Hardware Parametric Energy Consumption Analysis

-

Joint TACLe – EACO Workshop on
Analysis Techniques for Energy-Aware Computing

-

Marko van Eekelen
Paolo Parisen Toldin, Rody Kersten, Bernard van Gastel
Marc Schoolderman, Jascha Neutelings, Dorus Peelen
Stein Keijzers, Ennier Kelly

-

Bristol, U.K., 2014, September 10-11

EUROPEAN COOPERATION IN SCIENCE AND TECHNOLOGY
Timing Analysis on Code-Level
Open Universiteit www.ou.nl
Radboud Universiteit Nijmegen

Who is that guy?

Prof.dr. Marko van Eekelen

- **O**pen **U**niversity of the Netherlands (**OU**) (0.7 fte); **R**adboud **U**niversity Nijmegen (**RU**) (0.3 fte)

Organisation

- *OU: Head of the Computer Science Department, Programme leader Master Software Engineering*
- *RU: Nijmegen Scientific Director of the Laboratory for Quality Software (LaQuSo), Coordinator of the Security Bachelor*

Conferences

- *Chair Steering Committee of the **Trends in Functional Programming** (TFP) Symposia*
- *Initiated in 2009 the **Foundations and Practical Aspects of Resource Analysis** (FOPARA) biannual workshop series*
- *Steering Committee of the **Computing Science Education Research** Conferences (CSERC)*
- *Expert Committee for the **EAPLS Ph.D. Awards***
- *Chair of **FOPARA2015**, co-located with DICE2015 and **ETAPS2015***

Projects

- Project leader NL **AHA** project on **Resource Consumption Analysis** (2006-2010)
- Verification Work Package Leader EU Artemis CHARTER project (2009-2012)
  - Verification of Safety Critical Applications including **Resource Consumption**
  - Industrial partners from Avionics, Automotive, Surveillance and Healthcare
- Member of the NL **GoGreen** Smart Home project (2011-2015)

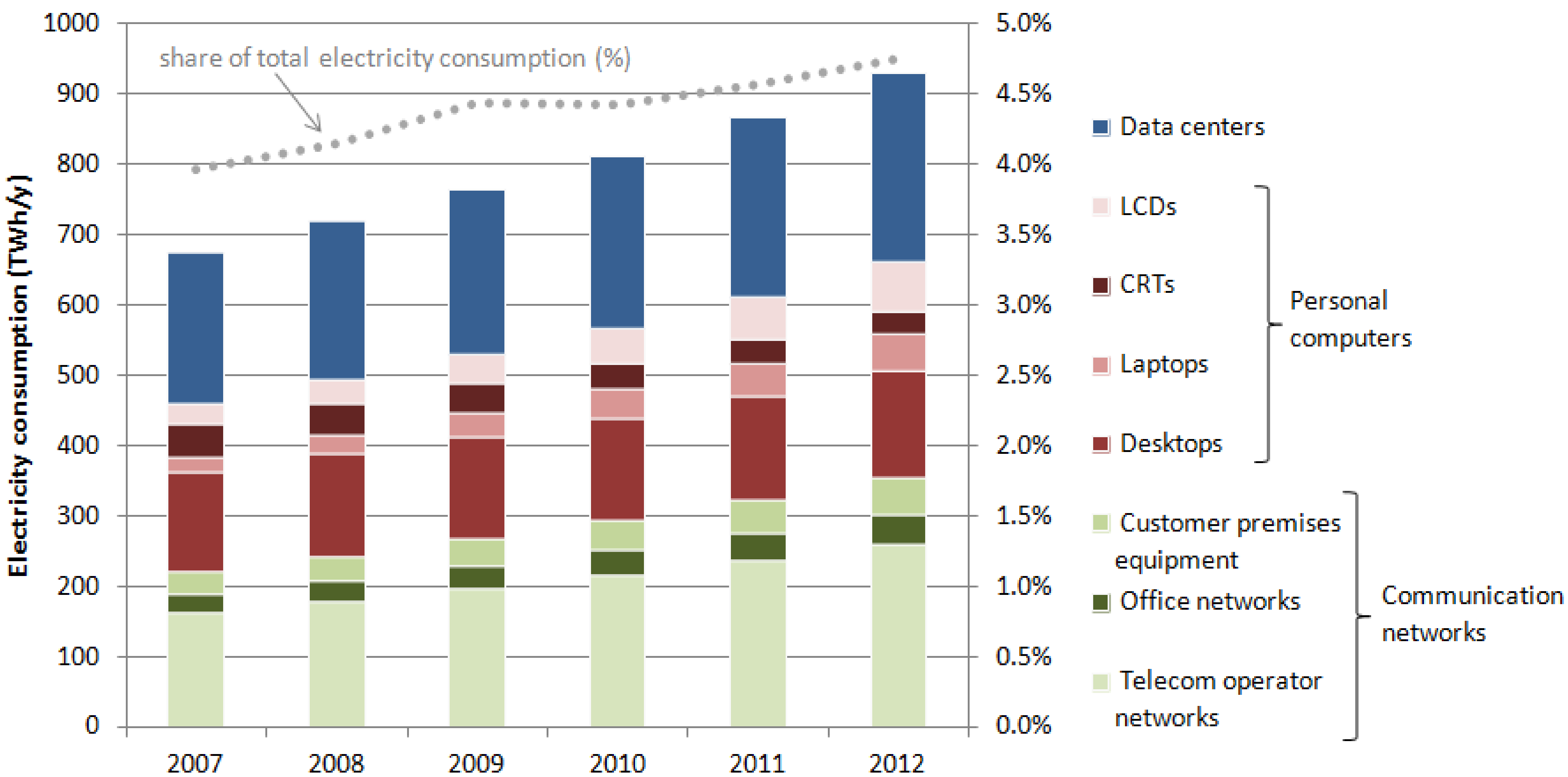
Contents of this talk

- Why Static Energy Consumption Analysis?
- Need for Hardware Parametricity
- Hoare Logic for Hardware Parametric Energy Consumption Analysis
- EcaLogic: a tool for Hardware Parametric Energy Consumption Analysis
- Example
- Current limitations, Work in Progress, Future work

- *Marc Schoolderman, Jascha Neutelings,Rody Kersten and Marko van Eekelen. **ECAlogic**: Hardware-Parametric Energy-Consumption Analysis of Algorithms. In Proceedings of the Foundations of Aspect-Oriented Languages Workshop, (FOAL 2014, Modularity 2014). ACM Digital Library 2014.*
- *Rody Kersten, Paolo Parisen Toldin, Bernard van Gastel, Marko van Eekelen. A **Hoare Logic** for Energy Consumption Analysis, In Proceedings of the 2013 International Workshop on Foundational and Practical Aspects of Resource Analysis (FOPARA2013). In colocation with WCET2013. Bertinoro. Italy. Lecture Notes Computer Science. Springer Verlag. To appear.*
- *Paolo Parisen Toldin, Rody Kersten, Bernard van Gastel, Marko van Eekelen. **Soundness Proof** for a Hoare Logic for Energy Consumption Analysis, Institute for Computing and Information Sciences, Radboud University Nijmegen, Technical Report ICIS-R13009, October 2013, Radboud University Nijmegen.*

Worldwide ICT energy consumption growing (6.6% per year)

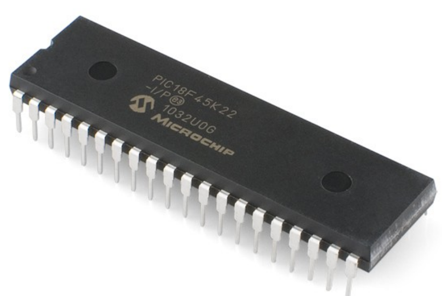
- Currently, only dealt with in hardware!



Source: D8.1. Overview of ICT energy consumption, FP7-288021 – Network for Excellence in Internet Science

Hardware Efficiency

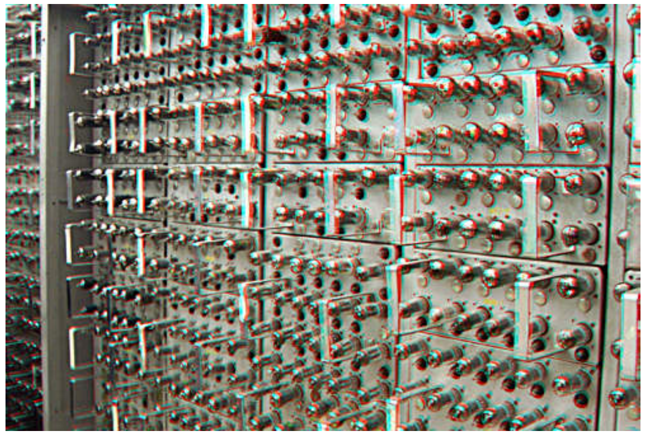
- Focus on many efficiency methods
  - Laws and regulations, long-term plan, etc
  - Hardware limits



~0.01W

~0.1W

~100W



~10,000W

Hardware improvement is needed

but it is the software that controls the hardware…

Importance of Energy Efficiency

- ICT controls the world and hence ICT controls most of the energy used in the world!
- Analysis of complete systems is needed: software+hardware, control+machine

Static vs Dynamic analysis

With dynamic analysis the properties of a running system is monitored at run-time.

With static analysis the properties of a running system are predicted at design-time.

Static analysis can help in
- Preventing problems at run-time
- Exploring the consequences of **design** choices
- Studying properties of the design

Dynamic analysis can help in
- Exploring the properties of the running system
- Monitoring the running system for problems
- Measuring key aspects of the running system

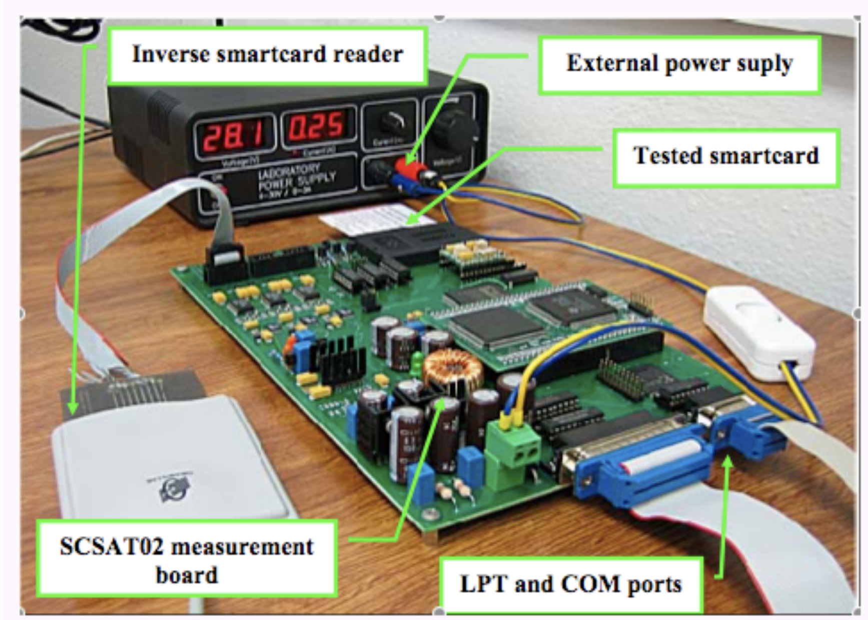
Hardware energy consumption is controlled by software

Hardware performance and energy consumption efficiency is improving steadily due to hardware specific improvements:

- Hardware improvements;
- Compiler improvements;
- Software analysis for specific hardware

We propose a *hardware parametric* approach

- Input for the analysis are models of the hardware components
- The analysis itself is generic. It uses the energy model as a parameter to ***statically*** derive upper-bounds for energy consumption



It requires *hardware component models*:

- i.e. hardware energy consumption models including internal state information
- key parameters guessed, provided by manufacturers and/or measured ***dynamically***

Component models and a Hoare logic…

- Basic hardware component description language
  - Finite state machine
    - energy levels attached to each state
    - constant power draw per state
    - energy usage: time consumed * state power level
  - Component functions may
    - change state, energy usage is constant per component function
  - Total energy consumption
    - sum of all incidental usage by component functions
      + sum over states (time spent in state * state power draw)

- Hoare logic with logical rules describing how energy consumption is approximated
  - …

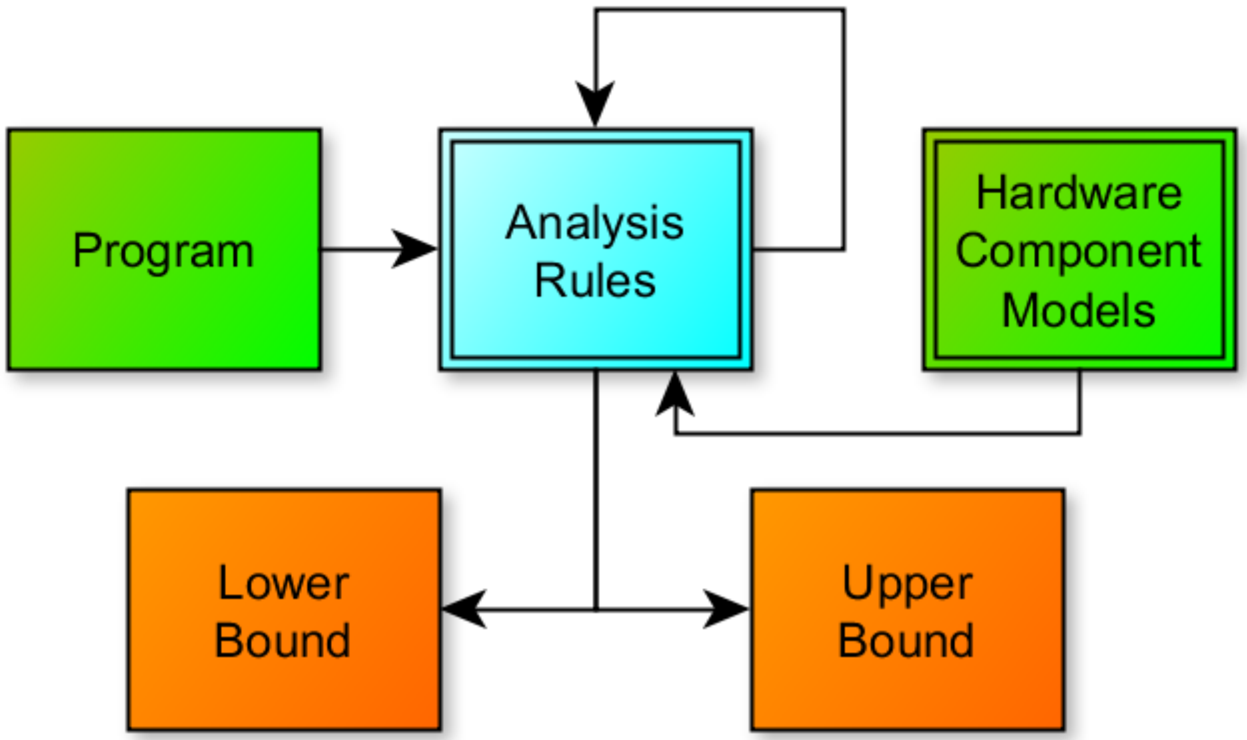
Component models and a Hoare logic…

- *Basic hardware component description language*
  - *Total energy consumption*
    - *sum of all incidental usage by component functions*
      *+ sum over states (time spent in state * state power draw)*

- Hoare logic with logical rules describing how energy consumption is approximated
  - Pre-assumes annotations for loop bounds and variable values
  - Includes an implicit component for the processor
  - Takes in the info from the component parameters
  - Result: symbolic bound on energy consumption
  - Soundness with respect to energy-aware semantics is proven (result is over-estimation of energy usage)

So, what do we have in the EcaLogic tool…

- Set of logic rules used for analysis
  - For every statement or expression; Result is an over-estimation
- energy consumed by the hardware is modelled in hardware component models
  - Incidental and time-dependent consumption



http://resourceanalysis.cs.ru.nl/energy/

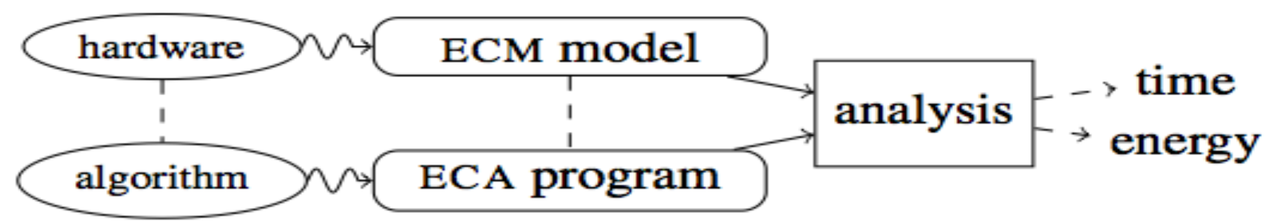
ECAlogic energy consumption analysis demonstrator

Load example program: Radio2.eca [Load]

Load additional component model: SSD.ecm [Load]

Code | Implicit | Radio | Sensor

```
component Radio(active: 0..1)

    initial active := 0

    component function on uses 400 time 400 energy
        active := 1
    end function

    component function off uses 200 time 200 energy
        active := 0
    end function

    component function queue(X) uses 30 time 30 energy
    component function send uses 100 time 100 energy

    function phi := 2 + 200 * active

end component
```



[Analyse] [Advanced]

Design time decision: which algorithm is best?

Wireless sensor node #1

```
function alwaysOn(N)
  Radio::on()
  while N > 0 bound N do
    Value := Sensor::measure()
    Radio::queue(Value)
    Radio::send()
    N := N−1
  end while
  Radio::off()
end function
```

Wireless sensor node #2

```
function buffering(N, B)
  while N > 0 bound N/B do
    K := B
    while K > 0 and N > 0 bound B do
      Value := Sensor::measure()
      Radio::queue(Value)
      K := K − 1
      N := N − 1
    end while
    Radio::on()
    Radio::send()
    Radio::off()
  end while
end function
```

- `buffering` consumes less energy when $B \geq 3$, $N \geq 3$
- `buffering` takes less time when $B \geq 12$

	time	*energy*
alwaysOn(N)	$600 + 195 \cdot N$	$83600 + 40200 \cdot N$
buffering(N,B)	$(130 + \frac{740}{B}) \cdot N$	$(1070 + \frac{105640}{B}) \cdot N$

Table: Comparing sensor node #1 and #2

N Number of samples
B Samples per packet

```
component Radio(active: 0..1)
  initial active := 0

  component function on uses 400 time 400 energy
    active := 1
  end function

  component function off uses 200 time 200 energy
    active := 0
  end function

  component function queue(X) uses 30 time 30 energy
  component function send uses 100 time 100 energy

  function phi := 2 + 200 * active
end component
```

Limitations of published work

First steps only

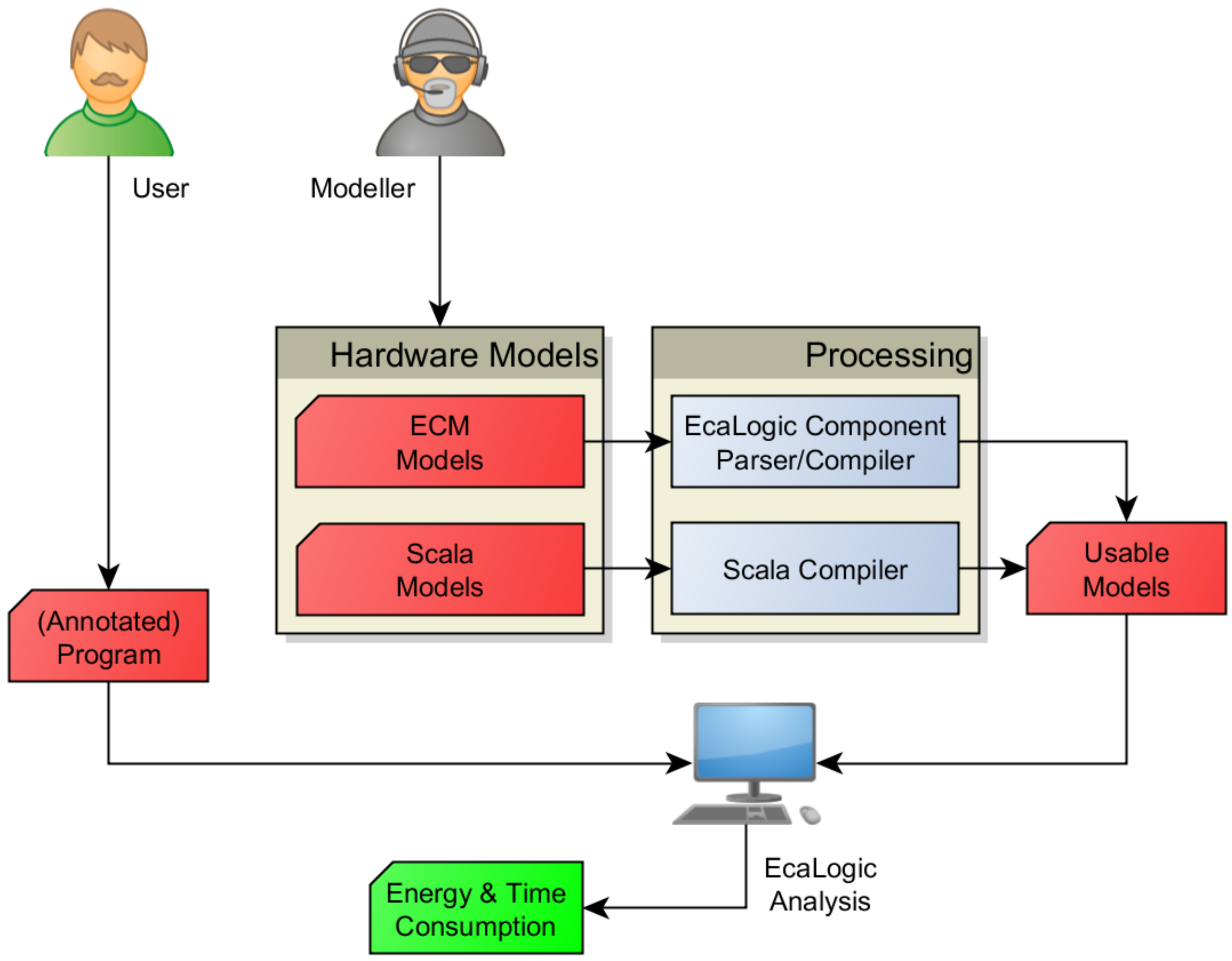
- Straightforward, simple language, no recursion, no energy signatures

Limitations

- Component state functions take up a constant amount of time and incidental energy
  - *mitigate by changing the program*
- Component states have a constant power draw
  - *mitigate by changing state to a higher energy level in component function*
- Component models must be finite state machines
  - *mitigate using abstraction*
- The effect of component state functions on the component states cannot depend on the arguments of the function
  Also, component models are independent, they cannot influence each other
  - *mitigate using multiple component state functions*

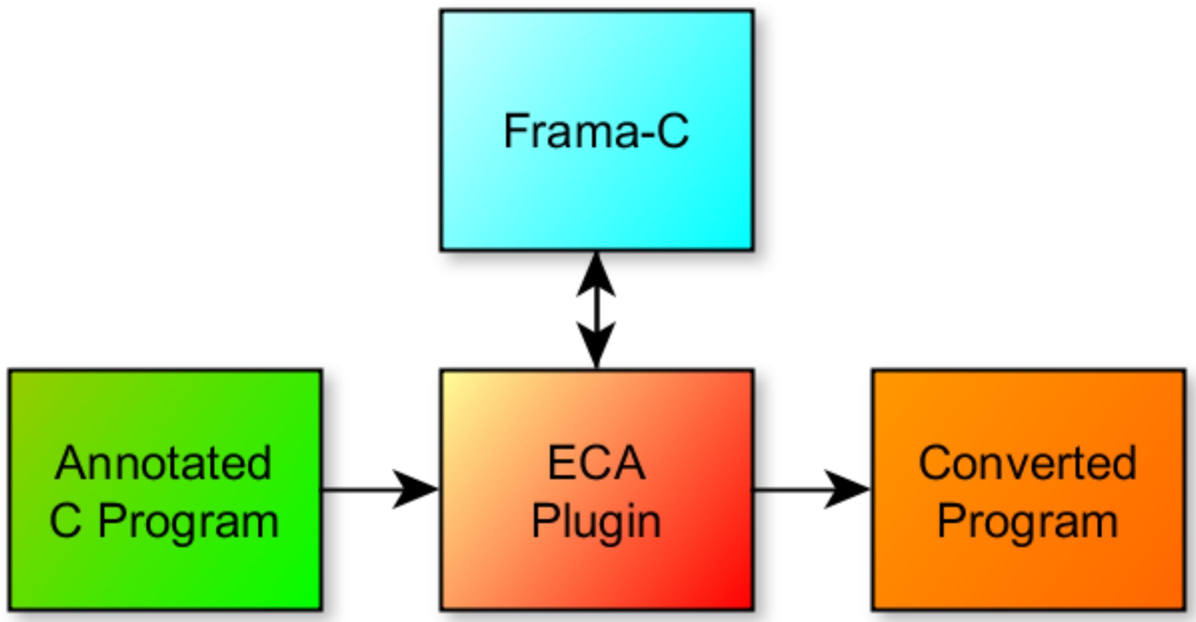
EcaLogic

- Implementation of the original analysis
  - Works on a toy language

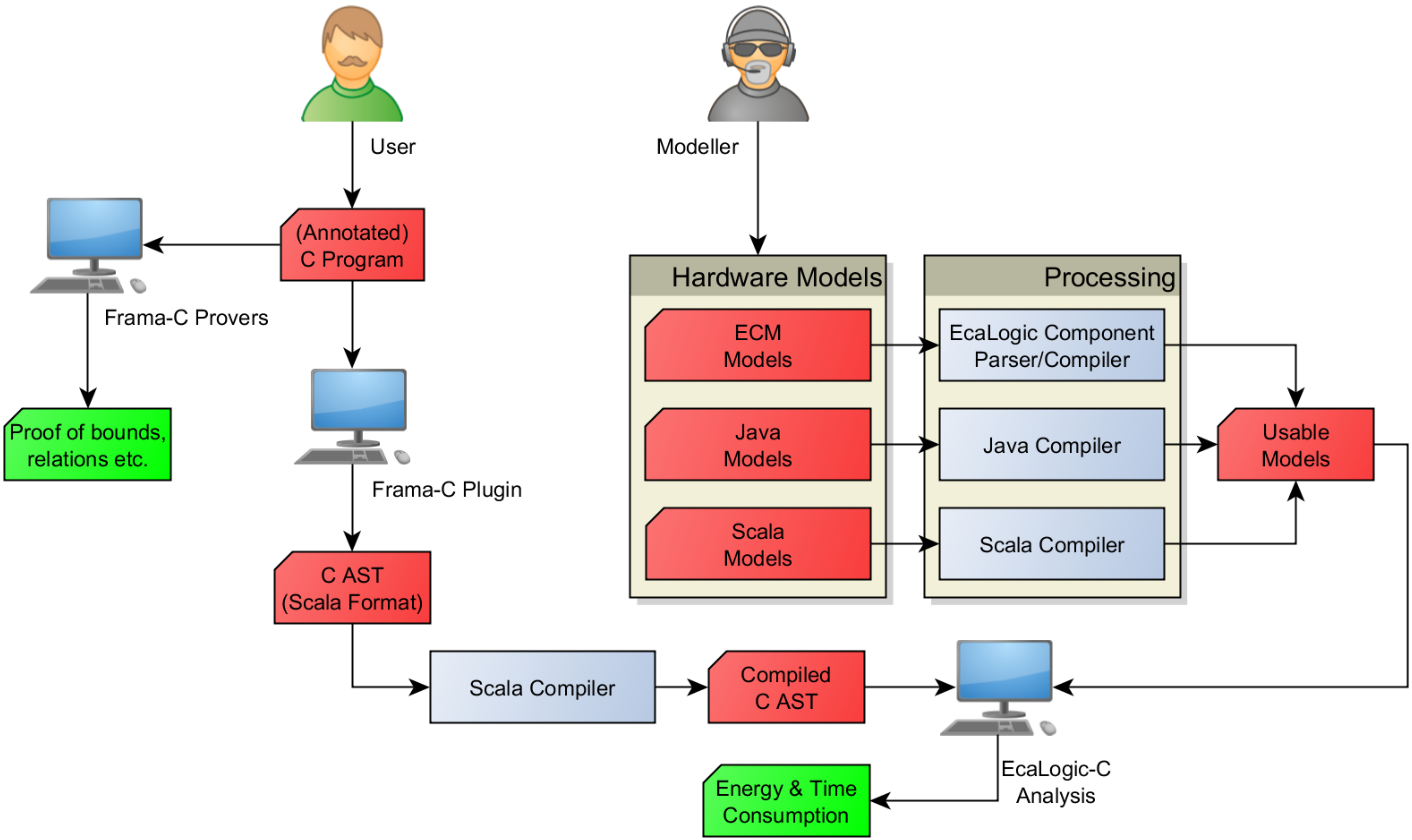


Work in progress with Stein Keijzers

- C is used a lot for control software
- Use Frama-C
  - Annotations can be proven with Frama-C
  - Create an ECA plugin to Frama-C converts the annotated C program to the extended EcaLogic core
    - data structures, memory manipulation, recursion



EcaLogic-C


User
Modeller
(Annotated) C Program
Frama-C Provers
Proof of bounds, relations etc.
Frama-C Plugin
C AST (Scala Format)
Hardware Models
ECM Models
Java Models
Scala Models
Processing
EcaLogic Component Parser/Compiler
Java Compiler
Scala Compiler
Usable Models
Scala Compiler
Compiled C AST
EcaLogic-C Analysis
Energy & Time Consumption

Work in progress with Ennier Kelly

- Analyse and compare energy consumption of two different DNS server implementations
- Measurements for server component models (SEF lab Amsterdam)
- Abstraction of full server code
- Run analysis

- Quite a challenge: both in abstraction and in measuring

Future work (unordered)

- Take away current limitations (recursion, implicit component, …)
- Add lower bounds
- Increase modularity adding energy signatures
- Add some form of concurrency in software/ hardware models
- Solve symbolic energy consumption comparisons automatically
- Go for a full, real language
- Apply in practice
- Set up library of various kinds of energy models
- Create smooth transition from modeling to actual system
- Study larger systems
- ….

Summary

We analyse energy consumption of software controlled ‘hardware’ with

- Energy models of the hardware components
- A hardware-parametric static analysis technique
- Promising first results:
  - Tool EcaLogic
  - Proven soundness of basic analysis using Hoare logic
- Lots of work to be done: students are welcome for research visits

FOPARA2015 *see you @ ETAPS in London?*

- Part of **ETAPS2015**, **London** UK, April 11-18, 2015
  - CC, ESOP, FASE, FOSSACS, POST, TACAS, …
- Co-located with DICE2015, Marco Gaboardi, April 11 and 12
- Post-event peer reviewing, presentation acceptance based on paper for local proceedings, proper **LNCS** postproceedings
- Topics:
  - original research results
  - relevant to the analysis of **resource (e.g. time, space, energy) consumption** by computer programs
  - Aim: to bring together the researchers that work on foundational issues with the researchers that focus more on practical results
    - **both theoretical and practical contributions** are encouraged
    - also encouraged are papers that **combine theory and practice**
- Special attention to TACLe, **TACLe funded**
- Call for papers, expected soon… Tentative submission: January, 2015