WattsKit, Software-Defined Power Monitoring of Distributed Systems

CCGrid'17: Performance Modeling and Evaluation (Session 18B)

17th May, 2017 – 10:55

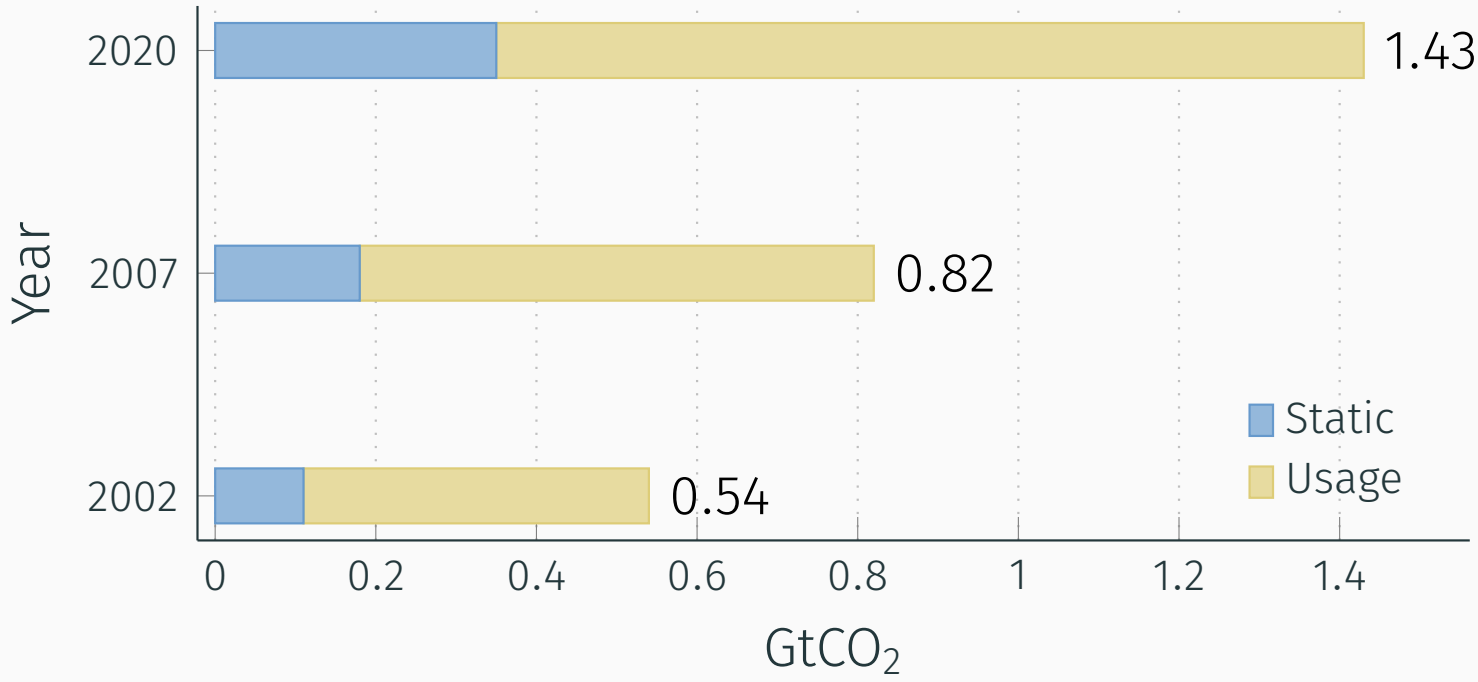
Authors:

Maxime	Colmant	ADEME / University Lille 1 / INRIA
Pascal	Felber	University Neuchâtel
Romain	Rouvoy	University Lille 1 / INRIA / IUF
Lionel	Seinturier	University Lille 1 / INRIA / IUF

Table of contents

Introduction

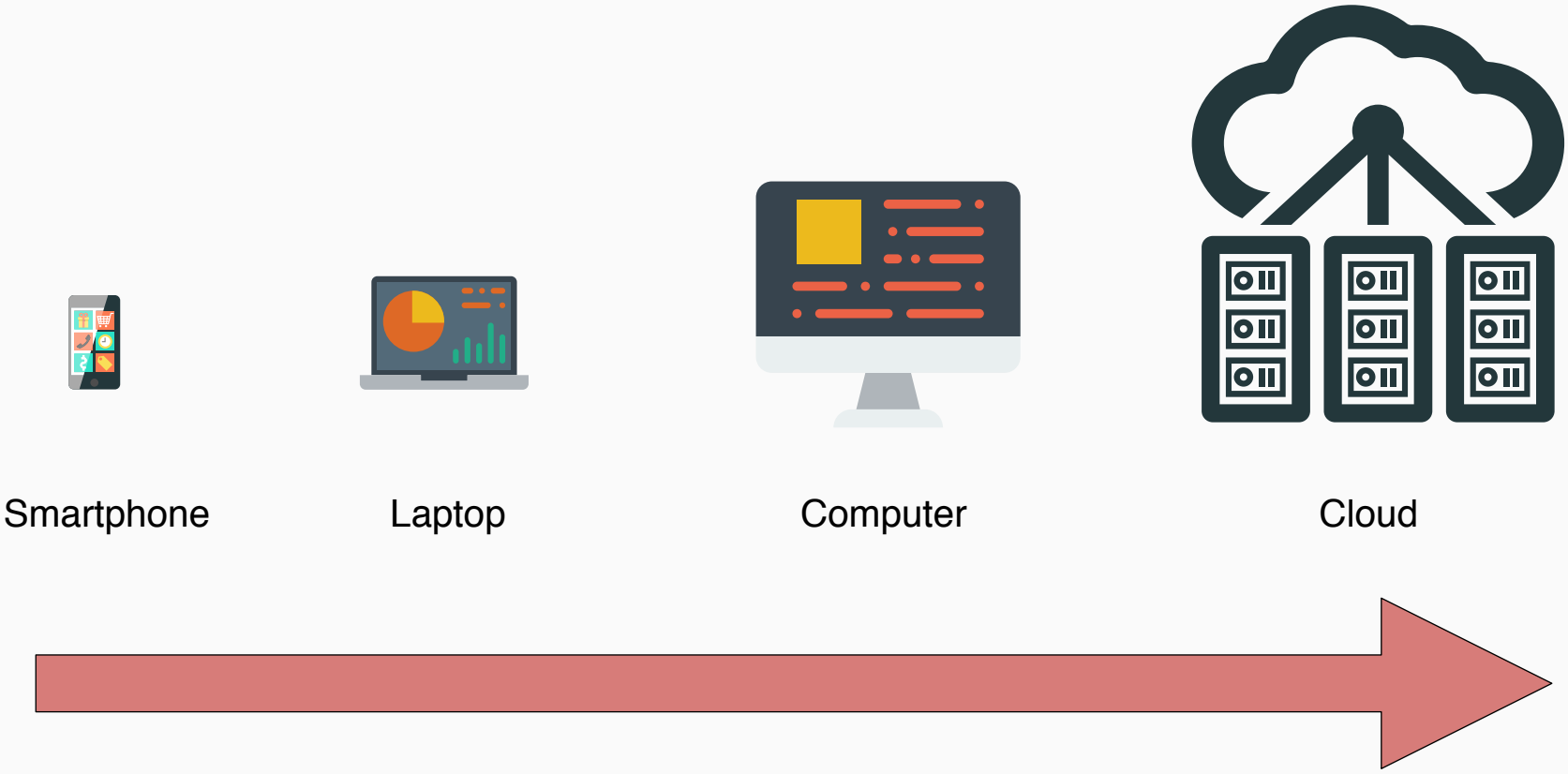
The Global ICT[1] Footprint[2]



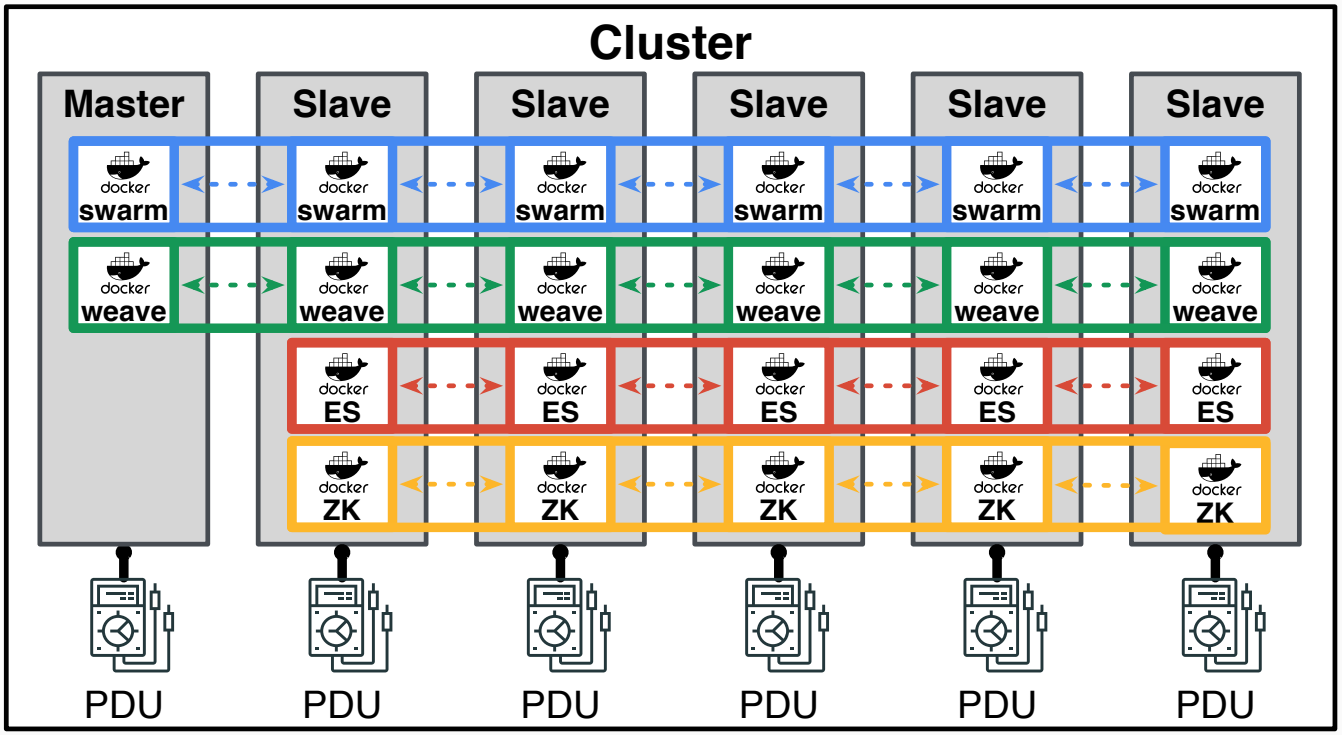
[1] Information and Communications Technology

[2] The Climate Group. *SMART 2020: Enabling the low carbon economy in the information age*. 2008.

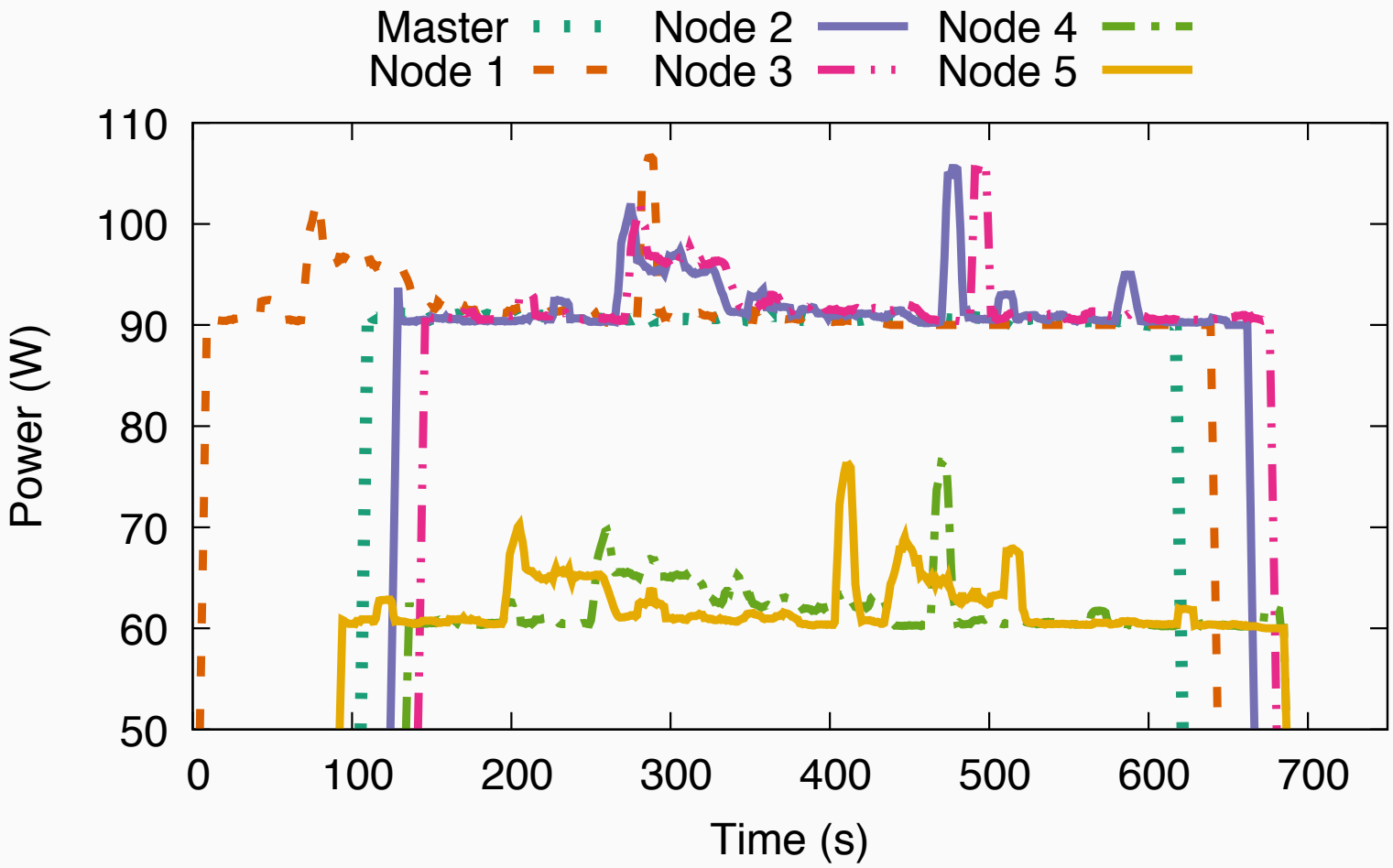
Multi-core CPU Architectures Are Everywhere!



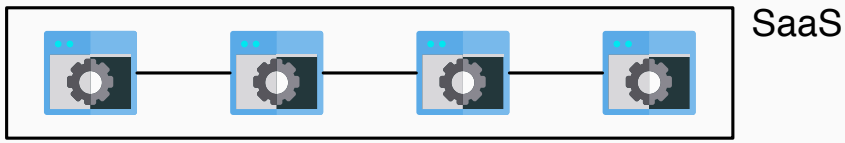
Case Study


Cluster
Master
Slave
Slave
Slave
Slave
Slave
docker swarm
docker weave
docker ES
docker ZK
PDU
PDU
PDU
PDU
PDU
PDU

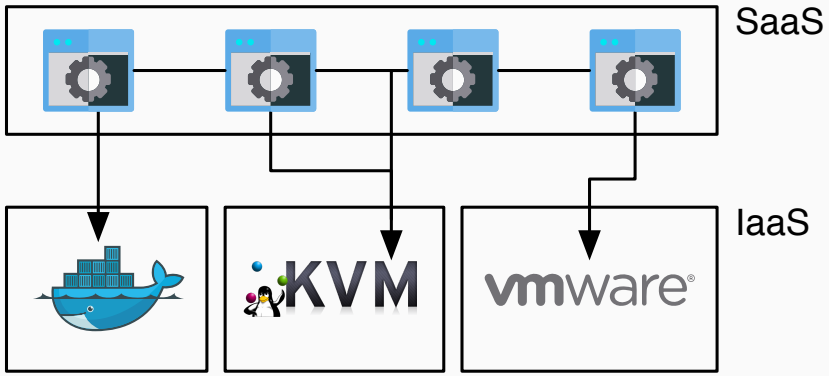
Case Study


Master
Node 1
Node 2
Node 3
Node 4
Node 5
Power (W)
110
100
90
80
70
60
50
0
100
200
300
400
500
600
700
Time (s)

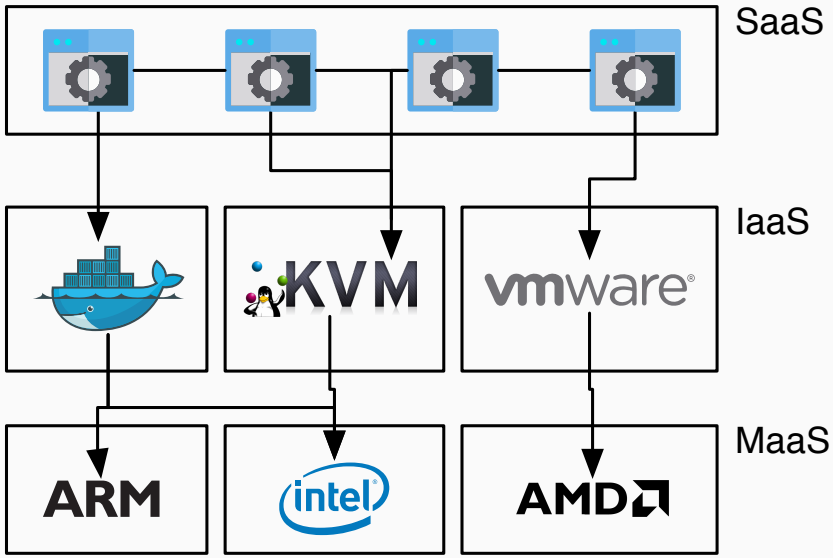
What is a Multi-Core Software System?



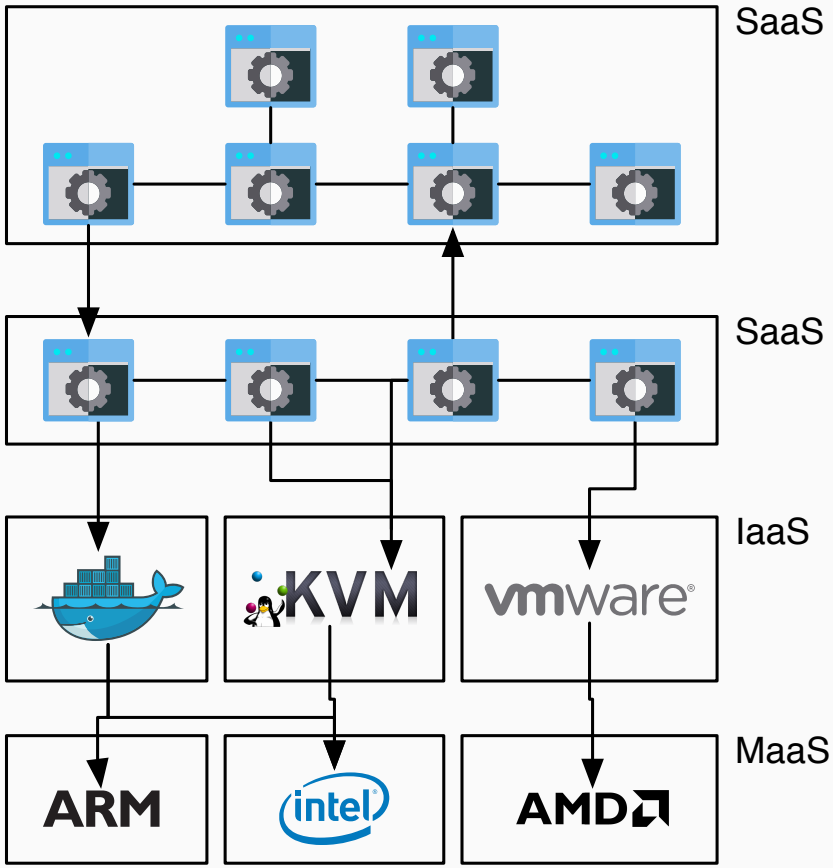
What is a Multi-Core Software System?



What is a Multi-Core Software System?

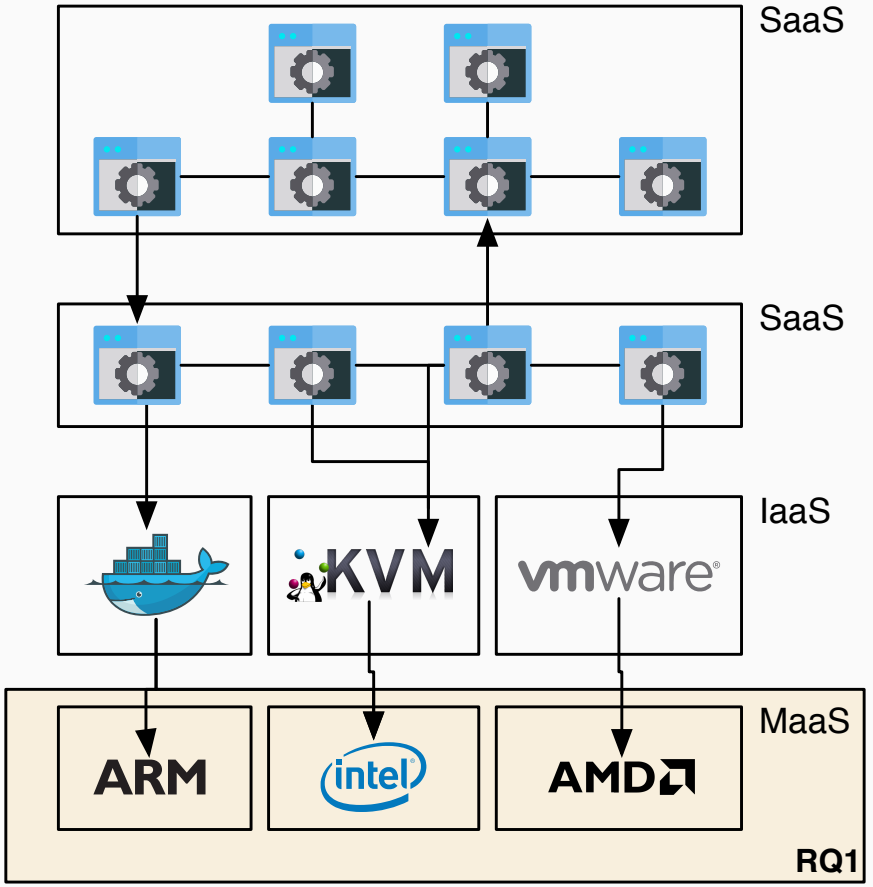


What is a Multi-Core Software System?

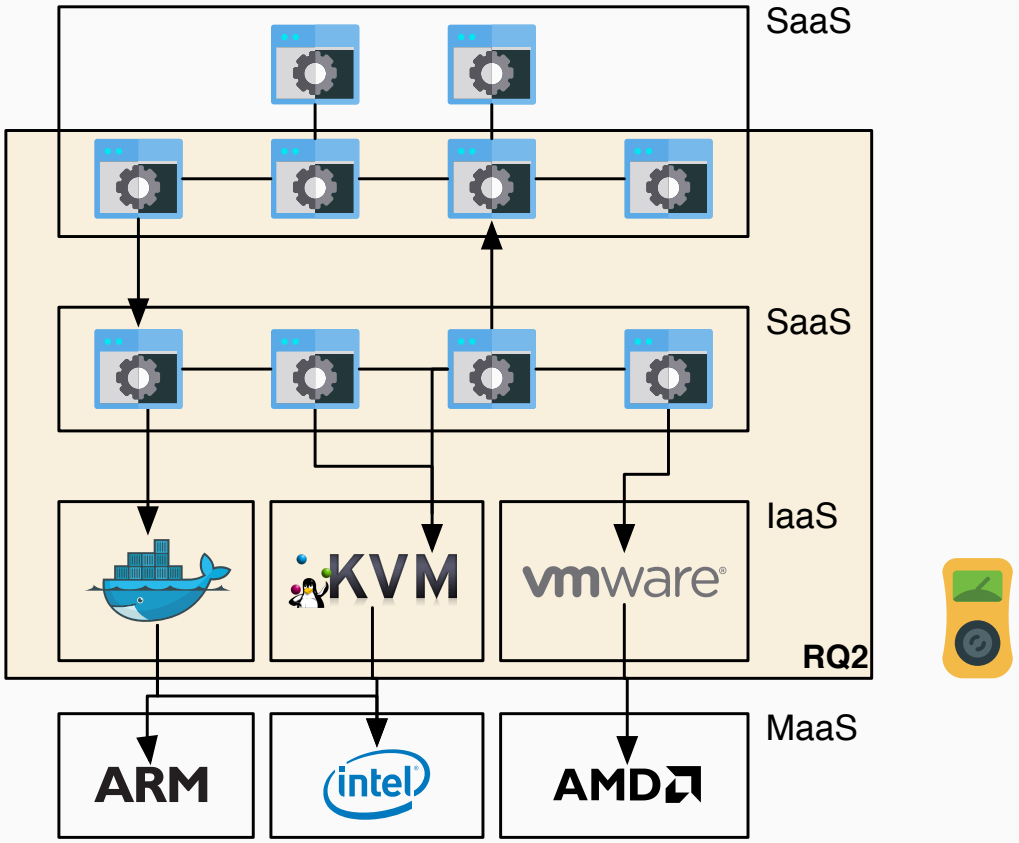


Research Questions

RQ1: Can we model the software power consumption regardless of the underlying architecture?

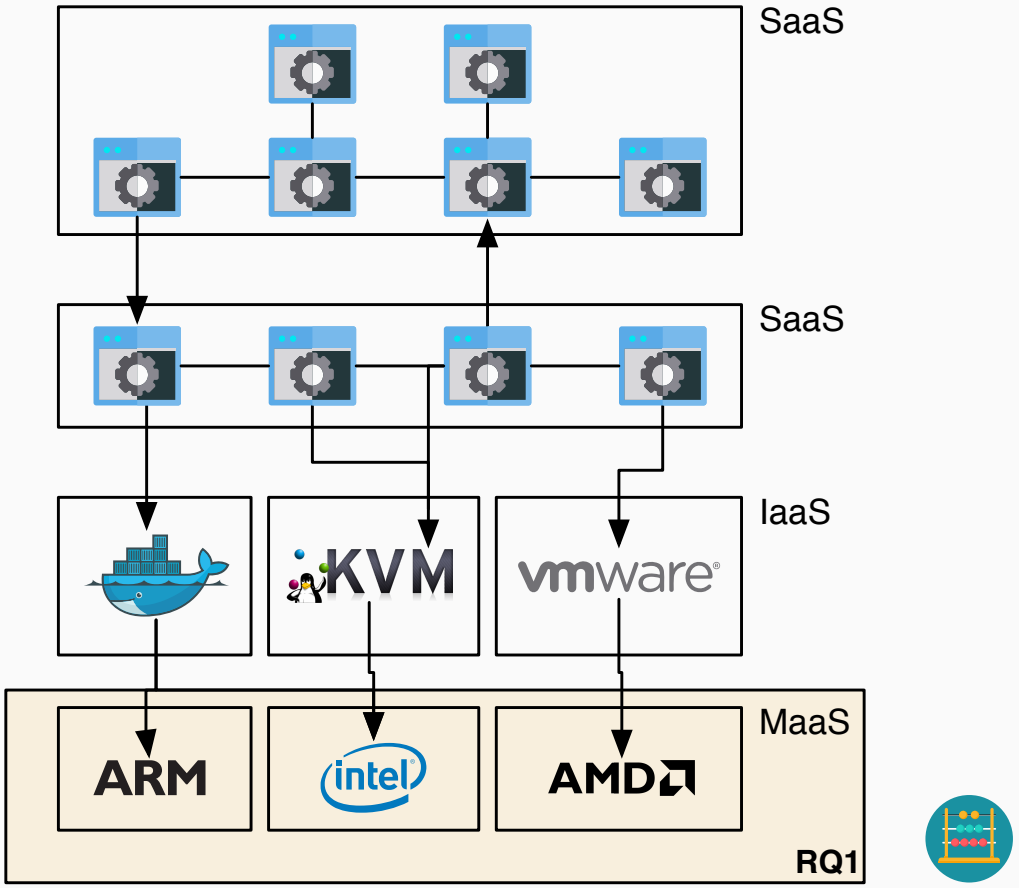


RQ2: Can we propose a uniform view of the service power consumption?

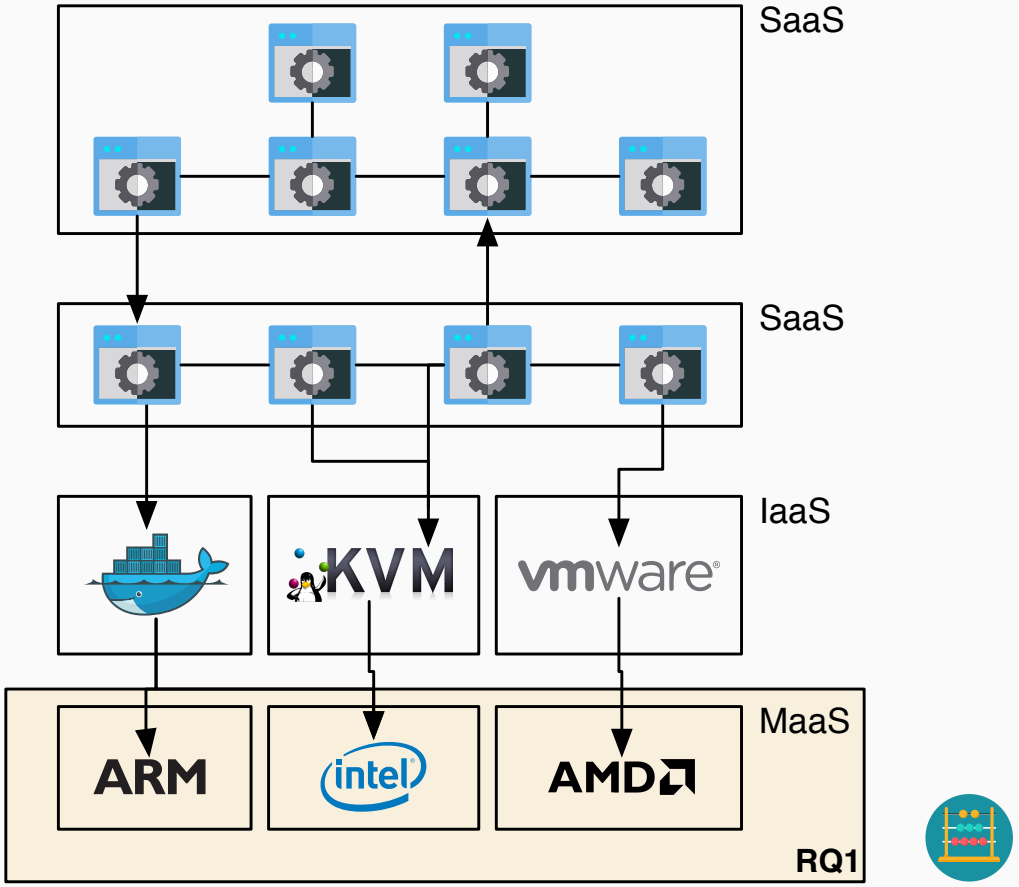


Contributions

RQ1: Can we model the software power consumption regardless of the underlying architecture?



RQ1: Can we model the software power consumption regardless of the underlying architecture?



Learning CPU Power Models

Existing CPU Power Models

Ref.	Processor(s)	Feature(s)	Regression(s)	Benchmarks
[Ber+10]	Core 2 Duo	14 PCs regrouped by component	multiple linear by component	*sampl.:* μ-benchs *eval.:* SPEC CPU 06
[Col+15]	Xeon W3520 & i3 2120	non-halted cycles reference cycles	polynomial	*sampl.:* stress *eval.:* PARSEC, SPECjbb
[CM05]	XScale PXA255	5 PCs	multiple linear	*eval.:* SPEC CPU 00, Java CDC/CLDC
[Dol+15]	Xeon E3-1275	3 PCs HW sensors	linear	*sampl.:* linpack, stream, iperf, IOR *eval.:* Quantum Espresso
[ERK06]	Turion, Itanium 2	HW sensors	multiple linear	*sampl.:* Gamut *eval.:* SPECs, Matrix, Stream
[IM03]	Pentium 4	15 PCs	multiple linear	*eval.:* μ-benchs, AbiWord, Mozilla, Gnumeric
[RRK08]	Core 2 Duo & Xeon, Itanium 2, Turion	HW sensors PCs	multiple linear	*sampl.:* calibration suite *eval.:* SPECs, stream, Nsort
[Yan+14]	Xeon E5620 & E7530	7 components 91 preselected	support vector	*sampl.:* NPB, IOzone, CacheBench *eval.:* SPEC CPU 06, IOzone
[Zha+14]	Sandy Bridge	non-halted cycles	linear	*eval.:* Google, SPEC CPU 06
???	ARM	???	???	???

Only for Intel or AMD architectures

Existing CPU Power Models

Ref.	Processor(s)	Feature(s)	Regression(s)	Benchmarks
[Ber+10]	Core 2 Duo	14 HPCs regrouped by component	multiple linear by component	*sampl.:* μ-benchs *eval.:* SPEC CPU 06
[Col+15]	Xeon W3520 & i3 2120	non-halted cycles reference cycles	polynomial	*sampl.:* stress *eval.:* PARSEC, SPECjbb
[CM05]	XScale PXA255	5 HPCs	multiple linear	*eval.:* SPEC CPU 00, Java CDC/CLDC
[Dol+15]	Xeon E3-1275	3 HPCs HW sensors	linear	*sampl.:* linpack, stream, iperf, IOR *eval.:* Quantum Espresso
[ERK06]	Turion, Itanium 2	HW sensors	multiple linear	*sampl.:* Gamut *eval.:* SPECs, Matrix, Stream
[IM03]	Pentium 4	15 HPCs	multiple linear	*eval.:* μ-benchs, AbiWord, Mozilla, Gnumeric
[RRK08]	Core 2 Duo & Xeon, Itanium 2, Turion	HW sensors HPCs	multiple linear	*sampl.:* calibration suite *eval.:* SPECs, stream, Nsort
[Yan+14]	Xeon E5620 & E7530	7 components 91 preselected	support vector	*sampl.:* NPB, IOzone, CacheBench *eval.:* SPEC CPU 06, IOzone
[Zha+14]	Sandy Bridge	non-halted cycles	linear	*eval.:* Google, SPEC CPU 06

HW sensors: coarse-grained CPU metrics

Existing CPU Power Models

Ref.	Processor(s)	Feature(s)	Regression(s)	Benchmarks
[Ber+10]	Core 2 Duo	14 HPCs regrouped by component	multiple linear by component	*sampl.:* μ-benchs *eval.:* SPEC CPU 06
[Col+15]	Xeon W3520 & i3 2120	non-halted cycles reference cycles	polynomial	*sampl.:* stress *eval.:* PARSEC, SPECjbb
[CM05]	XScale PXA255	5 HPCs	multiple linear	*eval.:* SPEC CPU 00, Java CDC/CLDC
[Dol+15]	Xeon E3-1275	3 HPCs HW sensors	linear	*sampl.:* linpack, stream, iperf, IOR *eval.:* Quantum Espresso
[ERK06]	Turion, Itanium 2	HW sensors	multiple linear	*sampl.:* Gamut *eval.:* SPECs, Matrix, Stream
[IM03]	Pentium 4	15 HPCs	multiple linear	*eval.:* μ-benchs, AbiWord, Mozilla, Gnumeric
[RRK08]	Core 2 Duo & Xeon, Itanium 2, Turion	HW sensors HPCs	multiple linear	*sampl.:* calibration suite *eval.:* SPECs, stream, Nsort
[Yan+14]	Xeon E5620 & E7530	7 components 91 preselected	support vector	*sampl.:* NPB, IOzone, CacheBench *eval.:* SPEC CPU 06, IOzone
[Zha+14]	Sandy Bridge	non-halted cycles	linear	*eval.:* Google, SPEC CPU 06

HPCs: fine-grained CPU metrics

Existing CPU Power Models

Ref.	Processor(s)	Feature(s)	Regression(s)	Benchmarks
[Ber+10]	Core 2 Duo	14 HPCs regrouped by component	multiple linear by component	*sampl.:* μ-benchs *eval.:* SPEC CPU 06
[Col+15]	Xeon W3520 & i3 2120	non-halted cycles reference cycles	polynomial	*sampl.:* stress *eval.:* PARSEC, SPECjbb
[CM05]	XScale PXA255	5 HPCs	multiple linear	*eval.:* SPEC CPU 00, Java CDC/CLDC
[Dol+15]	Xeon E3-1275	3 HPCs HW sensors	linear	*sampl.:* linpack, stream, iperf, IOR *eval.:* Quantum Espresso
[ERK06]	Turion, Itanium 2	HW sensors	multiple linear	*sampl.:* Gamut *eval.:* SPECs, Matrix, Stream
[IM03]	Pentium 4	15 HPCs	multiple linear	*eval.:* μ-benchs, AbiWord, Mozilla, Gnumeric
[RRK08]	Core 2 Duo & Xeon, Itanium 2, Turion	HW sensors HPCs	multiple linear	*sampl.:* calibration suite *eval.:* SPECs, stream, Nsort
[Yan+14]	Xeon E5620 & E7530	7 components 91 preselected	support vector	*sampl.:* NPB, IOzone, CacheBench *eval.:* SPEC CPU 06, IOzone
[Zha+14]	Sandy Bridge	non-halted cycles	linear	*eval.:* Google, SPEC CPU 06

Power models are mostly linear

Existing CPU Power Models

Ref.	Processor(s)	Feature(s)	Regression(s)	Benchmarks
[Ber+10]	Core 2 Duo	14 HPCs regrouped by component	multiple linear by component	*sampl.:* μ-benchs *eval.:* SPEC CPU 06
[Col+15]	Xeon W3520 & i3 2120	non-halted cycles reference cycles	polynomial	*sampl.:* stress *eval.:* PARSEC, SPECjbb
[CM05]	XScale PXA255	5 HPCs	multiple linear	*eval.:* SPEC CPU 00, Java CDC/CLDC
[Dol+15]	Xeon E3-1275	3 HPCs HW sensors	linear	*sampl.:* linpack, stream, iperf, IOR *eval.:* Quantum Espresso
[ERK06]	Turion, Itanium 2	HW sensors	multiple linear	*sampl.:* Gamut *eval.:* SPECs, Matrix, Stream
[IM03]	Pentium 4	15 HPCs	multiple linear	*eval.:* μ-benchs, AbiWord, Mozilla, Gnumeric
[RRK08]	Core 2 Duo & Xeon, Itanium 2, Turion	HW sensors HPCs	multiple linear	*sampl.:* calibration suite *eval.:* SPECs, stream, Nsort
[Yan+14]	Xeon E5620 & E7530	7 components 91 preselected	support vector	*sampl.:* NPB, IOzone, CacheBench *eval.:* SPEC CPU 06, IOzone
[Zha+14]	Sandy Bridge	non-halted cycles	linear	*eval.:* Google, SPEC CPU 06

Non free or private workloads

Challenges

1. Portability

Challenges

1. Portability
2. Accuracy

Challenges

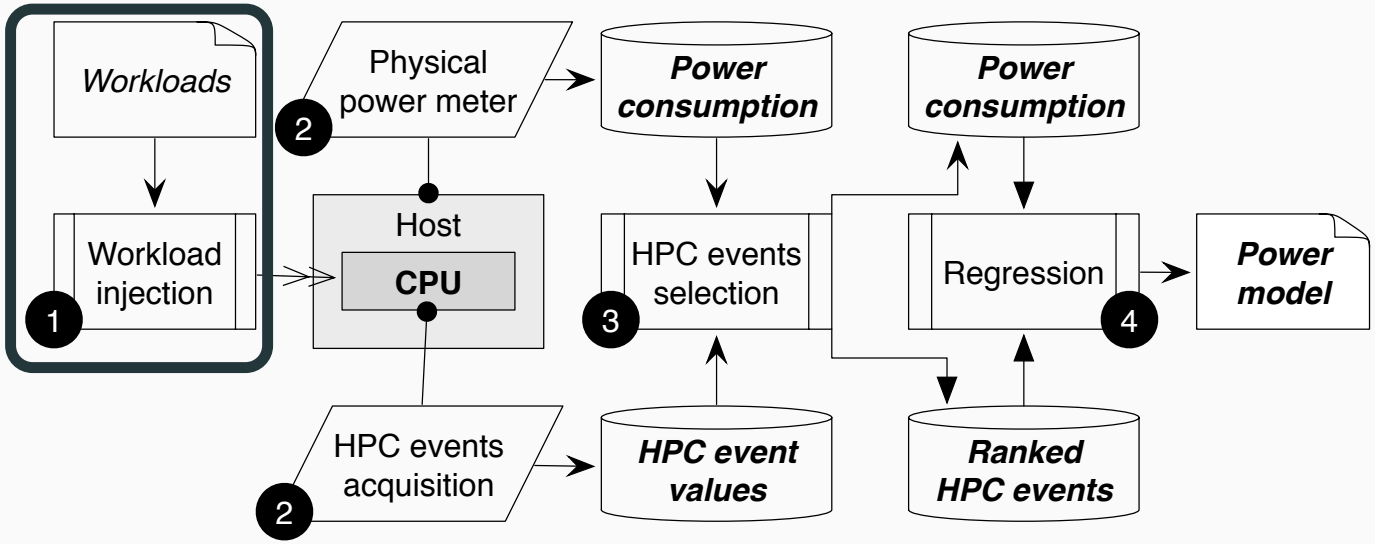
1. Portability
2. Accuracy
3. Reproducibility

Challenges

1. Portability
2. Accuracy
3. Reproducibility

Towards an automatic approach for learning CPU power models

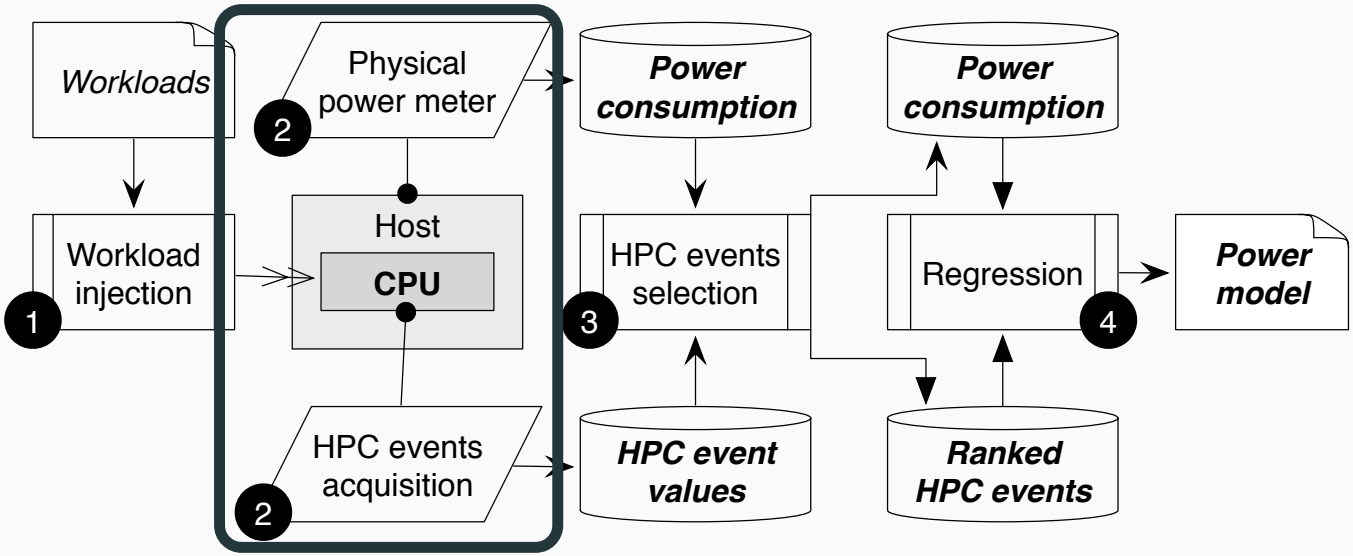
Our approach: Open-Testbed To Automatically Learn Power Models



1. Input workload injection
   - Configurable
   - PARSEC (open-source, multi-threaded)[3]
   - Run several applications (`x264`, `vips`, etc.)

[3] C. Bienia et al. "PARSEC 2.0: A New Benchmark Suite for Chip-Multiprocessors". In: *Proceedings of the 5th Annual Workshop on Modeling, Benchmarking and Simulation*. 2009.

Our approach: Open-Testbed To Automatically Learn Power Models

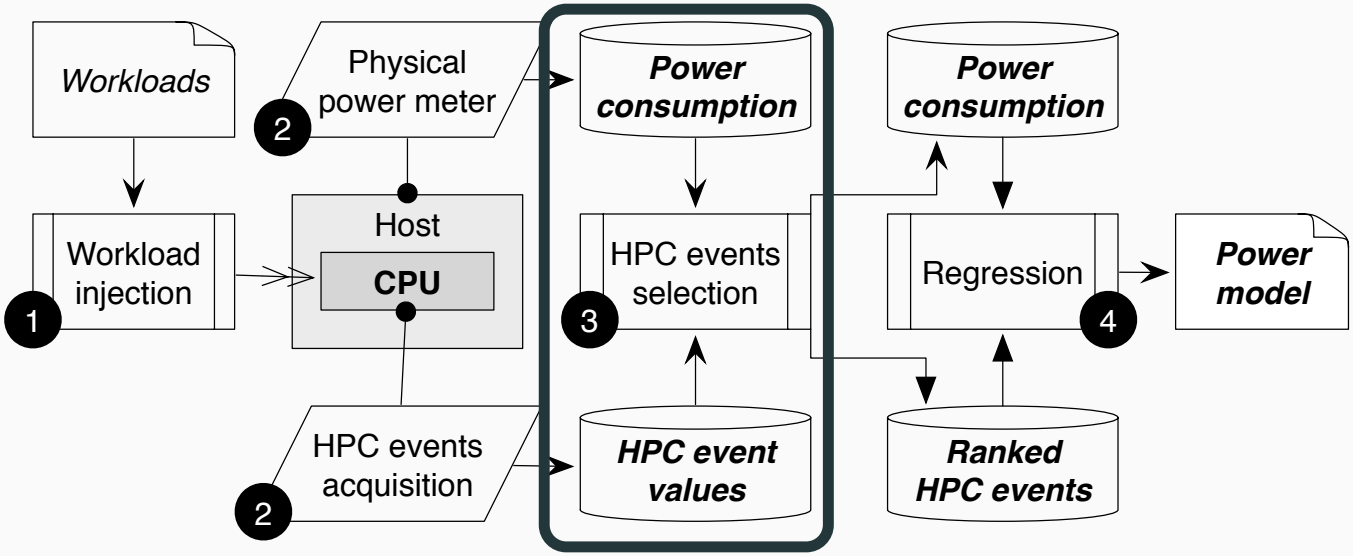


2 Acquisition of raw input metrics

- Automatically explore the high number of the available HPCs (Xeon W3520: 514 HPCs)
- Take care of HPC multiplexing[4]

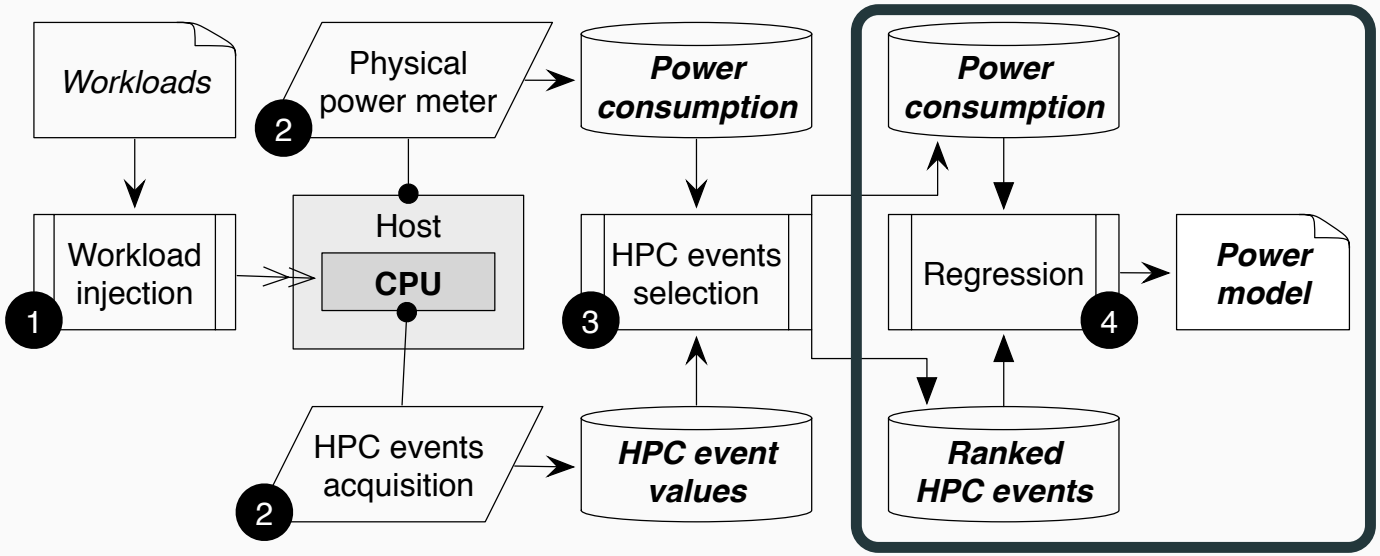
[4] Intel. *Intel 64 and IA-32 Architectures Software Developer's Manual*. 2015.

Our approach: Open-Testbed To Automatically Learn Power Models



3. Selection of relevant HPCs
   - Pearson coefficient (HPC $\Leftrightarrow$ Power)
   - 1^{st} phase: quickly filtering out uncorrelated HPCs (< 0.5) (Xeon W3250: 253 left out)
   - 2^{nd} phase: full sampling for the remaining HPCs

Our approach: Open-Testbed To Automatically Learn Power Models

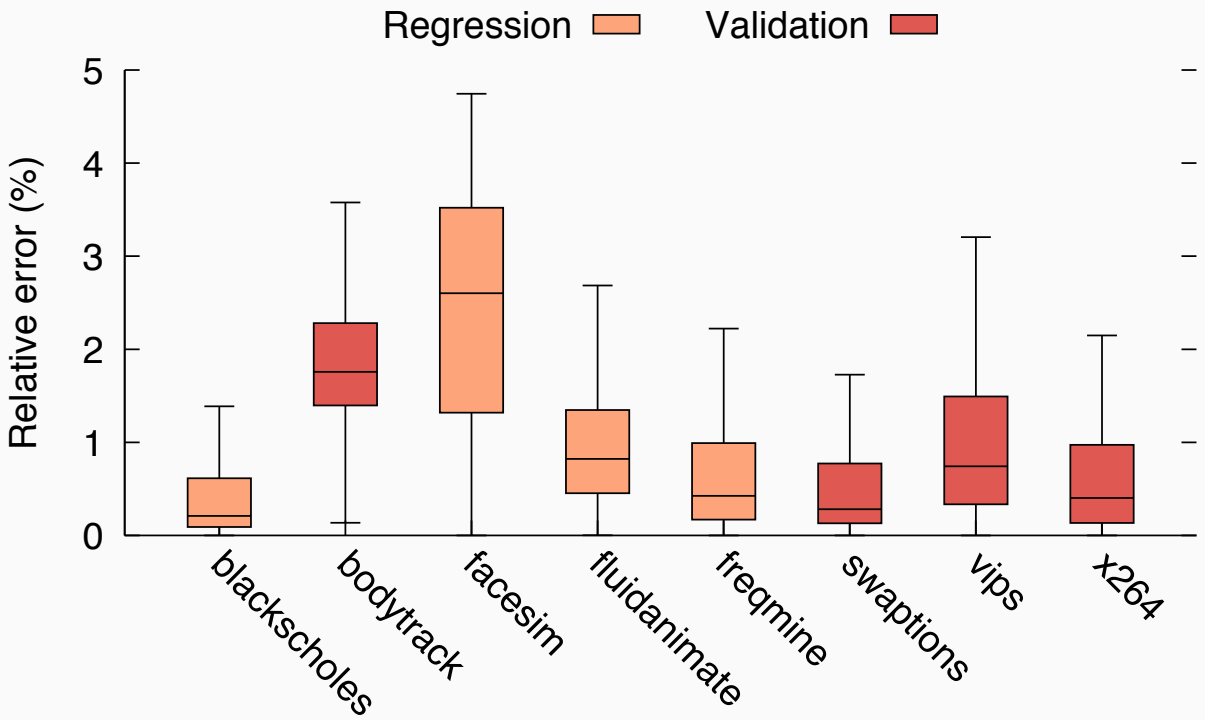


4. Power model inference
   - Minimize the number of HPCs
   - Robust ridge regression (SotA?)

Our approach: Open-Testbed To Automatically Learn Power Models

Relative errors for the PARSEC suite on a Xeon W3520.

$$P_{idle} = 92\ W\ ;\ P_{CPU} = \frac{1.40 \cdot \text{l1i:reads}}{10^8} + \frac{7.29 \cdot \text{lsd:inactive}}{10^9}$$



Summary

- **Portability**

  Beyond SotA: adaptive approach

Summary

- **Portability**

  Beyond SotA: adaptive approach

- **Accuracy**

  Avg. error: 1.35%

Summary

- **Portability**

  Beyond SotA: adaptive approach

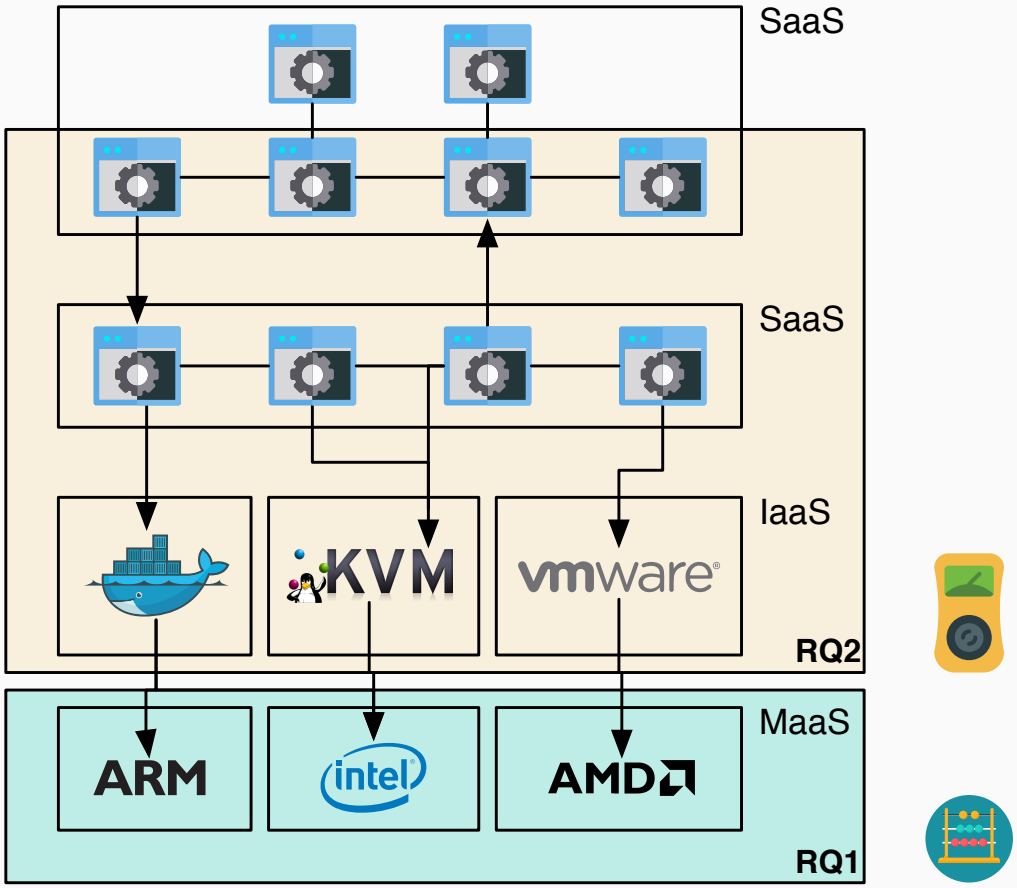
- **Accuracy**

  Avg. error: 1.35%

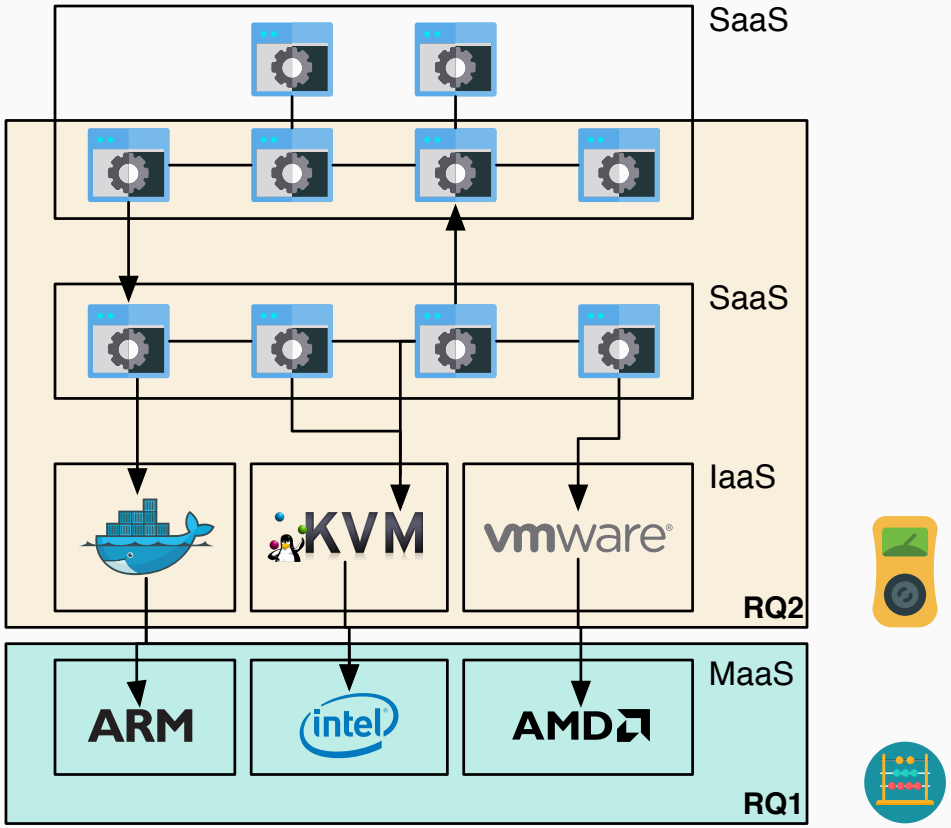
- **Reproducibility**

  Built on open-source workloads

RQ2: Can we propose a uniform view of the service power consumption?



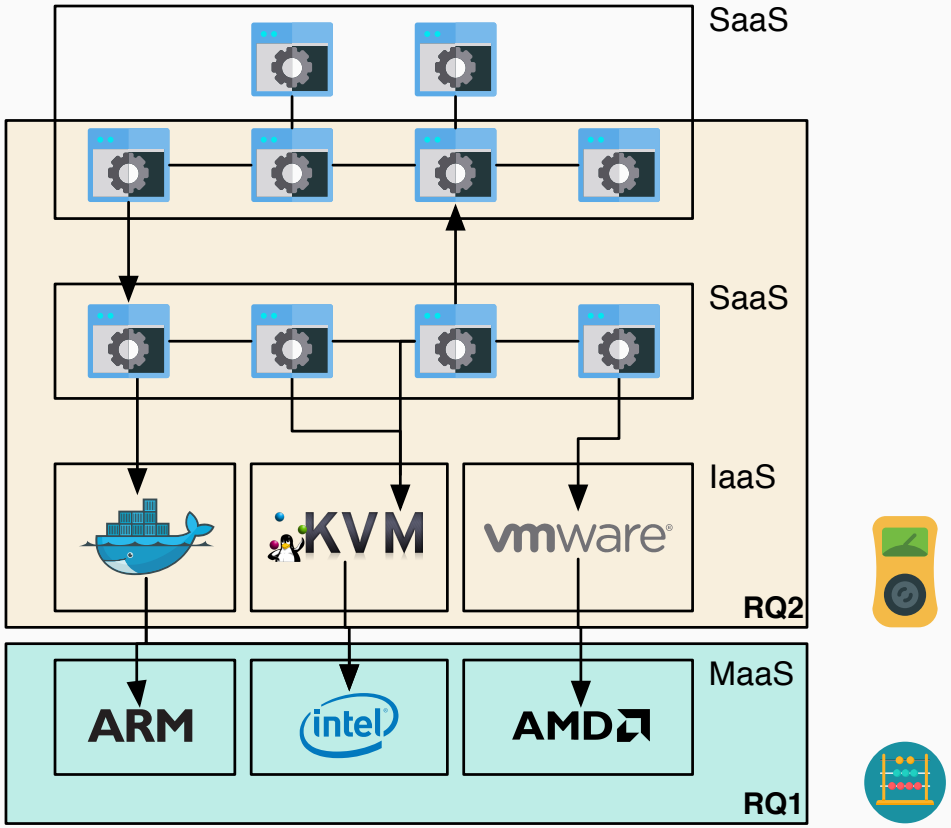
RQ2: Can we propose a uniform view of the service power consumption?



Challenges

1. Native
2. Distributed

RQ2: Can we propose a uniform view of the service power consumption?



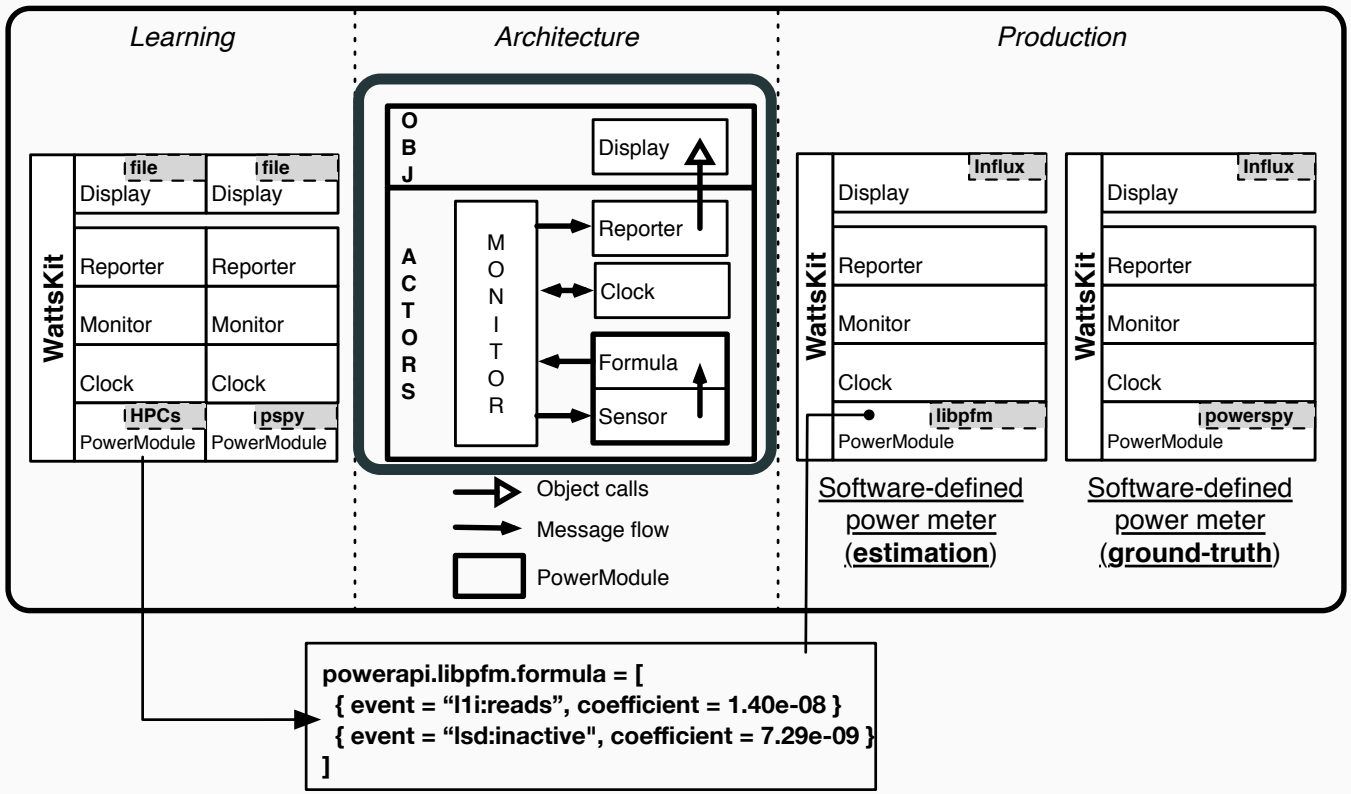
Challenges

1. **Native**
2. Distributed

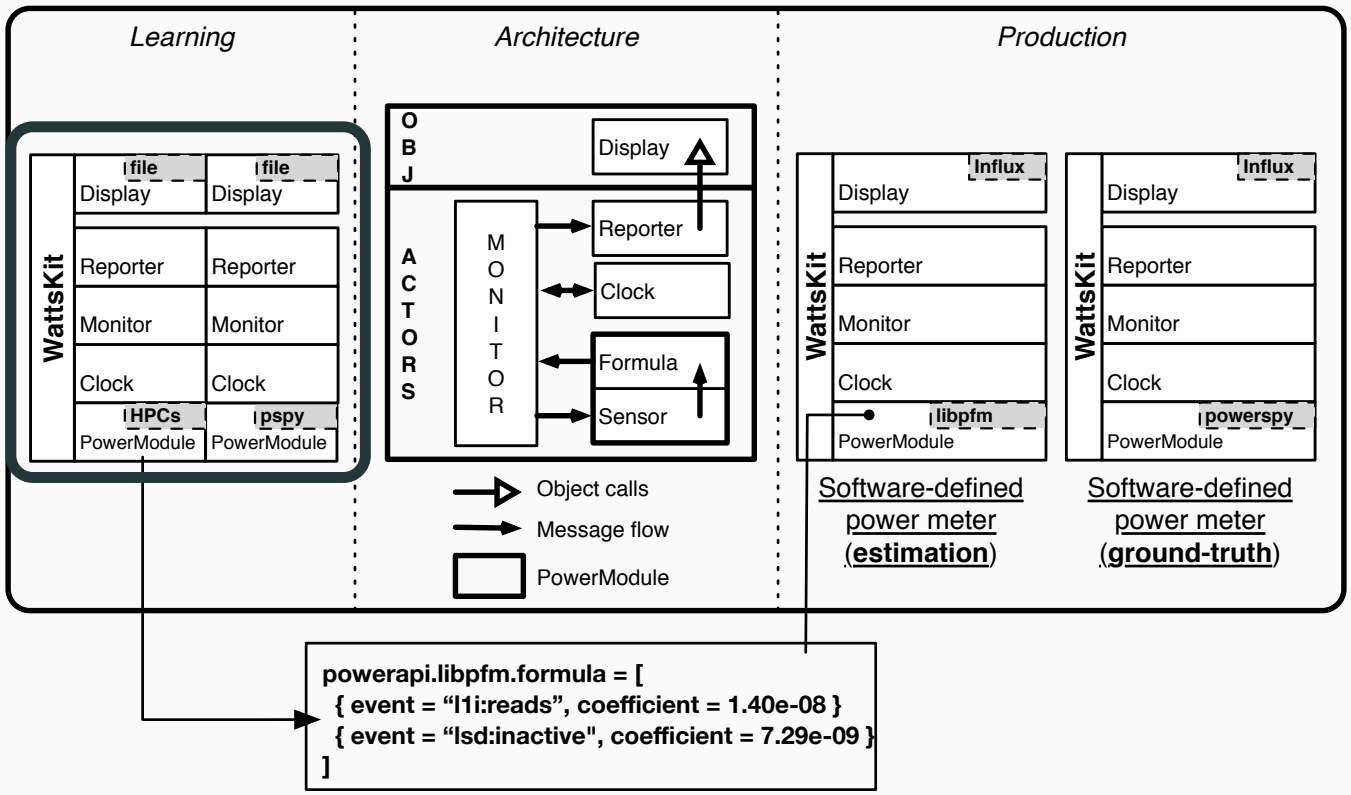
WattsKit: a Toolkit for Building SD Power Meters

- Code freely available: `wattskit.powerapi.org`
  - Scala / Akka
  - LoC: 8.7k
  - Docker
  - AGPLv3

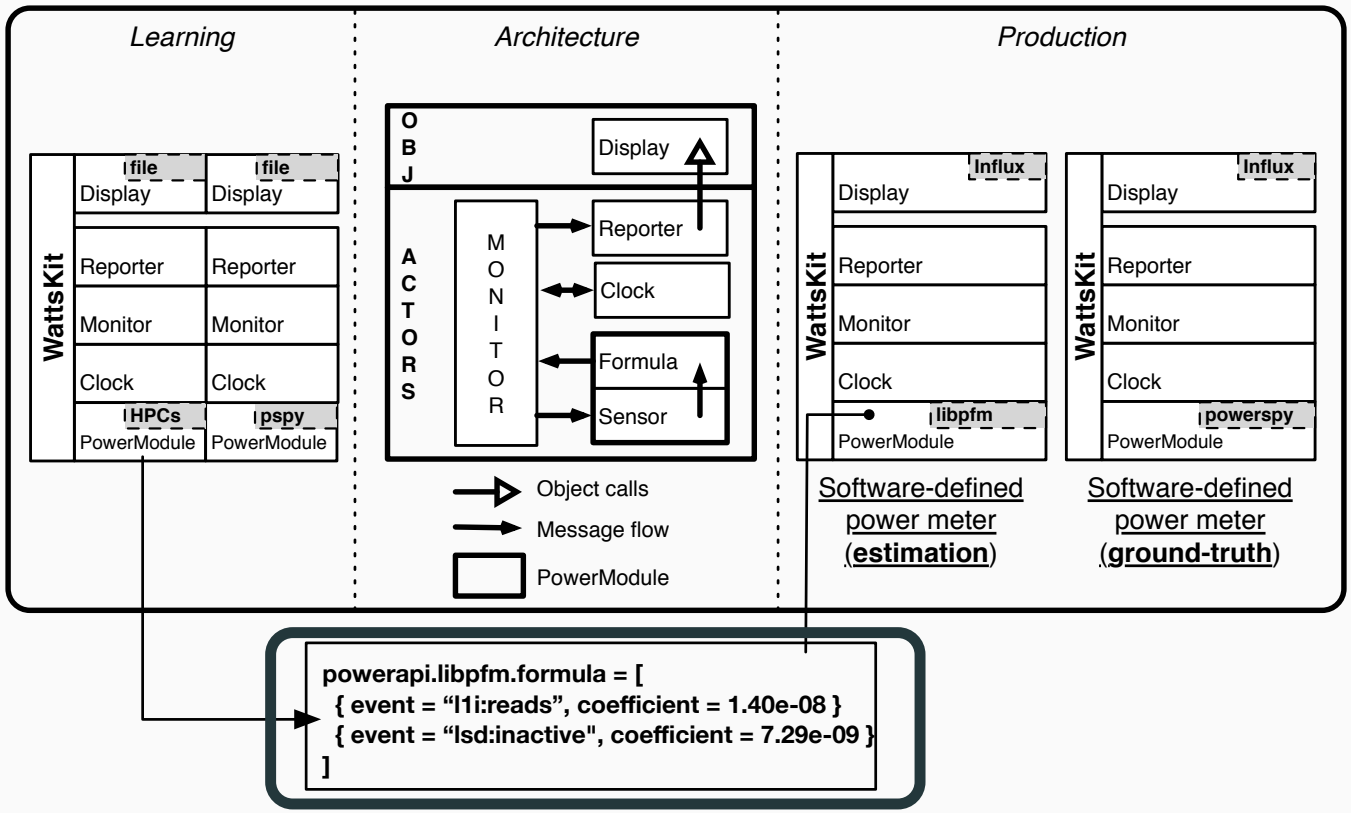
WattsKit's architecture & deployment.



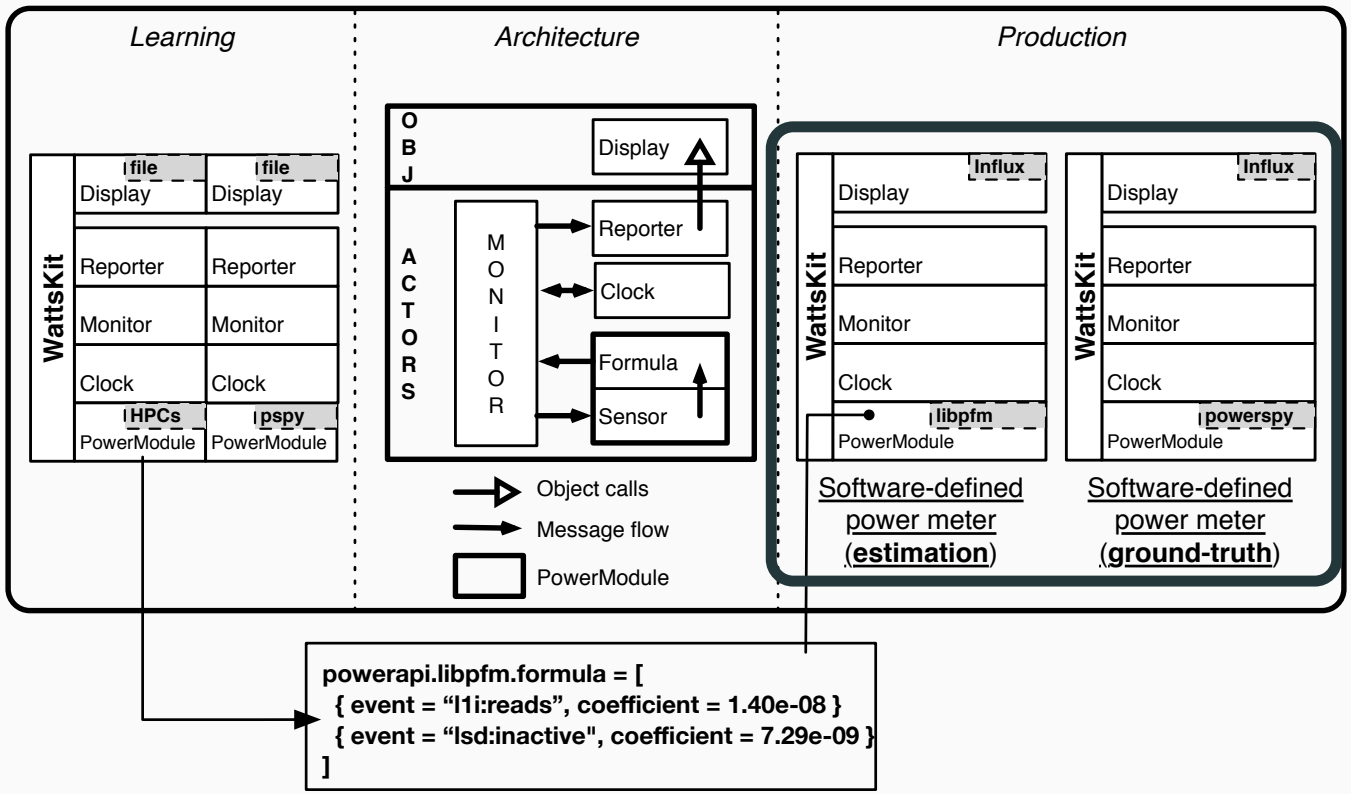
WattsKit's architecture & deployment.



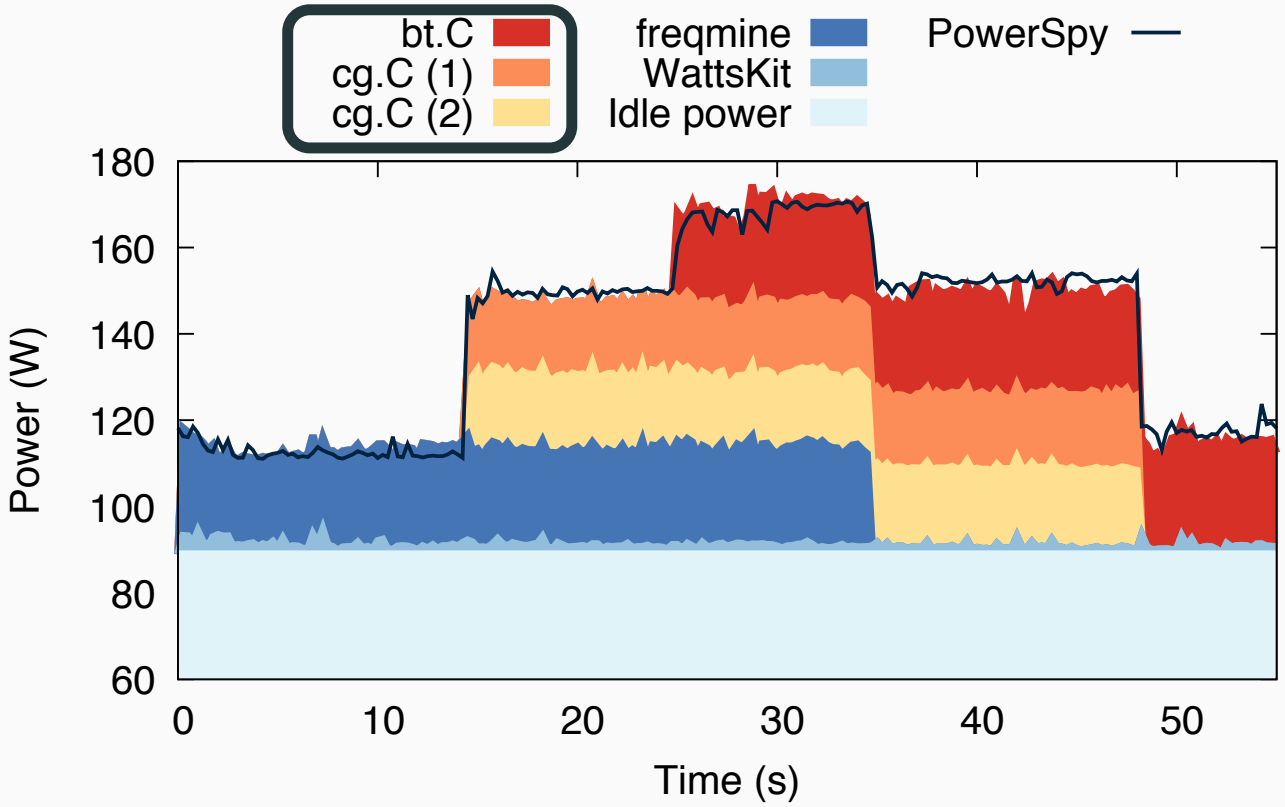
WattsKit's architecture & deployment.



WattsKit's architecture & deployment.

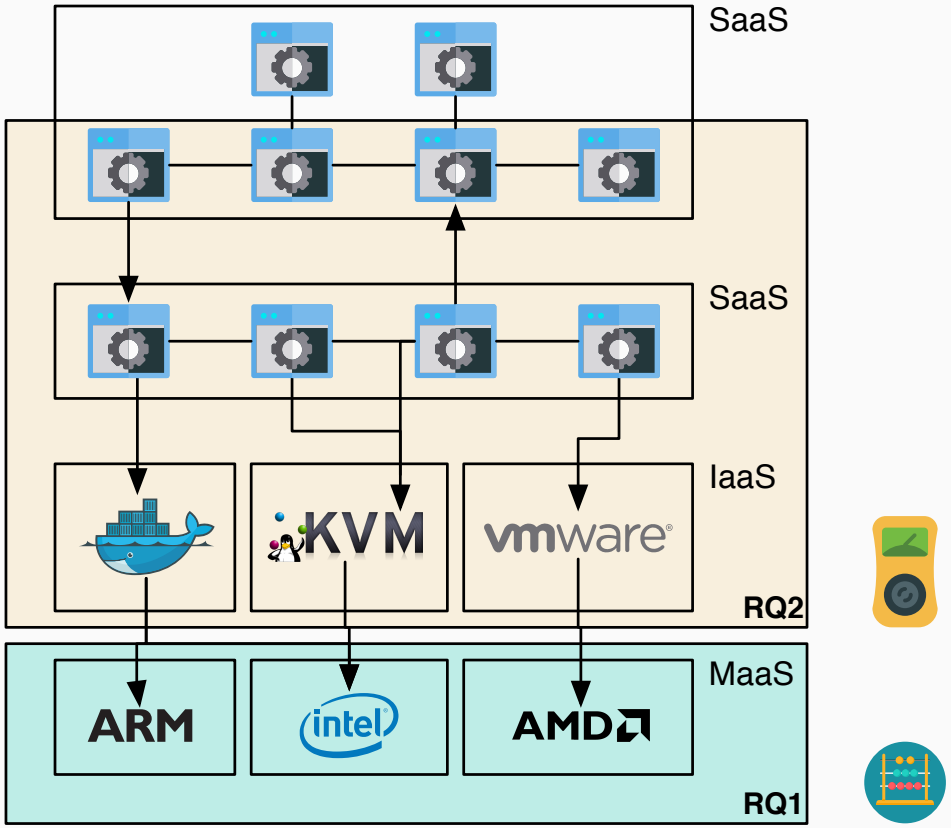


SD Power Meter For Monitoring Concurrent Apps



- On an Intel Xeon W3520
  - Monitoring freq.: 4Hz
  - Avg. error: 2%
  - Low overhead: 2 W

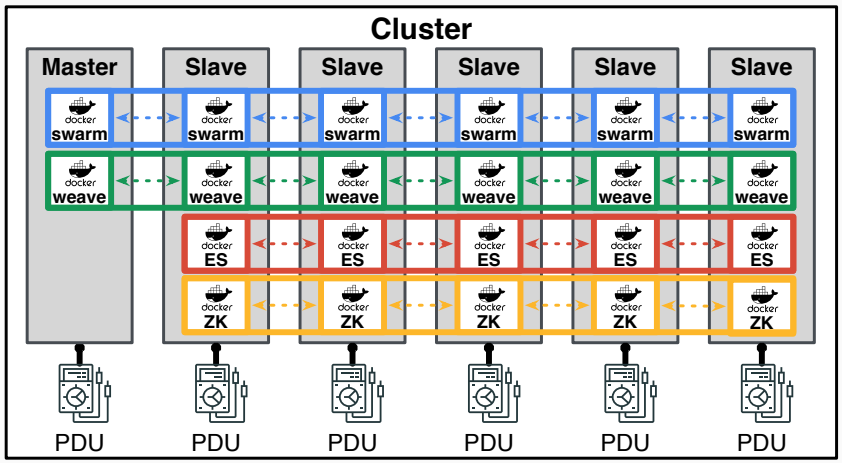
RQ2: Can we propose a uniform view of the service power consumption?

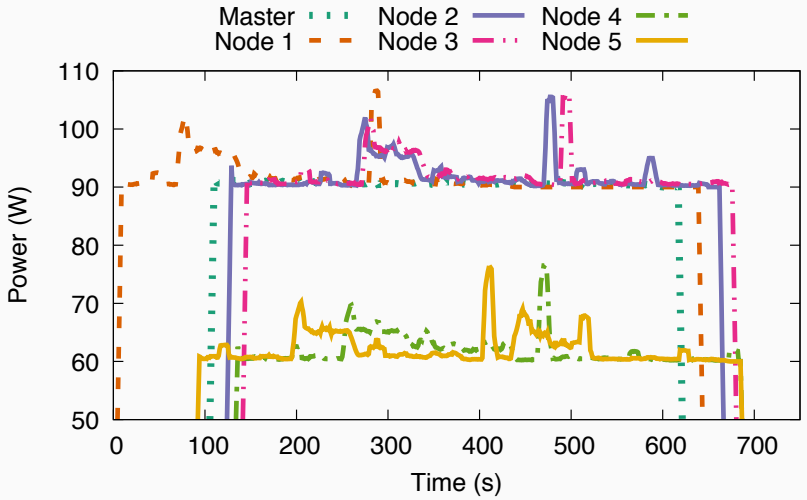


Challenges

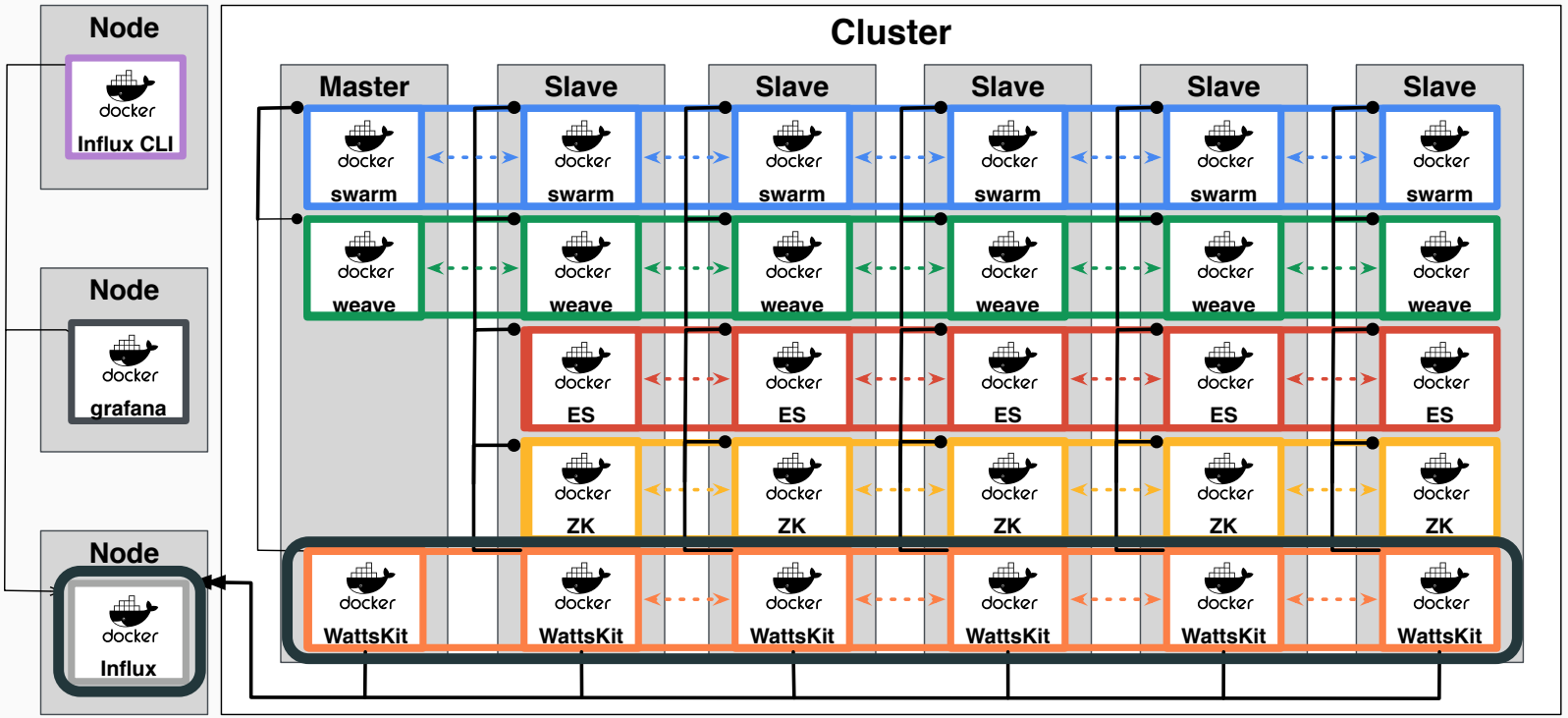
1. Native
2. **Distributed**

Current Coarse-Grained Solutions

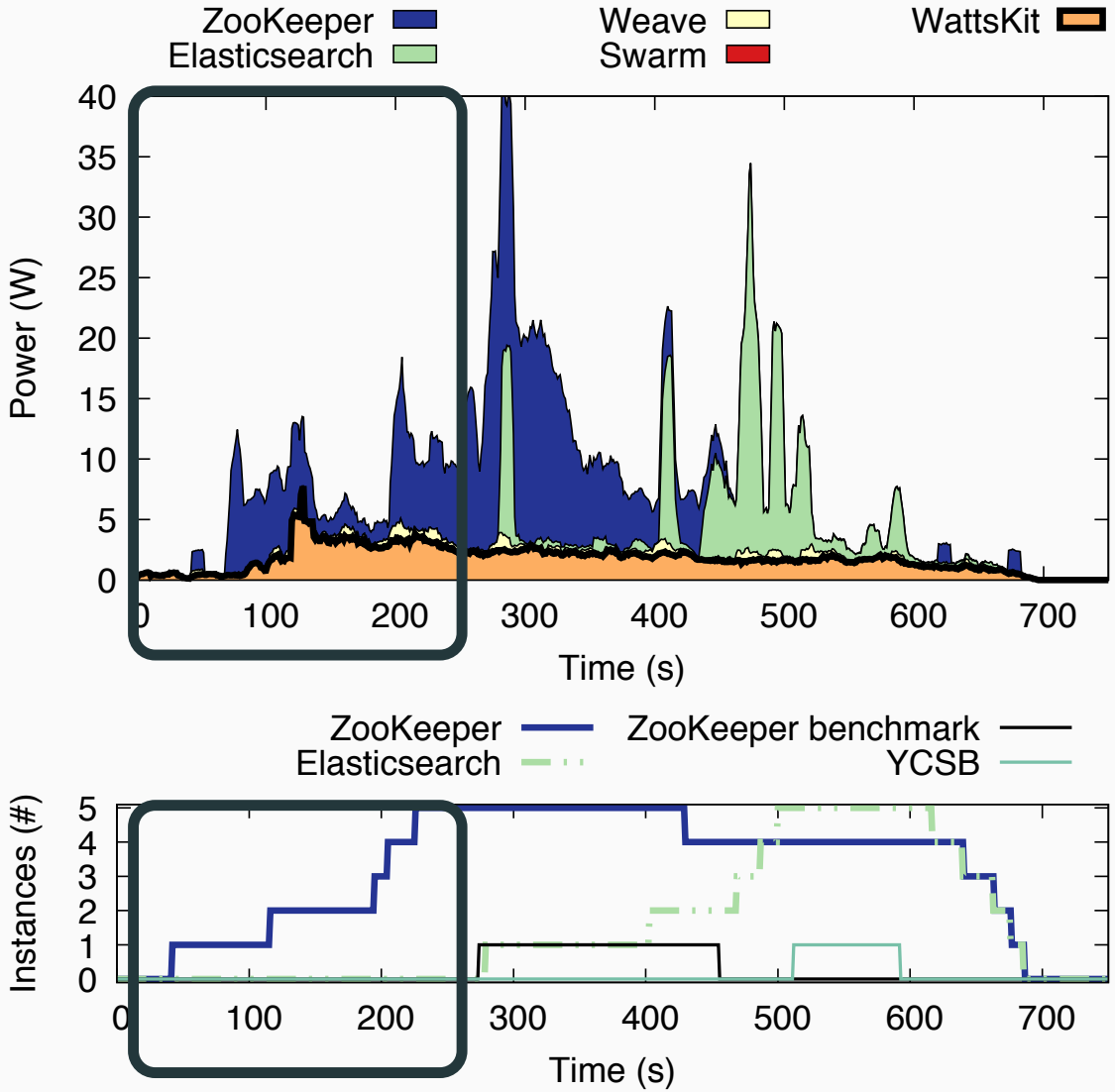




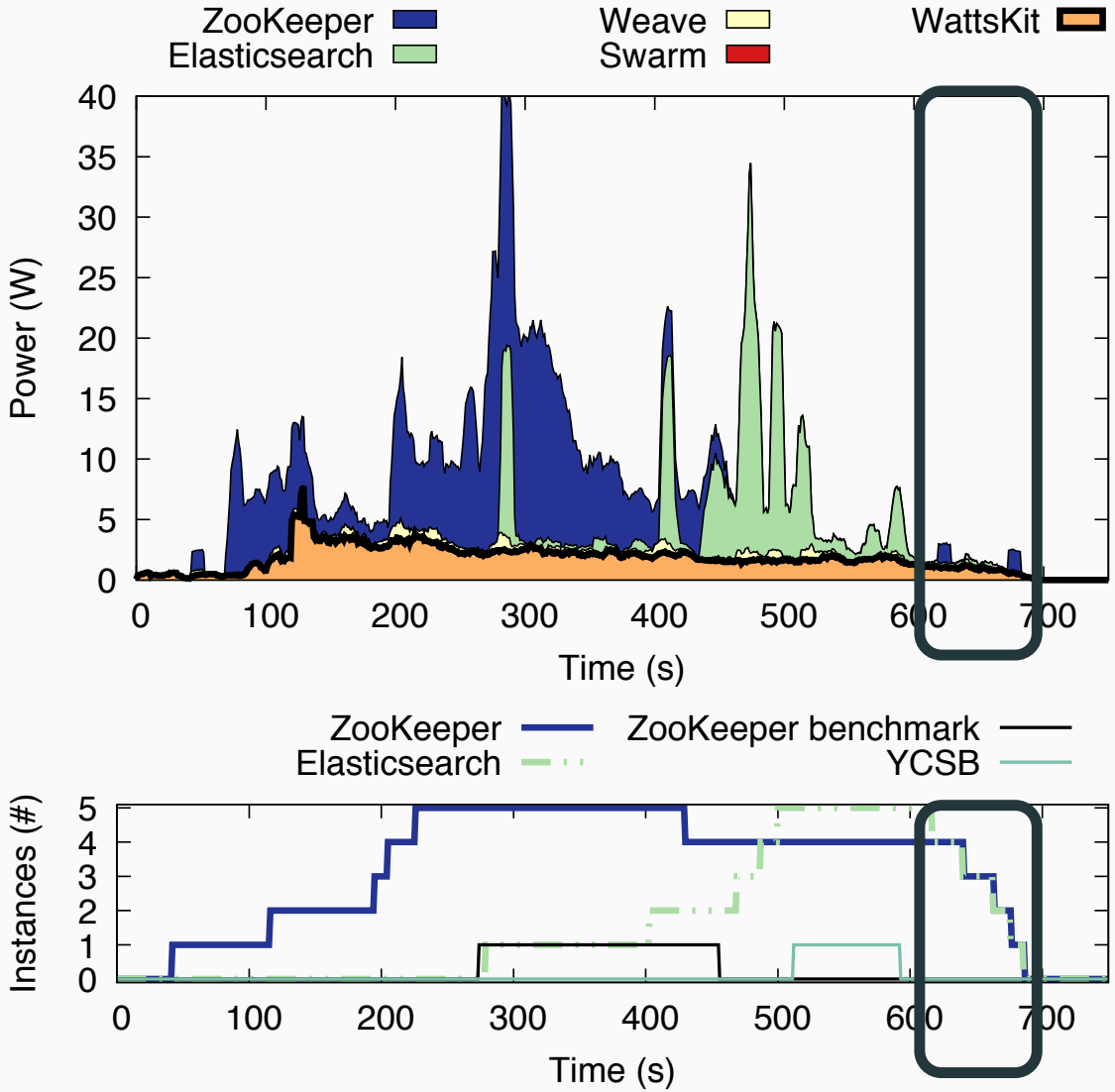
A Service-Level Power Monitoring



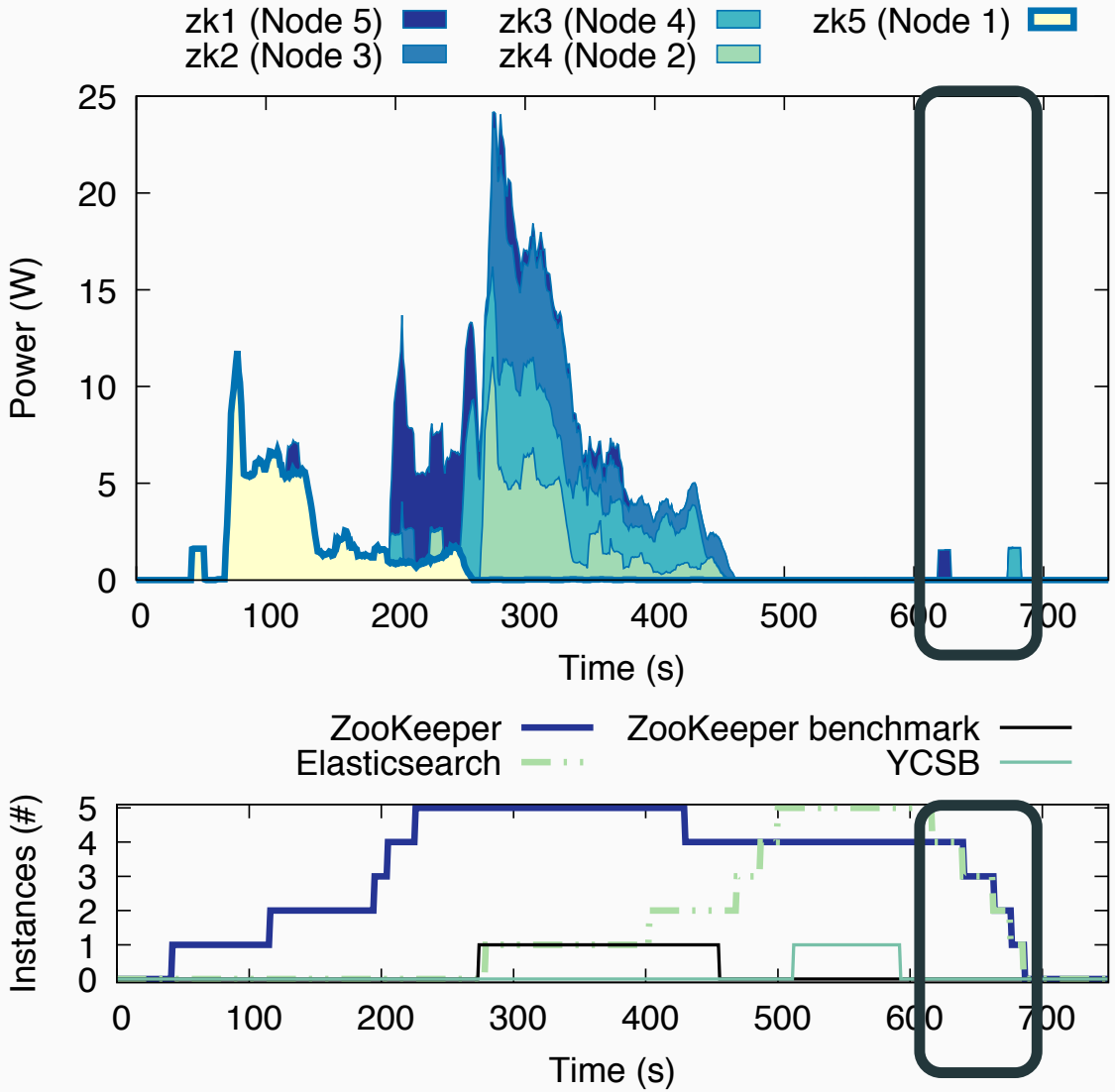
A Service-Level Power Monitoring



A Service-Level Power Monitoring



A Service-Level Power Monitoring



Conclusion

Contributions

WattsKit, Software-Defined Power Monitoring of Distributed Systems

Contributions

WattsKit, Software-Defined Power Monitoring of Distributed Systems

- **RQ1**: Can we model the software power consumption regardless of the underlying architecture?

  Open-testbed approach for learning multi-core power models

Contributions

WattsKit, Software-Defined Power Monitoring of Distributed Systems

- **RQ1**: Can we model the software power consumption regardless of the underlying architecture?

  Open-testbed approach for learning multi-core power models

- **RQ2**: Can we propose a uniform view of the service power consumption?

  In width energy monitoring, thanks to WattsKit

Thanks for your attention.

Maxime COLMANT
maxime.colmant@inria.fr

WattsKit, for distributed systems:
http://wattskit.powerapi.org/

BitWatts, for virtualized environments:
http://bitwatts.powerapi.org/

[Col+17] M. Colmant et al. “WattsKit: Software-Defined Power Monitoring of Distributed Systems”. In: *17th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGrid)*. 2017.

References I

[Ber+10] R. Bertran et al. “Decomposable and Responsive Power Models for Multicore Processors Using Performance Counters”. In: *Proceedings of the 24th ACM International Conference on Supercomputing*. 2010.

[BL09] C. Bienia and K. Li. “PARSEC 2.0: A New Benchmark Suite for Chip-Multiprocessors”. In: *Proceedings of the 5th Annual Workshop on Modeling, Benchmarking and Simulation*. 2009.

[CM05] G. Contreras and M. Martonosi. “Power Prediction for Intel XScale® Processors Using Performance Monitoring Unit Events”. In: *Proceedings of the International Symposium on Low Power Electronics and Design*. 2005.

[Col+15] M. Colmant et al. “Process-level Power Estimation in VM-based Systems”. In: *Proceedings of the 10th European Conference on Computer Systems (EuroSys)*. 2015.

[Col+17] M. Colmant et al. “WattsKit: Software-Defined Power Monitoring of Distributed Systems”. In: *17th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGrid)*. 2017.

[Dol+15] M. F. Dolz et al. “An analytical methodology to derive power models based on hardware and software metrics”. In: *Computer Science - Research and Development* (2015).

References II

[ERK06] D. Economou, S. Rivoire, and C. Kozyrakis. “Full-System Power Analysis and Modeling for Server Environments”. In: *In Workshop on Modeling Benchmarking and Simulation*. 2006.

[IM03] C. Isci and M. Martonosi. “Runtime Power Monitoring in High-End Processors: Methodology and Empirical Data”. In: *Proceedings of the 36th Annual IEEE/ACM International Symposium on Microarchitecture*. 2003.

[Int15] Intel. *Intel 64 and IA-32 Architectures Software Developer’s Manual*. 2015. URL: `http://www.intel.com/content/dam/www/public/us/en/documents/manuals/64-ia-32-architectures-software-developer-vol-1-manual.pdf` (visited on 08/01/2016).

[RRK08] S. Rivoire, P. Ranganathan, and C. Kozyrakis. “A Comparison of High-level Full-system Power Models”. In: *Proceedings of the Conference on Power Aware Computing and Systems*. 2008.

[The08] The Climate Group. *SMART 2020: Enabling the low carbon economy in the information age*. 2008. URL: `http://gesi.org/article/43` (visited on 09/23/2016).

[Yan+14] H. Yang et al. “iMeter: An integrated VM power model based on performance profiling”. In: *Future Generation Computer Systems* (2014).

[Zha+14] Y. Zhai et al. “HaPPy: Hyperthread-aware Power Profiling Dynamically”. In: *Proceedings of the USENIX Annual Technical Conference*. 2014.