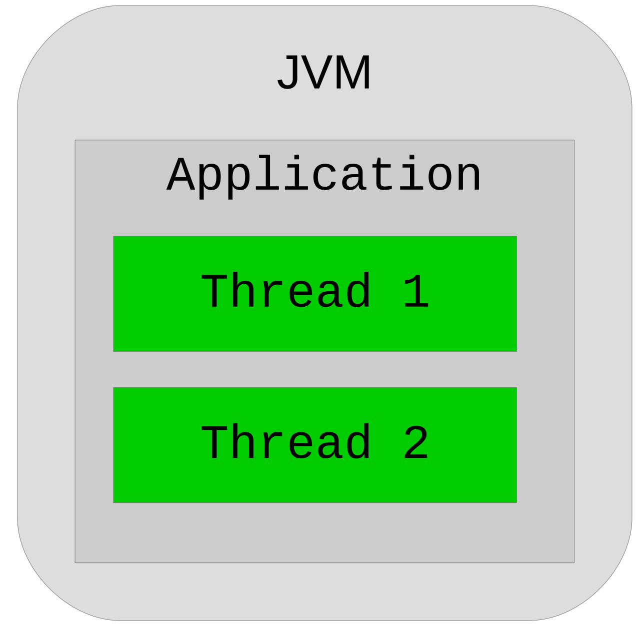
bar : 5 j

Thread 1 :: bar()

At timestamp: 2

Aligning Attribution to Methods





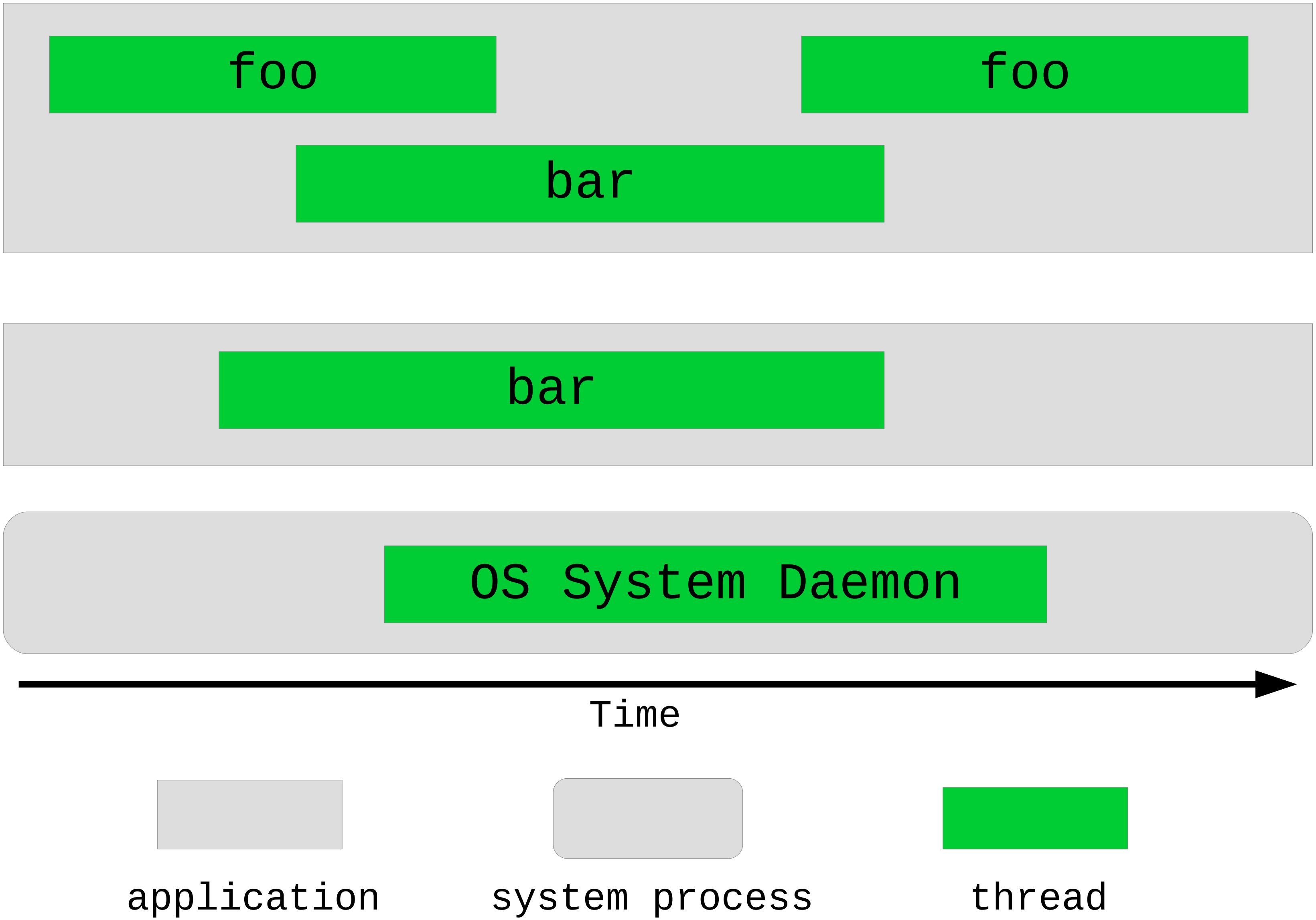
foo : 10 j

bar : 5 j

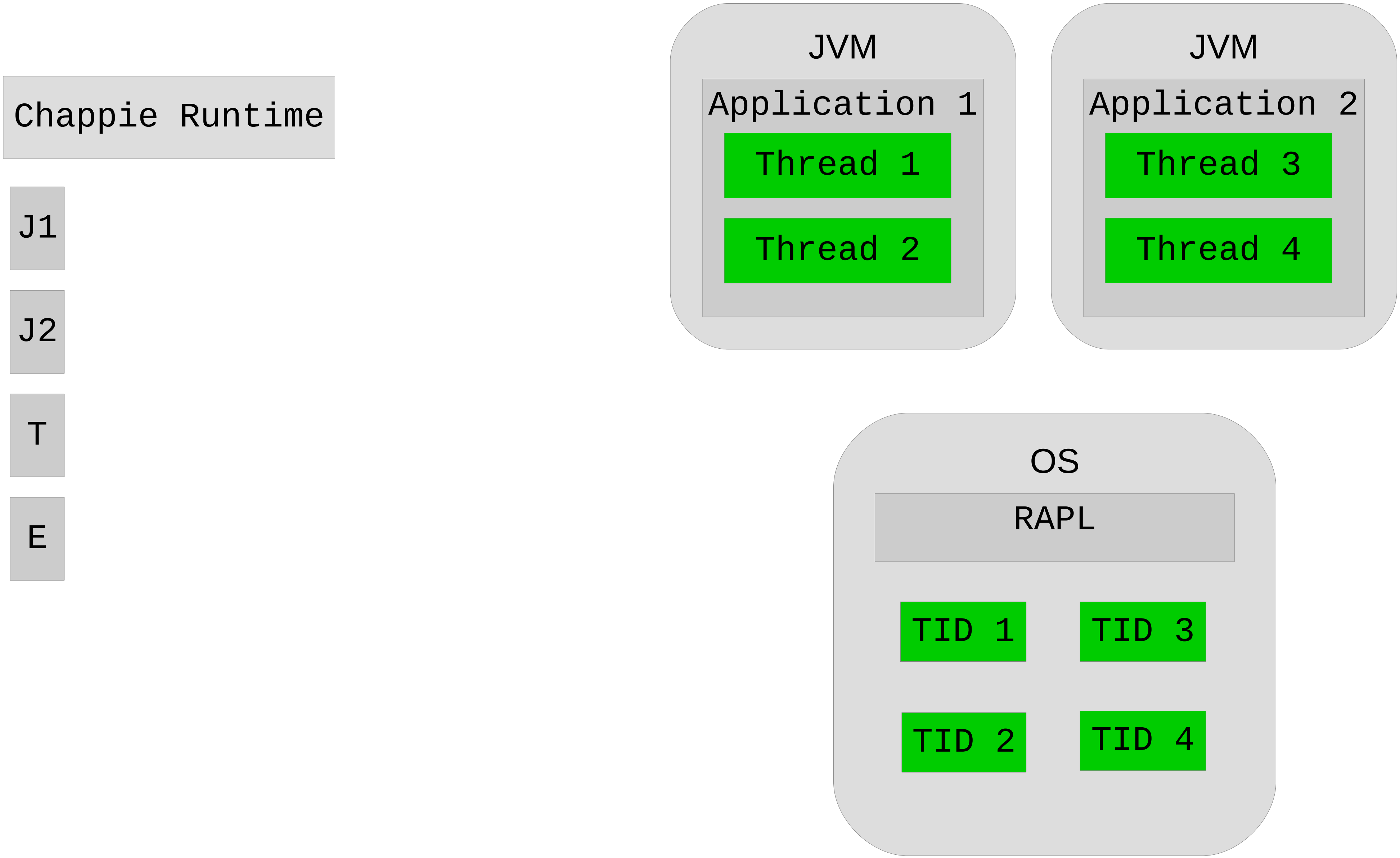
Thread 2 :: foo()

At timestamp: 5

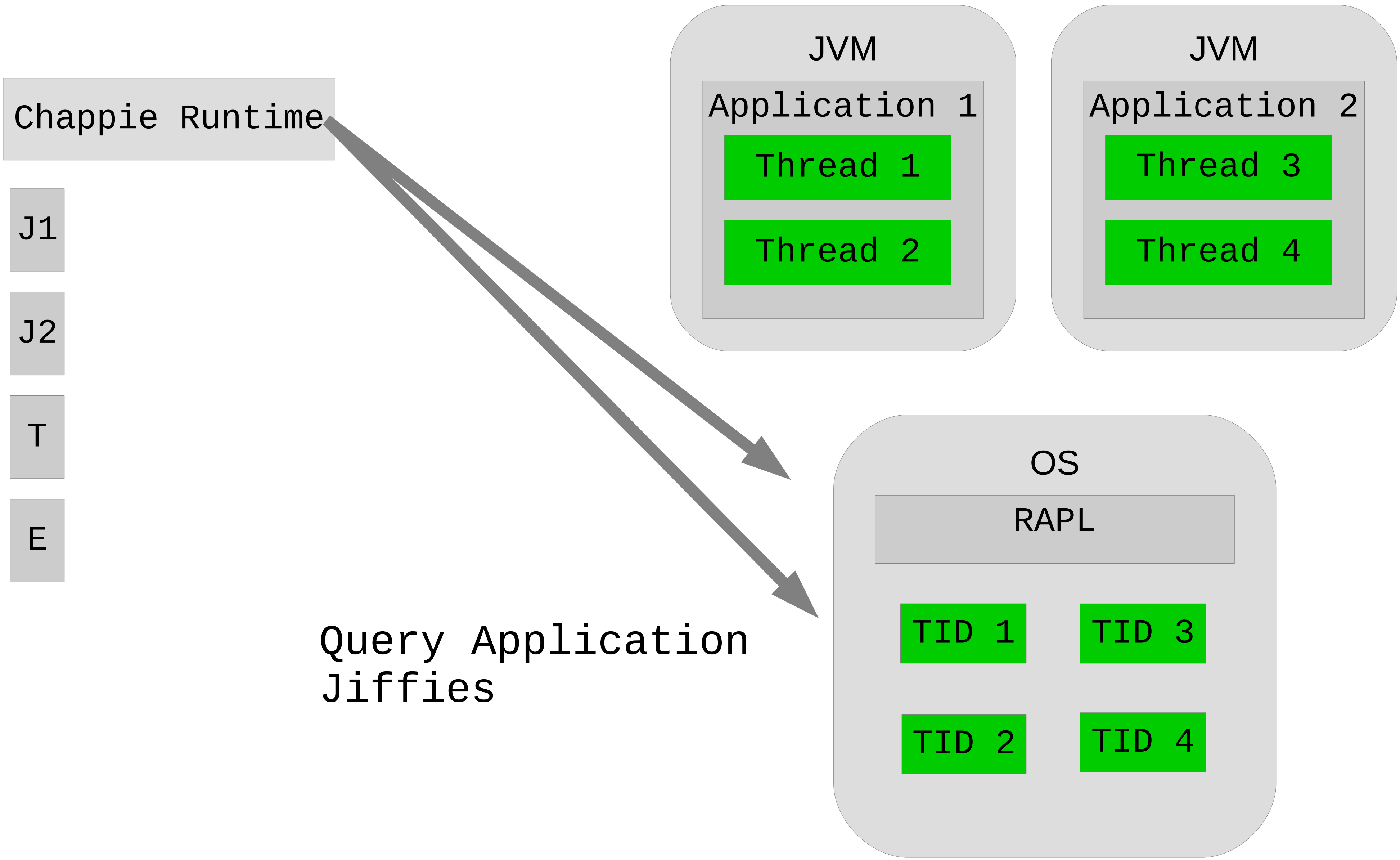
Addressing Problem #2: Co-Running Applications



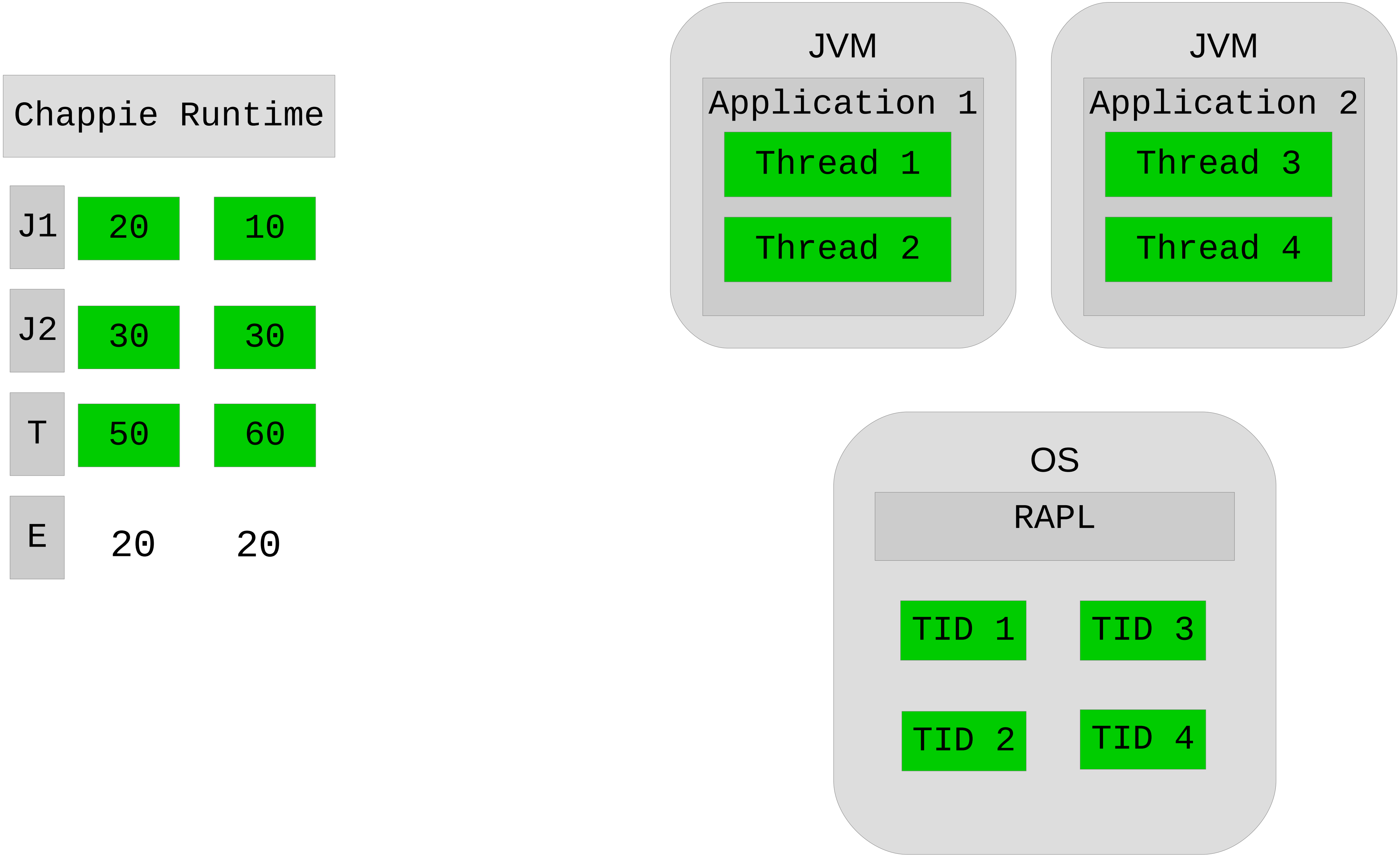
Co-Running Attribution



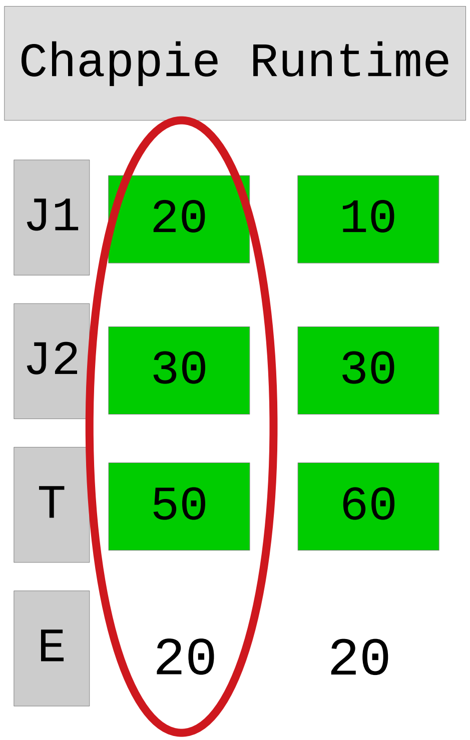
Co-Running Attribution

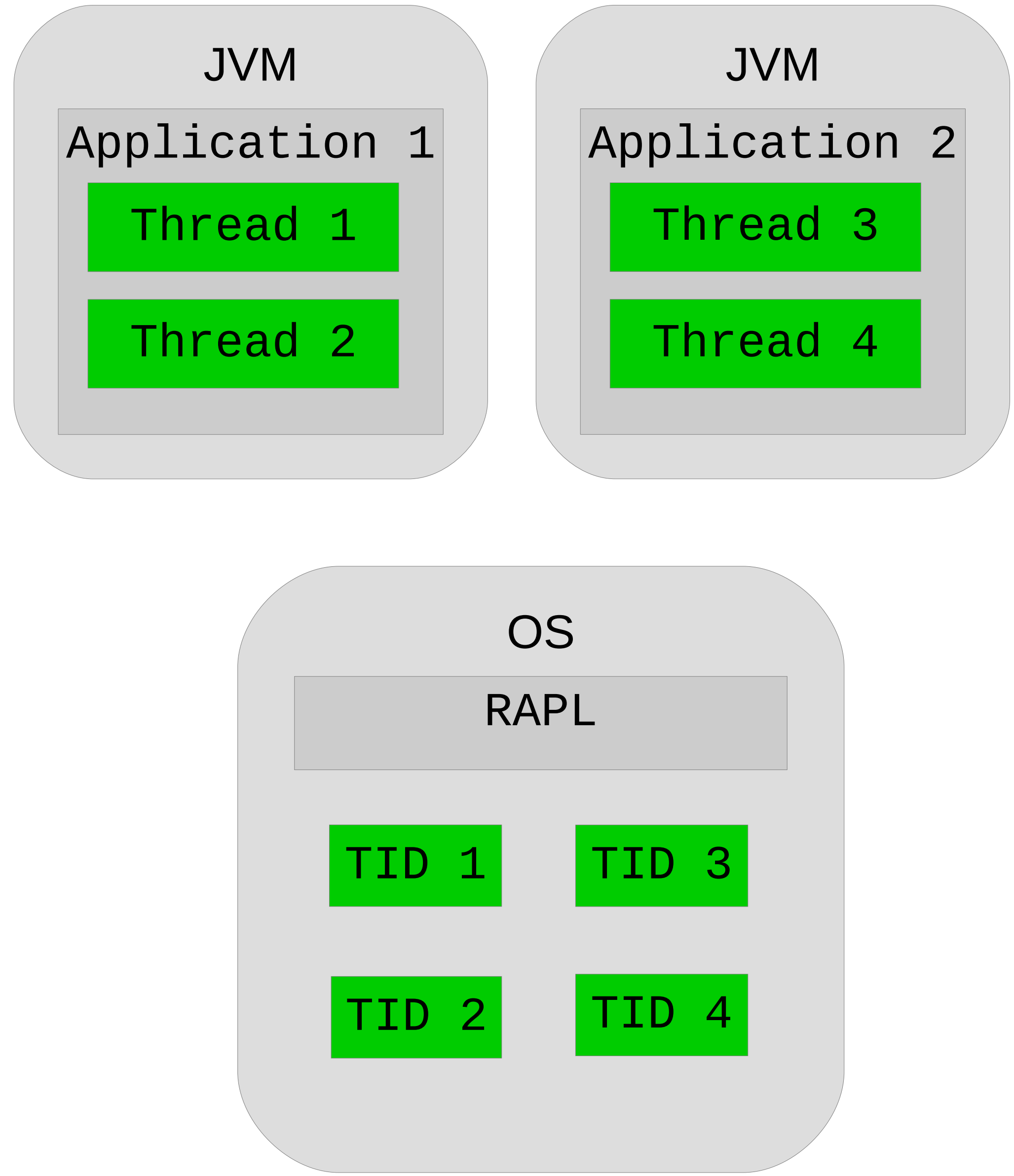


Co-Running Attribution



Co-Running Attribution

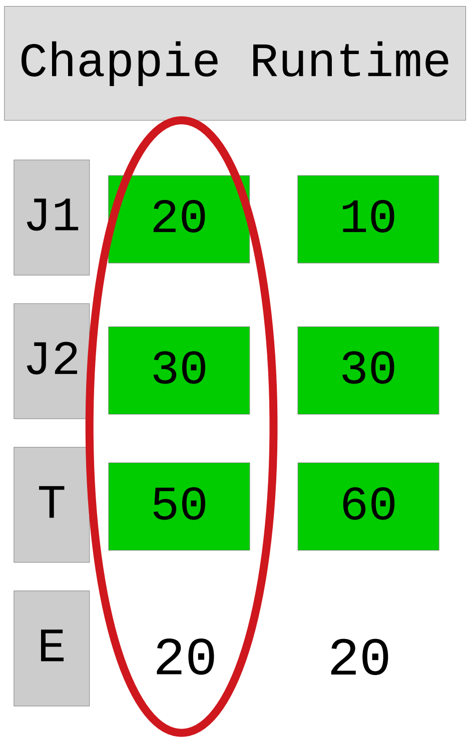




Application 1: 20/50 * 20j

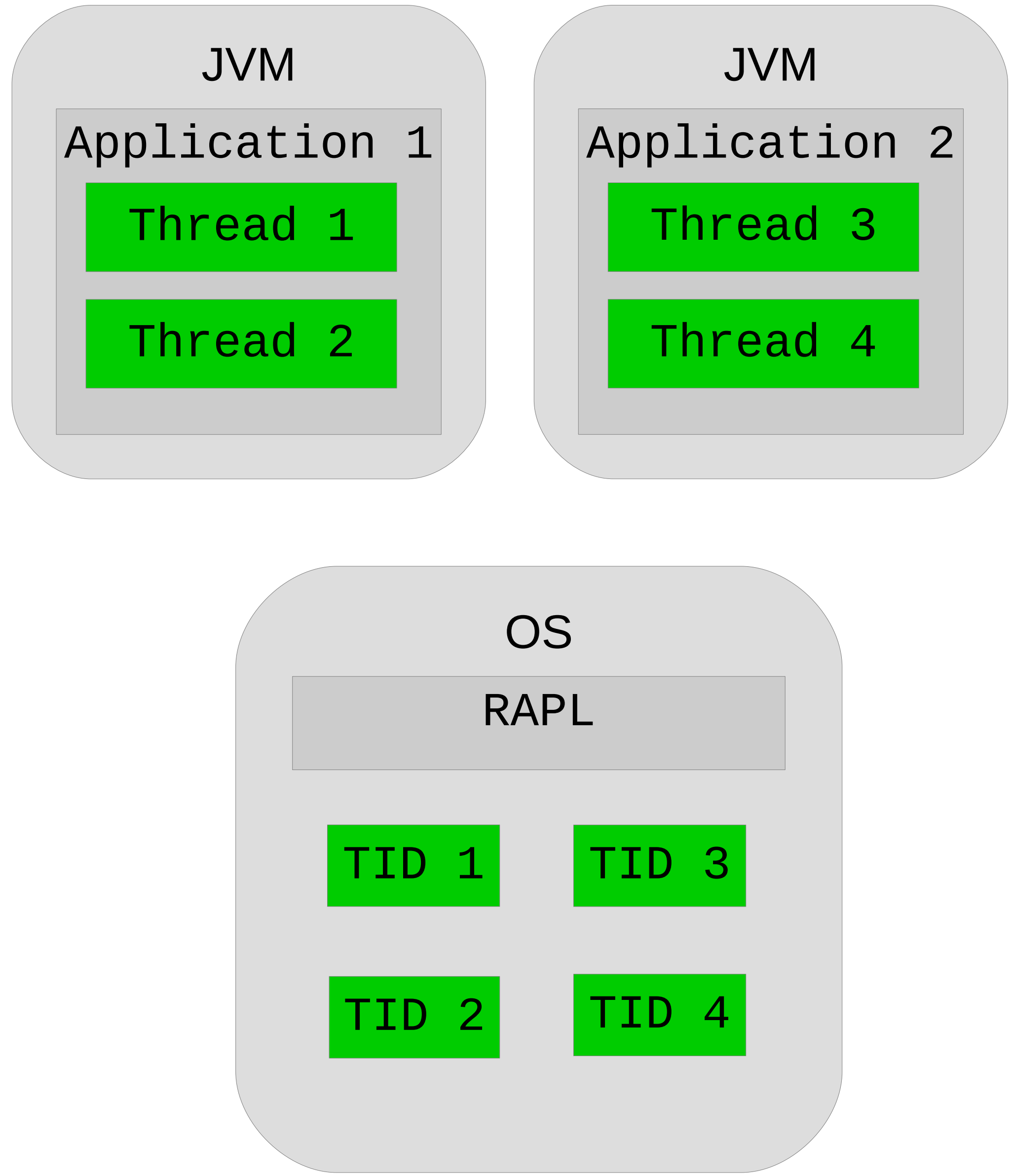
Application 2: 30/50 * 20j

Co-Running Attribution



Application 1: 8j

Application 2: 12j



Putting Two Tiers Together

- Higher rate intra-application accounting
- Lower sampling rate for inter-application accounting

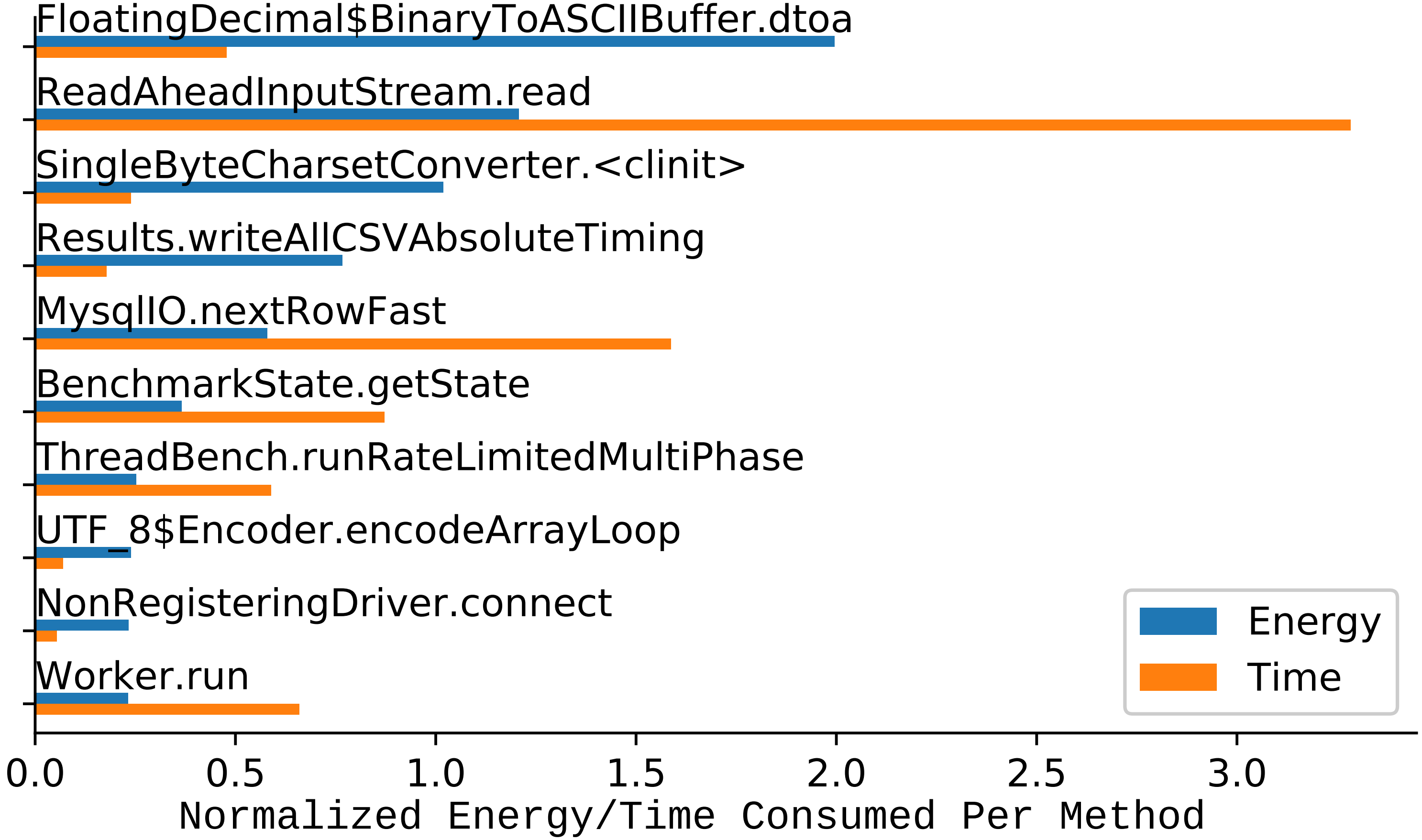
This Talk: **Chappie**

- A cross-layer, concurrency aware design for fine-grained energy accounting of multi-threaded java applications
- A low-overhead sampling-based implementation
- An evaluation on 20 benchmark applications analyzing per-method, per-thread, and per-application energy attribution

Experimental Setup

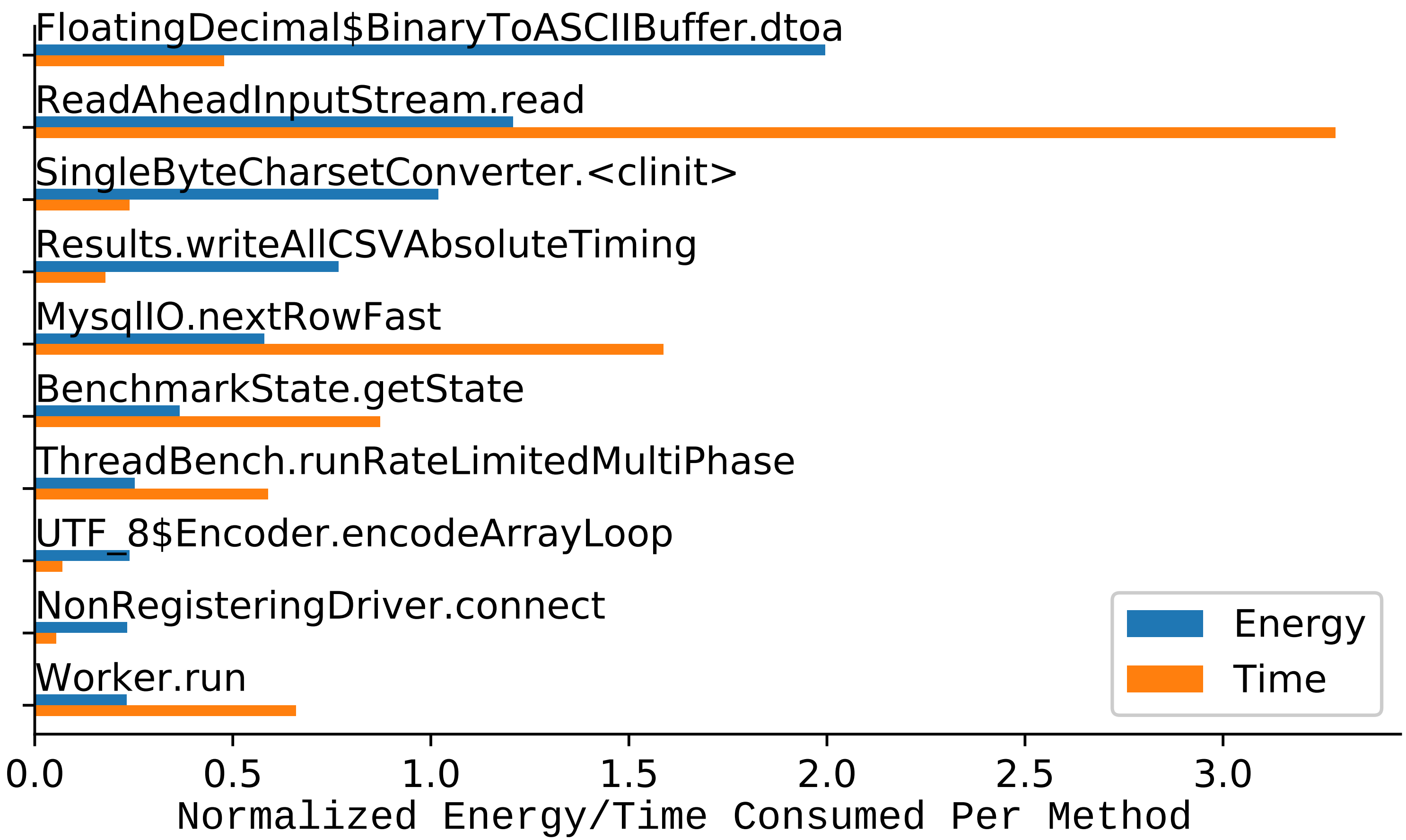
- Experiments performed on dual socket Intel E5-2623 2.20 GHz CPU, 10 cores per socket, 64 DDR RAM, Debian 4.9 OS, default linux power governor
- VM Sampling run at 4ms, OS Samping at 40ms
- Benchmarks from Dacapo, Graphchi, and OLTPBench
- Experiments run 20 times for Dacapo, 10 times for Graphchi and OLTPBench

Method Accounting



twitter

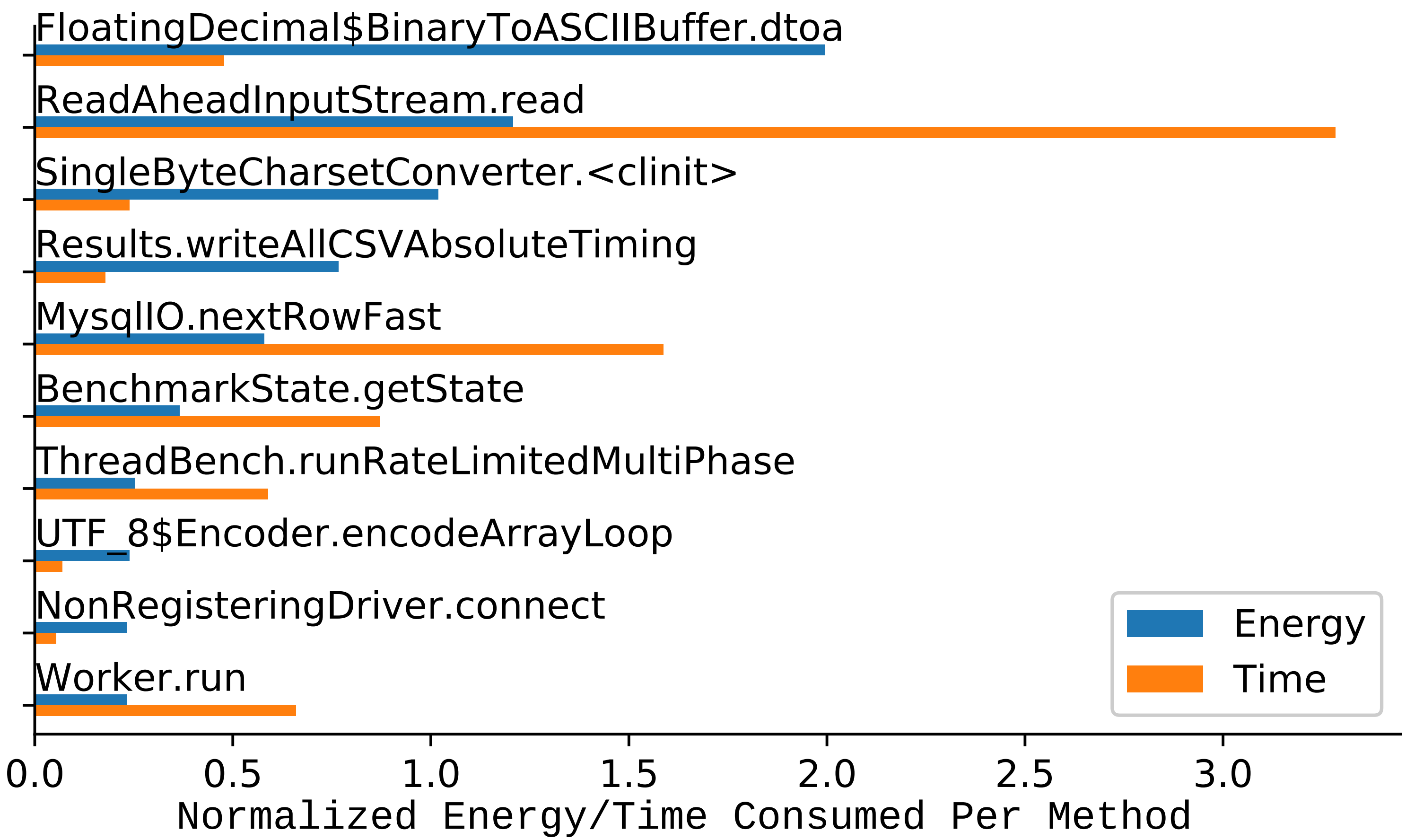
Method Accounting



twitter

Top 10 consuming methods for twitter

Method Accounting

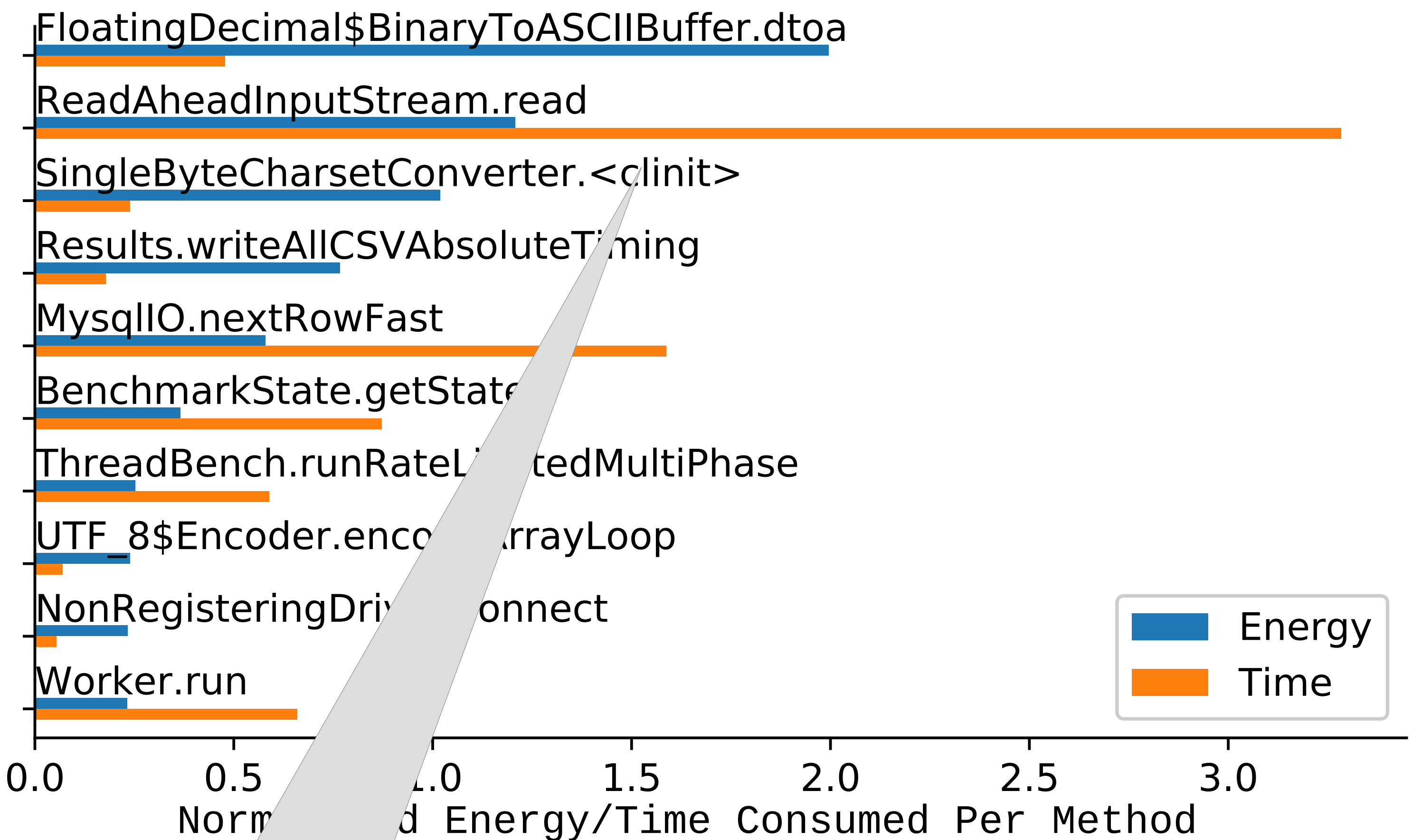


twitter

Top 10 consuming methods for twitter

Important for energy debugging and program understanding

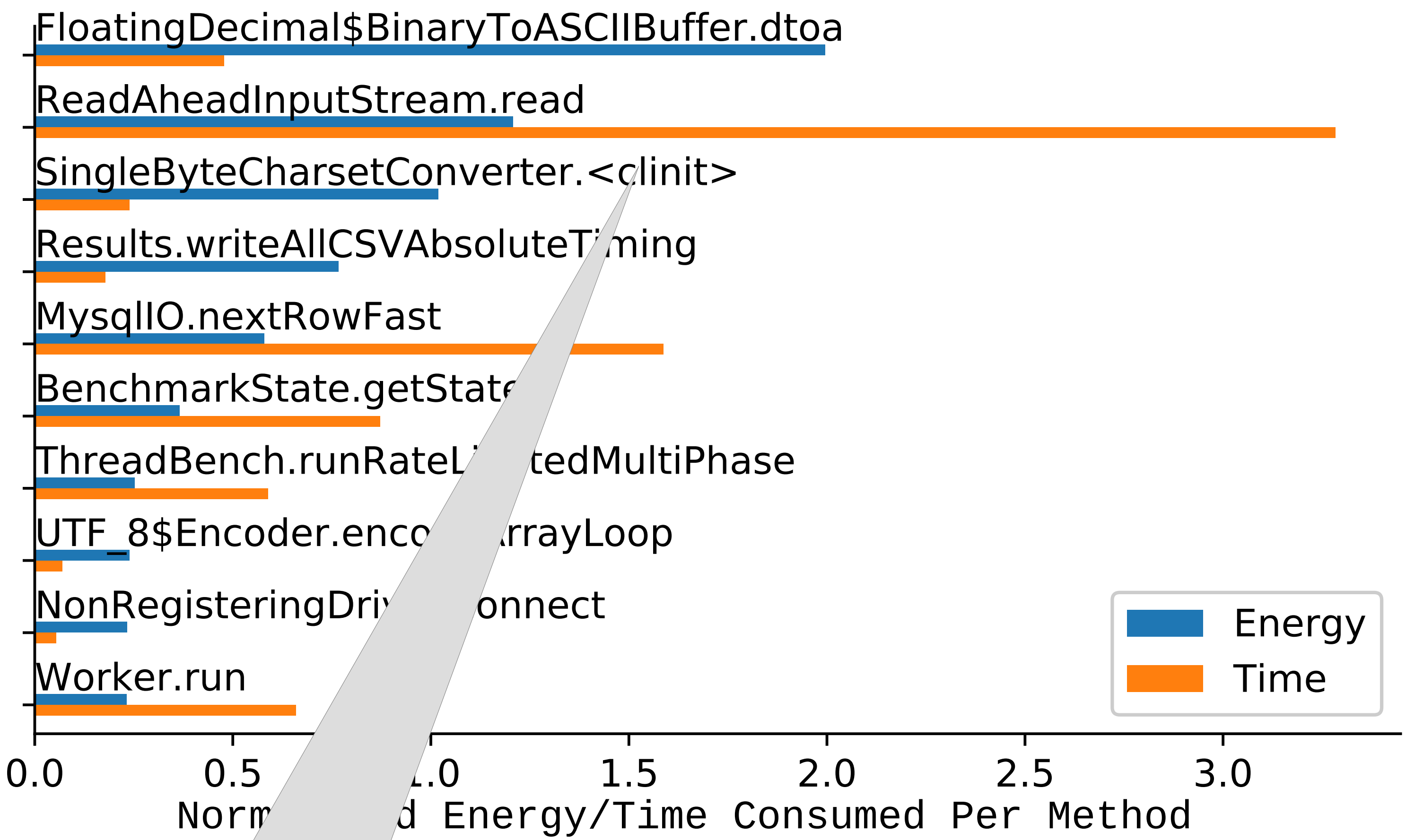
Method Accounting



twitter

Difference between energy and time

Method Accounting

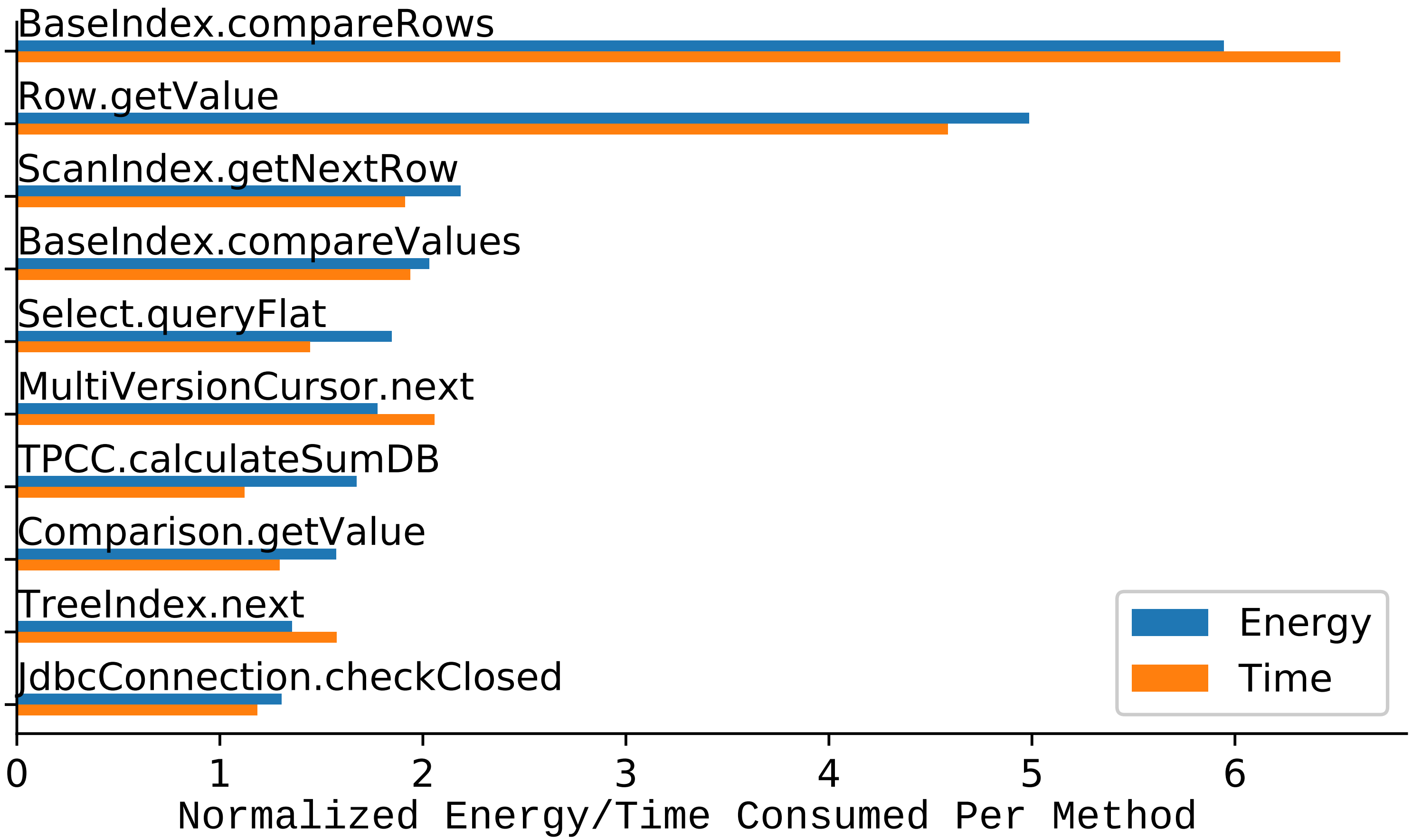


twitter

Difference between energy and time

Important for fine-grained "logical" power analysis

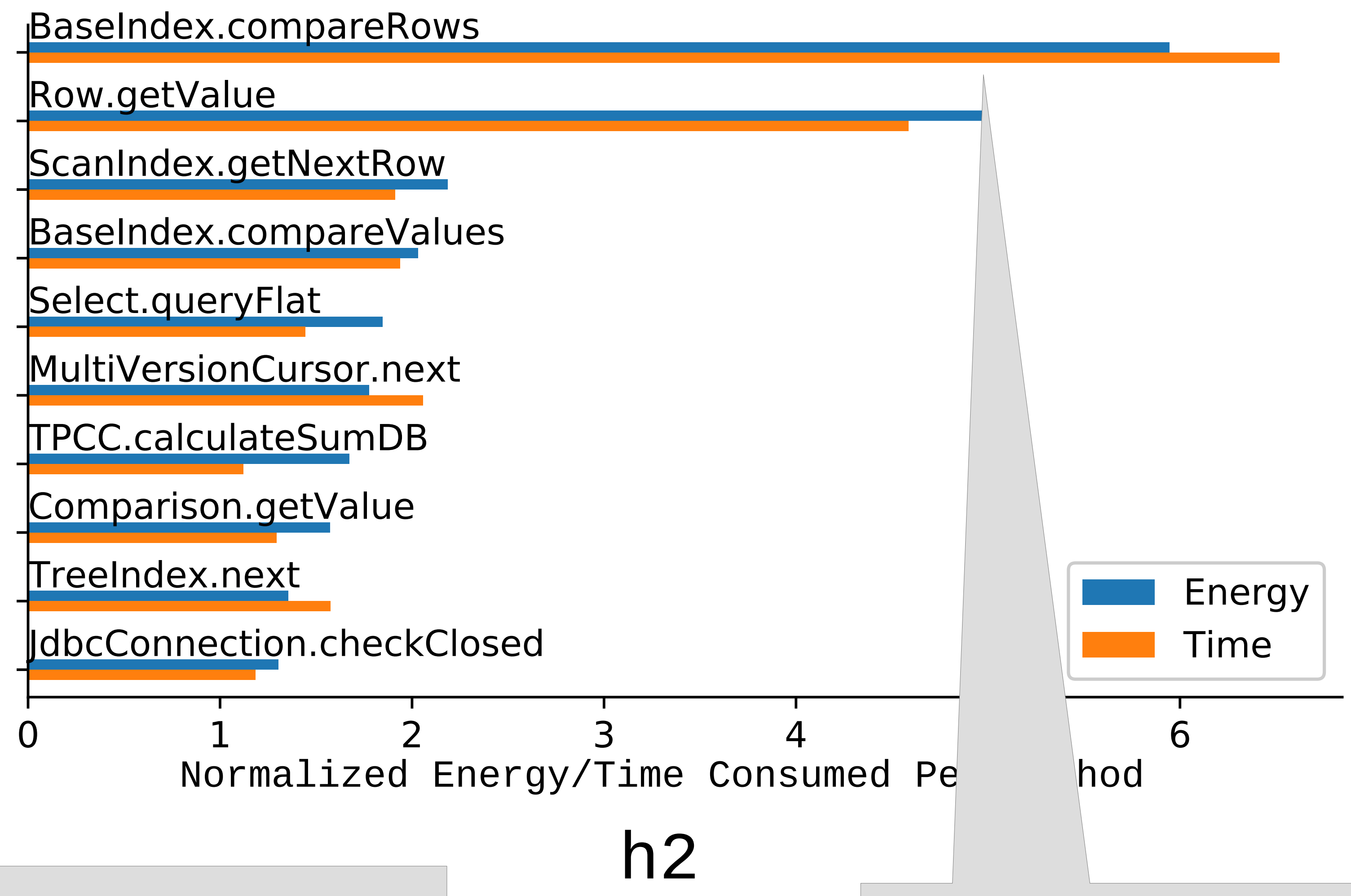
Method Accounting



h2

Top 10 consuming methods for h2

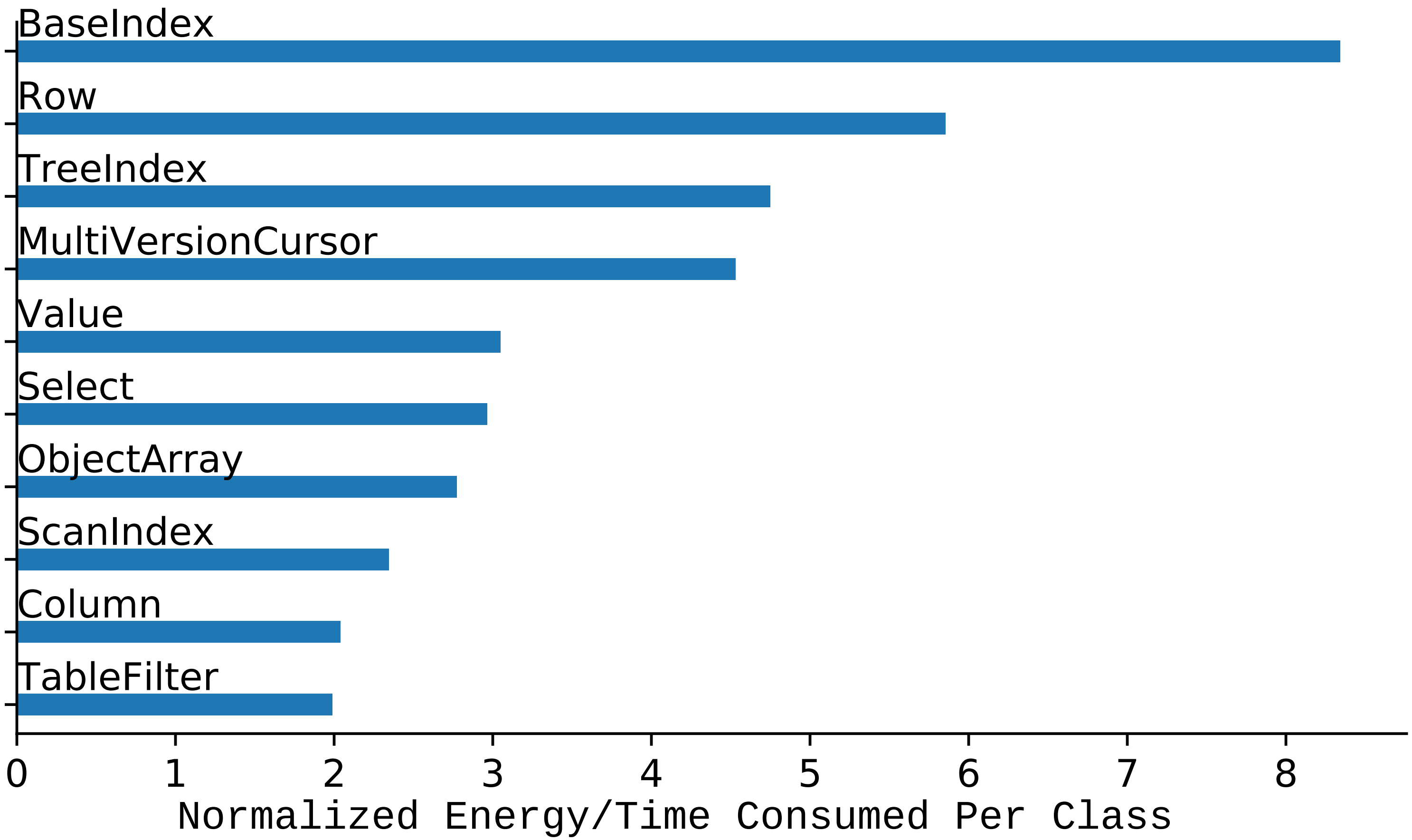
Method Accounting



Top 10 consuming methods for h2

Known optimization for h2

Class Accounting

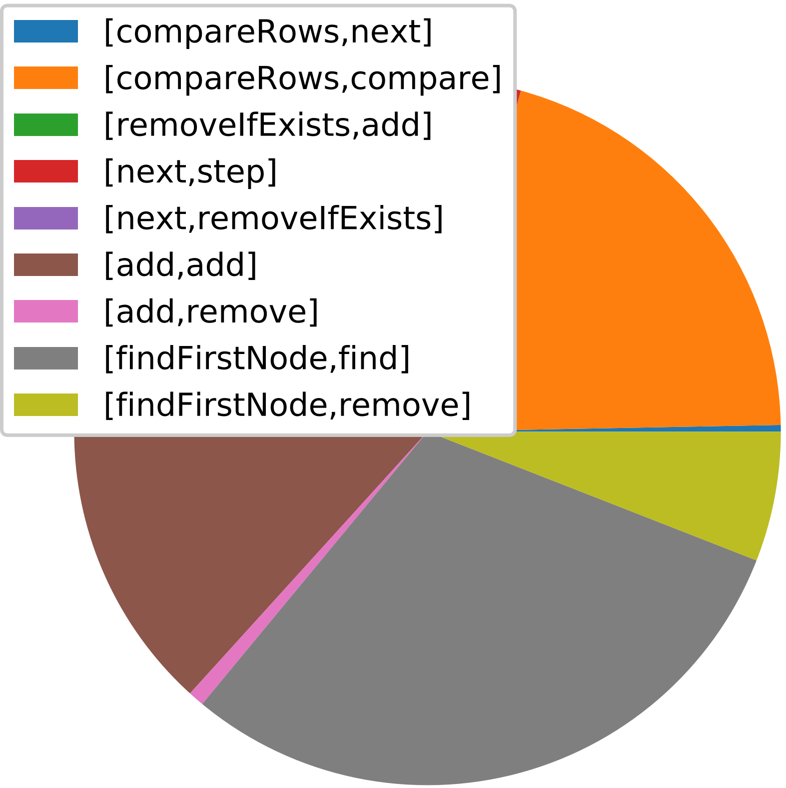


h2

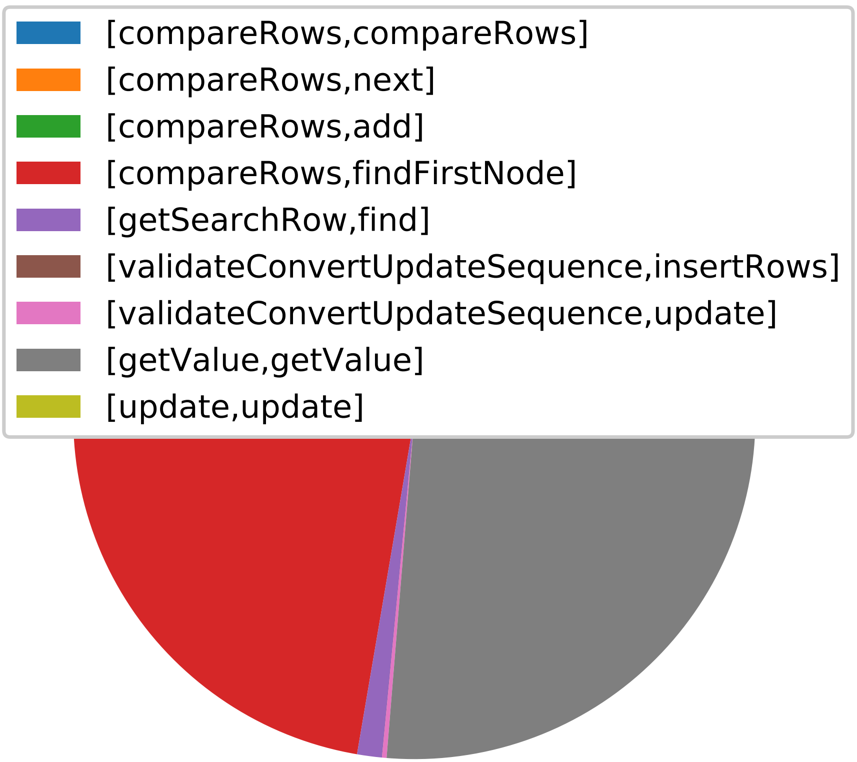
Top 10 consuming classes for h2

Context-Sensitive Method Accounting

BaseIndex.compareRows



Row.getValue

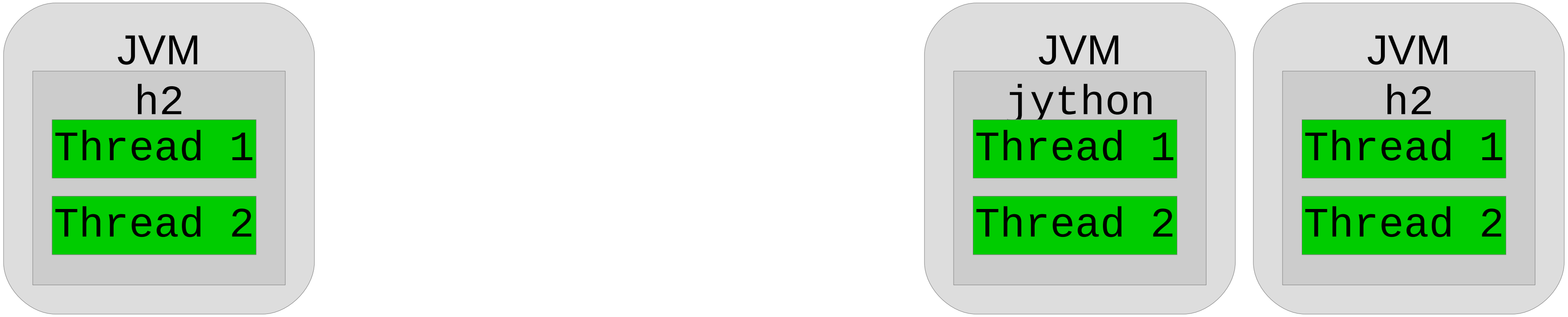


Percent Energy Per Calling Context for Method

h2

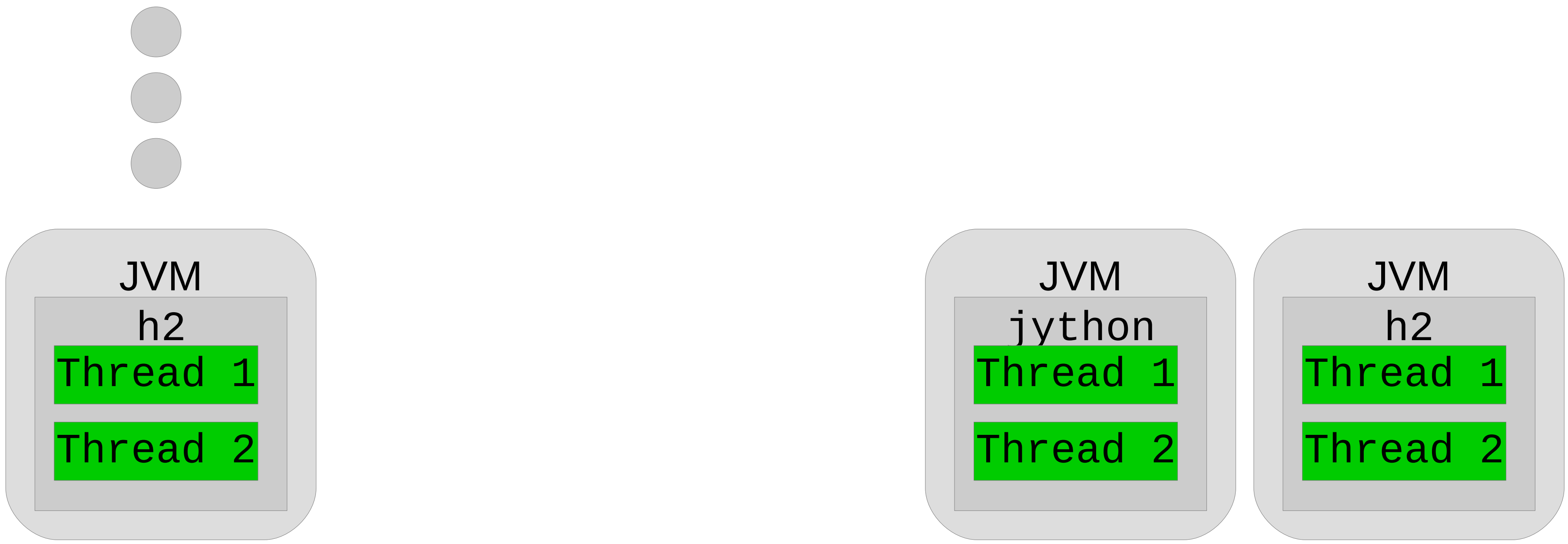
2-CFA Context

Analyzing Co-Running Applications



Analyzing Co-Running Applications

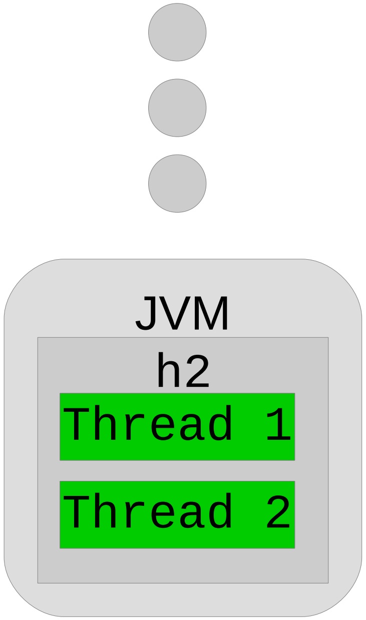
1. compareRows
2. getValue
3. getNextRow
4. compareValues

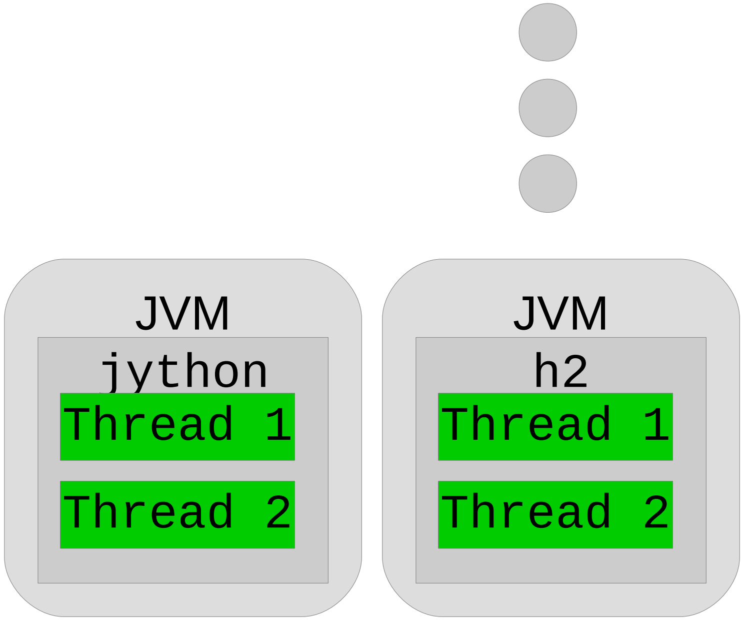


Analyzing Co-Running Applications

1. compareRows
2. getValue
3. getNextRow
4. compareValues

1. compareRows
2. getValue
3. compareValues
4. getNextRow





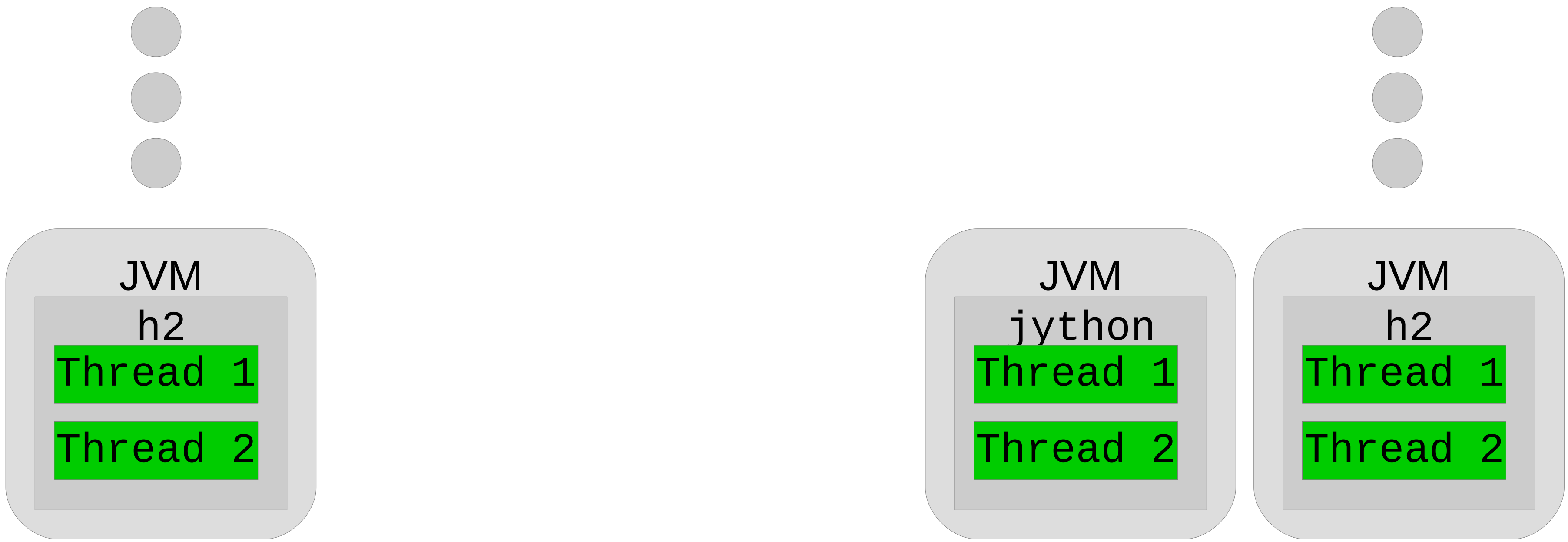
Analyzing Co-Running Applications

1. compareRows
2. getValue
3. getNextRow
4. compareValues

Pearson Correlation Coefficient (PCC)

PCC > 0.7 indicates strong correlation

1. compareRows
2. getValue
3. compareValues
4. getNextRow



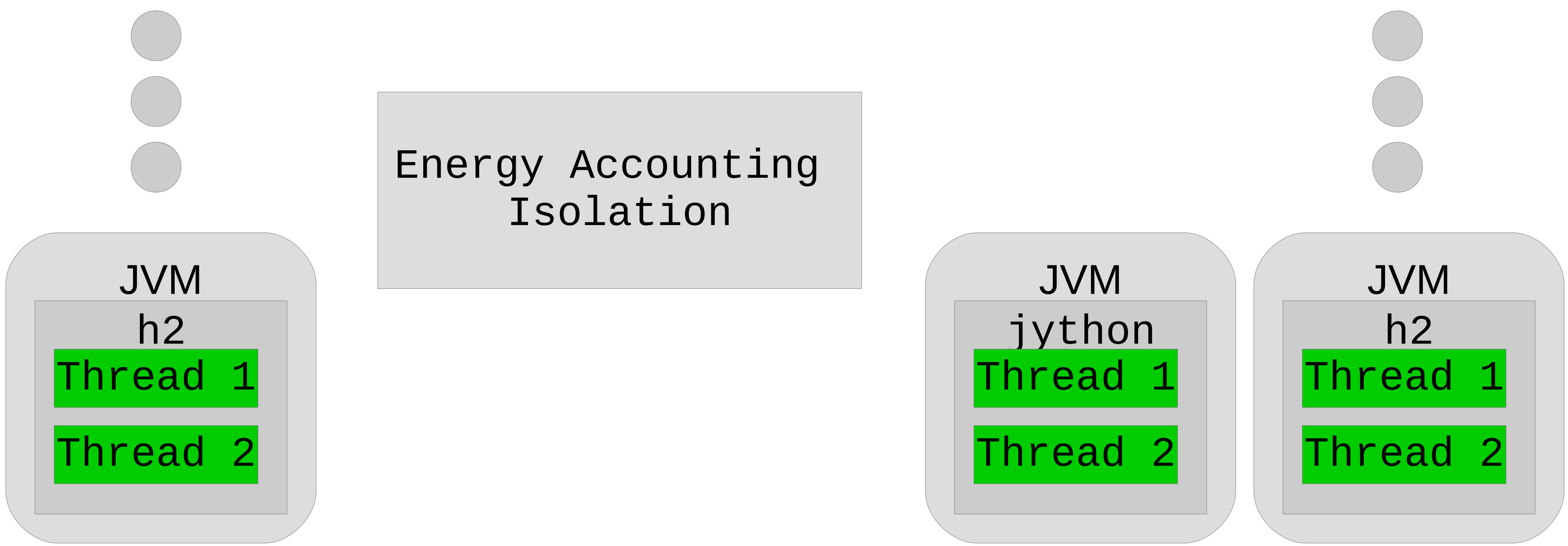
Analyzing Co-Running Applications

1. compareRows
2. getValue
3. getNextRow
4. compareValues

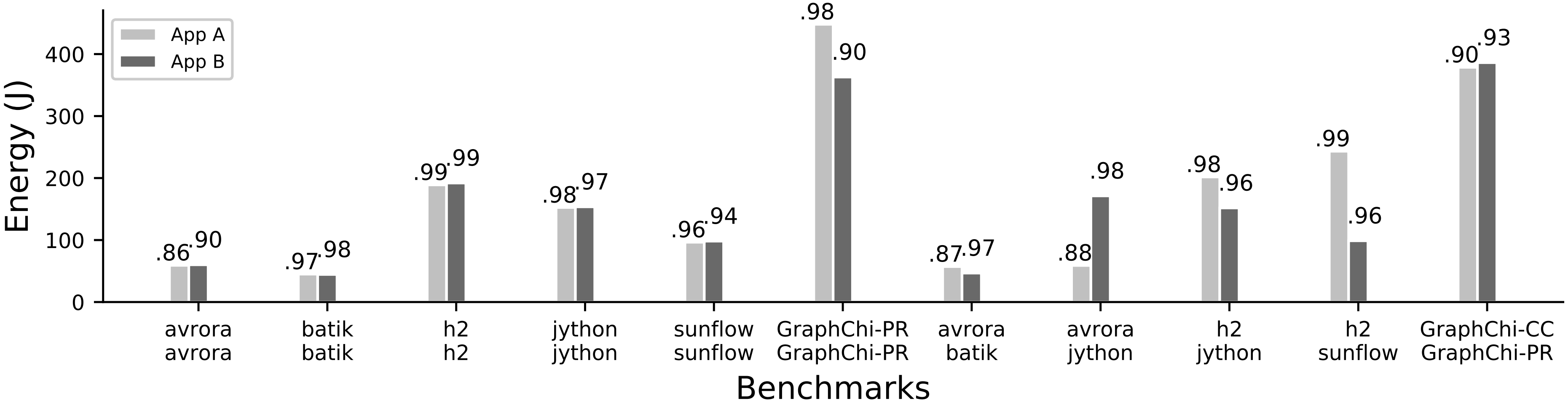
Pearson Correlation Coefficient (PCC)

PCC > 0.7 indicates strong correlation

1. compareRows
2. getValue
3. compareValues
4. getNextRow

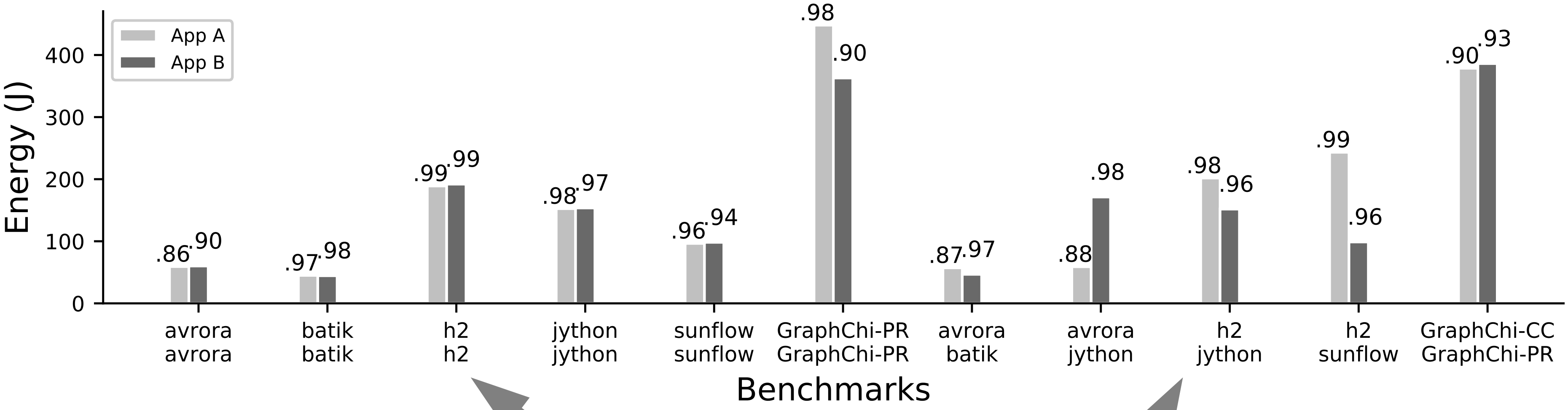


Co-Running Applications



Each pair represent co-running apps

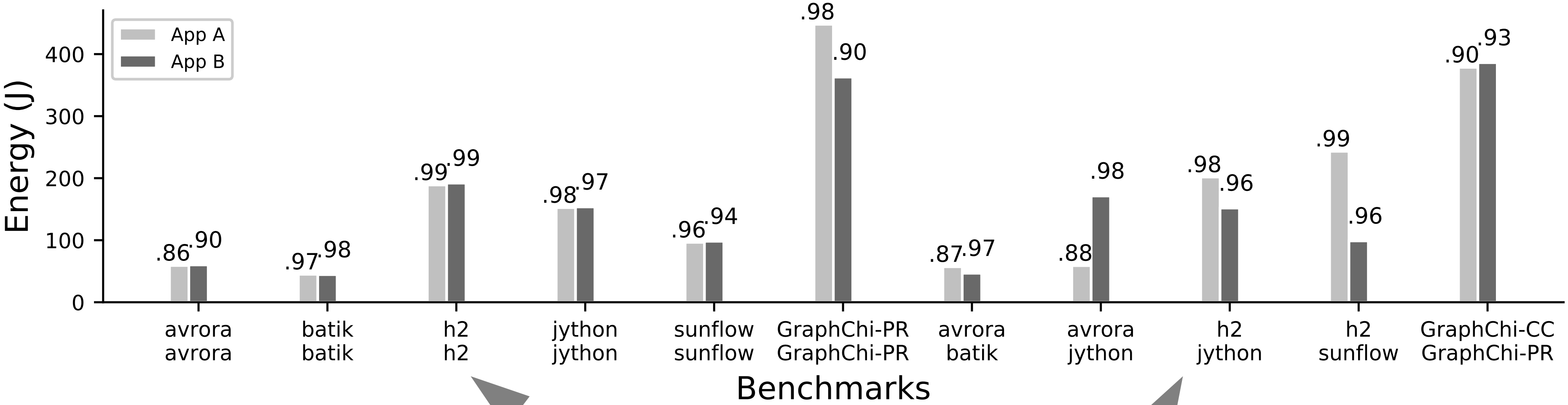
Co-Running Applications



Energy consistent across loads

Each pair represent co-running apps

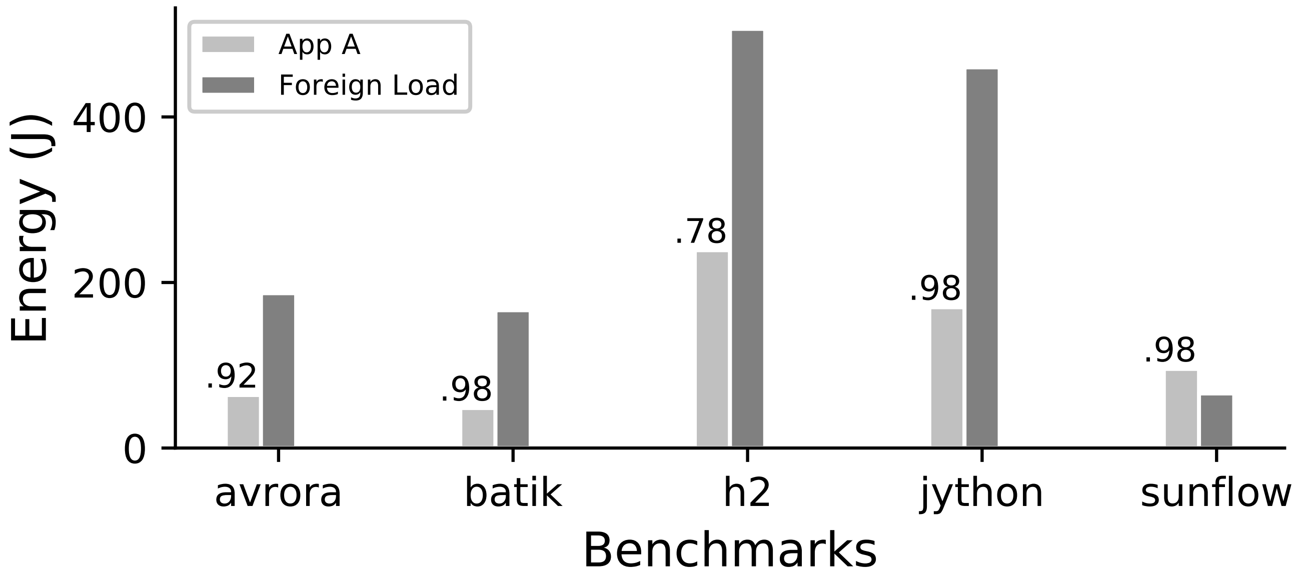
Co-Running Applications



PCC shows method ranking stable

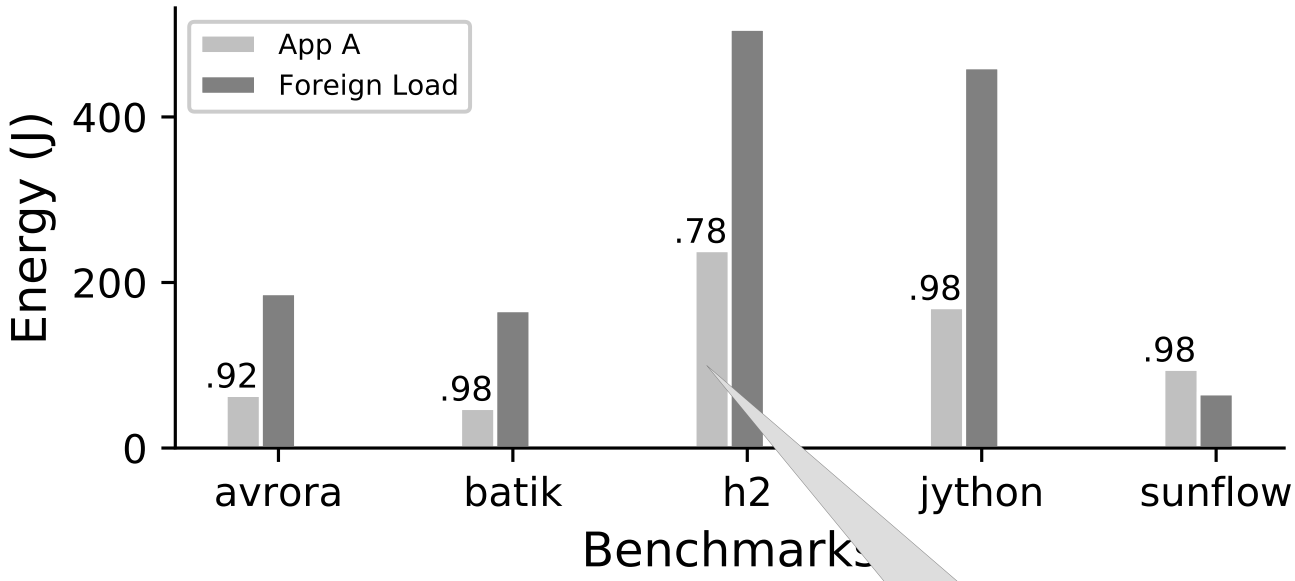
Each pair represent co-running apps

Application with Foreign Load



Benchmark run with PARSEC Ferret

Application with Foreign Load



Energy consistent, PCC stable

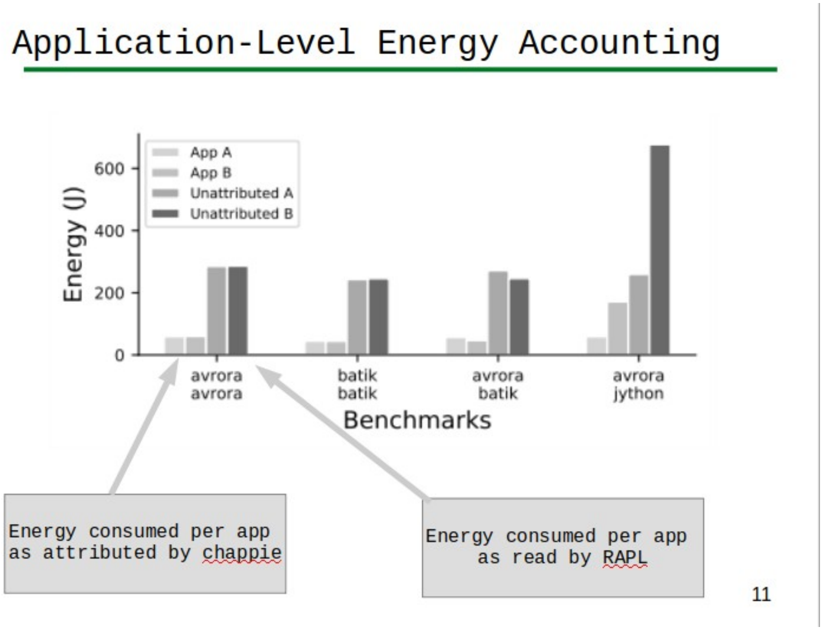
Benchmark run with PARSEC Ferret

Chappie Overhead

Benchmark	Total Threads	Time Overhead
avrora	11	4.6%
batik	12	12.3%
eclipse	23	8.8%
h2	46	4.0%
GraphChi-ALS	47	15.1%
Twitter	17	0.4%

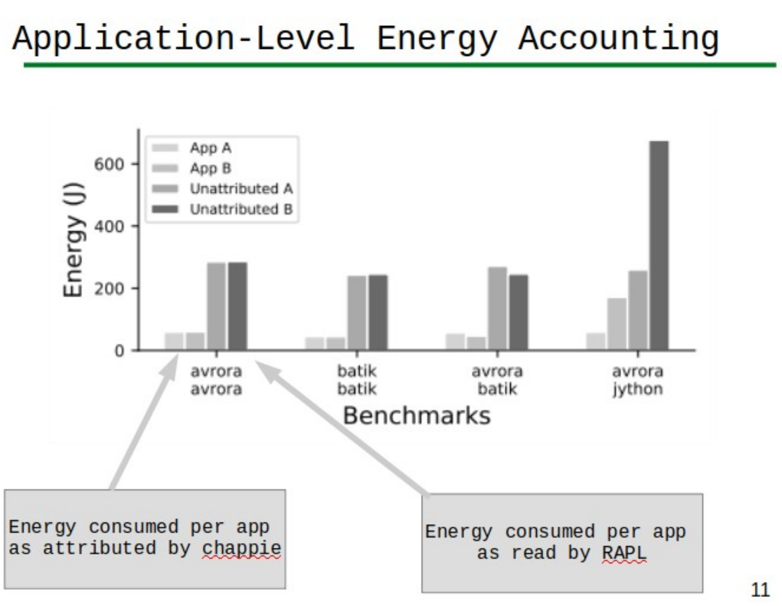
Summary

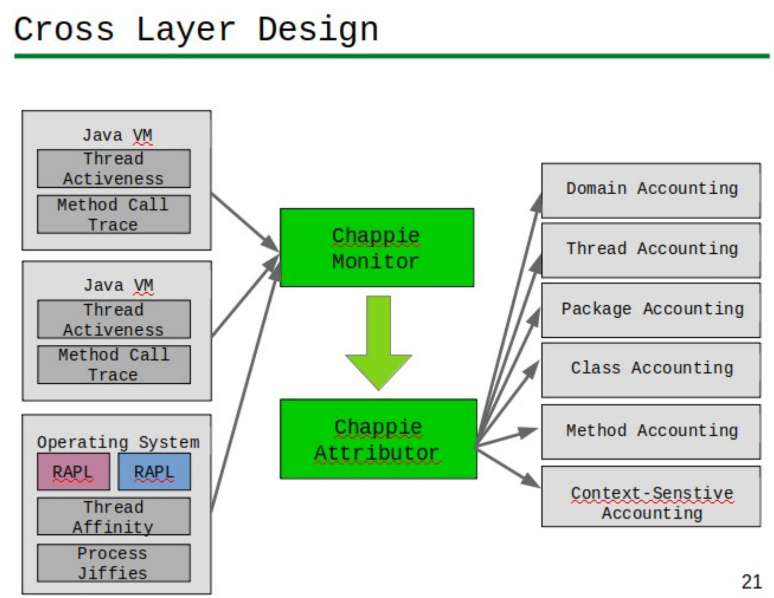
Summary



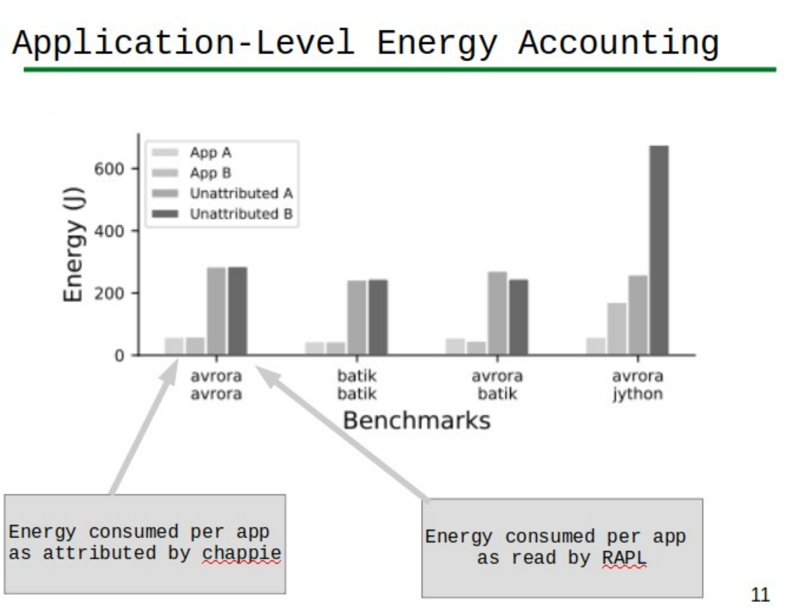
Application-Level Energy Accounting
App A
App B
Unattributed A
Unattributed B
Energy (J)
600
400
200
0
avrora avrora
batik batik
avrora batik
avrora jython
Benchmarks
Energy consumed per app as attributed by chappie
Energy consumed per app as read by RAPL
11

Summary

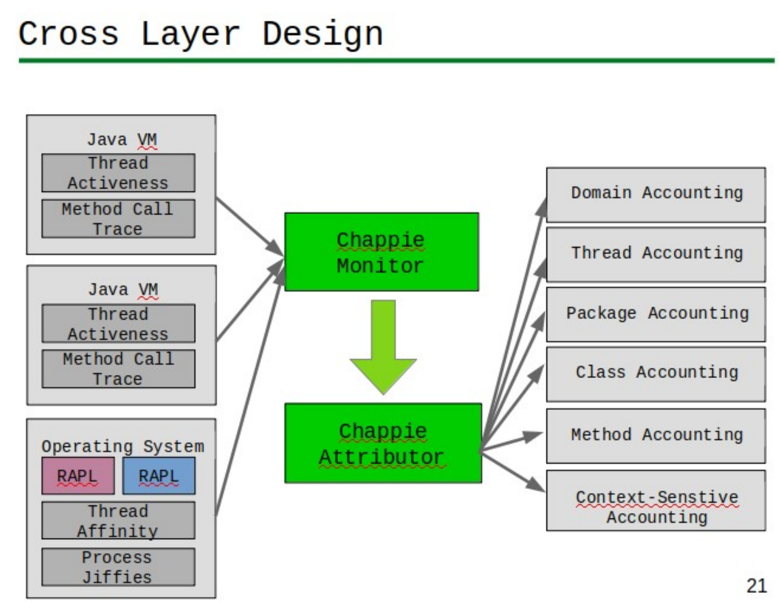

Application-Level Energy Accounting
Energy (J)
600
400
200
0
App A
App B
Unattributed A
Unattributed B
avrora avrora
batik batik
avrora batik
avrora jython
Benchmarks
Energy consumed per app as attributed by chappie
Energy consumed per app as read by RAPL
11


Cross Layer Design
Java VM
Thread Activeness
Method Call Trace
Java VM
Thread Activeness
Method Call Trace
Operating System
RAPL
RAPL
Thread Affinity
Process Jiffies
Chappie Monitor
Chappie Attributor
Domain Accounting
Thread Accounting
Package Accounting
Class Accounting
Method Accounting
Context-Senstive Accounting
21

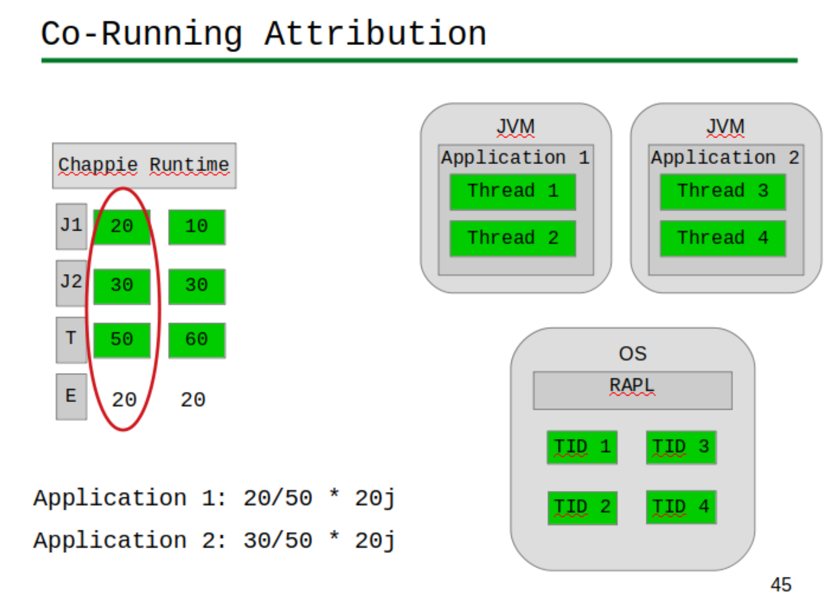
Summary



Application-Level Energy Accounting
Energy (J)
600
400
200
0
App A
App B
Unattributed A
Unattributed B
avrora avrora
batik batik
avrora batik
avrora jython
Benchmarks
Energy consumed per app as attributed by chappie
Energy consumed per app as read by RAPL
11



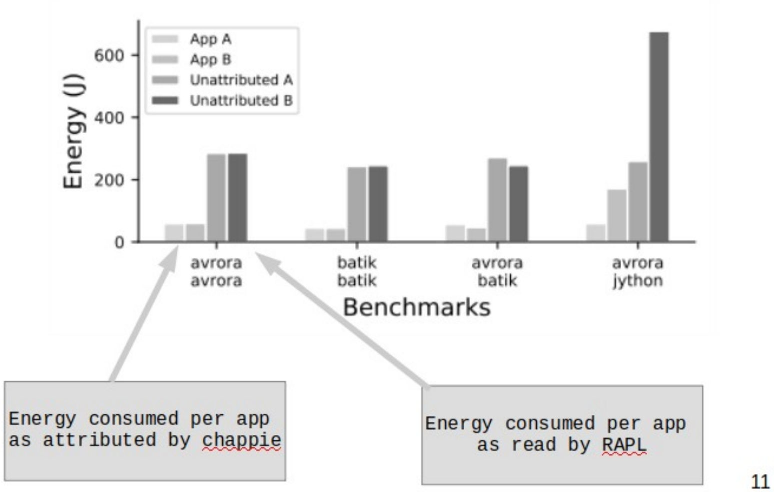
Cross Layer Design
Java VM
Thread Activeness
Method Call Trace
Java VM
Thread Activeness
Method Call Trace
Operating System
RAPL
RAPL
Thread Affinity
Process Jiffies
Chappie Monitor
Chappie Attributor
Domain Accounting
Thread Accounting
Package Accounting
Class Accounting
Method Accounting
Context-Senstive Accounting
21



Co-Running Attribution
Chappie Runtime
J1 20 10
J2 30 30
T 50 60
E 20 20
JVM
Application 1
Thread 1
Thread 2
JVM
Application 2
Thread 3
Thread 4
OS
RAPL
TID 1
TID 3
TID 2
TID 4
Application 1: 20/50 * 20j
Application 2: 30/50 * 20j
45

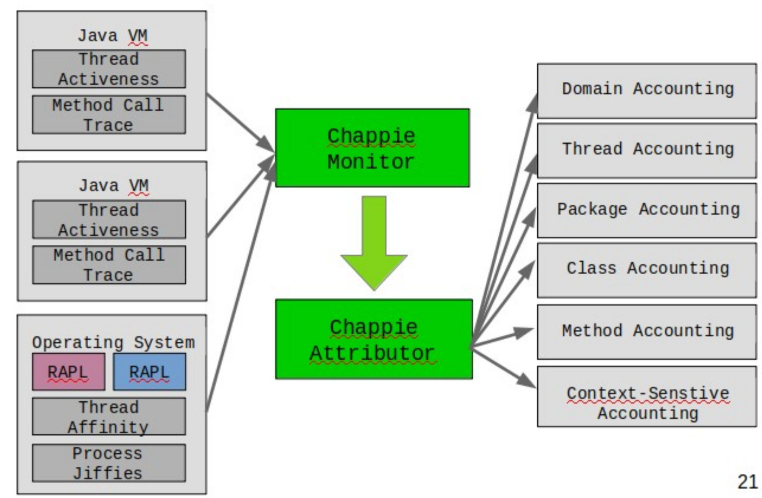
Summary

Application-Level Energy Accounting



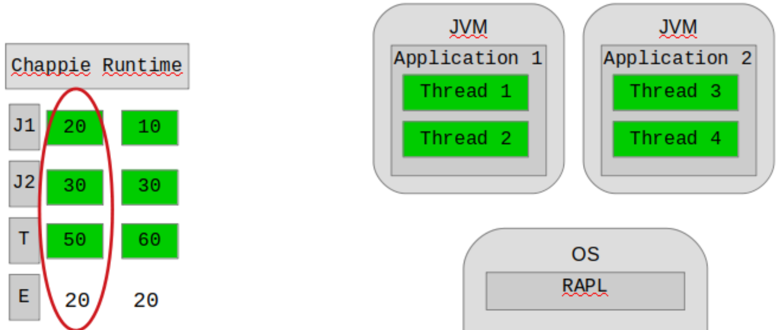
11

Cross Layer Design



21

Co-Running Attribution

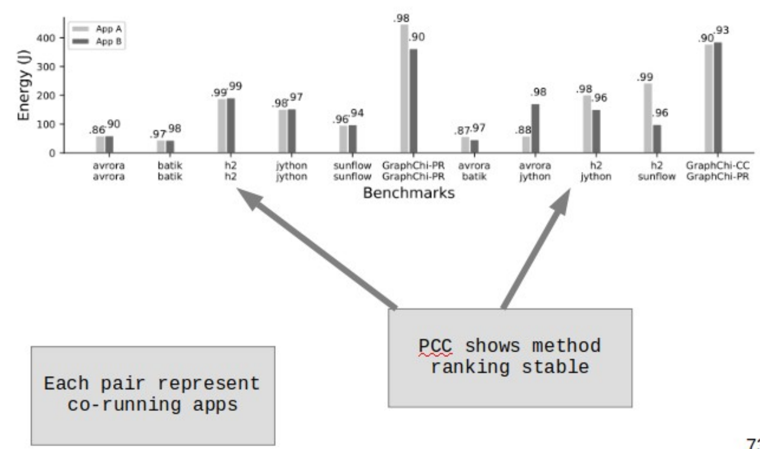


Application 1: 20/50 * 20j

Application 2: 30/50 * 20j

45

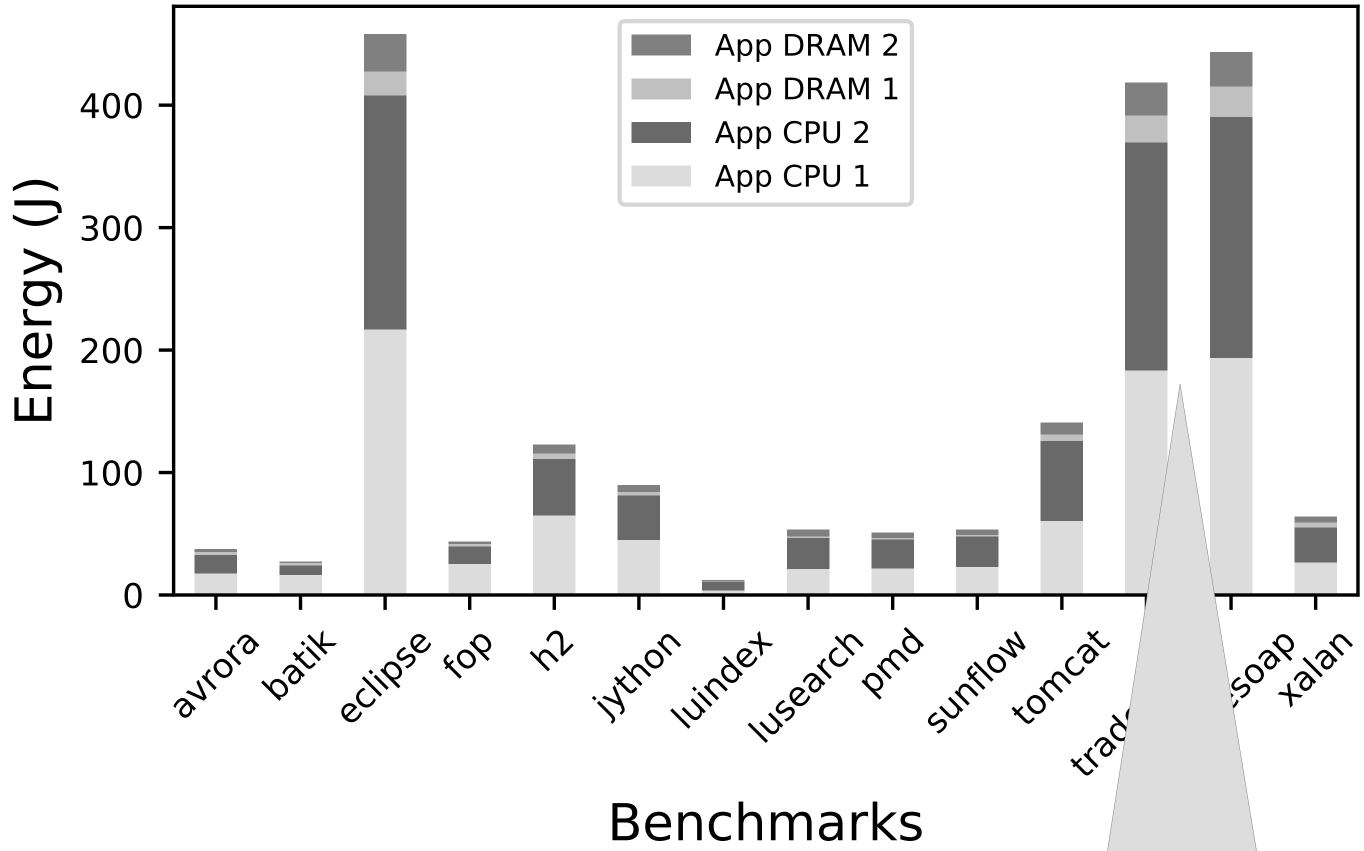
Co-Running Applications



73

Questions?

System Accounting



Energy Attributed Per-Domain

OS schedules threads consistently across sockets

Related Work

- Hardware and OS Energy Accounting
  - iCount (IPSN '08)
- Application-level Profiling and Energy Management
  - JouleTracke (DAC '11)
  - PowerScope (WMCSA '99)
- Energy Analysis
  - Tiwari (VLSI ‘96)
  - Greec (SCOPES ‘15)
  - Jayaseelen (RTAS ‘06)
  - Hao (GREENS ‘12)
- Runtime-Centric or Cross-Layer Approaches
  - Krishnan (Perf. Eval. Rev. ‘11)
  - Bertran (Future Generation Computer System ‘12)
  - JouleGuard (SOSP '15)
  - ARO (MobiSys ‘11)