The Green Lab

...on the road to greener software

Dr. Giuseppe Procaccianti

VU VRIJE UNIVERSITEIT AMSTERDAM

LOOKING FURTHER

Presentations are in order

- Dr. Giuseppe Procaccianti
- Postdoc, Software and Services
- Contact me:
  - g.procaccianti@vu.nl

Agenda

- **ICT is unsustainable**
- ...and Software is the culprit
- Measuring the impact of Software
- Best Practices for energy-efficient software
- Our playground: the Green Lab
- Conclusions

ICT is unsustainable

ICT is unsustainable

ICT is unsustainable

Total energy per view:

0.2 kWh

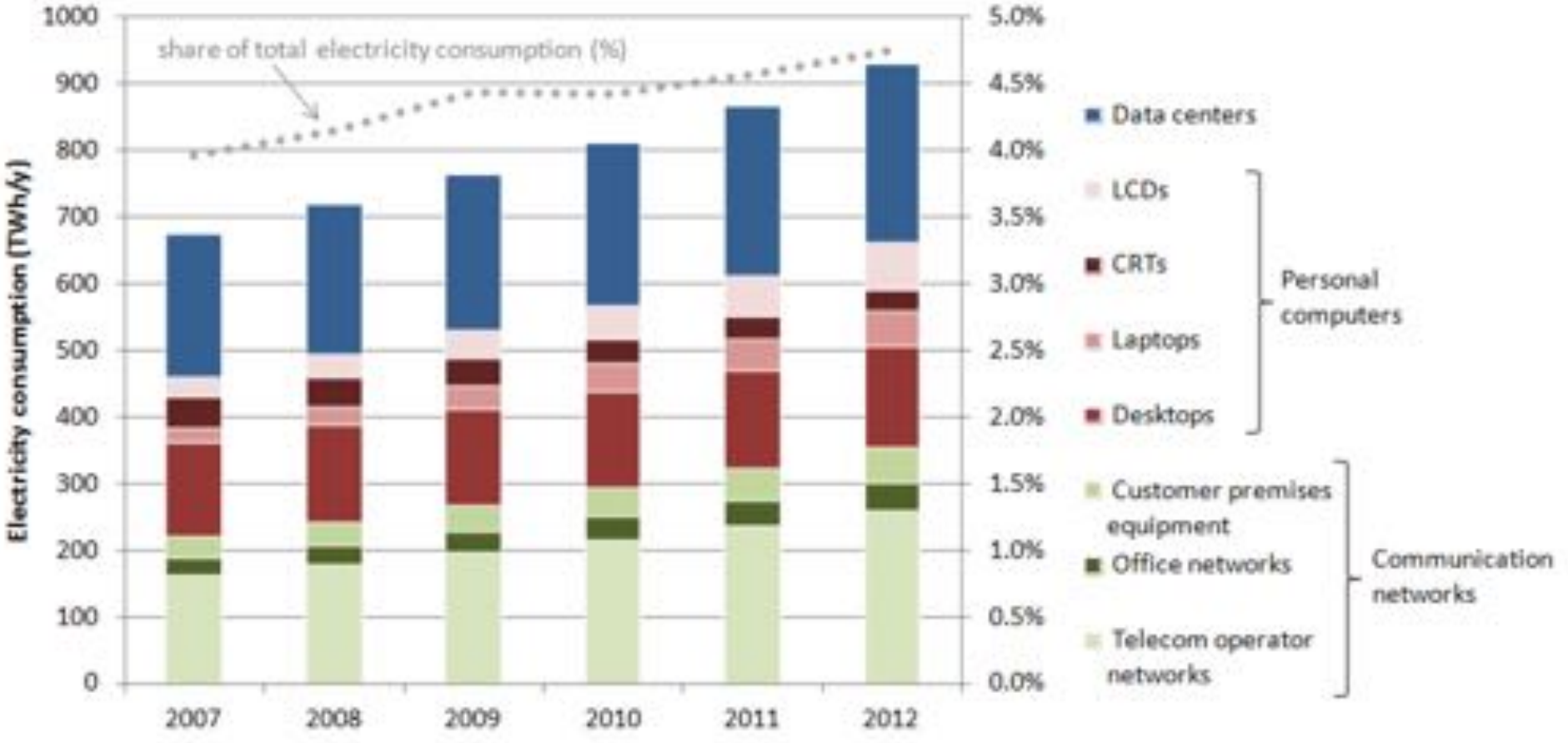
Total number of views:

2,307,082,001

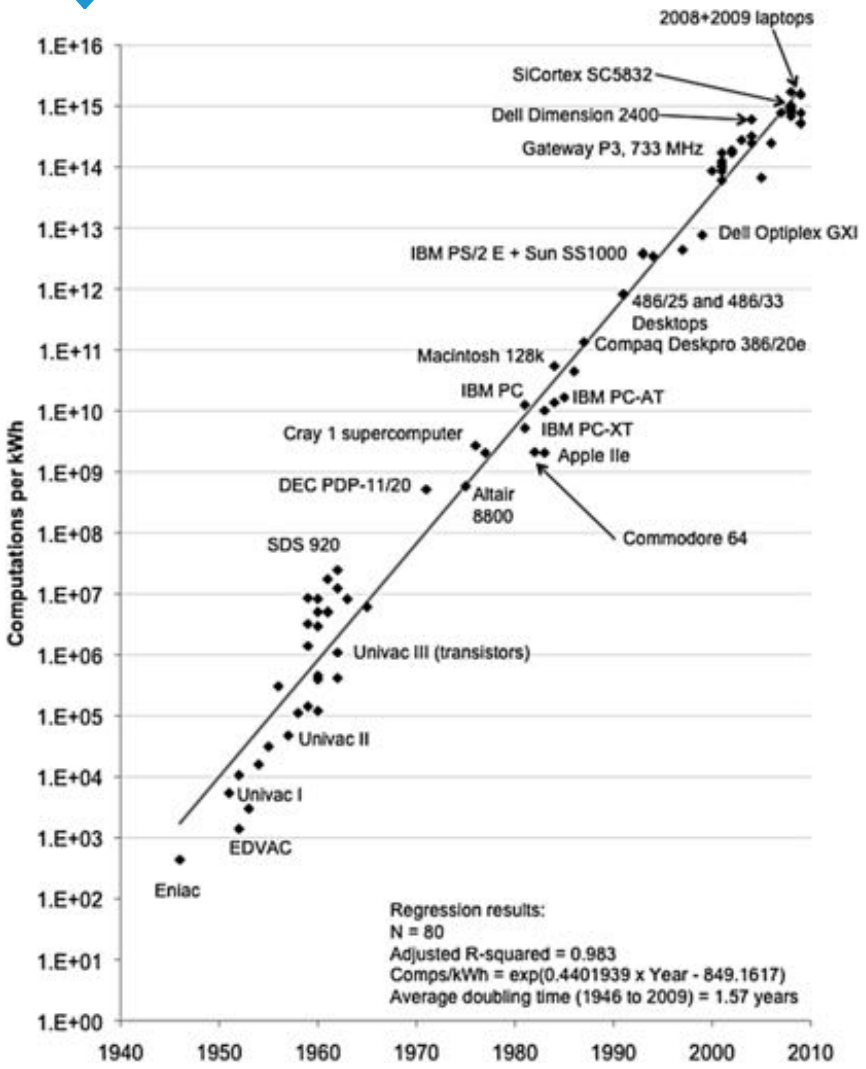
Total energy consumed: **450 GWh** in less than 3 years

x 45,000

ICT is unsustainable



ICT is unsustainable



"The energy efficiency of hardware doubles every 1.5 years."

(Koomey's law)

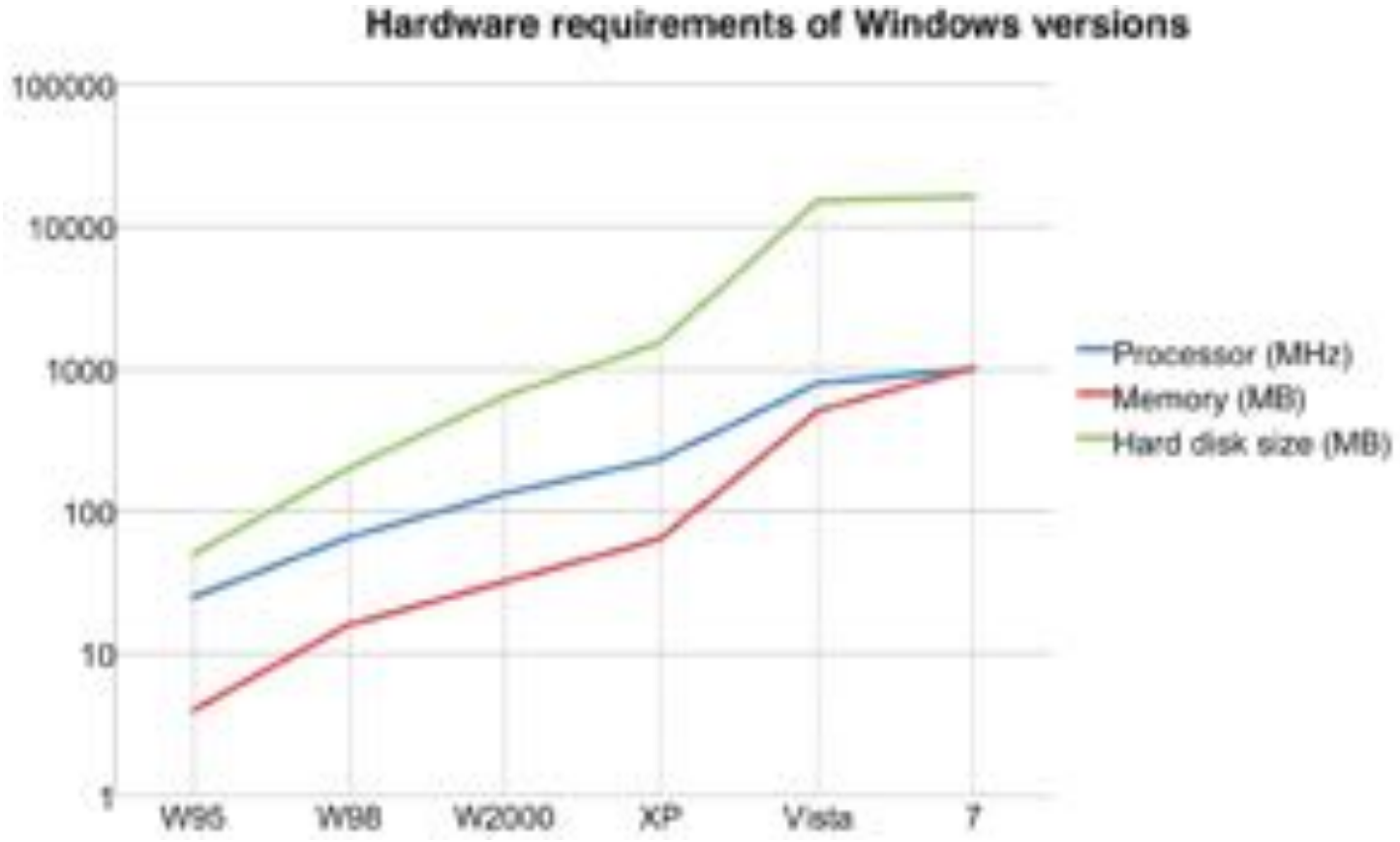
Agenda

- ICT is unsustainable
- **...and Software is the culprit**
- Measuring the impact of Software
- Our playground: the Green Lab
- The future

Software is unsustainable

"Software gets slower more rapidly than hardware gets faster."

(Wirth's law)



Software is unsustainable

1. **Software is a gas**
   Software always expands to fit whatever container it is stored in.

2. **Software grows until it becomes limited by Moore's Law**
   The initial growth of software is rapid, like gas expanding, but is inevitably limited by the rate of increase in hardware speed.

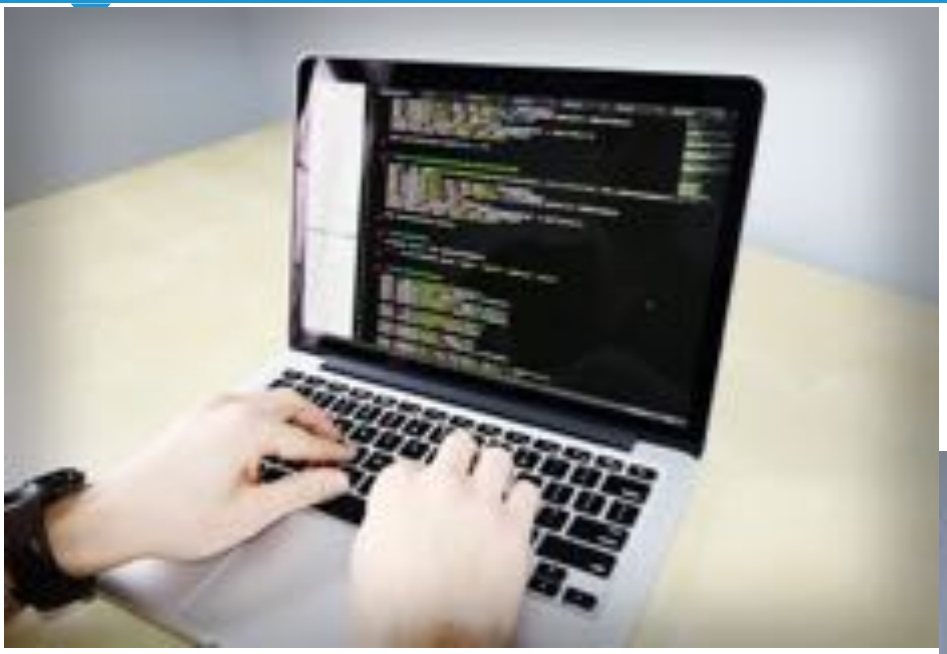
3. **Software growth makes Moore's Law possible**
   People buy new hardware because the software requires it.

4. **Software is only limited by human ambition and expectation**
   We'll always find new algorithms, new applications, and new users.

Nathan P. Myhrvold, Microsoft, ACM 1997

VU

Software is unsustainable



?

Agenda

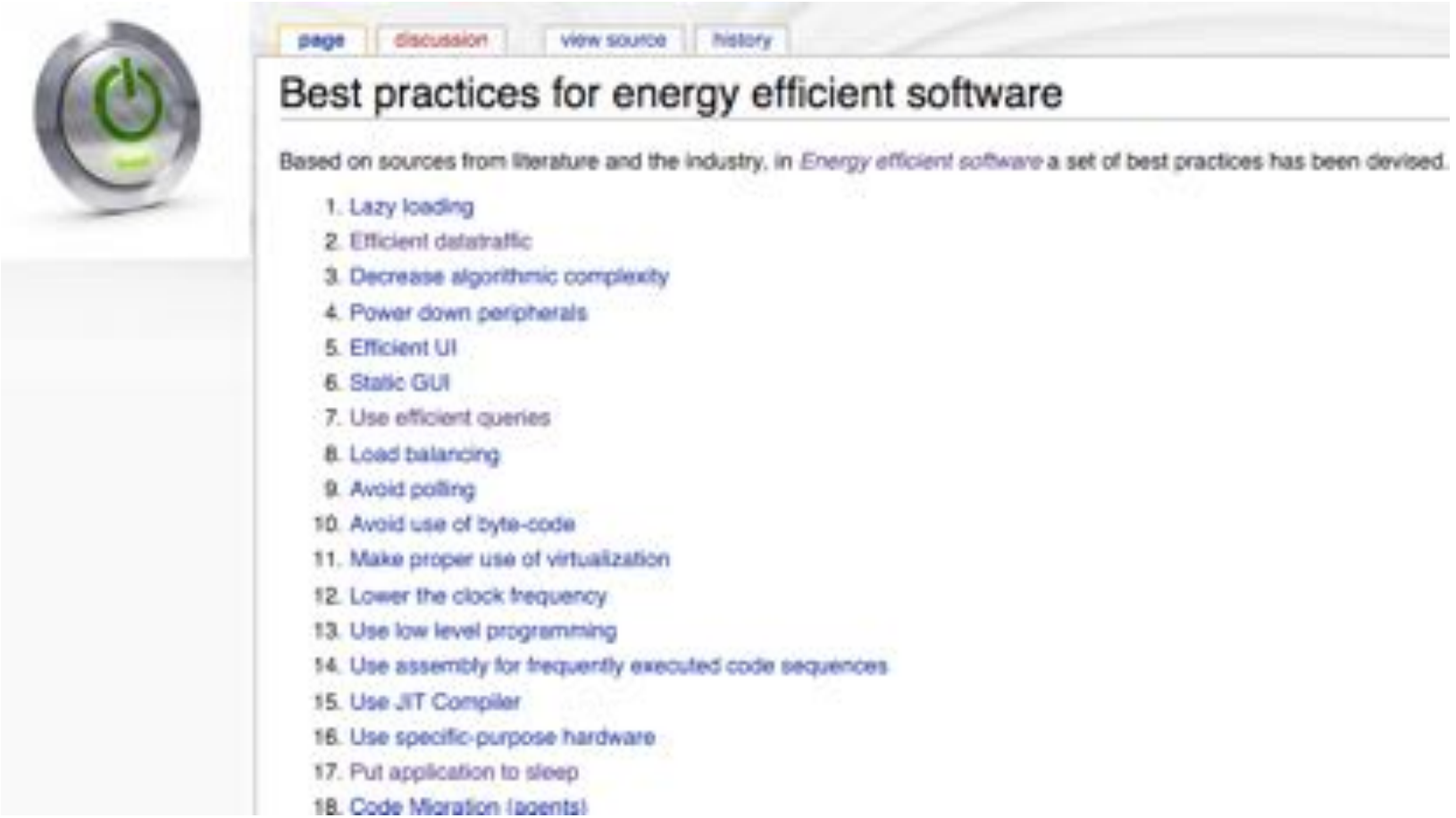
- ICT is unsustainable
- ...and Software is the culprit
- **Measuring the impact of Software**
- Our playground: the Green Lab
- The future

Best practices for Energy-efficient Software

- Many best-practices available in literature...
- ...without proper validation
- Poor documentation and context description

Green Software wiki

https://wiki.cs.vu.nl/green_software



page discussion view source history

Best practices for energy efficient software

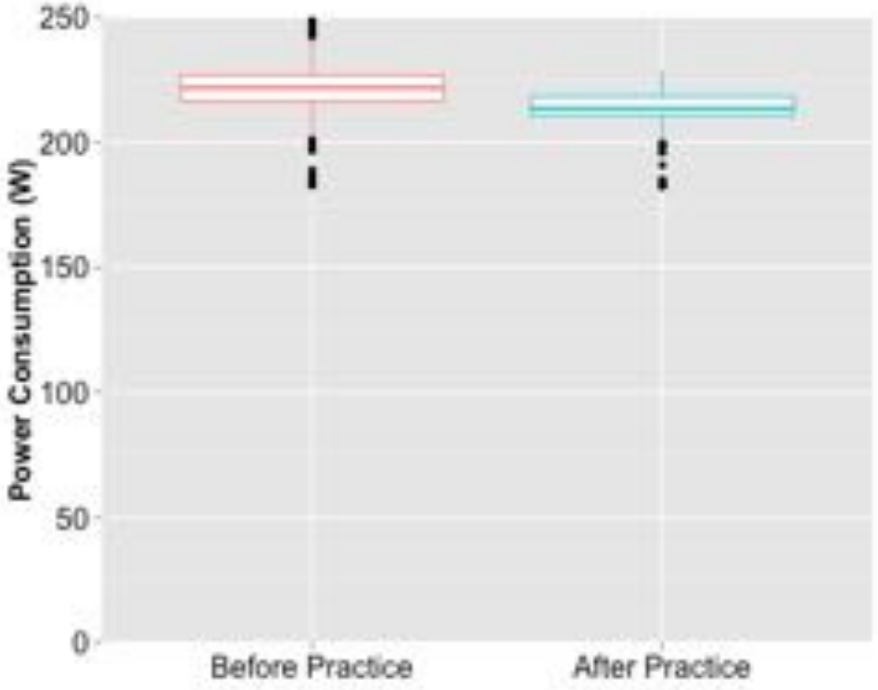
Based on sources from literature and the industry, in *Energy efficient software* a set of best practices has been devised.

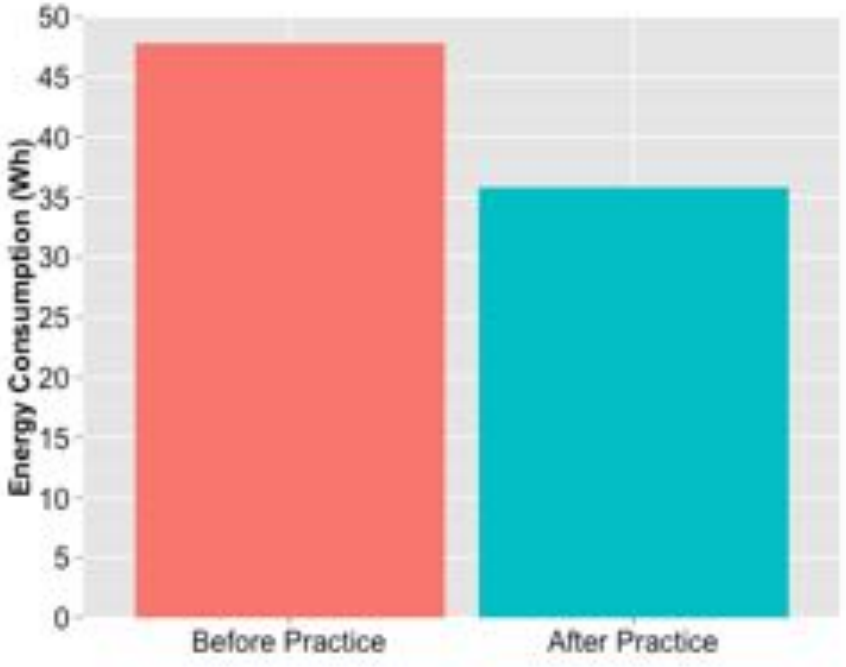
1. Lazy loading
2. Efficient datatraffic
3. Decrease algorithmic complexity
4. Power down peripherals
5. Efficient UI
6. Static GUI
7. Use efficient queries
8. Load balancing
9. Avoid polling
10. Avoid use of byte-code
11. Make proper use of virtualization
12. Lower the clock frequency
13. Use low level programming
14. Use assembly for frequently executed code sequences
15. Use JIT Compiler
16. Use specific-purpose hardware
17. Put application to sleep
18. Code Migration (agents)

VU

Green Software Practices evaluation [1]

Practice 1: use efficient queries





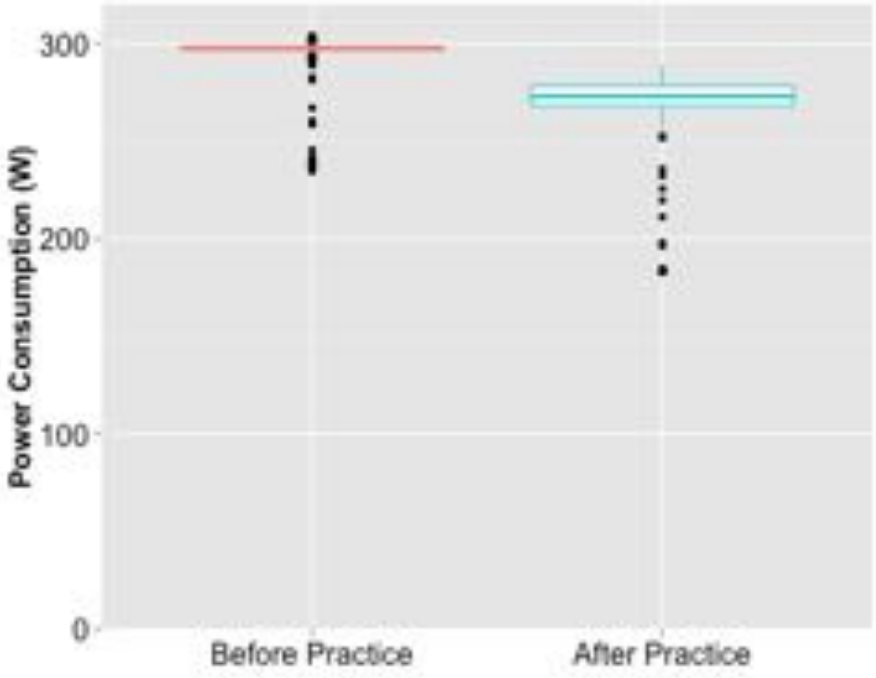
3% Power savings

25% Energy savings

[1] Procaccianti G., Fernandez H., Lago P. *"Empirical Evaluation of Two Best-Practices for Energy-Efficient Software Development"*. Accepted for publication in *Journal of System and Software*, 2016. Pre-print available on: http://dare.ubvu.vu.nl/handle/1871/54184

Green Software Practices evaluation [1]

Practice 2: put application to sleep



8.2% Power savings

8.4% Energy savings

[1] Procaccianti G., Fernandez H., Lago P. *"Empirical Evaluation of Two Best-Practices for Energy-Efficient Software Development"*. Accepted for publication in *Journal of System and Software*, 2016. Pre-print available on: http://dare.ubvu.vu.nl/handle/1871/54184

Design Tactics for Energy-efficient Software

- Energy Efficiency is a Software quality attribute
- Software must be *designed* for energy efficiency
- General, reusable tactics are needed

Design Tactics for Energy-efficient Software [2]

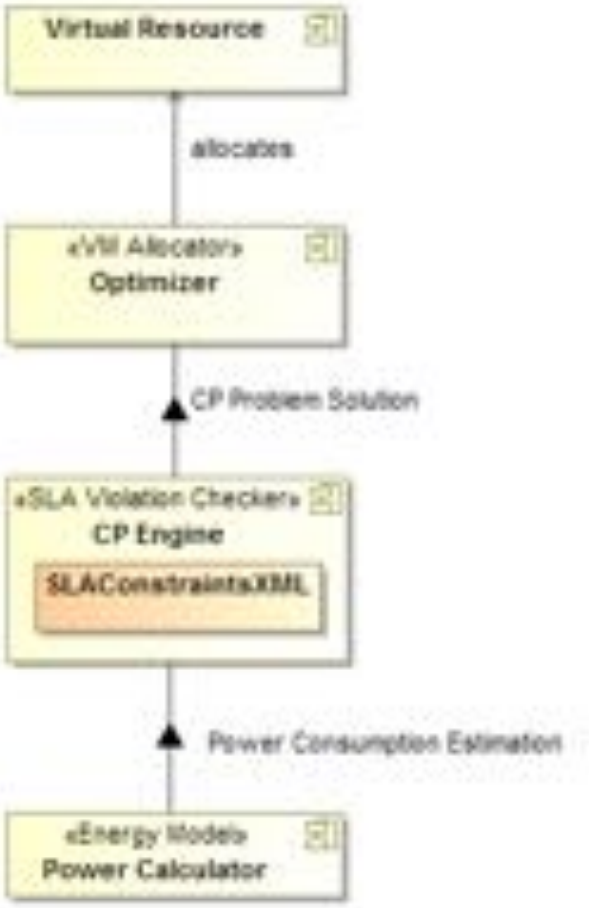
Strategy	**Tactic**
Energy Monitoring	Metering Modeling Static Classification
Self-Adaptation	Scaling Down Consolidation Workload Scheduling
Cloud Federation	Energy Brokering Service Adaptation

[2] Procaccianti, Giuseppe, Patricia Lago, and Grace A. Lewis. *"Green architectural tactics for the cloud."* Software Architecture (WICSA), 2014 IEEE/IFIP Conference on. IEEE, 2014.

Design Tactics for Energy-efficient Software [2]

Self Adaptation tactic**:**
Consolidation

(Implementation example)



[2] Procaccianti, Giuseppe, Patricia Lago, and Grace A. Lewis. *"Green architectural tactics for the cloud."* Software Architecture (WICSA), 2014 IEEE/IFIP Conference on. IEEE, 2014.

Agenda

- ICT is unsustainable
- ...and Software is the culprit
- Measuring the impact of Software
- Best Practices for energy-efficient software
- **Our playground: the Green Lab**
- The future

The Green Lab

- Our research initiative on Software Energy Efficiency
  - Laboratory
  - Master course
  - Network platform

The Lab

The Lab - support

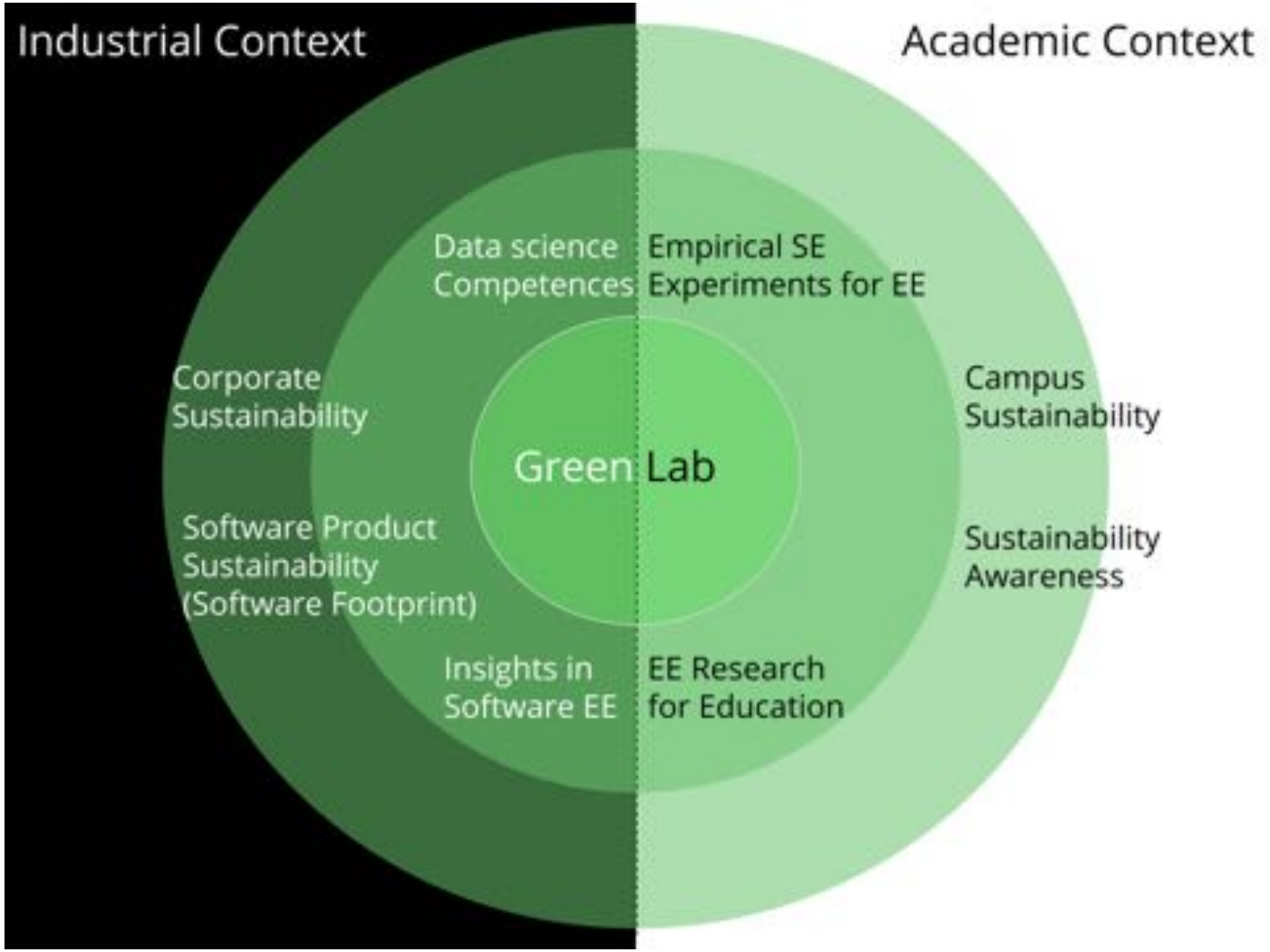
RAAK-mkb Greening the Cloud

The Green Lab course

Program: CS/SEG master

Aim: teaching students how to perform **experiments** related to software energy efficiency

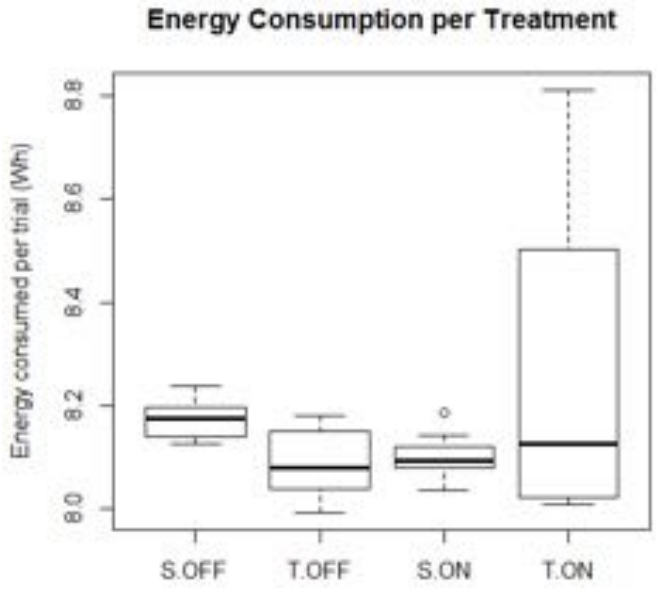
The Green Lab course

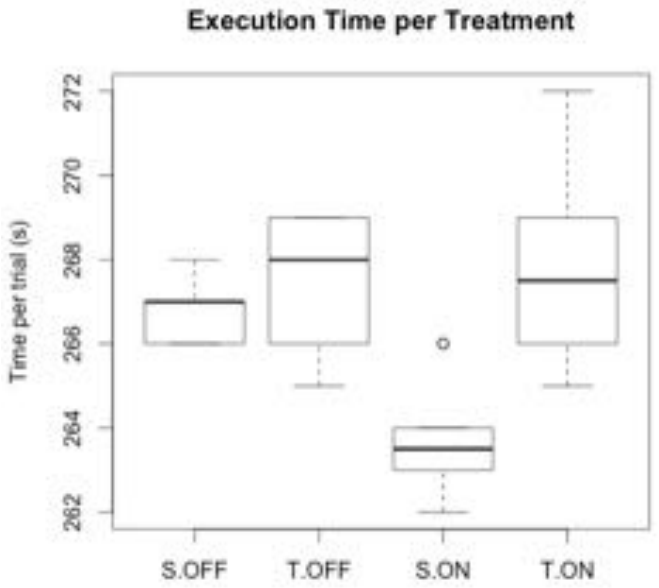


The Green Lab: case studies

Multi-tier business applications

- Evaluate the difference in performance and energy between different multi-tier architectures





The Green Lab: case studies

Multi-tier business applications

Conclusions

- Distributed configuration is more efficient
  - higher CPU usage
  - no increase in energy consumption
  - (slight) performance improvements

- Actionable recommendations:
  - Always turn on database sync
  - Always separate tiers

Agenda

- ICT is unsustainable
- ...and Software is the culprit
- Measuring the impact of Software
- Best Practices for energy-efficient software
- Our playground: the Green Lab
- **Conclusions**

Conclusions: takeaway message #1

Energy consumption is software-defined.

Conclusions: takeaway message #2

There is no *one-size-fits-all*.

Conclusions: takeaway message #3

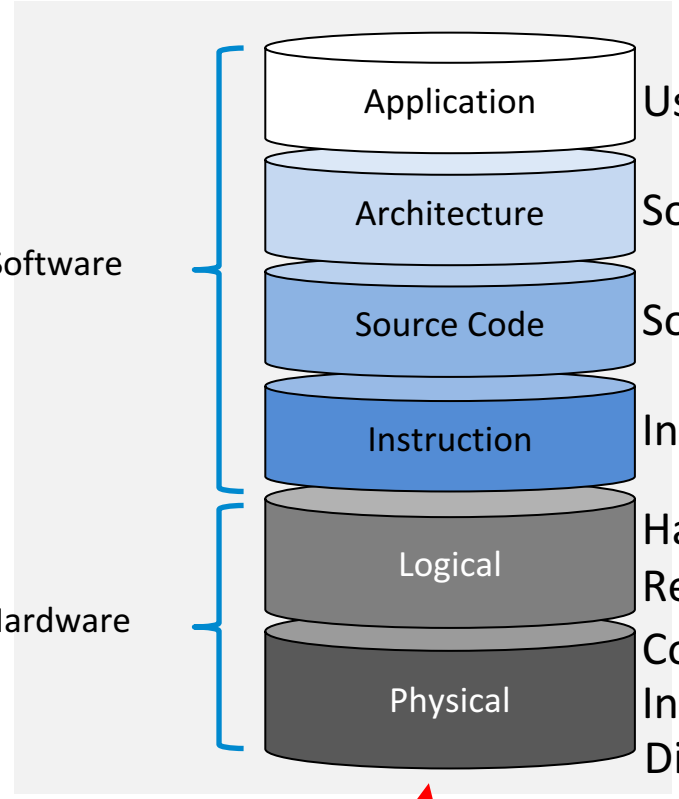
Measure, measure, measure.

Thank you!

g.procaccianti@vu.nl

EXTRA SLIDES

Measuring the impact of software



Usage scenarios, user behavior

Software components, architectural metrics

Source code metrics

Instructions, operands, parameters

Hardware vendor specs, power states, performance counters
Resources usage ratio (CPU, RAM...)

Component design characteristics
Indirect measurement (T, V, f...)
Direct measurement

Energy consumption

Measuring the impact of software

White-box

Software internals are *known*

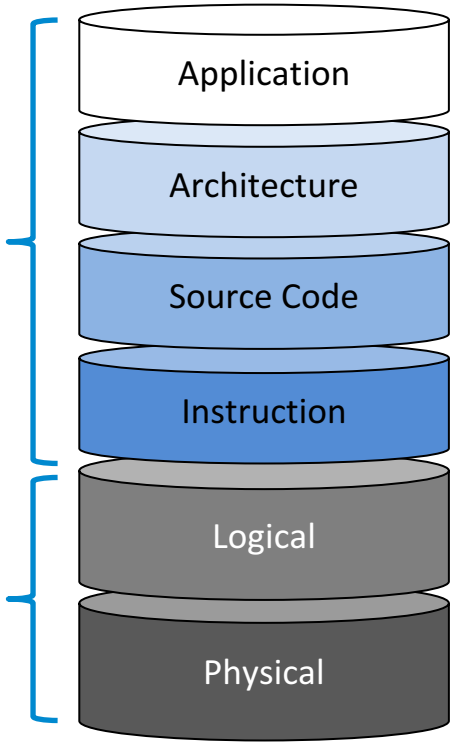
Black-box

Software internals are *unknown*

Measuring the impact of software

Software

Hardware



Usage scenarios, user behavior

Software components, architectural metrics

Source code metrics

Instructions, operands, parameters

Hardware vendor specs, power states....
Resources usage ratio (CPU, RAM...)

Component design characteristics
Indirect measurement (T, V, f...)
Direct measurement

Energy consumption

VU

Measuring the impact of software

Tool	Platform	White vs. Black-box
Joulemeter	Windows	Black-box
Intel Energy Checker	Windows/Linux	White-box
PowerTOP	Linux	Black-box
ARO	Mobile	White-box
PowerTutor	Mobile	Black-box
Apple Activity Monitor	Mac OS	Black-box