Formal Methods can help to save the world

Marko van Eekelen

Open Universiteit, Radboud University

Institute for Computing and Information Sciences
Radboud University

Open Universiteit
www.ou.nl

Marko van Eekelen: Who is that guy?

- **Radboud University** Nijmegen & **Open Universiteit** NL
  - RU: Digital Security (safety critical systems); OU: Chair CS department (resource analysis)
- **Functional Programming Background**
  - term graph rewriting, language Clean, Chair of TFP Steering committee (2008-2017)
- **Verification of functional properties**
  - Sparkle theorem prover, Strictness analysis
  - **Dutch Storm Surge Barrier (Maeslantkering)**
  - Current: Sovereign (2016-2020, supported by RWS, NRG, NASA)
    - Safety critical C-verification with Herman Geuvers, Freek Wiedijk, Léon Gondelman, Freek Verbeek
- **Analysis/verification of non-functional properties**
  - Security Analysis (Secure Metering 2009-2013, Online Banking 2012-2016)
  - **Size analysis** of strict functional programs & **Memory consumption analysis** for Java
    - ResAna tool (NL-NWO AHA project 2006-2010; EU_Artemis CHARTER 2009-2012)
  - **Energy Analysis of Software Controlled Systems** (Bernard van Gastel, Ph.D. 2016)
    - ECALogic tool; TACLe COST action, recent events organised in this area: FOPARA, -DICE, RAC

Marko van Eekelen Formal methods research @ Radboud University

Sovereign (2016-2020) NWO project overview

Project members:

- Prof.dr. Marko van Eekelen, Prof.dr Herman Geuvers
- Dr. Freek Wiedijk, Dr. Sjaak Smetsers, Dr. Freek Verbeek, Dr. Leon Gondelmans
- M.Sc. Dan Furmin, M.Sc, Marc Schoolderman

Partners:

- Venkat Natarajan, Nuclear Research Group (NRG)
- Bert van der Vegt, Rijkswaterstaat (RWS)

User Committee Members:

- Dennis Dams, Toegepast Natuurwetenschappelijk Onderzoek (TNO),
- Eric Verhulst, Altreonic,
- Jan Zwanenburg, Philips,
- Paul Zenden, Sioux

Interest:

- Alwyn Goodloe, National Aeronautics and Space Administration (NASA), USA
- Joe Kiniry, Galois Inc, USA

Goals: verification technology for safety critical systems; provable design techniques

Who is *Marko van Eekelen (still 0,3 @ Radboud Univ. but main task 0,7 @ OU)*

Head of the Computer Science Department of the Open University

emerging research group

27 OU-researchers + *4 post-docs*, 13 external Ph.D. students

- **Verification** Marko van Eekelen, Tanja Vos, *Pekka Aho,* Freek Verbeek, Jeroen Keiren, Stefano Schivo, Stijn de Gouw, Sung-Shik Jongmans
- **CS Education** Erik Barendsen, Johan Jeuring, Bastiaan Heeren, Josje Lodder, Harrie Passier, Sylvia Stuurman, Fenia Aivaloglou, Ebrahim Rahimi
- **Security** Harald Vranken**,** Hugo Jonker, Greg Alpar, Arjan Kok, *Fabian van den Broek, Sietse Ringers, Hassan Alizadeh*

- **Data Science / Artificial Intelligence** Arjen Hommersom, Twan van Laarhoven, Remko Helms, Stefano Bromuri
- **Requirements** Stef Joosten, Ella Roubtsova, Rogier van de Wetering, Rob Kusters, ……

Recent Open University awards/grants/projects

IPA research institute Best Thesis Award 2017

- Sung-Shik Jongmans

Computable Awards 2017

- ICT-educator of the year

Recent OU project grants (2016-2018)

- *Research projects*: Rubicon, Dagobert, Advise-ME, Impress, Testomat, Train4SmartServices, Veni, Open Math, File Carving
- *Impact projects*: Q-sense, Remote Voting, DHL, Cyberweerbaarheid Limburg

Recent OU participation in research projects of other universities

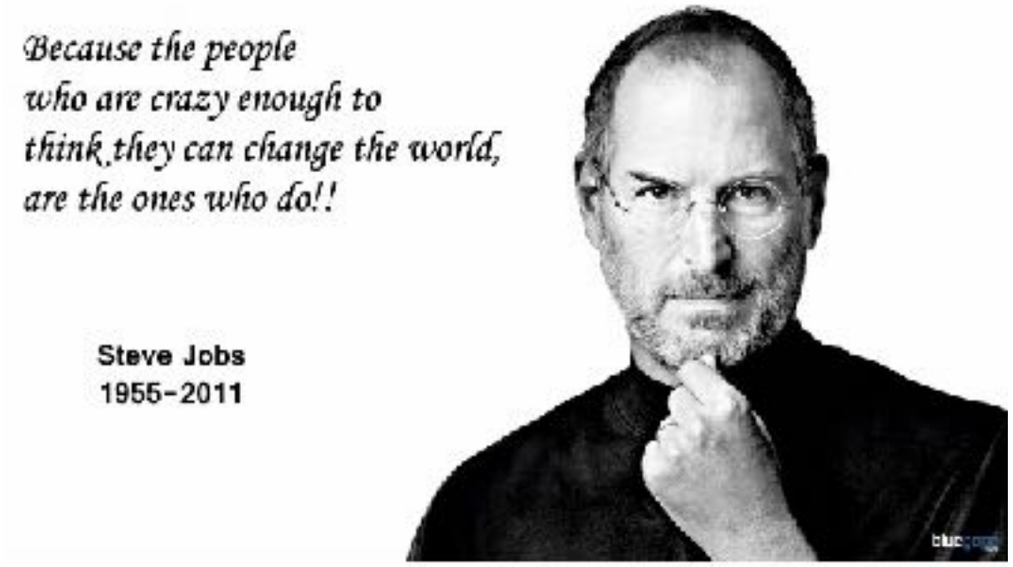
- Radboud: IRMA, Sovereign, PEP

On Premise OU Security course for the Dutch Tax and Customs Administration

- Security and IT, Software Security

Motivation for Resource Consumption Analysis

- IT is changing the world!



- but maybe it is about time that IT starts saving the world….?

History of resource analysis research track

- **funding**: NL-AHA, EU-CHARTER, EU-TACLe
- Size analysis for functional programs
  - [TLCA'07, TFP'07, TFP'08, IFL'08, TFP'12, FOPARA'13]
- Polynomial interpolation
  - [WFLP'07, ENTCS'08, LMCS'09, FOPARA'11, SymbComp'14, ……..]
- Loop bound and memory analysis
  - [PPPJ'10, JTRES'12, ConcComp'14]
- Energy analysis
  - [FOPARA'13, FOAL '14, FOPARA'15 , FOPARA'17, PASS'18, ……..]

Focus now on *systems with multiple components*

- model the system components
- analyse the ***implementation*** of the control software together with the component models

Blind spot for software energy consumption

So, how can we contribute to reduce this energy consumption?
……ehhh: how much energy does IT actually cost?

- a Google search
- your personal cloud activity
- big data calculations
- bitcoin mining (Vranken 2017)
- websites
- keeping all servers online 24/7; running your router through the night

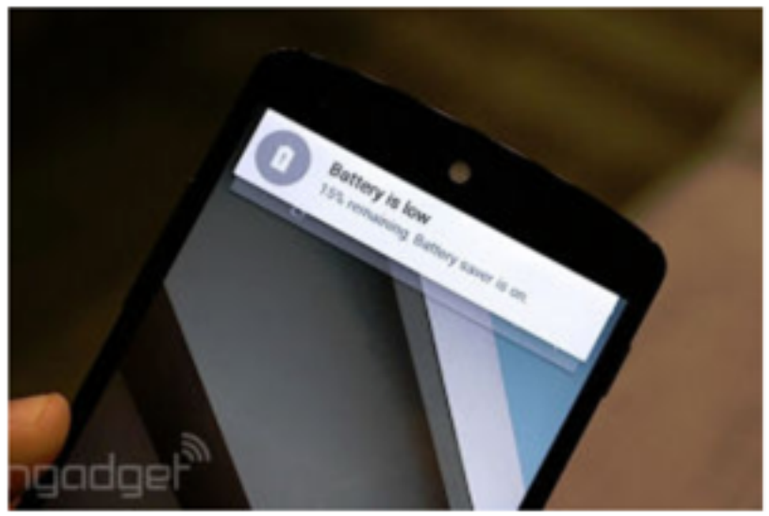
What can **you** do?

- in **programming class** discuss performance including energy
- use FM for **research** in **sustainability**

source: engadget.com

Software can make a big difference in energy consumption

Android O teases big changes to save your battery



07.03.14 AndroidL

Early Android L tests show serious battery life improvement

One of the big reveals for Android 4.4 KitKat's successor, Android L, was Project Volta -- new twe...

By Steve

Android Lollipop (5) has a standby time of 200 hours
Android Marshmallow (6) has a standby time of 533 hours
Both on a Nexus 5. By optimising at the software level.

Our focus: potential savings on the software control level.

Number of methods that are hardware specific, and yield nice results.

Software is in control of (external) hardware, but there is no general way of analysing energy consumption of hardware controlled by software.

Therefore, we focus on **external devices, controlled by software**.

It is not the ***Design*** but it is the ***Implementation*** that is in control.

Volkswagen

Panasonic

Our focus

During evaluation, several algorithms need to be measured for efficiency, but…

- measuring (physically) is hard!
- measuring (all models of a device) is unrealistic!

Therefore, we focus on a **static method** to **predict** energy consumption before execution, for multiple devices at once. This way, a programmer can use these analysis methods **during development**.

Our analysis should be **hardware-parametric**, to easily switch hardware models.

Our analysis method should be quick, i.e. **modular**, to easily analyse realistic/large programs.

derived energy semantics using dependent types

$$\frac{}{\Delta^s \vdash \langle c, \sigma, \Gamma\rangle \xrightarrow{e} \langle \mathcal{Z}(c), \sigma, \Gamma\rangle}\ \text{(sConst)} \qquad \frac{}{\Delta^s \vdash \langle x, \sigma, \Gamma\rangle \xrightarrow{e} \langle \sigma(x), \sigma, \Gamma\rangle}\ \text{(sVar)}$$

$$\frac{\Delta^s \vdash \langle e_1, \sigma, \Gamma\rangle \xrightarrow{e} \langle n, \sigma', \Gamma'\rangle \qquad \Delta^s \vdash \langle e_2, \sigma', \Gamma'\rangle \xrightarrow{e} \langle m, \sigma'', \Gamma''\rangle}{\Delta^s \vdash \langle e_1 \boxdot e_2, \sigma, \Gamma\rangle \xrightarrow{e} \langle n \,\overline{\boxdot}\, m, \sigma'', \Gamma''\rangle}\ \text{(sBinOp)}$$

$$\frac{}{\Delta^v \vdash c : \langle\, \text{const}_{\mathcal{N}(c)}\,,\ \text{id}\,\rangle}\ \text{(btConst)} \qquad \frac{}{\Delta^v \vdash x : \langle\, \text{lookup}_x\,,\ \text{id}\,\rangle}\ \text{(btVar)}$$

$$\frac{\Delta^v \vdash e_1 : \langle V_1, \Sigma_1\rangle \qquad \Delta^v \vdash e_2 : \langle V_2, \Sigma_2\rangle}{\Delta^v \vdash e_1 \boxdot e_2 : \langle V_1 \,\overline{\boxdot}\, (\Sigma_1 \ggg V_2), \Sigma_1 \ggg \Sigma_2\rangle}\ \text{(btBinOp)}$$

$$\frac{\Delta^v \vdash e : \langle V, \Sigma\rangle}{\Delta^v \vdash x := e : \langle V, \Sigma \ggg \text{assign}_x(V)\rangle}\ \text{(btAssign)}$$

$$\frac{\Delta^v \vdash e : \langle V_{ex}, \Sigma_{ex}\rangle}{\Delta^v,\ C.f = (x_f, V_f, \Sigma_f) \vdash C.f(e) : \langle [x_f \mapsto V_{ex}, \Sigma_{ex}] \ggg V_f,\ \text{split}(\Sigma_{ex}, [x_f \mapsto V_{ex}, \Sigma_{ex}] \ggg \Sigma_f)\rangle}\ \text{(btCmp)}$$

$$\frac{\Delta^v, f = (x_f, V_f, \Sigma_f) \vdash e : \langle V_{ex}, \Sigma_{ex}\rangle}{\Delta^v,\ f = (x_f, V_f, \Sigma_f) \vdash f(e) : \langle [x_f \mapsto V_{ex}, \Sigma_{ex}] \ggg \text{subst}(V_f, \text{rec}_f(f)),\ \text{split}(\Sigma_{ex}, [x_f \mapsto V_{ex}, \Sigma_{ex}] \ggg \text{subst}(\Sigma_f, \text{rec}_f(f)))\rangle}\ \text{(btCall)}$$

$\Delta^s, f =$ …

Δ^v …

⟨*fun-def*⟩ ::= …

⟨*bin-op*⟩ ::= …

⟨*expr*⟩ ::= …

⟨*stmt*⟩ ::= …

listin…

```
1 SoundSystem.on();
2 repeat #n begin
3   SoundSystem.playBeepAtHz(#hz);
4   System.sleep()
5 end;
6 SoundSystem.off()
```

listing 6.6 Example program.

```
1 repeat #n begin
2   SoundSystem.on();
3   SoundSystem.playBeepAtHz(#hz);
4   SoundSystem.off();
5   System.sleep()
6 end
```

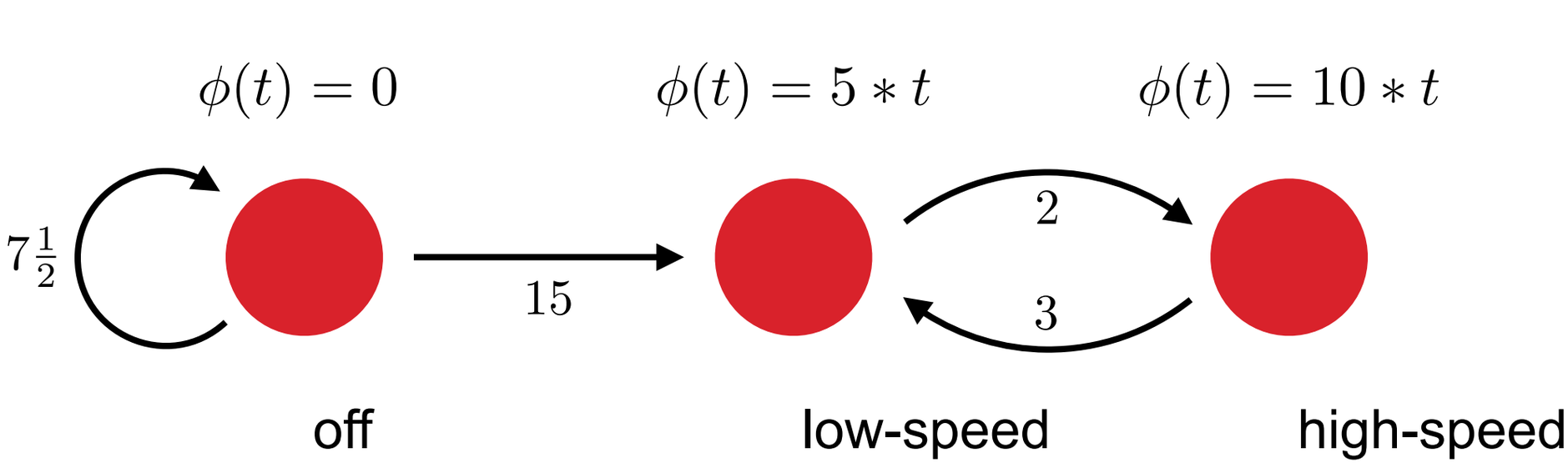
listing 6.7 Alternative program.

Implementation analysis linked to hardware models - automata

Time aware finite state models with energy consumption labels

For **time-dependent** energy usage:
each **state** has a power draw function ϕ

For **incidental** energy usage:
each **transition** has an energy consumption



Methods

We have two methods, which has a trade-off between precision and generality, and analysis time:

- **over-approximating analysis method**
  which reasons over **all possible execution paths** in a **sound** way,
  using a least-upper-bound of states,
  which has a prototype available;
- **precise analysis method**
  which reasons over **one execution path**,
  using a (really fast) simulation of code,
  for which a prototype is under development.

The ECALOGIC tool: http://ecalogic.cs.ru.nl/ecalogic-webapp/

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
```

Prototype of energy consumption analysis of software controlled systems

- Energy semantics **parametrised** by models of controlled devices
- **Prototype** analysis tool, work in progress
  - Written in C++, targeting the language Lua..
  - The prototype can analyse loops, recursive functions and data types.
- **Lua**
  - Lua was developed in 1993 at the University of Rio de Janeiro, Brazil.
  - Lua is a high-level imperative language used to extend other programs and implement higher level features. One example is Adobe Lightroom which consists of 40% Lua. Many games can be scripted using Lua.
  - Lua also has seen interest from the embedded devices community, as the runtime is very light (memory requirements expressed in kilobytes).
  - Advantage: easier to analyse more complex control programs, yet still able to run them directly on embedded systems.

Conclusion Energy Consumption

The ECA research is a new take on analysis of energy usage **in a general way**.

Our focus is on **control software** for **“external *hardware*”**.

‘Hybrid’ time aware finite state models are used to model the energy behaviour of hardware. The analysis is **hardware parametric**.

Many opportunities for real world contributions,

but also for future work

- first step: develop Lua prototype
- second step real world case study
  - hue lighting control
  - ……

Conclusion Formal Methods

- Formal Method research can be very practical in safety critical systems **helping to save the world from disasters**
  - Sovereign project
    - Rijkswaterstaat

- **Resource Consumption Analysis** can contribute to improving **Sustainability** **by saving resources**
  - Worst case execution time analysis
  - Memory consumption analysis
  - Energy consumption analysis